JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 17, 35-45 (2001)

Short Paper

A Self-Organizing Genetic Algorithm With a Eugenic Strategy

KAO-SHING HWANG, JENG-YIH CHIOU AND YUAN-PAO HSU
Department of Electrical Engineering
National Chung-Cheng University
Chiayi, Taiwan 711, R.O.C.

A Genetic Algorithm (GA) is general optimization technique suitable for solving nonlinear, multi-constraints, and combinatorial optimization problems. However, it may be slow in converging of failing to reach the global optimum. A novel strategy called Self-Organizing Eugenics Strategy (SOES) is proposed to overcome these shortcomings. In the proposed algorithm, a simplified adaptive resonance theory neural network (ART) is embedded to generate schemata, and the simulated annealing algorithm (SA) is applied to guiding the search toward an optimal solution. To illustrate its improved performance, the method is used to solve an optimization problem. Based on the results of experiments, the method demonstrates a better performance than the traditional GA and other genetic operations.

Keywords: genetic algorithm, self-organizing, eugenics strategy, adaptive resonance theory, simulated annealing algorithm

1. INTRODUCTION

Recently, the Genetic Algorithm (GA) has been applied to optimization and machine learning problems, such as function optimization, the traveling salesperson problem (TSP), the unit commitment (UC) problem, and so on [1-4]. It is a robust search algorithm due to its global search ability, which is based on the mechanics of natural genetics, natural selection, genetic recombination, and survival of the fittest [2]. Natural selection plays an important role in the performance of the decoded structures and chromosomes, because the processes of natural selection may cause the existing chromosomes to reproduce newer, better ones. Generally, reproduction and mutation may generate the chromosomes of children from those of their biological parents. However, the recombination process may create quite different chromosomes by combining those of their two parents.

Received December 24, 1999; revised March 28, 2000; accepted June 27, 2000
Communicated by Gen-Huey Chen.

For problem representation, GA, through an encoding mechanism, applies strings of binary bits (1s and 0s) representing chromosomes to describe multiple points in the search space of the problem domain. The value of each bit in the string is called an allele. They simulate natural evolution on populations of chromosomes during the search process. It is worth noting that GA solves the optimization problem by manipulation the genes in the chromosomes blindly without any knowledge about the information of the problem. The only information given is an evolution of the chromosome which is used to decide the selection of chromosomes so that better chromosomes can be reproduced. These features lead GA to be applicable to a wide variety of global optimization problems [5, 6]. The algorithm has also been used successfully for a wide range of applications including control, scheduling, design, robotics, machine meaning, combinatorial optimization, signal processing, image processing, economic forecasting, medical applications, and so on [7-13].

However, GA may be slow to converge or fail to reach the global optimum [14, 15]. This is due to the fact that the implementation of a conventional GA with the standard crossover and mutation operations always converges around optimal solutions, or in some worse cases, the method fails to converge to optimal solutions [3]. For example, at the beginning of the evolution any process, the GA initially converges rapidly, but then further improvement comes very slowly. Most of the runtime is spent in the later phase of the process where only small improvements are very slowly obtained. We introduce some approaches to overcome these shortcomings. The schema theorem is a famous approach, which can speed the rate of convergence of the traditional GA [2]. The theorem, originally derived by Holland, claimed that short, low-order, and highly fit schemata (called building blocks) combine to form a better string [2]. Generally, in a GA, highly fit, short, length schemata survive from one generation to the next by giving exponentially increasing copies to the observed best. All this takes place in parallel without special bookkeeping or memory; this processing leverage is called implicit parallelism. A schema is a similarity template that describes a subset of strings with similarities at certain string positions [2]. For example, the schema (-101-) describes a subset for four members: (01010, 01011, 11010, 11011), where '-' indicates "don't care," that can be 1 or 0. The meaning of the schema is clear if we think of it as a pattern matching device: a schema matches a particular string if, at every location in the schema, a '1' matches a '1' in the string, and a '0' matches a '0', a '-' matches either '1' or '0'. There are two quantities to be defined for a schema: schema order $o(H)$ and *schema defining length* $\delta(H)$. The order of schema is the number of fixed positions; for example, the order of the schema (-101-) is 3, and the order of the schema (-1---) is 1. The defining length of a schema is the distance between the first and last specific string position; for example, the defining length of the schema (011-1--) is 5-1=4; the defining length of the schema (0------) is 1-1=0. The fact that the schemata can speed the search of GA to find an acceptable solution has been proven. Although various approaches for schemata generation have been proposed [16, 17], most of these methods are heuristic or empirical and have limited evidence. They generally collect certain generations of chromosomes to do statistical analysis. Few can introduce automatic generation of schemata.

The adaptive resonance theory (ART) neural network is one of the self-organizing neural networks, which is controllable in pattern clustering [18, 19]. It includes a bottom-up competitive learning system coupled with a top-down pattern-learning system.

The capability of self-organizing and auto-clustering in the network inspired a systematic way to generate schemata for GA. Therefore, the property of autonomous clustering in an ART network may assist GA to generate useful schemata for guiding the search.

This paper is organize as follow: The strategy of Self-Organizing Eugenics Strategy (SOES), including the generation of schemata by means of a simplified ART neural network and SA to guide the search of the GA, is presented in next section. Several experiments, made to compare the performance with other approaches, are shown. In the final section we discuss the performance of the proposed algorithm and draw some conclusions.

2. THE SELF-ORGANIZING EUGENICS STRATEGY

The proposed algorithm, Self-Organizing Eugenics Strategy (SOES), consists of two special mechanisms within the framework of the original genetic algorithm. The SOES uses schemata, which are generated by an ART neural network, to guide search process and the simulated annealing algorithm for deciding the next generation. The SOES flowchart is shown in Fig. 1. The details of generating schemata and new generations are described in the next sub-sections. The fitness value of a schema is the condition for deciding which chromosome to put into the mating pool. The fitness value of a schema is the average fitness value of its all members.

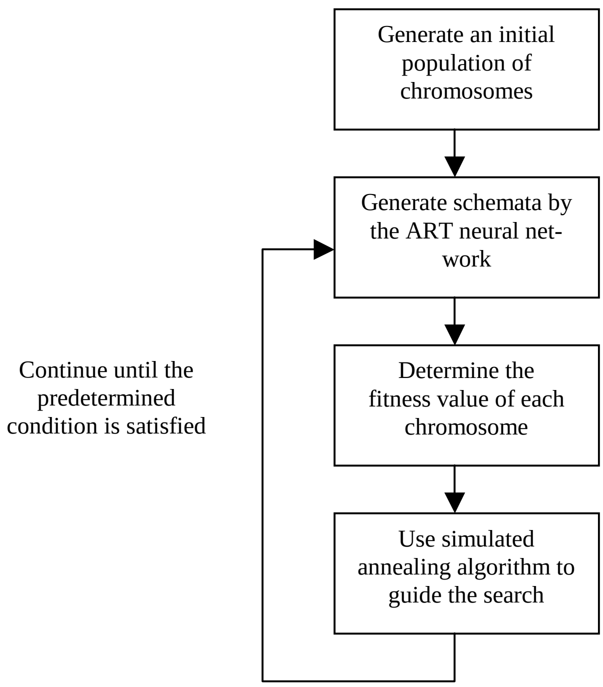

Fig. 1. The SOES flowchart.

2.1 Generating Schemata by ART

In the encoded string in a genetic algorithm, both “1” and “0” have their own individual meaning. But in the proposed method, only “1” is meaningful in the ART neural networks. Therefore, an additional procedure is needed for dealing with this. The ART neural network is first applied to generating the schemata of “1”. After inverting all the bits in the original string, the neural network is applied in turn to generating the schemata of “0”. In this way, both 1’s and 0’s schemata are obtained. The combinative schemata are generated by combining 1’s and 0’s schemata. An example of combining 1’s and 0’s schemata is illustrated in Table 1.

Table 1. Example of SOES to generate schemata by means of combining 1’s and 0’s schemata.

	Schemata
1’s	1011000
2’s	1100110
Final result	_01100_

In order to satisfy the schema theorem, an operation for recombination of two similar schemata is introduced [3]. For example, the schema (1100-1-) and the schema (1-0-110) can be combined into the schema (1-0--1-). The objective of the operation is to reduce schema order and schema defining length. The schema (1100-1-) and the schema (0-0-110) can’t be combined because the first bits of these schemata are specific and different. By means of the above operation, a final schema that is short and low-order can be obtained.

2.2 Using the Simulated Annealing Algorithm

A modified Simulated Annealing Algorithm (SA) is developed for guiding the search by the genetic algorithm in the proposed method. While the genetic algorithm operations, reproduction, crossover, and mutation, operate, a random number between 0 and 1 is generated to determine the solutions. If the random number is larger than p, which is obtained by a simulated annealing algorithm, the strings generated by the GA are accepted. Otherwise, the strings generated based on the schema is selected. The operations of the GA are roulette wheel parent selection, simple one-point crossover, and simple mutation. Guiding by the schemata means that crossover and mutation cannot destroy the structure of the schemata, and the high-fitness schemata receives more copies in the next population pool. The flowchart of the simulated annealing process is depicted as Fig. 2.

The simulated annealing algorithm used in SOES is different from the previous methods used in the other GAs in several aspects [20-22]. First, the definition of ΔE is different from the methods used in related genetic algorithms, which generate a solution randomly, make some random modification, and then accept the modification or retain the predecessor according to the probability p, that represents the state change probability. Nevertheless, the simulated annealing algorithm selects one of the two solutions; the

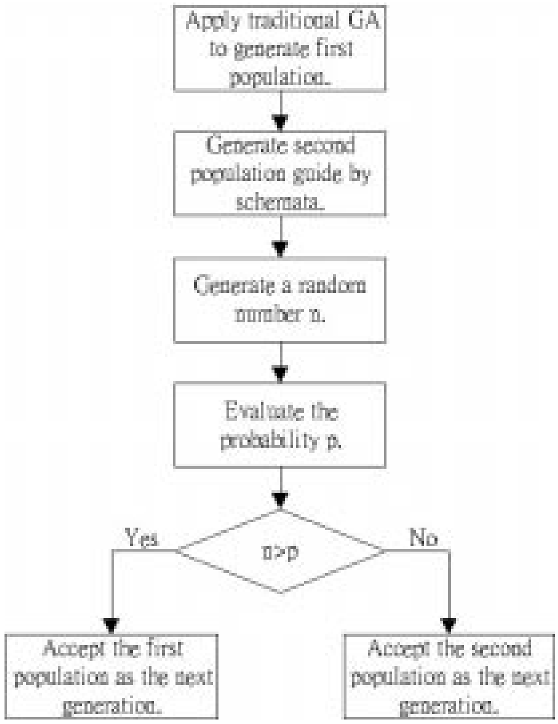

Fig. 2. The flowchart for simulated annealing.

operations of the GA generate one, while the other is generated based on schemata. The simulated annealing algorithm is applied to the operations of crossover, mutation, and reproduction. This differs form the way in which SA is usually applied in the process of mutation and crossover, or only mutation.

3. EXPERIMENTS AND RESULTS

In order to compare the performance of the proposed algorithm, SOES, with the traditional GA and other GAs several experiments were undertaken. In the experiments, an optimization problem is used to examine the schemata by comparing it to another method called Statistical Genetic Algorithm (STGA), whose schemata is generated by a statistical approach [16]. In the experiment, the relationship between the proposed mechanisms for generating schemata, and the simulated annealing algorithm is depicted. In addition, the speed of convergence and capability to escape from local minima are demonstrated.

The following objective function

$$f(x) = 1.5 + \left(\sum_{k=1}^{3} \frac{\sin(kx/2450)}{k} \right) \cdot \left(\sum_{k=1}^{3} \frac{\cos(kx/2450)}{k} \right), \ x \in (0,16383) \qquad (1)$$

is chosen for the experiments. The trajectory of f(x) is shown in Fig. 3. Note that it is a non-periodic function with four maxima. A simple binary encoding with 14 bit-width is used to represent the solutions.

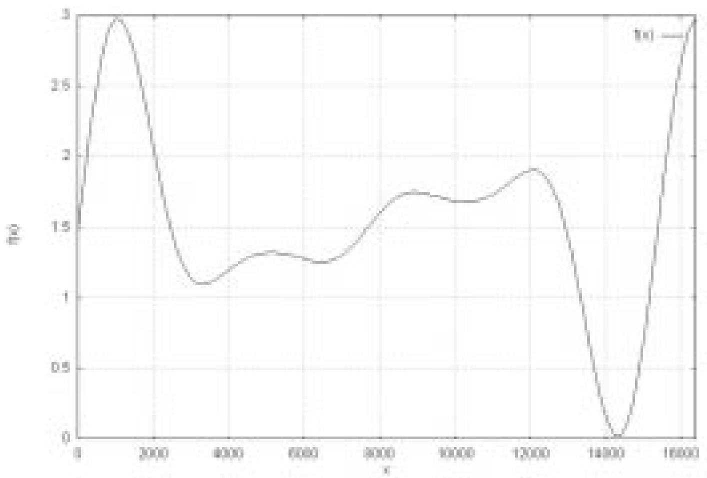

Fig. 3. The trajectory of $f(x)$ in Eq. (1).

The parameters of SOES, population size, the probability of crossover, the probability of mutation, and generations, are set equal to 40, 1.0, 0.01, and 60, respectively. In addition, the initial temperature of the simulated annealing algorithm is 1.0, and the decay of temperature is set to 0.95. These parameters in the genetic algorithms are the same as in the proposed genetic algorithm.

In this paper, the traditional GA means genetic algorithm; PGA denotes reproduction by roulette-wheel selection; RGA represents reproduction by remainder stochastic sampling without replacement; HGA is similar to the traditional one but with a reproduction operation guide by schemata. The operations of crossover and mutation of PGA, RGA, and HGA are all guided by schemata. Additionally, PGA, RGA, HGA have the same genetic operations and parameters as SOES, except for the way that schemata are generated.

Two approaches for generating schemata are used in the experiments. In the first approach, only one schema is generated in each generation. In the second approach, the generated schema survives until the end of the searching algorithm. The term ‘average’ in Table 2 indicates the average of the quality of all chromosomes, and the ‘best’ term in Table 2 denotes the average of the best chromosome’s quality. Meanwhile, the parameters of 0.1~0.9 in Table 2 mean the value of the vigilance test of the ART neural network. The experiments are all performed over 10 times. The parameters, 0.65, 0.75, and so on, in the above experiment are the thresholds of STGA. Table 2 shows the results that STGA uses approach 1 and 2 to generate its schemata, where a bit in a schemata is assigned to be ‘1’ if its statistic average is over the designated threshold. The genetic operations are the same as PGA’s. The operations of the other approaches in the experiments are the same as for PGA1.

Table 2. The results of maximizing *f*(*x*), Eq. (1), by using the traditional GA, PGA, RGA, HGA and SOES. The schemata of PGA, RGA, and HGA are generated by the method of STGA, 'Average' and 'Best' show the quality of chromosomes.

V/T	Tradition GA		PGA				RGA				HGA				SOES	
			App. #1		App. #2		App. #1		App. #2		App. #1		App. #2			
—	Ave.	Best	Ave.	Best	Ave.	Best	Ave.	Best	Ave.	Best	Ave.	Best	Ave.	Best	Ave.	Best
——	2.406	2.966	——	——	——	——	——	——	——	——	——	——	——	——	——	——
0.3 /——	——	——	——	——	——	——	——	——	——	——	——	——	——	——	2.658 /——	2.98 /——
0.6 /0.65	——	——	—— /2.490	—— /2.976	—— /2.581	—— /2.970	—— /2.619	—— /2.981	—— /2.515	—— /2.982	—— /1.610	—— /2.924	—— /1.438	—— /2.953	2.661 /——	2.978 /——
0.7 /0.75	——	——	—— /2.491	—— /2.977	—— /2.591	—— /2.981	—— /2.535	—— /2.982	—— /2.451	—— /2.981	—— /1.755	—— /2.976	—— /1.602	—— /2.978	2.633 /——	2.979 /——
0.8 /0.85	——	——	—— /2.414	—— /2.975	—— /2.591	—— /2.981	—— /2.535	—— /2.982	—— /2.465	—— /2.982	—— /1.667	—— /2.981	—— /1.933	—— /2.980	2.649 /——	2.981 /——
0.9 /0.95	——	——	—— /2.533	—— /2.980	—— /2.566	—— /2.980	—— /2.513	—— /2.982	—— /2.486	—— /2.982	—— /2.257	—— /2.981	—— /2.463	—— /2.980	2.668 /——	2.980 /——

The experiment is carried out to test the performance of the simulated annealing algorithm in the proposed algorithm, SOES. The same objective function as shown in the last section is adopted to examine the proposed method. Additionally, the parameters of the genetic algorithms are the same as in the previous experiment. The results of the experiment are shown in Table 3.

Table 3. The results of maximizing *f*(*x*), Eq. (1), by using the traditional GA, PGA, RGA, HGA and SOES.

Vigilance	Traditional GA		PGA		RGA		HGA		SOES	
	Average	Best	Average	Best	Average	Best	Average	Best	Average	Best
——	2.405889	2.9662353	——	——	——	——	——	——	——	——
0.1	——	——	2.480081	2.9812967	2.5269624	2.9817360	2.2396543	2.9726216	2.6576016	2.9804351
0.2	——	——	2.536199	2.9798520	2.5243470	2.9806032	2.1095119	2.9725703	2.6433140	2.9803856
0.3	——	——	2.566205	2.9789696	2.5474174	2.9816183	2.1304252	2.9796342	2.6502784	2.9811825
0.4	——	——	2.475651	2.9804562	2.5215261	2.9805557	1.9378338	2.9706275	2.6515274	2.9806199
0.5	——	——	2.492586	2.9801924	2.5274105	2.9816813	1.7094627	2.9687090	2.6181309	2.9810562
0.6	——	——	2.600610	2.9809402	2.5598003	2.9811106	1.8695396	2.9672863	2.6608574	2.9777749
0.7	——	——	2.572158	2.9789543	2.5065388	2.9816288	1.7570251	2.9059928	2.6331791	2.9791753
0.8	——	——	2.542021	2.9793401	2.5257237	2.9811246	1.6960438	2.9447752	2.6494052	2.9807703
0.9	——	——	2.633211	2.9796651	2.5351319	2.9813035	2.0019642	2.9575521	2.6683167	2.9802559

The other experiment is designed to demonstrate the performance of the simulated annealing algorithm of SOES. The experiment shows that the performance of SOES, which uses a simulated annealing algorithm in reproduction, crossover, and mutation operations, is better than the other GAs. The results of these genetic algorithm approaches to maximize the function *f*(*x*) are list in Table 2. The best results of each GAs in this experiment are graphed in Fig. 4. In Fig. 4, traditional GA means GA without guiding by schemata; RGA denotes GA with guiding by schemata, which was generated by the method of ART neural network. Fig. 5 shows the power of guiding by simulated annealing algorithm. In this experiment, TGA means GA without guiding simulated annealing algorithm; SOES represents GAs with simulated annealing algorithm. All schemata in this experiment are generated by ART neural network.

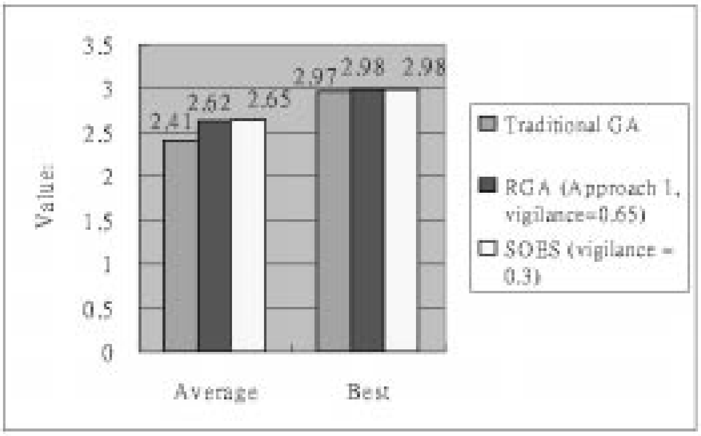

Fig. 4. The value indicates the quality of chromosomes of three GA shown in Table 2 to maximize *f(x)* in Eq. (1).

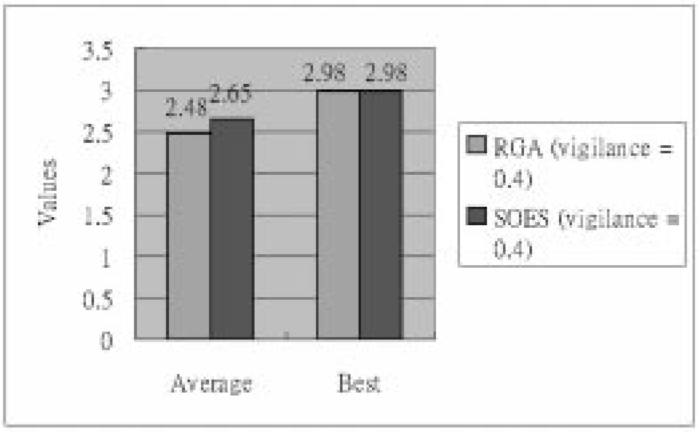

Fig. 5. The value indicates the quality of chromosomes of two GA shown in Table 2 to maximize *f(x)* in Eq. (1).

From the result of Fig. 4, it is observed that the performance of SOES is much better than traditional GA or the other algorithms whose schemata are generated by the method of STGA. The results located above 'average' and 'best' show that SOES has a faster convergence than the traditional GA of other genetic algorithms in the experiment. From the results, it can be seen that SOES has better capability to escape from local minima than the others; hence the ART generator can generate useful schemata for genetic algorithm, better even than the method of STGA.

The results of the experiment demonstrate the proposed mechanisms of generating schemata and show the simulated annealing algorithm can help the genetic algorithm converge faster and be more able to escape from local minima. Obviously, the simulated annealing algorithm is better than other genetic operations, such as the reproduction of roulette-wheel selection, the reproduction of remainder stochastic sampling without replacement, and the reproduction, crossover or mutation operations that are guided by schemata. In addition, SOES has a faster convergence speed and a better capability to escape from local minima than the other GA in the above experiment. It demonstrates that the genetic algorithm with the simulated annealing algorithm is much better than the genetic algorithm with other genetic operations as mentioned in the above experiment.

4. CONCLUSIONS

The proposed algorithm with the schemata and SA methods has some excellent characteristics, such as rapid convergence speed and the ability to escape from local minima. The experiments on an optimization problem are conducted to compare the performance of the proposed SOES with other genetic algorithms. According to the results of the experiments, the proposed SOES has better performance than the other genetic algorithms, such as the traditional GA and so-called STGA. However, these excellent results are obtained from experiments with an artificial experimental configuration. Therefore, future work will concentrate on applying SOES to some other practical applications, such as nonlinear, multi-constraints, combinatorial optimization problems that are difficult for traditional methods. In the experiments, most of the computation time of SOES is spent in generating schemata. Therefore, in future research we should seek an algorithm for obtaining the schemata more quickly.

REFERENCES

1. J. H. Holland, "Outline for a logical theory of adaptive system," *Journal of Association for Computing Machine*, Vol. 3, No. 1, 1962, pp. 297-314.
2. J. H. Holland, *Adaptation in Natural and Artificial Systems*, Ann Arbor: The University of Michigan Press, 1975.
3. D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, 1989.
4. L. Davis, *Handbook of Genetic Algorithms*, Van Nostrand, 1991.
5. J. Lis and A. E. Eliben, "A multi-sexual genetic algorithm for multiobjective optimization," in *Proceedings of IEEE International Conference on Evolutionary Computation*, 1997, pp. 59-64.
6. G. B. Sheble and K. Brittig, "Refined genetic algorithm-economic dispatch example," *IEEE Transactions on Power Systems,* Vol. 10, No. 1, 1995, pp. 117-124.
7. C. I. Marrison and R. F. Stengel, "Robust control system design using random search and genetic algorithms," *IEEE Transactions on Automatic Control*, Vol. 42, No. 6, 1997, pp. 835-839.

8. T. T. Maifeld and G. B. Sheble, "Genetic-based unit commitment algorithm," *IEEE Transactions on Power Systems*, Vol. 11, No. 3, 1996, pp. 1359-1370.
9. H. Vafaie and D. Jong, "Feature space transformation using genetic algorithm," *IEEE Transactions on Intelligence Systems*, Vol. 13, No. 2, 1998, pp. 57-65.
10. M. Sonka, S. K. Tadikonda, and S. M. Collins, "Knowledge-based interpretation of MR brain image," *IEEE Transactions on Medical Image*, Vol. 15, No. 4, 1996, pp. 443-452.
11. X. Ma, A. E. Keib, R. E. Smith and H. Ma, "A genetic algorithm based approach to thermal unit commitment of electric power systems," *Electric Power System Research*, Vol. 34, No. 1, 1995, pp. 29-36.
12. S. O. Orero and M. R. Irving, "A genetic algorithm for generator scheduling in power systems," *Electric Power & Energy Systems*, Vol. 18, No. 1, 1996, pp. 19-26.
13. D. Dasgupta and D. R. Megregor, "Thermal unit commitment using genetic algorithms," in *Proceedings of IEEE International Conference on General Transmission Distribution*, 1994, pp. 459-465.
14. B. P. Buckles, F. E. Petry and R. L. Kuester, "Schema survival rates and heuristic search in genetic algorithms," in *Proceedings of IEEE International Conference on Tools for Artificial Intelligence*, 1990, pp. 322-327.
15. T. N. Bui and B. P. Moon, "Genetic algorithm and graph partitioning," *IEEE Transactions on Computers*, Vol. 45, No. 7, 1996, pp. 841-855.
16. A. Agapie, and H. Dediu, "GA for deceptive problems: inverting schemata by a statistical approach," in *Proceedings of IEE International Conference on Evolutionary Computation*, 1996, pp. 336-340.
17. C. T. Lin, and G. Lee, *Neural Fuzzy Systems: a Neuro-Fuzzy Synergism to Intelligent Systems*, Prentice-Hall, 1996.
18. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, Vol. 220, No. 2, 1983, pp. 671-680.
19. K. Nara, A. Kitagawa, and T. Ishihara, "Implementation of genetic algorithm for distribution systems loss minimum reconfiguration," *IEEE Transactions on Power Systems*, Vol. 7, No. 3, 1992, pp. 1044-1051.
20. G. Boon and H. D. Chiang, "Optimal capacitor placement in distribution systems by genetic algorithm," *Electric Power and Energy System*, Vol. 15, 1993, pp. 155-163.
21. H. Chen N. S. Flann, and D. W. Watson, "Parallel genetic simulated annealing: a massively parallel SIMD algorithm," *IEEE Transactions on Parallel and Distributed Systems*, Vol. 9. No. 2, 1998, pp. 126-136.
22. R. Battiti and G. Tecchiolli, "Training neural nets with the reactive tabu search," *IEEE Transactions on Neural Networks*, Vol. 6, No. 5, 1995, pp. 1185-1200.

Kao-Shing Hwang (黃國勝) is an associate professor of Electrical Engineering Department at National Chung Cheng University, Taiwan. He received the B. S. Degree in Industrial Design from National Cheng Kung University, Taiwan, in 1981, and the M.M.E. and Ph.D. degrees in EECS from Northwestern University, Evanston, I.L., U.S.A., in 1989 and 1993, respectively. He was a design engineer at the SANYO Electric Co., Taipei, Taiwan during 1983-1985. Between 1987 and 1988, he joined the C & D Microsystem Co., Plastow, N. H., as a system programmer. His job involved designing PC card drivers, graphic animation. Since August 1993, he has been with National Chung

Cheng University, Taiwan. He is also the director of the Information Management Division at the university computer center. His areas of interest are neural networks and learning control, robotic compliance, and collision avoidance.

Jeng-Yih Chiou (邱正毅) was born in Taipei, Taiwan. He received a B.S. in Electrical Engineering from Tamkang University, Taiwan, in 1993, and M.S. from National Chung Cheng University, Taiwan, in 1996. He is currently a Ph.D. student in the electrical engineering department of National Chung Cheng University. His areas of interest are neural networks and learning control.

Yuan-Pao Hsu (徐元寶) is a lecturer of Information Engineering Department at National Penghu Institute of Technology, Taiwan. He received the B. S. and M. S. degrees in Electronic Engineering from National Taiwan University of Science and Technology, Taiwan, in 1987 and 1989. He is currently a Ph.D. student in the electrical engineering department of National Chung Cheng University. His research areas are hardware realization of neural networks, controller design, and chip design.