Proceedings of the 2019 Winter Simulation Conference
N. Mustafee, K.-H.G. Bae, S. Lazarova-Molnar, M. Rabe, C. Szabo, P. Haas, and Y.-J. Son, eds.

FEASIBILITY CUT GENERATION BY SIMULATION: SERVER ALLOCATION IN SERIAL–PARALLEL MANUFACTURING SYSTEMS

Mengyi Zhang
Andrea Matta

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

Arianna Alfieri

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

Giulia Pedrielli

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

ABSTRACT

Simulation–optimization problems exhibit substantial inefficiencies when applied to high–dimensional problems. The problem is exacerbated in case where feasibility also needs to be evaluated using simulation. In this work, we propose an approximate iterative approach to identify feasible solutions and quickly find good solutions to the original problem. The approach is based on discrete event optimization (i.e., a mathematical programming representation of the simulation–optimization problems) and Benders decomposition, which is used for cut generation *while* a system alternative is simulated. The procedure is currently tailored for the server allocation problem in the multi-stage serial-parallel manufacturing line constrained to a target system time on a specific sample path. Results on randomly generated instances show its effectiveness in quickly eliminating infeasible solutions, thus decreasing the required computational effort and keeping the optimality gap low.

1 INTRODUCTION

Optimization of complex discrete event systems may involve discrete–event simulation (DES) as a tool for performance evaluation, leading to simulation–optimization problems (Fu et al. 2005). Research on simulation–optimization has witnessed important developments in the last decade. Simulation–optimization techniques can be classified into two categories: black–box optimization and white–box optimization. Distinction between the two categories is whether simulation is treated as an evaluation tool targeting the relationship between its input and output, or the dynamical structure of the problem is exploited to provide more information besides the output itself. Most of the simulation–optimization techniques belong to the black–box category, including random search approaches (Andradóttir 2006), surrogate model–based algorithms (Jones et al. 1998; Osorio and Bierlaire 2013), and partitioning–driven procedures (Shi and Olafsson 2000; Hong and Nelson 2006). Perturbation analysis and discrete event optimization (DEO), instead, belong to the white–box category. Perturbation analysis (Ho and Cao 2012) is based on the computation of output changes induced by decision variable changes and is usually combined with

stochastic approximation (Robbins and Monro 1951). The basic idea behind DEO (Pedrielli et al. 2018) is, instead, to *fully* integrate the simulation and the optimization in a unique mathematical program, by explicitly modeling the event relationships with constraints representing the DES underlying the problem with the fixed sample path. DEO has been applied to various manufacturing problems such as the buffer allocation problem (Weiss and Stolletz 2015) and the bottleneck detection problem (Zhang and Matta 2018). However, previous works mainly dealt with production lines without parallel resources (e.g., single–server serial systems). When the possibility of having parallel resources is taken into account, even though the DEO model can be easily derived (Pedrielli et al. 2018), solving it remains a tough task, since the model is a mixed-integer linear program (MILP) with a huge number of binary variables.

Preliminary work on systems with parallel resources can be found in Zhang et al. (2018). This work extends the previous work on the Server Allocation Problem (SAP) in serial–parallel manufacturing systems to address high–dimension problems, which is a more relevant issue in practice. As in the previous work, the solution approach is based on the Benders decomposition of the DEO model of the problem. First of all, from an implementation point of view, the algorithms must be redesigned to efficiently exploiting the dual information from the simulation dynamics, which represents the primal information. Moreover, scaling up the algorithms from two–stage to multi-stage systems is not trivial due to the non linearity caused by the structure of parallel servers. Empirical analysis shows that the developed algorithms are able to handle a 17–stage system, whose solution region contains more than 10^{17} candidates, within reasonable time. The computation efficiency has been largely improved, due to the fact that a single feasibility cut, generated from a single simulation run, can remove more than one alternative from the feasible region using the information contained in the simulation model. This work establishes that the DEO framework, combined with Benders decomposition, is a promising family of solution methods for simulation–optimization of queuing systems.

The remainder of the paper is organized as follows. In section 2, the SAP in serial–parallel manufacturing systems is defined. The mathematical programming model and its solution approach are addressed in section 3 and section 4, respectively. Section 5 analyzes the performance of the solution algorithm in terms of computational time, number of iterations, and optimality gap. Section 6 concludes the paper.

2 PROBLEM DEFINITION

The SAP in a serial–parallel system consists of choosing the number of parallel servers to be allocated to each stage of a serial system to guarantee a maximum average target *system time* (i.e., the time between the arrival of a job to the system and its departure) over all the jobs while minimizing the total server cost. In manufacturing systems, limiting system time is usually applied when the product quality is sensitive to the time spent in the system due to contamination, or when customers require a certain lead time. Servers at each stage are identical but they may differ from one stage to another.

An example of such a system is depicted in Figure 1. Each stage is composed of several identical servers, with s_j representing the number of servers at stage j. The servers at a stage share the same input queue, and waiting jobs will be processed by the first available server. The input queue of the first stage has infinite capacity, while the inter–stage queue between stage j and stage $j+1$ has a finite known capacity b_j. Jobs arrive to the system following a general arrival process, and leave the system immediately upon completion at the last stage.

3 MATHEMATICAL PROGRAMMING MODEL

A DEO model integrating the optimization aspect of the SAP (i.e., the choice of the number of servers for each stage) and the simulation trajectory (i.e., the dynamics of the system in terms of event times) is presented in this section. In the model, the objective function deals with the minimization of the total server cost, while constraints guarantee the correct dynamic behavior of the system and the achievement of the

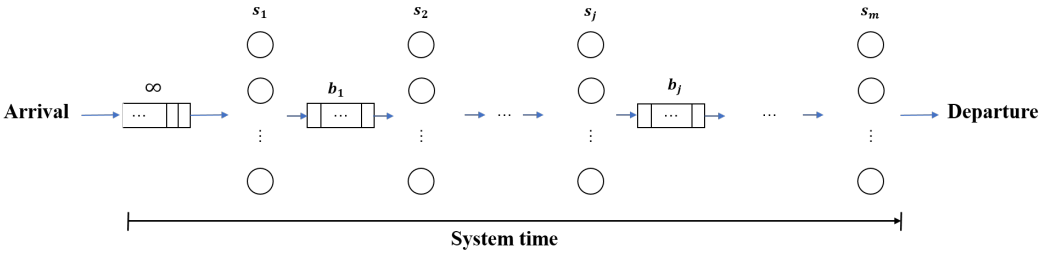


Figure 1: A m–stage parallel server queueing system.

average target system time. In the following, parameters and variables are introduced before presenting the mathematical programming model.

Parameters

$i, i^{'}$	job index ($i = 1, \cdots, N$)
j	stage index ($j = 1, \cdots, m$)
k	server index ($k = 1, \cdots, U_j$)
N	number of jobs
b_j	capacity of the inter-stage queue
L_j	lower bound of number of servers at stage j
U_j	upper bound of number of servers at stage j
r	iteration index
$t_{i,j}$	service time of the i–th job at stage j
e_i^a	arrival time of the i–th job
T^*	target system time
c_j	cost of a single server of stage j
D, M	big-M coefficients in the MILP model

Variables

$s_j \in \mathbb{Z}^+$	number of servers at stage j
$\mathbf{s}$	vector representation of of s_j
$y_{j,k} \in \{0,1\}$	server allocation variable. $y_{j,k} = 1$ if at least k servers are allocated to stage j, $y_{j,k} = 0$ otherwise.
$e_{i,j}^s \geq 0$	starting time of job i at station j
$e_{i,j}^d \geq 0$	departure time of job i at station j
$\delta_{ii^{'}j} \in \{0,1\}$	variable linking the arrival sequence and the departure sequence; $\delta_{ii^{'}j} = 1$ if the i–th arrival is the $i^{'}$–th departure at stage j; $\delta_{ii^{'}j} = 0$ otherwise.

The decision variables can be partitioned into optimization variables s_j, $y_{j,k}$ and simulation variables $e_{i,j}^s$, $e_{i,j}^d$, $\delta_{ii^{'}j}$ representing the event time and relationships within the simulation trajectory. Variables δ and $\mathbf{s}$ are linked by function $\Delta_{ii^{'}j}$, a black–box correspondence between starting events and departure events. This reflects the integrated simulation–optimization nature of the DEO model. The DEO model is as follows:

$$\text{MP–O:} \quad \min \left\{ \sum_{j=1}^{m} c_j s_j \right\} \tag{1}$$

s.t.

$$s_j \leq k - 1 + (U_j - k + 1) y_{j,k} \qquad k = L_j, \cdots, U_j,\ j = 1, \cdots, m \tag{2}$$

$$y_{j,k} = 1 \qquad k = 1, \cdots, L_j,\ j = 1, \cdots, m \tag{3}$$

$$s_j \geq \sum_{k=1}^{U_m} y_{j,k} \qquad j = 1,\cdots,m \tag{4}$$

$$e^s_{i,1} \geq e^a_i \qquad i = 1,\cdots,N \tag{5}$$

$$e^s_{i,j} - e^d_{i-k,j} \geq D(ky_{j,k} - s_j) \qquad k = L_m,\cdots,U_m,\ j = 1,\cdots,m,\ i = k+1,\cdots,N \tag{6}$$

$$e^d_{i',j} - e^s_{i,j} \geq M(\delta_{ii'j} - 1) + t_{i,j} \qquad j = 1,\cdots,m,\ i,i' = 1,\cdots,N \tag{7}$$

$$e^s_{i,j} - e^d_{i,j-1} \geq 0 \qquad j = 2,\cdots,m,\ i = 1,\cdots,N \tag{8}$$

$$e^d_{i,j} - e^s_{i-b_j,j+1} \geq 0 \qquad j = 1,\cdots,m-1,\ i = b_j+1,\cdots,N \tag{9}$$

$$\delta_{ii'j} = \Delta_{ii'j}(\mathbf{s}) \qquad j = 1,\cdots,m,\ i,i' = 1,\cdots,N \tag{10}$$

$$\frac{\sum_{i=1}^N e^d_{i,m} - \sum_{i=1}^N e^a_i}{N} \leq T^* \tag{11}$$

$$L_j \leq s_j \leq U_j,\ j = 1,\cdots,m$$

$$s_j \in \mathbb{Z}^+, \delta_{ii'j}, y_{j,k} \in \{0,1\}, e^s_{i,j}, e^d_{i,j} \geq 0 \qquad j = 1,\cdots,m, i,i' = 1,\cdots,N; k = L_j,\cdots,U_j$$

Equation (1) is the objective function, i.e., the minimization of the overall server cost. As servers at a given stage are identical, they have the same cost, while the cost of the servers in different stages can be different. Constraints (2) to (4) guarantee that $y_{j,1} = y_{j,2} = \cdots = y_{j,s_j} = 1$ and $y_{j,s_j+1} = y_{j,s_j+2} = \cdots = y_{j,U_j} = 0$, which is consistent with the definition of variables $y_{j,k}$. Constraints (5)–(10) deal with the dynamic behavior of the system. Specifically, constraints (5) state that the service of a job at the first stage can start only after it arrives. Constraints (6) show that the service of a job at each stage can start only if there is at least one server available. Moreover, with parallel servers, the departure sequence and the arrival sequence may differ from each other. Thus, constraints (7) link the i–th arrived job to the i'–th one leaving stage j though variable $\delta_{ii'j}$, which depends on the server allocation as represented in constraints (10), where $\mathbf{s}$ corresponds to the vector containing all the s_j, and $\Delta_{ii'j}$ is a black–box representation of this correspondence. These constraints are non linear and represent one of the difficulties in solving the model. Constraints (8) allow the service at any stage j to start only after the job leaves the upstream stage. Since the inter–stage queue has finite capacity, constraints (9) represent the fact that a job can leave stage j only if there is available space, i.e., if the number of jobs in the queue following stage j is smaller than b_j. Constraint (11) is the performance constraint, which bounds the average system time over all the jobs to be at most the target value T^*.

The number of binary variables and constraints of MP–O increase quadratically as the simulation length increases, but the complexity to solve the model increases exponentially. Thus, a Benders decomposition based solution approach is proposed in Section 4 to efficiently solve the problem.

4 SOLUTION APPROACH

Benders decomposition is a method developed within mathematical programming to solve a certain class of large scale complex problems. Specifically, it is based on the partition of variables and constraints to decompose the original problem into multiple smaller and/or easier subproblems. Usually, the decomposition leads to a master problem, containing the variables that make the problem difficult to solve, and to a subproblem, containing all the other variables and constraints. The master problem and subproblem are then solved in an iterative framework (Benders 1962).

4.1 Decomposition

Referring to the original model (MP–O), integer and binary variables s_j, $y_{j,k}$ and $\delta_{ii'j}$ are included in the master problem, while variables $e^s_{i,j}$ and $e^d_{i,j}$ are considered subproblem variables. Constraints are partitioned accordingly. The master problem and the subproblem are reported in the following.

Master problem

$$\text{MP–M–1:} \qquad \min \left\{ \sum_{j=1}^{m} c_j s_j \right\}$$
$$\text{s.t.}$$

(2),(3),(4),(10) + *(cuts generated from subproblem solution)*

Subproblem

$$\begin{aligned}
&\min\{\varepsilon\} && && (12)\\
&\text{s.t.} && &&\\
&e^s_{i,1} \geq e^a_i && : \eta_i \qquad i = 1,\cdots,N &\\
&e^s_{i,j} - e^d_{i-s_j,j} \geq 0 && : \beta_{i,j} \qquad j = 1,\cdots,m,\ i = s_j+1,\cdots,N &\\
&e^d_{i',j} - e^s_{i,j} \geq t_{i,j} && : \alpha_{i,j} \qquad j = 1,\cdots,m,\ \bar{\delta}_{ii'j} = 1 &\\
&e^s_{i,j} - e^d_{i,j-1} \geq 0 && : \gamma_{i,j} \qquad j = 2,\cdots,m,\ i = 1,\cdots,N &\\
&e^d_{i,j} - e^s_{i-b_j,j+1} \geq 0 && : \nu_{i,j} \qquad j = 1,\cdots,m-1,\ i = b_j+1,\cdots,N &\\
&\frac{\sum_{i=1}^N e^d_{i,m} - \sum_{i=1}^N e^a_i}{N} - \varepsilon \leq T^* && : \vartheta && (13)\\
&e^s_{i,j}, e^d_{i,j} \geq 0 && j = 1,\cdots,m; i, i' = 1,\cdots,N &
\end{aligned}$$

The subproblem is a feasibility problem by introducing a feasibility variable ε in (12) and (13); hence, only feasibility cuts will be generated. This is due to the fact that the only variables in the objective function of MP–O are s_j, and they cannot be a part of the subproblem. η_i, $\beta_{i,j}$, $\alpha_{i,j}$, $\gamma_{i,j}$, $\nu_{i,j}$ and θ are the dual variables associated to the related constraints. The feasibility cut from the subproblem is:

$$\sum_{i=1}^{N} e^a_i \bar{\eta}_i + D \sum_{j=1}^{m} (\bar{s}_j y_{j,\bar{s}_j} - s_j) \sum_{i=1}^{N} \bar{\beta}_{i,j} + \sum_{j=1}^{m}\sum_{i=1}^{N} t_{i,j}\bar{\alpha}_{i,j} + M \sum_{j=1}^{m}\sum_{i=1}^{N}\sum_{i'=1}^{N} (\delta_{ii'j} - 1) - \bar{\vartheta}(T^* + \frac{\sum_{i=1}^N e^a_i}{N}) \leq 0, \quad (14)$$

where $\bar{\eta}_i$, $\bar{\beta}^r_{i,j}$, $\bar{\alpha}_{i,j}$, $\bar{\vartheta}$ are the dual solution of the subproblem, and $\bar{s}_j$ are the server number of the system (i.e., master problem solution of last iteration).

According to the strong duality theory, the following equation holds:

$$\sum_{i=1}^{N} e^a_i \bar{\eta}_i + \sum_{j=1}^{m}\sum_{i=1}^{N} t_{i,j}\bar{\alpha}_{i,j} - \bar{\vartheta}(T^* + \frac{\sum_{i=1}^N e^a_i}{N}) = \bar{\varepsilon}.$$

Thus, the feasibility cut (14) can be rewritten as (15).

$$D \sum_{j=1}^{m} (\bar{s}_j y_{j,\bar{s}_j} - s_j) \sum_{i=1}^{N} \bar{\beta}_{i,j} + M \sum_{j=1}^{m}\sum_{i=1}^{N}\sum_{i'=1}^{N} (\delta_{ii'j} - 1) + \bar{\varepsilon} \leq 0 \quad (15)$$

However, the master problem MP–M–1 with feasibility cuts (15) and constraints (10) is not linear or convex. To address such complexity issue, i.e., to linearize the master problem, first of all, constraints

(10) have been considered as black-box, where the value of $\delta_{i,i'.j}$ is evaluated through simulation. Once the server allocation $\mathbf{s}$ is known, the system can be simulated, and if the i–th arriving job is the $i^{'}$–th one leaving stage j, $\Delta_{ii'j}(\mathbf{s})$ is set to 1, otherwise $\Delta_{ii'j}(\mathbf{s})$ is set to 0. Using simulation to address constraints (10), introduces a first approximation. However, this is necessary as, even though a MILP formulation of constraints (10) exists (Chan and Schruben 2008), the huge number of big-M constraints would make the model difficult to tackle.

A second approximation is introduced to linearize feasibility cuts (15); in particular, linearization is achieved by eliminating the term $M\sum_{j=1}^{m}\sum_{i=1}^{N}\sum_{i'=1}^{N}(\delta_{ii'j}-1)$, i.e., by neglecting the impact on the system time by a change in the departure sequence. The linearized cut is then:

$$D\sum_{j=1}^{m}(\bar{s}_j y_{j,\bar{s}_j}-s_j)\sum_{i=1}^{N}\bar{\beta}_{i,j}+\bar{\varepsilon}\leq 0. \tag{16}$$

In the feasibility cut (16), if the big-M coefficient D is too large, the linear relaxation is very weak, negatively impacting on the computational time. Combinatorial cuts (Codato and Fischetti 2006) could strengthen the model; however, it would still have an enumeration nature. D equal to the maximum inter–departure time can activate constraints (6) only for $k=s_j$, and deactivate the constraints for all $k\neq s_j$, since

$$D(s_j-k)\geq e^d_{i-k,j}-e^d_{i-s_j,j}\geq e^d_{i-k,j}-e^s_{i,j},\ \forall k\leq s_j-1.$$

If D is set to a smaller value, the cut is strengthened and its efficiency can be assured. However, it will also introduce an approximation. In the approximate formulation, D is set equal to the average inter–departure time $\bar{CT}$ of the system.

The approximate and linearized master problem is then as follows:

$$\text{MP–M–2:}\quad \min\{\sum_{j=1}^{m}c_j s_j\}$$

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

$$\bar{CT}\sum_{j=1}^{m}(\bar{s}^r_j y_{j,\bar{s}^r_j}-s_j)\sum_{i=1}^{N}\bar{\beta}^r_{i,j}+\bar{\varepsilon}^r\leq 0,\ \forall r$$

$$(2),(3),(4).$$

4.2 Solution Algorithm

The complete solution approach for the multi–stage SAP is shown in Figure 2. At the beginning, the arrival times e^a_i and processing times $t_{i,j}$ are generated from specific distributions, and the server number at each stage is set to the lower bound. By default, we set the lower bound equal to the minimal server number allowing system stability, and D equal to average inter–arrival time. However, it should be noticed that D and L_j are the two user–defined parameters of the approximate algorithm. When different values are chosen for these two factors, the algorithm performance might be different, and the effect can be seen in the empirical study in section 5.2. After simulating the system, the cut is then added to the approximate master problem that is solved to find the new best server numbers. In each iteration, the master problem, solved using mathematical programming, updates the lower bound of the solution. The new configuration is simulated and so on until the target performance is achieved.

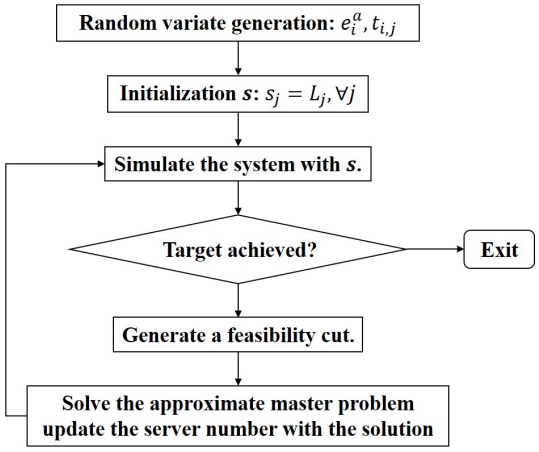


Figure 2: Algorithm.

5 NUMERICAL ANALYSIS

In this section, a numerical analysis is conducted on random generated instances. The aim is to study the algorithm performance in terms of optimality gap and computation time, the impact of parameter choice on the algorithm performance, and the analysis of high dimension problems.

The numerical tests have been performed on an Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz processor and 256GB RAM. The solution algorithm is coded in Java. Cplex 12.5 with default settings is used to solve the master problem.

5.1 Algorithm Performance with Different Problem Parameters

The performance of the algorithm is studied in terms of efficiency and optimality gap. Seven factors have been identified to possibly affect the performance: stage number, coefficient of variation (CV) of processing time, CV of inter–arrival time, queue space, equality of machine processing time, target system time and simulation length. The arrival rate is fixed and equal to 1/2 jobs per time unit. All the queues, with the exception of the first one, are assumed to have the same capacity. Experiments with both truncated normal distribution and exponential distribution for inter–arrival time and processing time are performed based on a two-level full factorial design with 10 replicates. Table 1 shows the high level and low level values of each factor.

In the systems with identical processing time among stages, the mean processing time is equal to 10 time units. In the systems with different processing time, the mean processing time is equal to 15 time units for the first stage, and 10 for the other stages. The target system time is equal to $(1+Target) \times overall\ mean\ processing\ time$ (e.g., for a system with 4 stages of identical average processing time equal to 10, the high level target system time is equal to $(1+12\%) \times 4 \times 10 = 44.8$ time units).

The exact solution of the problems is obtained using an enumeration algorithm on the same sample path. Starting from a number of servers equal to the lower bound, this number is increased by one if all the alternatives proved to have infeasible solution.

The optimality gap is then calculated as the difference of total server number provided by the proposed approximate DEO approach and the one provided by the enumeration algorithm. Furthermore, the computational times of the two approaches are also compared. As the exact DEO model has enumeration nature, the comparison on computation efficiency can show the performance of the approximation relative to the original model. In fact, the approximate approach reduces the computational time for simulation with respect to the cost of solving the MILP. On the contrary, the enumeration algorithm has negligible time for optimization but long simulation time, because it visits all the alternatives.

Table 1: Experimental conditions (factors and levels).

	Normal		Exponential	
Factor	High (1)	Low (-1)	High (1)	Low (-1)
Stage number	6	4	6	4
Processing time CV	1	0.5	–	–
Inter–arrival CV	1	0.5	–	–
Queue capacity	5	2	5	3
Machine Processing time	Identical	Different	Identical	Different
Target	12%	7.5%	12%	7.5%
Simulation length	100,000	10,000	100,000	10,000

5.1.1 Truncated Normal Distribution

In this experiment, the processing time and inter–arrival time follow a truncated normal distribution, where the normal distributed random variate smaller than zero and greater than 2μ is discarded.

For all the 1,280 samples, the optimality gap is equal to 0. The computation time of the proposed approach and of the enumeration algorithm in each sample is shown in Figure 3. The blue dots and the red dots refer to the samples in which the proposed DEO approach is faster than the enumeration, and in which it is slower, respectively. In 173 over the 1,280 samples in which the DEO approach requires longer computation time, the difference is smaller than 1 second, and the total computational time is smaller than 5 seconds. These results show that, for truncated normal distributed processing time and inter–arrival time, the approximation approach leads to null optimality gap, and significant improvement in the computation time.

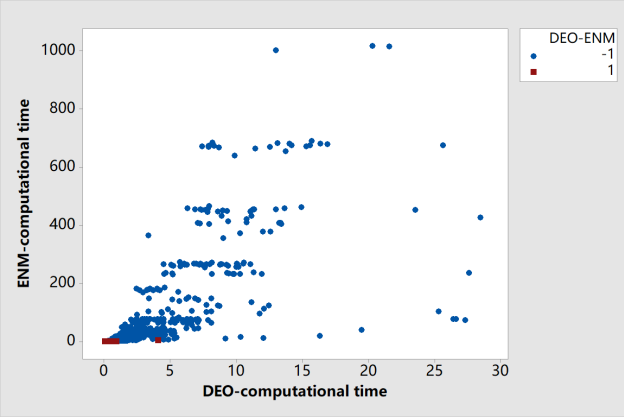
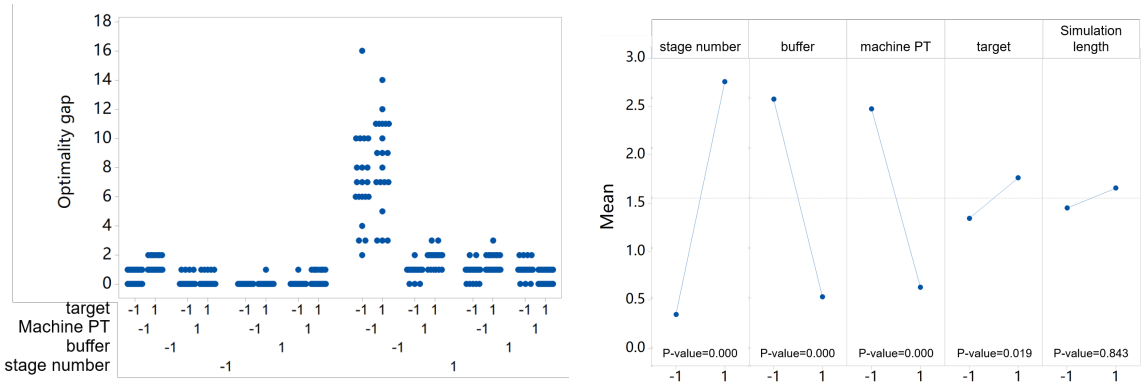


Figure 3: Computational time comparison between DEO algorithm and enumeration algorithm.

5.1.2 Exponential Distribution

In this experiment, the processing time and inter–arrival time follow an exponential distribution. Since the exponential distribution has always CV equal to 1, the coefficients of variation are not relevant. Hence, a two-level full factorial experiment with 10 replicates, 320 samples in total, is conducted. The same enumeration algorithm previously discussed is used to study the optimality gap also in this case.

For the experiment with exponential distribution, the approximate DEO approach solves all the samples faster than the enumeration approach, but the optimality gap is no longer null, instead, it is always between 0 to 3, except for the case with 6 stages, queue space equal to 3 and different mean processing time among stages, as shown in Figure 4(a). Figure 4(b) shows the main effect plot of the five factors (being the CV not



(a) Indivadual value plot.

(b) Main effect plots.

Figure 4: Optimality gap - exponential distribution.

relevant), with p-value of Kruskal-Wallis test provided. It can be shown that the system related factors such as stage number, queue spaces and machine processing time are more significant than the target system time and the simulation length.

The results show that, for exponentially distributed processing and inter–arrival time, the proposed approximate approach does not work as well as in the truncated normal case.

As the performance of the algorithm depends on parameters D and L_j, the impact of their choice on the algorithm performance is studied in section 5.2.

5.2 Effect of Algorithm Parameters

This section analyzes the impact of the server number lower bounds L_j (corresponding to the initial solution), and of the coefficient D on optimality gap and computation time.

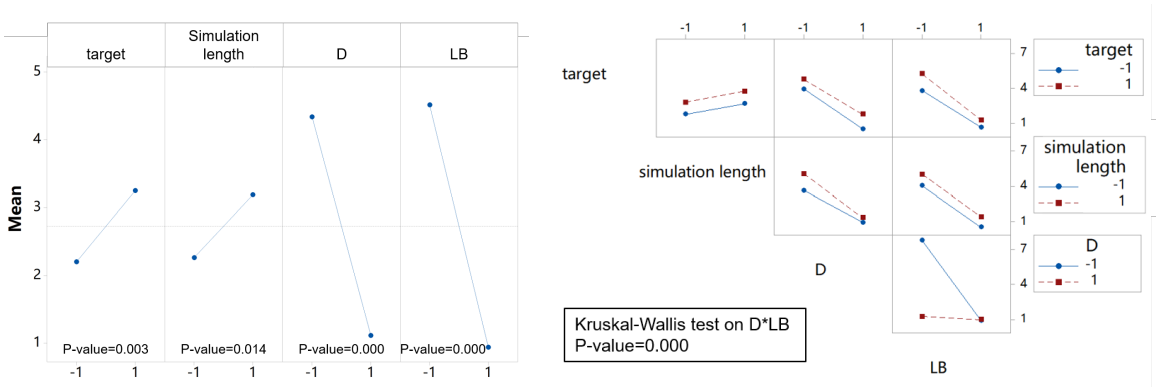
It is clear that increasing D will reduce the number of alternatives cut from the feasible region, thus possibly reducing the optimality gap. Instead, the effect of the initial solution is not that trivial. For example, considering the 6 stage system with queue space equal to 3 and different mean processing times among stages, the server allocation $(8\ 6\ 6\ 6\ 6\ 6)$ leads to an average system time of 119.0 time units over 10 replicates, higher than that of any other alternatives, which have system time shorter than 100 time units. Therefore, approximate cut generated after simulating the initial system can lead to very different optimality gap if different lower bounds L_j are used.

The 6–stage system used in the previous example, whose optimality gap ranges from 2 to 16 (as shown in section 5.1), is used to investigate also the interaction between D and L_j, and the target system time and the simulation length. A two-level full factorial experiment with 10 replicates, 160 samples in total, is conducted. Table 2 shows the high level and low level values of each factor.

Table 2: Parameters of algorithm parameter calibration.

Factor	High (1)	Low (-1)
Lower bound	(9 6 6 6 6 7)	(8 6 6 6 6 6)
D	$2\bar{CT}$	$\bar{CT}$
Target	12%	7.5%
Simulation length	100,000	10,000

The effect of the four factors and their interaction on the optimality gap are shown in Figure 5, with the p-value of Kruskal-Wallis test. Both the lower bound and the value of D significantly affect the optimality gap, as their interaction. On the contrary, L_j and D do not show any interaction with the target and the simulation length.



(a) Indivadual value plot. (b) Main effect plots.

Figure 5: Optimality gap - exponential distribution.

5.3 High Dimension Problems

In this section, the performance of the proposed approximate DEO approach is tested on server allocation problems with a number of stages ranging from 3 to 17. The processing time and inter–arrival time follow a truncated normal distribution, with coefficient of variation of processing time and inter–arrival time equal to 0.5. The arrival rate is equal to 1/2, and the machines in all the stages have the same mean processing time equal to 10 time units. The lower bound of server number is equal to 6. The queue spaces among all the stages are all equal to 5. The simulation length and the target system time is set to $100,000$ jobs and 67.2 time units, respectively. The value of D is set to the average inter–departure time.

The results, including the iteration number, the computation time and the solution are shown in Table 3. It should be noticed that the problem with stage number equal to 18 could not be solved within 48 hours.

Table 3: Results of approximate DEO with variate stage number.

Stage number	Iteration number	Computation time (second)	Solution
4	1	0.452	6 6 6 7
5	3	0.797	6 6 6 7 7
6	4	1.006	6 6 6 6 7 8
7	7	1.645	6 6 6 7 7 7 8
8	17	4.045	6 6 7 7 7 7 7 8
9	30	7.662	6 6 7 7 7 7 7 8 8
10	45	12.505	6 6 6 6 7 7 8 8 8 9
11	101	39.359	6 6 6 6 7 7 8 8 8 9 9
12	211	108	6 6 6 7 7 7 7 8 8 8 9 9
13	336	264	6 6 6 7 7 7 7 7 8 8 9 9 10
14	729	919	6 6 6 7 7 7 7 7 8 8 8 9 9 10
15	1,178	2,915	6 6 6 6 7 7 7 7 8 8 8 9 10 10 11
16	1,946	10,789	6 6 6 7 7 7 7 7 8 8 8 9 9 9 10 10
17	4,532	136,587	6 6 7 7 7 8 7 7 7 8 8 8 9 9 10 10 11

In Table 3, a pattern of increasing server number along the line can be noticed, i.e., more servers are allocated to the downstream stages. This behaviour is not consistent with the well-known *bowl phenomenon* of serial production lines (Rao 1976). This can be due to the fact that the bowl phenomenon is usually found when throughput is taken as performance indicator, while in the case discussed in this paper, the

system time is the performance indicator. To achieve a shorter system time, the blocking phenomenon should be reduced, while the starvation does not have any impact. Thus, a higher number of servers is put in the downstream stages to increase their capacity thus reducing blocking, even though this will increase starvation probability. These two different behaviours imply the reasonable conclusion that the optimal design the serial–parallel line can be different if different KPIs are used to evaluate the system.

Table 3 also shows that the iteration number and the computation time increase as the stage number increases. Specifically, the 17-stage problem requires up to $136,587[s] \approx 38[h]$ to be solved. The plot of the computation time (logarithmic scale) over the stage number is reported in Figure 6, with a fitted quadratic curve. The figure shows that the computation time increases more than exponentially. This can be explained by the fact that higher dimension problems usually have a larger set of alternatives and, hence, require more iterations to be solved. The master problem gets more and more constraints through iterations, and the computational burden of solving the MILP becomes heavier and heavier. Improvement might be obtained by reducing the dimension of the master problem, which will be the object of future research.

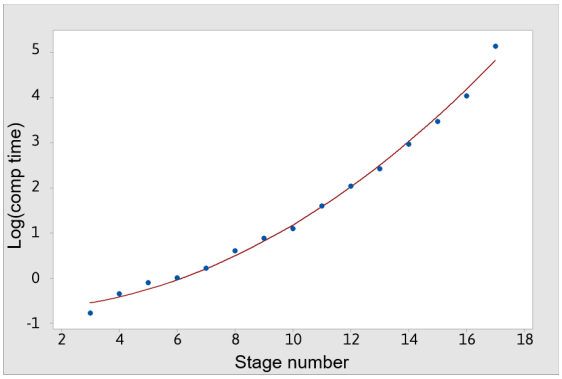


Figure 6: Logarithm of computation time over stage number.

6 CONCLUSION

This paper develops an approximate simulation–optimization algorithm for solving the server allocation problem (SAP) in multi-stage serial–parallel manufacturing systems. The approach is based on the Benders decomposition of a DEO formulation of the problem. To define the feasible region, feasibility cuts are generated based on the structure of the simulation model. Numerical analysis shows that the optimality gap is small in most of the cases, and improvements can be obtained by an adequate choice of the parameters. The proposed approach is able to solve the SAP of systems with up to 17 stages. Future work will be dedicated to the development of adaptive parameter choice and to the improvement of algorithm efficiency for high dimension problems. Moreover, the possibility to adapt the approach to other problems, different from SAP, or to SAP in systems with different architectures, will be investigated.

REFERENCES

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

Benders, J. F. 1962. "Partitioning Procedures for Solving Mixed-Variables Programming Problems". *Numerische Mathematik* 4(1):238–252.

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.

Fu, M. C., F. W. Glover, and J. April. 2005. "Simulation Optimization: a Review, New Developments, and Applications". In *Proceedings of the 37th Conference on Winter Simulation*, edited by M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 83–95. Piscataway, New Jersey: Institute of Electrical and Electronics Engineering, Inc.

Ho, Y.-C. L., and X.-R. Cao. 2012. *Perturbation Analysis of Discrete Event Dynamic Systems*, Volume 145. Berlin: 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.

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

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.

Rao, N. P. 1976. "A Generalization of the Bowl Phenomenonin Series Production Systems". *International Journal of Production Research* 14(4):437–443.

Robbins, H., and S. Monro. 1951. "A Stochastic Approximation Method". *The Annals of Mathematical Statistics* 22(3):400–407.

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

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., and A. Matta. 2018. "Data-driven Downtime Bottleneck Detection in Open Flow Lines". In *Conference on Automation Science and Engineering*. August 20th-24th, Munich, Germany, 1513–1518.

Zhang, M., A. Matta, A. Alfieri, and G. Pedrielli. 2018. "Simulation–Based Benders Cuts: a New Cutting Approach to Approximately Solve Simulation–Optimization Problems". In *Proceedings of the 2018 Conference on Winter Simulation*, edited by M. Rabe, A. A. Juan, N. Mustafee, A. Skoogh, S. Jain, and B. Johansson, 2225–2236. Piscataway, New Jersey: Institute of Electrical and Electronics Engineering, Inc.

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.