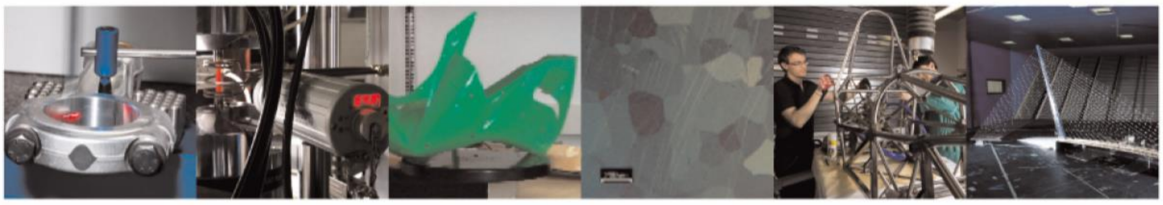


Mathematical programming models for joint simulation-optimization applied to closed queueing networks

Alfieri, Arianna; Matta, Andrea; Pedrielli, Giulia

Mathematical programming models for joint simulation–optimization applied to closed queueing networks

Arianna Alfieri · Andrea Matta · Giulia Pedrielli

Published online: 11 October 2013

1 Introduction

Simulation–optimization of Discrete Event Systems (DESs) is typically carried out by using two different and separate modules: a simulation module for the evaluation of the system performance and an optimization module for the generation of the candidate solutions.

Several methods can be used for optimization (Fu et al. 2005; Healy and Schruben 1991; Chick et al. 2003), as, for example, Response Surface Methodology (RSM) (Myers et al. 2009; Montgomery 2005), stochastic approximation (Kushner and Yin 1997), ranking and selection (Boesel et al. 2003), meta-heuristics (Hong and Nelson 2006) and mathematical programming (Robinson 1996). The output of the optimization module is a system configuration that is then given as input to the simulation module in order to evaluate its quality

A. Alfieri
Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy

A. Matta · G. Pedrielli (✉)
Politecnico di Milano, via Giuseppe La Masa 1, 20156 Milano, Italy
e-mail: giulia.pedrielli@mail.polimi.it

in terms of feasibility and related system performance. The simulation module is usually a discrete event simulator taking as input the outcome of the optimization module and generating the performance of the system as output (Law 2007). The system performance is the input for the next iteration of the optimization module. This iterative procedure continues until the optimal solution is found or a predefined stopping condition is met (Kleijnen 2008; Spall 2003).

This loop usually requires a large number of iterations to reach a good solution and this behaviour can be partially ascribed to the lack of system dynamics modelling within the optimization module, justifying the presence of an external performance evaluator.

An alternative way to simulate a DES is to use mathematical programming (Schruben 2000). The system behaviour is described by a set of equations (constraints) and the solution of the resulting mathematical model represents a single simulation run (Chan and Schruben 2008a).

The use of mathematical programming to simulate DESs naturally leads towards a deeper integration between simulation and optimization. Indeed, if the system dynamics can be de-scribed by means of constraints, these can be embedded within the optimization model. Under this perspective, the simulation module is used in sequence (and not in loop) with the optimizer to provide the evaluation of the system performance given the optimal configura-tion. As a result, a single simulation-optimization iteration is needed to obtain the optimal solution.

However, if optimization is considered, mathematical models are no longer Linear Programming (LP) models (as they usually are for simulation) and Integer Programming (IP) models have to be considered with the related computational burden (Garey and Johnson 1979; Brodsky et al. 2003; Ming-Guang et al. 2002).

Approximate LP models to simulate and optimize DESs have been proposed to overcome this difficulty (Matta 2008). In Alfieri and Matta (2012), an approximate representation of a class of multi-stage production systems with finite buffer capacities is presented. The pro-posed approximation consists in modelling queues as *Time Buffers* (TB), i.e., temporal lags between two events, instead of using the traditional *Space Buffers*. This approximation pre-serves the linearity of the mathematical model even when used for optimization purposes. In addition, an analysis of the structural properties and relationship of the approximate models with the corresponding exact formulations is proposed.

This paper extends the approximation developed in Alfieri and Matta (2012) to the simu-lation and optimization of closed queueing networks, which have been deeply studied in the literature (Matta and Chefson 2005; Chan and Schruben 2008b; Dallery and Liberopoulos 2000; Maggio et al. 2009a; Gershwin and Werner 2007).

Specifically, approximate LP formulations are proposed to evaluate the performance of closed queueing networks and to find the optimal configuration in terms of the number of customers populating the line at any given time (constant in loop systems), i.e., the minimum number of customers that allows to satisfy a predefined level of throughput. We refer to this optimization problem as *Pallet Allocation Problem* (PAP). Although the PAP is not combinatorial, as the Buffer Allocation Problem is (Dolgui et al. 2010), the stochasticity affecting the arrivals and service times, the high correlation between the events that occur in the stages of the line, and the non monotonicity of the throughput curve with respect to the number of pallets (Maggio et al. 2009b; Gershwin and Werner 2007) make the modelling of loop systems a difficult task.

In addition, the solution of the single loop PAP that we propose represents a first step towards the solution of the multiple–loop case (Zhang 2006).

The rest of the paper is organized as follows: Sect. 2 describes the problem, the notation and the main assumptions. Section 3 deals with the different types of blocking that can occur in closed networks. In Sects. 4 and 5 optimization and simulation models are presented, respectively. Section 6 focuses on the algorithms designed to solve the proposed models. The numerical results out of experiments from random generated problem instances are commented in Sect. 7. Section 8 concludes the paper.

2 Assumptions and notations

A closed queueing network is a system in which the number p of customers (the loop population) is kept constant. This type of system appears frequently in factories. Manufacturing processes which utilize pallets or fixtures can be viewed as loops since the number of pallets/fixtures circulating in the system remains constant. Similarly, control policies such as CONWIP and kanban create conceptual loops by limiting the number of parts in the system. Several contributions to the analysis and optimization of closed queueing networks can be found in the literature, proving the relevance of this special class of discrete event systems (Dallery and Frein 1989; Matta and Chefson 2005; Chan and Schruben 2008b; Bouhchouch et al. 1993; Maggio et al. 2009a; Gershwin and Werner 2007).

The class of closed queueing networks analysed in this work considers J machines decoupled by J buffers with finite capacity (c_j) and a single customer type. Each machine j, exception made for the first stage, $j = 1$, has an upstream finite capacity buffer, $j - 1$. The buffer J closes the network and it is the upstream queue of the first stage and the downstream queue of the last server.

We will refer to customer and part interchangeably assuming that each customer requires exactly one part.

Each customer i ($i = 1, \ldots, n$) arrives at the system at time a_i and can be processed by the first machine only in case part $i - p$ has been released from the last machine (being p the number of pallets). As a result, the release time of part i in the loop system is the maximum between its arrival time a_i and the time when part $i - p$ leaves the system. After having been processed by the first machine, parts go to the second machine and so forth until the last operation is performed on the last machine J. Once a part has completed the last operation, it is stored in the buffer J and it is released only when a new customer is available to be processed. We assume that no scheduling decision has to be considered, thus part $i + 1$ is processed after part i.

Customer i has to wait in queue $j - 1$ if the machine in stage j is busy in serving another customer k (with $k < i$).

Machines are modelled as perfectly reliable. The service times $\{t_{ij}\}$ of each customer i at each server j are known in advance from a random sampling or from a specific sample path, whereas transportation times are considered negligible or already included in service times.

3 Blocking phenomena in closed queueing networks

The closed queueing networks described in Sect. 2 can be affected by blocking phenomena due to:

- Buffers with finite capacity;
- Constant number of pallets allowed.

These two factors lead to three types of blocking phenomena: (1) deadlock, (2) server blocked because the downstream buffer is full (*server blocking*), (3) part blocked from being released from the system because of constant pallets (*customer blocking*). In the following each of these phenomena is separately described.

Deadlock In case the number of pallets circulating in the system is equal to the total capacity of the buffers, we incur in the deadlock phenomenon. If this condition occurs, then all servers are blocked. In order to avoid this situation, the maximum number of pallets P that can circulate in the system is bounded by the following quantity:

$$P = \sum_{j=1}^{J} c_j - 1. \tag{1}$$

Then at least a free position within the closed network has to be free in order to avoid this blocking.

Server blocking Each customer will be allowed to enter and being processed on the j -th machine only if the number of parts in the downstream stage is strictly less than the capacity c_j of the j -th queue. The capacity c_j is discrete and it includes the customer in service at stage $j + 1$. In this paper the Server Blocking is modelled assuming blocking before service (BBS) control rule. This type of blocking was already considered in Alfieri and Matta (2012) when modelling open queueing networks with finite buffer capacities and stochastic processing times.

Customer blocking Parts to be released from the network can be blocked, i.e., forced to wait in the last queue to keep the number of parts constant. This happens when no new part is ready to enter the system when a part is completed. To model this particular type of blocking, an additional *virtual* server, $J + 1$, was added to the J stages in the loop.

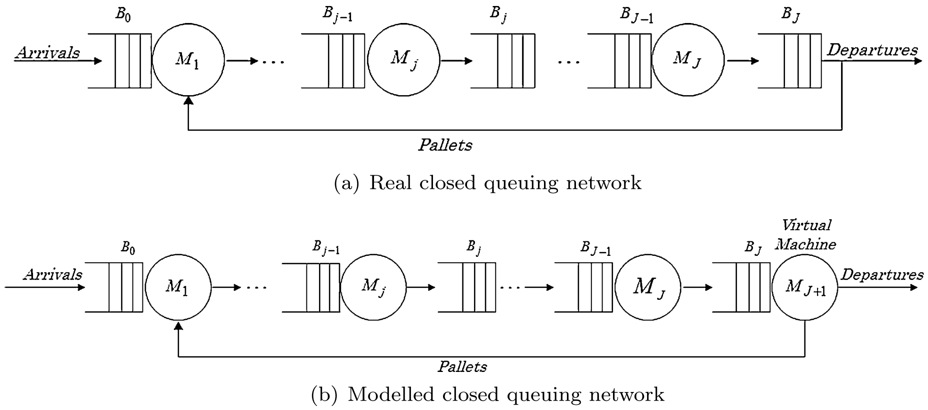
This server has processing time equal to 0 for each part and it is directly followed by the first machine (without decoupling buffer). This machine is blocked only in case it holds a customer and the first server is busy, i.e., Blocking After Service (BAS) rule is assumed for the virtual server.

A physical interpretation of the virtual machine is that it synchronizes a part entering the first stage with a corresponding customer leaving stage J , thus modelling blocking phenomena due to the constant number of customers in the line. Under a modelling perspective, i.e., in the scope of developing the mathematical models for simulation–optimization, the *virtual machine* enables the separation between the constraints modelling blocking due to the finite buffer capacity and the constraints modelling the blocking ascribed to the constant number of customers populating the system. This separation is fundamental to obtain the linear approximation of the simulation–optimization models.

Figure 1 represents the "real" system, i.e., the network with J stages, and the "modelled" system, i.e., the system with the additional virtual server. The two systems are "equivalent" in the sense established by Definition 1.

Definition 1 Equivalent Systems The modelled system is *equivalent* to the real system if, under the same sequences $\{a_i\}$ and $\{t_{ij}\}$, they produce exactly the same sample path.

Property 1 establishes the relationship between modelled and real system.



(a) Real closed queuing network

(b) Modelled closed queuing network

Fig. 1 Equivalent loop representations

Property 1 Let c'_j and c_j be the capacity for the last stage in the modelled and in the real loop system, respectively.

The modelled and the real systems are equivalent iff $c'_j = c_j - 1$.

Proof The last server in the modelled system, $j = J + 1$, has no downstream queue, then it is blocked only if the first machine is busy. Since, by definition of virtual server, $t_{i,J+1} = 0, \forall i$, the customer entering the virtual machine is not forced to wait a predefined amount of time. As a result, the virtual server can be interpreted as the last buffer slot of the network and $y_{i,J+1}$ is the time when part i leaves buffer J in the real system. Since the last buffer slot is, with the virtual machine, considered separately, the total buffer slots downstream server J are $c'_j + 1$. Hence, the modelled and real system are equivalent only when $c'_j + 1 = c_j$. $\square$

4 Optimization models

In this section, approximate LP formulations are proposed to find the optimal number of pallets populating the line, i.e., the minimum number of customers that can circulate into the system honouring a predefined target mean throughput ϑ that we estimate as:

$$\hat{\vartheta} = \frac{n - d}{y_{n,J+1} - y_{d,J+1}}, \tag{2}$$

where $y_{n,J+1}$ is the finishing time of the last part at the virtual machine $J + 1$ in the simulated sample path and corresponds to the time when the last customer leaves the system. The parameter d represents the end of the system warm-up (Law 2007).

4.1 Exact optimization model

The pallet allocation problem can be formulated as the following IP model:

$$\min \quad \sum_{k=1}^{P} e_k \cdot z_k \cdot k \tag{3}$$

s.t.

$$y_{i1} \geq a_i + t_{i1} \quad \forall i \tag{4}$$

$$y_{i+1,j} - y_{ij} \geq t_{i+1,j} \quad \forall j, i = 1, \ldots, n-1 \tag{5}$$

$$y_{i,j+1} - y_{ij} \geq t_{i,j+1} \quad \forall i, j = 1, \ldots, J \tag{6}$$

$$y_{i+c_j,j} - y_{i,j+1} \geq t_{i+c_j,j} \quad j = 1, \ldots, J, \; i = 1, \ldots, n - c_j \tag{7}$$

$$y_{i+k,1} - y_{i,J+1} \geq z_k \cdot t_{i+k,1} - (1 - z_k)M \quad \forall k, i = 1, \ldots, n-k \tag{8}$$

$$y_{i,J+1} - y_{i+k-1,1} \geq -(1 - z_k)M \quad \forall k, i = 1, \ldots, n-k \tag{9}$$

$$\hat{\mu} = \frac{y_{n,J+1} - y_{d,J+1}}{n-d} \leq \mu^* \tag{10}$$

$$\sum_{k=1}^{P} z_k = 1 \tag{11}$$

The decision variables $\{y_{ij}\}$ represent the finishing times of parts $i = 1, \ldots, n$ at servers $j = 1, \ldots, J + 1$ ($J + 1$ being the virtual server). The binary decision variable z_k is equal to 1 if a number of pallets equal to k ($k = 1, \ldots, P$) is assigned to the line. The objective is to minimize the total cost obtained multiplying k for the cost e_k to have k pallets in the line. In the following, without loss of generality, we will assume $e_k = 1$, $\forall k$.

Constraints (4) do not allow the service of customer i at the first machine to finish before its arrival time a_i plus its service time. A machine cannot serve two consecutive customers at the same time (5) and a customer cannot be processed by two different servers at once (6). Constraints (7) prevent a customer to leave a machine if the immediate downstream buffer is full.

Constraints (8) and (9) keep the number of parts in the system at a value k equal to the number of pallets assigned to the line. These constraints are made redundant when $z_k = 0$, by subtracting the big–M from the right hand side. Constraints (7) are separated from (8) and (9) thanks to the virtual machine as outlined in Sect. 3.

Constraint (10) bounds the performance measure to the limit μ^*. Finishing times can assume only positive values in the real domain because the arrival times are non negative parameters.

Finally, only one value k can be chosen, as stated by Eq. (11).

4.2 Approximate optimization model

In order to approximate the IP optimization model described in Sect. 4.1, the time buffer concept presented in Alfieri and Matta (2012) needs to be adapted to be applied to closed queueing networks.

4.2.1 Time buffer for closed queueing networks

If p customers circulate in the line, *two* events are directly influenced by the value of p: the start event of part $i + p$ (γ in Fig. 2), and the finish event of part $i + p - 1$ (β in Fig. 2) occurring at stage $j = 1$ at times $x_{i+p,1}$ and y_{i+p-1}, respectively.

As a result, two different time buffer types, **s** and **b**, need to be introduced. The time buffer **s** can anticipate $x_{i+p,1}$ and **b** plays the same role with respect to $y_{i+p-1,1}$. The arrow from node β to node i in Fig. 2 represents the influence between the departure time of

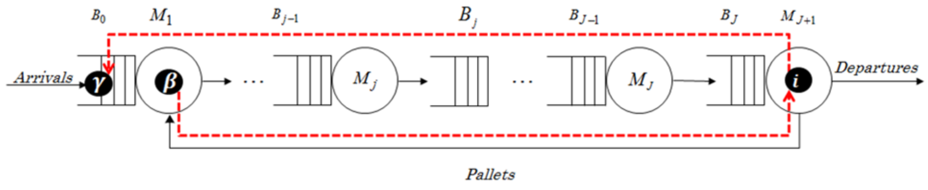
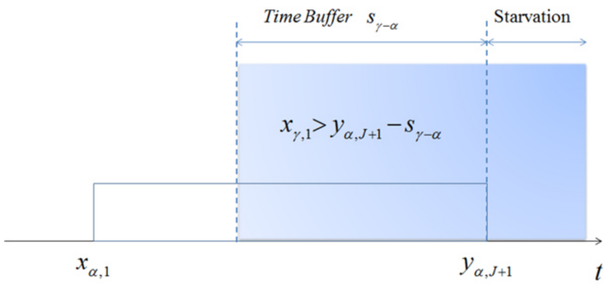


Fig. 2 Closed loop system behaviour

Fig. 3 Time buffer s_k with $k = \gamma - \alpha$


Time Buffer $s_{\gamma-\alpha}$
Starvation
$x_{\gamma,1} > y_{\alpha,J+1} - s_{\gamma-\alpha}$
$x_{\alpha,1}$
$y_{\alpha,J+1}$
t

customer i from the system ($y_{i,J+1}$) and the finishing time of customer $i + p - 1$ at the first server ($y_{i+p-1,1}$). The arrow from node i to node γ represents the influence of $y_{i,J+1}$ over $x_{i+p,1}$. Formally, given two customers, α and γ, to be served in the sequence $\alpha \rightarrow \gamma$, a time length of s forces the starting time of customer γ at the first stage (denoted as $x_{\gamma,1}$) and the finishing time of customer α at the last stage (denoted as $y_{\alpha,J+1}$) in the following way:

$$s_{\gamma-\alpha} \geq y_{\alpha,J+1} - x_{\gamma,1}.$$

Indeed, customer γ can start s time units before customer α leaves the system (Fig. 3). When $s = 0$, customer γ is forced to wait customer α to leave the system before entering, i.e., no more than $\gamma - \alpha$ parts can circulate in the loop.

Given two customers α and β, with $\alpha < \beta < \gamma$, a time length of b forces the departure time of customer α from the system (denoted as $y_{\alpha,J+1}$) and the finishing time of customer β at the first stage (denoted as $y_{\beta,1}$) in the following way:

$$b \geq y_{\beta,1} - y_{\alpha,J+1}.$$

Customer α can exit b time units before customer β finishes its process in the first stage (Fig. 4). When $b = 0$, customer α cannot be released until customer β leaves the first station, i.e., at least $\beta - \alpha$ customers circulate in the loop.

It is worth mentioning that the time buffer concept is different from that adopted in manual assembly lines, mainly related to dimensioning the speed of the conveyor between two adjacent machines. It also differs from the slack time in PERT graphs, which can be defined as the time available between the estimated completion time of the job and its due date.

4.2.2 *Time buffer model*

In the approximate optimization model, the decision variable z_k is replaced by variables s_k and b_k. Consequently, we need to consider whether the minimization of s_k or the minimization of b_k has to be taken into account.

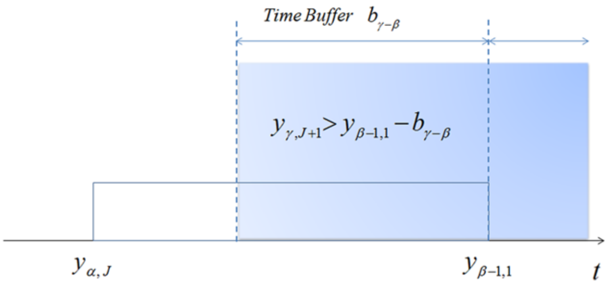

Time Buffer $b_{\gamma-\beta}$
$y_{\gamma,J+1} > y_{\beta-1,1} - b_{\gamma-\beta}$
$y_{\alpha,J}$
$y_{\beta-1,1}$
t

Fig. 4 Time buffer b_k with $k = \gamma - \beta$

In the case we minimize s_k, the approximate optimization objective function results as follows:

$$\min \sum_{k=1}^{P} s_k. \tag{12}$$

The larger the decision variable s_k is, the higher the importance of having a number of pallets k in the system will be.

In case b_k is the decision variable to be optimized, the following objective function must be used:

$$\min \sum_{k=1}^{P} b_k. \tag{13}$$

As for s_k, if b_k is large the importance of having a number of pallets k in the system will be high.
The approximate optimization model constraints can be easily devised from those defined for the exact formulation (4)–(11) by removing constraint (11) and replacing (8)and (9) by:

$$y_{i+k,1} - y_{i,J+1} \geq t_{i+k,1} - s_k \quad k = 1, \ldots, P,\ i = 1, \ldots, n-k \tag{14}$$

$$y_{i,J+1} \geq y_{i+k-1,1} - b_k \quad i = 1, \ldots, n-k+1,\ k = 1, \ldots, P \tag{15}$$

4.2.3 Stochastic time buffer model

When mathematical programming models are adopted for the simulation–optimization of multi-stage open queueing networks (Alfieri and Matta 2012), a sample path based framework can be used to solve the approximate optimization model (Robinson 1996; Pedrielli 2013).

The approximate models presented in the previous sections have been solved running multiple independent replications, obtaining a *set* of optimal solutions in terms of time buffer **s** (or **b**, depending on the objective function adopted).

However, the interpretation of the results coming from the different replications represents an issue difficult to tackle (Healy and Schruben 1991).

As an example, we might perform the sample path optimization for $\mathcal{R} = 5$ replicates, obtaining $\mathcal{R}$ optimal solutions $\{\mathbf{Z}_1^*, \ldots, \mathbf{Z}_{\mathcal{R}}^*\}$. In case the variables are integer, the average solution, $\bar{\mathbf{Z}}$, might be infeasible, i.e., not integer. Moreover, even if the average solution is integer, it might be such that $\hat{\mu_r}(\bar{\mathbf{Z}}) \geq \mu^*$, where μ^* is the target performance, hence resulting infeasible. The same considerations hold for the approximate optimization model. We propose the application of stochastic programming (Birge and Louveaux 1997; Higle and Sen 1996) to solve the PAP problem, tackling the described interpretation issue.

Two-stage stochastic programming approaches are based on the separation of the set of decisions variables in *first-stage decisions*, to be taken before the observation of any of the uncertain elements, and *second-stage decisions*, to be taken after the occurrence of uncertain events.

In our case, the first stage decisions correspond to the time buffers s_k or b_k, according to the adopted objective function (i.e., $\min s$ or $\min b$). The second stage decisions are represented by the finishing times $\{y_{ij}\}$ and the time buffer which is not minimized at the first stage.

Let Ω be the set of all the considered scenarios. Each scenario $\omega \in \Omega$ is completely defined by the realizations of the stochastic parameters $\{a_i(\omega)\}$ and $\{t_{ij}(\omega)\}$.

The variable y_{late} (we call it *late* since it represents the delay with respect to the finishing time needed to reach the target performance) was introduced to link the first and second stage decisions:

$$y_{\text{late}}(\omega) = \big(y_n(\omega) - y_d(\omega)\big) - \mu(n-d), \tag{16}$$

where $\mu = 1/\vartheta$. The expected value of the variable $y_{\text{late}}(\omega)$ is:

$$E\big[y_{\text{late}}(\omega)\big] = \frac{1}{|\Omega|}\sum_{\omega\in\Omega}\big[\big(y_n(\omega) - y_d(\omega)\big) - \mu(n-d)\big]. \tag{17}$$

This equation represents the difference between the expected target finishing time ($\mu(n-d)$), given as input, and the finishing time obtained from the optimization of the $\{y_{ij}(\omega)\}$ in the second stage ($y_n(\omega) - y_d(\omega)$).

The first stage problem, minimizing $\sum s_k$, can be written as follows:

$$\min$$

$$\sum_{k=1}^{P} s_k + \alpha \cdot E\big[y_{\text{late}}(\omega)\big] \tag{18}$$

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

$$s_k \geq 0 \quad \forall k \tag{19}$$

The term $\alpha \cdot E[y_{\text{late}}(\omega)]$ avoids solutions in which the target finishing time constraint is not satisfied (i.e., $y_{\text{late}}(\omega) > 0$). The coefficient α is an input parameter representing the weight of the stochastic part.

The second stage problem(s), one for each scenario, can be written as follows:

$$y_{\text{late}}(\omega) = \min\big(y_n(\omega) - y_d(\omega)\big) - \mu(n-d)$$

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

$$y_{i1}(\omega) \geq a_i(\omega) + t_{i1}(\omega) \quad \forall i$$

$$y_{i+1,j}(\omega) - y_{ij}(\omega) \geq t_{i+1,j}(\omega) \quad \forall j, i = 1, \ldots, n-1$$

$$y_{i,j+1}(\omega) - y_{ij}(\omega) \geq t_{i,j+1}(\omega) \quad \forall i, j = 1, \ldots, J$$

$$y_{i+c_j,j}(\omega) - y_{i,j+1}(\omega) \geq t_{i+c_j,j}(\omega) \quad j = 1, \ldots, J,\ i = 1, \ldots n - c_j$$

$$y_{i+k,1}(\omega) - y_{i,J+1}(\omega) \geq t_{i+k,1}(\omega) - s_k \quad i = 1, \ldots, n-k, \forall k$$

$$y_{i,J+1}(\omega) - y_{i+k-1,1}(\omega) \geq -b_k(\omega) \quad i = 1, \ldots, n-k+1, \forall k$$

All constraints are exactly the same as those presented in Sect. 4.2. The only difference is that each variable and parameter depends on the scenario ω.

If we minimize the function $\sum b_k$, the first and second stage models change as follows.

First stage problem

$$\min$$

$$\sum_{k=1}^{P} b_k + \alpha \cdot E\big[y_{\text{late}}(\omega)\big] \tag{20}$$

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

$$b_k \geq 0 \quad \forall k \tag{21}$$

Second stage problem The second stage problem differs from the one previously presented only in the constraints involving the time buffers that become:

$$y_{i+k,1}(\omega) - y_{i,J+1}(\omega) \geq t i \ + k,\ 1(\omega) - s_k(\omega)\ i = 1, \ldots, n \ - k,\ \forall k$$
$$y_{i,J+1}(\omega) - y_{i+k-1,1}(\omega) \geq -b_k i = 1, \ldots, n \ - k + 1,\ \forall k$$

The solution of the stochastic approximate optimization model is a unique time buffer [*here and now solution* (Birge and Louveaux 1997)].

4.3 Optimization models properties

The structural properties of the optimization models presented in the previous section are discussed in the following.

Proposition 1 *There always exists an optimal solution in which* $\{s_k\}_{k \leq P}$*is a non-increasing sequence and* $\{b_k\}_{k \leq P}$*is a non-decreasing sequence of non negative real values.*

Proof Consider constraints (14); customer i and customer $i+k$ are related through:

$$y_{i+k,1} - t_{i+k,1} \geq y_{i,J+1} - s_k. \tag{22}$$

On the other hand $y_{i,J+1}$ must also satisfy:

$$y_{i+k+1,1} - t_{i+k+1,1} \geq y_{i,J+1} - s_{k+1}. \tag{23}$$

Because of customer sequence constraints (5), the following condition must hold:

$$y_{i+k,1} - t_{i+k,1} \leq y_{i+k+1,1} - t_{i+k+1,1}. \tag{24}$$

If $s_k \leq s_{k+1}$, it would allow customer $i + k + 1$ to start service, at stage 1, before customer $i + k$. However, customer $i + k$ must precede customer $i + k + 1$ (Eq. (24)); hence, the additional time $s_{k+1} - s_k$ can never be used by customer $i + k + 1$ to start. As a result, there always exists an optimal solution in which $s_k \geq s_{k+1}$.

The proof does not change in the case we minimize the function $\sum b_k$, therefore it is not reported.

Proposition 2 *The solution obtained minimizing* $\sum_k s_k$ *corresponds, in the space domain, to the one with the minimum number of pallets.*

The solution obtained minimizing $\sum_k b_k$ *corresponds, in the space domain, to the one with the maximum number of pallets.*

Proof Consider the following inequalities obtained manipulating constraints (14) and (15):

$$-b_{k+1} \leq y_{i,J+1} - x_{i+k,1} \leq s_k\,. \tag{25}$$

If $\sum_k s_k$ is minimized, the interval defined in (25) is bounded from above. In other words, given a target performance μ, the solution $\tilde{p}$ we obtain minimizing the sum of s_k is the one with the lowest turnaround time, defined as the interval between the time x_{i1} when the customer i enters the loop and the time $y_{i,J+1}$ when it exits. Since the turnaround time is monotonic with respect to the number of pallets in the system (Proposition 3, Sect. 5.3), the solution with the lower turnaround time corresponds to the one with the lower number of pallets.

If $\sum_k b_k$ is minimized, the opposite situation happens leading to the solution with the higher turnaround time.

5 Simulation models

The models for optimization presented in Sect. 4 only provide an estimate of the average throughput lower bound (Alfieri and Matta 2012). Indeed, the values $\{y_{ij}\}$ obtained once the models are solved are feasible but there is no guarantee that they are the optimal (i.e., minimized) since they do not appear explicitly in the objective function. As a result, if we refer to Eq. (2), the value of ϑ is overestimated. In case the performance need to be precisely assessed, simulation models have to be developed (Schruben 2000). In this section, the exact and approximate models to simulate loop systems are presented together with their structural properties.

Simulation models receive, as input, data characterizing the arrival process of customers into the system $\{a_i\}$, the service time of each customer on each machine of the closed network $\{t_{ij}\}$ and the number of pallets populating the system (in terms of the integer value p or *time buffers*). Specifically, pallets or time buffer values are assumed to be the results of the solution of the optimization models (exact or approximate, respectively).

The output of the model are the finishing times $\{y_{ij}\}$ of the customers on every machine of the line, which are used to estimate the throughput of the system ϑ (using Eq. (2)).

5.1 Exact simulation model

A closed queueing network with J stages can be simulated by the following LP model (Chan and Schruben 2008b).

min

$$\sum_{i=1}^{n}\sum_{j=1}^{J} y_{ij} \tag{26}$$

s.t.

$$(4)–(7)$$

$$y_{i+p,1} - y_{i,J+1} \geq t_{i+p,1} \quad i = 1, \ldots, n - p \tag{27}$$

$$y_{i,J+1} - y_{i+p-1,1} \geq 0 \quad i = 1, \ldots, n - p + 1 \tag{28}$$

$$y_{ij} \geq 0 \; i, \; j$$

Constraints (27) limit the maximum number of parts in the system to p, while constraints (28) impose that there must at least p parts in the system: the i-th part is forced to wait for the $(i + p - 1)$-th part to exit the first machine before leaving the system.

5.2 Approximate simulation model

The approximate mathematical model for simulation differs from the exact one only in constraints (27) and (28) that are replaced by (14) and (15), respectively. However, in case of simulation, both $\{s_k\}$ and $\{b_k\}$ are known in advance (as the result of the optimization model) and the only decision variables are the finishing times $\{y_{ij}\}$.

5.3 Simulation models properties

It is possible to univocally describe a sample path resulting from the solution of the simulation model by the tuple $\mathcal{Q} = (\mathcal{P}, \mathcal{E})$, where $\mathcal{P}$ and $\mathcal{E}$ represent the closed–loop system configuration and the customer characteristics. Specifically, $\mathcal{P}$ represents the number of pallets in the system. In the case we refer to the approximate model $\mathcal{P} = \{\mathcal{S}, \mathcal{B}\}$, where $\mathcal{S}$ is the vector $\mathbf{s}$ containing time buffer capacities $\{s_k\}$ and $\mathcal{B}$ is the vector $\mathbf{b}$ containing time buffer capacities $\{b_k\}$. The quantity $\mathcal{E}$ is an $n \times (J + 1)$ matrix that contains the arrival times $\{a_i\}$ and the service times $\{t_{ij}\}$.

The number of pallets p populating the system is strongly related to the customer *turnaround time* defined as the time elapsing between the moment the customer enters the network (x_{i1}) until its departure $(y_{i,J+1})$. This relationship is defined in Proposition 3.

Proposition 3 *The turnaround time f_i of customer i is a monotonic function of the number of pallets p.*

Proof Let the function f_i be defined as follows:

$$f_i = y_{i,J+1} - x_{i,1}. \tag{29}$$

Let p be the fixed number of customers circulating in the loop and assume that the capacity of each buffer satisfies $c_j > n, \forall j$, i.e., each buffer could host all the parts visiting the closed system during the simulation run. In other words, we assume no deadlock or server blocking can happen.

We can derive x_{i1} from the time $x_{i+1,1}$ when customer $i + 1$ enters the system in the following way:

$$x_{i1} = \max\{x_{i+1,1} - t_{i1}, a_i\}. \tag{30}$$

Assuming $a_i = 0 \; \forall i$, the following holds:

$$x_{i1} = x_{i+1,1} - t_{i1}. \tag{31}$$

Considering together equations (29) and (31), we can write:

$$f_i = y_{i,J+1} - \left(x_{i+p,1} - \sum_{k=1}^{k=p-1} t_{i+k,1} \right). \tag{32}$$

From constraints (27) and (28), $y_{i,J+1} = x_{i+p,1}$; hence, Eq. (32) becomes:

$$f_i = \sum_{k=1}^{k=p-1} t_{i+k,1} \quad \forall i. \tag{33}$$

As p increases, the function f_i increases as well; hence, it is a monotonic function in the number of pallets.

Since the throughput is an increasing and then decreasing function of the number of pallets (Gershwin and Werner 2007), it follows from Proposition 3 that the throughput is non monotonic with respect to the turnaround time. As a result, the same throughput ϑ can be reached with different values of f_i.

If the approximate model is considered, the time buffers $\{s_k\}$ and $\{b_k\}$ are characterized in Remark 1.

Remark 1 Let $\mathcal{Q} = (\mathcal{S}, \mathcal{B}, \mathcal{E})$ be an approximate formulation of a closed queueing network sample path. The objective function value χ obtained solving the approximate simulation model is a monotonic function in the time buffers s_k and b_k.

The time buffers can also be exploited to derive a configuration in terms of the (near) op-timal number of pallets that populate the system. We will refer to this approximate solution as $\tilde{p}$.

To compute $\tilde{p}$, we consider that, if s_k (b_k) is positive, at least k pallets are needed in the system. In addition, when approximate simulation is run, s_k (b_k) directly influence the values taken by the decision variables $\{y_{ij}\}$, which are used to compute $\tilde{p}$. Specifically, for every part i, the condition $y_{i,J+1} \le y_{i+k,1}$ is verified and the maximum value of k for which the condition holds is collected as k_i. Indeed, if this condition holds, at least k pallets need to be allocated to the system. The approximate solution is then computed as:

$$\tilde{p} = \max_i k_i \tag{34}$$

One of the most relevant differences between the exact and approximate models is that the approximate models do not keep the number of parts in the system constant. The solution $\tilde{p}$ will then be, in general, different from the solution we would obtain solving the exact optimization model.

6 Algorithm

The procedure of single run simulation–optimization can be described as follows:

1. **Initialization**
   (a) Set the parameters describing the system: number of buffers and machines (J), buffer capacities (c_j);

(b) Set the simulation length as the number of parts to simulate (n);
(c) Set the warm–up length (d);
(d) Set the target throughput value ϑ;
(e) Generate the sample path, i.e., parameters t_{ij} and a_i, from a predefined probability distribution;

2. **Approximate Optimization**
   Feed the mathematical model defined by equations (12) or (13), (4)–(7), (14)–(15) and (10) with the input parameters from initialization. Solve the LP approximate optimization model and store the arrays **s** and **b**.
3. **Approximate Simulation**
   Feed the approximate simulation model with the following data:

   – initialization data except for the target throughput (the same data used for the approximate optimization);
   – optimal arrays **s** and **b**.

   (a) Solve the approximate simulation model defined by equations (4)–(7) and (14)–(15). Store the values of $\mathbf{y_1}$ (vector containing the finishing time of each part i at the first machine) and $\mathbf{y_{J+1}}$ (vector containing the finishing time of each part i at the last machine).
   (b) Compute the approximate integer solution using Eq. (34).

In the case stochastic programming approach is adopted, the procedure presented has to be modified as follows:

– Initialization: define the number of scenarios $|\Omega|$ (the probability of each scenario will be equal to $1/\Omega$) and generate the sample path, i.e., parameters $t_{ij}(\omega)$ and $a_i(\omega)$, from a predefined probability distribution for each scenario.
– Approximate optimization: feed the two-stage mathematical model. Solve the LP approximate optimization model and store the optimal vectors **s** and **b**.

7 Numerical results

Numerical experiments on random generated instances have been carried out on three test cases. The simulation–optimization methodology is applied under both multiple replication and stochastic programming approach. The same generated instances were also used to feed standard simulation models developed in Arena$^{©}$. The results obtained from standard simulation are the same as those obtained from the mathematical programming models, thus validating the correctness of the mathematical formulations.

7.1 Experimental settings

The multiple replication optimization has been performed over sample paths characterized by $n = 5000, d = 2000$ and different values of target throughput (ϑ^*). For each target ϑ^*, 10 independent replications were run using $\min \mathbf{b}$ and $\min \mathbf{s}$ as objective function, alternatively. For each replication, the exact optimization model was run to compute the optimal sample path solution.

The stochastic programming approach was characterized by $|\Omega| = 20$ independent scenarios (for each value of the target throughput ϑ^*); each scenario was characterized by a probability $\alpha = 0.05$ and a number of parts equal to $n = 3000, d = 1000$.

The customers are assumed to arrive at time $a_i = 0, \forall i$, i.e., all customers are available to be processed at time 0.

Table 4 contains the results of the approximate optimization with stochastic programming for all the experiment sets. In particular, column ϑ^* is the target throughput given as input to the approximate optimization model, p_{IP} represents the optimal solution obtained solving the IP model. Columns $\tilde{p}_s$ and $\tilde{p}_b$ contain the approximate solutions obtained solving the stochastic programming models when the function $\sum_k s_k$ or $\sum_k b_k$ is minimized, respectively.

Columns ϑ_s and ϑ_b are the throughput obtained solving the approximate simulation model, provided the optimal solution $\mathbf{s}^*$ and $\mathbf{b}^*$ (corresponding to the minimization of $\sum s_k$ and $\sum b_k$), respectively. More specifically, ϑ_s and ϑ_b have been computed solving the approximate simulation model with a sample path size $n = 31500, d = 5000$.

For all the experiments, the CPU time has been about 2 minutes for a single run of 5000 parts in the multiple replicate case, and 5 minutes for the solution of the stochastic problem.

7.2 Experiment set #1

The first experiment set is characterized by a number of machines $J = 4$. The buffer capacities are identical and equal to $c_j = 5, j = 1, \dots, 4$. The processing times for each machine were generated from an exponential distribution with parameters $\tau = 6$ (representing the mean), for stages $j = 1, 2, 4$, and $\tau = 7$ for stage $j = 3$.

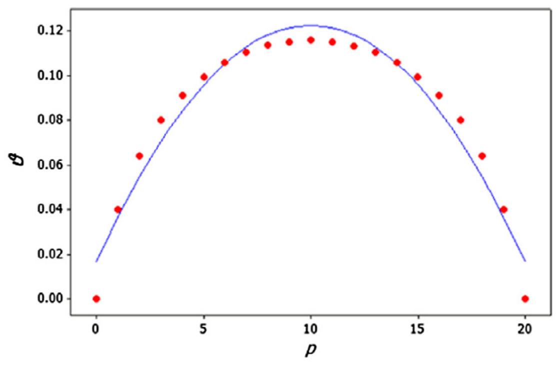
Exact simulation results The system has been studied solving the exact simulation model for every pallet configuration, $p = 1, \dots, P$ with $P = 19$. Numerical results are presented in Fig. 5(a), showing the mean throughput ϑ (dotted line), together with the quadratic fit (plain line), from the 10 independent replications. The half width related to the 95 % confidence interval for the average throughput was estimated to be less than 0.0354. The maximum throughput is achieved when the number of pallets allocated to the system is around 11 (based on the 10 sample paths).

Simulation–optimization results Three target throughput levels were tested: $\vartheta^* = (0.063, 0.106, 0.114)$.

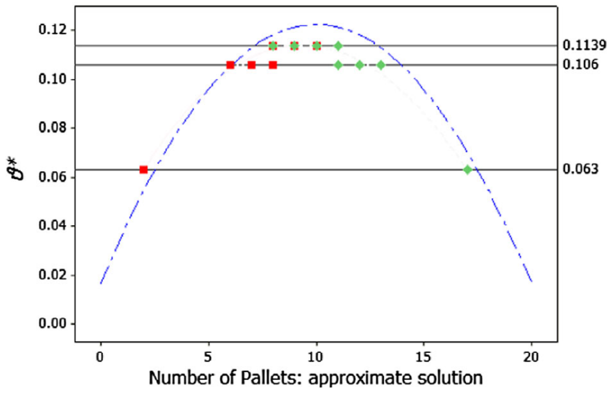
Figure 5(b) represents the results obtained from the 10 replications of simulation–optimization. The value of throughput ϑ (y axis) and the corresponding approximate number of pallets computed from the optimization models (x axis) are reported. Both the minimization of $\mathbf{s}$ and $\mathbf{b}$ are considered. The red dots represent the throughputs obtained simulating the optimal time buffer configuration when $\mathbf{s}$ is minimized (corresponding to the x value $\tilde{p}_s$), whereas the green dots have the same meaning but refer to the minimization of $\mathbf{b}$ (corresponding to the x value $\tilde{p}_b$). The minimization of $\mathbf{b}$, given the target throughput ϑ^*, leads to a solution that is symmetric to the one obtained by minimizing $\mathbf{s}$. More specifically, the solution obtained minimizing $\mathbf{b}$, for the same throughput, has the largest number of pal-lets. This result confirms Proposition 2. In addition, as the target throughput increases, approaching the maximum throughput, this difference almost vanishes. When $\vartheta = \vartheta^{max}$ and the throughput curve has no flatness, there is only one configuration returning the required value of the throughput. Hence, we expect that, for values of throughput approaching the maximum, there is no difference between minimizing s or b.

Table 1 reports the exact integer solution obtained running the IP model (p_{IP}) for each of the 10 replications and the approximate solutions obtained running the LP models for

Fig. 5 Results of ES#1 (exponential distribution)



(a) Average throughput vs number of pallets



(b) Approximate solutions obtained

optimization ($\tilde{p}_s$ being the result of the minimization of $\sum_k s_k$, while $\tilde{p}_b$ refers to the minimization of $\sum_k b_k$). The target throughput and the throughput obtained from the simulation are reported as well (ϑ^*, ϑ^s and ϑ^b, respectively). The approximate solution obtained minimizing **s** is always extremely close to the exact one. Moreover, as the required throughput approaches the maximum, the approximate solutions coming from the minimization of **b** are remarkably close to the exact optimum as well.

Table 4 reports the results obtained applying the stochastic programming approach (refer to the set of rows ES#1). Also in this case, the solution is close to the exact one, showing the robustness of the approach.

For the case $\vartheta^* = 0.063$, the value of ϑ_b (the throughput obtained from the simulation model when solved using the optimal time buffer solution $\mathbf{b}^*$) is lower than ϑ^*, i.e., it does not satisfy the throughput constraint. This cannot happen in the case simulation and optimization work on the same sample path. However, in the case of stochastic programming, the optimization model (returning $\mathbf{b}^*$) and the simulation model (returning ϑ_b) work on different samples and this explains the obtained result.

7.3 Experiment set #2

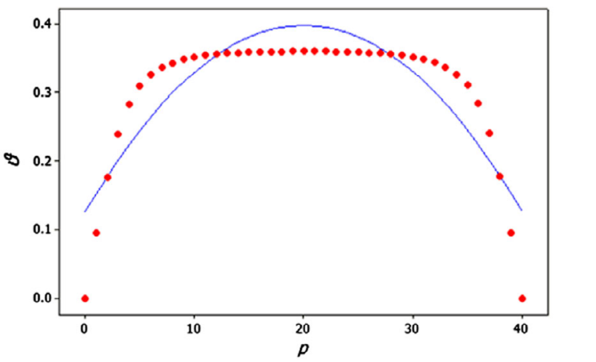
The second experiment set, as the first one, was designed with $J = 4$ machines. The buffers capacities were set to $c_1 = 15, c_2 = 5$ and $c_3 = c_4 = 10$. The processing times were gener-

Table 1 Approximate optimization results with multiple replication—experiment set ES#1

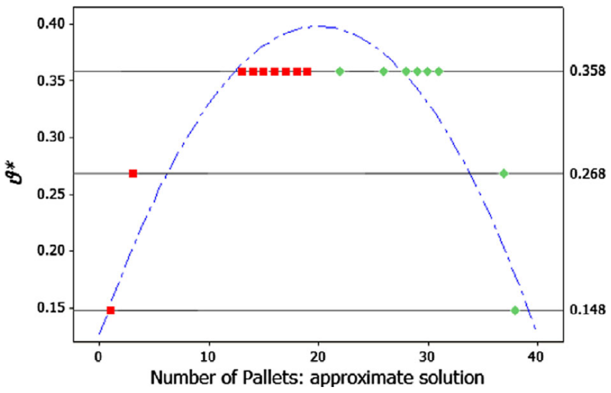
Run	ϑ	p_{IP}	$\tilde{p_s}$	ϑ_s	$\tilde{p_b}$	ϑ_b
1	0.063	2	2	0.063	17	0.063
2	0.063	2	2	0.063	17	0.063
3	0.063	2	2	0.063	17	0.063
4	0.063	2	2	0.063	17	0.063
5	0.063	2	2	0.063	17	0.063
6	0.063	2	2	0.063	17	0.063
7	0.063	2	2	0.063	17	0.063
8	0.063	2	2	0.063	17	0.063
9	0.063	2	2	0.063	17	0.063
10	0.063	2	2	0.063	17	0.063
1	0.106	7	7	0.107	12	0.106
2	0.106	6	6	0.106	11	0.106
3	0.106	6	6	0.106	13	0.106
4	0.106	7	7	0.106	12	0.106
5	0.106	7	7	0.106	12	0.106
6	0.106	6	7	0.107	13	0.106
7	0.106	7	7	0.106	13	0.106
8	0.106	6	7	0.106	12	0.106
9	0.106	6	6	0.107	12	0.106
10	0.106	7	8	0.107	12	0.106
1	0.114	9	10	0.114	9	0.114
2	0.114	8	8	0.114	10	0.114
3	0.114	8	9	0.114	10	0.114
4	0.114	9	10	0.114	10	0.114
5	0.114	9	9	0.114	10	0.114
6	0.114	9	9	0.114	11	0.114
7	0.114	9	9	0.114	11	0.114
8	0.114	8	9	0.114	10	0.114
9	0.114	8	8	0.114	10	0.114
10	0.114	9	9	0.114	8	0.114

ated from a uniform distribution for every machine ($\tau_1 = 0.5$, $\tau_2 = 1.5$, being τ_1 the mean and τ_2 the coefficient of variation).

Exact simulation results The system was studied solving the exact simulation model for each pallet configuration. Figure 6(a) shows the mean throughput together with the quadratic fit. The half width related to the 95 % confidence interval for the average throughput was estimated to be less than 0.0157. The throughput curve, in this case, is flat and multiple *equivalent* solutions (characterized by the same throughput) are present. In particular, solu-tions with $p = (19, 20, 21, 22)$ pallets give the same average throughput.



(a) Average throughput vs number of pallets



(b) Approximate solutions obtained

Fig. 6 Results of ES#2 (uniform distribution)

Simulation–optimization results Three target throughput levels were tested: $\vartheta^* = (0.148, 0.268, 0.358)$. Figure 6(b) reports the results obtained applying the multiple replication approach.

Also in this case $\tilde{p}_s$ and $\tilde{p}_b$ are symmetric and this result confirms Proposition 2.

Table 2 reports the exact integer solution obtained running the IP model for each of the 10 replications and those obtained from the LP models. As in ES#1, the approximate solution obtained minimizing **s** is always extremely close to the exact one. Moreover, as the required throughput approaches the maximum, the approximate solutions coming from the minimization of **b** are closer to the exact optimum as well. However, because of the curve flatness, the phenomenon is less evident than in ES#1.

Table 4 (block ES#2) details the results from the stochastic programming approach. It is worth to comment the results obtained for $\vartheta^* = 0.358$. In the third row, the solution $\tilde{p}_s = 25$ is "far" from the $p_{IP} = 16$. This is justified by the fact that this test case was characterized by a throughput curve particularly flat. In cases the throughput has this behaviour, the time approximation is always less effective.

7.4 Experiment set #3

The third test case is identical to the first experiment set, but the processing times were generated from a uniform distribution ($\tau_1 = 6, \tau_2 = 0.175$ for $j = 1, 2, 4$ and $\tau_1 = 7, \tau_2 =$

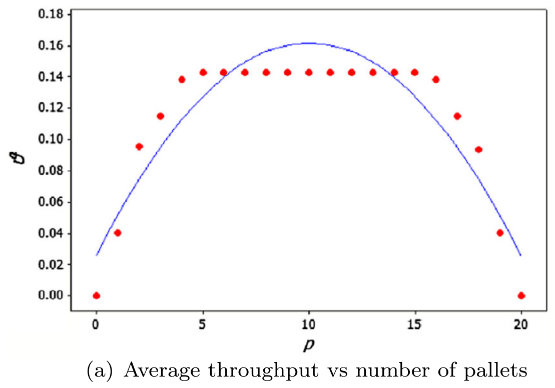
Table 2 Approximate optimization results with multiple replication—experiment set ES#2

Run	ϑ	p_{IP}	$\tilde{p_s}$	ϑ_s	$\tilde{p_b}$	ϑ_b
1	0.148	2	1	0.148	38	0.148
2	0.148	2	1	0.148	38	0.148
3	0.148	2	1	0.149	38	0.148
4	0.148	2	1	0.148	38	0.148
5	0.148	2	1	0.148	38	0.148
6	0.148	2	1	0.148	38	0.148
7	0.148	2	1	0.148	38	0.148
8	0.148	2	1	0.148	38	0.148
9	0.148	2	1	0.148	38	0.148
10	0.148	2	1	0.148	38	0.148
1	0.268	4	3	0.268	37	0.268
2	0.268	4	3	0.269	37	0.268
3	0.268	4	3	0.269	37	0.268
4	0.268	4	3	0.268	37	0.268
5	0.268	4	3	0.268	37	0.268
6	0.268	4	3	0.269	37	0.268
7	0.268	4	3	0.268	37	0.268
8	0.268	4	3	0.268	37	0.268
9	0.268	4	3	0.268	37	0.268
10	0.268	4	3	0.268	37	0.268
1	0.358	15	15	0.358	22	0.358
2	0.358	15	19	0.359	22	0.358
3	0.358	14	16	0.359	26	0.358
4	0.358	16	17	0.358	31	0.358
5	0.358	13	13	0.358	31	0.358
6	0.358	13	14	0.360	30	0.358
7	0.358	18	18	0.358	29	0.358
8	0.358	13	16	0.359	30	0.358
9	0.358	13	13	0.359	28	0.358
10	0.358	14	15	0.360	29	0.358

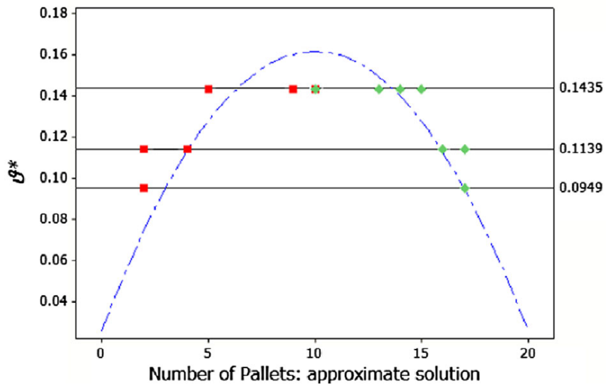
0.175 for $j = 3$, being τ_1 is the mean and τ_2 the coefficient of variation). Notice that the means of the processing times are exactly equal to those in ES#1.

Exact simulation results The system was studied solving the exact simulation model for each pallet configuration. Numerical results are presented in Fig. 7(a), showing the mean throughput and the quadratic fit. The half width related to the 95 % confidence interval for the average throughput was estimated to be less than 0.01. Also in this case, as in the second experiment set, the curve is flat. High throughputs are obtained in the range $p = (6, \dots, 15)$.

Fig. 7 Results of ES#1 (uniform distribution)



(a) Average throughput vs number of pallets



(b) Approximate solutions obtained

Simulation–optimization results Three target throughput levels were tested: $\vartheta^* = (0.095, 0.114, 0.143)$. Results of the multiple replication are reported in Fig. 7(b). The approximate solutions confirm the considerations already made for the other experimental settings.

Table 3 reports the exact integer solution obtained running the IP model for each of the 10 replications together with the approximate solutions obtained from the LP models. The approximate and the exact solutions are again very close when the approximate model minimizes $\sum_k s_k$. In case of minimization of $\sum_k b_k$, instead, a (near) maximum throughput is required to have such closeness between approximate and exact solution.

Also in this case, Table 4 (block ES#3) shows that the solution obtained minimizing $\mathbf{s}$ is close to the exact one.

8 Conclusion

In this paper we extended the simulation–optimization approach based on mathematical programming to closed queueing networks. The fundamental difference from previous works is that two sets of continuous decision variables are needed to represent the customer blocking behaviour.

With respect to the newly introduced decision variables, the models were analysed and different possible objective functions were considered. The effects of the different objective functions on the optimal continuous and integer solutions were studied.

Table 3 Approximate optimization results with multiple replication—experiment set ES#3

Run	ϑ	p_{IP}	$\tilde{p_s}$	ϑ_s	$\tilde{p_b}$	ϑ_b
1	0.095	2	2	0.095	17	0.095
2	0.095	2	2	0.095	17	0.095
3	0.095	2	2	0.095	17	0.095
4	0.095	2	2	0.095	17	0.095
5	0.095	2	2	0.095	17	0.095
6	0.095	2	2	0.095	17	0.095
7	0.095	2	2	0.095	17	0.095
8	0.095	2	2	0.095	17	0.095
9	0.095	2	2	0.095	17	0.095
10	0.095	2	2	0.095	17	0.095
1	0.114	3	2	0.114	17	0.114
2	0.114	3	2	0.114	16	0.114
3	0.114	3	4	0.114	16	0.114
4	0.114	3	2	0.114	17	0.114
5	0.114	3	2	0.114	17	0.114
6	0.114	3	2	0.114	17	0.114
7	0.114	3	2	0.114	16	0.114
8	0.114	3	2	0.114	17	0.114
9	0.114	3	2	0.114	16	0.114
10	0.114	3	2	0.114	17	0.114
1	0.143	5	5	0.143	15	0.144
2	0.143	9	10	0.143	13	0.167
3	0.143	9	10	0.143	14	0.143
4	0.143	10	10	0.143	13	0.143
5	0.143	10	9	0.143	10	0.143
6	0.143	10	9	0.143	15	0.143
7	0.143	10	9	0.143	14	0.143
8	0.143	10	10	0.143	13	0.143
9	0.143	10	10	0.143	14	0.143
10	0.143	10	10	0.143	14	0.143

The main drawbacks related to the multiple replications optimization approach were highlighted and modifications to the developed models were proposed leading to a two-stage stochastic programming approach. In both cases (multiple replication and stochastic programming), the models for optimization and for simulation were integrated to define the algorithm of simulation–optimization for loop systems.

Future research will be devoted to investigate the formal relationships between the approximate and the exact solutions and to develop different decomposition techniques to reduce the computational effort needed to solve the LP models, thus enhancing the possibility to efficiently tackle larger instances. Moreover, the solution of the single loop PAP that we propose represents a first step towards the solution of the multiple–loop case.

Table 4 Approximate optimization results with stochastic programming

Experiment Set	ϑ^*	p_{IP}	$\tilde{p_s}$	ϑ_s	$\tilde{p_b}$	ϑ_b
ES#1	0.063	2	2	0.065	17	0.056
	0.106	7	7	0.109	11	0.11
	0.114	11	10	0.132	11	0.114
ES#2	0.148	2	1	0.151	37	0.148
	0.268	4	3	0.268	35	0.270
	0.358	16	25	0.360	28	0.359
ES#3	0.094	2	1	0.095	17	0.094
	0.114	3	2	0.114	17	0.114
	0.143	10	9	0.143	14	0.143

References

Alfieri, A., & Matta, A. (2012). Mathematical programming formulations for approximate simulation of multistage production systems. *European Journal of Operational Research*. doi:10.1016/j.ejor.2011.12.044.

Birge, J., & Louveaux, F. (1997). *Introduction to stochastic programming*. *Springer series in operations research and financial engineering series*. Berlin: Springer.

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

Bouhchouch, A., Frein, Y., & Dallery, Y. (1993). Analysis of a closed-loop manufacturing system with finite buffers. *Applied Stochastic Models and Data Analysis*, *9*(2), 111–125. doi:10.1002/asm.3150090205

Brodsky, A., Pedersen, J., & Wagner, A. (2003). On the complexity of buffer allocation in message passing systems. *Journal of Parallel and Distributed Computing*, *65*(6), 692–713.

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

Chan, W. K., & Schruben, L. W. (2008b). Mathematical programming models of closed tandem queueing networks. *ACM Transactions on Modeling and Computer Simulation*, *19*(1). doi:10.1016/j.jpdc.2004.10.009.

Chick, S., Schmeiser, B., Sánchez, P. J., Ferrin, D., Morrice, D. J., & Jin, J. (2003). Simulation-based retrospective optimization of stochastic systems: a family of algorithms. In *Proceedings of the 2003 winter: Vol. 1. Simulation conference, 2003* (pp. 543–547).

Dallery, Y., & Frein, Y. (1989). A decomposition method for approximate analysis of closed queueing networks with blocking. In *Queueing networks with blocking* (pp. 193–216).

Dallery, Y., & Liberopoulos, G. (2000). Extended kanban control system: combining kanban and base stock. *IIE Transactions*, *32*, 369–386.

Dolgui, A., Eremeev, A., & Sygaev, V. (2010). A problem of buffer allocation in production lines: complexity analysis and algorithms. In *3rd international conference on metaheuristics and nature inspired computing*, Djerba, Tunisia.

Fu, M., Glover, F., & April, J. (2005). In *Simulation optimization: a review, new developments, and applications* (pp. 83–95).

Garey, M., & Johnson, D. (1979). *Computers and intractability: a guide to the theory of NP-completeness*. New York: Freeman

Gershwin, S. B., & Werner, L. (2007). An approximate analytical method for evaluating the performance of closed-loop flow systems with unreliable machines and finite buffers. *International Journal of Production Research*, *45*, 3085–3111.

Healy, K., & Schruben, L. W. (1991). Retrospective simulation response optimization. In *Proceedings of the 23rd conference on winter simulation*, WSC '91 (pp. 901–906). Washington: IEEE Computer Society.

Higle, J., & Sen, S. (1996). *Stochastic decomposition: a statistical method for large scale stochastic linear programming*. *Nonconvex optimization and its applications*. Norwell: Kluwer Academic.

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

Kleijnen, J. P. C. (2008). *Design and analysis of simulation experiments*. *International series in operations research & management science: Vol. 111*. Berlin: Springer.

Kushner, H., & Yin, G. (1997). *Stochastic approximation algorithms and applications*. Berlin: Springer.

Law, A. (2007). *Simulation modeling and analysis* (4th ed.). New York: McGraw-Hill.

Maggio, N., Matta, A., Gershwin, S., & Tolio, T. (2009a). A decomposition approximation for three-machine closed-loop production systems with unreliable machines, finite buffers and a fixed populatio. *IIE Transactions*, *41*(6), 562–574.

Maggio, N., Matta, A., Gershwin, S. B., & Tolio, T. (2009b). A decomposition approximation for three-machine closed-loop production systems with unreliable machines, finite buffers and a fixed population. *IIE Transactions*, *41*(6), 562–574.

Matta, A. (2008). Simulation optimization with mathematical programming representation of discrete event systems. In S. J. Mason, R. R. Hill, L. Monch, O. Rose, T. Jefferson, & J. W. Fowler (Eds.), *Proceedings of the 2008 winter simulation conference* Piscataway (pp. 1393–1400). New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Matta, A., & Chefson, R. (2005). Formal properties of closed flow lines with limited buffer capacities and random processing times. In *Proceedings of the European simulation and modelling conference*, Porto, Portugal (pp. 190–194).

Ming-Guang, H., Pao-Long, C., & Ying-Chyi, C. (2002). Buffer allocation in flow-shop-type production systems with general arrival and service patterns. *Computers & Operations Research*, *29*(2), 103–121.

Montgomery, D. (2005). *Progettazione e analisi degli esperimenti. Istruzione scientifica*. New York: McGraw-Hill.

Myers, R., Montgomery, D., & Anderson-Cook, C. (2009). *Response surface methodology: process and product optimization using designed experiments*. *Wiley series in probability and statistics*. New York: Wiley.

Pedrielli, G. (2013). *Discrete event systems simulation–optimization: time buffer framework*. PhD thesis, Mechanical Engineering Department, Politecnico di Milano, Italy.

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

Schruben, L. W. (2000). Mathematical programming models of discrete event system dynamics. In J. A. Joines, R. R. Bartona, K. Kang, & P. A. Fishwick (Eds.), *Proceedings of the 2000 winter simulation conference* Piscataway (pp. 381–385). New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Spall, J. C. (2003). *Introduction to stochastic search and optimization*. New York: Wiley.

Zhang, Z. (2006). *Analysis and design of manufacturing systems with multiple-loop structures*. PhD thesis, Mechanical Engineering Department, Massachussetts Institute of Technology, Berkeley.