International Journal of Control, Automation, and Systems, vol. 3, no. 2, pp. 236-243, June 2005

SA-selection-based Genetic Algorithm for the Design of Fuzzy Controller

Chang-Wook Han and Jung-Il Park

Abstract: This paper presents a new stochastic approach for solving combinatorial optimization problems by using a new selection method, i.e. SA-selection, in genetic algorithm (GA). This approach combines GA with simulated annealing (SA) to improve the performance of GA. GA and SA have complementary strengths and weaknesses. While GA explores the search space by means of population of search points, it suffers from poor convergence properties. SA, by contrast, has good convergence properties, but it cannot explore the search space by means of population. However, SA does employ a completely local selection strategy where the current candidate and the new modification are evaluated and compared. To verify the effectiveness of the proposed method, the optimization of a fuzzy controller for balancing an inverted pendulum on a cart is considered.

Keywords: Combinatorial optimization problem, fuzzy control, genetic algorithm, simulated annealing.

1. INTRODUCTION

Throughout the last decades, the role of optimization has steadily increased in such diverse areas, such as, electrical engineering, computer science, and communication. In practice, optimization problems become more and more complex. For instance, many large scale combinatorial optimization problems can only be solved approximately, which is closely related to the fact that many of these problems have been proved NP-hard. Their deterministic polynomial time algorithms are unlikely to exist. The quality of the final solution is in contradiction with computation time. To search an optimum of a function with continuous variables is difficult if there are many peaks and valleys. In these cases, traditional optimization methods are not competent. They will either be trapped to local optima or need much more search time. In recent years, many researchers have been trying to find new ways to solve these difficult problems, and stochastic approaches have attracted much attention [1-5].

Simulated annealing (SA) [6] and genetic algorithm (GA) [3] represent powerful combinatorial optimization methods with complementary strengths and weaknesses. Each method requires little knowledge regarding the problem which is to be optimized. The only necessary information required is knowing the fitness or cost function that needs to be applied to the candidate solutions. The two techniques initially begin a search through the space of candidate solutions with randomly generated candidates, and then they incrementally generate new candidates by applying operators. Each decision determining which candidates are pursued is controlled by a probabilistic decision procedure that guides the method into near optimal regions of the solution space. To get the synergy effect between GA and SA, many literatures considered the combination of each other [7-9]. Tabu search was combined with GA and SA, but much more memory space is needed to store the visited states [7]. In [8], chemotaxis algorithm was used to improve the performance of GA and SA, the evolution strategies controlled the step size of the variables and SA was used for termination and selection criteria for the evolution strategies in [9].

This paper proposes a new method that combines the recombinative power of GA and local selection of SA by using a SA-selection. The new algorithm takes advantage of those aspects of GA that lend themselves to population-based optimization, and avoid serial bottle necks of GA approaches by incorporating elements of SA. The proposed method is applied to the optimization of a fuzzy controller for balancing an inverted pendulum on a cart.

2. SIMULATED ANNEALING AND GENETIC ALGORITHM

Before proceeding with the detailed explanation of the proposed algorithm, we will briefly remind the

Manuscript received November 14, 2003; revised November 22, 2004; accepted March 9, 2005. Recommended by Editorial Board member Young Il Lee under the direction of Editor Jin Bae Park.

Chang-Wook Han and Jung-Il Park are with the School of Electrical Engineering and Computer Science, Yeungnam University, 214-1 Dae-dong, Gyongsan, Gyongbuk 712-749, Korea (e-mails: {cwhan, jipark}@yumail.ac.kr).

two basic traditional algorithms and their comparisons.

2.1. Simulated annealing (SA)

SA is a method that has attracted significant attention for large-scale optimization problems that have many local minima and make reaching the global minimum difficult by randomly generating a candidate solution, and then making successive random modifications.

The idea of SA comes from the physical annealing process done on metals and other substances. In metallurgical annealing, a metal body is heated to near its melting point, and then slowly cooled back down to absolute zero temperature. This process will cause the global energy function of the metal to reach on absolute minimum value eventually. If the temperature is dropped too quickly, the energy of the metallic lattice will be much higher than this minimum because of the existence of frozen lattice dislocations that would otherwise disappear eventually because of thermal agitation.

Analogous to this physical behavior, SA allows a system to change its state to a higher energy state occasionally such that it has a chance to jump out of local minima and seek the global minimum. The function to be minimized, i.e. the performance index, is analogous to the energy of metal, and the control parameter, called temperature, is analogous to the temperature of metal. Downhill moves are always accepted, whereas uphill moves are accepted with the probability that is a function of temperature. The acceptance probability is expressed in the following form:

$$P = \exp(-\Delta Q / T)\,, \tag{1}$$

where $\varDelta Q = Q_{new} - Q_{old}$ is the change of performance index which will be defined later, and T is the current temperature. If $\varDelta Q$ is negative then the new solution is always accepted. If $\varDelta Q$ is positive then the new solution could be accepted with the acceptance probability in (1). The possibility of accepting the uphill move permits escaping from the local minima.

After the decision of acceptance of a new solution, the current temperature is adjusted by cooling schedule, and the process is repeated until some convergence conditions are reached. Because of its speed, the exponential cooling schedule is used in this research. The temperature at any stage during the optimization may be expressed as follows:

$$T_k = \alpha^k T_0\,, \tag{2}$$

where T_0 is the initial temperature that is large enough, α is the cooling rate, and k is the time index.

2.2. Genetic algorithm (GA)

GA is a search algorithm based on an analogy with the process of natural selection and evolutionary genetics. The conventional GA has three basic operators, such as selection, crossover, and mutation.

In the selection operator, strings with high fitness have multiple copies in the next generation, whereas strings with low fitness have fewer copies or even none at all. In this paper, we apply proposed SA-selection to GA, while tournament selection is considered in GA to compare with the proposed method.

The crossover operator produces two offspring by recombining the information from two parents in two steps. First, a given number of crossing sites are selected along the parent strings uniformly at random. Second, two new strings are formed by exchanging alternate pairs of selection between the selected sites. In the simplest form, crossover with a single crossing site refers to taking a string, splitting it into two parts at a randomly generated crossover point, and recombining it with another string which has also been split at the same crossover point. This procedure serves to promote change in the best strings which could give them even higher fitness. This procedure will be executed continuously until the crossover rate is reached. Single crossing site is considered in this paper.

The mutation operator is held to escape the local minima in the search space of the artificial genetic approach. A random position of a random individual is chosen and is replaced by another value. The total number of bits selected to mutate is settled by the mutation rate. In general, the mutation rate is quite small and fixed. In this research, if mutate, the selected real gene is changed for the random generated value.

2.3. Comparing GA with SA

Although GA and SA are similar, there are some important differences between the two methods. One important distinction is that SA possesses a formal proof of convergence to the global optimum, which GA does not have [10,11]. This convergence proof relies on a very slow cooling schedule of setting the temperature. While this cooling schedule is impractical, it identifies a useful trade-off where longer cooling schedules tend to lead to better quality solutions. There is no such control parameter in GA, and premature convergence to local optima is a significant problem. Another difference between GA and SA is that SA accepts newly generated candidate solutions probabilistically based on their performance index, and only accepts inferior candidates some of the time. This is in contrast to GA, where new candidates are always accepted, even though they are significantly inferior to older candidates. This

Table 1. SA-selection function for choosing the surviving solution given the three candidates.

Case	*SA-selection(offspring[i], parent[i], best solution, T[i])*			
	SA(offspring[i], parent[i], T[i])	*SA(offspring[i], best solution, T[i])*	*SA(parent[i], best solution, T[i])*	return
1	*offspring[i]*	*offspring[i]*	*parent[i]*	*offspring[i]*
2	*offspring[i]*	*offspring[i]*	*best solution*	*offspring[i]*
3	*offspring[i]*	*best solution*	*parent[i]*	*offspring[i]*
4	*offspring[i]*	*best solution*	*best solution*	*best solution*
5	*parent[i]*	*offspring[i]*	*parent[i]*	*parent[i]*
6	*parent[i]*	*offspring[i]*	*best solution*	*offspring[i]*
7	*parent[i]*	*best solution*	*parent[i]*	*parent[i]*
8	*parent[i]*	*best solution*	*best solution*	*best solution*

characteristic can lead to disruption, where good candidates are lost or damaged, preventing optimal performance.

Despite these shortcomings, there are some distinct advantages of GA over SA. GA maintains a population of candidate solutions, while SA maintains only one solution (serial nature). This has many significant impacts on how the solution space is searched. GA can retain useful redundant information about what it has learned from previous searches by its representation in individual solutions in the population. Critical components of past good solutions can be captured, which can be combined together via crossover to form high quality solutions. SA, on the other hand, retains only one solution in the space, and exploration is limited to the immediate neighborhood.

In this research, the hybrid method that combines the recombinative power of GA and the annealing schedule of SA is presented.

3. PROPOSED ALGORITHM

As mentioned before, in this section, the new selection method called SA-selection is introduced and applied to GA to get the synergy effect between GA and SA. The main concept of SA-selection is to choose a single candidate solution between *parent*, *offspring*, and *best solution* of the generations, where offsprings are taken by applying crossover and mutation to the parents as shown in Fig. 1. To do so, the selection function *SA-selection(offspring[i], parent[i], best solution, T[i])* is defined, which applies the traditional SA function *SA(a, b, T)* multiple times to identify the single surviving candidate, where *i* of the SA-selection function indicates the index of the individual. The function *SA(a, b, T)* calculates the acceptance probability $P=exp(-(cost(a)-cost(b))/T)$, where *cost(a)* is the performance index of candidate *a*. If $cost(a) \leq cost(b)$, then candidate *a* will be selected. If $cost(a) > cost(b)$ and *P>random number in [0, 1]*, then candidate *a* will be selected too. Except these cases, candidate *b* will be selected. These processes are represented in Table 1. In this SA-selection function, *offspring[i]* and *parent[i]* are compared with each other, then with *best solution*. For example, consider case 2 in Table 1, where *offspring[i]* is accepted over both *parent[i]* and *best solution*, but *best solution* is accepted over *parent[i]*. Hence, *offspring[i]* is accepted overall and returned. But in case 3 and 6 a candidate is not uniquely determined. In these cases, *offspring[i]* is always accepted to maintain genetic diversity. This situation only occurs at high temperature, where the affect of temperature dominates, rather than individual solution cost that dominates at low temperature. Obviously, case 3 and 6 in Table 1 will not happen for the temperatures near zero.

Additionally, in the proposed algorithm, we use random generated initial temperatures for the SA-selection of each individual, so that some individuals tend to accept the best-fit solution's individual (exploitation), and some other individuals allow uphill move with the higher acceptance probability (exploration), respectively. This means that the population stores a diversity of annealing schedules too. For instance, an individual that has the initial temperature near zero tend to select the best-fit solution for all generations, while an individual that has high initial temperature maintains search diversity. If we use random generated initial temperatures for the SA-selection, it is not needed to tune the initial temperature because the initial temperature is set through a trial and error process in traditional SA. In the sequel, tournament-selection can be effectively replaced by SA-selection without increasing the number of performance index evaluations per generation, because SA allows uphill move to explore the search space in higher temperature, and exploit the search space accepting the best-fit solution's individual among *offspring*, *parent*, and *best solution* in lower temperature. The flowchart of the proposed algorithm is described in Fig. 1.

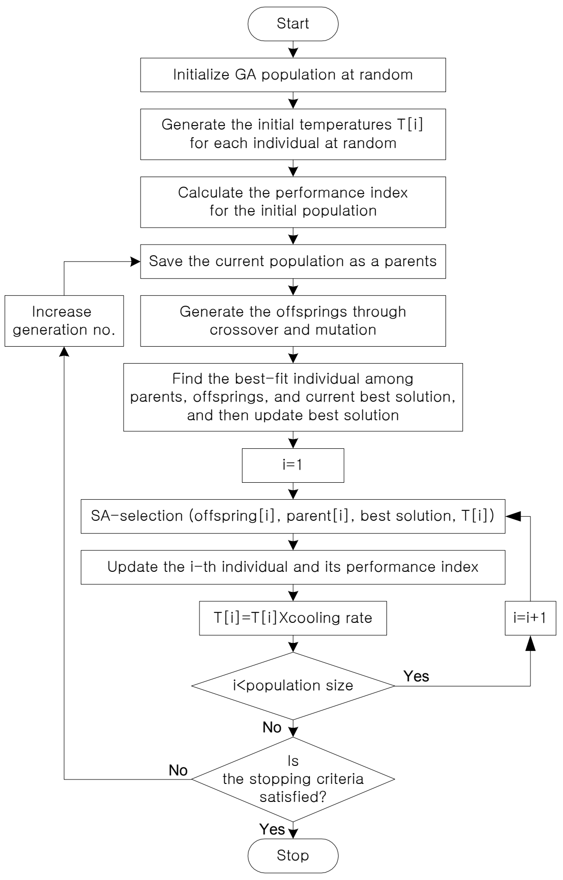

Start
Initialize GA population at random
Generate the initial temperatures T[i] for each individual at random
Calculate the performance index for the initial population
Save the current population as a parents
Increase generation no.
Generate the offsprings through crossover and mutation
Find the best-fit individual among parents, offsprings, and current best solution, and then update best solution
i=1
SA-selection (offspring[i], parent[i], best solution, T[i])
Update the i-th individual and its performance index
T[i]=T[i]Xcooling rate
i=i+1
i<population size
Yes
No
Is the stopping criteria satisfied?
No
Yes
Stop

Fig. 1. Flow chart of the proposed algorithm.

The following example shows how two parents produce offsprings and finally select one for each individual. We assume that genes can get integer value in [0, 9] and the length of string is 5.

(Example)
string of parent i : 48391
string of parent j : 27916

Crossover with the crossing site between the third and the fourth gene, and then mutate the second gene of parent j by generating random integer value in [0,9].

String of offspring i : 483|16
String of offspring j : 2$\underline{5}$9|91

Choose the surviving candidate using *SA-selection* function for each individual, i and j, with T[i] and T[j], respectively.

The proposed algorithm has the following pseudo-code:

```
begin
initialize population at random
randomly generate the initial temperature T[i] in a specified region
calculate the performance index for the initial population
for k=1 to stopping criterion
   begin
   save the current population as parents
   crossover
   mutation
   find the best-fit individual among the parents, offsprings, and current best solution, and then, update best solution
   for i=1 to population size
      begin
      i-th individual=SA-selection(offspring[i], parent[i], best solution, T[i])
      update the performance index of the i-th individual
      T[i]=T[i]×cooling rate
      end
   end
end
```

4. EMPIRICAL STUDIES

To verify the validity of the proposed algorithm, the optimizations of test functions and fuzzy controller are considered. The parameters of the optimization environment are included in Table 2 and these are fixed for all experiments; it has been found that under these conditions the optimization was carried out efficiently and resulted in meaningful results. Obviously we do not claim that these specific values are optimal; as a matter of fact any fine-tuning of the collection of the parameters may be quite tedious and somewhat counterproductive as any new problem will undoubtedly require very intensive experimentation. Standard form of GA with real number encoding is used for genetic optimization, where mutation operator generates random number in the range. The final temperatures of the proposed algorithm ($0.99^{200(500)} \times T_{rand\_init}$) and SA ($0.99^{40000(100000)} \times T_{init}$) are different because the proposed algorithm has a diversity of annealing schedules in the population, where some of the final temperature will be similar to that of SA, and SA needs very long convergence time.

Table 2. Experimental setup of the parameters of the optimization environment; the parameter values in the brackets ( ) are that of test functions.

GA	Population size	200
	Maximum number of generation	200(500)
	Crossover rate	0.8
	Mutation rate	0.1
SA	Initial temperature	1
	Cooling rate	0.99
	Maximum number of iteration	40,000(100,000)
Proposed algorithm	Random initial temperature	[0, 1]
	Cooling rate	0.99
	GA parameters are the same as above	

4.1. Test function problem

To show the validity and superiority of the proposed algorithm, three benchmark test functions [12], described in Table 3, are considered. In this example, the performance of the proposed algorithm is compared with that of SAGACIA algorithm [12]. In [12], they considered low dimensional test functions, rather than high dimensional ones, while this experiment optimizes 30-dimensional problems except one of them.

The results of optimization are summarized in Table 4. This table shows mean and standard deviation of the performance index when we run each algorithm 10 times per function. For the reasonable comparison, all the parameters are set to have the same computational burden for each algorithm. For the low dimensional problem (Shaffer's f6), the two algorithms always find the global optimum, but the proposed algorithm is always superior to SAGACIA algorithm in high dimensional problems (Sphere and Step). Since SAGACIA algorithm has a possibility to missing the best state at high temperature and mainly depends on random search, though it is based on population-based learning structure, it suffers from poor convergence properties, i.e. convergence speed and accuracy to a global optimum. As can be seen, the proposed algorithm is superior to SAGACIA algorithm in convergence speed and accuracy for high dimensional problems, and finally reaches near global optimum.

4.2. Fuzzy controller design problem

In this experiment, the optimization of a fuzzy controller for balancing an inverted pendulum system,

Table 3. Specifications of the test functions.

Name	Function	No. of variables	Variable range	Global optimum
Shaffer's f6	$f_{shaffer}(\mathbf{x}) = 0.5 - \frac{\sin^2(\sqrt{x_1^2 + x_2^2}) - 0.5}{(1+0.001(x_1^2+x_2^2))^2}$	2	[-4, 4]	1
Sphere	$f_{sphere}(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$	30	[-5.12, 5.12]	0
Step	$f_{step}(\mathbf{x}) = \sum_{i=1}^{n} \text{integer}(x_i)$	30	[-5.12, 5.12]	-180

Table 4. The results of each algorithm for the test function optimization.

Function	Algorithm	Mean	Standard deviation
Shaffer's f6	SAGACIA	1.0000	0.0000
	Proposed algorithm	1.0000	0.0000
Sphere	SAGACIA	9.2719	1.4716
	Proposed algorithm	0.2523	0.0115
Step	SAGACIA	-157.2	4.2641
	Proposed algorithm	-175.7	0.3174

commonly used as a benchmark test bed where a free-falling pole is mounted on a cart, is considered here. The control objective is to produce an appropriate actuator force to control the motion of the cart so that the pole can be balanced in a vertical position. Given that no friction exists in the system, and let $x_1 = \theta$ and $x_2 = \dot{\theta}$, then the state equation can be expressed as

$$\dot{x}_1 = x_2,$$
$$\dot{x}_2 = \frac{(M+m)g\sin x_1 - (F + mlx_2^2 \sin x_1)\cos x_1}{\{4/3(M+m) - m(\cos x_1)^2\}l}, \tag{3}$$

where M (mass of the cart) is 1.0Kg, m (mass of the pole) is 0.1Kg, l (half length of the pole) is 0.5m, g (gravity acceleration) is 9.8m/s^2, and F is the applied force in Newton.

The experiments for GA, SA, and the proposed algorithm are taken into account to show the effectiveness of the proposed algorithm. In this experiment, the comparison between the proposed algorithm and another hybrid algorithm is not considered, because it has been done already in previous complex examples.

To simplify the problem, only the control of the pole is considered for the inverted pendulum system, that is, the considered state is angle, θ, and angular velocity, $\dot{\theta}$, of the pole with respect to the vertical axis [13,14]. The fuzzy controller for this system consists of 25 possible rules that have antecedent parts with 5 fixed triangular membership functions (fuzzy sets) for each input variable as shown in Fig. 2, and 25 consequent part membership functions. Mamdani-type fuzzy model [15] is used in this example. The proposed algorithm optimizes the centers and widths of the consequent part membership functions of the rule.

The considered form of the chromosome to optimize the fuzzy controller is described in Fig. 3. The centers and widths of the consequent part membership functions are the phenotypes of the chromosome, and they are constrained in specified regions. We use this structure of chromosome for GA, SA, and the proposed algorithm all together.

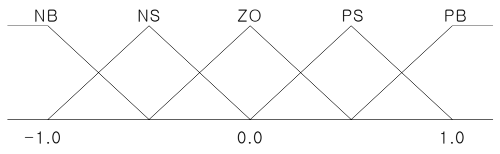


Fig. 2. Fixed antecedent part membership functions for each input variable.

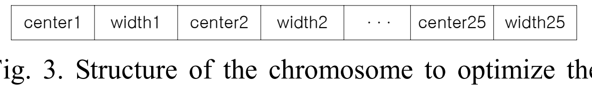


Fig. 3. Structure of the chromosome to optimize the fuzzy controller of the inverted pendulum.

In this experiment, the following performance index is used

$$Q = \sum_{i=1}^{q} [e_i^2 + \dot{e}_i^2], \quad (4)$$

where q is the number of input-output pairs. The performance index is evaluated by controlling the inverted pendulum during 20s (seconds), where the initial angle and angular velocity of the pendulum are 0.3rad (17.2°) and zero, respectively. The failure limit of the angle of the pendulum is ±0.52rad (±30.0°) and the sampling period is 0.01s. As mentioned before, the 25 centers and widths of consequent part membership functions are adjusted so that the optimized fuzzy controller can generate the desired performance.

Fig. 4 shows the averaged control result (average of 10 times simulation) of the inverted pendulum obtained by the optimized fuzzy controllers when each algorithm is executed 10 times, independently. In this figure, the initial state, which is the initial values of the pendulum angle and the angular velocity, is the same as the optimization process, which is 0.3rad and zero, respectively. The disturbance of 0.1rad is added to the pendulum angle at 10.0s. The result in this figure shows that the control performance obtained by the proposed algorithm is superior to GA and SA in convergence time and in disturbance rejection.

In Fig. 5, the averaged performance index in successive generations for GA and the proposed algorithm, and the averaged performance index in successive iterations for SA are described. From this figure, we can see that the performance of the proposed algorithm is greatly enhanced in learning speed and accuracy.

To check the generalization ability of the optimized fuzzy controllers, the length of the pendulum that is the most sensitive parameter is changed, while the other parameters of the inverted pendulum are all fixed. Fig. 6 depicts the averaged result obtained by the 10 optimized fuzzy controllers when the pendulum is 0.2m long. A small amount of the cart moving can

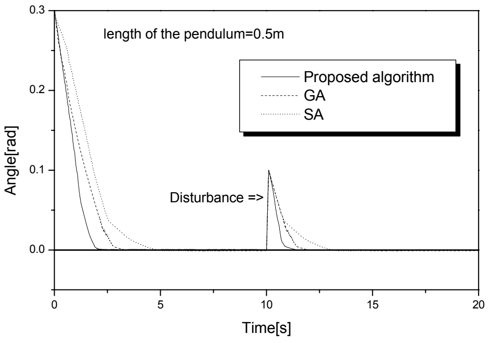
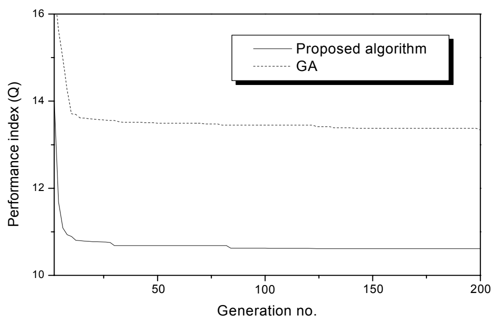
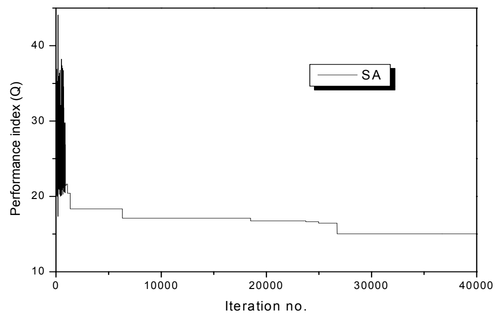


Fig. 4. Averaged control result over 10 times independent control (10 optimized fuzzy controllers) for GA, SA, and the proposed algorithm.



(a)



(b)

Fig. 5. Averaged performance index in successive generations for GA and the proposed algorithm (a), and the averaged performance index in successive iterations for SA (b).

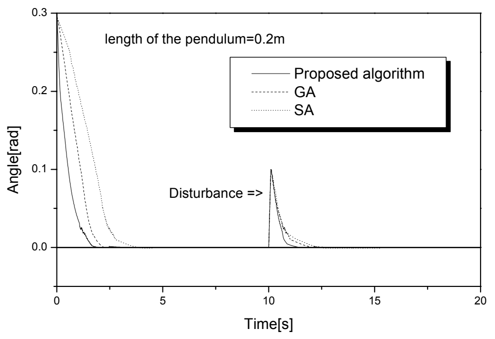


Fig. 6. Averaged control result over 10 times independent control (10 optimized fuzzy controllers) for GA, SA, and the proposed algorithm when the length of the pendulum is changed to 0.2m.

cause the pendulum to rotate because the pendulum is rather short (higher natural frequency). Therefore, the pendulum is balanced upright in a short time. Fig. 7 draws an example where the pendulum has a length of 1.0m. Since the pendulum is long, the pendulum has lower natural frequency and big momentum. In this case, it takes much more time to balance the pendulum upright because the cart has to move for a long distance to balance the pendulum.

Table 5 describes the settling time of each algorithm that is required for the output to settle within 1% of its final value, when the length of the pendulum is changed.

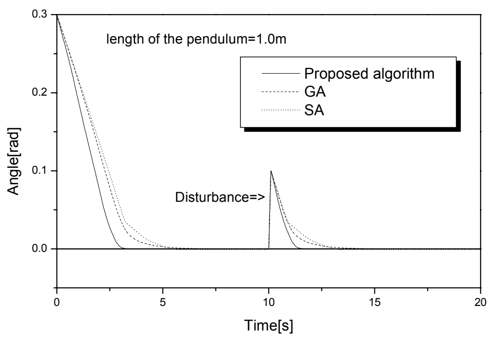


Fig. 7. Averaged control result over 10 times independent control (10 optimized fuzzy controllers) for GA, SA, and the proposed algorithm when the length of the pendulum is changed to 1.0m.

Table 5. 1% settling time of each algorithm when the length of the pendulum is changed; the values in the brackets ( ) are 1% settling time of the disturbance.

Algorithm \ l	l=0.2m	l=0.5m	l=1.0m
Proposed algorithm	1.69s (11.08s)	2.21s (11.12s)	2.98s (11.25s)
GA	2.07s (11.66s)	2.93s (11.79s)	4.88s (12.70s)
SA	3.41s (11.99s)	4.43s (12.70s)	5.03s (12.75s)

5. CONCLUSIONS

We have introduced a new selection method that combines the recombinative power of GA and annealing schedule of SA to get the synergy effect of GA and SA. In the proposed algorithm, tournament-selection was effectively replaced by SA-selection without increasing the number of performance index evaluations per generation. The proposed method was applied to the optimization of three test functions to show the effectiveness by comparing with another hybrid algorithm, i.e. SAGACIA. A number of simulations were considered with respect to the performance of the optimized fuzzy controller for the inverted pendulum and their generalization ability. The experimental results showed that the proposed algorithm is superior to GA, SA, and SAGACIA in terms of learning speed and accuracy. Moreover, the proposed algorithm can be effectively applied to any other combinatorial optimization problems.

REFERENCES

[1] K. De Jong, *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*, Ph.D. dissertation, Dept. Computer Sci., Univ. Michigan, Ann Arbor, MI, 1975.

[2] J. H. Holland, *Adaptation in Neural and Artificial Systems: An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence*, 2nd ed. Cambridge, MIT Press, 1992.

[3] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, MA, 1989.

[4] C.-W. Han and J.-I. Park, “Design of a fuzzy controller using random signal-based learning employing simulated annealing,” *Proc. of the 39th IEEE Conference on Decision and Control*, Sydney, Australia, pp. 396-397, December 2000.

[5] C.-W. Han and J.-I. Park, “A study on hybrid genetic algorithms using random signal-based learning employing simulated annealing,” *Proc. of the 2001 American Control Conference*, Arlington, Virginia, USA, pp. 198-199, June 2001.

[6] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, “Optimization by simulated annealing,” *Science*, vol. 220, no. 4598, pp. 671-680, May 1983.

[7] A. H. Mantawy, Y. L. Abdel-Magid, and S. Z. Selim, “Integrating genetic algorithms, tabu search, and simulated annealing for the unit commitment problem,” *IEEE Trans. on Power Systems*, vol. 14, no. 3, pp. 829-836, August 1999.

[8] B. Li and W. Jiang, “A novel stochastic optimization algorithm,” *IEEE Trans. on Systems, Man, and Cybernetics-Part B*, vol. 30, no. 1, pp. 193-198, February 2000.

[9] G. Alpaydin, G. Dundar, and S. Balkir, “Evolution-based design of neural fuzzy networks using self-adapting genetic parameters,” *IEEE Trans. on Fuzzy Systems*, vol. 10, no. 2, pp. 211-221, April 2002.

[10] F. Romeo and A. Sangiovanni-Vincentelli, “A theoretical framework for simulated annealing,” *Algorithmica*, vol. 6, pp. 302-345, 1991.

[11] G. Rudolph, “Convergence analysis of canonical genetic algorithms,” *IEEE Trans. on Neural Networks*, vol. 5, no. 1, pp. 96-101, Jan. 1994.

[12] B. Li and W. Jiang, “A novel stochastic optimization algorithm,” *IEEE Trans. on Systems, Man, and Cybernetics-Part B*, vol. 30, no. 1, pp. 193-198, February 2000.

[13] L.-X. Wang, “Automatic design of fuzzy controllers,” *Proc. of the American Control Conference*, vol. 3, pp. 1853-1854, 1998.

[14] M. Y. Shieh, C. W. Huang, and T. H. S. Li, “A GA-based Sugeno-type fuzzy logic controller for the cart-pole system,” *Proc. of the 23rd International Conference on Industrial Electronics, Control, and Instrumentation*, vol. 3, pp. 1028-1033, 1997.

[15] T. J. Procyk and E. H. Mamdani, “A linguistic self-organizing process controller,” *Automatica*, vol. 15, no. 1, pp. 15-30, 1979.

Chang-Wook Han received the B.S., M.S., and Ph.D. degrees in Electronic Engineering from Yeungnam University, Gyongsan, Korea in 1994, 1996, and 2002, respectively. From 1996 to 1997, he worked as an Engineer at Hyundai Heavy Industries Co., Ltd. During 2002-2003, he was a Post-doctoral Fellow in the Department of Electrical and Computer Engineering at the University of Alberta, Canada. He is currently a Lecturer in the School of Electrical Engineering and Computer Science, Yeungnam University, Korea. His research interests include fuzzy-neural networks, intelligent control, computational intelligence, and combinatorial optimization problems.

Jung-Il Park received the B.S. degree in Electronic Engineering from Kyoungbuk National University, Daegu, Korea in 1981, and the M.S. and Ph.D. degrees in Electronic Engineering from Seoul National University, Seoul, Korea, in 1983 and 1989, respectively. From 1989 to 1992 he worked as a Senior Engineer at Samsung Advanced Institute of Technology. He joined the faculty of the school of Electrical Engineering and Computer Science, Yeungnam University, Gyoungsan, Korea in 1992, where he has been a Full Professor since 2003. His current research interests are neural-based intelligent control, high precision motion control, linear motor control, network-based control, and mechatronics.