Proceedings of the 2005 International Conference on Simulation and Modeling
V. Kachitvichyanukul, U. Purintrapiban, P. Utayopas, eds.

OPTIMIZING A CALL CENTER SIMULATION MODEL

Juta Pichitlamken

Department of Industrial Engineering
Faculty of Engineering, Kasetsart University
50 Paholyothin Rd.
Bangkok 10900, THAILAND

ABSTRACT

We consider a call center simulation model with two types of traffic and two types of agents. Outbound calls are served only by blend agents, whereas inbound calls can be served by either inbound-only or blend agents. The objective is to find a staffing policy that satisfies some service requirements, in particular the probability that a waiting time of an incoming customer is smaller than some threshold. In this work, we apply GoLo, an optimization-via-simulation algorithm which consists of a global guidance system, a selection-of-the-best procedure and local improvement.

1 INTRODUCTION

We consider a telephone call center with two types of traffic, *inbound* and *outbound*, and two types of agents, *inbound-only* and *blend*. The number of agents of each type can vary from day to day and within each day. The inbound calls arrive according to a Poisson process whose rate may itself evolve as a stochastic process. When traffic is too high, new inbound calls must wait in a queue. For inbound traffic, we consider abandonment, i.e., some customers may not stay in the queue once learning that they are put on hold, or they may leave after spending some time waiting.

When the inbound traffic is low, and some blend agents are idle, an automatic dialer composes multiple outbound calls in parallel (trying to reach potential customers, e.g., for marketing or direct sales), in order to increase the productivity of the center. *Mismatches* occur when more customers are reached by outbound calls than the number of idle agents. The outbound calls are served only by blend agents, whereas inbound calls can be served by either type. See Pichitlamken et al. (2003) for more details.

Traditionally, call center operation is often modelled as a queueing system (see Koole and Mandelbaum 2002 and Gans, Koole, and Mandelbaum 2002 for extensive surveys). Beside Markovian queueing models, stochastic discrete-event simulation is also used because it is highly flexible, i.e., a simulation model can be tailored to specific details of call centers and is easy to modify. The simulation model also allows an analyst to do a what-if analysis and learn additional information that may otherwise not be available, e.g., times that customers are willing to wait before hanging up. The growing number of papers on call center simulation at the Winter Simulation Conference (`http://www.wintersim.org`) show that simulation is becoming an important analytical tool for call center management.

Among many issues in call center operation, we consider the so-called *staffing problem* where the goal is to determine the number of agents required in each time period to achieve a certain level of service quality. In our case, the quality of service (QoS) is defined as the fraction of inbound calls answered within 20 seconds or less. To our knowledge, most works on the staffing problem consider only inbound call centers with one type of agents, whereas our call center handles both inbound and outbound calls, and there are inbound-only and blend agents. Atlason, Epelman, and Henderson (2002) combines an iterative cutting plane method with simulation. An integer program is solved to generate a staffing policy, which is then subsequently evaluated via a simulation model to see if it provides a satisfactory service level. Related to this idea are Mason, Ryan, and Panton (1998) and Ingolfsson and Cabral (2002). All three papers consider call centers with only inbound calls.

In this work, we apply GoLo (<u>g</u>lobal <u>o</u>ptimization-<u>l</u>ocal <u>o</u>ptimization), an optimization-via-simulation tool (Pichitlamken and Nelson 2003), to a call center simulation model that we have developed in Pichitlamken et al. (2003). We use the term "optimization via simulation" to refer to the problem of maximizing or minimizing the long-run average performance measure of a computer simulation model, i.e., a stochastic simulation output. Fu (2002) and Swisher et al. (2004) provide comprehensive overview of optimization-via-simulation research and practice.

GoLo consists of a global guidance system, a selection-of-the-best procedure and local improvement (see Section 3 for more details). Like most optimization-via-simulation algorithms, GoLo considers a problem with a single (i.e., scalar) objective function. However, our call center is in a blend environment, where there are multiple objective functions, namely, a long-run quality of service requirement for inbound calls which depends on the number of inbound agents available, and long-run average operational cost which is a function of total number of agents employed and the number of successful outbound calls.

To apply GoLo to our call center problem, we divide the problem into two stages: First, we determine the number of inbound agents needed to satisfy the QoS requirement while having no blend agents. With this number of inbound agents, we then find the number of blend agents such that the average cost (labor cost minus revenue from outbound calls) is minimized.

This paper is organized as follows: Section 2 briefly discusses a call center simulation model under study. Section 3 outlines GoLo. We present our numerical results in Section 4, and we conclude in Section 5.

2 CALL CENTER SIMULATION MODEL

To make this paper self-contained, we briefly describe our call center simulation model. See Pichitlamken et al. (2003) for complete details.

Call center data are generally aggregated as *averages* over some time period (30 minutes in our case); therefore, it is natural to assume that the model parameters (e.g., inbound arrival rates) are constant over each half hour. A single simulation run consists of some number of independent and identically distributed (i.i.d.) "days," each of which has six half hours. We consider six half hours instead of 25 as we did in Pichitlamken et al. (2003) so as to make a simulation run not too long, for the purpose of experimentation.

We model the inbound call arrival process with a Poisson distribution with a stochastic rate. Let X_i be the number of inbound call arrivals in half hour i, with the probability mass function:

$$\Pr\{X_i = x\} \quad = \quad e^{-\Lambda_i}\frac{\Lambda_i^x}{x!}, \text{ where } \Lambda_i = W\lambda_i \qquad (1)$$

(Avramidis, Deslauriers, and L'Ecuyer 2004). The λ_i's are constants, and W is a gamma random variable with $\mathrm{E}[W] = 1$ and density

$$g(\lambda) = \frac{\beta^{-\alpha}}{\Gamma(\alpha)}\lambda^{\alpha-1}e^{-\lambda/\beta}. \qquad (2)$$

The arrival process parameters (the number of arrivals is per half hour) are: $\alpha = 29.7$, $\beta = 0.0336$, $\lambda_1 = 31.5$, $\lambda_2 = 45.9$, $\lambda_3 = 59.5$, $\lambda_4 = 67.8$, $\lambda_5 = 73.1$, and $\lambda_6 = 72.6$ per half hour.

The service times of inbound calls are modelled with a gamma distribution (2) where $\alpha = 0.924$, $\beta = 608.9$, and the service times are in seconds. The service times of outbound calls are modelled with exponential distribution with mean 440.2 seconds, i.e., the service rate $\mu_k = 1/440.2 \text{ sec}^{-1}, k = 1, 2, \ldots, 6$.

All inbound agents are identical, and so are blend agents. There is a single FIFO waiting queue for inbound calls. An inbound customer who is not served immediately hangs up with probability 0.005; otherwise, he joins the queue from which he will abandon if experiencing a waiting time greater than his *patience time*. We model this patience time as an exponential random variable with mean 5.0 seconds.

Our dialer model tries to emulate the real dialer in that the decision on when and how many outbound calls to make is based on the current state of the system. When the service of a customer ends, if the number of idle blend agents is N_2, the dialer makes outbound calls if $N_2 \geq 1$. The number of calls composed is $2N_2$. Each outbound call successfully reaches a customer with probability 0.25. The answering time for an outbound call, defined as the time required by the dialer to either reach the customer or recognize that the attempt is not successful, is exponentially distributed with mean 2 seconds. Call center agents' hourly wage, $2c$, is 10, and revenue per one successful outbound call, r, is 0.5.

3 SIMULATION-VIA-OPTIMIZATION ALGORITHM

The purpose of GoLo is to find a decision variable (often called "solution") $\mathbf{x} = [x_1, x_2, \ldots, x_q]^T$ that maximizes/minimizes some performance measure $\mu(\mathbf{x})$ where $\mathbf{x}$ is subjected to linear deterministic constraints, and $x_i, i = 1, 2, \ldots, q$, is a non-negative integer. These constraints define a feasible region, $\mathbf{\Theta}$. Because $\mathbf{\Theta}$ is finite and discrete, we can conceptually index the solutions $\mathbf{x}$ as follows: $\mathbf{\Theta} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_v\}$. The function $\mu(\mathbf{x})$ is unknown but can be estimated via simulation. The observed performance of $\mathbf{x}_i$ on replication p of the simulation is denoted by Y_{ip}, so that $\mu_i = \mathrm{E}[Y_{ip}]$. In the context of a call center simulation model, a decision variable $\mathbf{x}$ is a staffing policy which is a vector of size 12, $[x_{11}, x_{21}, x_{12}, x_{22}, \ldots, x_{16}, x_{26}]^T$, where x_{1k} is the number of inbound agents and x_{2k} is the number of blend agents in half hour $k, k = 1, 2, \ldots, 6$. A simulation output Y_{ip} is either (a) the sum of discrepancies between the simulated QoS and the target, or (b) the operation cost, when the staffing policy $\mathbf{x}_i$ is used on the p^{th} simulation run (see Equations (4) and (5) below).

GoLo consists of a global guidance system, a selection-of-the-best procedure, and local improvement. The global guidance system ensures the convergence of the search so that, given sufficient time, it reaches and selects one of the optimal solutions. Specifically, we adopt the philosophy of *Nested Partition* (NP) method (Shi and Ólafsson 2000). NP is based on identifying a sequence of "most-promising" subregions of the feasible space. When better solutions are found *inside* the current most-promising region, then the region is partitioned for finer exploration. On the other hand, when better solutions are found *outside* the current most-promising subregion, then NP backtracks to a superregion of it. The idea is to concentrate the computational effort where there appear to be good solutions but not be trapped locally.

A *hill-climbing* (HC) algorithm constitutes our local-improvement scheme. We chose HC because it is intuitively simple: The current solution on hand is compared with some (or all) of its neighboring solutions, and the winner becomes the next solution. This neighborhood selection of the best is repeated until some stopping criterion is satisfied. Using HC or not is an option in GoLo.

Each NP iteration, and each HC step, requires selecting the best solution from among a number of candidates (the sampled solutions for NP, and the neighboring solutions of the current best for HC). *Sequential Selection with Memory* (SSM) provides a highly efficient method for selecting the best—maximum or minimum expected performance—from among a small number of candidate solutions while controlling the chance of an incorrect selection (Pichitlamken and Nelson 2001). SSM is specially designed for use in optimization algorithms that revisit solutions because it exploits whatever data that have already been obtained. Under certain conditions, SSM guarantees to select the best, or a near-best, solution with a user-specified probability, where "near-best" means within a user-specified indifference level, $\delta > 0$.

SSM(`REGION`) is a modified SSM which intends to save simulation effort by terminating SSM when all surviving solutions belong to the *same* subregion. This is useful in an NP step where all we need to do is to identify the subregion that contains the best sampled solution, not necessarily the best solution itself.

3.1 GoLo Procedure

We give a high-level description of GoLo below (see Pichitlamken and Nelson (2003) for complete details).

1. *Initialization:* Set the iteration counter $k = 1$, the current most-promising region R_k to the feasible region $\boldsymbol{\Theta}$, the number of observations on the i^{th} solution $n_i(k) = 0$ for all $i \in \{1, 2, \ldots, v\}$, and the initial estimate of the optimal solution $\mathbf{x}_{\hat{i}^*_{k-1}}$ to a user-provided initial solution.

2. *Search and selection:* Repeat Steps 2a–2f until the simulation effort (i.e., clock time or the number of simulation replications allowed) is exhausted:

   (a) *Partitioning:* If the current most-promising region R_k is not a singleton, then partition R_k into disjoint regions $R_{k1}, R_{k2}, \ldots, R_{k\omega(R_k)}$. Let $M_k = \omega$ be the number of subregions. Then, if $R_k \neq \boldsymbol{\Theta}$, aggregate the surrounding region; let $M_k = M_k + 1$ and $R_{kM_k} = \boldsymbol{\Theta} \setminus R_k$.

   (b) *Sampling:* For each region $R_{k\ell}, \ell = 1, 2, \ldots, M_k$, randomly sample ϑ solutions from $R_{k\ell}$. (If $\mathbf{x}_{\hat{i}^*_{k-1}} \in R_{k\ell}$, include it as one of these ϑ sampled solutions from $R_{k\ell}$.) Aggregate *all* the sampled solutions $\mathbf{x}_i$ into a set through their indices i; let $\mathcal{S}_k$ denote the set of indices of sampled solutions.

   (c) *Selection of the best solution:* Take Δn_{free} observations of Y_{ip} from every solution $\mathbf{x}_i, i \in \mathcal{S}_k$. Use SSM or SSM(`REGION`) to select the best solution over $\mathcal{S}_k$, which we denote as $\hat{\mathbf{x}}^*(\mathcal{S}_k)$. If the simulation effort is exhausted, go to the *Search termination* step.

   (d) *Algorithm Hill Climbing*: If the criterion for using HC is satisfied, perform Algorithm Hill Climbing with $\hat{\mathbf{x}}^*(\mathcal{S}_k)$ as a starting solution. Let $\mathbf{x}_{\hat{i}^*_k}$ be the solution deemed best by HC. If the simulation effort is exhausted, go to the *Search termination* step.

   (e) *Updating the most-promising region:* If $\mathbf{x}_{\hat{i}^*_k} \in R_k$, then its subregion $R_{k\ell}$ that contains $\mathbf{x}_{\hat{i}^*_k}$ becomes the new most-promising region, R_{k+1}; otherwise, the search backtracks to the superregion of R_k, which can be either $\boldsymbol{\Theta}$ or R_{k-1}. Increment $k = k + 1$.

   (f) *Restarting:* Restart at iteration k if $R_{k-k_0+1} = R_{k-k_0+2} = \cdots = R_k$ by letting $R_k = \boldsymbol{\Theta}$; change the partitioning criterion.

3. *Search termination:* The best solution selected by GoLo is the one with the minimum cumulative sample average (if it is in a minimization context); i.e., the selected solution is $\mathbf{x}_{\hat{i}^*}$ where

$$\begin{aligned} \bar{Y}_i(r) &\equiv \sum_{p=1}^{r} Y_{ip}/r \\ \hat{i}^* &\equiv \arg\min_{1 \le i \le v} \left\{ \bar{Y}_i(n_i(k)) : n_i(k) > 0 \right\}. \end{aligned} \quad (3)$$

3.2 APPLYING GOLO TO A CALL CENTER SIMULATION MODEL

Because GoLo considers a problem with a single objective function, but our staffing problem has two, we divide the staffing problem for our call center simulation model into two stages:

1. Assuming that the call center is in inbound mode, determine the number of inbound agents needed to "almost" satisfy the target QoS (QoS*). Let $Q_k(\mathbf{x})$ be QoS of half hour k when a staffing policy $\mathbf{x}$ is employed. We want to find

$$\begin{aligned}\mathbf{x}' &= [x'_{11}, 0, x'_{12}, 0, \ldots, x'_{16}, 0]^T \\ &= \arg\min_{\mathbf{x}} \left\{ \sum_{k=1}^{6} \left| \mathrm{E}\left[Q_k(\mathbf{x})\right] - \mathrm{QoS}' \right| \right\} \quad (4)\end{aligned}$$

for some QoS$' \leq$ QoS*. We consider the following range of x'_{1k}: $\lfloor \lambda_k/\mu_k \rfloor \leq x'_{1k} \leq 2\lfloor \lambda_k/\mu_k \rfloor, \forall k$, where $\lfloor a \rfloor$ is a rounded down to the next largest integer, and λ_k and μ_k are the arrival rate and service rate of half hour k (see Section 2).

2. Let $N(\mathbf{x})$ be the number of successful outbound calls in a day when a staffing policy $\mathbf{x}$ is employed. With $x'_{1k}, k = 1, 2, \ldots, 6$, from (4), determine $x_{2k}, k = 1, 2, \ldots, 6$, such that

$$\begin{aligned}\mathbf{x} &= [x'_{11}, x_{21}, x'_{12}, x_{22}, \ldots, x'_{16}, x_{26}]^T \\ &= \arg\min_{\mathbf{x}} \left[c \sum_{k=1}^{6} (x'_{1k} + x_{2k}) - r\mathrm{E}\left[N(\mathbf{x})\right] \right] (5)\end{aligned}$$

where c is an agent's wage for half an hour and r is the revenue per one successful outbound call. To limit the scope of the search, we limit the range of x_{2k} to $\lceil 0.5x'_{1k} \rceil \leq x_{2k} \leq x'_{1k}, \forall k$, where $\lceil a \rceil$ is a rounded up to the next smallest integer, and x'_{1k} is determined from (4) above.

4 NUMERICAL EXPERIMENTS

In this section, we consider some issues affecting the performance of GoLo and how it performs relative to other optimization schemes on our call center staffing problem which is described in Section 2.

4.1 EXPERIMENTAL SETUP

We will first describe the competing optimization schemes:

NP. NP is our version of the algorithm, but not using SSM or HC. The algorithm takes Δn_{fixed} observations Y_{ip} from $\mathbf{x}_i$ on the first visit, and Δn_{free} additional observations on all other visits. NP selects the best solution over the set $\mathcal{S}_k$, $\hat{\mathbf{x}}^*(\mathcal{S}_k)$, as the one with the largest cumulative sample average, and it uses $\hat{\mathbf{x}}^*(\mathcal{S}_k)$ to determine the new most-promising region.

Random Search (RS) (Andradóttir 1999) RS is a modified hill-climbing algorithm. Let $I_k \in \{1, 2, \ldots, v\}$ denote the index of the current solution on iteration k. RS proceeds as follows:

1. *Initialization:* Set $k = 0$, I_k to the index of a user-provided solution, $\mathbf{x}_0$ (If not given, randomly sample a solution from $\mathbf{\Theta}$).
2. *Search:* Repeat Steps 2a–2c until the simulation effort is exhausted:
   (a) Uniformly sample a candidate solution $\mathbf{x}_{I'_k}$ over $\mathbf{\Theta} \setminus \{I_k\}$, the entire feasible region except for solution I'_k.
   (b) Take $\Delta n_{\text{fixed}} > 0$ observations of $Y_{I_k p}$ and $Y_{I'_k p}$, and compute the sample averages over these observations: $\bar{Y}_{I'_k}(\Delta n_{\text{fixed}})$ and $\bar{Y}_{I_k}(\Delta n_{\text{fixed}})$.
   (c) Update I_k and $C_i(k)$:

$$\begin{aligned} I_{k+1} &= \begin{cases} I'_k, \text{ if } \bar{Y}_{I'_k}(\Delta n_{\text{fixed}}) < \bar{Y}_{I_k}(\Delta n_{\text{fixed}}) \\ I_k, \text{ otherwise} \end{cases} \\ k &= k + 1. \end{aligned}$$

3. *Estimating the optimal solution:* The selected solution is $\mathbf{x}_{\hat{i}^*}$ from (3).

In the experiment, we use SSM(`REGION`) without doing a hill-climbing search. The fictitious target QoS (QoS$'$) in (4) is 0.67, while the actual target QoS* is 0.8. In the backtracking step, the superregion of R_k is R_{k-1}. For the partitioning step, the number of subregions partitioned per R_k (ω) is 2. The number of iterations without progress that triggers restart (k_0) is 6. The minimum number of observations taken from a sampled solution (Δn_{free}) is 1. The number of observations for NP (Δn_{fixed}) is 4. The indifference-zone parameter for SSM (δ) for problem (4) is 0.5. The first-stage number of observations for SSM (n_0) is 4. The confidence level for SSM ($1 - \alpha$) is 0.9.

Because the real performance $\mu(\mathbf{x})$ is unknown, we estimate it from a very long simulation run (1600 call center days). Each algorithm (GoLo, NP and RS) is given the same computational budget (number of observations). We then repeat the entire optimization run m^* times. On the m^{th} run, $1 \leq m \leq m^*$, we have an estimate of the optimal solution $\mathbf{x}_{\hat{i}^*(m)}$ from (3) which has corresponding *true* performance measure $\mu_{\hat{i}^*(m)}$. The results that we present below are the *averaged* values across m^* optimization runs, specifically,

$$\bar{\mu}_{\hat{i}^*}(m^*) = \frac{1}{m^*} \sum_{m=1}^{m^*} \mu_{\hat{i}^*(m)}. \qquad (6)$$

We compare the performance of the selection-of-the-best selection methods by observing $\bar{\mu}_{\hat{i}^*}(m^*)$.

4.2 EXPERIMENTAL RESULTS

In all the cases we have experimented, the averaged QoS exceeds the required QoS, QoS*, because of (4).

First, we consider some factors that affect the performance of GoLo. GoLo uses SSM, a sequential procedure, to do a local selection of the best. SSM needs an indifference-zone parameter, δ, which specifies a practical difference a user deems worth noticing. Within the context of GoLo, if δ is too small (i.e., a user is very demanding in differentiating solutions), GoLo will spend a lot of simulation effort in doing each local selection of the best (i.e., Step 2c in Section 3); as a result, GoLo does not have much time to explore the feasible space. On the other hand, if δ is too large, it implies that a user does not care much about distinguishing between solutions; in other words, it will be as if SSM is not there at all. Table 1 shows that the performance of GoLo depends on the size of δ, and it seems that too large or too small δ harms GoLo's performance (referring to (5), smaller is better for this staffing problem problem). In Table 1, one simulation run consists of 5 call center days, and each optimization run has a quota of 10000 simulation replications.

Table 1: GoLo Performance at Different Indifferent-Zone Parameters δ, $m^* = 30$, and Steady-State Expected Daily Cost $\bar{\mu}_{\hat{i}^*}(m^*)$ is Defined in (6).

δ	$\bar{\mu}_{\hat{i}^*}(m^*)$	Standard Error
1	231.17	1.83
3	215.20	1.81
5	209.36	1.17
10	206.30	1.11
15	206.68	1.01
20	207.32	0.92

Now we consider some conditions in which GoLo performs well compared to other algorithms. We find that the amount of variability plays an important role. Recall that our *single* simulation run consists of some number of i.i.d. call center days, say `NbDays`. The higher the `NbDays`, the less variability in simulation outputs. In Pichitlamken and Nelson (2003), we find that GoLo generally outperforms both NP and RS for problems with high variability, and Table 2 shows that this finding also holds for our call center problem. When `NbDays` is 5, GoLo outperforms NP and RS, but when `NbDays` is higher, NP outperforms others.

We now consider the relative performance of GoLo, NP and RS as simulation effort (i.e., number of simulation runs allowed) increases. From Table 3, we see that RS underperforms either GoLo or NP at all simulation efforts. When the simulation effort is small, it is not clear if GoLo outperforms NP or not, but as the simulation effort increases, GoLo clearly outperforms NP.

5 FUTURE WORKS

From the numerical experiment, we see that GoLo can be used to solve the staffing problem for a call center in a blend environment although the way in which we use GoLo is rather ad-hoc. A better problem formulation than (4) and (5) would be to have an inbound QoS requirement (4) as a probabilistic constraint with the objective of determining a staffing policy that minimizes some operational cost. Stochastic programming (see, for example, Birge and Louveaux 1997) considers an optimization problem with probabilistic constraints. We want to adapt GoLo to include such requirements.

REFERENCES

Andradóttir, S. 1999. Accelerating the convergence of random search methods for discrete stochastic optimization. *ACM Transactions on Modeling and Computer Simulation* 9 (4): 349–380.

Atlason, J., M. Epelman, and S. Henderson. 2002. Call center staffing with simulation and cutting plane methods. *Annals of Operations Research* 127:333–358.

Avramidis, A. N., A. Deslauriers, and P. L'Ecuyer. 2004. Modeling daily arrivals to a telephone call center. *Management Science* 50 (7): 896–908.

Birge, J. R., and F. Louveaux. 1997. *Introduction to stochastic programming.* Springer Series in Operations Research. York, Pennsylvania: Springer.

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

Gans, N., G. Koole, and A. Mandelbaum. 2002. Telephone call centers: Tutorial, review, and research prospects. *Manufacturing and Service Operations Management* 5:79–141.

Ingolfsson, A., and E. Cabral. 2002. Combining integer programming and the randomization method to schedule employees. Technical report, School of Business, University of Alberta, Edmonton, Alberta, Canada. Preprint.

Koole, G., and A. Mandelbaum. 2002. Queueing models of call centers: An introduction. *Annals of Operations*

Table 2: Steady-State Expected Daily Cost $\bar{\mu}_{\hat{i}^*}(m^*)$ at Different `NbDays`, $\delta = 5, m^* = 30$, and 10,000 Simulation Replications per One Optimization Run.

`NbDays`	GoLo	Std. Error	NP	Std. Error	RS	Std. Error
5	209.36	1.17	211.14	1.53	232.61	2.32
8	208.20	1.29	206.93	1.13	226.71	2.06
10	207.07	1.01	204.45	1.05	224.19	1.70

Table 3: Steady-State Expected Daily Cost $\bar{\mu}_{\hat{i}^*}(m^*)$ at Different Number of Simulation Replications Per One Optimization Run, `NbDays` $= 5$, and $\delta = 5$.

Nb of Simulation Replication	GoLo	Std Error	NP	Std Error	RS	Std Error
10000	209.36	1.17	211.14	1.53	232.61	2.32
20000	206.16	1.06	205.25	1.31	224.08	2.32
30000	205.91	1.12	206.95	0.99	228.22	1.68
50000	203.22	1.07	209.80	1.80	225.46	2.96
70000	202.24	1.11	204.76	0.94	223.22	3.91

Research 113:41–59.

Mason, A. J., D. M. Ryan, and D. M. Panton. 1998. Integrated simulation, heuristic and optimisation approaches to staff scheduling. *Operations Research* 46 (2): 161–175.

Pichitlamken, J., A. Deslauriers, P. L'Ecuyer, and A. N. Avramidis. 2003. Modeling and simulation of a telephone call center. In *Proceedings of the 2003 Winter Simulation Conference*, ed. S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, 1805–1812: Institute of Electrical and Electronics Engineers.

Pichitlamken, J., and B. L. Nelson. 2001. Selection-of-the-best procedures for optimization via simulation. In *Proceedings of the 2001 Winter Simulation Conference*, ed. B. A. Peters, J. S. Smith, D. J. Medeiros, and M. W. Rohrer, 401–407. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers. Available on line via ⟨`www.informs-cs.org`⟩.

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

Shi, L., and S. Ólafsson. 2000. Nested partitions method for stochastic optimization. *Methodology and Computing in Applied Probability* 2:271–291.

Swisher, J. R., P. D. Hyden, S. H. Jacobson, and L. Schruben. 2004. A survey of recent advances in discrete input parameter discrete-event simulation optimization. *IIE Transactions* 36 (6): 591–600.

ACKNOWLEDGMENTS

The author would like to thank Sugree Phatanapherom for his help in C++, KU's MAEKA for letting her use their machines and Pierre L'Ecuyer of University of Montreal for providing her with the call center simulation model.

AUTHOR BIOGRAPHY

JUTA PICHITLAMKEN is a Lecturer at the Department of Industrial Engineering, Kasetsart University. She received her Ph.D. from the Department of Industrial Engineering and Management Sciences at Northwestern University in 2002. She was a Postdoctoral Fellow at the University of Montreal during 2002-2003. Her research interests include ranking and selection procedures and simulation optimization. Her e-mail address is `juta.p@ku.ac.th`, and her home page is `http://pirun.ku.ac.th/~fengjtp`.