S.G. Henderson and B.L. Nelson (Eds.), *Handbook in OR & MS, Vol. 13*
Copyright © 2006 Elsevier B.V. All rights reserved
DOI: 10.1016/S0927-0507(06)13021-2

Chapter 21

Metaheuristics

Sigurdur Ólafsson

Department of Industrial and Manufacturing Systems Engineering,
Iowa State University, USA
E-mail: olafsson@iastate.edu

Abstract

Metaheuristics have been established as one of the most practical approaches to simulation optimization. However, these methods are generally designed for combinatorial optimization, and their implementations do not always adequately account for the presence of simulation noise. Research in simulation optimization, on the other hand, has focused on convergent algorithms, giving rise to the impression of a gap between research and practice. This chapter surveys the use of metaheuristics for simulation optimization, focusing on work bridging the current gap between the practical use of such methods and research, and points out some promising directions for research in this area. The main emphasis is on two issues: accounting for simulation noise in the implementation of metaheuristics, and convergence analysis of metaheuristics that is both rigorous and of practical value. To illustrate the key points, three metaheuristics are discussed in some detail and used for examples throughout, namely genetic algorithms, tabu search, and the nested partitions method.

Keywords: Simulation optimization, metaheuristics, tabu search, genetic algorithms, nested partitions method, convergence analysis

1 Introduction

Optimization in simulation has been a topic of intense investigation for decades, but for most of this period, the research effort was rarely transferred to simulation practice. This has changed dramatically over the past decade, however, and optimization routines are now part of most commercial simulation software. On the other hand, these commercial simulation optimizers draw little on the decades of academic research in this field. In fact, they almost exclusively use metaheuristics that have been most extensively studied to solve deterministic combinatorial optimization problems (April et al., 2003; Fu, 2002; Ólafsson and Kim, 2002).

Several classes of simulation optimization problems and solution methodologies have been analyzed in the literature. When the underlying space is continuous, gradient search methods are applicable and derivative estimation is a key issue (see Chapter 19), when the number of alternatives is small, statistical selection methods can be applied (see Chapter 17), and in many situations, metamodeling and response surface methods have been found useful (see Chapter 18). Another relevant line of research has focused on provably convergent random search methods (see Chapter 20). In this chapter we address the use of metaheuristics in simulation optimization. Our focus is on basic methodology and research issues, but we make a deliberate effort to highlight work that bridges the gap between simulation practice and research, and indicate promising research directions in this area that are motivated by the prevalence of metaheuristics.

In simulation practice, the choice and design of an algorithm always boils down to computational efficiency (Kelly, 2002). Rapid progress of the search and demonstrating improvement over the initial solution therefore takes precedence over possible convergence statements, and provably obtaining a global optimum is rarely a significant concern. While this is certainly a reasonable stance in practice, a consequence of this view may be that the simulation takes on a subservient role to the optimization (Fu, 2002). This, in our opinion, is an undesirable property for simulation optimization software. Thus, the challenge to the research community is to shift the focus back to the simulation itself, by rigorously accounting for the simulation noise in the optimization routines, and establishing convergence results that are both well founded and useful in practice.

Indeed, although more advanced implementations do consider statistical significance, there are some concerns about how metaheuristics developed for combinatorial optimization perform for simulation problems, and there are significant lessons that can be learned from the existing body of research. In particular, we believe that by understanding how to account for simulation noise in metaheuristics, their performance in practice may be improved. We also believe that convergence statements do have value, and that it is reassuring to know that given enough computational effort, the search will eventually be successful. Furthermore, it would be ideal to be able to terminate the search and know at that moment how good a solution has been obtained. With regards to metaheuristics, relatively little work has been done in the deterministic combinatorial optimization context to analyze convergence in this manner, and even less in the simulation optimization context. Thus, with the assumption that metaheuristics will continue to dominate simulation practice, we advocate a research agenda that focuses on how to account for simulation noise in these methods, and making convergence statements that are applicable to how the algorithms are actually employed.

As stated above, several simulation optimization packages have recently been developed for commercial simulation software. These include SimRunner® that is used by Promodel® and OptQuest®, which can be used by numerous simulation software packages such as Arena®, Crystal Ball®, Promodel®, SIMUL8®. Such packages use one or more common metaheuristics as their primary search method. Specifically, genetic algorithms are used in SimRunner®, and tabu search, genetic algorithms, and scatter search are all used in OptQuest®. A recent survey that discussed the use of simulation optimization in such commercial software can be found in April et al. (2003). In this chapter we focus on three metaheuristics methods that have either been used widely in practice or shed some light on the convergence issues to be studied. In particular, we study genetic algorithms, tabu search, and the nested partitions method. We review the state of the art with regard to the use of these methods for simulation optimization; consider how to account for simulation noise in their implementation; and explore what convergence statements can be made.

The remainder of the chapter is organized as follows. In Section 2 we give a brief review of metaheuristics methods and their uses in combinatorial optimization. A comprehensive review is beyond the scope of the chapter, and the focus is on common elements and what defines each method. In Section 3 we discuss how those methods can be applied to simulation optimization and focus in particular on what makes simulation optimization fundamentally different and how the challenges of noisy performance can be addressed. The next three sections discuss specific metaheuristics, namely genetic algorithms, tabu search, and the nested partitions method. The focus of these sections is on the issues that arise when these methods are applied in simulation optimization. Section 7 is a discussion of what type of convergence statements can be made for metaheuristics, with a focus on of both asymptotic and finite-time convergence results. Examples from genetic algorithms and nested partitions methods are used for illustration. Finally, in Section 8, we summarize the research agenda that we advocate for metaheuristics in simulation optimization.

2 Background to metaheuristics

Metaheuristics are designed to tackle complex optimization problems where other optimization methods have failed to be either effective or efficient. These methods have come to be recognized as one of the most practical approaches for solving many complex problems, and this is particularly true for the many real-world problems that are combinatorial in nature. The practical advantage of metaheuristics lies in both their effectiveness and general applicability. In the early research literature, specialized heuristics were typically developed to solve complex combinatorial optimization problems. This required a new approach to every problem and lessons learned from one problem did not always generalize well to a different class of problems. On the other hand, with the emergence of more general solution strategies, including such metaheuristics as tabu search, genetic algorithms and simulated annealing, the main challenge has become adapting the metaheuristics to a particular problem or problem class. This usually requires much less work than developing a specialized heuristic for a specific application, which makes metaheuristics an appealing choice for implementation in general purpose software. Furthermore, a good metaheuristic implementation is likely to provide near optimal solutions in reasonable computation times. For further reading, Glover and Kochenberger (2003) provide a good introduction and general reference to many of the most popular metaheuristics.

The applicability of metaheuristics as a preferred method over other optimization methods is primarily to find good heuristic solutions to complex optimization problems with many local optima and little inherent structure to guide the search. The metaheuristic approach to solving such problem is to start by obtaining an initial solution or an initial set of solutions, and then initiating an improving search guided by certain principles. The structure of the search has many common elements across various methods. In each step of the search algorithm, there is always a solution (or a set of solutions) θ_k, which represents the current state of the algorithm. Many metaheuristics, including simulated annealing, tabu search, variable neighborhood search, and GRASP, are solution-to-solution search methods, that is, θ_k is a single solution or point $\theta_k \in \Theta$ in some solution space Θ. Others, including genetic algorithms, scatter search and the nested partitions method, are set-based, that is, in each step θ_k represents a set of solutions $\theta_k \subseteq \Theta$. However, the basic structure of the search remains the same regardless of whether the metaheuristics is solution-to-solution or set-based.

Given a neighborhood $N(\theta_k)$ of the solution (set), a candidate solution (set) $\{\theta^c\} \subset N(\theta_k)$ is selected and evaluated. This evaluation involves calculating or estimating the performance of the candidate solution(s) and comparing them with the performance of θ_k and sometimes with each other. Based on this evaluation, the candidate may be either accepted, in which case $\theta_{k+1} = \theta^c$, or rejected, in which case $\theta_{k+1} = \theta_k$. We now have the following metaheuristic framework:

```
**Obtain** an initial solution (set) $\theta_0$ and set $k = 0$.
**Repeat**:
    Identify the neighborhood $N(\theta_k)$ of the current solution(s).
    Select candidate solution(s) $\{\theta^c\} \subset N(\theta_k)$ from the neighborhood.
    Accept the candidate(s) and set $\theta_{k+1} = \theta^c$ or reject it and
    set $\theta_{k+1} = \theta_k$.
    Increment $k = k + 1$.
**Until** stopping criterion is satisfied.
```

As we will see in this section, this framework can be applied to numerous metaheuristics.

The reason for the “meta-” prefix is that metaheuristics do not specify all the details of the search, which can thus be adapted by a local heuristic to a specific application. Instead, they specify general strategies to guide specific aspects of the search. For example, tabu search uses a list of solutions or moves called the tabu list, which ensures the search does not revisit recent solutions or becomes trapped in local optima. The tabu list can thus be thought of as a restriction of the neighborhood. On the other hand, methods such as genetic algorithm specify the neighborhood as all solutions that can be obtained by combining the current solutions through certain operators. Other methods, such as simulated annealing, do not specify the neighborhood in any way, but rather specify an approach to accepting or rejecting solutions that allows the method to escape local optima. Finally, the nested partitions method is an example of a set-based method that selects candidate solutions from the neighborhood with a probability distribution that adapts as the search progresses to make better solutions be selected with higher probability.

Within the framework presented here, all metaheuristics share the elements of selecting candidate solution(s) from a neighborhood of the current solution(s) and then either accepting or rejecting the candidate(s). With this perspective, each metaheuristic is thus defined by specifying one or more of these elements, but allowing others to be adapted to the particular application. This may be viewed as both a strength and a liability. It implies that we can take advantage of special structure for each application, but it also means that the user must specify those aspects, which can be complicated. For the remainder of this section, we briefly introduce a few of the most common metaheuristics and discuss how they fit within this framework. Three of those methods will then be analyzed in more detail as we discuss how to apply them for simulation optimization in subsequent sections.

One of the earliest metaheuristics is *simulated annealing* (Kirkpatrick et al., 1983; Eglese, 1990; Fleischer, 1995), which is motivated by the physical annealing process, but within the framework here simply specifies a method for determining if a solution should be accepted. As a solution-to-solution search method, in each step it selects a candidate $\theta^{c} \in N(\theta_k)$ from the neighborhood of the current solution $\theta_k \in \Theta$. The definition of the neighborhood is determined by the user. If the candidate is better than the current solution it is accepted, but if it is worse it is not automatically rejected, but rather accepted with probability

$$P[\text{Accept } \theta^{c}] = e^{(f(\theta_k)-f(\theta^{c}))/T_k}, \tag{1}$$

where $f : \Theta \to \mathbb{R}$ is an objective function to be minimized, and T_k is a parameter called the temperature. Clearly, the probability of acceptance is high if the performance difference is small and T_k is large. The key to simulated annealing is to specify a cooling schedule $\{T_k\}_{k=1}^{\infty}$, by which the temperature is reduced so that initially inferior solutions are selected with a high enough probability so local optimal are escaped, but eventually it becomes small enough so that the algorithm converges. We do not discuss simulated annealing any further here and refer the interested reader to Chapter 20 for its use in simulation optimization.

Tabu search is another popular metaheuristics (Glover, 1989, 1990; Glover and Laguna, 1997). As is the case for simulated annealing, it is a solution-to-solution search method where the neighborhood is specified by the user. However, the defining characteristic of tabu search is in how solutions are selected from the neighborhood. In each step of the algorithm, there is a list L_k of solutions that were recently visited and are therefore tabu. The algorithm looks through all of the solutions of the neighborhood that are not tabu and selects the best one, that is,

$$\theta^c = \operatorname*{arg\,min}_{\theta \in N(\theta_k) \cap \overline{L}_k} f(\theta), \tag{2}$$

where as before $f : \Theta \to \mathbb{R}$ is to be minimized. The candidate solution θ^c is accepted even if it is worse than the current solution, that is, $\mathrm{P}[\text{Accept } \theta^c] = 1$. Accepting inferior solutions allows the search to escape local optima, and the tabu list prevents the search from immediately reverting to the previous solution. Tabu search for simulation optimization is discussed further in Section 5. However, tabu search has many intricacies and variations that are not discussed in this chapter, and we refer the interested reader to Glover and Laguna (1997) for a more complete discussion of this method.

Other popular solution-to-solution metaheuristics include the *greedy randomized adaptive search procedure* (GRASP) and the *variable neighborhood search* (VNS). The defining property of GRASP is its multi-start approach that initializes several local search procedures from different starting points. The advantage of this is that the search becomes more global, but on the other hand, each search cannot use what the other searches have learned, which introduces some inefficiency. The VNS is interesting in that it uses an adaptive neighborhood structure, which changes based on the performance of the solutions that are evaluated. More information on GRASP can be found in Resende and Ribeiro (2003), and for an introduction to the VNS approach we refer the reader to Hansen and Mladenovic (1999).

Several metaheuristics are set-based or population based, rather than solution-to-solution. This includes genetic algorithms and other evolutionary approaches, as well as scatter search and the nested partitions method. All of these methods are readily adapted to simulation optimization, and both genetic algorithms and the nested partitions method are discussed in detail in later sections. All *evolutionary algorithms* are based on the idea of natural selection, where a population of solutions evolves, or improves, through a series of genetic operators (Goldberg, 1989; Liepins and Hilliard, 1989; Muhlenbein, 1997). This includes survival of the fittest or best solutions, crossover, which is simply a combination of two fit solutions, and mutation, which is a slight modification to fit solutions. From the perspective of the general framework, these operators define the neighborhood of a solution set. Thus, given a current set $\theta_k \subseteq \Theta$ of solutions, the neighborhood is defined as

$$N(\theta_k) = N^{crossover}(\theta_k) \cup N^{mutation}(\theta_k) \cup \theta_k, \tag{3}$$

where

$$\begin{aligned} & N^{crossover}(\theta_k) \\ & \quad = \{\psi \in \Theta \mid \psi \text{ is the crossover of two solutions } \zeta_1, \zeta_2 \in \theta_k\}, \\ & N^{mutation}(\theta_k) = \{\psi \in \Theta \mid \psi \text{ is a mutation of some } \zeta \in \theta_k\}. \end{aligned} \tag{4}$$

For evolutionary methods, the key feature is therefore this innovative definition of a neighborhood, which allows the search to quickly and intelligently traverse large parts of the solution space. The selection of candidates from the neighborhood is either deterministic or random, which is discussed further in Section 4. The selected candidates are then always accepted.

Scatter search is another metaheuristic related to the concept of evolutionary search. In each step a scatter search algorithm considers a set, $\theta_k \subseteq \Theta$, of solutions called the reference set. Similar to the genetic algorithm approach, these solutions are then combined into a new set $\theta_{k+1} \subseteq \Theta$. However, as opposed to the genetic operators, in scatter search the solutions are combined using linear combinations, which thus define the neighborhood $N(\theta_k)$. For references on scatter search, we refer the reader to Glover et al. (2003).

The final method that we mention in this section is the *nested partitions* method (Shi and Ólafsson, 2000a), which like genetic algorithms and scatter search is a set-based metaheuristic. Unlike any of the previous methods, however, the nested partitions method is global in that the neighborhood is always the entire set of feasible solutions. Thus, given a current set $\theta_k \subseteq \Theta$ of solutions, the neighborhood is always $N(\theta_k) = \Theta$. In addition to the global perspective, the defining element of this method is the adaptive distribution that is used to obtain sample solutions from this neighborhood. By going through a sequence of set partitions, with each partition nested within the last, the sampling is concentrated in sets that are considered promising, that is, where the optimal solution is believed to be contained. Once the random candidate solutions have been generated according to this adaptive distribution, they are always accepted, and the search continues with a new sampling distribution. The nested partitions method for simulation optimization is discussed in more detail in Section 6.

Although the metaheuristics discussed in this section are usually considered separately in the literature, we have made the case that they do in fact have many common elements that make it possible to analyze them within a common framework. An interesting step in this direction is the generalized hill-climbing (GHC) algorithm framework of Jacobson et al. (1998). The GHC framework is general enough to cover various stochastic local search algorithms, including both simulated annealing and tabu search. Analysis of the GHC can be found in Johnson and Jacobson (2002) and Sullivan and Jacobson (2001).

3 Accounting for simulation noise

Metaheuristics have been found to be particularly effective for combinatorial optimization, and it is therefore natural to examine if they perform similarly when applied to such problem in the stochastic or simulation optimization context. In this section, we define the simulation optimization problem, examine what makes this problem uniquely different from the corresponding deterministic problem, and discuss how metaheuristics need to account for the simulation noise when solving such problems.

We define the general simulation optimization problem to be studied as follows. Given a finite feasible region Θ, and performance measure $J : \Theta \to \mathbb{R}$, the objective is to find the solution $\theta^* \in \Theta$ that minimizes the objective, that is,

$$\theta^* = \arg\min_{\theta \in \Theta} J(\theta). \tag{5}$$

In addition to the general difficulties association with combinatorial optimization, we have the complexity that for any solution $\theta \in \Theta$, the performance $J(\theta)$ cannot be evaluated analytically and must be estimated using simulation. The performance is often an expectation of some random estimate of the performance of a complex stochastic system given the parameter θ, that is,

$$J(\theta) = \mathrm{E}\big[L(\theta)\big]. \tag{6}$$

Here, $L(\theta)$ is random variable, which depends on the parameter $\theta \in \Theta$, which we assume to be the sample performance from a simulation run.

Numerous metaheuristics, such as genetic algorithms, simulated annealing, the nested partitions method, and tabu search, have been adapted for solving the problem above. As mentioned above, metaheuristics, and in particular evolutionary algorithms, certainly dominate simulation practice, and numerous articles that discuss metaheuristics for simulation and stochastic optimization, mostly application oriented, have appeared in the research literature. For same examples of this line of research, see Azadivar and Wang (2000), Faccenda and Tenga (1992), Haddock and Mittenhall (1992), Hall and Bowden (1997), Tompkins and Azadivar (1995) and Watson et al. (2003).

Despite their definite success in practice, directly applying metaheuristics that are designed for combinatorial optimization problems to simulation optimization may ignore some of the differences that exist from deterministic environments. Metaheuristics are generally not designed to account for the simulation noise, which as we discuss below may be critical. As discussed in Section 2, these methods attempt to move from one solution to another better solution, or one set of solutions to another set of solutions, and thus iteratively improving the solution quality until the method terminates. However, when simulation is used to estimate the performance, determining what constitutes a better solution becomes an issue. Given two solutions θ_1 and θ_2, and simulation performance estimates $\widehat{J}(\theta_1)$ and $\widehat{J}(\theta_2)$, the fact that $\widehat{J}(\theta_1) < \widehat{J}(\theta_2)$ does not guarantee that $J(\theta_1) < J(\theta_2)$. In particular, we now have to look at the statistical significance. In other words, the question is whether $\widehat{J}(\theta_1)$ is statistically significantly smaller than $\widehat{J}(\theta_2)$? How this should be implemented in a search algorithm is an issue that has received very little attention. It could be argued that a solution should only be considered better, and hence the algorithm should move to this solution, only if the improvement is statistically significant at some level. However, it is also possible that a sequence of improvements that were not significant individually will become significant when taken together (Kelly, 2002), and many metaheuristics are quite robust to certain level of randomness.

Accounting for the simulation noise is a theoretically challenging issue that has real implications to simulation practice. However, this issue has received somewhat limited attention in the research literature to date. Many of the relevant research issues are application specific, but at least two general research directions seem promising. In particular, ideas from both ordinal optimization and statistical selection have been successfully applied to guide the search in stochastic environments, and we believe more research is warranted in both areas.

One approach to simulation optimization is to shift the focus from cardinal to ordinal comparison (Ho et al., 2000). Making a decision to move from one solution to another only requires comparison between the solutions, that is, if we know that $J(\theta_1) < J(\theta_2)$, then we know if the move should be made. Such comparisons are the focus of ordinal optimization (Ho et al., 1992). It can be shown that ordinal comparisons converge much faster than cardinal estimation (Dai, 1996), which could be utilized to guide the search. These considerations are relevant to any metaheuristic as they all move based on comparisons between solutions.

An alternative approach is to incorporate statistical selection methods into the search. For the nested partitions method, Ólafsson (2004) proposes using statistical selection to guide the movement of the search. In particular, for every step of the algorithm, a statistical selection procedure is used to determine the required amount of samples from each region so that the correct region is selected within an indifference zone with a given minimum probability. In related work, Shi and Chen (2000) use statistical selection to determine the simulation effort for each sample solution used by the nested partitions method. Various statistical selection methods have been studied extensively in simulation (see Chapter 17), and it should be possible to take advantage of these methods in a similar fashion for other metaheuristics.

Another issue in accounting for the simulation noise is how to recover from the incorrect moves that will inevitably be made due to this noise. Tabu search, for example, uses the tabu list to disallow moves to previously visited solutions. This feature enables it to escape local optima and is intuitively appealing in the deterministic context, when we know with certainty the quality of those previously visited solutions, as well as those solutions that are in the neighborhood of the previously visited solutions. However, in the simulation optimization context it is possible, and even likely, that an incorrect move was selected due to the simulation noise, which may make it worthwhile to revisit a solution for a reason that does not exist in the deterministic context. How this can be solved, while retaining the appealing property of the tabu list, is an issue that needs to be addressed.

In genetic algorithms, a solution is selected for survival or crossover due to its fitness. Fit solutions will reoccur either unchanged or as part of their offspring, whereas unfit solutions will not survive. However, in the simulation context, the fitness of these solutions is subject to the same simulation noise and thus good solutions may be judged unfit prematurely. The question now arises of how to account for the simulation noise. One approach might be to account for the simulation noise in every generation by making sure that the selection for survival or crossover is done at a statistically significant level, or by employing statistical screening procedures to select a subset of solutions for survival or crossover. However, genetic algorithms seem quite robust to inaccuracies in the performance estimates as it is actually preferable to have some randomness in the selection of solutions, and further research is certainly warranted to resolve these issues.

The nested partitions method includes a built-in mechanism for recovering from incorrect moves. In every step, the neighborhood of the current solution is the whole solution space, so any solution has a positive probability of being selected to be evaluation. This includes the surrounding region, that is, those solutions that are not in the set currently considered most promising. If the best solution is found in the surrounding region, the algorithm backtracks to an earlier (larger) set of solutions. This can be viewed as an automated mechanism to recover from an incorrect move made either due to the use of random samples to guide the search, or due to the simulation noise. These global search and backtracking mechanisms of the nested partitions method can therefore be used as a global guidance system to local improvement methods in simulation optimization (Pichitlamken and Nelson, 2003). In the context of combinatorial optimization, other efficient metaheuristics such as genetic algorithms (Shi et al., 1999) and tabu search (Shi and Men, 2003) have been incorporated into the nested partitions method. These methods are used to speed the evaluation of each region, but the nested partitions guide the overall search. In terms of computational efficiency, this approach has been found to perform favorably when compared to the direct use of either genetic algorithms or tabu search for many applications, and in the simulation optimization context, it has the added benefit of allowing the search to recover from mistakes made due to the simulation noise. Such promising results from combinatorial optimization applications also motivate further investigation of such global guidance systems for metaheuristics in simulation optimization.

Finally, an overarching issue in accounting for simulation noise for any metaheuristic is the question of how the inevitably very limited computational effort should be spent. That is, given a fixed computing-budget, there is a tradeoff between obtaining high quality estimates of each solution, and allowing the search to traverse the solution space quickly and exploring more solutions. At least some partial resolution to this issue is the goal of much ongoing research in this area, and has implications in the design, implementation, and analysis of the algorithms.

4 Genetic algorithm

This section looks closer at one of the popular metaheuristics for simulation optimization, namely genetic algorithms. As an approach to global optimization, genetic algorithms (GA) have been found to be applicable to optimization problems that are intractable for exact solutions by conventional methods (Holland, 1975; Goldberg, 1989). It is a set-based search algorithm, where at each iteration it simultaneously generates a number of solutions. In each iteration, a subset of the current set of solutions is selected based on their performance and these solutions are combined into new solutions. The operators used to create the new solutions are survival, where a solution is carried to the next iteration without change, crossover, where the properties of two solutions are combined into one, and mutation, where a solution is modified slightly. The same process is then repeated with the new set of solutions. The crossover and mutation operators depend on the representation of the solution, but not on the evaluation of its performance. They are thus the same even though the performance is estimated using simulation. The selection of solutions, however, does depend on the performance. The general principle is that high performing solutions (which in genetic algorithms are referred to as fit individuals) should have a better chance of both surviving and being allowed to create new solutions through crossover. The simplest approach is to order the solutions $J(\theta_{[1]}) \leqslant J(\theta_{[2]}) \leqslant \cdots \leqslant J(\theta_{[n]})$, and only operate on the best solutions. If a strict selection of say, the top k solutions, were required, this would complicate the issue significantly in the simulation optimization context, and considerable simulation effort would have to be spent to obtain an accurate ordering of the solutions.

Fortunately, genetic algorithms appear to be quite robust with respect to which solutions are selected to create the next set. Indeed, a purely deterministic selection of the top k solution is typically not the best approach for deterministic problems, and some randomness is usually introduced into the process. A popular example of this is the roulette strategy, which several authors have used for the application of genetic algorithms to stochastic problems (see, e.g., Dasgupta and Mcgregor, 1992; Grefenstette, 1992; Ishibuchi and Murata, 1996; Vavak and Fogarty, 1996; Yoshitomi and Yamaguchi, 2003). In the roulette strategy the probability of selecting a solution θ is calculated as follows,

$$P(\theta) = \frac{\hat{f}(\theta)}{\sum_{\mathrm{All}\,\theta} \hat{f}(\theta)} \tag{7}$$

Here $\hat{f}(\theta)$ is an estimate of the fitness function $f : \Theta \to \mathbb{R}$ that measures the quality of the solution and is to be maximized (higher value implies more fit). Thus, every solution has a positive probability of being selected, but the fitter solutions are selected with higher probability. Assuming an unbiased simulation estimate, this statement will continue to be true when the roulette strategy is applied to simulation optimization. However, existing studies in this area are based almost exclusively on numerical evaluations and the claim is simply made that genetic algorithms are robust and hence applicable to simulation optimization. While the robustness of genetic algorithms with respect to noise in the selection method is unquestionable, it would be desirable to have research that provides a better understanding into how much noise is acceptable, and thus how much simulation effort should be devoted to the evaluation of solutions during the search.

In another study, Boesel et al. (2003a) use genetic algorithm together with statistical selection method to develop a system for simulation optimization. Their approach consists of three phases. First, there is an initialization phase, where the parameters of the algorithms are specified. Second, there is the actual search phase, that is, the usual iteration of the genetic algorithm selection, crossover, and mutation operators. Finally, then the search terminates there is a solutions phase where the alternatives generated by the GA search are evaluated using ranking and selection. In particular, they use a screening and selection procedure to first quickly filter out inferior solutions and then determine the best solution by carrying out additional simulation runs for the remaining solutions. They also consider how best to implement the solution selection procedure in a noisy environment. Their design choice is to use what is called anti-ranks and a q-tournament selection (Back, 1996), which also appears to be quite robust with regards to the simulation noise.

Genetic algorithms and other evolutionary approaches appear to be readily adaptable to simulation optimization. As discussed in this section, the primary reason is that only the selection of solutions for survival, crossover, and mutation depends on the performance estimates, and a certain amount of noise is desirable in this process even for deterministic optimization. Thus, it is possible to envision that certain amount of simulation noise might even assist the search. However, most results that point in that direction are based on numerical evidence and we encourage further research in this area that focuses on how to best select solutions in the presence of noise.

5 Tabu search

Tabu search was introduced by Glover (1989, 1990) to solve combinatorial optimization problems and it has been used effectively for simulation optimization, most notably by the OptQuest simulation optimization software (April et al., 2003). It is a solution-to-solution method and the main idea is to make certain moves or solutions tabu, that is they cannot be visited as long as they are on what is called the tabu list. The tabu list L_k is dynamic and after each move, the latest solution θ_k, or the move that resulted in this solution, is added to the list and the oldest solution or move is removed from the list. Another defining characteristic of tabu search is that the search always selects the best nontabu solution from the neighborhood, even if it is worse than the current solution. This allows the search to escape local optima, and the tabu list ensures that the search does not revert back. Tabu search has numerous other elements, such as long-term memory that restarts the search with a new tabu list at previously found high quality solutions, and a comprehensive treatment of this methodology can be found in Glover and Laguna (1997).

Several aspects of tabu search are affected by the noisy estimates of simulation optimization. First, the simulation noise is relevant to one of the two main components of this approach, namely that in every step the best nontabu solution is selected from the neighborhood $N(\theta_k)$ of the current solution θ_k, that is, $\theta_{k+1} = \arg\min_{\theta \in N(\theta_k) \cap \overline{L}_k} \widehat{J}(\theta)$ where $\overline{L}_k$ denotes the compliment of L_k, or the nontabu solutions. The issue thus arises of how accurately this selection should be made, that is, is it worthwhile to spend considerable simulation effort on each solution in $N(\theta_k) \cap \overline{L}_k$ to make the correct selection with a high probability, or is this simulation effort better spent by making a quick selection and exploring more of the search space. Note that unlike genetic algorithms that select a set of solutions and randomness is beneficial even in the deterministic context, here it is desirable to select the single best solution. The answer to how accurate this selection needs to be is not clear and deserves further investigation.

The simulation noise is also relevant to the other main component of tabu search, namely the tabu list itself. In the deterministic context, the best solution is selected from a neighborhood $N(\theta_k)$ and thus if the search visits a solution θ_k again, the same solution will be selected from the neighborhood (assuming it is not tabu in either visit). This would imply a cycle, which is avoided by the use of the tabu list. However, in the presence of simulation noise, each time a solution is visited, a different neighborhood solution may be selected, even if the tabu list is identical. Also, if the incorrect solution is selected from the neighborhood $N(\theta_k)$, that is, another nontabu solution $\xi \in N(\theta_k) \cap \overline{L}_k$ satisfies $J(\xi) < J(\theta_{k+1})$, then it may be desirable to revisit θ_k to correct the mistake. Thus, for simulation optimization, the tabu list loses some of the appeal it has for deterministic optimization. In particular, cycles can be broken without it due to the simulation noise, and allowing the algorithm to revert back to an old solution enables the search to correct mistakes made due to the simulation noise. On the other hand, as noted earlier, tabu search has be effectively applied for simulation optimization, so it is certainly worthwhile to investigate how its performance is affected by the simulation noise and what implications this has in terms of its ideal implementation.

6 The nested partition method

Introduced by Shi and Ólafsson (2000a), the nested partition (NP) method is another metaheuristic for combinatorial optimization that is readily adapted to simulation optimization problems (Shi and Ólafsson, 2000b). The key idea behind this method lies in systematically partitioning the feasible region into subregions, evaluating the potential of each region, and then focusing the computational effort to the most promising region. This process is carried out iteratively with each partition nested within the last. The computational effectiveness of the NP method relies heavily on the partitioning, which, if carried out in a manner such that good solutions are clustered together, can reach a near optimal solution very quickly.

In the kth iteration of the NP algorithm, a region $\sigma(k) \subseteq \Theta$ is considered most promising. What this means is that the algorithm believes that this is the most likely part of the solution space to contain the optimal solution, and thus the computation effort should be concentrated in this region. As in the beginning nothing is known about the location of the optimal solution, the algorithm is initialized with $\sigma(0) = \Theta$. The most promising region is then partitioned into M subsets or subregions and the entire surrounding region $\Theta \backslash \sigma(k)$ is aggregated into one. Thus, in each iteration $M + 1$ disjoint subsets that cover the entire feasible region are considered. Each of these $M + 1$ regions is sampled using some random sampling scheme to obtain sets of solutions, and the simulated performance function values at randomly selected points are used to estimate the promising index for each region. This index determines the most promising region in the next iteration by taking the subregion scoring highest on the promising index. If the surrounding region rather than the subregion is found to have the highest promising index, the method backtracks to a previous solution corresponding to a larger region. The new most promising region is partitioned and sampled in a similar fashion. This generates a sequence of set partitions, with each partition nested within the last.

The effect of the simulation noise comes into play in the NP method in the selection of a region as the most promising region. If the performance estimates for each sample solution are noisy, the next most promising region is selected with less confidence. However, this is already a noisy selection even for deterministic problem since there are two sources of randomness:

- There is a sampling error due to a small sample of solutions being used to estimate the overall promise of what is often a large set of solutions (region).
- For each of the sample solutions, the performance is estimated using simulation, and is hence noisy.

The nested partitions method is therefore a set-based method that has an inherent noise in the selection of move, which as for genetic algorithms, appears to make it relatively insensitive to the simulation noise. Furthermore, as mentioned in Section 3, the nested partitions method includes a built-in mechanism for recovering from incorrect moves. Sometimes the simulation noise may cause the algorithm to move to the wrong subregion. However, as the search progresses the surrounding region continues to be sampled, which allows the algorithm to recover from incorrect moves through backtracking.

Regardless of the robustness of the method with regard to simulation noise, it is desirable to be able to control this noise. Ólafsson (2004) addresses this in a variant of the nested partitions method that uses statistical selection to guide the search. In this variant, the sampling of each region has two phases. First, a small sample is obtained from each region to get an initial idea of the performance variance within each region, and possibly screen out very poor regions. Then, based on these initial samples, a second set of samples is obtained from all viable regions to ensure that the correct selection of a region is made within an indifference zone with a pre-specified minimum probability. This can thus be thought of as a mechanism to control the noise in the performance estimates of each region, which guides the search. However, this only controls the overall noise and simply prescribes additional samples from each region. Some of the previously sampled solutions could be sampled again, which would improve the estimate of that solution's performance, or it could be completely different solutions. Thus, this method does not prescribe how much effort should be devoted to estimating the performance of each solution and how much should be devoted to obtaining more samples solutions. Investigating this balance is an important future research topic.

7 Making convergence statements

Research in simulation optimization has largely focused on algorithms where rigorous convergence statements can be made. However, such analysis has frequently been under either severely restrictive assumptions, or the design of the algorithm sacrifices efficiency for theoretically appealing properties. This has contributed to an unfortunate perception of a trade-off between rigor and practicality.

Much of this work in this area has focused on some type of asymptotic convergence. For example, many stochastic optimization procedures can be shown to converge to the optimal solution θ^* almost surely or with probability one (w.p.1), that is,

$$\hat{\theta}_k^* \overset{k\to\infty}{\longrightarrow} \theta^*, \quad \text{w.p.1.} \tag{8}$$

Here, $\hat{\theta}_k^*$ denotes the best solution found by the kth iteration, that is, the current estimate of the optimal solution. Other algorithms may converge in probability or in distribution. The drawback to any asymptotic analysis is that having such convergence results may be of limited practical value. It is of course somewhat reassuring to have asymptotic convergence, but since little can be inferred about what happens when the optimizer inevitably terminates in finite time, its value to a practitioner is often hard to argue.

Although most research on metaheuristics in the deterministic domain has focused on implementation and computational issues, the research has also resulted in numerous convergence proofs of these algorithms. For example, genetic algorithms have been investigated to determine how many iterations are needed before the optimal solution θ^* has been seen with a given probability, that is, the best solution found so far, $\hat{\theta}_k^*$, is equal to the optimum. In particular, Liepins (1992) shows that a genetic algorithm generates a homogeneous Markov chain, and uses this to analyze the number of iterations $k = t(1-\alpha)$ needed so that

$$P\left[\hat{\theta}^*_{t(1-\alpha)} = \theta^*\right] \geqslant 1 - \alpha. \tag{9}$$

Aytug et al. (1996) develop a bound on the number $t(1-\alpha)$ of iterations required by the genetic algorithm for deterministic optimization. Greenhalgh and Marshall (2000) further refine this bound and show that

$$t(1-\alpha) \leqslant \left\lceil \frac{\ln(\alpha)}{n \ln(1 - \min\{(1-\mu)^{\gamma-1}(\mu/(K-1)), (\mu/(K-1))^{\gamma}\})} \right\rceil, \tag{10}$$

where $\mu > 0$ is the mutation probability, and each solution is represented as a point in the space $\{0, 1, 2, \ldots, K-1\}^{\gamma}$, which means K is the string length, for example, $K = 2$ corresponds to the traditional binary string representation. Reynolds and Gomatam (1996) also analyze genetic algorithms for deterministic optimization using a Markov chain analysis, and prove convergence of certain variations.

Note that (9) does not guarantee the quality of the estimate $\hat{\theta}^*_{t(1-\alpha)}$ of the best solution, just that it is equal to the best solution with a minimum probability. In other words, it does not make a direct statement about the performance difference $|J(\hat{\theta}^*_{t(1-\alpha)}) - J(\theta^*)|$, even in expectation or probability. This shift away from cardinal comparison is related to the concept of ordinal optimization where the analysis is solely based on ordinal comparisons (Ho et al., 1992, 2000). In this approach, the objective is relaxed from finding the optimal solution to finding a subset $\Theta_G \subset \Theta$ of 'good enough' solutions (called goal softening). In the kth step, there is a set of selected solutions $\hat{\theta}_k^* \subset \Theta$, and the key convergence statements are made about the alignment probability $P[|\Theta_G \cap \hat{\theta}_k^*| \geqslant y]$ between those two sets. For example, if $\hat{\theta}_k^*$ is a set of $g = |\Theta_G|$ uniformly sampled solutions, then the alignment probability is (Ho et al., 1992)

$$P\left[|\Theta_G \cap \hat{\theta}_k^*| \geqslant y\right] = \sum_{i=y}^{g} \binom{g}{i}\binom{N-g}{g-i} \Big/ \binom{N}{g}, \tag{11}$$

where $N = |\Theta|$. We note that (9) is a special case of the alignment probability with $\Theta_G = \{\theta^*\}$, $y = 1$, and $k = t(1 - \alpha)$.

Significant research has also been devoted to the convergence of tabu search in the deterministic context. In an early work, by introducing the Metropolis criterion and annealing process of simulated annealing into the general framework of tabu search, Tian et al. (1997) proposed a stochastic tabu search strategy for discrete optimization. This algorithm is provably asymptotically convergent to a global optimal. Later, Hanafi (2000) proved convergence using the observation that if the neighborhood employed is strongly connected there will be a reversible path from each solution to the every other one, which is a conjecture originally stated by Glover (1990). Glover and Hanafi (2002) provided more efficient forms of convergent tabu search, offering specific bounds, and establishing the finite convergence of tabu search for deterministic combinatorial optimization.

Shi and Ólafsson (2000a) showed that in the case of combinatorial optimization problems, the nested partition method converges with probability one to a global optimum in finite time. In their analysis, they showed that the NP algorithm generates an absorbing Markov chain and that any absorbing state corresponds to a global optimum. Thus, the algorithm may spend a certain random number of iterations in the transient states, but since the number of states is finite, it will visit an absorbing state, and hence converge, in time T, which is finite with probability one, that is, $\mathrm{P}[\hat{\theta}^*_T = \theta^*] = 1$ for a random stopping time T, with $\mathrm{P}[T < \infty] = 1$. They also use the Markov chain analysis to provide bounds on $\mathrm{E}[T]$, the expected number of iterations required for convergence.

With only a few exceptions, both asymptotic and finite-time convergences results of metaheuristics have been done for deterministic problems only. However, notable exceptions are simulated annealing and the use of global guidance systems, and in particular the nested partitions method. The simulated annealing algorithm is discussed in Chapter 20 and we do not explore it further here. With regards to the nested partitions method, Shi and Ólafsson (2000b) prove the asymptotic convergence of NP algorithm for stochastic problems. They show that the NP algorithm generates an ergodic Markov chain, and used this to develop conditions under which the NP method converges to an optimal solution with probability one, that is, under which the algorithm satisfies (8). The key to the proof is to show that the estimate of the best solution, which is the singleton region visited the most often, converges with probability one to a maximizer of the stationary distribution of the chain. Under some mild condition, this maximum corresponds to the global optimum, and it follows that for simulation optimization the NP method converges asymptotically to an optimal solution. Pichitlamken and Nelson (2003) also employ the NP method as a global guidance system. In their approach, they combine NP with statistical selection of the best, which is used to control the selection error, and local improvement heuristics, which is used to speed the search. The result is an algorithm that is both provably convergent and exhibits good performance in practice.

Despite the reassurance of asymptotic convergence, for the practitioner it would be more useful to be able to make a convergence statement once the optimizer terminates. Ideally, one would like to be able to make a statement about how close in performance our estimate $\hat{\theta}_k^*$ of the optimal solution is to the true optimal solution θ^*, that is,

$$\left|J\left(\hat{\theta}_k^*\right) - J\left(\theta^*\right)\right| \leqslant \varepsilon, \tag{12}$$

where ε is an error that is either pre-specified or calculated after the algorithm terminates after the kth iteration. This is an unreasonably strong statement to hope for in simulation optimization. On the other hand, relaxed versions of such statements are common in the simulation literature when ranking and selection algorithms are used (see Chapter 17). In particular, relaxing the statement (12) to being with an indifference zone $\varepsilon > 0$ of the optimal performance with a given minimum probability $1 - \alpha$, that is,

$$\mathrm{P}\left[\left|J\left(\hat{\theta}_k^*\right) - J\left(\theta^*\right)\right| \leqslant \varepsilon\right] \geqslant 1 - \alpha, \tag{13}$$

makes it a more reasonable goal for simulation optimization. Such convergence statements are commonly made for statistical selection, but those are only applicable when the number of alternatives is small.

Utilizing the idea of relaxing the convergence to obtaining a solution that has performance within an indifference zone of the optimum with a minimum probability, Ólafsson (2004) analyzed the finite time behavior of a variant of the nested partitions method called the two-stage nested partitions (TSNP). This variant uses a statistical selection method in each step to guide the amount of computational effort of the algorithm. They show that (13) is satisfied when the algorithm terminates at a random time $k = T$, given that statistical selection methods are used to ensure that the correct selection is made in each step within the same indifference zone ε and with a given probability $\mathrm{P}(CS)$. This probability can be calculated based on the final probability $1 - \alpha$, and the depth d of the partitioning tree, that is,

$$\mathrm{P}(CS) = \frac{(1-\alpha)^d}{\alpha^d + (1-\alpha)^d}. \tag{14}$$

The depth d is the number of partitions needed until the regions become singletons. This number is proportional to $\log(|\Theta|)$, and is therefore monotone increasing, albeit slowly, in the problem size. Ólafsson (2004) also provides simple bounds on the expected number of steps until termination, namely

$$\mathrm{E}[T] \leqslant \frac{d}{2P(CS) - 1}. \tag{15}$$

We finally note that by selecting the indifference zone small enough, that is, if θ^* is unique and

$$\varepsilon < \left| \min_{\theta \in \Theta \setminus \{\theta^*\}} J(\theta) - J\bigl(\theta^*\bigr) \right| \tag{16}$$

then (13) reduces to (9). This illustrates that these approaches are related, although for the NP for simulation optimization, the time is necessarily random and thus the bounds (15) are in terms of expectation whereas (10) is deterministic, and (13) is more general than (9) in that it provides a performance statement for the final solution when it is not the optimal solution.

Even though they are the most popular metaheuristics for simulation optimization in practice (April et al., 2003), there has been very little work done regarding the convergence of genetic algorithms when it comes to its application to stochastic problems. Thus, more work is needed to analyze both asymptotic convergence and finite time convergence properties. An interesting analysis of the output of genetic algorithm is the work done by Boesel et al. (2003b), which builds on the work by Boesel et al. (2003a) described in Section 4. As mentioned before, in order to apply deterministic genetic algorithms in stochastic setting, the authors combined the search with both multiple comparison procedure and ranking and selection procedures. Thus, in this approach the search is separated from the solution selection. Genetic algorithm is first used to generate a set of solutions, and these solutions are then further analyzed to select the one that is truly the best. This does not prove convergence, or even make a statement regarding the quality of the final solution relative to the optimal solution, but it does rigorously analyze the output generated by the genetic algorithm and accounts for the simulation noise. This analysis does, however, seem to naturally complement the analysis of Liepins (1992), Aytug et al. (1996) and Greenhalgh and Marshall (2000) and Equation (10) in particular, which could to be extended to the simulation optimization setting by accounting for the simulation noise.

Most convergence analysis for simulation optimization algorithms appears to be based on the algorithm generating a homogeneous Markov chain. This is in particular true for the various variants of the nested partitions method (Shi and Ólafsson, 2000b; Ólafsson, 2004), and allows for eloquent analysis of the method. However, the disadvantage is that to guarantee time homogeneity, independence must be enforced between iterations, and independent simulation runs must be made for solutions, even if they have been previously estimated. Due to the computational expense of simulation runs, this is inefficient (Kim and Ólafsson, 2002). On the other hand, this property may sometimes be helpful when the variance of the estimates is high and previous biased estimates have caused the algorithm to make an incorrect move. In such cases, obtaining new estimates in every step may help the algorithm to recover from wrong moves as discussed in Section 3. However, we believe further research is needed to extend existing convergence results to allow reuse of simulation runs, and to determine the proper balance that also enables the algorithm to recover from incorrect moves.

8 Future directions

Due to their prevalence and effectiveness in simulation practice, we believe that metaheuristics will continue to be an important part of simulation methodology in the future, which also makes this an important area of research. Such research should focus on the simulation aspects of the problem, that is, making the simulation a driver rather than being secondary to the optimization routine. However, as this research is carried out, it is important to take into account the concerns of practitioners, where computational efficiency is the key issue. Thus, effectiveness should never be sacrificed at the expense of theoretical insights.

In this chapter, we have outlined two areas that we believe to be of particular importance, accounting for simulation noise in guiding the search and deriving convergence results. We start with the first issue of accounting for the simulation noise in the design of the metaheuristics. Too often, metaheuristics that are designed for deterministic, most often combinatorial, optimization problems are simply applied to a simulation problem with only minimal consideration into the effects of the simulation noise. As outlined in Section 3, two of the concerns include determining if a new (candidate) solution is really better and should be selected, and recovering from incorrect moves made due to the noisy performance estimates. Fortunately, several areas of research appear to be promising. Ideas from ordinal optimization and statistical selection can be incorporated to help determine if a move should be made, which addresses the first issue. Second, the local search can be embedded into a global guidance system, such as the nested partitions method, that allows the search to recover from incorrect moves. Such combination of metaheuristics has been found effective in combinatorial optimization, and we believe that such careful consideration of the simulation noise will improve the computational performance of the metaheuristics, and hence have great practical value.

The second main area of research discussed in this chapter is providing convergence results, which we explored in Section 7. We do believe that the ability to make such convergence statements is of value, but also think that the focus should be shifted away from the traditional asymptotic convergence results to finite time behavior and goal softening. For example, rather than proving asymptotic convergence to a global optimum, it would be of more practical value to show that a solution has been obtained that is within an indifference zone of the optimal performance with a pre-specified minimum probability. This terminology stems from ranking and selection, which can be used to guide search methods in simulation optimization. As noted in Section 7, this idea is also related to the goal softening of ordinal optimization. Further research along those directions would both make existing use of metaheuristics more rigorous and may lead to insights that would improve the algorithm performance.

References

April, J., Glover, F., Kelly, J.P., Laguna, M. (2003). Practical introduction to simulation optimization. In: *Proceedings of the 2003 Winter Simulation Conference*. IEEE Press, Piscataway, NJ, pp. 71–78.

Aytug, H., Bhattacharrya, S., Koehler, G.J. (1996). A Markov chain analysis of genetic algorithms with power of 2 cardinality alphabets. *European Journal of Operational Research* 96, 195–201.

Azadivar, F., Wang, J. (2000). Facility layout optimization using simulations and genetic algorithms. *International Journal of Production Research* 38 (17), 4369–4383.

Back, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, New York.

Boesel, J., Nelson, B.L., Ishii, N. (2003a). A framework for simulation optimization software. *IIE Transactions* 35 (3), 221–229.

Boesel, J., Nelson, B.L., Kim, S. (2003b). Using ranking and selection to "clean-up" after simulation optimization. *Operation Research* 51 (5), 814–825.

Dai, L. (1996). Convergence properties of ordinal comparison in the simulation of discrete event dynamic systems. *Journal of Optimization Theory & Applications* 91 (2), 363–388.

Dasgupta, D., Mcgregor, D.R. (1992). Nonstationary function optimization using the structured genetic algorithm. *Parallel Problem Solving from Nature* 2, 145–154.

Eglese, R.W. (1990). Simulated annealing: A tool for operation research. *European Journal of Operational Research* 46, 271–281.

Faccenda, J.F., Tenga, R.F. (1992). A combined simulation/optimization approach to process plant design. In: *Proceedings of the 1992 Winter Simulation Conference*. IEEE Press, Piscataway, NJ, pp. 1256–1261.

Fleischer, M.A. (1995). Simulated annealing: Past present and future. In: *Proceedings of the 1995 Winter Simulation Conference*. IEEE Press, Piscataway, NJ, pp. 155–161.

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

Glover, F. (1989). Tabu Search – Part I. *ORSA Journal on Computing* 1, 190–206.

Glover, F. (1990). Tabu Search – Part II. *ORSA Journal on Computing* 2, 4–32.

Glover, F., Hanafi, S. (2002). Tabu search and finite convergence. *Discrete Applied Mathematics* 119, 3–36.

Glover, F., Kochenberger, G.A. (2003). *Handbook of Metaheuristics*. Kluwer Academic Publishers, Boston, MA.

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

Glover, F., Laguna, M., Marti, R. (2003). Scatter search. In: Ghosh, A., Tsutsui, S. (Eds.), *Theory and Applications of Evolutionary Computation: Recent Trends*. Springer-Verlag.

Goldberg, D.E. (1989). *Genetic Algorithm in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA.

Greenhalgh, D., Marshall, S. (2000). Convergence criteria for genetic algorithm. *SIAM Journal on Computing* 30, 269–282.

Grefenstette, J.J. (1992). *Genetic Algorithm for Changing Environments*. *Parallel Problem Solving from Nature 2*. Elsevier, Amsterdam, The Netherlands.

Haddock, J., Mittenhall, J. (1992). Simulation optimization using simulated annealing. *Computer and Industrial Engineering* 22, 387–395.

Hall, J.D., Bowden, R.O. (1997). Simulation optimization by direct search: A comparative study. In: *Proceedings of the 6th International Industrial Engineering Research Conference*. IEE, Norcross, GA, pp. 298–303.

Hanafi, S. (2000). On the convergence of tabu search. *Journal of Heuristics* 7, 47–58.

Hansen, P., Mladenovic, N. (1999). An introduction to variable neighborhood search. In: Voss, S., et al. (Eds.), *Meta-Heuristics: Advances and Trends in Local Search Paradigms for Optimization*. Kluwer Academic Publishers, Boston, MA, pp. 433–458.

Ho, Y.C., Sreenivas, R., Vakili, P. (1992). Ordinal optimization of discrete event dynamic systems. *Journal of DEDS* 2 (2), 61–88.

Ho, Y.C., Cassandras, C.G., Chen, C.-H., Dai, L. (2000). Ordinal optimization and simulation. *Journal of the Operational Research Society* 51, 490–500.

Holland, J.H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, MI.

Ishibuchi, H., Murata, T. (1996). Multi-objective genetic local search algorithm. In: *Proceedings of IEEE International Conference on Evolutionary Computation*. IEEE Press, Piscataway, NJ, pp. 119–124.

Jacobson, S.H., Sullivan, K.A., Johnson, A.W. (1998). Discrete manufacturing process design optimization using computer simulation and generalized hill-climbing algorithm. *Engineering Optimization* 31, 247–260.

Johnson, A.W., Jacobson, S.H. (2002). On the convergence of generalized hill climbing algorithms. *Discrete Applied Mathematics* 119, 37–57.

Kelly, J.P. (2002). Simulation optimization is evolving. *INFORMS Journal on Computing* 14 (3), 223–225.

Kim, J., Ólafsson, S. (2002). Two-stage NP method with inheritance. In: *Proceedings of the 2002 Winter Simulation Conference*. IEEE Press, Piscataway, NJ, pp. 279–284.

Kirkpatrick, S., Gelatt Jr., C.D., Vecchi, M.P. (1983). Optimization by simulated annealing. *Science* 220, 671–680.

Liepins, G.E. (1992). On global convergence of genetic algorithms. *SPIE Neutral and Stochastic Methods in Image and Signal Processing* 1766, 61–65.

Liepins, G.E., Hilliard, M.R. (1989). Genetic algorithm: Foundations and applications. *Annals of Operations Research* 21, 31–58.

Muhlenbein, H. (1997). Genetic algorithms. In: Aarts, E., Lenstra, J.K. (Eds.), *Local Search in Combinatorial Optimization*. Wiley, New York, pp. 137–172.

Ólafsson, S. (2004). Two-stage nested partitions method for stochastic optimization. *Methodology and Computing in Applied Probability* 6, 5–27.

Ólafsson, S., Kim, J. (2002). Simulation optimization. In: *Proceedings of the 2002 Winter Simulation Conference*. IEEE Press, Piscataway, NJ, pp. 79–84.

Pichitlamken, J., Nelson, B.L. (2003). A combined procedure for optimization via simulation. *ACM Transactions on Modeling and Computer Simulation* 13 (2), 108–133.

Resende, M.G.C., Ribeiro, C.C. (2003). Greedy randomized adaptive search procedures. In: Glover, A., Kochenberger, A. (Eds.), *Handbook of Metaheuristics*. Kluwer Academic Publishers, Boston, MA.

Reynolds, D., Gomatam, J. (1996). Stochastic modeling of genetic algorithms. *Artificial Intelligence* 82, 303–330.

Shi, L., Chen, C.H. (2000). A new algorithm for stochastic discrete resource allocation optimization. *Journal of Discrete Event Dynamic Systems: Theory and Applications* 10, 271–294.

Shi, L., Men, S. (2003). Optimal buffer allocation in production lines. *IIE Transactions* 35, 1–10.

Shi, L., Ólafsson, S. (2000a). Nested partitions method for global optimization. *Operations Research* 48, 390–407.

Shi, L., Ólafsson, S. (2000b). Nested partitioned method for stochastic optimization. *Methodology and Computing in Applied Probability* 2, 271–291.

Shi, L., Ólafsson, S., Chen, Q. (1999). A new hybrid optimization method. *Computers and Industrial Engineering* 36, 409–426.

Sullivan, K.A., Jacobson, S.H. (2001). A convergence analysis of generalized hill climbing algorithms. *IEEE Transaction on Automatic Control* 46 (8), 1288–1293.

Tian, P., Ma, J., Fan, Z. (1997). A stochastic tabu search strategy and its global convergence. In: *IEEE International Conference on Systems, Man and Cybernetics*. IEEE Press, Piscataway, NJ, pp. 410–414.

Tompkins, G., Azadivar, F. (1995). Genetic algorithms in optimizing simulated systems. In: *Proceedings of the 1995 Winter Simulation Conference*. IEEE Press, Piscataway, NJ, pp. 757–762.

Vavak, F., Fogarty, T.C. (1996). Comparison of steady state and generational genetic algorithms for use in nonstationary environments. In: *Proceedings of IEEE International Conference on Evolutionary Computation*. IEEE Press, Piscataway, NJ, pp. 192–195.

Watson, J., Beck, J.C., Howe, A.E., Whitley, L.D. (2003). Problem difficulty for tabu search in job-shop scheduling. *Artificial Intelligence* 143 (2), 189–217.

Yoshitomi, Y., Yamaguchi, R. (2003). A genetic algorithm and the Monte Carlo method for stochastic job-shop scheduling. *International Transaction of Operational Research* 10, 577–596.