International Journal of Fuzzy Logic and Intelligent Systems, vol. 3, no. 2, December 2003 pp. 233-238

Species Adaptation Evolutionary Algorithm for Solving the Optimization Problems

Dong Wook Lee and Kwee-Bo Sim

School of Electrical and Electronic Engineering, Chung-Ang University

Abstract

Living creatures maintain their variety through speciation, which helps them to have more fitness for an environment. So evolutionary algorithm based on biological evolution must maintain variety in order to adapt to its environment. In this paper, we utilize the concept of speciation. Each individual of population creates their offsprings using mutation, and next generation consists of them. Each individual explores search space determined by mutation. Useful search space is extended by differentiation, then population explorers whole search space very effectively. If evolvable hardware evolves through mutation, it is useful way to explorer search space because of less varying inner structure. We verify the effectiveness of the proposed method by applying it to two optimization problems.

Key words : species adaptation, genetic algorithms, optimization problem, mutation

Ⅰ. Introduction

The genetic algorithms (GAs) [1-3] based on natural selection have been studied widely as a solution of the intelligent information processing system. It is a sort of population-based optimization method. Although GAs provide many opportunities to obtain a global optimal solution, the performance of GAs is more or less limited in some cases. Two typical cases are the optimization of multimodal functions and covering problems. In the former case GAs may concentrate effort on some suboptimal peak. In the latter case, the GAs user wishes to simultaneously find a number of peaks, but the GAs have lost all diversity and concentrates on one peak only.

In nature, the living creatures maintain their variety through speciation. There are many kinds of species in nature and each species adapts to its environment. A niche can be viewed as a subspace in the environment that can support different types of life. A species is defined as a group of individuals with similar biological features capable of interbreeding among themselves but that are unable to breed with individuals outside their group. So in nature the reason of being various species is speciation which new species is differentiated in order to adapt a niche [4].

GAs based on biological evolution must maintain variety in order to find feasible solutions. In this point of view, several niching (or speciation) methods have been proposed. Niching methods maintain population diversity and permit GAs to investigate many peaks in parallel.

Goldberg and Richardson proposed fitness sharing [2] that is the best known and also used among niching techniques. DeJong proposed crowding method [5] that insert new elements in the population by replacing similar elements. Mahfoud improved standard crowding [6] by introducing competition between offsprings and parents of identical niches. Harik developed restricted tournament selection [7] for multimodal optimization. Darwen and Yao proposed speciation with implicit fitness sharing [8]. Harvey propose species adaptation genetic algorithm (SAGA) [9] for evolutionary robotics.

In this paper, we propose species adaptation evolutionary algorithm (SAEA) based on both mutation and search space differentiation. GAs have two major operator, crossover and mutation, for maintaining variety. In this paper, however, we don't use crossover operator for applying it to evolvable hardware. So, In our system mutation helps population to maintain variety, and search space differentiation helps them to explore search space effectively.

The hardware that the structure can be changed automatically by evolutionary algorithm (EA) is called evolvable hardware (EHW) [10]. The appearance of field programmable gate array (FPGA) make it possible that hardware evolves. We regard the bit string of the hardware structure as a chromosome of EA and evolve the hardware structure based on its fitness. However the fitness landscape of EHW is very rugged and has many local peaks. So traditional GAs are not suitable for this problem. While niching or speciation methods are more effective to solve this problem. In EHW, when bit string as an individual in EA is mutated, it can re-configure its structure of mutated area [10]. However much part of structure must be changed in the case of crossover. Also crossover cannot preserve the characteristics of each individual. So, mutation is more effective search than crossover. Instead of crossover operation, search space differentiation make it possible that SAEA explores whole search space and doesn't fall into local optima.

GAs and its major operator, mutation are explained in

Manuscript received November 11, 2003 ; revised November 31, 2003.
This research was supported by the Braintech 21 from Ministry of Science and Technology of Korea.

Section 2. Proposed algorithm, species adaptation evolutionary algorithm (SAEA), is explained in Section 3. This section consists of two part that are SAEA and individual exploration and differentiation. In Section 4, simulation results of two optimization problems using SAEA are shown. Finally, conclusions are given in Section 5.

II. Genetic Algorithms

2.1 GAs

GAs [1-3] based on natural selection have been studied widely as a solution of the intelligent information processing system. It was proposed by Holland [1] as a computational model of living system's evolutionary process and has become popular as a population-based optimization method. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search [2]. In every generation, new artificial creatures are created using bits and pieces of the fittest of the old. Also new part often helps artificial creature to fit for the environment. Because it is population-based optimization method, GAs have robustness for survival in many different environments. However even GAs provide many opportunities to obtain a global optimal solution, the performance of GAs is more or less limited depending on the fitness landscape of function [11].

2.2 Mutation

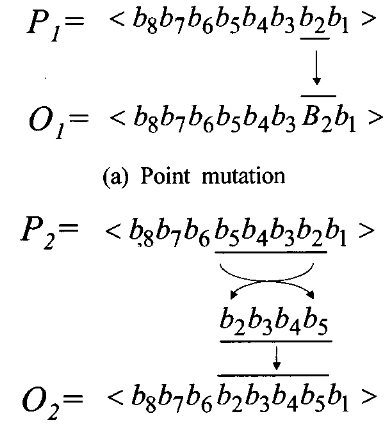
Major operators of GAs are crossover and mutation. Crossover creates new individual by exchanging a part of chromosome which of selected individual. So it is major operator in GAs. It selects randomly two individuals of population, and creates two new individuals by exchanging genes at selected point. However, because the parents create offsprings depended on them through crossover operation, crossover operation can have limited performance by chromosome structure and coding method.

Mutation, another major operator, exchanges the value in some locus of gene for allele. It causes structural differentiation and maintains variety of population. There are point mutation, inversion, translation, insertion, and so on.

Point mutation is accomplished by replacing the gene of some locus of each individual with allele under specified probability p_m. For example, in Fig. 1 (a) after the parent's bit string $p_1=\langle b_n b_{n-1} \cdots b_2 b_1 \rangle$ is mutated at second locus of gene, the offspring's bit string $o_1=\langle b_n b_{n-1} \cdots B_2 b_1 \rangle$ is created.

Inversion is an operator that inverses bit-string which is determined by selecting two random points based on specified probability p_i. For example, in Fig. 1 (b) after one random points a $(1 \le a < n)$ and the length of inversing string l $(l \le n-a)$ are determined, a part of bit string is inversed from point a to point $(a+l-1)$.

Duplication duplicates some part of chromosome. Addition inserts some part of chromosome and the length of chromosome increases. Also there are other mutation operators. In our system, we use point mutation and inversion as major operators.



$P_1 = \langle b_8 b_7 b_6 b_5 b_4 b_3 b_2 b_1 \rangle$
$O_1 = \langle b_8 b_7 b_6 b_5 b_4 b_3 B_2 b_1 \rangle$
(a) Point mutation
$P_2 = \langle b_8 b_7 b_6 b_5 b_4 b_3 b_2 b_1 \rangle$
$b_2 b_3 b_4 b_5$
$O_2 = \langle b_8 b_7 b_6 b_2 b_3 b_4 b_5 b_1 \rangle$

(b) Inversion ($a=2$, $l=4$)

Fig. 1 Process of mutation operator

In the case of EHW, its inner structure is represented by bit string [10]. This bit string is used as a chromosome in order to evolving EHW by using GA. If bit string is mutated by mutation operation such as inversion, point mutation, or duplication, inner structure of EHW is changed in structural variation.

III. Species Adaptation Evolutionary Algorithm

3.1 Species Adaptation Evolutionary Algorithm

In this paper, we propose species adaptation evolutionary algorithm (SAEA). This method uses nature's property. In nature, the living creatures maintain their variation by speciation. In SAEA, the major operator is mutation, and each individual of population evolves respectively.

The first stage of SAEA is speciation process. In the first stage, initial population is constructed by creating the number N_1 of individual randomly. Each of N_1 individuals is mutated, and then the number M_1 of offsprings is obtained. Therefore $N_1 \times M_1$ individual is obtained in intermediate s.

The number S $(S < M_1)$ of offsprings of the number M_1 of offsprings consists of next generation. The number S is speciation parameter and represents species differentiation number. In selecting the number S of offsprings, best offspring (s_1) is selected and the others (the number $S-1$ of offsprings) are selected by selectiveness $sel(s_i)$ given by (1). Selectiveness is proportional to both fitness and hamming distance from best offspring.

$$sel(s_i) = a \cdot H(s_1, s_i) \cdot fit(s_i) \quad (1)$$

where $H(s_1, s_i)$ is hamming distance between best s_1 and selected offspring s_i, $fit(s_i)$ is fitness value of s_i, a is a proportional constant. This process is repeated until sufficient species are obtained.

To maintain the computational cost as same as the cost of first generation. We control the number of offsprings (M_i) of individual. Fitness evaluation times of first generation is $N_1 \times M_1$ and that of second generation is $N_2 \times M_2 = N_1 \times S \times M_2$. therefore the number of offsprings created by mutation from a individual is determined by (2)

$$M_{i+1} = \frac{M_i}{S} \quad (2)$$

where M_{i+1} is the number of offsprings created from each individual of the $(i+1)$-th generation, M_i is the number of sons created from each individual of the i-th generation, and S is the number of differentiation.

If $M_i < S$ then $M_i = S$. Until the population size becomes specific size, the speciation process do last. In this process, each individual of population is differentiated into some number of species. If the population size becomes specific size, differentiation is stopped and each individual explores limited search space using mutation operator.

The second stage is exploring process using determined speices. In second stage each individual creates the number S of offsprings. Then best individual among the parent and the offsprings is selected into a parent of next generation. This selection method is same as $(1+S)$-selection of evolution strategies [3].

In the first stage, each individual is considered as one species. One individual, one species, become species in next generation. So, in the first stage, individual extensions search space through differentiation. Also if the population size is settled down, evolution proceeds to the second stage. In the second stage, individual concentrates on exploring useful search space that individual in first stage explore. Fig. 2 shows the conceptional diagram of speciation process.

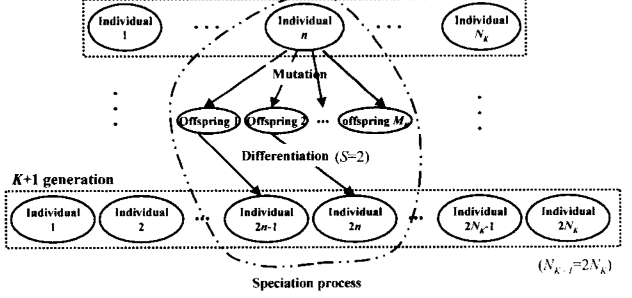


Fig. 2 Diagram of speciation process

3.2 Individual exploration and differentiation

In traditional GAs, an individual survives by fitness value depend on whole individual of population. So individuals with high fitness are increased rapidly in population. This is called "premature convergence." Because of this phenomenon, the diversity of population is decreased. In order to prevent this phenomenon, we utilize individual exploration and differentiation.

Each individual creates offsprings using mutation. These offsprings survived in proportion to the fitness and hamming distance from best offspring. Because each species is independent each other. each individual (species) is created in parallel [12, 13]. In the case of bit string, hamming distance is calculated by (3)

$$H(P_1, P_2) = \sum_{i=1}^{N} |b_{1i} - b_{2i}| \quad (3)$$

where $H(P_1, P_2)$ is hamming distance between P_1 and P_2, P_1 and P_2 are individual of population, and b_{1i} is the i-th bit of P_1, and b_{2i} is the i-th bit of P_2.

When a parent creates an offspring through mutation operation, the range of hamming distance is determined based on the probability of mutation. If an offspring created by mutation maintain slow fitness continuously, the probability of mutation must be increased and then search space is extended. In order to explore whole search space effectively, in the first stage of evolution, it is needed to maintain that the number of individuals of population is small relatively and the number of offsprings is big. By this setting up, useful individuals are differentiated from parent individuals.

An offspring individual in certain hamming distance is created by mutation. Some of these are differentiated and consists of next generation. These individuals are determined by hamming distance and fitness. Each individual explores search space determined by hamming distance depend on mutation operation, and search space is differentiated by differentiation of individual. So whole search space is explored effectively.

Fig. 3 is the conceptional diagram of individual exploration and differentiation. In this case, the length of bit string is 10 and hamming distance determined by mutation is two.

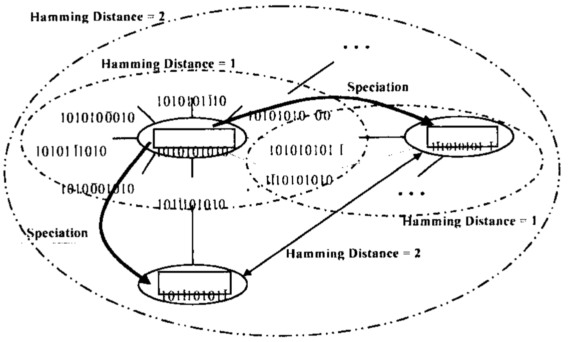


Fig. 3 Individual exploration and differentiation

IV. Simulation Results

4.1 Results of Two Optimization Problem

In order to verify the performance of the SAEA, we consider the optimization problem of a simple function of one variable. The function [3] is defined as

$$f_1(x) = x\sin(10\pi x) + 2.0 \tag{4}$$

and is drawn in Fig. 4. The problem is to find x from the range $[-1.0, 2.0]$ which maximizes the function f_1, i.e., to find x_0 such that $f(x_0) \ge f(x)$, for all $x \in [-1.0, 2.0]$.

Fitness evaluation function $f_1(x)$ is GA-Easy problem so optimal solution is to obtain easily using simple GA. In this problem, parameters of two algorithms are the same as Table 1. Fig. 5 shows the fitness change of GA and SAEA. This is a result of a typical run in simulation. Each algorithm executes 50 times. In the case of GA, the best solution has fitness value 0.9999. However, in the case of species adaptive adaptation evolutionary, optimal solution that has fitness value 1.0 is obtained in all simulation. Though proposed algorithm uses only mutation operation, it reaches to optimal solution all 50 times within 10 generation. As the results, proposed algorithm is more useful than GA even in GA-Easy problem.

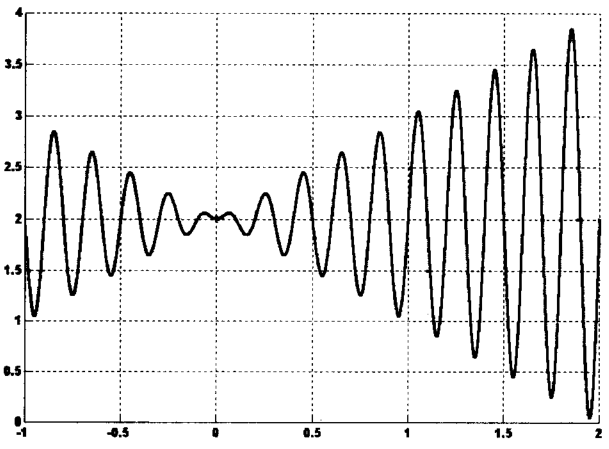


Fig. 4 Graph of the function $f_1(x)$

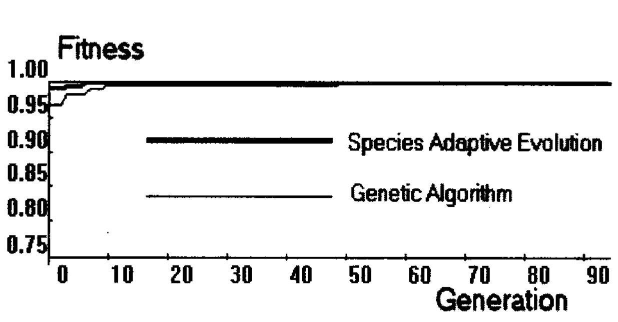


Fig. 5 Best fitness of both GA and SAEA in GA-Easy problem.

Table 1. Parameters of each algorithm

(a) Parameters of genetic algorithm

Parameter of genetic algorithms	
Length of bit string	30
Population size (N)	80
Probability of crossover (p_c)	0.6
Probability of mutation (p_m)	0.01
Generation	150

(b) Parameters of species adaptation evolutionary algorithm

Parameter of species adaptation evolutionary algorithm	
Length of bit string	30
Initial Population size (N_1)	10
Final Population size (N_f)	40
Number of differentiation (S)	2
Probability of mutation (p_m)	0.1
Generation	150

The other test function is a false peak function [14] that has several false peaks. This function is defined as

$$f_2(x) = \max[X_1, X_2] \tag{5}$$

where $X_1 = \sqrt{\dfrac{x_1^2 + \cdots + x_N^2}{N}}$ and

$X_2 = \sqrt{\dfrac{x_1^2 + (1-x_1)^2 + \cdots + (1-x_N)^2}{N+1}}$.

Fitness evaluation function $f_2(x)$ is false peaks problem, and it is so difficult for simple genetic algorithm to obtain optimal solution because of too many local maxima [14]. Because it is difficult to visualize 30-boolean variable false-peak function in limited area, we plot the landscape of 10-boolean variable false-peaks function instead of that in Fig 6. Here, the horizontal axis is the decimal number of the binary string and vertical axis is its fitness value. As shown in Fig. 6, there is one optimal solution which are all 1's. However it is easy to see that there are several false local optima including all 0's. These features imply that worst solutions have a greater chance of being mutated into optimal solutions and that better solutions are prone to be mutated into local optima.

In this problem, parameters of two algorithms are set such as table 1. The Fig. 7 shows best individual which of most good result through 50 simulations. Proposed algorithm seeks best individual that has fitness 1.0 in 46 times. However GA seeks only best individual that has fitness 1.0 in 25 times, the other case fall into local maximum. Also Table 2 shows SAEA has shorter the average generation of finding solution in successful runs than GA.

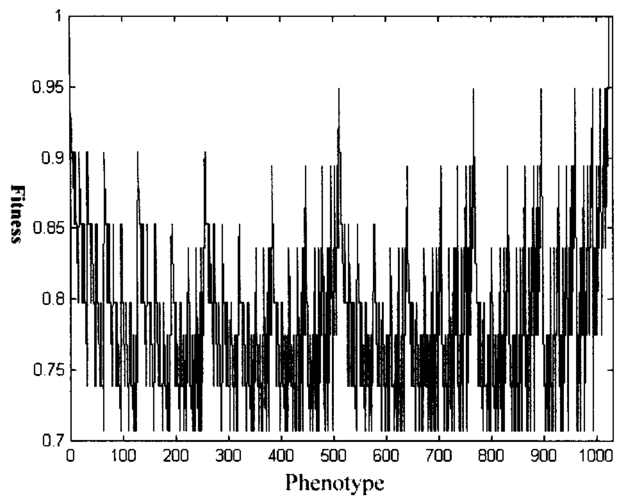


Fig. 6. Graph of the function $f_2(x)$

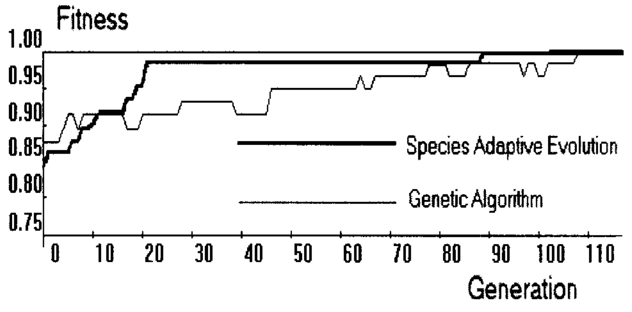


Fig. 7. Best Fitness of both GA and SAEA in false-peak problem.

Table 2. Simulation results

function		GA	SAEA
f_1	number of success	33 (50)	50 (50)
	avg. gen. of success	10 (/33)	9 (/50)
f_2	number of success	25 (50)	46 (50)
	avg. gen. of success	130 (/25)	110 (/49)

Through the simulation of two evaluation functions, proposed algorithm changes the chromosome by mutation and differentiation effectively. Then it is not easy to fall into local optimum while it is differentiated into useful search space. In SAEA, only mutation operator is used, so it spends less time to evolution than GA. Also in first generation there are many kinds of species, so individuals of the population have either low fitness or high fitness. However most individuals of the population evolved by GA have high fitness value. This explains the population evolved by proposed algorithm is more fit for various natures.

If the proposed algorithm is adapted for evolvable hardware, it will be useful method. When only mutation operator is used for evolving evolvable hardware, only mutated part of inner structure is changed. It spends short time to evolve. Also population size in proposed algorithm is smaller than GA, so it is more effective.

V. Conclusion

In this paper, we propose species adaptation evolutionary algorithm (SAEA) for realization of evolvable hardware using both mutation and differentiation of search space. This method is modeled on biological evolution through speciation in nature. Also it is easy to be realized and applied but the population size must be controlled because of increasing size by differentiation. We verify the effectiveness of proposed algorithm through simulations, applying it to solve two function optimization problems. In these simulations, proposed algorithm finds best individual effectively. Also, if this method using mutation adapts to evolvable hardware, the speed of evolution is going to be increased.

There are still many issues to be addressed. One of them is to schedule strategy of probability of mutation. This scheduling plays an important role in differentiation and evolution for the population. Another issue is effective methodology of differentiation.

References

[1] J. H. Holland, *Adaptation Natural and Artificial Systems*, Ann Arbor, University of Michigan Press, 1975.

[2] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addision-Wesley, 1989.

[3] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Verlag, 1995.

[4] B. Sareni, L. Krahenbuhl, "Fitness sharing and niching methods revisited," *IEEE Trans. on Evolutionary Computation*, vol. 2, no. 3, pp. 97-106, 1998.

[5] K. A. DeJong, "An analysis of the behavior of a class of genetic adaptative systems," *Ph.D. dissertation*, Univ. of Michigan, Ann Arbor, 1975.

[6] S. W. Mahfoud, "Niching methods for genetic algorithm," *Ph.D. dissertation*, Univ. of Illinois, Urbana-Champaign, 1995.

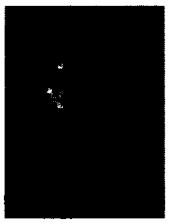
[7] G. Harik, "Finding multimodal solutions using restricted tournament selection," *Proc. of 1996 IEEE Int. Conf. Genetic Algorithm*, pp. 24-31, 1995.

[8] P. J. Darwen, X. Yao. "Speciation as automatic categorical modularization," *IEEE Trans. on Evolutionary Computation*, vol. 1, no. 2, pp. 101-108, 1997.

[9] I. Harvey, "Evolutionary robotics and SAGA: the case for hill crawling and tournament selection," *Artificial Life III*, Addison Wesley, 1993.

[10] A. Thompson, "An evolved circuit, intrinsic in silicon, entwined with physics," *Evolvable Systems: From Biology*

to Hardware, no. 1259 of *LNCS*, pp. 390-405, 1997.
[11] J.R. Koza, "Genetic evolution and co-evolution of computer programs," *Artificial Life II*, Addison Wesley, 1991.
[12] H. Tamaki, and Y. Nishikawa, "A paralleled genetic algorithm based on a neighborhood model and its application to the job shop scheduling," *Parallel Problem Solving from Nature 2*, Elsevier Science Publishers, 1992.
[13] M. G. Schleuter,, "ASPARAGOS: An asynchronous parallel genetic optimization strategy," *Proc. 3rd Int. Conf. Genetic Algorithms*, Morgan Kaufman,. 1989.
[14] K. A. DeJong, and W. M. Spears, "Using genetic algorithms to solve np-complete problems," *Proc. of the 3rd ICGA*, pp. 124-132, 1989.



Dong-Wook Lee
Dong-Wook Lee received his B.S., M.S., and Ph.D. degrees in the Department of Control and Instrumentation Engineering from Chung-Ang University in 1996, 1998, and 2000, respectively. Since 2002, he has been with the Information and Telecommunication Research Institute at Chung-Ang University, where he is currently a Research Professor. His areas of interest include Artificial Life, Evolutionary Computation, Artificial Brain, and Artificial Immune Systems. He is a member of KITE, KIEE, ICASE, and KFIS.
E-mail : dwlee@wm.cau.ac.kr

Kwee-Bo Sim
Kwee-Bo Sim received his B.S. and M.S. degrees in the Department of Electronic Engineering from Chung-Ang University in 1984 and 1986 respectively, and Ph.D. degree in the Department of Electrical Engineering from The University of Tokyo, Japan, in 1990. Since 1991, he has been a faculty member of the School of Electrical and Electronic Engineering at Chung-Ang University, where he is currently a Professor. His areas of interest include Artificial Life, Intelligent Robot, Intelligent Systems, Artificial Brain, Multi-Agent System, Distributed Autonomous Robotic System, Machine Learning, and Adaptation Algorithm, Soft Computing (Neuro, Fuzzy, Evolutionary Computation), Evolvable Hardware, Artificial Immune System, Intrusion Detection System. He is a member of IEEE, SICE, RSJ, KITE, KIEE, ICASE, and KFIS.

Phone : +82-2-820-5319
Fax : +82-2-817-0553
E-mail : kbsim@cau.ac.kr