Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

SIMULATION OPTIMIZATION WITH MATHEMATICAL PROGRAMMING REPRESENTATION OF DISCRETE EVENT SYSTEMS

Andrea Matta

Dipartimento di Meccanica, Politecnico di Milano
Via La Masa 1
20156 Milano, Italy

ABSTRACT

Optimization–via–simulation consists in applying iteratively two detached models until an optimality condition is reached: a simulation model for predicting the system performance, and a model for generating potential optimal solutions. Mathematical programming representation has been recently used to describe the behavior of discrete event systems as well as their formal properties. This paper proposes explicit mathematical programming representations for jointly simulating and optimizing discrete event systems. The main advantage of such models is the rapidity of searching for the optimal solution, given to the explicit knowledge of objective function and constraints. Three types of formulations are proposed for solving the buffer allocation problem in flow lines with finite buffer capacities: an exact mixed integer linear model, an approximate LP model and a stochastic programming model. Numerical analysis shows that the computational time required to solve resource allocation problems can be significantly reduced by using the proposed formulations.

1 INTRODUCTION

Simulation is one of the most popular techniques to study the behavior of Discrete Event Systems (DES). Discrete event simulation is widely used to analyze the detailed behavior of manufacturing systems, logistic systems, health care systems etc, for estimating their major performance measures such as throughput, flow times, resource utilizations, etc (Law 2007). In particular, simulation is used in all those situations in which it is not possible to define analytical mathematical expressions for describing the system behavior, because of the high related complexity (e.g. number of components and interactions in the system) and the different sources of randomness that characterize most of systems in reality. The main characteristic of simulation is the possibility of predicting the system performance in an implicit way, without forcing the analyzer to define complex mathematical equations modeling the system.

An alternative way of modeling DES has been proposed by Schruben (2000). A DES can be mapped into a mathematical programming formulation where the optimal solution represents the trajectory of the discrete event system. In particular, the dynamic behavior of the studied system is represented by an optimization model in which the sum of finishing and starting activity events is minimized constrained to the linear routing of customers flowing into the system, and to system constraints such as limited buffer capacities, maximum sojourn times etc. The optimization problem is linear and it also has a corresponding dual problem, which in turn is mapped into an oriented graph where the nodes are the activity events and the edges are the time intervals between two possible events (Chan and Schruben 2003). Optimizing the flow in the oriented graph corresponds to solve the dual problem and, as a consequence, to find the system trajectory during a defined time period. In addition, this graph has the nice property that its set of edges represents the feasible area of the primal problem. Therefore, dealing with edges corresponds to dealing with the set of all possible system trajectories. Mathematical programming representation (MPR) of DES can also be view as a max–plus–type representation (Baccelli et al. 1992, Chan 2005).

This alternative way of explicitly representing DES can be exploited for deriving structural properties of the studied system (Chan and Schruben 2003, Chan 2005, Matta and Chefson 2005) and for optimization purposes (Chan and Schruben 2006). This paper deals with the second issue. In particular, the goal is to discuss how the explicit mathematical programming representation of DES can be used to allocate resources. Different formulations of optimization models are presented and discussed in this work. All models present the original feature that they act both as performance evaluation and optimization models simultaneously. In current practice, optimization and performance evaluation models are generally decoupled in optimization

978-1-4244-2708-6/08/$25.00 ©2008 IEEE

for simulation (Fu 2002). Indeed, a simulation model is typically a computer code used for predicting the performance of the system with a certain configuration, the optimization model is an algorithm on the top of simulation that searches for the best configuration according to some defined criteria (see also Figure 1). This paper proposes an MPR of DES that can be used for optimization while some performance measures are contemporarily calculated. Entering into the *black box* of the simulation model for optimization purposes represents the novelty of this paper, together with positioning MPR of DES into the more general stochastic programming technique. Production flow lines are the system taken as reference in this study; the classical buffer allocation problem is considered and a set of different mathematical programming formulations is proposed for simultaneous simulation and optimization.

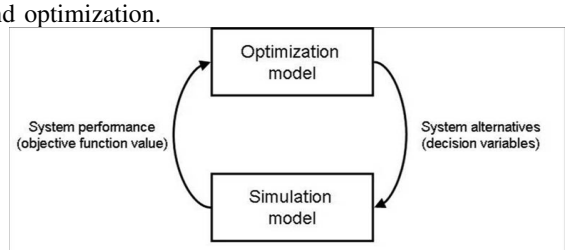

Figure 1: Optimization for simulation.

The paper is organized as follows. Mathematical programming models for simulating production flow lines are described in the next section. Integrated simulation and optimization models for deciding the amount of buffer space to distribute between machines are presented in section 3. Section 4 reports the application of MPR of DES to some test cases. Finally conclusions are drawn in the last section.

2 ANALYSIS OF OPEN FLOW LINES

2.1 Assumptions

Open flow lines are composed of a certain number of machines separated by intermediate buffers with limited capacity. In this paper machines are assumed to be perfectly reliable and characterized by random processing times generally distributed. The sequencing of parts is fixed and known a priori. The generic part i (with $i = 1,\ldots,N$) arrives at the system at time A_i and is processed sequentially from the first machine to the last one. The part waits in the buffer B_{j-1} if machine M_j is busy because processing another part k (with $k < i$). After having been processed by the first machine, parts go to the second machine and so forth until the last operation is performed at the last machine; finally parts leave the system. Machines and buffers are denoted with the notation M_j and B_j respectively, with $j = 1,\ldots,K-1,K$; each buffer B_j, located immediately downstream machine M_j, has a finite capacity C_j (with $j = 1,\ldots,K-1$). Transportation times are considered negligible or already included in machining times. Finally, the blocking before service control rule is assumed for machines (Dallery and Gershwin 1992). For sake of simplicity the last machine is never blocked, thus parts completing the service at the last machine can always leave the system. The production rate or throughput of the line is defined as the number of parts produced in a time interval, thus its expectation is:

$$E[P] = \lim_{t\to\infty} \frac{N_t}{t} \tag{1}$$

where N_t is the number of parts produced after a period of length t.

2.2 Performance evaluation model

A linear programming (LP) model is now described to simulate a generic open flow line with K machines separated by buffers with finite capacity. This simplified version of LP model can be obtained from Chan and Schruben (2003):

$$\min_{F} \quad \sum_{i=1}^{N}\sum_{j=1}^{K} F_{i,j} \tag{2}$$

subject to:

$$F_{i,1} \geq A_i + t_{i,1} \qquad \forall i \tag{3}$$

$$F_{i,j+1} - F_{i,j} \geq t_{i,j+1} \qquad \forall i; j = 1,\ldots,K-1 \tag{4}$$

$$F_{i+1,j} - F_{i,j} \geq t_{i+1,j} \qquad i = 1,\ldots,N-1; \forall j \tag{5}$$

$$F_{i+C_j,j} - F_{i,j+1} \geq t_{i+C_j,j} \qquad i = 1,\ldots,N-C_j \quad j = 1,\ldots,K-1 \tag{6}$$

$$F_{i,j} \geq 0 \qquad \forall i,j \tag{7}$$

where $t_{i,j}$ and $F_{i,j}$ are the processing and finishing time of part i at machine j respectively. Constraints (3) simply impose that the service at the first machine cannot start before the arrival time of the same part at the system plus its first processing time. Constraints (4) state that a part cannot be contemporary processed by two different machines at the same time. Constraints (5) mean that a machine cannot process two different parts at the same time. Constraints (6) impose that a part cannot leave a machine if the immediate downstream buffer is full. Finally finishing times must be nonnegative; this constraint is redundant if all arrival times are nonnegative.

The solution of the linear problem provides the optimal values for decision variables F. The problem solution corresponds to the dynamic behavior of the DES, i.e. the optimal values are exactly the finishing time events of machining operations in a real or simulated system that has the same ordering of parts, the same arrival events and the same processing times. See again the paper of Chan and

Schruben (2003) for more details. If constraints (6) are not present, the model represents the behavior of a flow line with infinite buffers among machines. Matta and Chefson (2005) consider closed flow lines by adding two sets of constraints modeling the fact that the number of parts in the system is always constant. When arrival and processing times are sampled from known statistical distributions, the output of the LP model corresponds to a replication of a simulation model representing the same flow line. The average throughput of the flow line can be estimated from the optimal solution:

$$\hat{P} = \frac{N}{F_{N,K}} \tag{8}$$

where $F_{N,K}$ is the finishing time of the last part at the last machine in the simulated sample path.

MPR models present both advantages and drawbacks when compared to traditional computer codes of discrete event simulation models:

- *Model development*. It is more straightforward to write equations and to solve them with any available solver than developing a simulation model in a computer code.
- *Computational time*. LP uses simplex based techniques to find out the optimal solution, this is faster than the large number of mathematical operations simulation generally performs. Note that efficient techniques can be used because the decision variables F are defined in the continuous domain. This does not hold for more complex systems in which the sequencing of parts is not fixed, or the routing of parts is dynamically dependent on the system conditions, etc. Indeed, for these more complex systems it is necessary to use integer variables in addition to continuous F, thus hardly increasing the problem complexity. For this reason the modeling power of MPR is quite restricted to a certain class of DES.
- *Formal properties*. Thanks to the explicit mathematical formulation of the system model, it is possible to derive structural properties of the analyzed system such as monotonicity, concavity etc of major system performance (Chan 2005). Also deadlocks can be easily identified (Matta and Chefson 2005); indeed, if a system deadlock will occur during simulation, the set of the feasible area of the LP model is empty or the objective function is unbounded. This can be checked by using standard algorithms for feasibility checking of LP models.
- *Sensitivity analysis*. Perturbation analysis (PA) indicators can be calculated by using classical sensitivity analysis of linear programming (Chan and Schruben 2006, Zhang and Chan 2007).

3 BUFFER ALLOCATION: MODELS FORMULATION

The buffer allocation problem, i.e. deciding the distribution of buffer space among the machines of the line, is a well known problem both in industrial research and practice. Depending on the type of the goal pursued during the optimization, there are two types of buffer allocation problems. In the *primal problem* the total cost of the allocated buffer capacity is minimized constrained to a minimum value of expected throughput of the line P^*:

$$\begin{aligned} \min_{C\in\Omega} \quad & \mathbf{a^T}\cdot\mathbf{C} \\ s.t. \quad & E[P] \geq P^* \end{aligned}$$

where P is the system throughput, Ω is a finite set of $\mathbb{R}^{K-1}$ with finite coordinates and $\mathbf{a}$ is a cost vector. On the other hand, the average system throughput is maximized constrained to a maximum budget available for the buffer allocation in the *dual problem*:

$$\begin{aligned} \max_{C\in\Omega} \quad & E[P] \\ s.t. \quad & \mathbf{a^T}\cdot\mathbf{C} \leq a^* \end{aligned}$$

where a^* is the available budget. See (Gershwin and Schor 2000) for a complete description of the buffer allocation problem and the related literature. In the remainder of this paper different types of MPR are proposed for solving the two types of buffer allocation problems.

3.1 Primal

3.1.1 MILP formulation

The primal buffer allocation problem can be formulated in a MILP model that integrates both performance evaluation and optimization:

$$\min_{x,F} \quad \sum_{j=1}^{K-1} a_j \cdot \sum_{k=L_j}^{U_j} x_{j,k}\cdot k$$

subject to:

$$\begin{aligned} F_{i,1} &\geq A_i + t_{i,1} && \forall i \\ F_{i+1,j} - F_{i,j} &\geq t_{i+1,j} && i = 1,\ldots,N-1 \\ & && \forall j \\ F_{i,j+1} - F_{i,j} &\geq t_{i,j+1} && \forall i \\ & && j = 1,\ldots,K-1 \\ F_{i+C_j,j} - F_{i,j+1} &\geq t_{i+C_j,j}x_{j,k} - (1-x_{j,k})M && i = 1,\ldots,N-k_j \end{aligned}$$

$$F_{i,j} - F_{i-1,j} \geq p_{i,j} \qquad j = 1,\ldots,K-1 \quad (9)$$
$$\forall k$$

$$\sum_{k=L_j}^{U_j} x_{j,k} = 1 \qquad \forall j \quad (10)$$

$$F_N - F_d \leq T^* \quad (11)$$

$$F_{i,j} \geq 0 \qquad \forall i,j$$

$$x_{j,k} \in \{0,1\} \qquad \forall j,k \quad (12)$$

where $x_{j,k}$ is a binary variable equal to one if a capacity k (with $k = L_j,\ldots,U_j$) is assigned to buffer B_j; the values L_j and U_j are the bounds defined by the analyst of the optimization problem for the $j-th$ buffer. The constraint (11) imposes that the throughput must be greater or equal to a minimum value, this implies defining an inequality between the time necessary to produce $N-d$ pieces and the available time T^*. The parameter T^* is defined by the user when he decides the minimum value of throughput mean to be satisfied. When $x_{j,k} = 1$ all the constraints related to buffer B_j with an assigned capacity equal to k are activated; otherwise the constraint is made redundant by subtracting from the right–hand side a large value M. The index d represents the end of the system warm–up, identifiable with well-known techniques (Law 2007). Finally, only one capacity k must be chosen for each buffer B_j as imposed by equation (10). Parameter k_j in constraint (9) is the capacity of buffer B_j when a capacity k (with $k = L_j,\ldots,U_j$) is selected; this superior limit to the definition of constraints is due to the fact that the last stream of pieces of the sample path cannot be cause of blocking because simulation simply ends after the $N-th$ unit leaves the system.

This MILP model has the characteristic to behave both as a performance evaluation model, by estimating the maximum flowtime of the system with a given buffer allocation, and as an optimization model, by choosing the right space distribution among machines. This is a special issue, since it allows having an intrinsic integration between simulation and optimization models, which are generally separated. More specifically, the model is able to check the feasibility of the throughput constraint during the optimization process. Thus the optimal solution, if existing, provides the buffer allocation respecting the throughput constraint. Note that, differently from the model presented in the previous section, solution variables F do not correspond with the finishing times of the DES simulated with the same buffer allocation because some F tend to assume large values to respect constraints (9). However the $N-d$ parts in the sample path will be surely produced in a time inferior to T^* because of the presence of constraint (11) that guarantees the minimum value of throughput mean decided by the user. Thus the correctness of the result is always preserved. To obtain the system performance with the identified optimal buffer allocation, i.e. to calculate the correct F values according to the optimal buffer allocation, it is necessary to solve the performance evaluation model or to run any other valid simulation model. It is important to mention that the solution of this problem is optimal only for the particular generated sample path and not in general for all possible system trajectories.

Optimizing buffers with this proposed model corresponds to apply sample–path optimization technique with only one simulation replication (Robinson 1996, Gurkan et al. 1994, Fu et al. 2005). Indeed, a sample path is randomly generated and a deterministic optimization problem is obtained and solved. The main advantage in this case is that the faced problem is a standard MILP problem that can be tackled with the several methods and heuristics developed in the last fifty years.

Branch and bound algorithms can be used to solve the above MILP model. However solving this model is not an easy task because of constraints (9) that put together continuous F and binary x variables. Most of branch and bound method uses an LP relaxation of the model, allowing decision variables x to take values from the continuous domain [0,1], to estimate a lower bound of the objective function value. In this case it happens that the lower bound is very weak due to the structure of constraints (9), as a consequence the integrality gap is large and the required computational time to solve the MILP problem is relevant.

The solution of the LP relaxation could also be useful to understand the relative benefits of adding space to a specific buffer. Indeed, due to the specific structure of constraints (9) of the problem, all $x_{j,k}$ in the optimal solution assume null values except for $k = L_j, L_j+1$; in particular, for $k = L_j$ $x_{j,k}$ is close to 1, while its complement to the unit is allocated to $k = L_j+1$. The value of x_{j,L_j+1} is related to the difficulty of respecting the finite capacity constraint, the larger the value of x_{j,L_j+1} is and the more the space to be allocated at buffer B_j must be. For this reason the x_{j,L_j+1} values can be interpreted as a sort of indicators that are proportional to the gradient of the objective function. Thus a possible, and also very informal, usage of LP relaxed formulation, in addition to the classical branch & bound, is in combination with gradient–based optimization heuristics, e.g. Hill Climbing algorithm (Pichitlamken and Nelson 2003).

3.1.2 LP approximate formulation

Constraints (9) of the MILP model are active when $x_{j,k} = 1$ and redundant otherwise. An alternative way of deactivating these constraints is to define a continuous nonnegative surplus time variable s that can make redundant the constraints when necessary. Thus if the weighted sum of these surplus variables is minimized, a very fast, but approximate, solution of the buffer allocation problem can be obtained. More specifically, $s_{j,k}$ is null if it is never necessary to use the surplus variable to deactivate constraints related to buffer

B_j with capacity k, and is positive otherwise. The larger the value of $s_{j,k}$ is and the higher the importance of having the capacity k at buffer B_j will be. This simplified model differs from the MILP model in the objective function:

$$\min_{s,F} \sum_{j=1}^{K-1} a_j \cdot \sum_{k=L_j}^{U_j} s_{j,k} \cdot w_k$$

where w_k are simple weights, and constraints (9) that become:

$$F_{i+C_j,j} - F_{i,j+1} \geq t_{i+C_j,j} - s_{j,k} \quad \begin{array}{l} i = 1,\ldots,N-k_j \\ j = 1,\ldots,K-1; \quad \forall k \end{array}$$

Constraints (9) and (11) hold also in this new formulation while constraint (10) cannot be used since decision variable s are continuous. Weights w_k can be chosen so that large buffers are penalized; the relationship of weights can be linear $w_k = k$, or quadratic $w_k = k^2$, or it can assume any other reasonable form. If $s_{j,k}$ is positive, it means that at least one constraint corresponding to buffer B_j with capacity k has been deactivated, i.e. the amount k of buffer capacity is necessary. The larger the decision variable $s_{j,k}$ is and the higher the importance of having a buffer of capacity k at B_j will be. The selected buffer quantity for each buffer B_j corresponds to the largest index k for which the time variable $s_{j,k}$ is positive; note that the solution of the model provides *real* values for the surplus time variables s, then the *integer* solution of the buffer allocation is extrapolated with the heuristic rule just described. Since this problem can be solved with very low computational efforts, its solution can be used as a starting point in local search algorithms.

3.1.3 Stochastic programming formulation

MPR of DES can be view as a particular case of stochastic programming. Indeed, if arrival times and processing times are considered as samples from known statistical distributions, the LP performance evaluation model of the flow line can be formulated as follows:

$$\min_{F} \quad \sum_{i=1}^{N} \sum_{j=1}^{K} F_{i,j}$$

subject to:

$$\begin{array}{rl} F_{i,1} \geq A_i(\omega) + t_{i,1}(\omega) & i = 1,\ldots,N \\ F_{i,j+1} - F_{i,j} \geq t_{i,j+1}(\omega) & \forall i \\ & j = 1,\ldots,K-1 \\ F_{i+1,j} - F_{i,j} \geq t_{i+1,j}(\omega) & i = 1,\ldots,N-1 \\ & \forall j \\ F_{i+C_j,j} - F_{i,j+1} \geq t_{i+C_j,j}(\omega) & i = 1,\ldots,N-C_j \\ & j = 1,\ldots,K-1 \\ F_{i,j} \geq 0 & \forall i,j \end{array}$$

where ω is the particular sample. In stochastic simulation the system performance is calculated for every generated sample ω^r, with $r = 1,\ldots,R$, where R is the number of replications. Thus, accordingly with the replication approach, R right-hand side vectors are sampled and R LP deterministic problems are solved; this corresponds to the *wait and see* approach of stochastic programming in which the analyzer waits for the manifestation of the uncertainty and then makes decisions (Birge and Louveaux 1997). The *expected value* solution, obtained solving the same problem with expected right-hand side vectors instead of sampled values, has no more sense in this case because it corresponds to evaluating the system with all activities being deterministic.

The MILP primal problem can be view as a particular case of a two–stage stochastic programming problem with recourse (Birge and Louveaux 1997), in which x and F are the first and second stage decision variables respectively and the sources of uncertainty are arrivals and processing times, both right-hand side values. Stochastic Decomposition (SD) techniques are generally used to solve two–stage stochastic programming models with recourse, however the described primal buffer allocation problem presents some characteristics that make it difficult to solve. The objective function depends only on the first stage variables, because the recourse function is missing. This makes the problem really hard, since the research of the solution in the first stage is "blind" and SD techniques could not perform well due to the low efficiency of generated cuts.

Another problem is related to the throughput constraint in inequality (11). Indeed, one way of dealing with this constraint is to impose that the inequality must be satisfied for every possible realization ω. However this approach could be too conservative and more realistic requirements should be identified. In these cases *chance constraints* are often used to ensure feasibility of the second stage problem with probability equal at least to $1-\varepsilon$:

$$Prob\{F_N - F_d \leq T^*\} \geq 1 - \varepsilon \tag{13}$$

where ε is a real small value. Nemirovski and Shapiro (2006) discuss this type of constraints and propose an algorithm to formulate and solve them efficiently.

3.2 Dual

3.2.1 MILP formulation

The MILP dual version of the buffer allocation problem is similar to that of the primal problem except for the objective function, in which the expected production rate is maximized, and the main system constraint, i.e. a limited

budget instead of a minimum throughput to deal with. Using equation (8), the objective function of the dual problem is:

$$\min_{x,F} \quad F_N \tag{14}$$

constrained to

$$\sum_{j=1}^{K-1} a_j \sum_{k=L_j}^{U_j} k \cdot x_{j,k} \leq a^* \tag{15}$$

in addition to constraints (9),(10) and (12). As for the primal problem, this formulation can be solved exactly by branch and bound algorithms. Again the integrality gap is large due to the weak lower bound obtained from the LP relaxation of the problem and for this reason the time to solve this problem can be very large for long lines.

3.2.2 LP approximate formulation

The dual problem can be approximately solved by using continuous time variables s to deactivate constraints following the same approach presented in section 3.1.2. In this case the sum of these time variables must be limited to a threshold value α; the larger α is and the higher the expected production rate will be. Thus, the objective function of the LP approximate dual problem is the same as in function (14) constrained to

$$\sum_{j=1}^{K-1} \sum_{k=L_j}^{U_j} s_{j,k} \leq \alpha \tag{16}$$

in addition to constraints (9) and (12). Given a value of α, the selected buffer quantity for each buffer B_j corresponds to the largest index k for which the time variable $s_{j,k}$ is positive. The value of α can be properly chosen so that the buffer capacity cost deriving from solving the approximate model coincides with the total available budget a^*. A simple algorithm is the bisection method that, starting from an upper vale for α, iteratively updates α until the total buffer capacity allocated with the approximate LP dual model is equal to a^*. This algorithm exploits the monotonicity property of the expected throughput as a function of the total buffer capacity (Buzacott and Shantikumar 1993). Indeed, according to this property the optimal solution of the dual problem must be a point on the border of the feasible set at which all the available budget is used. A way to initialize α is to assign the sum of the optimal values s of the approximate LP primal problem built by imposing a minimum expected throughput very close to the bottleneck's one; then the bisection algorithm is launched.

3.2.3 Stochastic programming formulation

The dual allocation problem can be formulated as a two–stage stochastic programming model in which the recourse function is the only term in the objective function to be minimized:

$$\min_{x,F} \quad E[F_N(\omega)]$$

The problem has complete recourse, i.e. the second stage problem has always a feasible solution. Thus complexities deriving from introducing chance constraints do not hold here, differently from the primal problem. The objective function is convex and SD algorithms can be efficiently applied to solve this type of problem (Higle and Sen 1996, Sen and Higle 2005). SD provides a piecewise linear approximation of the recourse function without the necessity of solving the second stage problem problems for all sampled scenarios ω^r, i.e. without simulating the flow line in all replications; this is a major issue that may lead to significant reductions of computational efforts compared to stochastic optimization problems (Zhao and Sen 2006).

4 NUMERICAL ANALYSIS

In this section the application of the proposed formulations is reported on two test cases. The first case is a flow line of three machines with random processing times exponentially distributed having rates equal to 7, 7 and 6. The boundaries of the problem are $L_j = 0$ and $U_j = 20$ for $\forall j = 1, \ldots, K-1$. For simplicity $\mathbf{a}$ is a unitary cost bi–dimensional vector and the first machine is assumed to be never starved, thus all arrival times are null. For the primal problem the requested minimum average throughput is 5.776; for the dual problem the available budget for buffer allocation is 20 and the relationship of weights is quadratic. This case has been faced by Pichitlamken and Nelson (2003) in a more general problem in which mean service rates of machines are optimized together with buffer capacities. The optimal solution of the dual problem for this system is $\mathbf{C} = (8, 12)$ with an expected throughput in steady state of 5.776. Tables 1 and 2 show the solutions of the exact and LP approximate formulations of the primal and dual problems respectively for different values of simulated parts machined by the system. It can be noticed that the approximate formulation provide near–optimal solutions in all analyzed cases. Computational time of the LP approximate model is of the order of minutes, much lower than that necessary for the MILP model (order of hours). The convergence of the solution as N increases is not treated in the experiments, since this issue is not faced in the paper. The convergence of stochastic optimization problems is discussed in several papers; see the work of Shapiro (1996) and the recent discussion of Birge (2007) with references therein. SD algorithm exploits the special structure

of the two–stage problem and finds out the exact solution in smaller time than classical branch and bound algorithms applied to the MILP problem. For instance, for solving the dual problem for $N = 10000$ the ILOG CPLEX solver needs approximately 5 hours while SD only 32 minutes for 10 scenarios, each one with $N = 3000$ (both experiments were carried out on a Core 2 Duo E6850 3.0Ghz/1333MHz/4MB). This CPU time reduction confirms the results of Zhao and Sen (2006), who compare stochastic programming technique with sample–path based simulation–optimization.

Table 1: Case 1: primal problem ($d = 2000, P^* = 5.776$).

	C	
N	exact	approximate
3000	(8,9)	(8,9)
4000	(8,10)	(10,11)
5000	(8,12)	(11,13)
6000	(8,14)	(10,11)
7000	(11,11)	(9,10)
8000	(9,12)	(10,11)
9000	(10,12)	(10,12)
10000	(10,12)	(10,12)
15000	(9,12)	(10,12)
20000	(9,12)	(10,12)
25000	(10,12)	(10,12)

Table 2: Case 1: dual problem ($d = 2000$, $a^* = 20$).

	exact		approximate	
N	**C**	$\hat{P}$	**C**	$\hat{P}$
3000	(9,11)	5.879	(9,11)	5.879
4000	(9,11)	5.883	(9,11)	5.883
5000	(8,12)	5.783	(9,11)	5.783
6000	(8,12)	5.731	(9,11)	5.728
7000	(8,12)	5.734	(9,11)	5.734
8000	(8,12)	5.755	(9,11)	5.755
9000	(9,11)	5.745	(9,11)	5.745
10000	(9,11)	5.761	(9,11)	5.761
15000	(9,11)	5.762	(9,11)	5.762
20000	(9,11)	5.753	(9,11)	5.753
25000	(9,11)	5.728	(9,11)	5.728

The second case is a flow line composed of five machines with random processing times lognormally distributed. The two parameters of the lognormally distributed processing times, i.e. the mean μ and standard deviation σ of the variable's natural logarithm, are both equal to 1 except for the first machine, which represents the bottleneck having $\mu = 1.5$ time units. The boundaries of the problem are $L_j = 0$ and $U_j = 20$ for $\forall j = 1, \ldots, K-1$. For simplicity **a** is a unitary cost 4–dimension vector. For the dual problem the available budget is 29 and the relationship of weights is quadratic. Table 3 shows the solutions of the exact and LP approximate formulations of the dual problem for some values of N. Again it can be noticed that the approximate formulation provide near–optimal solutions in the analyzed cases; the reduction of computational time in experimentation was around 95% in this case. Similar results have been obtained for many other cases that are not reported in this paper.

Table 3: Case 2: dual problem ($d = 2000$, $a^* = 29$).

	exact		approximate	
N	**C**	$\hat{P}$	**C**	$\hat{P}$
3000	(12,6,5,6)	0.1366	(10,8,6,5)	0.1359
4000	(12,6,7,4)	0.1343	(10,8,6,5)	0.1335
5000	(13,6,6,4)	0.1357	(10,8,6,5)	0.1338
6000	(13,6,5,5)	0.1319	(10,8,6,5)	0.1314
7000	(13,6,5,5)	0.1321	(10,8,6,5)	0.1317

5 CONCLUSIONS

This work proposes a set of different MPRs of DES that combine performance evaluation and optimization in a unique model. The LP approximate formulation can be used as a fast global search algorithm to rapidly identify a promising area in the solution space. The stochastic programming formulation has a special structure that can be exploited by SD algorithms to efficiently solve the sample–path optimization problem in a restricted area of the solution space. The proposed models have been applied for solving the buffer allocation problem in production lines, however they can be used to optimize other kinds of DES (e.g. kanban–based systems, base stock, conwip, assembly systems etc). Future work will be dedicated to develop a complete MPR–based algorithm for resource allocation of DES able to solve complex real cases.

ACKNOWLEDGMENTS

This work has been granted by the Mechanical Department of Politecnico di Milano. The author would like to thank prof. L.W. Schruben and prof. G. Shantikumar for their useful comments during his visit at the IEOR Department of University of California, Berkeley.

REFERENCES

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

Birge, J. 2007. Unattained convergenge for sampling methods in large-scale optimization models and a remedy

with batch means. In *INFORMS Simulation Society Research Workshop: Simulation for better decisions in an uncertain world*.

Birge, J., and F. Louveaux. 1997. *Introduction to stochastic programming*. Springer–Verlag.

Buzacott, J., and J. Shantikumar. 1993. *Stochastic models of manufacturing systems*. Prentice–Hall.

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

Chan, W. K., and L. W. Schruben. 2003. Properties of discrete event systems from their mathematical programming representations. In *Proceedings of the 2003 Winter Simulation Conference*, ed. S. Chick, P. J. Sanchez, D. Ferrin, and D. J. Morrice, 496–502: Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

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

Dallery, Y., and S. B. Gershwin. 1992. Manufacturing flow line systems: A review of models and analytical results. *Queueing Systems Theory and Applications, Special Issue on Queueing Models of Manufacturing Systems* 12 (1-2): 3–94.

Fu, M. 2002. Optimization for simulation: Theory vs. practice. *Journal on Computing* 14 (3): 192–215.

Fu, M., F. Glover, and J. April. 2005. Simulation optimization: A review, new developments, and applications. In *Proceedings of the 2005 Winter Simulation Conference*, ed. M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, 83–95: Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Gershwin, S. B., and J. Schor. 2000. Efficient algorithms for buffer space allocation. *Annals of Operational Research* 93:117–144.

Gurkan, G., A. Y. Ozge, and S. M. Robinson. 1994. Sample–path optimization in simulation. In *Proceedings of the 1994 Winter Simulation Conference*, ed. J. D. Tew, S. Manivannan, D. A. Sadowski, and A. F. Seila, 247–254: Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Helber, S., K. Schimmelpfeng, R. Stolletz, and S. Lagershausen. 2008. Using linear programming to analyze and optimize stochastic flow lines. Working paper.

Higle, J., and S. Sen. 1996. *Stochastic decomposition: a statistical method for large scale stochastic linear programming*. Kluwer Academic Publisher.

Law, A. 2007. *Simulation modeling and analysis*. 4th ed. McGraw-Hill.

Matta, A., and R. Chefson. 2005. Formal properties of closed flow lines with limited buffer capacities and random processing times. In *Proceedings of the European Simulation and Modelling Conference*, 190–194. Porto, Portugal.

Nemirovski, A., and A. Shapiro. 2006. *Scenario approximations of chance constraints*, Chapter I-3. Springer London.

Pichitlamken, J., and B. L. Nelson. 2003. A combined procedure for optimization via simulation. *ACM Transactions on Modeling and Computer Simulation* 13 (2): 155–179.

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 *Proceedings of the 2000 Winter Simulation Conference*, ed. J. A. Joines, R. R. Bartona, K. Kang, and P. A. Fishwick, 381–385: Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Sen, S., and J. Higle. 2005. The c3 theorem and a d2 algorithm for large scale stochastic integer programming. *Mathematical Programming* 104:1–20.

Shapiro, A. 1996. Simulation based optimization–convergence analysis and statistical inference. *Stochastic Models* 12 (3): 425–454.

Zhang, H., and W. K. Chan. 2007. Mathematical programming based–perturabtion analysis for $gi/g/1$ queues. In *Proceedings of the 2007 Winter Simulation Conference*, ed. S. G. Henderson, B. Biller, M.-H. Hsieh, J. Shortle, J. D. Tew, and R. R. Barton, 553–559: Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

Zhao, L., and S. Sen. 2006. A comparison of sample–path–based simulation-optimization and stochastic decomposition for multi-location transhipment problems. In *Proceedings of the 2006 Winter Simulation Conference*, ed. L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 238–244: Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

AUTHOR BIOGRAPHY

ANDREA MATTA is assistant professor at Politecnico di Milano, where he currently teaches manufacturing and integrated production systems. His research area includes analysis, design and management of production and service systems. His email address is <andrea.matta@polimi.it>.