Journal of the Korean Nuclear Society
Volume 29, Number 1, pp. 68~77, February 1997

Determination of the Weighting Parameters of the LQR System for Nuclear Reactor Power Control Using the Stochastic Searching Methods

Yoon Joon Lee and Kyung Ho Cho
Cheju National University
Department of Nuclear and Energy Engineering
1 Ara-dong, Cheju 690-756, Korea

(Received July 24, 1996)

Abstract

The reactor power control system is described in the fashion of the order increased LQR system. To obtain the optimal state feedback gain vectors, the weighting matrix of the performance function should be determined. Since the conventional method has some limitations, stochastic searching methods are investigated to optimize the LQR weighting matrix. Using the modified genetic algorithm combined with the simulated annealing, a new optimizing tool named the hybrid MGA-SA is developed to determine the weighting parameters of the LQR system. This optimizing tool provides a more systematic approach in designing the LQR system. Since it can be easily incorporated with any forms of the cost function, it also provides the great flexibility in the optimization problems.

1. Introduction

The control design techniques have developed significantly over the last decade. Although the PID control has been used and proved to be powerful in various fields of the applications, new control techniques are widespread at present with the computer aided control design. In the nuclear field, the plant control is one of the important issues at the present stage of the digitalization. One of the control techniques that could replace the present PID is the linear quadratic regulator (LQR) method. This method is an important subset of the Wiener-Hopf-Kalman optimal control. The most attractive feature of the LQR method is that it can provide the systematic environments for the control design. So the problem of system design could be boiled down to the problem of the determining the proper weighting matrices. In reality, however, it is not easy to determine the suitable weighting matrices, particularly when the system order is high. Since there is no direct method to control the frequency response characteristics during the design process, the LQR weightings are usually determined through the so-called divide-and-conquer approach. Thus, the result is highly dependent upon the designer's knowledge. As a result, we have experienced the tedious and time-consuming calculations [1,2]. This experience motivated us to develop a fast and reliable computing tool that could replace the tremendous jobs.

To determine the proper control parameters for the design of the nuclear power plant, we considered in this work two kinds of the stochastic searching methods as the candidate solution methodologies -

one is the Simulated Annealing (SA [3-5]) and the other is the Genetic Algorithm (GA [9-14]). Also, a modified version of the GA is described. Finally, a hybrid approach is proposed by combining the modified GA with the SA.

2. Optimization Goal and Stochastic Searching Methods

The overall configuration of the digital reactor power control system is illustrated in Fig. 1. All the details regarding the physical backgrounds and the system modeling were described in Refs. [1] and [2].

To make the output follow the input command signal, the integrated error signal is augmented as a new variable, and the resulting system becomes an order increased regulating system (OIRS). Since this scheme still maintains the LQR frame, its margins are guaranteed and the tracking properties are found to be much better than those of the ordinary feedback system. The system equation is described as

$$\xi(k+1) = \Psi\xi(k) + \Lambda r(k), \quad y(k+1) = H\xi(k) \qquad (1)$$

where ξ is the state variable vector and Ψ, Λ, H are system matrices. The optimal performance function of the order increased LQR is

$$J_0 = \frac{1}{2}\sum_{k=0}^{\infty}(x(k)^T Q x(k) + u(k)^T R\ u(k)) \qquad (2)$$

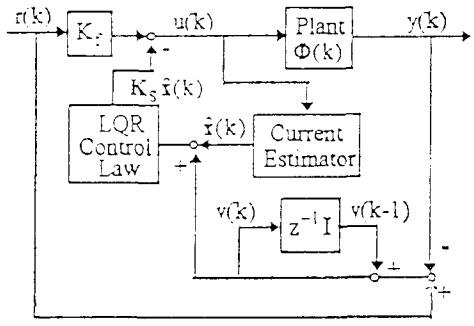

Fig. 1. Overall Configuration of the Digital Reactor Power Control System

where $X=(x(k)\ v(k))$, and Q, R are the weighting matrices, respectively.

The design purpose of the order increased LQR system is to determine the state feedback gain K, which, in turn, is determined by the weighting matrices of Q and R. The order of weighting matrix Q is 6 having 36 elements at maximum. Since it is almost impossible to determine all these parameters in a conventional method, Q is commonly assumed to be a diagonal matrix. In the previous work [2], it was assumed that $Q=[Q_{ij}]$, where $Q_{ij}=0$ at $i\neq j$, $Q_{ii}=q$ for $i=1, 2, \cdots, 5$ and $Q_{66}=r$. Thus, only two variables, q and r, are to be determined. After numerous simulations, the 'best' values of q and r are found to be 3. e^{-6} and 0.003, respectively. The corresponding responses of the output and the control input are shown in Figs 7 and 8, respectively. However, this approach still has some problems. First of all, it is hard to assure that the design result is the best one.

Another better solution may exist. Secondly, when the number of unknown parameters becomes large, it is very difficult to fix them optimally. These problems motivated us to investigate the stochastic searching methods.

Since our goal was to determine the optimal weighting matrix Q in a stochastic way, we formulated the problem somewhat simply as follows:

Find $$X = [x_1, x_2, \ldots, x_6] \qquad (3)$$

to minimize $$Cost(X) = \sum_{k=0}^{\infty}|y(k)-y_0| + \sum_{k=0}^{\infty}|u(k)| \qquad (4)$$

subject to $$1.00e^{-6} < x_i < 1.00e^{-1}, i = 1,2,\ldots,6 \qquad (5)$$

where,

- X : a candidate solution vector
- x_i : diagonal elements of the weighting matrix Q
- $Cost(X)$: cost function for the candidate solution vector X
- $y(k)$: normalized output of the system at time step k
- y_0 : normalized target value of the system

output

$u(k)$: normalized velocity of the control rod at time step k.

The first term of the cost function described in the Eq. (4) indicates how well the system response y of the candidate solution X follows the command input. Also, the second term of the Eq. (4) means the total traveling length of the control rod. More elaborated cost function may be available instead of the Eq. (4).

As can be seen in the Eq. (4) above, it is very difficult to describe the cost function $Cost(X)$ and its Jacobian and Hessian precisely as a function of X. Thus, no gradient-based searching methods, such as the Newton Method and its variances [15], are suitable searching tools for our problem. This fact implies that a stochastic searching technique like the SA or GA is inevitable.

The stochastic searching methods usually have the abilities of hill climbing and backtracking [4]. The former provides the ability of escaping from a valley (local minima). The latter allows one to try a new search from visited solutions. To determine the proper control parameters for the design of the nuclear reactor power LQR control system, we considered two kinds of the stochastic searching methods in this work-one is the SA and the other is the GA. The former is a one-point searching scheme and is suitable for the final local search. On the other hand, the latter is a multi-point searching one and thus suitable for the global search in the earlier searching stage [13].

3. Determination of the Control Parameters by the SA

The SA has been applied to several kinds of optimization problems, and proved to be powerful for the combinatorial optimization, especially when the objective function cannot be expressed analytically [6-8]. Fig. 2 shows the pseudo-code of the optimal parameter searching using the SA, where X and

```
X = Initial solution X0;
T = Initial temperature T0;
while(stopping criterion is not satisfied) do
    while (not yet in equilibrium) do
        X': Random neighbor of X;
        ΔC : Cost(X') − Cost(X);
        Prob = Min(1, exp(−ΔC/T));
        if rand[0, 1] < Prob then X: = X';
    endwhile
    Update T;
endwhile
Output best solution;
```

Fig. 2. Pseudo-code of the Optimal Parameter Searching by the SA

$Cost(X)$ are the ones defined by the Eqs. (3) and (4) above. The main components of the algorithm are the cooling schedule and the neighborhood generation mechanism.

Once a current solution X has been obtained, a new solution X' called *neighbor* is generated from the current solution X through a proper neighborhood generation mechanism. Whenever the neighbor X' decreases the cost (*i. e.*, $\Delta C = Cost(X') - Cost(X) < 0$), it is accepted as a new current solution and the procedure continues to search for a new neighbor from this solution. Even though the neighbor increases the cost (*i. e.*, $\Delta C > 0$) to a certain degree, it is also possible that the neighbor can be accepted as a new solution if the probability of the acceptance, $Prob = exp(-\Delta C/T)$, is greater than a random number chosen between 0 and 1 at that instance. Here, ΔC is the cost change and T is the annealing parameter equivalent to the temperature in the physical annealing process. The stochastic search as above is the hill climbing ability of the SA [4].

As it can be seen in Fig. 2, it is necessary to determine 1) how to delimit the initial temperature and update it, 2) when a state is in equilibrium at an arbitrary temperature, 3) how to generate the neighbors, and 4) when to stop the whole procedure. Several studies have been reported in relation to these issues

[10,14,15]. In this work, a neighbor $X' = [X'_1, X'_2, \cdots, X'_6]$ is generated from the current solution $X = [X_1, X_2, \cdots, X_6]$ as follows :

$$x_i' = x_i + \delta x_i \cdot rand[-1,1], \quad i = 1,2,\ldots,6 \qquad (6)$$

where δx_i is the pre-specified upper bound of the perturbation for the parameter x_i, and $rand[a,b]$ means a number chosen randomly between a and b.

For the cooling schedule, we used the so-called banner schedule. In other words, the temperature T^* at the k-th annealing step is updated in a simple way, i. e., $T^{*-1} = \alpha^* T^*$ with $\alpha = 0.8 \sim 0.95$. It should be noticed that the small value of α implies that the system will be cooled rather rapidly and may be frozen to local optima. The whole procedure stops either when the annealing parameter T^* becomes less than a predefined value T_{min}, or when none of the neighbors tried at T^* has been accepted as a new solution. Further details of the SA can be found at the related works [3-5].

4. Determination of the Control Parameters by the MGA

The GA which was first proposed by J. Holland in 1975 and revived by D. E. Goldberg [9] in the mid 1980's, has proven to be a useful tool in a variety of search and optimization problems over the last years [12-14,16,17]. The GA is based on the survival-of-the-fittest principle in nature. According to evolutionary theories, only the fittest elements in a population are more likely to survive and generate their offspring, thus transmitting their biological heredity to new generations. In computing terms, the GA maps a problem onto a set of (typically binary) strings to represent a candidate solution called an individual. Each solution is associated with a *fitness* value to measure how good it is. The GA then manipulates the most promising strings to search improved solutions. By using three major operators of reproduction, crossover and mutation, it searches the optimal design parameters of the given problem. Since the GA does not depend on the coupling between the parameters, it provides more flexibility in dealing with the concerned system. Fig. 3 shows a typical pseudo-code of a simple Genetic Algorithm (SGA).

```
Initialize parameters of the GA ;
Randomly generate initial population as old_population ;
generation := 0 ;
while (termination conditions not satisfied) do
  Clear new_population ;
  Evaluate fitness of each individual in old_population ;
  Update best_individual ;
  while (no_of_individual < population_size) do
    Select two parents from old_population based on their fitness ;
    Perform the crossover upon parents to produce two offsprings ;
    Mutate each offspring based on mutation_rate ;
    Place offsprings to new_population ;
    no_of_individual := no_of_individual + 2 ;
  endwhile
  Replace old_population with new_population ;
  generation := generation + 1 ;
endwhile
Return best_individual ;
```

Fig. 3. Typical Pseudo-Code of a SGA

Throughout the repeated generation changes in the GA, the average fitness of the candidate solutions gradually increases. That is, the attributes of the candidate solutions will be improved toward an unknown optimal one, and some of these solutions will converge to the globally best one from which it is almost impossible to get further improvements. Ultimately, the GA differs from the traditional searching techniques in several aspects [11-14] :

- The GA is the direct searching method independent of the coupling of the design parameters and does not need any intensive system information such as derivatives.
- The GA makes use of a stochastic search and not a deterministic one. Owing to this feature, it can escape from the local traps, but may tend to wander around the true solution.
- The GA operates on several solutions simultaneously, gathering information from current search points to direct subsequent searches.

Modified Genetic Algorithm (MGA)

To improve the performance of the SGA, we modified the SGA by applying following schemes which are the major improvements from the SGA:

- linear fitness scaling
- elite policy
- variable number of crossover site
- modified crossover scheme
- ancestor-pool
- periodic re-initialization of the population

Representation of an Individual

The SGA basically works on the binary strings, i. e., the parameter domain is to be discretized to a certain resolution. The feasible searching domain varies depending upon not only the parameter range but also the discretizing resolution of the parameter. The more finely discretized the searching domain is, the larger the feasible searching domain becomes. Contrary to the conventional representation of an individual in the SGA like the form of $individual = [x_1, x_2, \cdots, x_n]$ where x_i is the design parameters, we devised an exponential-wise representation as follow:

$$individual = [a_1, b_1, a_2, b_2, \cdots, a_n, b_n]$$
$$\text{with } x_i = a_i \cdot e^{b_i},\ i = 1,2, \cdots, n \qquad (7)$$

This kind of the individual representation is very useful either when there exists little information about the parameter range or when it ranges widely. It also helps the GA search the dominant order of each parameter, maintaining the required number of significant digits without any drastic increase of the feasible searching domain.

Modified Crossover Scheme

We used a modified crossover scheme instead of the well-known bit-wise crossover to avoid the hamming-cliff effects. Thus, no encode and decode processes are required. If i is equal to the crossover site s selected randomly, the child's parameter $C1X_i$, $C2X_i$ are calculated by a linear inter or extrapolation scheme as shown in Fig. 4. Before the crossover site, i. e., $1 \leqq = i < s$, the parameter $P1X_i$ of parent P1 is copied to the parameter $C1X_i$ of children C1. Also, the parameter $P2X_i$ of parent P2 is copied to the parameter $C2X_i$ of children C2. After the crossover site, i. e., $s < i \leqslant n$, the parameter $P1X_i$ of parent P1 is copied to the parameter $C2X_i$ of children C2. As in the same manner, the parameter $P2X_i$ of parent P2 is copied to the parameter $C1X_i$ of children C1.

Ancestor-Pool and Periodic Re-initialization of the Population

In the GA, lots of candidate solutions have to be tested. Also, the fitness calculation usually takes the major portion of the total execution time. In this work, therefore, we attempted to save the execution time by skipping the fitness calculation for a candidate solution if their genetic properties (the values of the design parameters) coincide with those of a previously visited solution (ancestor). By introducing a pre-specified size of storage pool named hereafter an

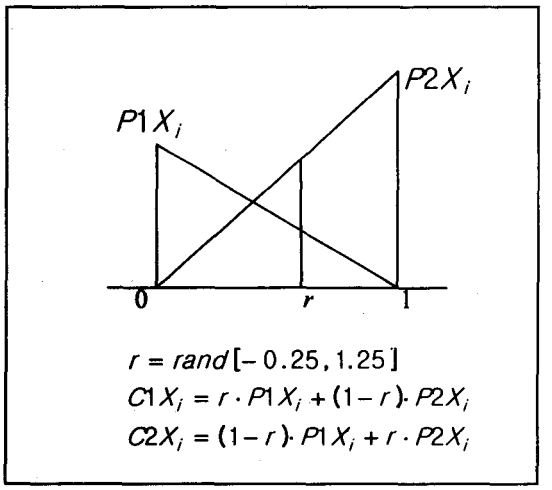

Fig. 4. Linear Inter/Extrapolation Scheme Used in Modified Crossover

ancestor-pool, we could thus reduce some amount of the total execution time in the MGA.

It is not always guaranteed that the up-to-best solution will be retained throughout the generation changes in the SGA. Thus, the elite-policy or one of its variations is often employed to retain these solutions hoping that their offspring contributes to the fitness improvement. However, the dense application of the elite-policy may lead to a pre-maturing result of the searching process. On the other hand, we are sure that the up-to-best solution and its families can never be lost undesirably by using the ancestor-pool in the MGA. In other words, all the individuals in the population are re-initialized periodically (e. g., after every 5 generation change) from the up-to-best solutions stored in the ancestor-pool. This periodic re-initialization of the population makes the searching process more stable and less divergent. Also, it accelerates the improvements of the solutions. However, it should be noted that too frequent re-initialization of the population may also drive the searching process toward a pre-maturing result. Reasonable interval of the population re-initialization should be chosen carefully.

To determine the proper weighting matrices of the LQR system by using the MGA, we modified the aforementioned formulation as follows :

$$\textit{Find} \qquad C = \left[a_1, b_1, a_2, b_2, \ldots, a_6, b_6\right] \tag{8}$$

$$\textit{to maximize} \qquad \textit{fitness}(X) = 1/\textit{Cost}(X) \tag{9}$$

subject to

$$\textit{Cost}(X) = \sum_{k=0}^{\infty} |y(k) - y_0| + \sum_{k=0}^{\infty} |u(k)| \tag{10}$$

$$X = \left[x_1, x_2, \ldots, x_6\right] \tag{11}$$

$$x_i = a_i \cdot e^{b_i}, \quad i = 1,2,\ldots,6 \tag{12}$$

$$1.00 < a_i < 9.99, \quad i = 1,2,\ldots,6 \tag{13}$$

$$-6 < b_i < -2, \quad i = 1,2,\ldots,6 \tag{14}$$

where,

C : chromosome of an individual (or a candidate solution)

X : a candidate solution vector

x_i : diagonal elements of the weighting matrix Q

$Cost(X)$: cost function for the candidate solution vector

$y(k)$: normalized output of the system at time step k

y_0 : normalized target value of the system output

$u(k)$: normalized velocity of the control rod at time step k

Combination of the MGA with the SA

As mentioned above, the SA and the GA have their own advantages and disadvantages. In other words, the SA is a one-point searching scheme and suitable to locate the final solution. However, its performance in the improvement of the solution is generally rather slow at the earlier stage of the searching process compared with the GA. The other difficulties in the SA are the proper design of the cooling schedule and the neighborhood generation mechanism. On the other hand, the GA is a multi-point searching scheme and thus more efficient than the SA for the global search in the earlier searching stage. However, the GA also has its own drawback in that it tends to wander around the true solution at the final stage of the searching process due to this multi-point searching scheme [16,17]. Thus, we may expect more improved performance in searching an optimal solution through the combination of these two methods so that the GA should compensate the drawbacks of the SA or vice versa. Following is our proposed algorithm which we named a hybrid MGA-SA;

Firstly, perform the stochastic random but evolutionary search by using the MGA for the large feasible domain until a solution of pre-specified quality is found. Then, start the SA to locate a final solution with the MGA's result as an initial solution.

5. Application Results and Discussion

All the searching modules based upon the aforementioned optimization algorithms have been implemented in the C-language, and tested on the PC Pentium 120MHz machine. The population size is 20 both for the SGA and the MGA.

Fig. 5 shows the curves of the run-time and best-cost vs. the generation for the SGA and the MGA. As can be seen in the figure, the MGA saved about 10% of the run time compared with the SGA. Also, the MGA gives more converged result than the SGA does. Furthermore, the MGA seems to be more stable and less divergent than the SGA owing to the usage of the ancestor-pool.

Fig. 6 shows the curves of the best-cost and the relative improvements vs. the run-time to compare the performance of the tested stochastic approaches. Here, the 'cost0' indicates the reference cost obtained through the divide-and-conquer method [2]. As can be seen in the figure, the hybrid MGA-SA shows the superior results both in the run-time and the minimization of the cost function.

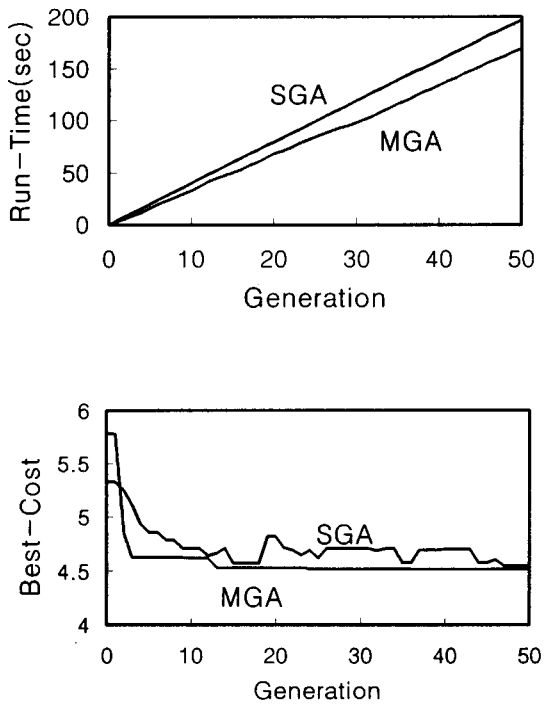

Fig. 5. Run-Time and Best-Cost vs. Generation

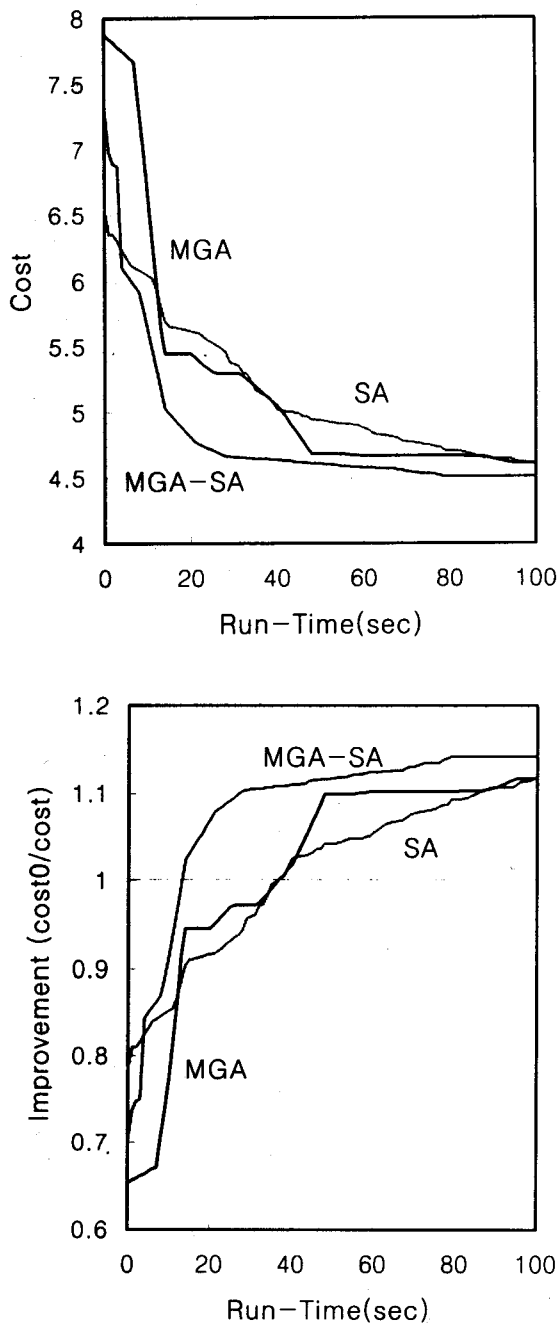

Fig. 6. Best-Cost and the Relative Improvements vs. the Run-Time

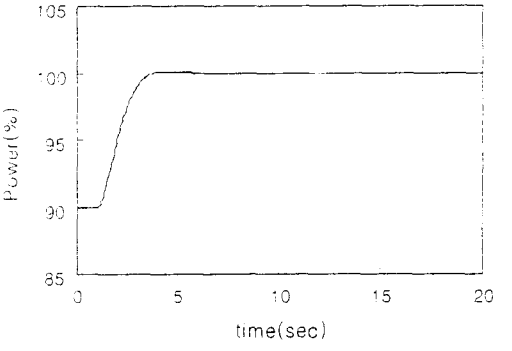

Fig. 7. Reactor Power Output by the Conventional Method [2]

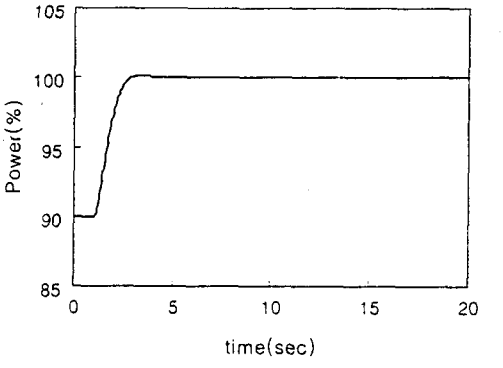

Fig. 9. Reactor Power Output by the MGA-SA

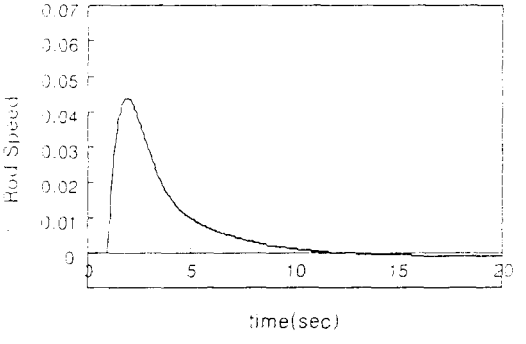

Fig. 8. Normalized Control Rod Speed by the Conventional Method [2]

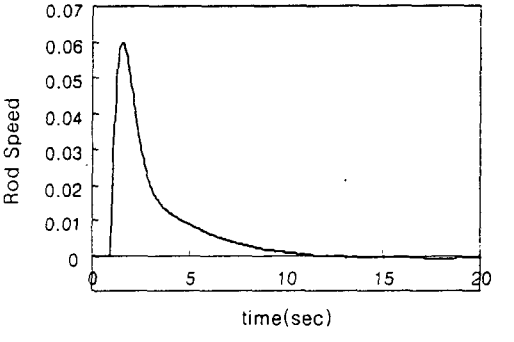

Fig. 10. Normalized Control Rod Speed by the MGA-SA

Table 2. Results of 10 Runs of the MGA-SA

run	x1	x2	x3	x4	x5	x6	cost	imprmt
1	0.00105	0.00016	1.00E−06	0.00044	0.00061	0.00967	4.5093	1.141
2	0.00075	0.08600	1.00E−06	0.00016	0.00873	0.00966	4.5091	1.141
3	0.00059	0.09840	1.00E−06	1.00E−06	0.00019	0.00965	4.5091	1.141
4	0.00058	0.00036	1.00E−06	0.00034	0.00054	0.00967	4.5088	1.141
5	0.00351	0.00065	1.00E−06	0.00001	0.00007	0.00966	4.5085	1.142
6	0.00028	0.00069	1.00E−06	0.00019	0.00026	0.00966	4.5085	1.142
7	0.00624	0.00031	1.00E−06	1.00E−06	0.00009	0.00967	4.5084	1.142
8	0.00370	0.00051	1.00E−06	1.00E−06	0.00012	0.00967	4.5078	1.142
9	0.00808	0.00046	1.00E−06	1.00E−06	0.00629	0.00969	4.5075	1.142
10	0.05040	0.02820	1.00E−06	1.00E−06	0.00053	0.00995	4.5018	1.143
Avg	0.00752	0.02157	1.00E−06	0.00011	0.00174	0.00970	4.5079	1.142
ref*	3.00E−6	3.00E−6	3.00E−6	3.00E−6	3.00E−6	0.003	5.1456	1.000

*best result of the ref[2] obtained by the conventional approach

Figs 7 and 8 show the responses of the OIRS for the case of conventional design. Fig. 7 describes the reactor output when the power is step increased by 10% from the initial state of 90% power. The peak value of the output is sufficiently lower than the FSAR's set forth value of 103%. The relative control rod velocity is shown in Fig. 8. All these results are obtained throughout numerous simulations. However, it can not be affirmed that these results are uniquely the best ones.

On the other hand, Figs 9, 10 and Table 1 illustrate the results of the MGA-SA which show the relative improvements of more than 14% compared with the above results [2]. Throughout the 10 test-runs, only the weighting parameter x_3 and x_6 showed converged values of $1.e^{-6}$ and $9.70e^{-3}$, respectively. Other parameters ranged rather widely and showed no representatively converged values. This can be explained by the system model in which the temperature feedback effects and the error integration effects play the major role in the transient.

7. Conclusions

We applied several stochastic searching algorithms such as the SGA, the MGA, the SA and the hybrid MGA-SA to the determination of the weighting parameters of the LQR reactor power control system, and investigated their performances throughout several test runs. These stochastic searching methods usually gave reasonable results for our problems. Among these methods, however, the hybrid MGA-SA gave the best performances both in the execution time and in the solution's quality.

The MGA which was the modified version of the SGA showed several advantages of 1) time saving, 2) never lost of the up-to-best solutions by periodic re-initialization of population, 3) stable and less divergent search, 4) easy determination of the effective order of the design parameters, and disadvantage of pre-maturing compared with the SGA.

Among the weighting parameters, the dominant ones x_3 and x_6 whose representative values we obtained by using the MGA-SA are $1.e^{-6}$ and $9.70e^{-3}$, respectively. Other parameters range rather widely and show no representative values. This is because the state variables corresponding to these parameters are less dominant ones in our LQR system.

Even though the cost function we dealt with in this work was somewhat simple and was carrying almost minimum interactions with the concerned LQR system, the proposed approach gave the satisfactory results. This fact implies that the great advantages of the proposed approach are the easiness of the implementation and robustness in finding a qualified result.

References

1. Y.J. Lee, "The Control Rod Speed Design for the Nuclear Reactor Power Control Using Optimal Control Theory," *J. of the KNS*, **26**(4), pp. 536-537 (1994)
2. Y.J. Lee, "A Conceptual Design of the Digital Nuclear Power Control System by the Order Increased LQR Method," *Procd. of ANS Topical Meeting on NPIC*, Penn. State Univ., pp. 827-834 (1996)
3. S. Kirkpatrick, C.D. Gelatt Jr. And M.P. Vecchi, "Optimization by Simulated Annealing," *Science*, **220**, pp. 671-680, (1983)
4. P.J.M. van Laarhoven and E.H.L. Aarts, *Simulated Annealing : Theory and Applications*, D. Reidel Publishing Co., Boston, (1988)
5. D.F. Wong, H.W. Leong and C.L. *Liu, Simulated Annealing for VLSI Design*, Kluwer Academic Publishers, Boston, (1988)
6. K.H. Cho and K. Lee, "Tool Path Optimization for NC Turret Operation Using Simulated Annealing," *J. of KSME*, **17**(5), pp. 1183-1192, (1993)
7. K.H. Cho and K. Lee, "An Effective Method for

the Nesting on Several Irregular Raw Sheets," *J. of KSME*, **19**(8), pp. 1183-1192, (1995)

8. J.C. Park, K. Lee and K.H. Cho, "Optimization of Tool/Workpiece Orientation in Designing Die-Face of Automobile Outer Panels" *Proc Instn Mech Engrs, Part B : J. of Engineering Manufacture*, **209**, pp. 9-18, (1995)
9. D.E. Goldberg, *Genetic Algorithm in Search, Optimization & Machine Learning*, Addison-Wesley Publishing Co., NY, (1989)
10. J.R. Koza, *Genetic Programming on the Programming of Computers by Means of Natural Selection*, MIT Press, (1992)
11. K. DeJong, "Learning with Genetic Algorithms : An Overview," *Machine Learning* **3**, Kluwer Academic Publishers, pp. 121-138, (1988)
12. B.P. Buckles and F.E. Petry, *Genetic Algorithms*, IEEE Computer Society Press, Los Alamitos, (1992)
13. M. Srinivas and L.M. Patnaik, "Genetic Algorithms : A Survey," *Computer*, **27**(6), pp. 17-26, (1994)
14. L. Jose and C. Alippi, "Genetic-Algorithm Programming Environments," *Computer*, **27**(6), pp. 28-43, (1994)
15. J.D. Faires and R.L. Burden, *Numerical Methods*, PWS Publishing Company, Boston, pp. 25-50, (1993)
16. K.H. Cho, S.T. Ko and H.S. Ko, "A Proposal of New Method for EICT Image Reconstruction-A Hybrid Approach Using Genetic Algorithm and Newton-Raphson Method-," *J. of KSEE*, **33** (B-4), pp. 91-100, (1996)
17. K.H. Cho, N.G. Hyun and J.B. Choi, "Determination of the Optimal Parameters for the Meson Spectra Analysis Using the Hybrid Genetic Algorithm and Newton Method," *J. of the Korean Physical Society*, **29**(4), pp. 420-427, (1996) 12