# Chapter 20
# Simulation-Optimization in Support of Tactical and Strategic Enterprise Decisions

**Juan Camilo Zapata, Joesph Pekny, and Gintaras V. Reklaitis**

## 20.1 Introduction

The modern enterprise has developed highly complex supply chains in order to efficiently satisfy demand while remaining competitive. Supply chains have become distributed global networks that encompass not only the manufacture and delivery of goods but also the activities associated with their development. Moreover, local "here and now" decisions must be made in the presence of future uncertainty while also considering their global and long-term implications. This coupling of wide problem scope with multiple sources of internal and external uncertainties, such as production line breakdowns, raw material availability, market demand, exchange rate fluctuations, developmental failures, etc., has resulted in supply chain decision-making processes that are of high complexity and a very large scale (Zapata et al. 2008).

The need for techniques capable of determining the optimal set of decisions for this kind of systems has motivated the development of stochastic programming, stochastic dynamic programming, and simulation optimization. Stochastic programming and stochastic dynamic programming rely on the ability to articulate a tractable mathematical formulation of the system, which can be very difficult for complex supply chain applications. Furthermore, owing to the large size of problem spaces, nonlinearity of objective functions and constraint, and the discrete nature of many decisions, the resulting stochastic program may not always be solvable using state-of-the-art stochastic programming methods. Hence, the focus of this chapter is on simulation optimization, which couples the flexibility of discrete event simulation to accommodate arbitrary stochastic elements and model the dynamics and complexities of real-world systems without the need to develop formal mathematical models, and the ability of optimization schemes to systematically search the decision space. However, similar to stochastic programming and related techniques, simulation optimization can easily become computationally very demanding, and

J.F. Pekny (✉)
College of Engineering, Purdue University, West Lafayette, IN 47907, USA
e-mail: pekny@purdue.edu

K.G. Kempf et al. (eds.), *Planning Production and Inventories in the Extended Enterprise*, International Series in Operations Research & Management Science 151,
DOI 10.1007/978-1-4419-6485-4_20, 

thus requires that a range of sometimes rather subtle issues be addressed effectively to obtain a viable trade-off between solution time, modeling effort, and solution quality.

The chapter is organized as follows. Section 20.2 provides a summary of the different simulation-optimization methods that are available, aimed at guiding the reader in the selection of the most adequate technique for his/her particular problem. Section 20.3 presents two industrial case studies in which simulation optimization was used to support the decision-making process. Finally, concluding remarks are presented in Section 20.4.

## 20.2 Simulation-Optimization Solution Strategies

This section reviews the existing simulation-optimization methods, including their strengths and weaknesses. The aim of the review is to explain at a conceptual level the underlying algorithms and provide relevant references. To facilitate the presentation of the different methods we start by formalizing the problem in a mathematical sense. The problem to be solved can be expressed as

$$\min_{\theta \in \Theta}(\max)\ J\ (\theta), \tag{20.1}$$

where $\theta$ is the decision vector of $p$ parameters, the feasible region $\Theta \subset \Re^p$ is the set of possible values of the parameter $\theta$, and $J(\theta) = E\left[L(\theta, \omega)\right]$ represents the expected value of a performance measure $L(\theta, \omega)$. Notice that $L(\theta, \omega)$ is a random variable that can take different values depending on the specific realizations of the stochastic effects of the system, $\omega$. Therefore, the problem exhibits not only the typical challenges of finding an optimal solution but also those of estimating the performance measure.

In general, simulation-optimization methods are classified based on the continuous or discrete nature of the decision space (Fu 1994). In addition, methods for discrete variables are further cataloged according to the number of feasible solutions (small or large (including infinite)), and the ordered (i.e., represents different levels or degrees of the underlying characteristic (e.g., safety inventory level)) or unordered (i.e., represents categories that cannot be quantified (e.g., queue discipline)) nature of the variables. Figure 20.1 shows the classification scheme and the methods that fall in each class. It is important to highlight though that different classes of methods are often used in combination within a single computational scheme. On the continuous side, methods that mix response surface methodologies (RSM) and stochastic approximation (SA) have been developed (Ho 1992). In the case of discrete variables, hybrid approaches that combine different methods within a class, as well as methods in different classes, have been proposed. For example, Hall and Bowden (1996) combine metaheuristics with pattern search; Nozari and Morris (1984) combine ranking and selection (R&S) and pattern search, and Pichitlamken and Nelson (2003) combine R&S and metaheuristics.

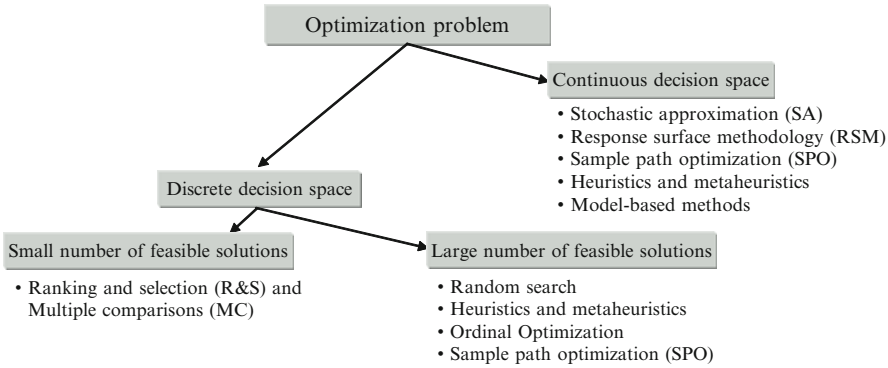


**Fig. 20.1** Classification of simulation-optimization techniques

### *20.2.1 Small Number of Discrete Feasible Solutions: Ranking and Selection (R&S) and Multiple Comparisons (MCs)*

The techniques available for problems with a small number of feasible solutions focus on the exhaustive comparison of all feasible solutions rather than on the search algorithms (Fu 2002). The presence of uncertainties transforms the comparison process into an inference exercise that uses the statistical machinery developed for the calculation of confidence intervals.

The basic concept behind Multiple Comparisons (MC) is very simple. The differences in performance measure, $\hat{J}(\theta_i) - \hat{J}(\theta_j)$, for some kind of pairwise comparison of the possible solutions are estimated from simulations. Then, the corresponding confidence intervals are examined in search of an absolute winner (i.e., in the case of an all-pairwise comparison, the $\theta_i$ whose confidence intervals in regard all other possible solutions are strictly negative (strictly positive)). However, it is not possible to guarantee a solution a priori since the confidence intervals may not be tight enough. Therefore, all the techniques in this class are aimed at exploiting the opportunities presented by simulation to reduce variance (e.g., common random numbers) and hence tighten the confidence intervals using the minimum possible number of simulations (Fu 1994).

Ranking and selection also uses confidence intervals but within the context of the correct selection concept. These methods measure in some way how far the chosen solution is from the optimal one. In general, two approaches have been proposed to measure that "distance." The first is known as the indifference zone. In this case, the objective is to obtain a solution that is within a certain range (indifference zone), $\delta$, of the optimal solution, $\theta_*$, with a specified probability of correct selection (PCS), $P^*$, (i.e., $P\{J(\theta_i) - J(\theta_*) < \delta\} \geq P^*$). The second approach, referred to as subset selection, guarantees that with a certain probability, a particular group of solutions chosen from the original set will contain at least one solution, $\theta_s$, that is within a specified indifference zone (i.e., $P\{J(\theta_s) - J(\theta_*) < \delta\} \geq P^*$).

From an implementation perspective, R&S methods follow two formulations (Fu et al. 2005), which are as follows:

1. Minimize the number of simulations subject to the PCS exceeding a given level (a traditional approach that offers little control over computational requirements).
2. Maximize the PCS subject to a given simulation budget constraint.

The latter formulation is also known as optimal computing budget allocation, and manages the computational effort by sacrificing the predictability of the confidence levels. Swisher et al. (2004) and Kim and Nelson (2006) provide extensive lists of references for both R&S and MC methods. These two classes of methods were originally considered to be two different strategies (Fu 1994), but Nelson and Matejcik (1995) established the connection between the two by showing that most indifference zone procedures can also provide confidence intervals for a certain type of multiple comparison method.

## *20.2.2 Large Number of Discrete Feasible Solutions*

### 20.2.2.1 Random Search

Random search methods move successively from one feasible solution to a neighboring one based on probabilistic arguments. All methods in this class (see the review by Banks (1998)) follow the same algorithmic structure as follows:

1. Initialization with a feasible solution
2. Probabilistic generation of a new decision vector, obtained from a set of neighboring feasible solutions
3. Estimation of performance measures and comparison with the values from the previous iteration
4. Evaluation of stopping criteria and return to Step 2 if not satisfied

The methods in this class are characterized by the definition of the neighborhood (the set in which the algorithm can move from one solution to another in a single iteration), the selection strategy of the next decision vector, and the manner in which the optimum is chosen. A representative example of this class of methods is simulated annealing, which attempts to achieve a global optimum by allowing moves leading to nonimproving solutions with a certain probability that depends on the stage of the procedure. Nonimproving moves leading to a poorer solution are more likely to be accepted early in the process; as the search progresses towards a global optimum, the probability of accepting non-improving moves tends to zero. A step-by-step description of a version of the method for a minimization problem is as follows (Alrefaei and Andradottir 1999):

*Step 1*. Initialize the decision variables, $\theta_0$, the number of iterations, $n = 0$, the optimal solution, $\theta_0^* = \theta_0$, and $A_0(\theta) = C_0(\theta) = 0$ for each $\theta$, where $A_i(\theta)$ is the sum of all the estimates of the performance measure $J(\theta)$,

$\hat{J}(\theta_n)$, obtained from simulations in the $_i$ first iterations and $C_i(\theta)$ is the number of replicates in the $_i$ first iterations.

*Step 2*. Generate a neighbor solution, $\theta'_n$, of the current point $\theta_n$ based on the chosen transition probability matrix, $R(\cdot,\cdot)$. This means that for all $\theta \in N(\theta_n)$, where $N(\theta_n)$ is the neighborhood of $\theta_n$ the probability of being selected in the next iteration is given by $P(\theta'_n = \theta) = R(\theta_n, \theta)$.

*Step 3*. Estimate $\hat{J}(\theta_n)$, and $\hat{J}(\theta'_n)$, using simulation. If $\hat{J}(\theta'_n) \leq \hat{J}(\theta_n)$, then let $\theta_{n+1} = \theta'_n$. Otherwise, sample a uniform distribution $U_n \sim U[0,1]$ and an exponential distribution $e_n \sim \exp\left[\hat{J}(\theta_n) \leq \hat{J}\left(\theta'_n\right) / T\right]$, and if $U_n \leq \dot{e}_n$ then let $\theta_{n+1} = \theta'_n$. Otherwise let $\theta_{n+1} = \theta_n$. Notice that $T$(known as the temperature) is the iteration-dependent parameter used to decrease the probability of accepting nonimproving moves as the number of iterations increase.

*Step 4*. Let $n = n + 1$, $A_n(\theta) = A_{n-1}(\theta) + \hat{J}(\theta)$, and $C_n(\theta) = C_{n-1}(\theta) + 1$, for $\theta = \theta_n$ or $\theta'_n$, and $A_n(\theta) = A_{n-1}(\theta)$ and $C_n(\theta) = C_{n-1}(\theta)$ for all $\theta$ that have been explored but are different from $\theta_n$ and $\theta'_n$ $\left(\theta \in \Theta_E \setminus \{\theta_n \text{ or } \theta'_n\}\right)$. Finally, select the $\theta$ associated with the smallest average value of the performance measure, $A_n(\theta) / C_n(\theta)$, from the set of decisions explored, $\Theta_E$, $\left(\min_{\theta \in \Theta_E} A_n(\theta) / C_n(\theta)\right)$.

Ideally, the method returns to Step 2 and the algorithm is repeated until convergence is reached. However, resource and time limitations may require the use of a user-defined stopping criteria, such as number of iterations, a threshold value for the performance measure, etc.

The advantages of random search methods are their model independence (i.e., no explicit mathematical model of the system needs to be developed) and the existence of theoretical convergence proofs under certain conditions. However, in practice, convergence can be slow and dependent on the selection of the neighborhood structure (Banks 2005) and does not scale well with the number of variables in the decision space.

#### 20.2.2.2 Ordinal Optimization

These methods are based on the observation that in most cases it is much easier, in terms of computation, to directly find the ordering among candidate solutions than it is to estimate the performance measure of each candidate solution and rank the solutions based on this measure. This idea can be explained with the following simple example (Fu et al. 2005). Assume that there are only two possible decision vectors $\theta_1$ and $\theta_2$, and the decision maker wants to know which of the two results in the smallest expected value of the performance measure ($J(\theta_1) < J(\theta_2)$ or $J(\theta_1) > J(\theta_2)$). One approach can be to estimate each of the expected performance measures independently, $\hat{J}(\theta_1)$ and $\hat{J}(\theta_2)$, until the standard error for each estimate is less than the indifference amount, $\varepsilon$, and compare the resulting values. On the other

hand, an ordinal optimization method would define a variable $X = L(\theta_1) - L(\theta_2)$ and determine whether $E[X]$ is positive or negative. The latter strategy is more efficient because the estimation of $E[X]$ requires lesser number of simulations when compared with the estimation of $\hat{J}(\theta_1)$ and $\hat{J}(\theta_2)$. Swisher et al. (2004) provide an extensive list of references for this technique.

## *20.2.3 Continuous Decision Variables*

### 20.2.3.1 Stochastic Approximation (SA)

Stochastic approximation refers to a group of methods that attempt to mimic the gradient search method traditionally used in deterministic optimization. As in its deterministic counterpart, SA searches for a local optimum to the problem given by (20.1), that satisfies the first-order condition

$$\hat{\nabla} J(\theta) = 0, \tag{20.2}$$

where $\hat{\nabla} J(\theta)$ represents the estimated gradient of the performance function. The analogy to the deterministic case also applies to the general structure of SA algorithms which are based on the following iterative form:

$$\theta_{n+1} = \prod_{\Theta} \left( \theta_n - a_n \hat{\nabla} J(\theta_n) \right). \tag{20.3}$$

Here $\theta_n$ is the solution vector at the beginning of iteration $_n$, $\{a_n\}$ is a positive sequence of step sizes, and $\prod_{\Theta}$ represents some projection back into the feasible set $\Theta$ when the iteration leads to a solution outside the set. Similar to that in the deterministic case, the algorithm determines at each iteration the value of the decision vector based on the gradient and the step size values calculated for that iteration and the value of the decision vector in the previous iteration. Algorithms referred to as Robust SA algorithms differ slightly in that they use the iterative process based on (20.3), but instead of returning the final value of the decision vector as the optimum, they return an average (e.g., moving horizon or exponentially weighted moving average) of a certain number of iterates to reduce the variance in the estimation (Fu 2002). The set of constraints that determine the feasible region also exhibit some differences when compared with the deterministic case. In general, the feasible region is determined by a mix of deterministic and probabilistic constraints. Probabilistic constraints limit the probability of constraint violations but not the magnitude of the violations. They are expressed as follows:

$$P\left( f(\theta, \omega) \geq 0 \right) > 1 - \alpha, \tag{20.4}$$

where $f$ is the random vector representing the left hand side of a set of constraints whose realizations depend on the set of decisions $\theta$, and the presence of

uncertainties $\omega$. $P$ is the vector of probabilities of violating the constraints, and $\alpha$ ($\alpha \in (0, 1)$) are the tolerance levels of the decision maker to these violations. Notice, however, that only a few algorithms can handle this kind of constraint (see Andradottir (1998) and Kushner and Yin (2003) and references hereafter).

For all SA algorithms to properly converge it is required that the step size goes to zero at a rate that is not too fast (to avoid premature convergence to a suboptimal solution), and not too slow (to ensure eventual convergence). Mathematically, these conditions are commonly represented as $\sum_{n=1}^{\infty} a_n = \infty$ and $\sum_{n=1}^{\infty} a_n^2 < \infty$. In addition, it is required that the bias of the objective function gradient estimate, $\hat{\nabla} J(\theta)$ in (20.3), goes to zero (Fu 1994). In theory, the appropriate step size rate can be achieved using a simple harmonic series ($a_n = a/n$), but in practice this choice results in slow convergence rates. Since the performance of any SA algorithm is quite sensitive to this sequence, researchers have developed different strategies aimed to speed up convergence. Heuristic decrements in step size have been proposed (e.g., Chapter 9 in Banks (1998) and references therein), as well as the use of a constant step size in the early stages of the iterative process followed by heuristic decrements (Fu 2002).

The need to obtain unbiased estimates of the objective function gradient in an efficient manner has motivated most of the different techniques used in SA. The remainder of this section provides an introduction to each of these developments and a summary of their main strengths and limitations.

### Finite Differences (FD)

Similar to that in numerical differentiation the idea is to use a secant as an approximation to the gradient (a tangent). Therefore, the value of $\hat{\nabla} J(\theta)$ at iteration $n$ is given by

$$\hat{\nabla} J_n = \left[\hat{\nabla}_1 J_n \ldots \hat{\nabla}_p J_n\right], \tag{20.5}$$

where $\hat{\nabla}_i J_n$ can be calculated using forward differences as follows:

$$\hat{\nabla}_i J_n = \frac{\hat{J}(\theta_n + c_n e_i) - \hat{J}(\theta_n)}{c_n} \tag{20.6}$$

or central differences as follows:

$$\hat{\nabla}_i J_n = \frac{\hat{J}(\theta_n + c_n e_i) - \hat{J}(\theta_n - c_n e_i)}{2c_n}. \tag{20.7}$$

Here $e_{\mathrm{j}}$ denotes the $i$th unit vector (e.g., $e_i = (0, \ldots, 0, 1, 0, \ldots 0)$) and $c_n$ a small positive number that can take a different value at each iteration. The use of forward or central differences is driven by the trade-off between estimation bias and computational burden. The calculation based on central differences requires the simulation

of $2p$ sets with $\theta_n \pm c_n e_i$ values, while the one based on forward differences requires only $p$+1 simulations. However, the estimators obtained using central differences usually have smaller bias than those obtained using forward differences, which often leads to a smaller number of iterations, $n$.

When finite differences are used to obtain the gradients for (20.3), the SA technique is called the KieferWolfowitz algorithm (Kiefer and Wolfowitz 1952). This algorithm has the following two important advantages with respect to other SA techniques: (1) implementation is straightforward due to its simplicity, and (2) it is not model-dependent (i.e., no explicit mathematical model of the system needs to be developed, which means that this technique can be applied to systems with any level of complexity). However, the KieferWolfowitz algorithm converges to the true local optimum only when very small $c_n$ values (i.e., $c_n \to 0$) are used. The problem of using small $c_n$ values is that the estimated gradients, $\hat{\nabla} J_n$, exhibit large variances that often slow down the convergence rate. This limitation has been addressed in some situations by using common random numbers (Fu 1994).

### Simultaneous Perturbation (SP)

This technique as well as the other gradient estimation techniques in the remainder of this section was developed in response to the significant computational requirements of methods based on finite differences. SP uses the same conceptual framework of finite differences, but reduces the number of simulations required by perturbing all components of the decision vector simultaneously. Specifically, the value for any $\hat{\nabla}_i J_n$ can be obtained from the results of the simulations for just two sets of $\theta$ values, $(\theta_n + c_n \Delta_n)$ and $(\theta_n - c_n \Delta_n)$, with the following expression:

$$\hat{\nabla} J_n = \frac{\hat{J}\left(\theta_n + c_n \Delta_n\right) - \hat{J}\left(\theta_n - c_n \Delta_n\right)}{2 c_n \Delta_n}, \tag{20.8}$$

where $\Delta_n = \left(\Delta_{n1}, \ldots, \Delta_{np}\right)$ represents a vector of independent identically distributed (i.i.d.) random perturbations with zero mean. Though the elements of $\Delta_n$ may be assigned different kinds of distributions according to the specific characteristics of the problem at hand (Spall 1999). Sadegh and Spall (1998) showed that the optimal distribution for these elements, based on asymptotic distribution results, is a symmetric Bernoulli (i.e., the probability of success is 0.5). In a more recent development, deterministic perturbation sequences have been proposed to enhance the convergence rate of the stochastic approximation method based on simultaneous perturbation (SPSA) (Bhatnagar 2003).

Spall (1992) found that the SPSA method was superior (i.e., the difference between the actual minimum value and the estimated one was smaller for the same amount of computational effort) to the KieferWolfowitz algorithm for a fairly complicated numerical study. He also proved that both approaches have the same asymptotic convergence rate in spite of SPSA's significantly lower computational requirements at each iteration. In theory, the superiority of the SPSA method grows

with the dimension $p$ of the decision vector as the computational burden of SPSA is independent of its dimension. However, this potential for higher efficiency can only be realized if the number of iterations required to converge to the global optimum does not increase to a level that exceeds the savings obtained by reducing the number of simulations in each iteration. To realize as much of that potential as possible, not only must the selection of $a_n$ in (20.3) be carefully made (as in any other SA technique), but also that of $c_n$ and $\Delta_n$ in (20.8). Though the selection is problem-dependent and there are no universal rules, Spall (1998, 1999) provides some recommendations as a starting point.

Perturbation Analysis (PA)

Though the name may lead one to think that there is some kind of connection between PA and SP, these two techniques use completely different conceptual frameworks. PA does not explore through simulation the region around the decision vector $\theta_n$ to determine in which direction to move for the next iteration; instead, it determines such a direction by using only the output of the simulations with the current value of $\theta_n$. Hence PA infers the behavior of the system with $\theta_n + \Delta\theta_n$, where $\Delta\theta_n$ is a small perturbation, from the information obtained with $\theta_n$, and uses it to estimate the gradient for the next iteration. This may seem a little too "magical;" in the words of the developers of the technique (Ho and Cao 1991): "At first this may sound counterintuitive, stemming from the philosophical belief that one cannot get something for nothing. A more concrete and equally intuitive objection is that sample paths of the simulation under $\theta_n$ and $\theta_n + \Delta\theta_n$ will in general sooner or later become totally different as they evolve in time even for very small $\Delta\theta_n$". However, all the techniques belonging to the PA class accomplish this seemingly impossible objective by using some kind of sample path analysis. The pioneering technique in the field is known as infinitesimal perturbation analysis (IPA). The basic idea behind IPA is that it is possible to reconstruct a perturbed path from a nominal one by keeping track of the changes in the timing of events, if the sequence of events in the simulation do not change (the critical timing path stays constant). The relevance of IPA is that it is capable of simultaneously implementing such an accounting exercise for a multitude of perturbations, and when applicable is highly efficient (i.e., exhibits fast convergence) (Ho and Cao 1991). However, it is only suitable for continuous performance measures and requires complete knowledge of the underlying model, that is, an explicit model that relates inputs and outputs has to be available. In broad terms, the method can be described in three steps. The first step consists in developing a recursive (e.g., indexed by the number of customers arriving) explicit mathematical model that relates the outputs of the simulation that are part of the performance measure (e.g., inventory levels and total time of a customer in the system) with those random variables (e.g., quantity produced and service time at the teller) which depend directly on the decision variables (e.g., base stock and mean value of service time). Next, the model is differentiated with respect to the decision variables and the results are substituted into the function that represents the expected value

of the performance measure gradient. Finally, the differentials of the random variables with respect to the decision variables are substituted based on the perturbation generation rule and the gradient is calculated using simulation outputs The perturbation generation rule allows one to obtain the changes on the random variable, $\omega$, caused by changes in the decision variables in terms of the information collected from simulation runs with $\theta_n$. The perturbation generation rule is given by

$$\frac{\mathrm{d}\omega}{\mathrm{d}\theta} = \frac{\mathrm{d}F^{-1}(\theta, \xi)}{\mathrm{d}\theta}, \tag{20.9}$$

where $F(\theta, \omega)$ is the cumulative distribution function of $\xi$ with parameter $\theta$, and random variable $\omega$; and $\xi$ is a random variable independent of $\theta$ (e.g., if $\omega$ is exponentially distributed, with mean $\theta$, $\omega = F^{-1}(\theta, \xi) = -\ln(1-\xi)\cdot\theta$ and $\xi \sim U[0, 1)$). However, from an implementation perspective, it is more convenient to use an equivalent formula that does not require the form of the inverse function:

$$\frac{\mathrm{d}\omega}{\mathrm{d}\theta} = \frac{\mathrm{d}F(\theta, \omega)\ \mathrm{d}\theta}{\mathrm{d}F(\theta, \omega)\ \mathrm{d}\omega}. \tag{20.10}$$

To clarify the method let us consider a very simple problem (Fu 1994) (For a more complex case in the context of inventory management refer to Tayur et al. (1999)). Find the mean service time $\theta$ of a first come first serve single server M/M/1 queue, which minimizes the sum of expected mean time in the system over a given number of customers served, $L$:

$$\min_{\theta \in \Theta} E\left[\frac{1}{N}\sum_{i=1}^{N} T_i\right], \tag{20.11}$$

where $T_i$ is the time in the system for the $i$th customer and $N$ is the number of customers served. Note that $T_i$ is the only output of the simulation that is part of the performance measure, $L = \left[1/N \sum_{i=1}^{N} T_i\right]$. Under the conditions described, $T_i$ satisfies the recursive Lindley equation

$$T_{i+1} = \omega_{i+1} + \begin{cases} T_i - A_{i+1} & \text{if } T_i \geq A_{i+1} \\ 0 & \text{if } T_i < A_{i+1} \end{cases}, \tag{20.12}$$

where $\omega_i$ is the service time for the $i$th customer and $A_i$ is the interarrival time between the $(i-1)$ th and the $i$th customers. Notice that $\omega_i$ is the random variable that depends on the decision variable, which implies that (20.12) is the explicit model referred to in the first step of the method. Continuing with the method, (20.12) is differentiated to obtain

$$\frac{\mathrm{d}T_{i+1}}{\mathrm{d}\theta} = \frac{\mathrm{d}\omega_{i+1}}{\mathrm{d}\theta} + \begin{cases} \mathrm{d}T_i/\mathrm{d}\theta & \text{if } T_i \geq A_{i+1} \\ 0 & \text{if } T_i < A_{i+1} \end{cases}. \tag{20.13}$$

and expression (20.13) is substituted recursively into itself for each customer, and the resulting expressions in terms of $\mathrm{d}\omega_i \,/\, \mathrm{d}\theta$ are substituted into the expression for the expectation of the gradient of $L$, $E\left[\mathrm{d}L \,/\, \mathrm{d}\theta\right] = E\left[1 \,/ N \sum_{i=1}^{N} \mathrm{d}T_i \,/\, \mathrm{d}\theta\right]$, to obtain

$$E\left[\frac{\mathrm{d}L}{\mathrm{d}\theta}\right] = \sum_{s=1}^{S}\left[\frac{1}{n_s}\sum_{i=1}^{n_s}\sum_{j=1}^{i}\frac{\mathrm{d}\omega_{(j,s)}}{\mathrm{d}\theta}\right] \tag{20.14}$$

where $S$ is the number of simulations, $n_s$ the number of customers served in simulation $s$, and the $(j, s)$ subscript denotes the $j$th customer in the $s$th simulation. Alternatively, the expectation can be estimated with the output from a single simulation. This is possible because the system is regenerative, which means that at random times $0 = t_0 < t_1 < t_2 < \cdots$ the future of the stochastic process becomes a probabilistic replica of itself. Therefore, instead of using results from multiple simulations, it is enough to extend the duration of only one run and split it into i.i.d. periods (regenerative cycles). In this case, the expression for the expectation is given by

$$E\left[\frac{\mathrm{d}L}{d\theta}\right] = \frac{1}{N}\sum_{m=1}^{M}\sum_{i=1}^{n_m}\sum_{j=1}^{i}\frac{\mathrm{d}\omega_{(j,m)}}{\mathrm{d}\theta}, \tag{20.15}$$

where $M$ is the number of regenerative cycles, $n_m$ the number of customers served in the $m$th regenerative cycle, and the $(j, m)$ subscript denotes the $j$th customer in the $m$th busy period, i.e., $(j, m) = j + \sum_{i=1}^{m-1} n_i$.

The final step of the method requires the substitution of the random variable differentials. As $\omega_i$ is exponentially distributed, by using (20.10), it can be shown that $\mathrm{d}\omega_{(j,s)} \,/\, \mathrm{d}\theta = \omega_{(j,s)} \,/\, \theta_n$. Therefore, the final expression for the expectation of the gradient is given by

$$E\left[\frac{\mathrm{d}L}{\mathrm{d}\theta}\right] = \sum_{s=1}^{S}\left[\frac{1}{n_s}\sum_{i=1}^{n_s}\sum_{j=1}^{i}\frac{\omega_{(j,s)}}{\theta_n}\right]. \tag{20.16}$$

The use of IPA to estimate the performance measure gradients in the SA approach provides a framework that converges faster than that based on finite differences. However, it has the following two important drawbacks: (1) it is model-dependent and (2) it estimates $E\left[\mathrm{d}L \,/\, \mathrm{d}\theta\right]$ instead of the desired $\mathrm{d}E[L] \,/\, \mathrm{d}\theta$. Clearly, the second aspect is not an issue when the expectation (integration) and differentiation operators can be interchanged. However, in most cases this is only possible when $L$ is almost surely continuous with respect to $\theta$ (Ho and Cao 1991). From an implementation perspective this means that it has to be possible to develop a "transformation" that allows one to represent the system in terms of random variables whose distributions do not depend on decision variables, and the performance function based on the transformation has to be continuous in $\theta$ for almost every $\omega$.

In addition, as mentioned above, if the CPT changes or the performance measure is discontinuous, IPA is not valid. Though this problem has been addressed for some conditions using the same conceptual framework (Fu and Hu 1997), such extensions are not as straightforward as IPA.

### Likelihood Ratio (LR)

As in the case of PA, LR methods, which are also known as score function (SF) methods, use only the output of the simulations with the current value of the decision vector $\theta_n$ to estimate the gradient of the expected value of the performance measure. The methods in this class require milder continuity requirements for the performance measure $L(\theta, \omega)$, than those stipulated by PA. This is possible as the gradient is calculated by differentiating the probability distribution function of the performance measure instead of the performance measure itself. However, applicability of this idea is limited to problems whose decision variables $\theta$ are parameters of the distributions that represent the uncertainty in the system. This means that decision variables that are not part of the characterization of the uncertainties, such as reordering points and safety inventory levels, cannot be part of $\theta$. To overcome this limitation, Rubinstein and coworkers (Kleijnen and Rubinstein 1996; Rubinstein and Shapiro 1993) have proposed transformations for some types of problems that move the parameters lying outside the characterization of the probability distributions into them.

LR methods are strongly connected to the importance sampling concept, commonly used to derive estimators with a reduced variance. The basic idea behind LR methods can be illustrated by deriving the gradient of the performance measure for static systems (i.e., a system that does not evolve in time, such as reliability problems) with probability density functions that only depend on a single parameter. In this case, the expected value of the performance measure has the form

$$E[L] = \int L(\omega) \mathrm{d}F(\theta, \omega) = \int L(\omega) f(\theta, \omega) \mathrm{d}\omega, \tag{20.17}$$

where $F(\theta, \omega)$ represents the cumulative distribution of $\omega$ and $f(\theta, \omega)$ the density function. Differentiating (20.17) with respect to $\theta$, interchanging integration and differentiation, and multiplying and dividing by $f(\theta, \omega)$ we obtain

$$\begin{aligned} \frac{\partial E[L]}{\partial \theta} &= \frac{\partial}{\partial \theta} \int L(\omega) f(\theta, \omega) \mathrm{d}\omega = \int L(\omega) \frac{\partial f(\theta, \omega)}{\partial \theta} \mathrm{d}\omega \\ &= \int L(\omega) \frac{\partial f(\theta, \omega)}{\partial \theta} \frac{f(\theta, \omega)}{f(\theta, \omega)} \mathrm{d}\omega = \int L(\omega) \frac{\partial \ln f(\theta, \omega)}{\partial \theta} f(\theta, \omega) \mathrm{d}\omega \\ &= E\left[ L(\omega) \frac{\partial \ln f(\theta, \omega)}{\partial \theta} \right] = E\left[ L(\omega) S^{(1)}(\theta, \omega) \right] \end{aligned} \tag{20.18}$$

where $S^{(1)}$ is called the efficient score function. Notice that multiplication and division by $f(\theta, \omega)$ are necessary to obtain an expression that has the mathematical form of an expectation. The expectation form is convenient because it allows the estimation of the desired quantity from simulated data by averaging it over a given set of realizations. Therefore, (20.18) can be expressed as

$$\frac{\mathrm{d}E[L]}{\mathrm{d}\theta} = \frac{1}{N}\sum_{i=1}^{N} L(\omega_i)S^{(1)}(\theta, \omega_i) \tag{20.19}$$

where $N$ is the total number of simulations, and $L(\omega_i)$ and $S^{(1)}(\theta, \omega_i)$ are particular realizations of $L(\omega)$ and $S^{(1)}(\theta, \omega)$, respectively. Though (20.19) is readily implementable, it usually does not result in the fastest possible convergence. Equation (20.18) can be improved from a variance reduction perspective by exploiting the ideas behind importance sampling. Specifically, by multiplying and dividing the integrand in (20.17) by $g(\omega)$, where $g(\omega)$ is a probability distribution whose support (set of values of $\omega$ for which $g(\omega)$ is strictly greater than zero) is included in the support of $f(\theta, \omega)$ for every $\theta$, the gradient of $E[L]$ can be expressed as

$$\frac{\mathrm{d}E[L]}{\mathrm{d}\theta} = \int L(\omega)\frac{\partial W(\theta, \omega)}{\partial \theta}\mathrm{d}G(\omega) = E\left[L(Z)\frac{\partial W(\theta, Z)}{\partial \theta}\right] \tag{20.20}$$

where $G(\omega)$ is the cumulative probability distribution of $g(\omega)$, $W(\theta, \omega) = f(\theta, \omega) \,/\, g(\omega)$, $Z$ is a random variable with density $g(\omega)$, and $\partial W(\theta, Z) \,/\, \partial\theta = W(\theta, Z)S^{(1)}(\theta, Z)$. Notice that the change in the random variable is not more that a change in notation to emphasize that the expectation is with respect to $g(\omega)$, instead of $f(\theta, \omega)$. The estimator of the gradient in this case is given by

$$\frac{\mathrm{d}E[L]}{\mathrm{d}\theta} = \frac{1}{N}\sum_{i=1}^{N} L(Z_i)W(\theta, Z_i)S^{(1)}(\theta, Z_i). \tag{20.21}$$

Though it is a common strategy to select $g(\omega) = f(\theta_0, \omega)$ for some fixed value $\theta_0$, the accuracy of the estimator is determined by its variance, which depends on $g(\omega)$. Therefore, the selection of $g(\omega)$ and the calculation of the estimator's variance are integral parts of this technique. Rubinstein and Shapiro (1993) provide a complete presentation of this methodology in the context of static as well as for dynamic systems (e.g., queuing networks).

In terms of strengths and weaknesses, the use of gradients estimated with LR for SA can result in very rapidly converging algorithms because LR exploits the structure of the performance measure. However, it requires complete knowledge of the density function of the uncertainties, careful selection of $g(\omega)$, and the satisfaction of certain regularity conditions which guarantee the interchangeability of

the differentiation and integration operators in (20.18). Specifically, a function $h(\omega)$ with finite Lebesgue integral that satisfies

$$|L(\omega)\partial f(\theta,\omega) \,/\, \partial(\omega)| \leq h(\omega) \tag{20.22}$$

has to exist.

Finally, it is important to note that LR often works in systems where IPA fails and can be more easily extended to higher derivative estimates for higher order Newton-like methods than (20.3). However, when IPA works, the gradients usually have much lower variances than those obtained with LR methods (Fu 1994).

### Frequency Domain Analysis

Frequency domain analysis (FDA) estimates the gradient of the expected value of the performance measure by using harmonic analysis. The method is based on the idea that the change in the magnitude of the performance measure, caused by perturbing the vector of decision variables $\theta$, with sinusoidal functions allows the determination of the sensitivity of the system to each of those variables in a single simulation. In theory, the use of distinct frequencies for $\theta_i$ makes possible the estimation of each variable's contribution to the performance measure. In this method, $\theta$ is perturbed according to

$$\theta(t) = \overline{\theta} + \alpha \sin(wt), \tag{20.23}$$

where $\overline{\theta}$ is a vector of nominal values for the decision variables, $\alpha$ is the vector of oscillation amplitudes, $w$ is the vector of oscillation frequencies, called the driving frequencies, and $t = 1, 2, \ldots, T$ is a "time" index. Notice that t is rarely the simulation time, instead it is a problem specific discrete label for the transient entities processed through the simulation (e.g., number of customers).

Conceptually, the method exploits the orthogonality (i.e., if $g$ and $f$ are two functions, they are orthogonal if $\int_a^b f(x)g(x)\mathrm{d}x = 0$) of the harmonic basis (i.e., sine and cosine), to isolate the impact of each decision variable on the performance measure gradient. Specifically, the method assumes that the performance measure can be approximated by a polynomial meta-model that can be transformed into a trigonometric (harmonic) one using (20.23). The polynomial meta-model is obtained by assuming that the relationship between $L$ and $\theta(t)$ can be locally approximated around $\overline{\theta}$ by a second-order Taylor expansion:

$$\begin{aligned} L\left(t|\theta(t)\right) &= L(\overline{\theta}) + \sum_{j=1}^{p} \sum_{\tau=-\infty}^{\infty} g_j(\tau)(\theta_j(t-\tau) - \overline{\theta}_j) \\ &\quad + \sum_{j=1}^{p} \sum_{\tau=-\infty}^{\infty} g_{jj}(\tau)(\theta_j(t-\tau) - \overline{\theta}_j)^2 \end{aligned}$$

$$+\sum_{j=1}^{p-1}\sum_{m=j+1}^{p}\sum_{\tau=-\infty}^{\infty}\sum_{\nu=-\infty}^{\infty} g_{jm}(\tau,\nu)\left(\theta_j(t-\tau)-\overline{\theta}_j\right)$$
$$\times\left(\theta_m(t-\nu)-\overline{\theta}_m\right)+O\left(\left\|\theta(t)-\overline{\theta}\right\|_\infty^3\right)+\varepsilon\left(t|\theta(t)\right) \quad (20.24)$$

where $p$ is the number of decision variables; $\|\cdot\|_\infty$ the infinity norm, $O(\cdot)$, the order of magnitude of the error generated by the truncation of the Taylor series, $\varepsilon\left(t|\theta\left(t\right)\right)$ represents the stochastic part of the model, and $g$ are the so called memory filters that weight past values of $\theta(t)$. The Taylor expansion is advantageous as the summations of filters in each term of (20.24) can be obtained through regression analysis from simulation results, and can be associated with the gradient and higher order differentials of the expected value of the performance measure, $J(\theta)$. The relationship can be derived by setting $\theta(t)=\overline{\theta}$ in (20.24) and differentiating it with respect to the decision variables:

$$\sum_{\tau=-\infty}^{\infty} g_j(\tau)=\frac{\partial J(\theta)}{\partial\theta_j},$$
$$\sum_{\tau=-\infty}^{\infty} g_{jj}(\tau)=\frac{\partial^2 J(\theta)}{2\partial\theta_j^2},$$
$$\sum_{\tau=-\infty}^{\infty}\sum_{\nu=-\infty}^{\infty} g_{jm}(\tau,\nu)=\frac{\partial^2 J(\theta)}{\partial\theta_j\,\partial\theta_m} \quad (20.25)$$

It is important to note that the following three assumptions are behind this derivation: (1) $\varepsilon(t|\theta\left(t\right))$ has a stationary covariance (i.e., it is fixed for all $t$) with mean 0, 2. The summation of covariances from all the time periods is bounded, and (3) $J(\theta)$ is relatively smooth (i.e., twice continuously differentiable) (Ho and Cao 1991).

By substituting (20.23) into (20.24), the following form of the meta-model is obtained:

$$L\left(t\,|\theta\left(t\right)\right) = L(\overline{\theta})+\sum_{j=1}^{p}\sum_{\tau=-\infty}^{\infty} g_j\left(\tau\right)a_j\sin\left(w_j(t-\tau)\right)$$
$$+\sum_{j=1}^{p}\sum_{\tau=-\infty}^{\infty} g_{jj}(\tau)a_j^2\sin^2\left(w_j(t-\tau)\right)$$
$$+\sum_{j=1}^{p-1}\sum_{m=j+1}^{p}\sum_{\tau=-\infty}^{\infty}\sum_{\nu=-\infty}^{\infty} g_{jm}(\tau,\nu)\,a_j a_m\sin\left(w_j\,(t-\tau)\right)$$
$$\times\sin\left(w_m\,(t-\nu)\right)+O\left(\left\|\tilde{\theta}(t)-\overline{\theta}\right\|_\infty^3\right)+\varepsilon\left(t\mid\theta(t)\right) \quad (20.26)$$

where $\tilde{\theta}(t) - \overline{\theta} = \left(a_1 \sin(w_1 t), a_2 \sin(w_2 t), \ldots, a_p \sin(w_p t)\right)$. Equation (20.26) can be further manipulated using trigonometric identities and some algebra to derive the following more convenient form:

$$\begin{aligned} L\left(t|\,\tilde{\theta}\,(t)\right) &= B(0) + \sum_{j=1}^{p}\left[A(w_j)\sin(w_j t) + B(w_j)\cos(w_j t)\right] \\ &+ \sum_{j=1}^{p}\left[A(2w_j)\sin(2w_j t) + B(2w_j)\cos(2w_j t)\right] \\ &+ \sum_{j=1}^{p-1}\sum_{m=j+1}^{p}\left[A(w_j \pm w_m)\sin((w_j \pm w_m)t)\right] + B(w_j \pm w_m) \\ &\times \cos((w_j \pm w_m)t) + O\left(\left\|\tilde{\theta}(t) - \overline{\theta}\right\|_{\infty}^{3}\right) + \varepsilon(t \mid \tilde{\theta}\,(t)) \end{aligned} \tag{20.27}$$

where the coefficients $A(w_j)$, $A(2w_j)$, $A(w_j \pm w_m)$, $B(0)$, $B(w_j)$, $B(2w_j)$, and $B(w_j \pm w_m)$ are in terms of the memory filters and sinusoidal functions; for instance, $A(w_j) = a_j \sum_{\tau=-\infty}^{\infty} g_j(\tau)\cos(w_j \tau)$. Therefore, by taking the limit as $w_j \to 0$ and using the results in (20.25) it can be proved that the gradient and higher order derivatives of the performance measure can be obtained from the estimates of these coefficients (Jacobson and Schruben 1999). Specifically, the estimate of the ith component of the gradient has the following form:

$$\frac{\partial E[L]}{\partial \theta_i} = \frac{2}{a_i T}\sum_{t=1}^{T} L\left(\theta\left(t\,|\tilde{\theta}\,(t)\right)\right)\sin(w_i t), \tag{20.28}$$

where $i = 1, \ldots, p$, and $t \mid \tilde{\theta}\,(t)$ denotes that $L(\theta)$ is sampled at each "time" $t$ from a simulation in which the input for the decision vector is given by (20.23). From a theoretical perspective, the main advantage of the FDA method is the combination of model independence (excluding the indexing issue) and minimum simulation requirements. However, the determination of the specific values of the frequencies $w_i$ is not a trivial task as they have to be selected in such a way that aliasing (i.e., the effect that causes different signals to become indistinguishable) is prevented, and the need to make them tend to 0 ($w \to 0$) translates into very long simulation horizons. In addition, the method is limited to systems in steady-state and exhibits an unavoidable trade-off between the variance of the gradient estimator (the larger $\alpha$ the better) and its bias (the smaller $\alpha$ the better) (Jacobson and Schruben 1999).

Finally, regarding the indexing issue, it is important to note that although simple indices based on the concept of transient entities processed are limited to very simple systems (Fu 1994), Hazra et al. (1997) have suggested a strategy to discretize the

global simulation clock of the simulation and used it as a "time" index that can fit any kind of system. However, this approach can be difficult, if not impossible, to implement in some commercially available discrete event simulation software.

#### 20.2.3.2 Response Surface Methodology (RSM)

RSM encompasses two types of strategies. The first consists of the use of regression techniques to construct an approximate functional relationship (meta-model) between the decision variables and the performance measure that fits the entire decision space, $\Theta$, or a subset of $\Theta$, and the subsequent use of optimization methods on the meta-model to analytically estimate an optimum (Wan and coworkers (2006) provide an example of this technique in the context of the pharmaceutical industry). The second strategy, known as sequential RSM, follows a philosophy similar to SA, consisting of three steps that are repeated iteratively until a convergence criterion is satisfied. First, a meta-model in the region surrounding the decision vector obtained in the previous iteration is constructed. Next, the meta-model is differentiated to obtain a functional form of the gradient, and substituted into

$$\theta_{n+1} = \theta_n - a_n \hat{\nabla} J(\theta_n). \tag{20.29}$$

Finally, the next iterate for the decision vector is computed from (20.29) through a line search. In spite of the similarities between sequential RSM and the SA methods discussed above, RSM differs due to its inability to mathematically show an asymptotical convergence, and the use of functional forms for the gradient instead of numerical values.

In the literature, the most commonly found RSM algorithm is a mix of the two philosophies described above, which uses a two-phase design of experiments based polynomial regression strategy (Fu 1994). In Phase I, first-order experimental designs (i.e., consider only linear (main) effects) are used iteratively until the linear response surface becomes inadequate (i.e., the interaction effects become larger than the main effects), while in Phase II, a quadratic response surface (fitted using second-order experimental designs) of the area identified in Phase I is used to analytically determine the optimum.

The most important considerations in the implementation of any RSM method are the inclusion of variance reduction techniques (e.g., common random variables, control random variables, etc.) and the selection of the experimental designs. Every type of design provides a different trade-off between variance (due to sample variation) and bias (due to poor model fit), which results in a particular performance of the algorithm. Jacobson and Schruben (1989), Safizadeh (1990) and Kleijnen in Banks (1998) provide an exhaustive set of references for RSM strategies, including algorithms that allow the inclusion of deterministic constraints.

The attractiveness of conventional RSM methods is rooted in their applicability to any kind of system. However, its "black box" nature that does not allow rigorous convergence analysis, its typical blind search of the solution space (which usually

leads to the excessive use of simulation runs in unimportant areas), and the limited ability of low-degree polynomials to fit complex functions (which can provide poor results when the performance measure is represented by functions with sharp ridges and flat valleys (Azadivar 1999)) has limited its use to simple problems. Though the last two shortcomings have been addressed with the use of better model fitting methods (e.g., Wan et al. (2005) show that RSM may perform considerably better than SPA when support vector machines are used for model regression), these techniques are considerably more involved from a statistical perspective than traditional regression techniques.

#### 20.2.3.3 Sample Path Optimization (SPO)

Conceptually, the methods in this class use an approach similar to the first type of RSM strategy described above. Specifically, the system is sampled multiple times, and the information collected is used to generate a functional approximation of the performance measure that is optimized using deterministic optimization tools. The main difference between SPO and RSM is that the latter uses regression techniques to obtain the functional approximation, whereas the first uses an explicit model obtained from first principles (like IPA and LR), or completely avoids the need for an explicit model by exploiting the structure of the problem. Though there are no specific rules to derive the explicit model (the key step in SPO), the basic idea is to be able to generate expressions in which the expected value of the performance measure is explicitly represented in terms of decision variables and random variables independent of the decision variables:

$$\hat{J}(\theta) = \frac{1}{N} \sum_{i=1}^{N} h(\theta, \xi_i), \tag{20.30}$$

where $N$ is the number of simulations and $\xi_i$ the $i$th realization of the $\theta$ independent random variables. In some cases, a model with this kind of structure can be directly derived, but in most of real-world problems that is not possible. Therefore, similar to that in IPA, transformations have to be implemented to obtain objective functions with underlying random variables independent of $\theta$. This means that any model suitable for IPA can be solved with SPO. Alternatively, for some problems in which the effect of the decision variables only enters the problem through the distribution of the underlying random variables, approximations based on likelihood ratios and importance sampling have been developed to obtain the required functional form of the performance measure (e.g., Banks (1998), Rubinstein and Shapiro (1993), and Shapiro (1996) and references therein). Finally, in some cases it is possible to develop routines that do not require the derivation of an explicit model. A very simple example of this subgroup of problems is the allocation of a fixed amount of buffer space among a group of servers such that the time to overflow the system is maximized. The SPO strategy is to run the simulation multiple times with the

buffers between the servers unconstrained until the total availability of buffer space is exhausted and then to chose the most frequent allocation. Healy and Fu (1992, 1997) and Healy and Schruben (1991) provide a complete presentation of this example and more involved problems, including cases with discrete decision spaces.

In general, SPO has several important advantages, which are as follows: (1) the strategy that uses explicit models can deal with problems in which the decision variables are subject to constraints of the type $E[k(\theta)] < 0$, where $k(\theta)$ can be derived in the same way as the performance measure, (2) it can be easily implemented in commercial simulators, due to its modularity (i.e., first simulation and second optimization), and (3) it can be applied to some problems with discrete decision spaces. However, it also has considerable limitations such as the following: (1) it is restricted to systems that have reached a steady state, (2) it usually requires a lot more evaluations than SA (Azadivar 1999; Fu and Healy 1992), (3) similar to IPA and LR, it is problem-specific (due to the need explicit models), (4) its effectiveness is highly dependent on the ability to develop explicit models that allow the calculation of first- and second-order derivatives (usually required by deterministic nonlinear optimization techniques), and (5) The solutions provided by SPO methods may not be optimal as this technique solves the problem: $E[\min_{\theta \in \Theta} L(\theta, \omega)]$ instead of the desired problem: $\min_{\theta \in \Theta} E[L(\theta, \omega)]$.

### *20.2.4 Metaheuristics*

A metaheuristic is a general framework consisting of black-box procedures that can be applied to different kinds of problems without significant changes. The applicability of these techniques to problems with continuous or discrete decision spaces is dictated by the particular structure of the method and the way in which it is adapted to the problem at hand. Metaheuristics are the dominant strategies used in commercial optimization software for simulation optimization (Fu 2002), as well as in the solution of large-scale problems. This is due to the fact that many of the methods mentioned above are model-dependent and/or require a high level of expertise for their implementation. In this section, we provide a short description of the metaheuristics available for simulation optimization and a set of relevant references. Special attention is given to the most commonly used methods, genetic algorithms (GA), tabu search (TS), and scatter search (SS).

#### 20.2.4.1 Pattern Search

Pattern search methods are sequential algorithms that move from iteration to iteration based on some characteristic or pattern in the observations, instead of relying on gradients or randomization. Conceptually, these techniques try to use some form

of memory but at a very basic level and are mentioned here mostly for historical purposes. The most important techniques in this class are:

1. The Hooke and Jeeves method (1961), which is based on the idea that if a direction has produced an improvement in the estimated performance measure, then one should continue moving in that direction,
2. The simplex method (Jacobson and Schruben 1989) and references therein), not to be confused with the classical algorithm for linear programming, which compares the estimated performance measures from an initial set of possible solutions, eliminates the worst performer, and replaces it by a new one determined by the centroid of the remaining solutions, and
3. The complex method (Azadivar (1999) and references therein), which is the simplex method modified to handle constrained problems.

#### 20.2.4.2 Genetic Algorithms (GA)

The set of GA is one of a class of algorithms inspired by the biological principles of evolution known as evolutionary algorithms. This technique searches for the optimal decision vector, $\theta$, based on a performance measure (fitness function, in GA terminology), by iteratively updating a population of good decision vectors. The decision vector associated with each member of the population is encoded as a string of symbols (genes) that form a chromosome, and is generated from the members of the population in the previous iteration through random genetic operators (Sect. 20.3.2 provides a specific example). In general, the algorithm can be described as follows:

*Step 1*. Initialize the population with a set of members generated using previous knowledge of the problem and/or a random process, and estimate their performance according to the chosen fitness function. The number of simulations required for the estimation of the performance measure is determined by a stopping criterion such as confidence intervals, convergence efficiency, or computational budget.

*Step 2*. Create new chromosomes (reproduction) by using genetic operators. The best known operators are crossover and mutation. Crossover consists in the random exchange between two members (parents) of part of their chromosomes, and mutation is a random alteration of some of the genes in a given member.

*Step 3*. Estimate the fitness function of the newly created chromosomes and select from this group and the population in the previous iteration (generation) the members of the next generation based on the superiority of their estimated performance measures.

*Step 4*. Check for "convergence": stop or go to Step 2. Genetic algorithms are not guaranteed to converge; therefore, the definite termination condition is usually specified as a maximal number of generations or an acceptable fitness level for the best individual.

It is important to note that though the general structure of the algorithm always follows these steps, specific procedures for encoding, initialization, reproduction, and selection can be chosen based on the problem at hand to enhance the performance of the algorithm. Reeves and Rowe (2003) provide an excellent guide to the GA technique.

#### 20.2.4.3 Scatter Search (SS)

Similar to GA, scatter search is a population-based evolutionary algorithm. However, it uses a completely different approach for the generation of new population members (decision vectors). Specifically, the members of the population (called the reference set) are combined in a systematic way, instead of randomly. The combination strategies are generalized forms of linear combinations that consider at least all pairs of members in the reference set. SS also differs from GA in the size of the population; reference sets tend to be small compared with the populations used in GA. In general, SS algorithms can be described as follows (Laguna 2003):

*Step 1*. Generate a starting set of decision vectors as diverse as possible and apply heuristics to these vectors in an attempt to improve their performance. From the resulting population, choose the vectors with the best estimated performance to be part of the initial reference set. Notice that the notion of "best" is not only limited to the value of the performance measure; a solution may be added to the reference set if it improves the diversity of the set, regardless of the performance measure.

*Step 2*. Create new members consisting of systematic generalized linear combinations of two or more members of the current reference set.

*Step 3*. Apply the heuristic process used in Step 1 to improve the members created in Step 2.

*Step 4*. Extract a collection of the "best" improved solutions from Step 3 and use them to replace the worst performing members in the reference set. If the reference set does not change, stop. Otherwise go to step 2.

Laguna (2003) provides a complete presentation of this methodology, including the different member combination strategies available and their suitability according to the type of problem at hand.

#### 20.2.4.4 Tabu Search (TS)

As in random search (Sect. 20.2.2.1), TS explores the solution space by moving successively from one feasible solution to a neighboring one. However, instead of using probabilistic arguments to guide the search, it uses a strategy based on the ideas of adaptive memory and responsive exploration. This means that TS redefines the solution neighborhood at each iteration based on the information previously collected to avoid visiting already explored areas or areas characterize by poor performance.

The method accomplishes this by selecting certain attributes or combination of attributes that cannot be part of the new solutions (are tabu). The memory structure used in TS uses two types of information, namely, explicit and attributive. The explicit part is captured by recording good solutions or highly attractive but unexplored neighborhoods of those good solutions; while the attributive part records information about solution attributes that change in going from one solution to another (e.g., increase in the risk level of a portfolio of projects). Glover and Laguna (1997) provide an exhaustive presentation of the concepts and applications of TS.

### *20.2.5 Other Methods*

In addition to the methods just described, there are simulation-optimization techniques which, in spite of not being widely used at present, could be viable options for specific problems or could become so as they are further developed. This group of methods includes neural networks (Glover et al. 1999), branch and bound for discrete systems (Norkin et al. 1998), nested partitions (Shi and Olafsson 2000), and the collection of algorithms known as model-based methods. Model-based methods, instead of generating actual solutions, construct probability distributions for the solution space that can be used to estimate where the best solutions are located. The following techniques belong to this group: swarm intelligence, estimation of distribution algorithms (EDAs), the cross-entropy (CE) method and model reference adaptive search (Fu et al. 2005).

## 20.3 Two Industrial Problems

In this section two case studies based on actual industrial problems are presented to illustrate the potential of simulation optimization as a decision support tool. The presentation of each case study includes a short description of the problem, a discussion supporting the selection of a specific simulation-optimization method, and a summary of the implementation of the method and the results obtained.

### *20.3.1 Inventory Management*

Any enterprise that manufactures products faces uncertainties in a range of factors such as demand, prices and availability of raw materials, production lead times, currency exchange variability, etc. Some of these factors directly affect the profitability of the enterprise by limiting the operating margins, while others have an indirect impact such as inability to meet customer needs or the accumulation of excess inventory. The inability to meet customer needs results in both loss of “here and now”

and long-term profit as poorly served customers may not come back. Therefore, in any industrial setting, customer satisfaction level (CSL; the expected value of the ability to meet customer demand) is recognized as an important performance measure. A high level of customer satisfaction can be achieved by maintaining high inventories to hedge against uncertainty (e.g., fluctuations in demand or availability of raw material). However, additional inventory entails increased holding cost (including opportunity cost of invested capital and warehouse space). Decision makers attempt to minimize the impact of this trade-off between customer satisfaction and inventory holding cost on the profitability of the enterprise by specifying different safety stock levels for each product across the supply chain.

A great deal of work has been done to develop analytical strategies that allow the determination of the optimal allocation of safety stocks (Jung et al. 2004). However, those strategies fall short when the enterprise manufactures multiple products that share production facilities with limited capacity and scheduling constraints, experience significant queue effects and lead times, and faces uncertain demand from several customers. This kind of environment is common to many industrial and pharmaceutical manufacturers, including the particular case we were confronted with.

We looked into the operation of the supply chain of a major US polyethylene producer whose main source of uncertainty is demand and who wanted to reach specific levels of customer satisfaction. The company uses a decision-making strategy in which CSLs are specified by top management according to certain strategic considerations and aggregated data, while minimization of the cost of delivering the products is left to planners and the people in operations. Thus, the problem to be addressed is the determination of how much, where and when to produce, and the safety stock levels for each product. The company has two production sites, which have different layouts and capacities that directly supply the seven sales regions into which USA is divided. It produces five types of polyethylene (A, B, C, D, and E) in ten different grades (0–9), in two types of packaging (box or bag) for a total of 100 (5 types $\times$ 10 grades $\times$ 2 packages) stock keeping units (SKUs). The demand for each SKU is characterized as a normal distribution, whose mean value changes on a weekly basis according to internal forecasting models.

The first step in developing a simulation-optimization strategy for a problem is to determine which group of techniques (continuous or discrete, and small or large number of feasible solutions) is appropriate according to the characteristics of the solution space and the limitations of each method. In this case, it is clear that the inventory levels can take any integer value, which due to the combinatorial nature of the problem rules out any of the algorithms that fall under the "small number of discrete feasible solutions" class. The discrete character of the decision space could be also used to disregard the methods for systems with continuous decision spaces, but the levels of inventories required by an operation like the one here considered are high enough that the use of such techniques in conjunction with rounding needs to be considered as they may provide near optimal solutions. In the remaining class, "large number of discrete feasible solutions," ordinal optimization and SPO can be ignored. The first method is disregarded due to the size of the decision space, and the second due to our inability to develop an explicit model that characterizes

the performance measure (customer service level). The lack of an explicit model is also the reason to disregard the methods for continuous decision spaces SA+PA and SA+LR. The SA+FDA also has to be disregarded as it requires the system to reach steady state, a condition that is not achievable in this problem due to the seasonal demand fluctuations. This leaves us with the following set of potential solution methods: random search, SA+FD, SA+RSM, metaheuristics, and any of these four methods in combination with one of the methods under the "small number of discrete feasible solutions" class.

Once the options have been narrowed down based on the solution space and the limitations of the methods, the selection process has to be driven by the strengths of the remaining options with regard to the problem at hand. For the problem considered here it is important to understand the connection between CSL, defined as service level, the production strategy, and the safety stock level of a product under uncertain demand. Over a given range of demand variance there are three possible operational regimes. In regime I, production facilities have sufficient spare capacity to cope with any change in demand. Therefore, in this regime, a relatively low or even zero safety stock level may be sufficient to achieve the desired customer satisfaction. In regime II, the production capacity maybe quite strained when the demand for different products spike at some point in time. In this regime, if there is not enough safety stock, the CSL for some products sharing production facilities may fail to reach their target values. Finally, in regime III, the capacity available cannot satisfy the combined expected demands of the different products. In this regime, the safety stock and production resources must be assigned strategically to meet the demands of some customers in preference to the others. For the problem at hand, the sites owned by the company have enough capacity to operate in regimes I and II. This means that no customer priority has to be used to allocate production capacity and therefore any desired level of inventory for each product is realizable. Notice that this condition and the hierarchical decision-making strategy used by the company (i.e., tactical decisions such as service level dictate operational goals) allows for the use of a decomposition strategy. The idea is to use a multilevel optimization approach instead of an integrated approach in which production quantities along the time horizon and safety inventory levels are considered together in a massive stochastic program. The multilevel strategy is composed of a simulation-optimization strategy that determines the optimal stock levels based on long-term customer satisfaction, and deterministic (expected values) rolling horizon planning and scheduling optimizations, embedded in the simulation, which allocate production resources by minimizing cost. Figure 20.2a illustrates the "outer" optimization on the safety stock levels, and Fig. 20.2b the inner problem in which the simulation of the system constantly interacts with the planning and scheduling models in a rolling horizon fashion. The planning model is formulated as an LP for a 3-month horizon that takes into account production, transportation, inventory holding, and shortage costs; whereas the schedule is generated for 40 days using the VirtECS scheduling software (Advanced Process Combinatorics Inc., 2004).

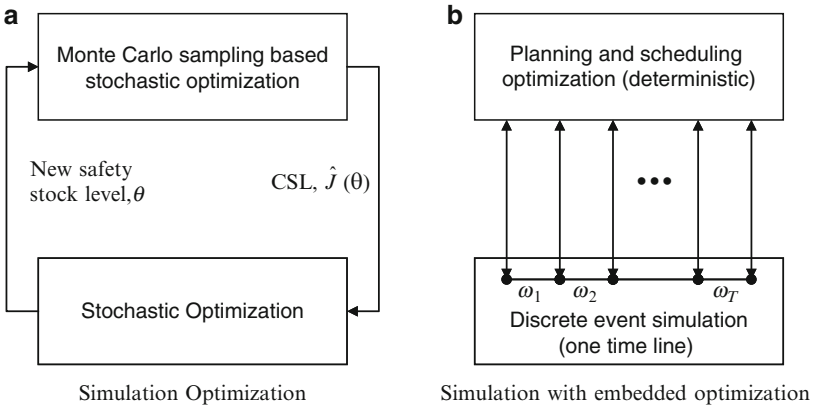


**Fig. 20.2** Configuration of simulation and optimization strategies

The outer optimization problem can be mathematically represented as follows:

$$\min_{\theta} J(\theta) = \sum_{i=1}^{100} \mu_i \left| L_i^{\Delta}(\theta) \right| \tag{20.31}$$

subject to

$$L_i(\theta) + L_i^{\Delta}(\theta) \geq L_i^{\text{target}} \quad \forall i \tag{20.32}$$

where $\mu_i$ is the penalty for missing the target CSL for product i, $\theta = (\theta_{11}, \ldots, \theta_{is})$ is the decision vector including the safety stock levels of each product i in each production facility $s$, $L_i(\theta)$ is the CSL (expected value of the probability of fully meeting every demand for product i), and $L_i^{\Delta}(\theta)$ is the deviation with respect to the target CSL, $L_i^{\text{target}}$. Notice that the CSLs are the only variables in the objective function (20.31). This condition combined with the fact that the level of customer satisfaction is a monotonic increasing function of $\theta$ (the larger the safety stock the higher the customer satisfaction), implies that the best local adjustment to each decision variable has to be inversely proportional to the magnitude of the penalty resulting from deviating from the target CSL, $\mu_i \left| L_i^{\Delta}(\theta) \right|$. Though the adjustment is local in the sense that it does not consider the effects and constraints associated with the embedded planning and scheduling problems, the monotonic nature of CSL($\theta$) guarantees convergence to a global optimal solution if the estimator of $L_i(\theta)$ is unbiased. Therefore, an efficient simulation-optimization strategy for this problem should be capable of exploiting the fact that if the performance measure improves in a particular direction, then one should continue moving in that direction. The only method in the shortlisted group capable of doing that is the pattern

search metaheuristic. This metaheuristic was then selected and implemented in the recursive algorithm below:

*Step 1*. Initialize safety stock levels, $\theta_{is}^{n}$, where $n = 0$ for all $i$ and $s$

*Step 2*. Estimate $J_i\left(\theta^n\right)$ and $L_i^{\Delta}\left(\theta^n\right)$ using simulation

*Step 3*. Check for convergence of the estimated performance measure - if $\left|\hat{J}\left(\theta^n\right) - \hat{J}\left(\theta^{n-1}\right)\right| \leq \varepsilon$ stop. Otherwise, continue

*Step 4*. Calculate the new safety stock level

$$\theta_{is}^{n+1} = \theta_{is}^{n} + \alpha\beta_{is}\left(\mu_i\,\hat{L}_i^{\Delta}\left(\theta\right)\right)$$

where $\alpha$ is a step size factor that can be adjusted by trial and error, and $\beta_{is}$ is the distribution factor, which represents the ratio of product supply from each production site in the previous iteration

*Step 5*. Check for convergence, if $\left|\hat{J}\left(\theta^n\right) - \hat{J}\left(\theta^{n-1}\right)\right| \leq \varepsilon$ stop. Otherwise, go to Step 2.

The algorithm was used to solve a case in which the coefficient of variation of the different demands was assumed to be 30%. Figure 20.3, where $Ai - x$ denotes the

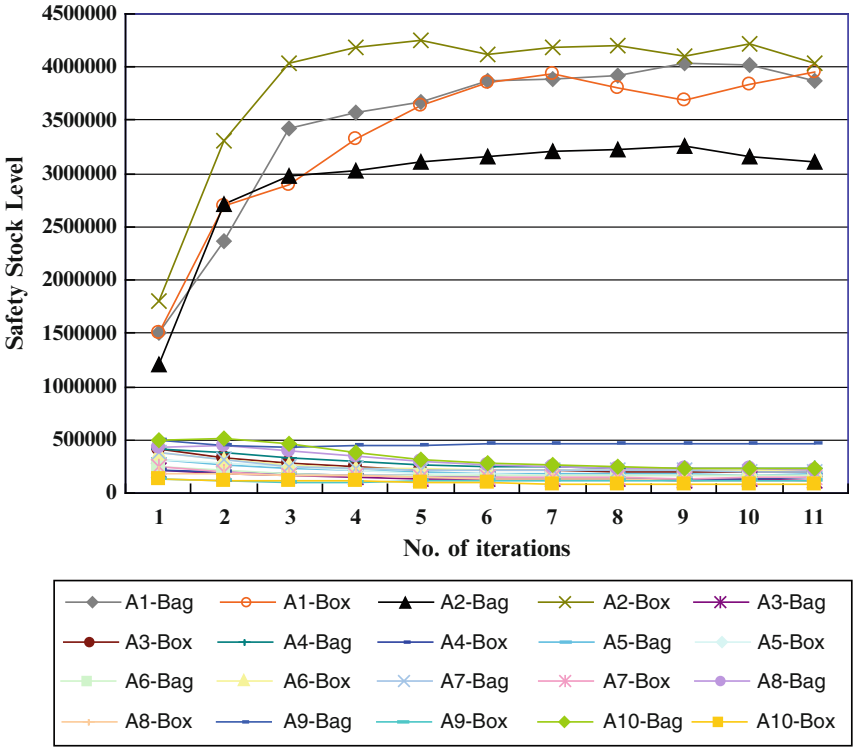


**Fig. 20.3** Trajectory of safety stock levels of A type products at plant 1 as the optimization proceeds

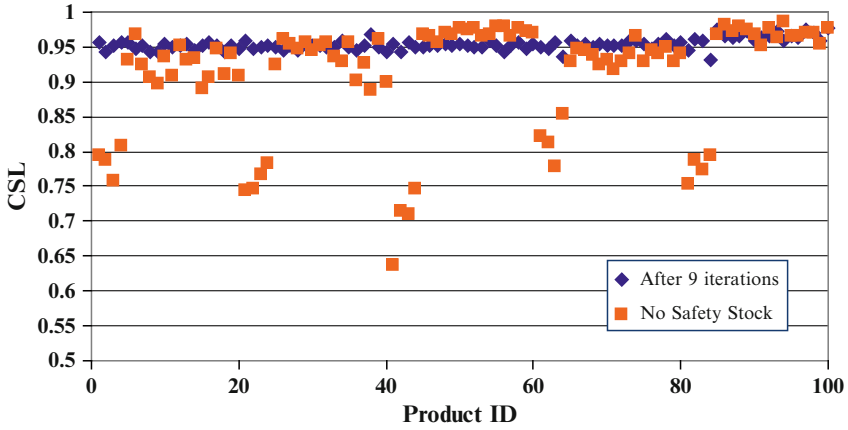


**Fig. 20.4** Comparison of the Customer Satisfaction Levels (CSL) with and without safety stock

final product of type A and grade i packaged in facility x, shows the iterative process for the safety stocks of the type A products in one of the production facilities when the starting values are zero. It is important to note that four products, A0-bag, A0-box, A1-box, and A1-bag, make up 80% of the demand for type A and the rest, from A2-bag to A9-box, make up the remaining 20% (the same is true for the rest of the polyethylene types). As expected, the products with a larger demand need higher safety stocks in order to cope with the 30% variability. Figure 20.4 summarizes the estimated CSLs without safety stock, $\hat{L}_i\ (\theta^0)$, and after nine iterations of the algorithm $\hat{L}_i\ (\theta^9)$, showing the efficiency of the computational framework in solving the outer optimization problem. Notice that the change is more pronounced in the group of major products (the first four type-grade-package triplets that take 80% of the demand) which go from the 0.6–0.8 range to levels very close to the 0.95 target, and in some of the minor products that show lower CSLs in the presence of safety stock. The latter counterintuitive result can be attributed to the additional strain imposed on production by the increase in the safety stock levels of the major products.

### *20.3.2 Portfolio Selection of New Compounds to be Developed in the Pharmaceutical Industry*

The hierarchical decision-making strategy mentioned in the previous problem is not only used when dealing with tactical (e.g., set service level) and operational decisions (e.g., set safety stock levels), but also when strategic decisions need to be made. This means that strategic decisions are usually made based on aggregated data, representing the capacity of the organization at lower levels,

and those decisions are pushed down as fixed goals. Though such a hierarchical approach provides solutions close to the optimal one when the system has low levels of uncertainty; that is rarely the case in highly uncertain and constrained environments. A good solution at the tactical and operational levels can be obtained for the specific goals dictated by the strategic decision makers, but the quality of these goals with respect to the attainable optimum remains unknown. Such a situation does arise in the context of pharmaceutical products development.

The selection of a portfolio of drugs to be developed is a strategic decision that has uncertain financial implications on the order of billions of dollars which are only realized over the long term (decades). This decision-making process is further complicated by the low probability of success of new compounds (high attrition rates), unpredictable changes in regulations, technologies and health trends, dependencies between projects (drugs) from a variety of perspectives, uncertainties in terms of duration and cost in each stage of the development process, and limited human and capital resources. In addition, as in any other kind of portfolio there are solutions that have the same exposure to risk, but a different level of rewards. Therefore, the problem to be addressed consists in choosing a prioritized portfolio on the reward-risk-efficient frontier (i.e., the portfolios with the maximum level of rewards for a given level of risk) for the level of risk considered acceptable by the enterprise. Notice that such a selection, in addition to being influenced by all the uncertainties mentioned above, is constrained by the limited amount of renewable (e.g., equipment) and nonrenewable resources (e.g., budget for clinical trials), and the strategies used by decision makers at the tactical and operational levels to allocate them. Therefore, the optimization strategy has to be able to capture the impact of these constraints on the behavior of the system.

There are three major stages in the lifecycle of a new drug, which are: discovery, development and commercialization. The discovery stage tends to be highly unpredictable and case specific, while the other two follow a well-defined path. This situation, coupled with the limited availability of the renewable and nonrenewable resources necessary to simultaneously develop all the compounds rated as promising by discovery (lead molecules), has directed all the attention, from a modeling and optimization perspective, to the development and commercialization stages. Once a molecule is promoted to the status of a lead molecule, it goes through a network of tasks similar to that shown in Fig. 20.5. Though small variations in the drug development lifecycle occur from company to company, Fig. 20.5 depicts a fairly realistic model of what happens in this kind of industry. In the figure, tasks are represented by rectangles, while decision points are presented as diamonds. In general, these tasks can be classified into two groups, evaluation and commercialization, and manufacturing. The purpose of the tasks in the first group (upper row in Fig. 20.5) is to determine the safety and efficacy of the drug and satisfy all requirements to make it commercially available if these two aspects are favorable. The second group (lower row in Fig. 20.5) encompasses all the tasks necessary to scale up the laboratory procedures into commercial size manufacturing facilities. A complete explanation of the activities covered by each task can be found in Blau et al. (2004). We examined

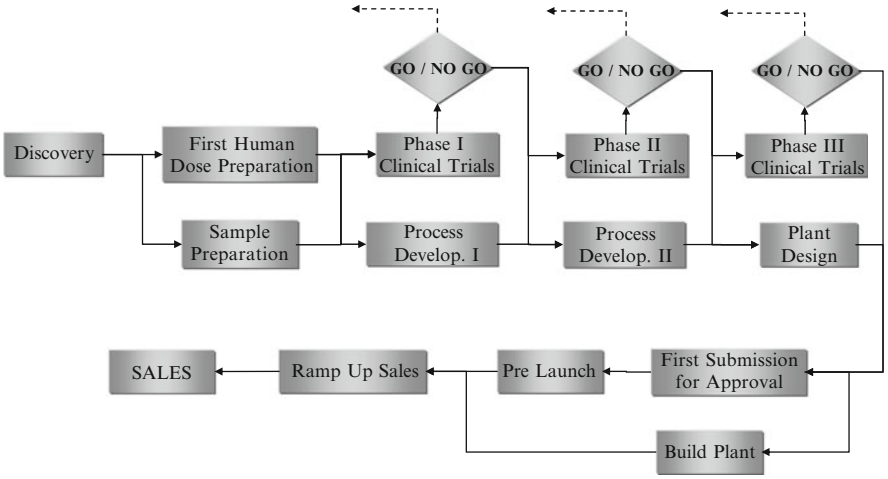


**Fig. 20.5** Schematic of a pharmaceutical R&D pipeline model

the portfolio of a US-based pharmaceutical company that had a total of nine lead compounds with a 20-year patent protection whose development process can be approximated by the model in Fig. 20.5.

As in the previous case study, the first step in determining a suitable simulation-optimization method for the problem is to narrow down the options based on the characteristics of the solution space. The fact that a group of compounds and their corresponding priorities need to be selected from a finite set eliminates all techniques under the "continuous decision space" class. The number of potential strategies can be further reduced by taking into account the combinatorial nature of the problem. The nine compounds and their priorities can be mixed and matched into almost one million different permutations, ruling out any strategy in the "small number of feasible solutions" class, and ordinal optimization. SPO is also disregarded due to our inability to develop a model that characterizes the performance measure in terms of the decision variables. This leave us with the following set of potential strategies: random search, metaheuristics, and any of these two in combination with one of the methods under "small number of discrete feasible solutions." The final choice of a method is driven by the strengths of the remaining options relative to the problem at hand. In portfolio problems, the desired outcome is not just a single optimal point but a characterization of the efficient reward-risk frontier. Hence, the use of a random search method, though feasible, would be highly inefficient as it would be necessary to run it multiple times to construct the efficient frontier. With this point of departure, a trial and error process was implemented to find a metaheuristic capable of solving the problem. Tabu search was examined, but was discarded as it was not possible to stop the method from getting stuck in certain areas of the solution space. In the second iteration, a GA was tested with excellent results. This method was selected not only because it provided the desired output

(i.e., an efficient frontier), but also because it allowed a natural representation for the decision variable, a vector of prioritized projects. There is currently no formal structured way to select a metaheuristic; it is more an art than a science. Though some directions are provided in the references provided in this chapter, the black box nature of these approaches makes their performance unpredictable.

Before describing the GA in detail, it is important to point out the modeling assumptions and simplifications used in the case study. The model only considers the uncertainty generated by the probabilities of success/failure at the end of the clinical trials, which are modeled by Bernoulli distributions. The rest of the potentially uncertain variables (costs, sales per year, and task durations) are approximated with their mean values. These model simplifications were necessary not due to limitations in the optimization framework but due to the lack of reliable information to characterize those uncertainties. The model also captures four types of dependencies between projects, which are as follows: (1) resource dependencies, (2) manufacturing cost dependencies, (3) financial return dependencies, and (4) technical success dependencies. Learning curve effects frequently lead to resource dependencies. A common example occurs when the development times are reduced for the trailing candidate of two functionally similar drug types. Cost dependencies occur when the combined cost of a development activity for two drug candidates is less than the sum of their individual costs because of resource sharing. For example, it may be possible to use the same production facilities for two chemically or biologically similar drug candidates. Financial return dependencies occur when there is synergism or competition in the marketplace. For example, cannibalization can occur when two drug candidates are aimed at developing products that compete with each other in the marketplace. Technical dependencies occur when the technical success or failure of a drug candidate affects the probability of technical success of an as-yet-untested trailing drug candidate. For example, two drug candidates might be developed to release an active ingredient in a controlled fashion. If the precedent candidate is successful, the probability of success of the as-yet-untested second candidate will be increased. The specific realizations of the dependencies considered in this problem are described by Blau et al. (2004).

The final consideration for the model is the representation of the strategy used to allocate and reallocate resources after a project failure and at the end of each year. The resource allocation policies were obtained following the framework conceived by Varma (2005), which uses a simulation of the task network in Fig. 20.5 and an observer. The simulation includes an integer program (IP) for short-term resource allocation that can assign three different levels of resources (associated with specific durations) to each task, namely, most likely (ML) value, and a certain percentage below and above of the most likely value. The observer tallies each of the outputs from the IP and determines the allocation policies by relating the most frequent decisions observed to the corresponding realization of the pipeline state space. This minimizes the size of the state space (composition of the portfolio and development stage of each compound) while keeping as much information as possible by breaking it into drug states $S_i = \{\mathrm{DS}_i, \mathrm{NLEV}_i, \mathrm{NHEV}_i\}$, where DSi is the development stage of drug i, NLVEi, the number of drugs having lower expected value than drug

i in the same development stage, and NHEVi is the number of drugs having a higher expected value than drug i in the same development stage.

The optimization problem to be solved by the GA can be mathematically expressed for the specific case in which rewards are measured by the expected positive net present value (EPNPV) and risk by the probability of losing money ($P$ (NPV($\theta$) < 0)) as:

$$\min_{\theta} J\ (\theta) = \text{EPNPV}\ (\theta) \tag{20.33}$$

subject to

$$P(\text{NPV}(\theta) < 0) < \beta \tag{20.34}$$

where NPV is the net present value, $\theta$ is the prioritized portfolio of drugs, and $\beta$ is the upper bound for the probability of losing money that needs to be varied in the (0, 1) interval to obtain the efficient frontier.

The GA is encoded such that each gene contains the number of a drug candidate (with 0 indicating that a project was not selected), and its position in the chromosome represents the priority given to the compound. For example, the chromosome 203000400 corresponds to a portfolio that consists of three compounds: 2, 3 and 4, of which compound 2 has the highest priority. A fitness function Zk of the following form is used:

$$Z_k = \alpha \left( \frac{\text{EPNPV}_k - \text{EPNPV}_{\min}}{\text{EPNPV}_{\max} - \text{EPNPV}_{\min} + \gamma} \right) + (1 - \alpha) \left( \frac{\text{Risk}_{\max} - \text{Risk}_k}{\text{Risk}_{\max} - \text{Risk}_{\min} + \gamma} \right) \tag{20.35}$$

where EPNPVmin and EPNPVmax are the minimum and maximum expected positive net present values, respectively, in the current population; $\text{Risk}_{\min}$ and $\text{Risk}_{\max}$ are the maximum and minimum risk levels in the current population, measured as the probability of losing money, $\gamma$ is a small positive number that prevents division by zero, and $\alpha$ weights the present value vs. the level of risk in a convex linear combination. The GA proceeds to find chromosomes that improve the fitness function by generating new chromosomes through the use of some genetic operators and estimating the fitness function values using simulations of the model in Fig. 20.5 (Zapata et al. 2008). Notice that the NPV and PNPV for each simulation can be calculated from the discounted development costs accumulated as the compounds move through the pipeline and the returns realized when the drug hits the market.

The GA was run for different percentages of the amount of resources that can be allocated above or below the ML value, including a base case in which reallocation of resources was implemented based on the original priorities given by the GA sequence (i.e., no information about realized uncertainties and the state of the pipeline is used) and no flexibility in the quantity of resources was considered. Figure 20.6 presents the results for the base case. All the points corresponding to the maximum EPNPV for a given level of risk are linked to form an approximate reward-risk-efficient frontier. At first sight, it looks like its shape reflects the

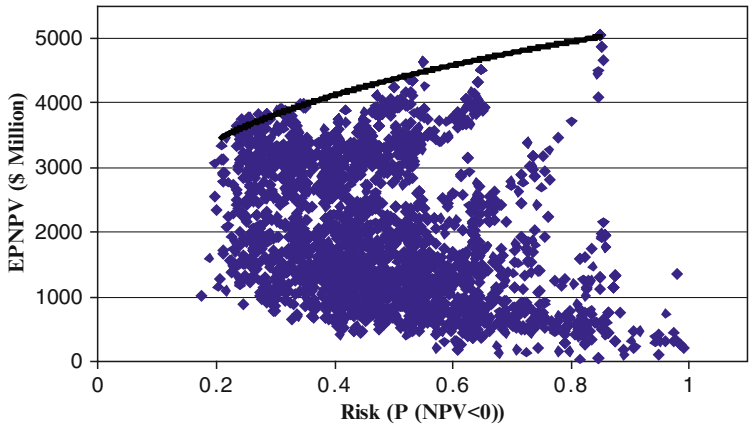


**Fig. 20.6** Efficient frontier base case

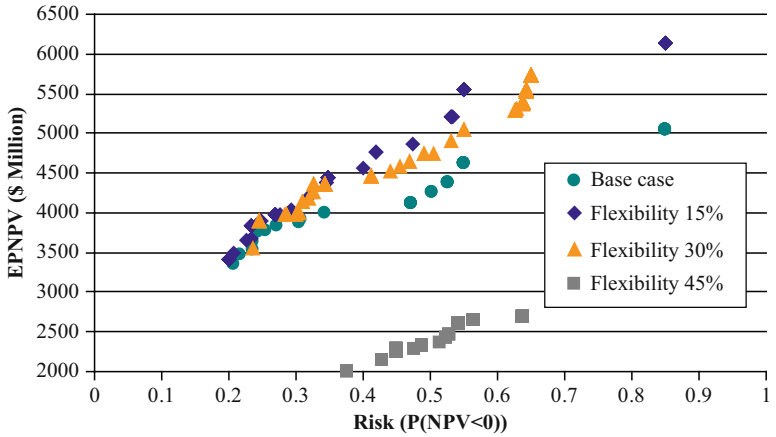


**Fig. 20.7** Efficient frontiers for with and without dynamic resource allocation

general form found by Markowitz in financial portfolios (Luenberger 1998), but a closer look reveals that the direct correlation between rewards and risk is violated in the middle section; the depression in the efficient frontier implies that there are efficient portfolios which bear more risk but result in lower rewards). This counterintuitive result was not observed when flexibility in allocating resources was considered. Figure 20.7 shows the dominating portfolios for the three different dynamic resource allocation cases considered. The compositions of the portfolios on the efficient frontier in the base case and those with dynamic resource allocation are remarkably different in the region where the depression is found. These results are significant as they reveal that it is not possible to decouple the strategic

and tactical decision-making processes without becoming substantially suboptimal. Therefore simulation-optimization strategies like the one here presented are essential to be able to accurately model the system and optimize it to improve the quality of the decisions made.

However, it is important to highlight that the computational burden required to solve the problem was very high. It took between 3 and 5 days on a 64 bit Sun-Sparc Ultra-Enterprise with 25, 400 MHz processors, and 8 M CPU cache per processor to run each case. This burden is bearable if we consider that these kinds of decisions are commonly made every 6 months, but would be unacceptable in a decision-making process that has to be repeated with a much greater frequency.

## 20.4 Conclusions

A summary of the simulation-optimization methods currently available was provided. Our discussion was organized by classifying methods into those intended for small discrete, large discrete, and continuous decision spaces. In the first category, the number of feasible solutions is small and therefore the focus of the methods is on the exhaustive comparison of possible solutions through statistical inference. The size of large discrete and continuous decision spaces shifts the focus to methods based on search algorithms, with the exception of ordinal optimization that uses statistical inference to exhaustively compare possible solutions. The majority of the methods in these two categories are model-independent and therefore can be applied to any problem. However, this very advantage is responsible for slow convergence rates (random search), unpredictable convergences (RSM and metaheuristics), and high computational burden (SA with FD and RSM). Though in principle two methods, SA with SP (for any system) and SA with FDA (for systems in steady state), are immune to these issues, the difficulty in parameterizing them results for the most part in slow convergence rates during execution. By contrast, the three model-dependent methods, SA with PA and LR and some types of SPO, tend to exhibit a faster convergence but are applicable to a limited number of very simple problems. At the end, the selection of a method for most problems is more an art than a science and requires a significant amount of trial and error. This situation has led practitioners to mainly use metaheuristics (especially GA and SS) and RSM due to their flexibility to accommodate any type of problem and their relative simplicity.

The chapter also presented two industrial case studies, in which simulation-optimization methods were successfully used. The case studies served to illustrate not only the implementation of a few methods, but also to highlight some of the considerations that are relevant in the selection of a method. From these case studies and the initial discussion in this chapter is evident that simulation optimization is the right tool to support several complex industrial decision-making processes.

However, in general, simulation optimization requires a significant level of technical sophistication from the user, especially in the area of statistics, as well as large amounts of computational resources.

## References

Alrefaei MH, Andradottir S (1999) Simulated annealing algorithm with constant temperature for discrete stochastic optimization. Manage Sci 45:748–764.

Andradottir S (1998) Review of simulation optimization techniques. Presented at 1998 Winter Simulation Conference, Washington, DC, USA.

Azadivar F (1999) Simulation optimization methodologies. Presented at 1999 Winter Simulation Conference, Phoenix, AZ, USA.

Banks J (1998) Handbook of simulation:principles, methodology, advances, applications, and aractice. Wiley, New York.

Banks J (2005) Discrete-event system simulation. 4th edn. Pearson Prentice Hall, Upper Saddle River, NJ.

Bhatnagar S et al (2003) Two-timescale simultaneous perturbation stochastic approximation using deterministic perturbation sequences. ACM Trans Model Comput Simul 13:180–209.

Blau GE et al (2004) Managing a portfolio of interdependent new product candidates in the pharmaceutical industry. J Prod Innov Manage 21:227–245.

Fu MC (1994) Optimization via simulation: a review. Ann Oper Res 53:199–247.

Fu MC (2002) Optimization for simulation: theory vs. practice. INFORMS J Comput 14:192–215.

Fu MC, Healy KJ (1992) Simulation optimization of inventory systems. Presented at 1992 Winter simulation conference, Arlington, VA, USA.

Fu MC, Healy KJ (1997) Techniques for optimization via simulation: an experimental study on an (s, S) inventory system. IIE Trans (Institute of Industrial Engineers) 29:191–199.

Fu M, Hu J-Q (1997) Conditional Monte Carlo: gradient estimation and optimization, applications. Kluwer Academic, Boston.

Fu MC, Glover FW, April J (2005) Simulation optimization: a review, new developments, and applications. Presented at 2005 winter simulation conference, Orlando, FL, USA.

Glover F, Laguna M (1997) Tabu Search. Boston, MA: Kluwer Academic.

Glover F, Kelly JP, Laguna M (1999) New advances for wedding optimization and simulation. Presented at 1999 winter simulation conference, Phoenix, AZ, USA.

Hall JD, Bowden RO (1996) Simulation optimization for a manufacturing problem. Presented at Southeastern simulation conference, Huntsville, AL, USA. Society for Computer Simulation.

Hazra MM, Morrice DJ, Park SK (1997) Simulation clock-based solution to the frequency domain experiment indexing problem. IIE Trans (Institute of Industrial Engineers), 29, 769–782.

Healy, K. and Schruben, L.W (1991) Retrospective simulation response optimization. Presented at 1991 winter simulation conference, Phoenix, AZ, USA.

Henderson, S.G.and Nelson, B.L (2006) Handbooks in operations research and management science: simulation. Elsevier, Amsterdam.

Ho Y-C, Cao X-R (1991) Perturbation analysis of discrete event dynamic Systems. Kluwer Academic, Boston, MA.

Ho YC et al (1992) Optimizing discrete event dynamic systems via the gradient surface method. Presented at 30th IEEE conference on decision and control part 1 (of 3), Brighton, England.

Hooke R, Jeeves TA (1961) Direct search solution of numerical and statistical problems. J ACM 8:212.

Jacobson SH, Schruben LW (1989) Techniques for simulation response optimization. Oper Res Lett 8:1–9.

Jacobson SH, Schruben L (1999) Harmonic analysis approach to simulation sensitivity analysis. IIE Trans (Institute of Industrial Engineers) 31:231–243.

Jacobson SH, Buss AH, Schruben LW (1991) Driving frequency selection for frequency domain simulation experiments. Oper Res 39:917.
Jung JY et al (2004) A simulation based optimization approach to supply chain management under demand uncertainty. Comput Chem Eng 28:2087–2106.
Kiefer JC, Wolfowitz, J (1952) Stochastic estimation of the maximum of a regression function. Bull Am Math Soc 58:465–465
Kleijnen JPC, Rubinstein RY (1996) Optimization and sensitivity analysis of computer simulation models by the score function method. Eur J Oper Res 88:413–427.
Kushner HJ, Yin G (2003) Stochastic approximation and recursive algorithms and applications. 2nd edn. Applications of mathematics, vol. 35. Springer, New York xxii, p. 474.
Laguna M, Martâi R (2003) Scatter search: methodology and implementations in C. Kluwer Academic, Boston, MA.
Luenberger DG (1998) Investment science. Oxford University Press, New York.
Nelson BL, Matejcik FJ (1995) Using common random numbers for indifference-zone selection and multiple comparisons in simulation. Manage Sci 41:1935.
Norkin VI, Pflug GC, Ruszczynski A (1998) A branch and bound method for stochastic global optimization. Math Program 83:425–450.
Nozari A, Morris JS (1984) Application of an optimization procedure to steady-state simulation. Presented at 1984 winter simulation conference, Dallas, TX, USA.
Pichitlamken J, Nelson BL (2003) A combined procedure for optimization via simulation. ACM Trans Model Comput Simul 13:155–179.
Reeves CR, Rowe JE (2003) Genetic algorithms: principles and perspectives: a guide to GA theory. Kluwer Academic, Boston, MA.
Rubinstein RY, Shapiro A (1993) Discrete event systems: sensitivity analysis and stochastic optimization by the score function method. Wiley Chichester, NY.
Sadegh P, Spall JC (1998) Optimal random perturbations for stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Trans Autom Control 43:1480–1484.
Safizadeh MH (1990) Optimization in simulation. Current issues and the future outlook. Naval Res Logist 37:807–825.
Shapiro A (1996) Simulation based optimization. Presented at 1996 winter simulation conference, Coronado, CA, USA.
Shi L, Olafsson S (2000) Nested partitions method for global optimization. Oper Res 48:390–407.
Spall JC (1992) Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. IEEE Trans on Autom Control 37:332–341.
Spall JC (1998) Implementation of the simultaneous perturbation algorithm for stochastic optimization. IEEE Trans Aerospace Electron Syst 34:817–823.
Spall JC (1999) Stochastic optimization and the simultaneous perturbation method. Presented at 1999 winter simulation conference, Phoenix, AZ, USA.
Swisher JR et al (2004) A survey of recent advances in discrete input parameter discrete-event simulation optimization. IIE Trans 36:591–600.
Tayur S, Ganeshan R, Magazine M (1999) Quantitative models for supply chain management. Kluwer Academic, Boston, MA.
Varma VA (2005) Development of computational models for strategic and tactical management of pharmaceutical R&D pipelines. PhD Thesis, Purdue University.
Varma VA et al (2007) Enterprise-wide modeling and optimization an overview of emerging research challenges and opportunities. Comput Chem Eng 31:692–711.
Wan X, Pekny JF, Reklaitis GV (2005) Simulation-based optimization with surrogate models application to supply chain management Comput Chem Eng 29:1317–1328.
Wan X, Pekny JF Reklaitis GV (2006) Simulation based optimization for risk management in multi-stage capacity expansion. Presented at computer-aided chemical engineering, 21: 16th European symposium on computer aided process engineering and 9th International symposium on process systems engineering.
Zapata JC, Varma VA, Reklaitis GV (2008) Impact of tactical and operational policies in the selection of a new product portfolio. Comput Chem Eng 32:307–319.