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

SIMULATION–BASED BENDERS CUTS: A NEW CUTTING APPROACH TO APPROXIMATELY SOLVE SIMULATION–OPTIMIZATION PROBLEMS

Mengyi Zhang
Andrea Matta

Dept. of Mechanical Engineering
Politecnico di Milano
Via La Masa 1
Milan, 20156, ITALY

Arianna Alfieri

Dept. of Management and Production Engineering
Politecnico di Torino
Corso Duca degli Abruzzi 24
Turin, 10129, ITALY

Giulia Pedrielli

Dept. of Computing, Informatics, and Decision Systems Engineering
Arizona State University
699 S Mill Avenue
Tempe, AZ 85281, USA

ABSTRACT

Large solution space is one of the main features of simulation–optimization problems. Reducing the cardinality of the set of alternatives is a key point for increasing the efficiency of simulation–optimization methods. In this work, a new cutting approach is proposed for this purpose. The approach exploits the Benders Decomposition framework that can be effectively applied when the simulation–optimization problems are represented using Discrete Event Optimization models. Benders Decomposition subproblems represent the simulation components, hence, cuts can be easily generated observing the values of the variables while a system alternative is simulated, without solving any subproblem. The cut generation procedure is proposed to approximately solve the Server Allocation Problem in a tandem queueing system. Results on randomly generated instances show its effectiveness in decreasing the computational effort by reducing the solution space.

1 INTRODUCTION

Discrete Event Simulation (DES) has established itself as the main tool for the evaluation of the performance of complex stochastic systems in a plethora of applications (Fronckowiak et al. 1996; Tako and Robinson 2012; Günal and Pidd 2010). As a result of this success, optimization and control have increasingly included the use of DES as part of their procedures.

Considering the area of research that uses simulation as a means to evaluate the function to optimize, simulation–optimization (Fu et al. 2005) has witnessed an important development in the last decade. Several families of techniques have been adapted from non–linear optimization field in order to include Monte Carlo based techniques that account for the noise of simulation replications. Random search approaches (Andradóttir 2006), surrogate model based algorithms (Jones et al. 1998; Osorio and Bierlaire 2013), and partitioning driven procedures (Shi et al. 2000; Hong and Nelson 2006) have been deeply explored, reaching a substantial improvement in terms of algorithmic efficiency.

All of the aforementioned approaches share a common characteristic: they consider simulation as a pure black box and very little is used of the simulation model structure. Most of the methods consider the relationship between input and output and use it to inform the search procedure, indeed. Instead, perturbation analysis approaches use events from simulation to calculate first order derivatives of performance measures. Perturbation analysis has been successfully applied in pair with gradient based methods (Ho and Cao 2012), but it is limited by the structural assumptions that the discrete event system must satisfy. Another family of approaches considering the information about the simulation events, called Discrete Event Optimization (DEO), has been recently proposed by the authors (Pedrielli et al. 2015a; Zhang et al. 2016; Pedrielli et al. 2018). The basic idea behind DEO is to *fully* integrate the simulation and the optimization in a unique model, using mathematical programming.

The DEO approach shows how DES can be used not only for performance evaluation of the modeled stochastic system but also to define the feasible region of the optimization problem. However, several challenges need to be solved in order to make DEO a usable tool, especially when solving large scale optimization problems. (1) One major challenge is the improvement of the computational efficiency (DEO models are large and complex due to the fact that events appear as decision variables, and that the variables are sometimes binary); (2) DEO approaches are single run; as a result, efficient methods to adaptively set the simulation run length are necessary, and such an extension is not trivial.

While both challenges are particularly important, this paper focuses on the first problem and explores the use of Benders Decomposition (BD) to effectively cut the solution space thus reducing the size of the original DEO model. This decomposition approach is not new in simulation–optimization. In fact, the Buffer Allocation Problem (BAP) of open flow lines was solved in Weiss and Stolletz (2015) using a BD approach. The original aspect of our paper is that the cuts are generated exploiting the information included in the simulation sample path instead of solving mathematical programming problems as standard BD approaches require. The fact that cuts can be easily generated observing the values of the variables while a system alternative is simulated is an important advantage. Another advantage is that, in the cases where the simulation components include binary variables, simulation sample path enables to generate approximate cuts without much loss in the effectiveness. Further, the cut is independent from the method used to select the alternative to simulate, and this makes possible to combine the proposed cut generation approach to other simulation–optimization approaches as random searches or partitioning procedures. This last advantage is shared with the standard BD approach proposed in Weiss and Stolletz (2015).

In order to showcase the novel approach, we focus on a specific problem, i.e., the Server Allocation Problem (SAP). The case is simple yet meaningful enough to allow the understanding of how the procedure works in details and to highlight which elements are tailored to the specific discrete event dynamics of the system and which elements can be generalized to different systems.

The remainder of the paper is structured as follows. Section 2 motivates the proposed approach. The problem and the cutting methodology is discussed in section 3. Section 4 shows the results of numerical experiments on randomly generated instances of the SAP. Section 5 concludes the paper.

2 BACKGROUND & MOTIVATION

DEO was pioneered in Matta (2008), and fully proposed in Pedrielli et al. (2018). Its applications ranged from the BAP to inventory control (Pedrielli et al. 2015b). In Tan (2015), a similar approach is used for the optimization of continuous flow manufacturing systems. Weiss and Stolletz (2015) proposed a BD based approach to solve the BAP.

All the aforementioned approaches were tailored to the specific application at hand. A first generalization of DEO was discussed in Pedrielli (2013), Matta et al. (2014), Pedrielli et al. (2015a) that also investigated the *time buffer* as a general approximation mechanism to allow for the efficient solution of the problem. While this approximation goes in the direction of improving the solution efficiency, deriving rigorous bounds on the performance is not trivial, especially when the complexity of the system dynamics increases.

In this paper, a decomposition approach is proposed to solve the DEO problems with complex system dynamics, with the aim to explore general guidelines and novel theory. The approach is based on BD as proposed in Weiss and Stolletz (2015) and Zhang et al. (2017). In those works, as simulation components are LP models that can be included in the BD subproblem, cut generation faces no computational difficulty. Instead, the cut generation procedure herein proposed, exploiting simulation, can be used when the simulation component contains also integer/binary variables. The use of simulation for cut generation allows to avoid the use of standard operations research optimization techniques. This ability to eliminate the need for optimization is important, as it allows to increase the efficiency, while maintaining good accuracies.

3 PROBLEM AND METHODOLOGY

In this section, the problem is defined, and a simulation–based cutting approach is proposed for the solution of the two-stage SAP problem modeled according to the DEO framework. The methodology is based on the standard BD with two main differences: (1) cuts are generated using simulation; (2) an approximation is introduced in the cuts and in the master problem to handle non-linear constraints.

Section 3.1 introduces the problem. Section 3.2 presents the complete DEO model of the two–stage SAP, together with the relevant notation. Standard BD master and subproblem, and cut generation are presented in section 3.3.1 and 3.3.2, respectively. The approximate cuts and master problem are discussed in section 3.3.3.

3.1 The Server Allocation Problem (SAP)

The SAP in a two–stage system consists in choosing the number of parallel identical servers to be allocated to each stage to guarantee a maximum average target *system time* (time between the arrival of the customer to the system and its departure) for all the customers while minimizing the total server cost. An example of such a system is depicted in Figure 1. Each stage is composed by several identical servers, with m_1 and m_2 representing the number of servers in the first and second stage, respectively. The servers of a stage share the same queue, and waiting jobs will be processed by the first available server. The first queue has infinite capacity, while the inter–stage queue has a finite known capacity B. Jobs arrive with a general arrival process. Jobs leave the system immediately upon completion at the second stage.

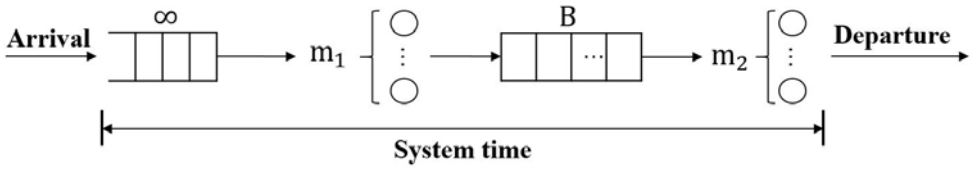


Figure 1: Two–stage parallel server queueing system.

3.2 SAP Formulation

The DEO model for the two–stage SAP consists in an optimization component and a simulation component. In the optimization component, the decision variables define the system configuration, i.e., the number of servers allocated to each stage while constraints bound the maximum acceptable numbers of servers. The simulation component has decision variables that model the occurrence times of the events and constraints defining the system dynamics. Also, constraints exist linking the optimization and the simulation component.

Notations and parameters

N	number of customers in simulation	l	iteration index
B	capacity of the inter-stage buffer	$t_{i,s}$	service time of the i–th customer at stage s
L_{Ms}	lowerbound of server number of stage s	e_i^a	arrival time of the i–th customer

i	arrival sequence index ($i = 1,...,N$)	s	stage index ($s = 1,2$)
j	departure sequence index ($j = 1,...,N$)	T^*	target system time
r	server index ($r = 1,...,U_M$)	c_s	cost of a single server of stage s
U_M	upperbound of server number of each stage		

Optimization variables

$m_s \in \mathbb{Z}^+$	number of servers of stage s
$y_{s,r} \in \{0,1\}$	server allocation variable; $y_{s,r} = 1$ if at least r servers are allocated to stage s, $y_{s,r} = 0$ otherwise

Simulation variables

$e^s_{i,s} \geq 0$	starting time of $i-$th customer at stage s, which is also the i–th arrived customer
$e^d_{j,s} \geq 0$	departure time of $j-$th customer at stage s
$\delta_{ijs} \in \{0,1\}$	$\delta_{ijs} = 1$ if the i–th arrival is the j–th departure at stage s; $\delta_{ijs} = 0$ otherwise

Original DEO model

$$\min\{c_1 m_1 + c_2 m_2\} \tag{1}$$

s.t.

$$m_s \leq r - 1 + M y_{s,r} \qquad r = 1,...,U_m,\ s = 1,2 \tag{2}$$

$$m_s \geq \sum_{r=1}^{U_m} y_{s,r} \qquad s = 1,2 \tag{3}$$

$$e^s_{i,1} \geq e^a_i \qquad i = 1,...,N \tag{4}$$

$$e^s_{i,s} - e^d_{i-r,s} \geq D(r y_{s,r} - m_s) \qquad r = 1,...,U_m,\ s = 1,2,\ i = r+1,...,N \tag{5}$$

$$e^d_{j,s} - e^s_{i,s} \geq M(\delta_{ijs} - 1) + t_{i,s} \qquad s = 1,2,\ i,j = 1,...,N \tag{6}$$

$$e^s_{j,2} - e^d_{j,1} \geq 0 \qquad j = 1,...,N \tag{7}$$

$$e^d_{j,1} - e^s_{j-B,2} \geq 0 \qquad j = B+1,...,N \tag{8}$$

$$\delta_{ijs} = \Delta_{ijs}(m_1, m_2) \qquad s = 1,2, i,j = 1,...,N \tag{9}$$

$$\frac{\sum_{j=1}^N e^d_{j,2} - \sum_{i=1}^N e^a_i}{N} \leq T^* \tag{10}$$

$$L_{Ms} \leq m_s \leq U_M,\ s = 1,2$$

$$m_s \in \mathbb{Z}^+, \delta_{ijs}, y_{s,r} \in \{0,1\}, e^s_{i,s}, e^d_{j,s} \geq 0 \qquad s = 1,2; i,j = 1,...,N; r = 1,...,U_m$$

Equation (1) is the objective function, i.e., the minimization of the overall server cost. As servers in a given stage are identical, they have the same cost, while the cost of the servers in the two stages can be different. Constraints (2) and (3) link the number of servers in each stage to the binary variables used for the server number selection. In fact, from the definition of $y_{s,r}$, it can be easily seen that $y_{s,1} = y_{s,2} = ... = y_{s,m_s} = 1$, $y_{s,m_s+1} = y_{s,m_s+2} = ... = y_{s,U_M} = 0$. Constraints (4) to (10) deal with the simulation trajectory. Constraints (4) state that the service of a customer can start only after it arrives. Constraints (5) show that the service of a customer can start only if there is one server available. With parallel servers, the departure sequence and the arrival sequence may differ from each other. Thus, constraints (6) link the i–th arrived customer to the j–th leaving the stage s. Constraints (7) allow the service at the second stage to start only after the customer leaves the first stage. Since the inter–stage buffer has finite capacity, constraints (8) represent the fact that a customer can leave the first stage only if there is available space in the queue. Constraints (9) link the value of variables δ_{ijs} to the value of m_1 and m_2: once the m_1 and m_2 are known, the system can be simulated, and if the i–th arriving customer is the j–th leaving stage s, $\Delta_{ij1}(m_1, m_2)$ is set to be 1, otherwise $\Delta_{ijs}(m_1, m_2)$ is set to be 0. Constraint (10) is the performance constraint, which bounds the average system time of all the customers to be smaller than or equal to the target value T^*.

The number of binary variables in the model is equal to $2N^2 + 2U_M$, most of which are from the simulation component. Furthermore, constraints (5) and (6) are big–M constraints, where D and M are both large numbers. Moreover, constraints (9) cannot be written in an explicit and linear way, as operator $\Delta_{ij1}(m_1, m_2)$ is the result of the discrete event simulation.

3.3 Model Decomposition and Cuts Generation Procedure

The model in section 3.2 cannot be efficiently solved using state-of-the-art methods, mainly due to the large number of binary variables and to constraints (9) that are not linear and explicit. Thus, an alternative solution approach is proposed in section 3.4. The proposed solution procedure is based on the BD approach, as shown in section 3.3.1. The cuts are generated and simplified based on the simulation trajectory, as shown in section 3.3.2. Section 3.3.3 shows how the cuts can be further approximated, and the master problem becomes a solvable mixed integer linear programming, whose complexity has been significantly reduced.

3.3.1 Decomposition

According to BD, the original problem is decomposed into a master problem and a subproblem. The subproblem must be a linear program, as the standard Benders cuts are generated from the subproblem based on the strong duality theory.

Master problem

$$\begin{aligned} &\min\{c_1 m_1 + c_2 m_2\} \\ &\text{s.t.} \\ &(2), (3), (9), \text{generated cuts} \end{aligned}$$

The master problem variables are $m_s, y_{s,r}, \delta_{ijs}$ and the constraints are the ones that only contain those variables.

Subproblem

$$\begin{aligned} &\min\{\varepsilon\} && && \\ &\text{s.t.} && && \\ &e^s_{i,1} \geq e^a_i && : a_{i,1} \quad && i = 1,...,N && (11) \\ &e^s_{i,s} - e^d_{i-\bar{m}_s,s} \geq 0 && : u_{i,s} \quad && i = \bar{m}_s + 1,...,N && (12) \\ &e^s_{i,2} - e^d_{i,1} \geq 0 && : a_{i,2} \quad && j = 1,...,N && (13) \\ &e^d_{j,s} - e^s_{i,s} \geq t_{i,s} && : v_{ijs} \quad && \bar{\delta}_{ijs} = 1 && (14) \\ &e^d_{j,1} - e^s_{j-B,2} \geq 0 && : w_j \quad && j = B+1,...,N && (15) \\ &\frac{\sum_{j=1}^N e^d_{j,2} - \sum_{i=1}^N e^a_i}{N} - \varepsilon \leq T^* && : \theta && && (16) \end{aligned}$$

In equations (11)–(16), the symbols behind the constraint formulation ($a_{i,s}, u_{i,s}, v_{ijs}, w_j, \theta$) represent the dual variables associated to the constraints. Instead, $\bar{m}_s$ and $\bar{\delta}_{ijs}$ are the value of the master problem variables from the optimal solution of the master problem.

It can be noticed that the decision variables in the subproblems are the event occurrence times, i.e., $e^s_{i,s}$ and $e^d_{j,s}$, and the constraints are the system dynamics constraints and the big-M constraints (i.e., (5) and (6)) linking optimization and simulation variables. However, once the solution of the master problem is known, many of such constraints can be eliminated. Moreover, since the objective function of the original

problem does not contain any subproblem variables, the subproblem is a so–called *feasibility problem*. However, as the mathematical programming model of the subproblem is a representation of the simulation trajectory, the discrete event times from simulation are feasible for constraints (11)–(15). Consequently, only one feasibility variable, ε, has to be introduced to reach feasibility for the performance constraint (16).

3.3.2 Classical Benders Cuts Generation

Knowing the optimal value of the dual variables $\bar{a}_{i,s}, \bar{u}_{i,s}, \bar{v}_{ijs}, \bar{w}_j, \bar{\theta}$, the classic Benders feasibility cut is the following:

$$\sum_{i=1}^{N} e_i^a \bar{a}_{i,1} + \sum_{s=1}^{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \bar{v}_{ijs}(M(\delta_{ijs}-1)+t_{i,s}) + \sum_{s=1}^{2}\sum_{i=1}^{N} \bar{u}_{i,s} D(\bar{m}_s y_{s,\bar{m}_s} - m_s) - \bar{\theta}(T^* + \frac{\sum_{i=1}^{N} e_i^a}{N}) \leq 0. \tag{17}$$

The dual of the feasibility subproblem is a network flow problem, and its optimal solution can be derived from the event relationship graph after the system configuration has been simulated (Chan and Schruben 2008), namely:

$$\bar{\varepsilon} = \frac{\sum_{j=1}^{N} \bar{e}_{j,2}^d - \sum_{i=1}^{N} e_i^a}{N} - T^*. \tag{18}$$

From the strong duality theory, the optimal solution of the subproblem and the dual subproblem should be equal, i.e.,

$$\sum_{i=1}^{N} e_i^a \bar{a}_{i,1} + \sum_{s=1}^{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \bar{v}_{ijs} t_{i,s} - \bar{\theta}(T^* + \frac{\sum_{i=1}^{N} e_i^a}{N}) = \bar{\varepsilon} = \frac{\sum_{j=1}^{N} \bar{e}_{j,2}^d - \sum_{i=1}^{N} e_i^a}{N} - T^*. \tag{19}$$

Hence, given (18) and (19), the feasibility cut (17) can be rewritten as:

$$\sum_{s=1}^{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \bar{v}_{ijs} M(\delta_{ijs}-1) + \sum_{s=1}^{2}\sum_{i=1}^{N} \bar{u}_{i,s} D(\bar{m}_s y_{s,\bar{m}_s} - m_s) + \bar{\varepsilon} \leq 0. \tag{20}$$

3.3.3 Improved Benders Cuts

Even though the original problem can be decomposed following the classic BD procedure, as mentioned in section 3.3.1, the master problem is not trivial to solve. The two obstacles are: 1) constraints (9) are not linear; 2) the feasibility cuts have very low efficiency because of M and D from big–M constraints. To overcome such difficulties, we propose a solvable approximate master problem, where (21) is approximated from (20).

Approximate master problem

$$\begin{aligned} &\min\{c_1 m_1 + c_2 m_2\} \\ &\text{s.t.} \\ &(2), (3) \\ &\sum_{s=1}^{2}\sum_{i=1}^{N} \bar{u}_{i,s}^l d_{i,s}^l (\bar{m}_s^l y_{s,\bar{m}_s^l} - m_s) + \bar{\varepsilon}^l \leq 0 \end{aligned} \tag{21}$$

At each iteration l, the approximate master problem is considered to (a) fix the values of δ_{ijs} to $\bar{\delta}_{ijs}^l$; (b) identify a relatively small value for D, called $d_{i,s}^l$ in the following. In both (a) and (b), the information from simulation trajectory is used.

(a) Fixing δ_{ijs}
In the original formulation, the values of δ_{ijs} vary with the different system configurations (m_1, m_2), and are consistent with the simulation trajectory. To keep the consistency with the simulation trajectory, δ_{ijs} are updated and fixed after simulating the system configured by the master problem solution at each iteration, and these values are used for the approximate master problem at the next iteration. If variables δ_{ijs} are fixed to the values $\bar{\delta}^l_{ijs}$, constraints (9) can be removed, and $\sum_{s=1}^{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\bar{v}^l_{ijs}M(\delta^l_{ijs}-1)$ is removed from the cut because $\bar{v}^l_{ijs}(\bar{\delta}^l_{ijs}-1)=0$. Since δ_{ijs} are fixed at each iteration, they are no more master problem variables. Differently from the decomposition in section 3.3.1, the optimization variables are in the approximate master problem, the continuous simulation variables in the subproblem for cut generation, and the discrete simulation variables are in neither problems.

(b) Identifying small values $d_{i,s}$ for the big–M constraints
In the original formulation, D is used to inactivate constraints (5) with $m_s \geq r+1$, i.e.,

$$D \geq \frac{e^d_{i-r,s}-e^s_{i,s}}{m_s-r}, \forall i, r, \forall m_s \geq r+1.$$

When $m_s = r$, constraints (5) are active, and $e^s_{i,s} \geq e^d_{i-m_s,s}$. Thus, $\frac{e^d_{i-r,s}-e^d_{i-m_s,s}}{m_s-r} \geq \frac{e^d_{i-r,s}-e^s_{i,s}}{m_s-r}$, and the maximal inter–departure time can be a large enough value for D to inactivate the constraints.

In the approximate formulation, we can give different values to D for different customers i at different stages s, i.e., $d_{i,s}$ instead of a single value of D, and a possible value of $d_{i,s}$ is

$$d^l_{i,s} = \min\left\{\bar{e}^{dl}_{i-\bar{m}^l_s+1,s} - \bar{e}^{dl}_{i-\bar{m}^l_s,s}, \frac{1}{\lambda}\right\},$$

where the quantity $\bar{e}^{dl}_{i-\bar{m}^l_s+1,s} - \bar{e}^{dl}_{i-\bar{m}^l_s,s}$ is the inter–departure time in simulation, and $\frac{1}{\lambda}$ is equal to the average inter–arrival time in long term, which is equal to the average inter–departure time in a stable system. This setting assures the efficiency of the cut, but it also introduces an approximation.

Fixing δ_{ijs} and identifying the $d^l_{i,s}$ values lead to an approximate master problem and to approximate cuts. However: (a) it is essential to construct a solvable master problem. Moreover, this technique can be used in other systems, whose simulation components contain binary variables. This is one of the original contributions of this work, because optimization problems of such kind of systems have not been addressed in the literature. Instead, (b) the approximation of the $d_{i,s}$ is optional. To make an informed choice, it should be noticed that there is a trade-off: small values of $d_{i,s}$ increase the efficiency, but increase the risk of cutting the optimum from feasible region as well. On the contrary, if $d_{i,s}$ is set to be large, higher computational burden will be required but quality of the solution will be improved.

Although both (a) and (b) introduce approximation, the numerical results show that the optimal solution can be found in most of the cases, and, when a difference between the found solution and the optimum exists, it is quite small, as discussed in section 4.

3.4 Solution Approach

The complete solution approach for the two–stage SAP is shown in Figure 2. At the beginning, an initial value for the server number at both stages is given. These initial values are the lower bounds and may correspond to an infeasible system.

Once the number of servers has been fixed, the system is simulated to get the value of the subproblem variables and derive the feasibility cut as described in the previous sections. The cut is then added to the master problem that is solved to find the new best configuration (given the added cuts).

At each iteration, the master problem, solved using mathematical programming, updates the lower bound of the solution. The new configuration is simulated to find the values of the subproblem variables and so on until the target performance is achieved.

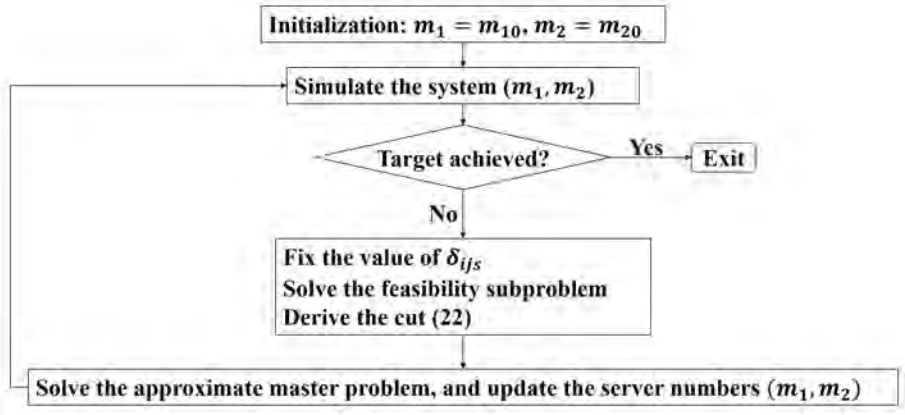


Figure 2: Solution approach.

4 EMPIRICAL ANALYSIS

In this section, the proposed solution approach is applied to randomly generated instances to evaluate its performance in terms of: solution effectiveness, cut efficiency and cut convergence. The inter–arrival time follows an exponential distribution, with rate λ equal to 1 time unit. The unit cost c_1 and c_2 are assumed equal to 1 (which means to minimize the total number of servers in the system). The upper bound of the server number U_M is equal to 70. The values of the other parameters have been changed depending on the aim of the specific experiments (i.e., effectiveness, efficiency and convergence).

4.1 Solution Effectiveness

To test the solution effectiveness, the experiment has been conducted using different values for the buffer space B, the target system time T^*, and the ratio between the mean service time and the mean inter–arrival time at each stage s, α_s, and the distributions of the service time at both stages (exponential and beta(2,2) scaled on $(0, 2\alpha_s)$). The values for each of these parameters are reported in the left part of Table 1. For each parameter combination, 100 different replications are solved by generating 100 different sample paths, each with $N = 200000$ and $N = 20000$. As stated in section 3.2, the number of binary variables will be over $2N^2$, i.e., 4×10^{10} and 4×10^8, respectively. The initial values of m_1 and m_2 have been chosen as the minimum number of servers in a stable system.

The results, reported in the right part of Table 1, are the percentage of cases (P_{opt}) in which the solution found is equal to the optimal solution (found by enumeration using the same sample path described below), the maximum gap between the solution and the optimum, the average number of iterations (I) to find the solution, the average number of iterations (I_{enum}) to reach the optimum from the lower bound using enumeration. The high values of P_{opt} and the maximum gap of 1 show the effectiveness of the proposed approach. Moreover, it can be notice that the efficiency of the approach is higher than using the enumeration approach, with 80.4% fewer iterations on average.

As discussed in section 3.3.3, the efficiency and the effectiveness of the solution are affected by the value of D. For the system with $T^* = 61$, $B = 30$, $(\alpha_1, \alpha_2) = (10, 50)$, and exponential distributed service times, the value of D is varied from 0.2 to 10, and the impact on the iteration number and P_{opt} is shown in Figure 3. As the value of D increases, the effectiveness is improved, but the efficiency becomes lower. After 20.75 iterations, on average, the optimal solution can be found in 100% of the cases, with $D \geq 4$. The efficiency is much higher than for the enumeration, which reaches the optimum after 86.64 iterations, as shown in Table 1.

Table 1: Parameter values (on the left) and solution effectiveness (N=200000 and 20000) (on the right).

Parameters				N=200000				N=20000			
B	T^*	α_1, α_2	distribution	P_{opt}	max gap	I	I_{enum}	P_{opt}	max gap	I	I_{enum}
5	61	30,30	exp, exp	0.91	1	20.49	104.74	0.83	1	18.35	110.1
5	61	30,30	exp, beta	0.9	1	18.96	92.82	0.83	1	17.02	97.27
5	61	30,30	beta, exp	0.94	1	18.11	88.99	0.88	1	16.32	89.33
5	61	30,30	beta, beta	0.87	1	14.65	75.13	0.88	1	13.44	72.67
30	61	30,30	exp, exp	0.78	1	16.01	105.6	0.81	1	14.66	109.54
5	65	30,30	exp, exp	0.89	1	12.35	32.72	0.95	1	10.41	32.28
5	70	30,30	exp, exp	0.97	1	8.87	16.98	0.96	1	6.82	17.01
5	61	10,50	exp, exp	0.9	1	16.58	86.64	0.81	1	14.38	88.17
5	61	10,50	exp, beta	0.98	1	15.03	73.46	0.91	1	13.08	71.49
5	61	10,50	beta, exp	0.92	1	14.62	77.24	0.86	1	13.19	82.14
5	61	10,50	beta, beta	0.87	1	11.78	64.91	0.95	1	11.14	64.35
30	61	10,50	exp, exp	0.78	1	12.8	87.13	0.86	1	11.65	87.37
5	65	10,50	exp, exp	0.92	1	10.08	25.27	0.97	1	8.09	23.63
5	70	10,50	exp, exp	0.96	1	7.74	11.95	0.94	1	5.68	11.30
5	61	50,10	exp, exp	0.85	1	16.69	88.7	0.94	1	14.61	87.03
5	61	50,10	exp, beta	0.88	1	14.92	80.17	0.93	1	12.71	77.07
5	61	50,10	beta, exp	0.92	1	14.67	71.06	0.87	1	13	71.24
5	61	50,10	beta, beta	0.9	1	11.97	65.51	0.9	1	11.13	63.61
30	61	50,10	exp, exp	0.83	1	13.23	88.56	0.88	1	11.59	87.26
5	65	50,10	exp, exp	0.96	1	10.35	25.62	0.92	1	8.13	23.74
5	70	50,10	exp, exp	0.98	1	8.03	12.75	0.95	1	5.61	11.56

4.2 Cuts Efficiency and Convergence

The cut efficiency can be evaluated in terms of the number of solutions cut from the feasible region with each cut, i.e., the number of master problem solutions that violate the generated cut (21). For each specific case reported in Table 1, this number depends on the simulated system configured with $(\bar{m}_1, \bar{m}_2)$. Figure 4(a) shows the numbers of solutions cut on average (over the 100 sample paths) for the problem with $B = 5, T^* = 61, \alpha_1 = 30, \alpha_2 = 30$, and service time following exponential distribution at each stage.

First of all, as shown in Figure 4(a), many numbers are greater than 1, i.e., the cut can remove more than one solution from the feasible region. This shows that, by exploiting the simulation trajectory, we can cut also systems whose performance has not been evaluated. As this behavior is independent from how the master problem is solved, it shows that not considering simulation simply as a black box can increase the solution efficiency.

Moreover, it can be noticed that cut efficiency is high when unbalanced systems are simulated, and the master problem will avoid allocating more servers to the faster stage and, at the same time, fewer servers to the slower stage. On the contrary, cut efficiency is low when it is getting closer to the feasible region. This is reasonable, i.e., more computational budget should be allocated to the area near the feasible region. The efficiency close to the boundary is also low, but if combinatorial cuts (Codato and Fischetti 2006) based on the knowledge of SAP are introduced, the cut efficiency in these regions could be improved.

Finally, as the number of customers increases, the cuts seem to converge, i.e., the coefficients of equations (21) stabilize. As shown in constraints (21), in fact, there are three parameters: $\sum_{i=1}^{N} \bar{u}_{i,1}^{l} d_{i,1}^{l}$, $\sum_{i=1}^{N} \bar{u}_{i,2}^{l} d_{i,2}^{l}$, and $\bar{\varepsilon}^{l}$. The three parameters can be reduced to two by dividing all of them by $\bar{\varepsilon}^{l}$, i.e., $e_1 = \sum_{i=1}^{N} \bar{u}_{i,1}^{l} d_{i,1}^{l} / \bar{\varepsilon}^{l}$ and $e_2 = \sum_{i=1}^{N} \bar{u}_{i,2}^{l} d_{i,2}^{l} / \bar{\varepsilon}^{l}$. Figure 4(b) shows, for the problem with $\bar{m}_1 = \bar{m}_2 = 31$, $B = 5$, $T^* = 61$, $\alpha_1 = \alpha_2 = 30$ and exponential distributed service time, how the parameters, namely e_1 and

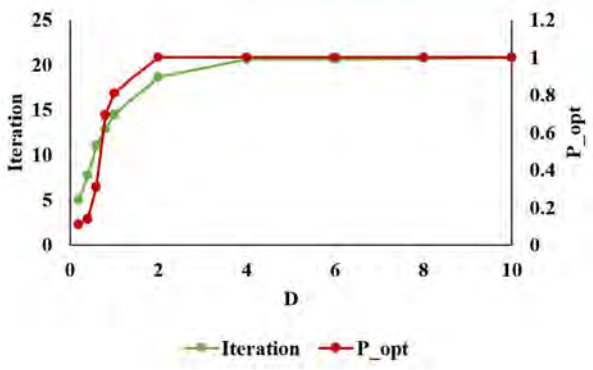
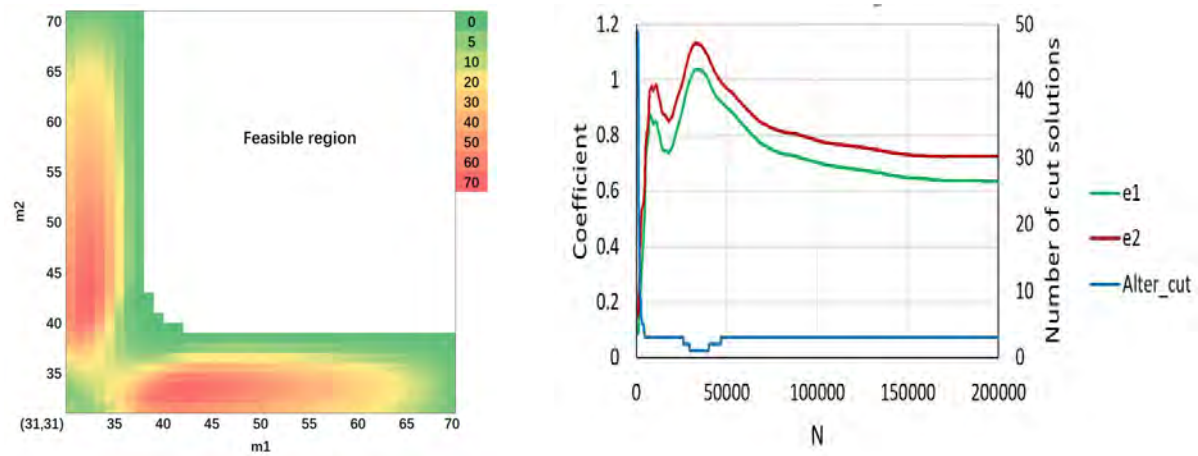


Figure 3: Impact of D on performance.



(a) Average number of cut solutions per simulation execution (from 100 sample paths).

(b) Cut convergence.

Figure 4: Cut efficiency and convergence.

e_2, and the number of cut solutions change as N increases for a single sample path. When N is small, the parameters show a transient pattern, and they converge as N increases. The convergence speed depends on the simulated system, e.g., the service time and the buffer spaces. The number of cut solutions converge to the same value at $N = 50000$. Thus, for an integer programming master problem, the stopping criteria of simulation can be less strict than the cut convergence criteria. This will be a future research interest.

5 CONCLUSIONS

In this paper, a novel simulation-based cutting approach is investigated that increases the efficiency of the DEO framework with respect to the previous approaches embedding Benders Decomposition.

The novelty of the approach lies in the use of simulation to generate and improve cuts. In fact, similar to BD traditional approaches, we separate the problem into a master and a subproblem. This is quite natural using DEO formulation, as it includes an optimization and a simulation component. However, instead of solving, at each iteration, the so-called *subproblem* using standard optimization techniques, whose complexity depends on the type of decision variables (integer or continuous), we simulate the system configuration (defined by the master problem) and extract from it the values of the subproblem

variables needed to define the cuts. Moreover, simulation is also used to fix integer / binary variables that would make the master problem non–linear, thus reducing the complexity of its solution too.

The procedure has been tested on the two-stage Server Allocation Problem (SAP) and it showed to be performing satisfactorily in terms of solution quality, empirical cut convergence, and cut efficiency. Specifically, 1) the achieved solution has been proved to be the optimum in most of the cases, and near optimal in the remaining ones; 2) as the sample path increases, the coefficients of the cut equation converge to stable values in the executed experiment; 3) cuts are able to considerably reduced the number of solutions to evaluate.

Nevertheless, solving problems of increased complexity is necessary in order to establish stronger empirical results meanwhile developing the general theory behind the approach. The multi-stage SAP will be addressed in future work to investigate the combinatorial cuts, the cut efficiency, convergence and solution effectiveness of complex problems. In fact, DEO models share the common characteristics to have well structured event relationship graphs, which resemble network flow problems. As a consequence, we believe the proposed approach can be extended to more general cases. As the convergence of DEO models has been proved in Pedrielli et al. (2018), future research will also investigate cut convergence, with the specific objective to (1) efficiently allocate the computational effort across iterations, (2) embed generated cuts into general partitioning algorithms.

REFERENCES

Andradóttir, S. 2006. "An Overview of Simulation Optimization via Random Search". *Handbooks in Operations Research and Management Science* 13:617–631.

Chan, W. K., and L. Schruben. 2008. "Optimization Models of Discrete-Event System Dynamics". *Operations Research* 56(5):1218–1237.

Codato, G., and M. Fischetti. 2006. "Combinatorial Benders' Cuts for Mixed-Integer Linear Programming". *Operations Research* 54(4):756–766.

Fronckowiak, D., A. Peikert, and K. Nishinohara. 1996. "Using Discrete Event Simulation to Analyze the Impact of Job Priorities on Cycle Time in Semiconductor Manufacturing". In *Proceedings of Advanced Semiconductor Manufacturing Conference and Workshop, 1996.*, 151–155. IEEE/SEMI.

Fu, M. C., F. W. Glover, and J. April. 2005. "Simulation Optimization: a Review, New Developments, and Applications". In *Proceedings of the 2005 Winter Simulation Conference*, edited by M. E. Kuhl et al., 83–95. Piscataway, New Jersey: IEEE.

Günal, M. M., and M. Pidd. 2010. "Discrete Event Simulation for Performance Modelling in Health Care: a Review of the Literature". *Journal of Simulation* 4(1):42–51.

Ho, Y.-C. L., and X.-R. Cao. 2012. *Perturbation Analysis of Discrete Event Dynamic Systems*, Volume 145. Springer Science & Business Media.

Hong, L. J., and B. L. Nelson. 2006. "Discrete Optimization via Simulation Using COMPASS". *Operations Research* 54(1):115–129.

Jones, D. R., M. Schonlau, and W. J. Welch. 1998. "Efficient Global Optimization of Expensive Black-Box Functions". *Journal of Global optimization* 13(4):455–492.

Matta, A. 2008. "Simulation Optimization with Mathematical Programming Representation of Discrete Event Systems". In *Proceedings of the 2008 Winter Simulation Conference*, edited by T. Jefferson et al., 1393–1400. Piscataway, New Jersey: IEEE.

Matta, A., G. Pedrielli, and A. Alfieri. 2014. "Event Relationship Graph Lite: Event Based Modeling for Simulation-Optimization of Control Policies in Discrete Event Systems". In *Proceedings of the 2014 Winter Simulation Conference*, edited by S. J. Buchley et al., 3983–3994. Piscataway, New Jersey: IEEE.

Osorio, C., and M. Bierlaire. 2013. "A Simulation-Based Optimization Framework for Urban Transportation Problems". *Operations Research* 61(6):1333–1345.

Pedrielli, G. 2013. *Discrete Event Systems Simulation-Optimization: Time Buffer Framework*. Ph. D. thesis, Mechanical Engineering Department, Politecnico di Milano, Italy.

Pedrielli, G., A. Matta, and A. Alfieri. 2015a. "Discrete Event Optimization: Single-Run Integrated Simulation-Optimization Using Mathematical Programming". In *Proceedings of the 2015 Winter Simulation Conference*, edited by C. M. Macal et al., 3557–3568. Piscataway, New Jersey: IEEE.

Pedrielli, G., A. Matta, and A. Alfieri. 2015b. "Integrated Simulation-Optimization of Pull Control Systems". *International Journal of Production Research* 53(14):4317–4336.

Pedrielli, G., A. Matta, A. Alfieri, and M. Zhang. 2018. "Design and Control of Manufacturing Systems: a Discrete Event Optimisation Methodology". *International Journal of Production Research* 56(1–2):543–564.

Shi, L. et al. 2000. "Nested Partitions Method for Stochastic Optimization". *Methodology and Computing in Applied probability* 2(3):271–291.

Tako, A. A., and S. Robinson. 2012. "The Application of Discrete Event Simulation and System Dynamics in the Logistics and Supply Chain Context". *Decision Support Systems* 52(4):802–815.

Tan, B. 2015. "Mathematical Programming Representations of the Dynamics of Continuous-Flow Production Systems". *IIE Transactions* 47(2):173–189.

Weiss, S., and R. Stolletz. 2015. "Buffer Allocation in Stochastic Flow Lines via Sample-Based Optimization with Initial Bounds". *OR Spectrum* 37(4):869–902.

Zhang, M., A. Matta, A. Alfieri, and G. Pedrielli. 2017. "A Simulation-Based Benders Cuts Generation for the Joint Workstation, Workload and Buffer Allocation Problem". In *Conference on Automation Science and Engineering, 2017*. Xi'an, China.

Zhang, M., A. Matta, and G. Pedrielli. 2016. "Discrete Event Optimization: Workstation and Buffer Allocation Problem in Manufacturing Flow Lines". In *Proceedings of the 2016 Winter Simulation Conference*, edited by T. M. K. Roeder et al., 2879–2890. Piscataway, New Jersey: IEEE.

AUTHOR BIOGRAPHIES

MENGYI ZHANG is PhD candidate of Department of Mechanical Engineering at Politecnico di Milano, Italy. Her research interests include simulation optimization of manufacturing systems. Her email address is mengyi.zhang@polimi.it.

ANDREA MATTA is Professor at Politecnico di Milano, where he currently teaches integrated manufacturing systems and manufacturing. His research area includes analysis and design of manufacturing and health care systems. He is Editor-in-Chief of Flexible Services and Manufacturing Journal. His email address is andrea.matta@polimi.it.

ARIANNA ALFIERI is Professor at Politecnico di Torino, where she currently teaches production planning and control and system simulation. Her research area includes scheduling, supply chain management and system simulation optimization. Her email address is arianna.alfieri@polito.it.

GIULIA PEDRIELLI Giulia Pedrielli is currently Assistant Professor for the School of Computing Informatics System Design Systems Engineering in Arizona State University. Her research activity in the area of stochastics and simulation with a particular interest in simulation based optimization. Her email address is giulia.pedrielli@asu.edu.