추계적 생산시스템의 최적 설계를 위한 전자 알고리즘을 이용한 시뮬레이션 최적화 기법 개발*

Simulation Optimization for Optimal Design of Stochastic Manufacturing System Using Genetic Algorithm

이영해**, 유지용***, 정찬석**
Young Hae Lee, Ji Yong Yu, Chan Seok Jeong

Abstract

The stochastic manufacturing system has one or more random variables as inputs that lead to random outputs. Since the outputs are random, they can be considered only as estimates of the true characteristics of the system. These estimates could greatly differ from the corresponding real characteristics for the system. Multiple replications are necessary to get reliable information on the system and output data should be analyzed to get optimal solution. It requires too much computation time practically. In this paper a GA method, named Stochastic Genetic Algorithm(SGA) is proposed and tested to find the optimal solution fast and efficiently by reducing the number of replications.

* 본 연구는 1999년도 한양대 공학기술연구소 지원으로 수행됨
** 한양대학교 산업공학과
*** 마이티정보시스템(주)

1. Introduction

Most approaches for optimization of system attempt to determine the values for variable of system without considering probabilistic components in the system. But many systems have stochastic nature. The system that has this characteristics is called a stochastic system. Because most manufacturing systems are modeled stochastically, there may exit difficulties in optimizing them. The objective functions of the complicated stochastic systems cannot be expressed analytically and it is impossible to be optimized by the classical mathematical programming techniques. For that reason, the stochastic system is analyzed and optimized through simulation.

Computer simulation programs are much more expensive to run than evaluating analytical functions. This makes the efficiency of the optimization algorithm more crucial. When we use simulation methodology to solve stochastic problems, we have to assign certain values to system decision variables, run simulation, analyze the output data, adjust the design variables, and run the simulation again. It is a very complicated and time-consuming task.

Solving the problem of reducing the number of replications is an essential part for this research. Since GA tends to generate the same chromosome more than once, a special database to record information of chromosome is considered in SGA method and the number of replications can be reduced by pruning the same chromosomes which exist in database of past generations. A new method named SGA could find the optimal solution fast and efficiently by reducing the number of replications and the computation time.

2. Literature review

There are several ways stochastic optimization problems can be classified. One of these ways is classification by number of solutions. For the case of the infinite number of solutions, there are the gradient-based methods including perturbation analysis, the likelihood ratio method, frequency domain experimentation and the importance sampling method, etc.. For the case of the finite number of solutions, there are ranking and the selection methods and the multiple comparisons with the best, etc..

2.1 Gradient Based Search Methods

Classical stochastic optimization algorithms are iterative schemes based on gradient estimation. The methods estimate the response function gradient(∇f) to assess the shape of the objective function and employ deterministic mathematical programming techniques.

Proposed in the early 1950s, Robbins-Monro and Kiefer-Wolfowitz are the two most commonly used algorithms for an unconstrained stochastic optimization. These algorithms converge extremely slowly when the objective function is flat and often diverge when the objective function is steep. Additional difficulties include absence of good stopping rules and handling constraints.

More recently, Andradottir[1] proposed a stochastic optimization algorithm that converges under more general assumptions

than these classical algorithms. Leung and Suri[12] reported better results with the Robbins-Monro algorithm when applied in a finite-time single-run optimization algorithm than when applied in a conventional way[3].

In the stochastic counterpart method (also known as sample path optimization) a relatively large sample is generated and the expected value function is approximated by the corresponding average function[4,5]. The average function is then optimized by using a deterministic non-linear programming (LP) method. This allows statistical inference to be incorporated into the optimization algorithm which addresses most of the difficulties in stochastic optimization and increases the efficiency of the method[3].

2.2 Importance Sampling Methods

The importance sampling has been used effectively to achieve significant speed ups in simulations involving rare events, such as failure in a reliable computer system or ATM communication network[6]. The basic idea of importance sampling is to simulate the system under a different probability measure (e.g. with different underlying probability distributions) so as to increase the probability of typical sample paths involving the rare event of interest. For each sample path (observation) during the simulation, the measure being estimated is multiplied by a correction factor to obtain an unbiased estimate of the measure in the original system. The main problem in importance sampling is to come up with an appropriate change of measure for the rare event simulation problem at hand.

2.3 Ranking and Selection Methods

Ranking and selection methods are frequently employed for practical problems, for instance, finding the best combination of parts manufactured on various machines to maximize productivity, or finding the best location for a new facility to minimize cost. In these optimization problems, some knowledge of the relationship among the alternatives is available. These methods have the ability to treat the optimization problem as a multi-criteria decision problem. When the decision involves selecting the best system design, the technique of indifference-zone ranking may be employed. When the decision involves selecting a subset of system designs that contains the best design, the technique of subset selection may be employed. In either case, the decisions are guaranteed to be correct with a pre-specified probability. Many ranking and selection procedures can be found in [7].

2.4 Multiple Comparisons With The Best

If the problem is to select the best of a finite number of system designs, multiple comparison with the best (MCB) is an alternative to ranking and selection. In MCB procedures inference about the relative performance of all alternatives tested is provided. Such inference is critical if the performance measure of interest is not the sole criterion for decision making, e.g., expected throughput of a manufacturing system may be the performance measure of interest but cost of maintaining the system is also important. According to Hsu and Nelson[8], MCB

combines the two most frequently used ranking and selection techniques, namely, indifference zone and subset selection inference. Goldsman and Nelson[9] devised an MCB procedure for steady state simulation experiments based on batching. This MCB procedure can be implemented in a single run of each alternative under consideration, which is important if restarting simulation experiments is unwieldy and/or expensive.

2.5 Genetic Algorithm

Heuristic methods discussed below represent the latest developments in the field of direct search methods (requiring only function values) that are frequently used for simulation optimization. Many of these techniques balance exploration with exploitation thereby resulting in efficient global search strategies. Frequently used heuristic methods are Genetic Algorithm(GA), Simulated Annealing(SA) and Tabu Search(TS)[3].

Genetic algorithms were designed by John Holland[10]. They are randomized search processes based on natural selection, "survival of the fittest". Genetic algorithms are designed to solve problems with large, non-linear, poorly understood search space which traditional optimization methods find difficult[11].

A genetic algorithm assigns each candidate solution (usually encoded as bit strings) in a population with an associated fitness value measuring the candidate's survivability, this process is called evaluation. And the GA evolves this population of individual candidates into a new population using three operators: selection, crossover, and mutation. A GA does its search through an iterative process. The process of one generation involved with selection, crossover and mutation is called one cycle of iteration. Selection probabilistically chooses better candidate solutions for a new generation. Crossover and mutation manipulate candidate solutions to generate new individuals for selection to process again[11]. The general structure of genetic algorithm is described in Figure 1.

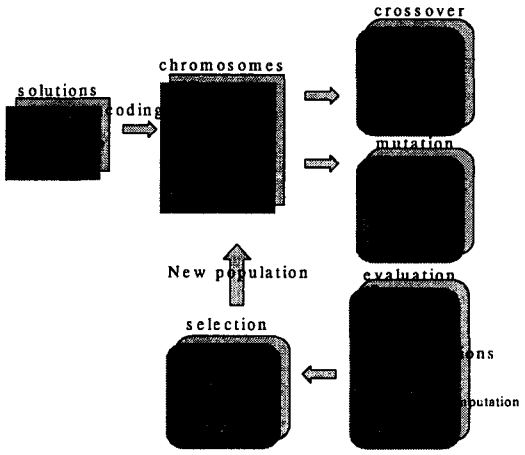

Figure 1. The general structure of genetic algorithm

2.6 General Procedure of Simulation Optimization

Simulation optimization problem can be presented as

$$Min\ E[f(x)], x \in R^n \qquad (1)$$

Here, $f(x)$ is a stochastic response function of simulation model and is presented $f(x) = g(x) + \varepsilon(x)$. $g(x)$ is a deterministic function and $\varepsilon(x)$ is a stochastic function which has $E[\varepsilon(x)] = 0$ for all x. $g(x)$ is called as an objective function in equation (1) and

$f(x)$ is called as a response function. Generally, because $g(x)$ is not obvious , $f(x)$ is analyzed by a simulation model.

Simulation optimization can be defined as a process of finding the best input variable values among the possibilities without explicitly evaluating each possibility. The objective of simulation optimization is to minimize the resources spent while maximizing the information obtained in a simulation experiment. The general procedure of simulation optimization is summarized as follows[3]:

Step 0: Define $f(x)$ and determine alternative.
Step 1: Repeat simulation execution to get $f(x)$ values and collect output data.
Step 2: Analyze output data. If output data is satisfied, go to step 3. If not, go to Step 1.
Step 3: Select the best alternative.

This procedure is illustrated in Figure 2. It is a difficulty of simulation optimization that procedure of simulation optimization have to repeat Step 1 and Step 2. If number of decision variables are many, It require too much computation time to be practical.

3. Stochastic Genetic Algorithm

Stochastic Genetic Algorithm(SGA) presented in this paper is a stochastic optimization method using GA, simulation and statistics method. GA is used to search for new alternative and simulation is used to evaluate alternative and stochastic method is used to analyze output data. The structure of SGA is illustrated in Figure 3.

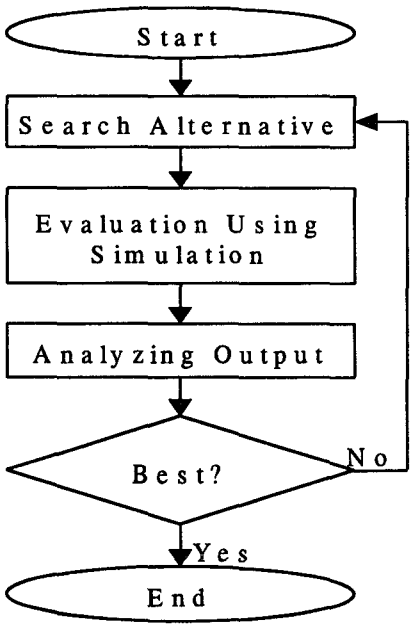

Figure 2. Simulation Optimization Flow

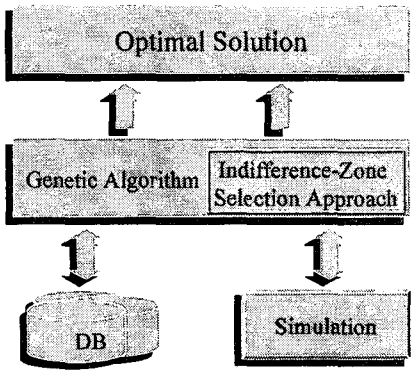

Figure 3. The structure of stochastic GA

The GA is used to get the optimal solution. This algorithm has received considerable attention regarding their potential as an optimization technique for complex problems and has been successfully applied in the area of industrial engineering. The indifference -zone selection approach is used to estimate the final number of replications to be taken in order to meet the indifference-zone probability

requirement.

Many of systems can be analytically formulated and optimized by various techniques of mathematical programming. However, there are so many complicated systems for which the objective function cannot be analytically expressed. Because of the complex nature, the system is evaluated through simulation. The simulation is used to evaluate chromosomes in GA.

We could reduce the time consuming effort on the repetition of simulation experiments with using a database which has recorded information of chromosomes.

3.1 Procedure of SGA

The output of simulation of stochastic system is considered only as estimates of the true characteristics of a model. We needs many replications to get reliable information on a single solution in the stochastic optimization problem. It require too much computation to be practical. It is critical to reduce computation. The new stochastic GA quickly find optimal alternative as to reduce number of replication. The procedure of the new stochastic GA is illustrated in Figure 4.

There are two stage in the procedure of stochastic GA. In the first stage we make

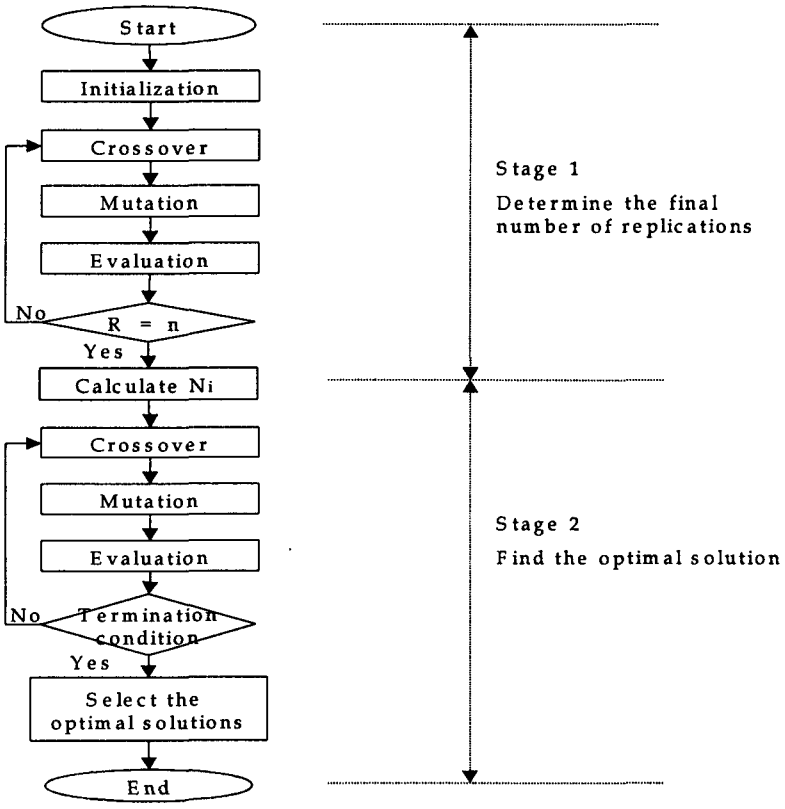

Figure 4. The procedure of the new stochastic GA

initial number of replication of each chromosome, then use the resulting variance estimates to determine how many more replications from each system are necessary for second stage to reach a decision. It is similar to statistical procedure for solving stochastic systems. But it is different in the aspect of gathering sample data and finding optimal solution.

Let n be the initial number of replication. GA procedure is processed until n is equal to the number of accumulated replications R. And the final number of replications N is calculated. N is the number of replications which an alternative is replicated to guarantee statistically. GA procedure is processed again until the terminal condition is satisfied. Finally, we select the best solution.

3.2 Database for Chromosomes

GA tends to generate the same chromosome more than once. And because we want to avoid wasting simulation effort on repeat evaluations, we establish a special database to record information of chromosome. The database store two types of chromosomes. One is the top p% of the current population. Another is specially managed chromosomes that are good alternatives. Figure 5 presents the illustration of the database.

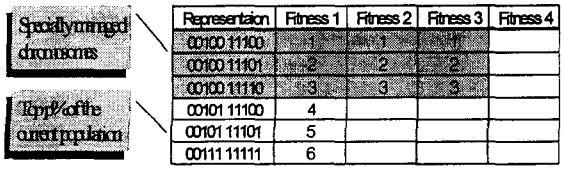

Figure 5. The illustration of the database

3.3 Representation of GA

In order to supply GA for stochastic system, we need to encode decision variables into a chromosome. According to type of decision variable, the type of chromosome gene is selected[12]. How to encode a solution of the problem into a chromosome is a key issue for GA. In Holland's work, encoding is carried out using binary strings. For many GA applications, especially for the problems from industrial engineering world, the simple GA was difficult to apply directly because the binary string is not a natural coding. During the past 10 years, various non-string encoding techniques have been created for particular problems-for example, real number coding for constrained optimization problems and integer coding for combinatorial optimization problems. Choosing an appropriate representation of candidate solutions to the problem at hand is the foundation for applying genetic algorithms to solve real world problems, which conditions all the subsequent steps of genetic algorithms. For any application case. it is necessary to perform analysis carefully to ensure an appropriate representation of solutions together with meaningful and problem-specific genetic operators[12].

One of the basic features of genetic algorithms is that they work on coding space and solution space alternatively: genetic operations work on coding space(chromosomes), while evaluation and selection work on solution space. Natural selection is the link between chromosomes and the performance of their decoded solutions. For the non-string coding approach, three critical issues which emerged

concerning with the encoding and decoding between chromosomes and solutions are the feasibility of a chromosome, the legality of a chromosome and the uniqueness of mapping.

Feasibility refers to the phenomenon of whether a solution decoded from a chromosome lies in the feasible region of a given problem. Legality refers to the phenomenon of whether a chromosome represents a solution to a given problem. The infeasibility of chromosomes originates from the nature of the constrained optimization problem. All methods, conventional ones or genetic algorithms, must handle the constraints. For many optimization problems, the feasible region can be represented as a system of equalities or inequalities (linear or nonlinear). The illegality of chromosomes originates from the nature of encoding techniques. For many combinatorial optimization problems, problem-specific encodings are used and such encodings usually yield to illegal offspring by a simple one-cut-point crossover operation[12]. We need to consider these problems carefully when designing, a new non-binary-string coding so as to build an effective genetic algorithm.

3.4 Initialization and Parameters

The intial populations are obtained by random sampling from the solution space. The user-defined parameter δ is the smallest difference in expected performance that is practically significant to the user. Parameter p is a percent of population. We specially manage the top p percent of population to get optimal alternative and to reduce the number of replications. Parameter n is the initial number of replications. We have to define the value of δ, overall confidence level($1-\alpha$), the value of n and the value of p before running stochastic GA algorithm.

3.5 Final Number of Replications

The final number of replications to be taken in order to meet the indifference-zone probability requirement is obtained by Indifference Zone Selection Approach[13]. When the replication numbers of chromosomes included in the top p percent of population is n, the number of replication is established. The procedure to determine the final number of replication for solution i N_i is as follows:

Step 1. Let k be the number of chromosome included in top $p\%$ of the population. Find h_α for n, k and α (see the tables in Bechhofer *et al.*[14]).

Step 2. Let X_{ij} be the output from the jth replication of solution i. Take sample X_{i1}, X_{i2}, X_{i3},...X_{in} from each k chromosome of the population simulated independently. **Step 3.** Calculate the sample means and marginal sample variances such as

$$\overline{X}_i = \frac{1}{n}\sum_{j=1}^{n} X_{ij}, \quad S_i^2 = \sum_{j=1}^{n}(X_{ij} - \overline{X}_i)^2/(n-1)$$

Step 4. Compute the final number of replications for solution i such as

$$N_i = \mathrm{MAX}\ \{n, [\left(\frac{h_\alpha S_i}{\delta}\right)^2]\}$$

* [ ] : integer round-up function.

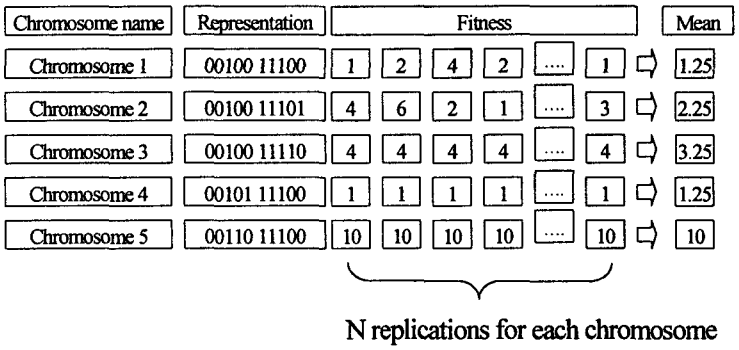

Figure 6 The illustration of the traditional evaluation

3.6 Evaluation

We have to rank chromosomes to select some of them that are copied for the next generation. The evaluation of chromosome is considered only as estimates of the true characteristics of a chromosome. In a stochastic GA it is not possible to conclusively rank any population of solution without multiple replications.

The traditional method is as follows. First, calculate the replications number N that alternative is replicated to guarantee statistically valid. Second, evaluate the objective function $f(x)$ N times a chromosome. Third, calculate its mean. This method spends much time. The Figure 6 illustrates the traditional evaluation method.

Genetic Algorithm is an iterative process. we use this characteristics to reduce the number of replications. Using information of past generations, the number of replications can be reduced. The basic idea to reduce the number of replications is that the chromosomes included in the top p% of population are stored and the fitness values of the chromosomes are reused when those are required.

The process of evaluating the fitness of a chromosome consists of the following steps (refer to Figure 7):

Step 1. Convert the chromosomes gene-type to its pheno-type.

Step 2. Evaluate the objective function .

Step 3. Convert the value of objective function into fitness.

Step 4. If the same chromosome is in the database, calculate the mean of fitness with fitness of stored chromosome. Substitute the mean of fitness for fitness obtained in Step 3.

3.7 Ranking

The Chromosomes are ranked after evaluating the fitness. The proposed procedure of the ranking is as follows:

Step 1. Sort the chromosomes.

Step 2. Let M is the maximum number of replications among chromosomes included in top p% of population and let Ri is the replication numbers of chromosome I. Find the maximum number M = max Ri. If all R for i = $1, .. , k$ are equal, the procedure is finished.

Step 3. Take $(M - Ri)$ additional evaluations

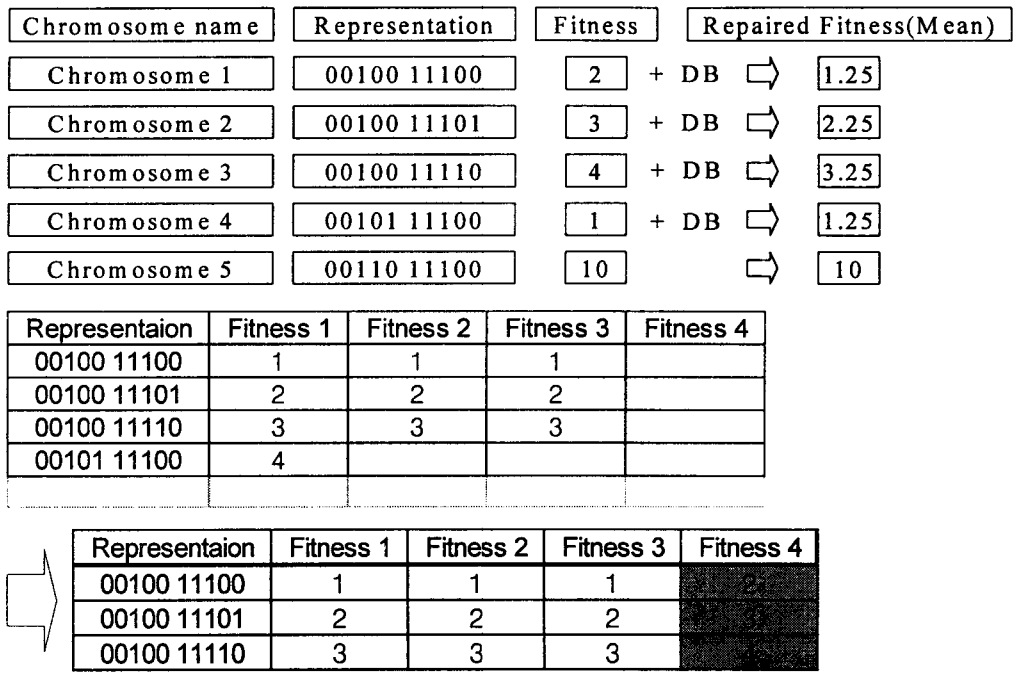

Figure 7. The illustration of the evaluation

from chromosome i, for $i = 1, .. , k$ and go Step 1.

This method can reduce the calculation time. But it may not be accurate. Thus, we use the elitist policy in selection to supplement the weakness.

3.8 Selection

Chromosomes from the current population are selected with a given probability and copies from these individuals are created to constitute the mating pool. Selection of individuals is based on their fitness relative to the current population, i.e., the strongest individual will have a higher probability of being in the mating pool. Fitness is determined by the objective function that we wish to optimize[12].

Roulette wheel selection is similar to spinning a roulette wheel in which the size of the roulette wheel slot for each individual in the population is allocated in proportion to its fitness. Thus the individuals with higher fitness have better chance of being selected relative to the less fit individuals. However, roulette wheel selection does not guarantee that high fitness candidate solutions propagate to the next generation. This selection method is less effective because of its high stochastic error. So we use the elitist selection method to support roulette wheel selection. The elitist selection method guarantees a certain number of high fitness individuals will propagate to the next generation, and decreases stochastic

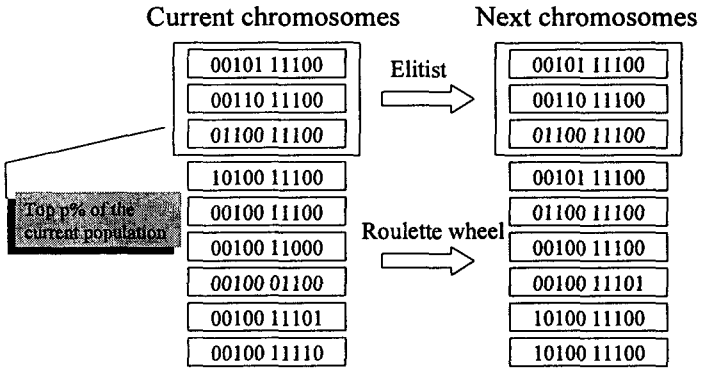

Figure 8. The illustration of the selection

errors. Elitist selection ensures that the best chromosome is passed onto the new generation if it is not selected through the process of roulette wheel selection (refer to Figure 8).

4. Experiments

4.1 Known Function Problem

The following discrete minimization problem is used as a test battery[15].

Objective Function :

$$\text{Min}\frac{1}{18}\{(x_1+10x_2)^2+5(x_3-x_4)^2+(x_2-2x_3)^2+10(x_1-x_4)^2\}+\varepsilon_1$$

Subject to :

$$0 \le x_i \le 31,\ i=1,2,3,4.$$

$$\varepsilon_1 \sim N(\ 0,\ 10\ )$$

ε_1 is a normal distributed random variable with zero mean and a standard deviation of 10. This random variable is added to the objective function to give stochastic characteristics. The parameters for SGA are as follows:

- The special managed chromosomes : Top 10 % of the population
- The value of Indifference Zone δ : 1
- Confidence level($1-\alpha$) : 95%
- The initial number of replications : 40
- Crossover rate : 0.8 %
- Mutation rate : 0.1 %
- Population Size : 100

The Table 1 shows the result of the experiment. We obtained the optimal solution point (0, 0, 0, 0) and the expected optimal value was 0.32. The final number of replications is 237. The results according to various standard deviations are shown in Table 2.

Table 1. The obtained result for the given problem

Proposed Method						
Rank	X1	X2	X3	X4	Replication/Time	Fitness(Mean)
1	0	0	0	0	237/25.76	0.327702616
2	3	0	1	2	237/25.76	2.757622025
3	2	0	0	3	237/25.76	3.305363645
4	1	1	0	1	237/25.76	6.108468007
5	4	0	1	5	237/25.76	6.737129675
6	6	0	1	5	237/25.76	7.963941507
7	4	0	0	5	237/25.76	8.474482719
8	7	0	1	7	237/25.76	14.2071865

Table 2. The results according to various standard deviations

Standard deviation	1	10	50	100	1000
optimal solution	0,0,0,0	1,0,1,1	0,0,1,1	1,0,0,1	1,0,1,2
expected optimal value	0.18	1.27	1.35	0.19	1.73

Computer with Pentium celeron 333 (64M RAM) was used for the experiment and the time consumption was 25.76 second, which is better than the traditional GA method (32.56 second). Even though the traditional GA method computed similar fitness values, it required longer computation time because of the larger number of replications for simulation.

4.2 Manufacturing System

We now apply SGA to a manufacturing system[13]. A company is going to build a new manufacturing facility consisting of an input/output (or receiving/shipping) station and five work stations as shown in Figure 9. The machines in a particular station are identical, but the machines in different stations are dissimilar. One of the goals is to determine the number of machines needed in each work station. It has been decided that the distances between the six stations will be as shown in Table 3 (the input/output station is numbered 6).

Table 3. Distance between stations

Station	1	2	3	4	5	6
1	0	150	213	336	300	150
2	150	0	150	300	336	213
3	213	150	0	150	213	150
4	336	300	150	0	150	213
5	300	336	213	150	0	150
6	150	213	150	213	150	0

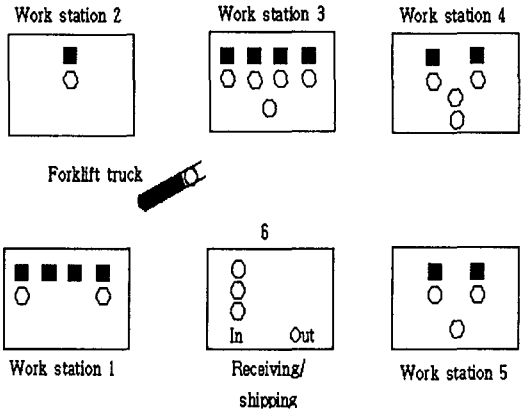

Figure 9. Layout of the given manufacturing system

Assume that jobs arrive at the input/output station with inter arrival times that are independent exponential random variables with a mean of 1/15 hour. There are three types of jobs, and jobs are of types 1, 2, and 3, with respective probabilities 0.3, 0.5, and 0.2. Each job begins at the input/output station, travels to the work stations on its routing, and then leaves the system at the input/output station. The routings for the different job types are given in Table 4.

Table 4. Routing for the three job types

Job type	Work stations in routing
1	3, 1, 2, 5
2	4, 1, 3
3	2, 5, 1, 4, 3

A job must be moved from one station to another by a forklift truck, which moves at a constant speed of 5 feet per second. Another goal of the simulation study is to determine the number of forklift trucks required. When a forklift becomes available, it processes requests by jobs in increasing order of the distance between the forklift and the requesting job (i.e., the rule is shortest distance first). When the forklift finishes moving a job to a work station, it remains at that station if there are no pending job requests.

Table 5 Mean service time for each job type and each operation

Job type	Mean service time for successive operation(hours)
1	0.25, 0.15, 0.10, 0.30
2	0.15, 0.20, 0.30
3	0.15, 0.10, 0.35, 0.25, 0.20

If a job is brought to a particular work station and all machines there are already busy or blocked, the job joins a single FIFO queue at that station. The time to perform an operation at a particular machine is a gamma random variable with a shape parameter of 2, whose mean depends on the job type and the work station to which the machine belongs. The mean service time for each job type and each operation is given in Table 5. Thus, the mean total service time averaged over all jobs is 0.77 hour.

We simulate the proposed manufacturing facility to determine how many machines are needed at each work station and how many forklift trucks are needed to achieve the expected throughput of 120 jobs per 8-hour day, which is the maximum possible. Among those system designs that can achieve the desired throughput, the best system design will be chosen on the basis of measures of performance such as average time in system, maximum input queue sizes, proportion of time each work station is busy, proportion of time transporters are moving, etc.

For each proposed system design, a simulation of length 320 hours was made (40 eight-hour days), with the first 64 hours of each replication being a warmup period. We also used the method of common random numbers to simulate the various system designs. This guarantees that a particular job will arrive at the same point in time, be of the same job type, and have the same sequence of service-time values for all system designs on a particular replication. Job characteristics will, of course, be different on different replications.

The objective is to minimize system cost per unit time. The cost includes machine cost, queue cost and throughput cost. Machine cost and queue cost are setup cost, and throughput cost is operation cost by the unit time. So objective function can't contain machine cost and queue cost directly.

Machine cost and queue cost are modified to cost of unit time. C_i is the cost of ith machine per unit time. C_i is calculated by machine setup cost per unit time added to

maintenance cost per unit time. The machine cost is the sum of all machine costs the as $\sum_{all\ i} n_i C_i$. C_q is the cost of ith queue per unit time. C_q is calculated as similar as C_i. The total queue cost is $n_q C_q$

The system is required to work exactly objective throughput. If the observed throughput is under or over target throughput, penalty cost ($C_{penalty}$) is added to system cost.

Objective function is as follows:

$$\min \sum_{all\ i} n_i C_i + n_q C_q + C_{penalty} \left| n_{obj} - n_{obs} \right|$$

where

n_i: the number of ith machine type

C_i: the cost of ith machine per unit time

n_q : the total number of queue (stochastic)

C_q: the cost of basic structure of queue (per unit time)

$C_{penalty}$: the cost of penalty for a underachieving or over-achieving the target throughput

n_{obj} : the number of objective throughput

n_{obs} : the number of observed throughput (stochastic)

The detailed objective function and constrains are given as follows:

$$Min \quad 10000*n_1+11000*n_2+23000*n_3+22000*n_4 +17000*n_5+6000*n_q+2000*|3840-n_{obs}|$$

Subject to :

$$1 \le n_i \le 7, \quad i=1,2,3,4,5.$$

Table 6 shows the result of applying the proposed method to the given manufacturing system. The optimal solution is obtained as (6, 3, 6, 3, 3) and the expected optimal value is 525,800.

Table 6. The obtained result for the given manufacturing system

Rank	n_1	n_2	n_3	n_4	n_5	Fitness (Mean)
1	6	3	6	3	3	525800
2	6	4	7	2	4	622500
3	4	5	7	4	7	623300
4	6	6	7	2	4	683800

5. Conclusion

Using simulation optimization methodology to solve stochastic optimization problem is time-consuming task with large number of replications of simulation. Multiple replications are necessary to get reliable information on the system and output data should be analyzed to get optimal solution. It requires too much computation time practically.

In this paper a GA method, named Stochastic Genetic Algorithm (SGA) was developed and tested to find the optimal solution fast and efficiently by reducing the number of replications. Since GA method tends to generate the same chromosome more than once, a special database to record information of chromosome is considered in the developed method and the computation time can be reduced by pruning the same chromosomes, which exist in the generation. The results with

a known function and a manufacturing simulation model shows that the developed method works well for the stochastic optimization problem.

Reference

[1] Andradottir, S., "A New Algorithm for Stochastic Optimization", *Proceeding of the 1990 Winter Simulation Conference*(1990), pp. 364-366.

[2] Leung, Y.T. and Suri, R., "Finite-Time Behavior of Two Simulation Optimization Algorithm", *Proceeding of the 1990 Winter Simulation Conference*(1990), pp. 372-376.

[3] Carson, Y. and Anu, M. "Simulation Optimization: Methods and Applications", *Proceedings of the 1997 Winter Simulation Conference* (1997), pp. 118-126.

[4] Shapiro, A., "Simulation Based Optimization", *Proceeding of the 1996 Winter Simulation Conference*(1996), pp. 332-336.

[5] Gurken, G., Ozge, A.Y. and Robinson, S., "Sample Path Optimization in Simulation", *Proceeding of the 1994 Winter Simulation Conference*(1994), pp. 247-254.

[6] Shahabuddin, P., "Rare Event Simulation in Stochastic Models", *Proceeding of the 1995 Winter Simulation Conference*(1995), pp. 178-185.

[7] Gupta, S.S. and Panchapakesan, S., Multiple-Decision Procedure - Theory and Methodology of Selecting and Ranking Populations, John Wiley, New York, 1979.

[8] Hsu, J.C. and Nelson, B.L., "Optimization over a finite Number of System Designs with One-Stage sampling and Multiple Comparisons with the Best", *Proceeding of the 1988 Winter Simulation Conference* (1988), pp. 451-457.

[9] Goldsman, D. and Nelson, B.L., "Statistical Screening, Selection, and Multiple Computer Simulation", *Proceedings of the 1998 Winter Simulation Conference*(1998), pp. 159-166.

[10] Holland, J., Adaptation In Natural and Artificial Systems, The University of Michigan Press, Ann Arbour, 1975.

[11] Xiaohua, L. and Sushil, L.J., Combining Genetic Algorithm and Case-based Reasoning for Structure, University of Nevada, Reno, 1996.

[12] Gen, M. and Cheng, R., Genetic Algorithms and Engineering Design, Wiley, New York, 1997.

[13] Law, A.M. and Kelton, W.D., Simulation Modeling & Analysis, McGrew-Hill, New York, 1991.

[14] Bechhofer, R.E., Santner, T.J. and Goldsman, D., Design and Analysis of Experiments for Statistical Selection, Screening and Multiple Comparisons, John Wiley and Sons, New York, 1995.

[15] Lee, Y.H. and Azadivar, F., "Optimization of Discrete Variable Stochastic Systems By Computer Simulation", *Mathematics and Computers in Simulation*, Vol.30 (1998) , pp. 331-345.

저자소개

이영해

1977 고려대학교 산업공학, 학사
1983 Univ. of Illinois, 산업시스템공학 , 석사
1986 Univ. of Illinois, 산업공학 및 경영과학, 박사
1986-현재 한양대학교 산업공학과 교수
1995-현재 한국시뮬레이션학회 부회장
2000. 6-현재 한국SCM연구회 회장
경력: 한국시뮬레이션학회, 대한산업공학회, 한국경영과학회, 대한설비관리학회 이사, Purdue Univ., 오사카대학 방문교수, 대우중공업(주)
관심분야: Supply Chain Management, Web-based Simulation, Simulation Output Analysis, Simulation Optimization

유지용

1995 한양대학교 산업공학, 학사
1999 한양대학교 산업공학, 석사
1999-현재 마이티 정보 시스템 주식회사
관심분야: Simulation Optimization

정찬석

1988년 육군사관학교 전산학, 학사
1997년 미 공군대학원(AFIT), 운영분석, 석사
1999-현재 한양대학교 산업공학과 박사과정
관심분야: Supply Chain Management, Simulation Optimization