Proceedings of the 2005 Winter Simulation Conference
M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, eds.

SIMULATION OPTIMIZATION USING TABU SEARCH: AN EMPERICAL STUDY

Abdullah Konak

Information Sciences and Technology
Penn State-Berks
Reading, PA 19610, U.S.A.

Sadan Kulturel-Konak

Management Information Systems
Penn State-Berks
Reading, PA 19610, U.S.A.

ABSTRACT

This paper proposes alternative strategies to perform simulation within a simulation optimization algorithm based on tabu search. These strategies are tested empirically on a stochastic knapsack problem. Results have shown that the way simulation is implemented and the number of simulation replications have a profound effect on the performance of tabu search.

1 INTRODUCTION

The term "simulation optimization" is usually used in the literature with regard to the following problem setting. A system or function has a set of input variables and a single (or multiple) output(s). The relationship between the input variables and the system output is so complex that simulation is used to estimate the output for a given setting of the input variables. The underlying optimization problem is to find a feasible setting of the input variables in order to minimize (or maximize) the system output. Simulation optimization approaches are among the most frequently used tools to solve many real-world problems. Today, many commercial simulation software packages include simulation optimization toolboxes allowing users to search for a good system design in an automated fashion. Andradottir (1998) and Fu (2002) surveyed common approaches to simulation optimization. April et al. (2003), Law and McComas (2000, 2002), Ling, Liang and Da-zhong (2003), Olafsson and Kim (2002), Swisher et al. (2000) also discussed latest advancements in simulation optimization.

The research in this paper focuses on a class of combinatorial optimization problems where the objective function is evaluated using simulation since its exact computation is analytically impossible and/or computationally infeasible. Examples of such problems include network design, facilities planning, scheduling, reliable system design, supply chain, and financial systems. Traditional simulation optimization approaches such as response surface and stochastic approximation are not usually applicable in the case of combinatorial optimization problems due to non-convexity and discontinuity of the solution space. Most simulation optimization approaches to combinatorial optimization use meta-heuristic techniques such as genetic algorithms (GA), tabu search (TS), simulated annealing (SA), or ant colony (AC), that are based on ranking or comparison of candidate solutions visited during the search. When simulation is used to evaluate the objective function, however, ranking and comparison of candidate solutions involves an error due to the randomness in simulation outputs. Therefore, the performance of a heuristic search algorithm may depend on the level of noise in the objective function. Clearly, the magnitude of this noise can be reduced by increasing the number of simulation replications to evaluate solutions. However, this approach may not be practical if good solutions are sought in a limited time, or it will be inefficient if a rigorous evaluation is unnecessary. It is usually difficult to determine the minimum number of simulation replications that will reduce the effect of the randomness on the performance of a search algorithm at a negligible level.

Several researchers studied the impact of noise on the performance of meta-heuristic approaches and proposed strategies to maximize it for a given CPU time limit. The majority of these studies focus primarily on GA. Fitzpatrick and Grefenstette (1988) explored the tradeoffs between the number of simulation replications to evaluate each solution and the number of solutions evaluated in each generation (population size) under a given CPU time limit. They showed empirically that there exists an optimal number of replications that provides the best GA performance. Harik et al. (1999) reported that GA performance highly depends on population size at the presence of noise, and the higher the noise level is, the bigger the population size should be.

Aizawa and Wah (1993) proposed using a dynamic number of simulation replications as opposed to a static one. In this approach, the number of replications varies from generation to generation as being lower at the beginning and higher toward the end. The justification for this approach is that a GA population usually includes very different solutions at the beginning and similar ones toward the end of the search, and more simulation replications are required to accurately rank solutions similar to each other. Miller and Goldberg (1995) studied how selection methods and parameters of a GA can be tuned to improve the performance in noisy objective functions. Arnold and Beyer (2003) studied the performance of different evolutionary strategies at the presence of noise.

Boesel, Nelson and Ishii (2003) developed a GA based simulation-optimization software system where the number of replications to evaluate each solution is adjusted based on the variance of the simulation output. In addition, this system features error control during and at the end of the search to minimize computational effort. The error control during the research uses a group ranking concept where the solutions in a group are statistically equals and have the same rank. Groups are formed based on the expected value and variance of solutions. For the error control at the end of the search, instead of only a single best solution, a set of solutions which are not statistically superior to each others are kept during the search, and at the termination these solutions are compared to each other using additional simulation replications to identify the best solution from this set within a statistical error margin.

As seen in the literature cited above, the majority of work on simulation optimization using heuristic approaches were mainly studied and developed for GA. Although TS is frequently used to solve combinatorial optimization problems, similar work has not been conducted for TS. In this paper, we investigate the performance of TS in simulation optimization on a test problem. In addition, we devise and empirically test several strategies to maximize the performance of TS in simulation optimization.

2 TABU SEARCH

TS, which was first developed by Glover (1989, 1990), is used in this study. Since its first introduction, TS has become an effective heuristic method for many combinatorial optimization problems with large and complex search spaces in scheduling, telecommunications, transportation, routing, network design, graph theory, manufacturing, financial analysis, and constraint satisfaction. Excellent material about TS can be found in Glover, Taillard and de Werra (1993) and Glover and Laguna (1997).

TS guides a local (neighborhood) search procedure to find a global optimum. Therefore, TS with its local search property is an appropriate optimization tool to search neighborhoods efficiently. A TS heuristic has two important features: the move operator and the tabu list. A new solution (i.e., candidate solution) is produced by a move operator with small perturbations to a current solution. The set of all solutions produced by the move operator is called the neighborhood of the current solution. All solutions in the neighborhood are evaluated according to their objective function values, and the best objective function value defines the best candidate solution. However, not necessarily all moves improve the objective function; therefore, it is possible to re-visit a solution, which is called cycling. To avoid cycling, recently performed moves are stored on a tabu list for a certain number of iterations. In basic TS, moves on the tabu list are prohibited; therefore, at each iteration the algorithm is forced to select the best candidate from not recently selected moves (i.e., ones not on the tabu list). In some cases, however, a tabu move may improve upon the best feasible so far, then it can be accepted as the best candidate, which is called an aspiration criterion. Moreover, another rationale behind using an aspiration criterion is that tabu conditions may also forbid moves directing the search towards unvisited yet attractive solutions. After all neighborhoods of the current solution are investigated and the best candidate is determined, the tabu list is updated, the best candidate is assigned to the current solution, and the entire process starts again until the stopping condition is met.

3 PROBLEM DESCRIPTION

In this paper, the stochastic knapsack problem (SKP) is used as the test problem. The SKP is formulated as follows:

$$\begin{aligned}
&Max \quad z = \Pr\left(\sum_{i=1}^{n} a_i x_i \geq C\right) \\
&st \\
&\sum_{i=1}^{n} w_i x_i \leq W \\
&x_i \in \{0,1\} \quad \forall i
\end{aligned}$$

where w_i and a_i are the weight and the return of item i. In the formulation, the weights are deterministic, but returns a_i, ..., a_n are independent random variables with known density functions. The objective is to maximize the probability that the total return will exceed a threshold value C. Additional information on the SKP and its application areas can be found in Kleywegt and Papastavrou (1998). Note that the distribution of $\sum_{i=1}^{n} a_i x_i$ can be analytically derived for some cases (e.g., a_i are normally distributed) and the objective function, $z(\mathbf{x})$, can be exactly calculated for a given decision variable vector $\mathbf{x}=\{x_i\}$. However, the objective of the research herein is not solving the SKP, rather it is used as a test problem to understand the relationship between TS performance and the level of noise in the objective function. Therefore, we assume that a_i, ..., a_n are independent general random variables and

$z(\mathbf{x}) = \Pr\left(\sum_{i=1}^{n} a_i x_i \geq C\right)$ is estimated using a simulation as follows:

```
Select a sample size $K$, and set $s$=0
For $k$=1, ..., $K$ do {
    $sum$=0
    For $i$=1,...,$n$ do {
        If $x_i$=1, then {
        Generate random variable $a_i$
        $sum$=$sum$+ $a_i$
        }
    }
    If $sum \geq C$, then set $s = s + 1$
}
Return an estimation of $z(\mathbf{x})$ as $\hat{z}(\mathbf{x}) = s / K$.
```

We consider a generic TS with algorithmic features that are frequently applied in the literature. The TS algorithm starts with a random feasible solution **x**(0). At iteration *t*, swap moves create a neighborhood *N*(**x**(*t*)) of the current solution **x**(*t*) as follows: For $i<j\leq n$, swap move (*i*, *j*) exchanges the values of decision variables x_i and x_j, and swap move (*i*, *i*) flips the value of decision variable x_i only. Note that if $x_i = x_j$, swap move (*i*, *j*) does not generate a new solution from **x**(*t*); therefore, such moves are not applied. The neighborhood *N*(**x**(*t*)) may include infeasible solutions due to the weight constraint. Several approaches exist to deal with infeasibility constraints in TS Kulturel-Konak et al. (2004). One of the most widely used constraint handling approaches is penalty functions. An infeasible solution **x** is penalized using an adaptive penalty approach as follows:

$$f(\mathbf{x}) = \hat{z}(\mathbf{x}) - \theta(t)\max(w(\mathbf{x}) - W, 0) / w(\mathbf{x})$$

where $\theta(t)$ is the adaptive penalty factor at iteration *t*, and $w(\mathbf{x})$ is the weight of solution **x**. Initially $\theta(0) = 1$, and in the following iterations its value is updated as follows. $\theta(t+1) = 2\theta(t)$ if the best candidate solution was infeasible in iterations *t*, *t-1*, ..., *t-ρ*; $\theta(t+1) = 0.5\theta(t)$ if the best candidate solution was feasible in iterations *t*, *t*-1, ..., *t-ρ*; and otherwise, $\theta(t+1) = \theta(t)$. This penalty approach was first proposed by Gendreau, Hertz and Laporte (1994) and shown to be robust on various problems by Kulturel-Konak et al. (2004).

The tabu list is a recency-based memory where an $n\times n$ matrix $\mathbf{r}=\{r_{ij}\}$ stores swap moves that have been applied to create current solutions in most recent iterations. If a new current solution **x**(*t*+1) is created from solution **x**(*t*) using a swap move (*i*, *j*), then r_{ij} is set to a random positive number between $ts_{\min}$ and $ts_{\max}$, and every other entity of **r** is reduced by one. A swap move (*i*, *j*) is said to be tabu and not allowed if $r_{ij}\geq 0$. However, a simple aspiration criterion is used by allowing a tabu move that will result in a solution improving upon the best feasible solution ($\mathbf{x}^{**}$). Note that the majority of TS implementations also use a similar aspiration criterion. A subset $\tilde{N}(\mathbf{x}(t))$ of $N(\mathbf{x}(t))$ which includes non-tabu solutions, and the set of solutions allowed by the aspiration criterion is called admissible neighborhood. Solution $\mathbf{x}\in \tilde{N}(\mathbf{x}(t))$ with the maximum fitness value $f(\mathbf{x})$ is selected as the current solution **x**(*t*+1) in the next iteration. The overall algorithm of the TS is given as follows:

```
Set $t$=0, generate a random solution $\mathbf{x}(t)$
Do {
        Generate $\tilde{N}(\mathbf{x}(t))$ from $\mathbf{x}(t)$
        Evaluate $f(\mathbf{x})$ for each $\mathbf{x}\in\tilde{N}(\mathbf{x}(t))$, and update $\mathbf{x}^{**}$ and $\mathbf{x}^{*}$.
        Set $\mathbf{x}(t+1)$= $\mathbf{x}^{*}$, update $\mathbf{r}$ and $\theta(t)$,
        set $t$=$t$+1
        } While (the stopping criterion is not satisfied)
Estimate $z(\mathbf{x}^{**})$ with a very high number of replications
Return $\mathbf{x}^{**}$.
```

4 EXPERIMENTAL STUDY AND DISCUSSIONS

The main objective of computational experiments in this section is to compare alternative strategies to evaluate new solutions and update the best candidate $\mathbf{x}^{*}$ and the best feasible solution $\mathbf{x}^{**}$. The test problem used in experiments is a 100-item binary SKP. For each item *i*, the return coefficient a_i is exponentially distributed with a mean of μ_i which is randomly generated in a way that it is positively correlated with weight w_i. Since the overall objective is to devise a strategy for implementing simulation within the TS procedure that will maximize its performance for a given CPU time, a stopping criterion of 60-second CPU time limit was used. At the termination of the TS procedure, the best feasible solution found $\mathbf{x}^{**}$ is estimated using 10^6 simulation replications to minimize error while comparing alternatives strategies. The other parameters of the TS procedure are as follows: $\theta(0) = 1$, $ts_{\min}$=10, $ts_{\max}$=20, and ρ=5.

Strategy 1:

```
Evaluate $\mathbf{x}$ by using $K_1$ simulation replications
If $f(\mathbf{x}) \geq f(\mathbf{x}^{*})$, then set $\mathbf{x}^{*} = \mathbf{x}$
If $\hat{z}(\mathbf{x}) \geq \hat{z}(\mathbf{x}^{**})$ and $w(\mathbf{x}) \leq W$, then set $\mathbf{x}^{**} = \mathbf{x}$
```

This strategy is a naive approach which mimics a TS with a deterministic objective function. Nonetheless, it is frequently used in practice. The TS algorithm was run for different values of K_1 from 50 to 6,400 as shown in Figure 1. Note that since the CPU time is fixed, the less the effort to evaluate each solution, the higher the number of solutions searched. For each value of K_1, the algorithm was run 20 times by starting with a different random initial solution in each run. Figure 1 gives a box-plot of the results, which clearly shows a trade-off between the effort to evaluate solutions and the total number of solutions evaluated. These results suggest that an optimum level of K_1 exists. When K_1 is increased above a certain level, the performance deteriorates since an adequate number of solutions cannot be searched with the CPU time limit; and when K_1 is less than a certain level, the performance also deteriorates since the objective function is too noisy and the search becomes almost random.

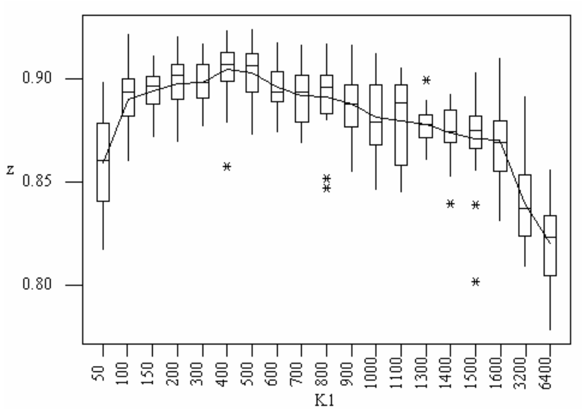


Figure 1: A box-plot of solutions found in 20 replications for different levels of K_1

Strategy 2:

```
Evaluate x by using $K_1$ simulation replications
If $f(\mathbf{x}) \geq f(\mathbf{x}^*)$, then set $\mathbf{x}^* = \mathbf{x}$
If $\hat{z}(\mathbf{x}) \geq \hat{z}(\mathbf{x}^{**})$ and $w(\mathbf{x}) \leq W$, then {
    Evaluate x with additional $K_2$ simulation replications
    Update $\hat{z}(\mathbf{x})$
    If $\hat{z}(\mathbf{x}) \geq \hat{z}(\mathbf{x}^{**})$, then set $\mathbf{x}^{**} = \mathbf{x}$.
    }
```

The main objective of this strategy is to reduce level of error while updating $\mathbf{x}^{**}$. The TS algorithm was run for different values of K_1 from 50 to 1,000 and K_2=2,000. Results were compared to the ones found by Strategy 1 for the same levels of K_1. Figure 2 shows the average value of $\hat{z}(\mathbf{x}^{**})$ over 20 replications for both strategies. Clearly, Strategy 2 is superior in all cases. In addition, this strategy was proved to be more robust with respect to low values of parameter K_1. Notice that Strategy 2 is almost flat for low values of K_1.

Strategy 3:

```
Evaluate x by using $K_1$ simulation replications
If $f(\mathbf{x}) \geq f(\mathbf{x}^*)$, then {
    Evaluate x with additional $K_3$ replications
    Update $f(\mathbf{x})$
    If $f(\mathbf{x}) \geq f(\mathbf{x}^*)$, then set $\mathbf{x}^* = \mathbf{x}$.
    }
If $\hat{z}(\mathbf{x}) \geq \hat{z}(\mathbf{x}^{**})$ and $w(\mathbf{x}) \leq W$, then {
    Evaluate x with additional $K_2$ replications
    Update $\hat{z}(\mathbf{x})$.
    If $\hat{z}(\mathbf{x}) \geq \hat{z}(\mathbf{x}^{**})$, then set $\mathbf{x}^{**} = \mathbf{x}$.
    }
```

The main objective of this strategy is to reduce level of error while updating the best candidate solution $\mathbf{x}^*$, thereby reducing the probability of taking a wrong turn during the search. The TS algorithm was run for all combinations of K_1=50, 100, 200 and K_3=300, 400, 500 while keeping K_2 constant at 2,000. Results were analyzed by the two-way ANOVA. Both K_1 and K_3 turned out to be statistically significant with p-values of 0.06 and 0.09, respectively. The best combination was K_1=50 and K_3=500. Figure 2 depicts the confidence intervals for K_1 and K_3. The results indicate that the best approach to evaluate solutions might be a hierarchical one which initially evaluates each solution roughly and then evaluates promising ones rigorously. This approach effectively allocates computational effort between evaluating and exploring new solutions.

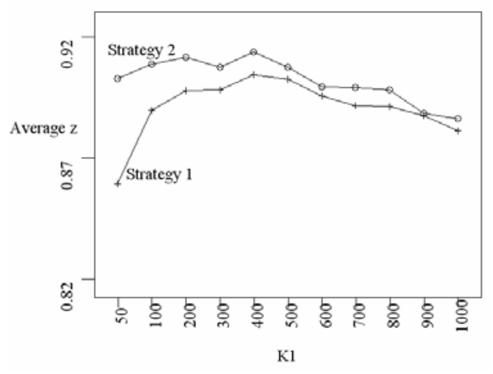
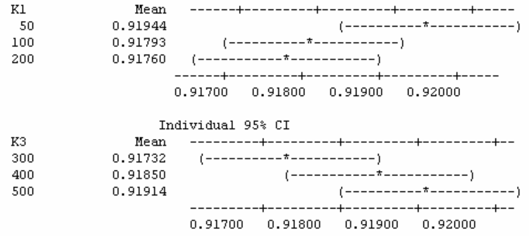


Figure 2: A compression of Strategies 1 and 2



```
                           Individual 95% CI
K1             Mean    ------+---------+---------+---------+-----
 50         0.91944                       (----------*-----------)
100         0.91793        (----------*-----------)
200         0.91760    (-----------*-----------)
                     ------+---------+---------+---------+-----
                     0.91700   0.91800   0.91900   0.92000

                         Individual 95% CI
K3             Mean    ---------+---------+---------+---------+--
300         0.91732     (----------*-----------)
400         0.91850                (-----------*-----------)
500         0.91914                       (----------*-----------)
                       ---------+---------+---------+---------+--
                       0.91700   0.91800   0.91900   0.92000
```

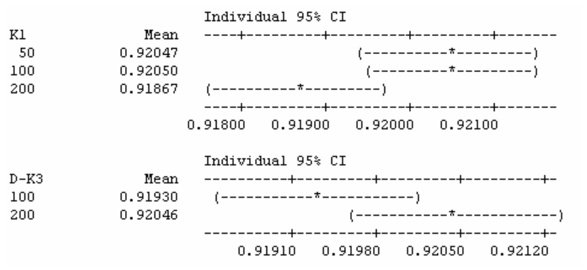
Figure 3: 95% confidence intervals for the factor levels used in the two-way ANOVA for Strategy 3

Strategy 4:
This approach is an extension of Strategy 3 where the best candidate $\mathbf{x}^*$ and the best feasible solution found $\mathbf{x}^{**}$ are gradually evaluated on an as needed basis.

```
Set $K_3$=0
For $\mathbf{x} \in \tilde{N}(\mathbf{x}(t))$ do {
    Evaluate x by using $K_1$ simulation replications
    If $f(\mathbf{x}) \geq f(\mathbf{x}^*)$, then {
        If $K_3 \leq K_{3\max}$, then {
            Evaluate $\mathbf{x}^*$ using additional $\Delta K_3$ replications
            Update $\hat{z}(\mathbf{x}^*)$
            set $K_3 = K_3 + \Delta K_3$
            }
        Evaluate x using additional $K_3$ replications
        Update $\hat{z}(\mathbf{x})$.
        If $f(\mathbf{x}) \geq f(\mathbf{x}^*)$, then set $\mathbf{x}^* = \mathbf{x}$
    }
    If $\hat{z}(\mathbf{x}) \geq \hat{z}(\mathbf{x}^{**})$ and $w(\mathbf{x}) \leq W$ then {
        If ($K_2 \leq K_{2\max}$), then {
            set $K_2 = K_2 + \Delta K_2$
            Evaluate $\mathbf{x}^{**}$ using additional $\Delta K_2$ replications
            Update $\hat{z}(\mathbf{x}^{**})$.
        }
        Evaluate x using additional $K_2$ replications
        Update $\hat{z}(\mathbf{x})$.
        If $\hat{z}(\mathbf{x}) \geq \hat{z}(\mathbf{x}^{**})$, then set $\mathbf{x}^{**} = \mathbf{x}$.
        }
}
```

In this approach, the best candidate $\mathbf{x}^*$ is reevaluated each time a solution **x** contests its best candidacy. Therefore, if a solution **x** is highly superior in its neighborhood, it could be identified as the best candidate without requiring a large number of simulation replications. On the contrary, if solutions in a neighborhood have similar fitness values, then this procedure will increase the number of simulation replications to reduce the error while comparing solutions. To test this approach, the TS algorithm was run for K_1=50, 100, 200, ΔK_3 =100, 200 with $\Delta K_{3\max}$=600, and ΔK_2= 500 with $K_{2\max}$=2,000. Results were analyzed by the two-way ANOVA using K_1 and ΔK_3 as the factors, which turned out to be statistically significant with *p*-values of 0.02 and 0.06, respectively. Figure 4 depicts the confidence intervals for factor K_1 and ΔK_3 (D-*K*3 in the figure). The results are similar to Strategy 3. Basically, the best strategy seems to evaluate each solution roughly first, then to evaluate promising ones with a higher number of replications.

Strategy 5:
This is an extension of Strategy 3 where a solution **x** is re-evaluated with additional K_3 replications if $\hat{z}(\mathbf{x}) \geq \hat{z}(\mathbf{x}^*)$ and $w(\mathbf{x}) \leq W$. The main idea of this strategy is not to waste simulation time by rigorously evaluating infeasible solutions. The TS algorithm was run using this strategy and Strategy 3 with K_1=50, 100, 200, K_3 =300, 400, 500, and K_2=2,000, 3000. Then, the results obtained by both strategies were compared using a paired-*t* test. As a result of this comparison, Strategy 3 was proved to be more effective than Strategy 5 with a *p*-value of almost zero despite the fact that Strategy 5 was able to search 19% more solutions than Strategy 3 on the average. This result was unexpected since we had assumed that the penalty of an infeasible solution would have a higher weight than its objective function value in the fitness of the solutions. However, this relationship turned out to be very complex when an adaptive penalty function such as the one used herein. When neighborhoods including both feasible and infeasible solutions were closely analyzed, it turned out that a rigorous estimation of *z*(**x**) is also important for infeasible solutions since an infeasible solution was frequently selected as the best candidate.



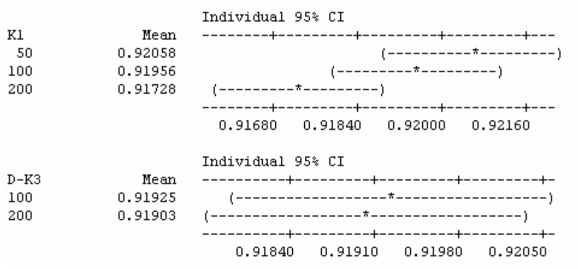
```
                         Individual 95% CI
K1          Mean    ----+---------+---------+---------+-------
 50      0.92047                     (----------*---------)
100      0.92050                      (---------*---------)
200      0.91867    (----------*---------)
                    ----+---------+---------+---------+-------
                  0.91800   0.91900   0.92000   0.92100

                         Individual 95% CI
D-K3        Mean    ---------+---------+---------+---------+-
100      0.91930     (-----------*-----------)
200      0.92046                     (-----------*------------)
                    ---------+---------+---------+---------+-
                       0.91910   0.91980   0.92050   0.92120
```

Figure 4: 95% confidence intervals for the factor levels used in the two-way ANOVA for Strategy 4

Strategy 6:
The strategies described above do not consider the standard deviation of simulation estimation $\hat{z}(\mathbf{x})$ while comparing solutions. Strategy 6 is an extension of Strategy 4 where standard deviation of the estimation is used while comparing a solution **x** with the best feasible solution $\mathbf{x}^{**}$. Strategy 4 is modified such that a new solution **x** is reevaluated if $\hat{z}(\mathbf{x}^{**}) - \hat{z}(\mathbf{x}) \leq k\sigma_{\hat{z}(\mathbf{x}^{**})-\hat{z}(\mathbf{x})}$ where $\sigma_{\hat{z}(\mathbf{x}^{**})-\hat{z}(\mathbf{x})}$ is the standard deviation of the difference between estimations $\hat{z}(\mathbf{x}^{**})$ and $\hat{z}(\mathbf{x})$. Therefore, this strategy reduces the probability of cases where $z(\mathbf{x}) \geq z(\mathbf{x}^{**})$ and $\hat{z}(\mathbf{x}) < \hat{z}(\mathbf{x}^{**})$ compared to the other strategies. However, its main disadvantage is that more solutions are expected to be evaluated rigorously. The runs were performed with the same parameter settings used in Strategy 4 and *k*=1. When the results were compared to Strategy 4 using a paired-*t* test, no significant difference was observed between these two

strategies in terms of solution quality (p-value=0.222). However, when results were analyzed by a two-way ANOVA using K_1 and ΔK_3 as the factors, which were both statistically significant in Strategy 4, ΔK_3 turned out to be not significant with a p-value of 0.85. In other words, this strategy performed equality well for both levels of ΔK_3 (see Figure 5). Considering the standard deviation while comparing new solutions with the best feasible solution provides a cushion against the error of rejecting a true new best feasible solution at the first comparison, reducing the probability of this error. Therefore, this feature may undermine the function of ΔK_3 at some degree.

Figure 6 shows a box plot of the results over 20 replications for the best case of each strategy. The best strategy seems to be strategy 4. Although it is difficult to draw general conclusion based on a single problem, it seems that evaluating solutions in a hierarchical manner is a promising approach.



```
                          Individual 95% CI
K1          Mean    --------+---------+---------+---------+---
 50      0.92058                       (----------*---------)
100      0.91956                  (---------*---------)
200      0.91728     (---------*---------)
                    --------+---------+---------+---------+---
                      0.91680   0.91840   0.92000   0.92160

                          Individual 95% CI
D-K3        Mean    ----------+---------+---------+---------+-
100      0.91925       (------------------*------------------)
200      0.91903    (------------------*------------------)
                    ----------+---------+---------+---------+-
                        0.91840   0.91910   0.91980   0.92050
```

Figure 5: 95% confidence intervals for the factor levels used in the two-way ANOVA for Strategy 6

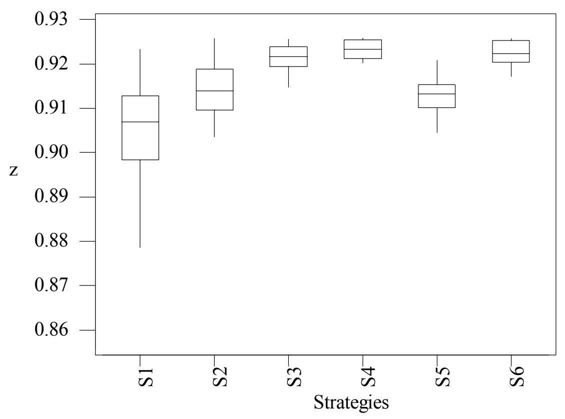


Figure 6: A box plot of the results over 20 replications for the best case of each strategy

5 CONCLUSIONS

The research in this paper introduced alternative strategies in simulation optimization using TS. These strategies are demonstrated and empirically compared using a SKP test instance. Experimental results have demonstrated that the performance of a simulation optimization using TS may highly depend on how the simulation is conducted during the search and how its output is used to guide search.

Although it is difficult to generalize the results found in this paper since they depend on a single problem instance, it is clear that TS performance can be improved by simple adjustments in a TS procedure instead of increasing the number of simulation replications to reduce the noise. It would be an interesting further research to experiment similar ideas on different problems in order to draw general conclusions.

REFERENCES

Aizawa, A. N. and B. W. Wah 1993. Dynamic control of genetic algorithms in a noisy environment. In *Proceedings of ICGA-93: Fifth International Conference on Genetic Algorithms*, 17-22 July 1993, ed. 48-55. Urbana-Champaign, IL, USA, Morgan Kaufmann.

Andradottir, S. 1998. Simulation Optimization. In *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*, ed. J. Banks, J. S. Carson, B. L. Nelson and D. M. Nicol, New York: John Wiley.

April, J., F. Glover, J. P. Kelly and M. Laguna 2003. Practical introduction to simulation optimization. In *Proceedings of the 2003 Winter Simulation Conference*, 7-10, eds. S. Chick, P.J. Sanchez, D. Ferrin, and D.G. Morrice, 71-78. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers.

Arnold, D. V. and H.-G. Beyer 2003. A comparison of evolution strategies with other direct search methods in the presence of noise. *Computational Optimization and Applications* 24: 135-159.

Boesel, J., B. L. Nelson and N. Ishii 2003. A framework for simulation-optimization software. *IIE Transactions* 35: 221-229.

Fitzpatrick, J. M. and J. J. Grefenstette 1988. Genetic algorithms in noisy environments. *Machine Learning* 3: 101-120.

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

Gendreau, M., A. Hertz and G. Laporte 1994. A tabu search heuristic for the vehicle routing problem. *Management Science* 40: 1276-1290.

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. and M. Laguna 1997. *Tabu Search*. Boston, MA.: Kluwer Academic Publishers.

Glover, F., E. Taillard and D. de Werra 1993. A User's Guide to Tabu Search. *Annals of Operations Research* 41: 3-28.

Harik, G., E. Cantu-Paz, D. E. Goldberg and B. L. Miller 1999. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation* 7: 231-253.

Kleywegt, A. J. and J. D. Papastavrou 1998. The dynamic and stochastic knapsack problem. *Operations Research* 46: 17-35.

Kulturel-Konak, S., B. A. Norman, D. W. Coit and A. E. Smith 2004. Exploiting Tabu Search Memory in Constrained Problems. *INFORMS Journal on Computing* 16: 241-254.

Law, A. M. and M. G. McComas 2000. Simulation-based optimization. In *Proceedings of the Winter Simulation Conference 2000*, 10-13 Dec. 2000, eds. J.A. Joines, R. R. Barton, K. Kang, and P. Fishwick. 46-9. New Jersey: Institute of Electrical and Electronics Engineers.

Law, A. M. and M. G. McComas 2002. Simulation-based optimization. In *Proceedings of the 2002 Winter Simulation Conference*, 8-11 Dec. 2002, eds. E. Yuceasan, C.H. Chen, J. L. Snowdon, J.M. Charnes41-4. New Jersey: Institute of Electrical and Electronics Engineers.

Ling, W., Z. Liang and Z. Da-zhong 2003. Advances in simulation optimization. *Control and Decision* 18: 257-271.

Miller, B. L. and D. E. Goldberg 1995. Genetic algorithms, tournament selection, and the effects of noise. *Complex Systems* 9: 193-212.

Olafsson, S. and J. Kim 2002. *Simulation optimization*. In Proceedings of the 2002 Winter Simulation Conference, 8-11 Dec. 2002, ed. 79-84. New Jersey: Institute of Electrical and Electronics Engineers.

Swisher, J. R., P. D. Hyden, S. H. Jacobson and L. W. Schruben 2000. A survey of simulation optimization techniques and procedures. In *Proceedings of the Winter Simulation Conference 2000*, eds. J.A. Joines, R. R. Barton, K. Kang, and P. Fishwick, 10-13 Dec. 2000, ed. 119-28. New Jersey: Institute of Electrical and Electronics Engineers.

AUTHOR BIOGRAPHIES

ABDULLAH KONAK, Ph.D. is an Assistant Professor of Information Sciences and Technology at Penn State Berks. He received his B.S. degree in Industrial Engineering from Yildiz Technical University, Istanbul, Turkey, M.S. in Industrial Engineering from Bradley University, and Ph.D. in Industrial Engineering from University of Pittsburgh. Previous to this position, he was an instructor in the Department of Systems and Industrial Engineering at Auburn University for two years. His current research interest is in the application of Operations Research techniques to complex engineering problems, including such topics as telecommunications network design, network reliability analysis/optimization, facilities design, and data mining. He is a member of IIE and INFORMS. His email address is <konak@psu.edu> and his Web address is <www.personal.psu.edu/auk3>.

SADAN KULTUREL-KONAK, Ph. D. is an Assistant Professor of Management Information Systems at Penn State Berks. She received her degrees in Industrial Engineering: B.S. from Gazi University, Turkey in 1993, M.S. from the Middle East Technical University, Turkey in 1996 and from the University of Pittsburgh in 1999,and Ph.D. from Auburn University in 2002. Her research interests are in modeling and optimization of complex systems and robustness under uncertainty with applications to facility layout. She is a member of INFORMS, IIE, SWE, the Operational Research Society of Turkey, Alpha Pi Mu, and Phi Kappa Phi. Her email address is <sadan@psu.edu> and her Web address is <www.personal.psu.edu/sxk70>.