# A Comparison Study between Genetic Algorithms and Bayesian Optimize Algorithms by Novel Indices

Naoki Mori
College of Engineering
Osaka Prefecture University
1-1 Gakuencho, Sakai city
Osaka 599-8531, JAPAN
mori@cs.osakafu-u.ac.jp

Masayuki Takeda
College of Engineering
Osaka Prefecture University
1-1 Gakuencho, Sakai city
Osaka 599-8531, JAPAN
takeda@cs.osakafu-u.ac.jp

Keinosuke Matsumoto
College of Engineering
Osaka Prefecture University
1-1 Gakuencho, Sakai city
Osaka 599-8531, JAPAN
matsu@cs.osakafu-u.ac.jp

## ABSTRACT

Genetic Algorithms (GAs) are a search and optimization technique based on the mechanism of evolution. Recently, another sort of population-based optimization method called Estimation of Distribution Algorithms (EDAs) have been proposed to solve the GA's defects. Although several comparison studies between GAs and EDAs have been made, little is known about differences of statistical features between them. In this paper, we propose new statistical indices which are based on the concepts of crossover and mutation, used in GAs, to analyze the behavior of the population based optimization techniques. We also show simple results of comparison studies between GAs and the Bayesian Optimization Algorithm (BOA), a well-known Estimation of Distribution Algorithms (EDAs).

## Categories and Subject Descriptors

F.2 [**Analysis of Algorithms and Problem Complexity**]: Miscellaneous

## General Terms

Algorithms

## Keywords

Genetic Algorithms, Bayesian Optimization Algorithms, Diversity, Population-based Optimization Methods

## 1. INTRODUCTION

Recently, a vast number of population based search and optimization algorithms have been proposed and applied successfully to many kinds of problems. One of the popular and excellent population based algorithms is the genetic algorithm (GA)[4, 7].


*GECCO'05,* June 25–29, 2005, Washington, DC, USA.


GAs are a search and optimization technique based on the mechanism of real life evolution. Dynamics of GAs have many similarities to many-particle systems in physics. One of the authors has proposed two novel GAs based on hints from physics called the ThermoDynamical Genetic Algorithm (TDGA)[13, 11, 9, 12] and the Spin Glass based Genetic Algorithm (SGGA)[5, 6].

Typically GAs utilize three operators called *mutation*, *crossover* and *selection*. In the GA search, new solutions are generated by crossover and mutation. The role of mutation is increasing diversity of the population by random perturbation. On the other hand, crossover produces new solutions by combining two solutions in the previous population. If several high-quality partial solutions are combined, we expect to obtain new high-quality solutions by crossover. High-quality partial solutions are often called *building blocks*[7].

Although GAs are a very powerful method, GAs sometimes fail to search when crossover works not well. Especially, when the structure of building blocks is complex, GA's defects are emphasized because crossover destroys the building blocks rather than combining them. To solve this problem of GA, another sort of population based optimization method called Estimation of Distribution Algorithms (EDAs)[14] have been proposed. The main difference between EDAs and GAs is that EDAs utilize not individuals in the population but global information to make a new population. EDAs make a distribution model from the selected population, and generate new solutions using this model. EDAs are expected to learn *the linkage of genes* and preserve building blocks better than GAs. Several methods of representing the distribution model have been proposed. We focus on the Bayesian Optimization Algorithm (BOA)[1, 2, 17, 18], a well-known EDA, which utilizes Bayesian networks to represent the joint distribution of genes.

The comparison study of dynamics of GAs and EDAs is very important and interesting. However little is known about differences of statistical features between them because there is no appropriate statistical indices to analyze the search process. In this paper, we propose new statistical indices, based on the concepts of crossover and mutation used in GAs, to analyze the behavior of population based optimization techniques. We also show some results of comparison studies between GAs and BOA taking a knapsack problem as an example.

## 2. GENETIC ALGORITHM

### 2.1 Thermodynamical Genetic Algorithm

In the selection operator used in conventional GAs, an individual having a good fitness value has more probability of being selected, and thus of yielding offspring in the next generation.

While it is a basic mechanism to find the optimal solution by focusing search on the neighborhood of good solutions, it also brings about the problem of premature convergence. To solve this problem, one of authors has proposed *the ThermoDynamical Genetic Algorithm* (TDGA)[13].

In the TDGA, the selection operation is designed to minimize the free energy of the population. The free energy $F$ is defined by:

$$F = \langle E \rangle - HT, \tag{1}$$

where $\langle E \rangle$ is the mean energy of the system, $H$ is the entropy and $T$ is a parameter called the temperature. It is known that the free energy $F$ is minimum in the Gibbs distribution, and this is called the principle of minimum free energy[3]. TDGAs introduce this principle to selection operation.

Minimization of the free energy can be interpreted as taking a balance between minimization of the energy function (the first term in the RHS of Eq. (1), or equivalently maximization of the fitness function in GAs by regarding $-E$ as the fitness function) and maintenance of the diversity measured by the entropy (the second term in the RHS of Eq. (1)). Hence, individuals having relatively small energy (or relatively large fitness) values will be given priorities to survive in the next generation. At the same time, individuals having rare genes will also be preserved owing to their contribution to minimization of the free energy via increase in the entropy term $HT$ of Eq. (1). Thus, the diversity of the population can be controlled by adjusting the temperature $T$ explicitly.

In Eq. (1), the energy function $E$ is easily given from the optimum function and the temperature $T$ is only a parameter, thus calculating these values does not represent a problem.

However, there is some problem to estimate entropy $H$. $H^{\mathrm{ALL}}$ which is entropy of the variety of individuals is obtained as follows:

$$H^{\mathrm{ALL}} = -\sum_i p_i \log p_i \tag{2}$$

where $p_i$ is the ratio of species $i$. Each genotype represents one species. If the chromosome is $M$-bit binary string, the number of species is equal to $2^M$. In GAs, the number of the population is extremely smaller than the possible species number, so using $H^{\mathrm{ALL}}$ directly is meaningless. Therefore we use another entropy $H^1$ which is calculated from each allele as follows:

$$H^1 = \sum_{k=1}^{M} H_k^1, \quad H_k^1 = -\sum_{j\in\{0,1\}} P_j^k \log P_j^k \tag{3}$$

where $H_k^1$ is entropy of the locus $k$ and $P_j^k$ is the existence probability of the gene $j$ on the locus $k$. $M$ is the chromosome length. One of authors proved that $H^{\mathrm{ALL}}$ in Eq. (2) and $H^1$ in Eq. (3) are equal when each locus's gene takes a value independently[13].

In order to minimize the free energy, we use an approximate greedy method which adds the individual making free energy minimum in the temporary generation into the next generation one after another[13].

### 2.2 Spin Glass based Genetic Algorithm

The authors have proposed the *Spin Glass based Genetic Algorithm*(SGGA)[5, 6], which utilizes the dynamics of spin glasses[15]. The basic structure of SGGA is Cellular GAs. Each individual is arranged at a lattice point of 2-D field.

In SGGAs, there exists interaction $J$ between the Neumann neighborhoods. Therefore, each lattice has 4 different $J$(up, down, left, right). Interaction between lattice $i$ and $j$ is represented by $J_{ij}$, and we set $J_{ij}$ be symmetrical with $i$ and $j$. The value of $J_{ij}$ is adjustable parameter in SGGA. Let $d(i,j)$ be the Hamming distance between an individual at a lattice $i$ and an individual at a lattice $j$, and $M$ be the chromosome length. We define the interaction energy $E_{ij}^{\mathrm{I}}$ as follows:

$$E_{ij}^{\mathrm{I}} = J_{ij} \cos(\frac{d(i,j)}{M}\pi) \tag{4}$$

SGGAs renew the individual by *the state transition*. In this procedure, *neighborhood energy* $E$ is utilized for the criteria. Let $f_i$ be the fitness of individual at a lattice $i$. $E$ is defined as follows:

$$E = -\sum_{i\in\mathcal{N}} f_i - \frac{1}{2}\sum_{i,j\in\mathcal{N}} E_{ij}^{\mathrm{I}} \quad (i \neq j) \tag{5}$$

where $\mathcal{N}$ represents the target neighborhood. The first term in the RHS of Eq. (5) achieves the maximization of the fitness and the second term in the RHS of Eq. (5) controls the diversity. Lower $E$ is more stable in SGGA. If all $J$ are positive, SGGAs become ferromagnetic type. On the other hand, all $J$ are negative, SGGAs become anti-ferromagnetic type. Generally, both positive $J$ and negative $J$ exist by mixture in SGGA in order to make a spin glass like state.

The genetic operators of SGGA are following.

#### *State transition*

*State transition* in SGGA is a operator like a selection in conventional GA. In this operation, the Metropolis method[8] which is typical SA's transition rule is utilized to realize the transition between two states. The probability of transition between two states $s$ and $s'$ is represented as follows:

$$\begin{cases} 1 & \text{if } E(\boldsymbol{s}') < E(\boldsymbol{s}) \\ \exp\left(\dfrac{-E(\boldsymbol{s}') + E(\boldsymbol{s})}{T}\right) & \text{otherwise} \end{cases}$$

where $E(s)$ represents the energy of state $s$ defined in Eq. (5). $T$ is the parameter called temperature.

#### *Crossover*

*Crossover* in SGGA is applied 2 individuals in the neighborhood. Original parents state and the smallest energy offsprings state are selected for transition. Figure 1 shows the outline of SGGA crossover.

#### *Mutation*

*Mutation* in SGGA is applied 1 individual. Original state before mutation and the state after mutation are selected for transition. Figure 2 shows the outline of SGGA crossover.

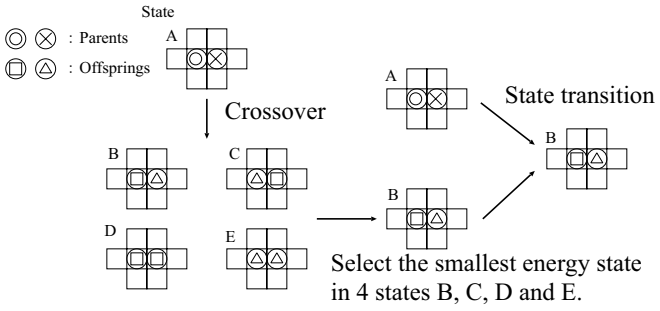


**Figure 1: Crossover of SGGA**

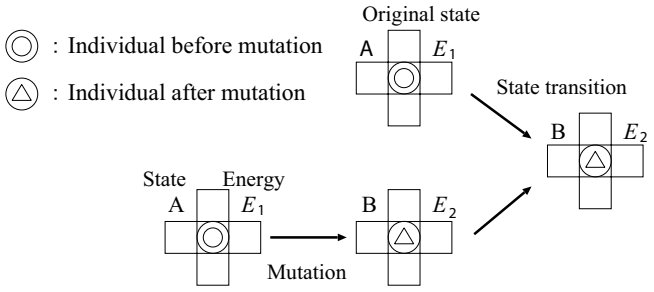


**Figure 2: Mutation of SGGA**

## 3. BAYESIAN OPTIMIZATION ALGORITHM

The Bayesian optimization algorithm (BOA)[1, 2, 17, 18] is one of the excellent population based search algorithm by using a global information about the set of promising solutions. The BOA is one of Estimation of Distribution Algorithms (EDAs)[14]. Figure 3 shows the outline of EDA.

The important feature of BOA is the way of representing a distribution model in Figure 3. BOAs utilize Bayesian networks to represent the distribution model. Figure 4 shows the example of a Bayesian network which represents a joint distribution of 4 genes $X_1 \sim X_4$.

The algorithm of the BOA follows:

1. Let $t = 0$ and generate the initial population randomly. Select appropriate values for the maximum number of generations $N_g$.

2. Select a set of promising individual $S(t)$ from $P(t)$.

3. Construct the Bayesian networks $B$ using a chosen metric and constraints.

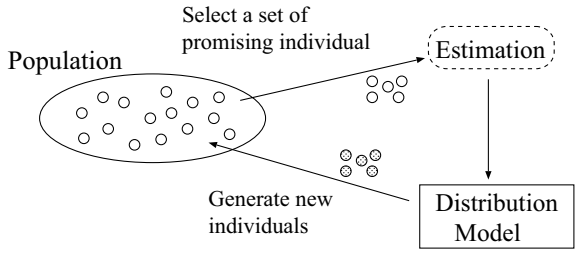


**Figure 3: Outline of EDA**

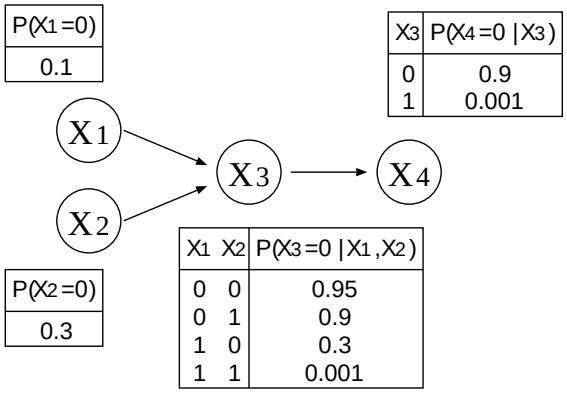


**Figure 4: Bayesian network of 4 genes**

4. Generate a set of new individual $O(t)$ according to the joint distribution encoded by $B$.

5. create a new population $P(t+1)$ by replacing some strings from $P(t)$ with $O(t)$.

6. Let $t = t + 1$. If $t < N_g$, go to Step 2, otherwise terminate the algorithm.

## 4. COMMON INDICES OF POPULATION BASED SEARCH ALGORITHMS

Several indices have been utilized to analyze dynamics of population-based search algorithms. The following two are the main indices in a GA analysis.

1. Search performance

2. Fitness information

One of the authors has proposed the diversity measure calculated by the entropy of each locus $H^1$ in Eq. (3)[10]. Besides these indices, following indices are used occasionally.

1. The number of different phenotypes

2. Entropy of the phenotype

3. The number of different genotypes

4. The number of different optimum solutions

5. The number of allele lost

6. The age information of individuals[9]

Indices above except search performance are obtained by the population in one generation. Since the state of previous population is unnecessary to calculate the indices above, we call these *the static indices*.

On the other hand, only a few indices which represent the relation between the current population and previous populations have been proposed although they are very important. We call such indices *the dynamic indices*.

In the next section, we propose several type of *the dynamic indices* in order to analyze the search dynamics deeply.

# 5. NEW INDICES OF POPULATION BASED SEARCH ALGORITHMS

## 5.1 Necessity of New Indices

The symbols used in this section are as follows:

| | | |
|---|---|---|
| $t$ | : | the generation number |
| $P(t)$ | : | population in generation $t$ |
| $i_t$ | : | individual $i$ in generation $t$ |
| $d(i,j)$ | : | The Hamming distance between individual $i$ and $j$ |
| $M$ | : | chromosome length |
| $g_k^x$ | : | gene of individual $x$ on locus $k$ |

To understand the search dynamics well, *the dynamic indices* is required. One dynamic index is *the cumulative number of genotypes*.

This index represents the total number of solutions which are sampled by a search algorithm. If the genotype which has been already selected past is sampled again, the cumulative number of genotypes does not increase. The cumulative number of genotypes shows the number of effective fitness evaluations. Information about $P(k),\ k = 0, 1, 2 \cdots t-1$ is necessary to calculate this index in $t$.

In addition, indices which represent the relation between $P(t)$ and $P(t-1)$ are also important because almost every population based-algorithm is a Markov process. We propose new statistical indices of this type in this paper.

## 5.2 Basic Concepts

First, we define several basic concepts.

*Definition 1.* Individual set generated by $i$ and $j$

The individual set generated by individual $i$ and $j$, $S_{i,j}$, is defined as follows:

$$S_{i,j} = \{x | g_k^x = g_k^i \text{ or } g_k^x = g_k^j,\ k = 1, 2, \cdots, M\} \quad (6)$$

That is, $S_{i,j}$ is the set of all possible individuals which are generated by uniform crossover between $i$ and $j$. We can easily extend this concept to $n$-parents cases.

If the Hamming distance between $i$ and $j$ is $d(i,j)$ and the chromosome is the binary-string, $|S_{i,j}| = 2^{d(i,j)}$.

*Definition 2.* Mutation length from $x$ to $i$ and $j$

The mutation length from $x$ to $i$ and $j$, $L_{i,j}^x$, is defined as follows:

$$L_{i,j}^x = \min_{k \in S_{i,j}} d(x,k) \quad (7)$$

where $d(x,k)$ represents the Hamming distance between $x$ and $k$. That is, $L_{i,j}^x$ is the Hamming distance between $x$ and the most similar individual to $x$ in $S_{i,j}$.

*Definition 3.* Twist frequency of $x$ to $i$ and $j$

Let $h$ be generated by crossover between $i$ and $j$. Next, check the gene of $h$ on each locus where genes are different between $i$ and $j$. Next, investigate which parent's gene was selected in $h$ on all loci. The locus where gene is the same between $i$ and $j$ is skipped. If a $h$'s gene on inspected locus is equal to $i$'s gene, record the symbol **i**. If $h$'s gene is equal to $j$'s gene, record the symbol **j** likewise. We can obtain the string as **iijiiji** by arranging each symbol from left to right. Next, count the number of changes **i** $\rightarrow$ **j** or **j** $\rightarrow$ **i**. The number is 4 in this example **iijiiji**. We define this number as the basic twist frequency of $h$ to $i$ and $j$.

If $x$ could not be produced by crossover between $i$ and $j$, the twist frequency of $x$ to $i$ and $j$ ($\xi_{i,j}^x$) is obtained as follows:

1. Find the most similar individual to $x$ in $S_{i,j}$. Let this individual be $x'$.
2. $\xi_{i,j}^x$ is the basic twist frequency of $x'$ to $i$ and $j$.
3. If there are several candidates for $x'$, select the $x'$ to minimize the basic twist frequency of $x'$ to $i$ and $j$.

If $i$ and $j$ are the identical, $\xi_{i,j}^x = 0$. If $i$ and $j$ are different and $h$ is generated by crossover between $i$ and $j$, the following equation is satisfied.

$$0 \leq \xi_{i,j}^x \leq d(i,j) - 1 \quad (8)$$

## 5.3 Mutation length and twist frequency of $x$ to $P(t)$

The mutation length of $x$ to $P(t)$, $L_{P(t)}^x$, is defined as follows:

$$L_{P(t)}^x = \min_{i,j \in P(t)} L_{i,j}^x \quad (9)$$

$L_{P(t)}^x$ is the distance between $x$ and the most similar individual in $\cup_{i,j \in P(t)} S(i,j)$.

The twist frequency of $x$ to $P(t)$ is defined as the twist frequency of $x$ to $i$ and $j$, where $i$ and $j$ are individuals that reach the minimum value defined in Eq. (9). If there are several candidates for $i$ and $j$, select $i$ and $j$ to minimize the twist frequency.

Finally, we obtain the mutation length and the twist frequency of $P(t)$ to $P(t-1)$ by taking the average of each value for all individuals in $P(t)$ to $P(t-1)$. We assume $i \neq j$ in all indices. We also define the generalized mutation rate by dividing the mutation length by the chromosome length. The proposed three indices, *the mutation length(ML), the twist frequency(TF)* and *the generalized mutation rate(GMR)*, are intended to represent the detail of search dynamics.

# 6. COMPUTER SIMULATION

We analyzed the dynamics of Simple GAs (SGAs)[4], TDGAs[13], SGGAs[5, 6] and BOAs[1, 2, 17, 18] by using the proposed indices.

## 6.1 Problem

We utilized the following knapsack problem as an example:

**Table 1: The setting of problem**

| Type | knapsack problem |
|---|---|
| The number of items | 50 |
| The number of fitness evaluations | 40000 |
| The number of fitness evaluations (performance simulation) | 50000 |

## 6.2 Parameter of each algorithm

Table 2 $\sim$ 5 show the parameters of each algorithm. Those parameters were set as the best values in the preliminary experiments.

The basic implementation of BOA is referred to the BOA with decision graphs (dBOA)[16].

**Table 2: The parameters of BOA**

| | |
|---|---|
| Population size | 100 |
| The number of generations | 800 |
| The ratio of new individual (%) | 50 |
| Selection | tournament selection |
| Tournament size | 4 |

**Table 3: The parameters of SGA**

| | |
|---|---|
| Population size | 100 |
| The number of generations | 400 |
| Selection | tournament selection |
| Tournament size | 4 |
| Crossover type | uniform |
| Crossover rate | 0.6 |
| Mutation rate of each locus | $\frac{1}{25}$ |

**Table 4: The parameters of TDGA**

| | |
|---|---|
| Population size | 100 |
| The number of generations | 200 |
| Crossover type | uniform |
| Crossover rate | 0.6 |
| Temperature | 5.0 |
| Mutation rate of each locus | $\frac{1}{25}$ |

**Table 5: The parameters of SGGA**

| | |
|---|---|
| Population size (H×W) | 10×10 |
| The number of generations | 200 |
| Crossover type | uniform |
| Crossover rate | 1.0 |
| Mutation rate of each locus | $\frac{1}{50}$ |
| The ratio of positive interaction | 0.25 |
| The absolute value of interaction | 10.0 |
| Temperature | 5.0 |

# 7. RESULTS

Figure 5 shows the performance of each algorithm. The abscissa indicates the number of fitness evaluations, and the ordinate indicates the performance, which is the number of successful runs (ie. found optimum solution) in 30 runs. It is shown in Figure 5 that the performance order is TDGA(best), SGA, SGGA and BOA(worst).

Figure 6 shows the number of cumulative species of each algorithm. It is shown in Figure 6 that the number of cumulative species of SGA is increasing in proportion to the number of fitness evaluations. On the other hand, the numbers of cumulative species of other 3 algorithms become constant at the early stage of searching. Figure 7 shows the enlarged graph of Figure 6. From this figure, the number of cumulative species of TDGA is the smallest in all algorithms, although TDGA has the best performance. This result shows TDGA can sample the individuals effectively.

Figure 8 shows the number of current species of each algorithm. It is shown in Figure 8 that the numbers of current species of BOA and SGGA always decline, while that of TDGA increases at a middle stage of searching. This result shows that the thermodynamical selection rule in TDGA can maintain diversity adaptively. On the other hand, BOA and SGGA are difficult to increase or maintain diversity. This fact is shown more clear in following entropy graphs. The numbers of current species of SGA is hight and almost constant from the beginning. This result shows that SGA can maintain diversity well, but fail to shrink the search space.

Figures 9 $\sim$ 12 show the variation of the entropy of each locus of each algorithm. Figures 10 and 11 show that TDGA and SGA can maintain diversity well, while Figures 9 and 12 show that BOA and SGGA lose diversity quickly. This result strongly suggests that maintaining diversity is important in solving knapsack problems. In Figure 9, the entropy of BOA becomes 0 the early stage of searching. This is because there is no mechanism of maintaining diversity in BOA. This result suggests to introduce a mechanism of maintaining diversity into BOA, e.g. the thermodynamical selection rule[13], has a positive effect.

Figures 13 $\sim$ 16 show the GMR of each algorithm. Those figures show that the GMR falls to 0 in the early stage of searching, except in the case of SGA. This result and Figure 10 show that SGA is able to maintain diversity by random perturbation. Figure 15 show that the GMR of TDGA is not so high compared to that of SGA, while entropy of TDGA in Figure 11 is not so low. This results show TDGAs maintain diversity effectively and the population change of TDGA is more moderate than that of SGA.

Figures 17 $\sim$ 20 show the TF of each algorithm. Those figures show that the TF falls to 0 in the early stage of searching, except in the case of SGA. This result shows that the effect of uniform crossover is only dominant at an early stage of searching in BOA, TDGA and SGGA.

In BOA, TDGA and SGGA, the GMR and the TF decline very quickly. To achieve more detail analysis, we need to observe the GMR and the TF between $P(t)$ and $P(t - \Delta t)$ $(\Delta t > 1)$.

To summarize, our indices have provided a much clearer understanding of the reasons for the differences in effectiveness of different algorithms on these typical knapsack problems than has been available previously.

# 8. CONCLUSION

In this paper, the authors propose novel indices called *the mutation length, the twist frequency* and *the generalized mutation rate* and show the results of comparison studies between 3 GAs and BOA. By means of several computational experiments, it has been confirmed that the proposed indices are effective to understand the dynamics of population-based optimization methods.

Deeper analysis of the features of these indices and applying the proposed analyzing method to another cases are subjects of further study. To extend the proposed method to

Genetic Programmings which have a complicated genotype is also important study.

## 9. ACKNOWLEDGMENTS

This research was partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Scientific Research on Priority Areas(2), 14084211, 2002-2005. This research was also partially supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for Young Scientists (B), 14750373, 2002-2004. We would like to thank Dr. Martin Pelikan for the excellent BOA program code[16]. Finally, the authors would like to acknowledge helpful discussions by Prof. Hajime Kita of Kyoto University and Dr. R. I. McKay of UNSW@ADFA.

## 10. REFERENCES

[1] S. Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. In *Tech. Report*, pages No. CMU–CS–94–163. Carnegie Mellon University, 1994.

[2] S. Baluja and S. Davies. Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space. In *Proc. of the Fourteenth International Conference on Machine Learning*, pages 30–38, 1997.

[3] T. Fukao. *Thermodynamical theory of distributed System*. Shyoukoudou, 1987.

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

[5] K. Hamada, N. Mori, and K. Matsumoto. A spin glass based genetic algorithm. In *Proceedings of the 45th ISCIE Conference (SCI'01)*, pages 39–40, 2001.

[6] K. Hamada, N. Mori, and K. Matsumoto. A spin glass based genetic algorithm-ii. In *Proceedings of the 46th ISCIE Conference (SCI'02)*, pages 593–594, 2002.

[7] J. H. Holland. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, 1975.

[8] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[9] N. Mori, S. Imanishi, H. Kita, and Y. Nishikawa. Adaptation to changing environments by means of the memory based thermodynamical genetic algorithm. In *Proceedings of the 7th International Conference on Genetic Algorithms*, pages 299–306, 1997.

[10] N. Mori and H. Kita. The entropy evaluation method for the thermodynamical selection rule. In *Proc. of the 5th the Genetic and Evolutionary Computation Conference (GECCO'99)*, page 799, 1999.

[11] N. Mori, H. Kita, and Y. Nishikawa. Adaptation to a changing environment by means of the thermodynamical genetic algorithm. In *Proceedings of the 4th Conference on Parallel Problem Solving from Nature*, pages 513–522, 1996.

[12] N. Mori, H. Kita, and Y. Nishikawa. Adaptation to a changing environment by means of the feedback thermodynamical genetic algorithm. In *Proceedings of the 5th Conference on Parallel Problem Solving from Nature*, pages 149–158, 1998.

[13] N. Mori, J. Yoshida, H. Tamaki, H. Kita, and Y. Nishikawa. A thermodynamical selection rule for the genetic algorithm. In *Proc. of 2nd IEEE Conference on Evolutionary Computation*, pages 188–192, 1995.

[14] H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions i. binary parameters. In *Proceedings of the 4th Conference on Parallel Problem Solving from Nature*, pages 178–187, 1996.

[15] H. Nishimori. *Spin Glass Theory and Statistical Mechanics of Information*. Iwanami, 1999.

[16] M. Pelikan. http://www.cs.umsl.edu/~pelikan/software.html, 2000.

[17] M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. BOA: The Bayesian optimization algorithm. In *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-99*, volume I, pages 525–532, 13-17 1999.

[18] M. Pelikan, D. E. Goldberg, and E. Cantu-Paz. Linkage problem, distribution estimation, and bayesian networks. *Evolutionary Computation*, 8(3):311–340, 2000.

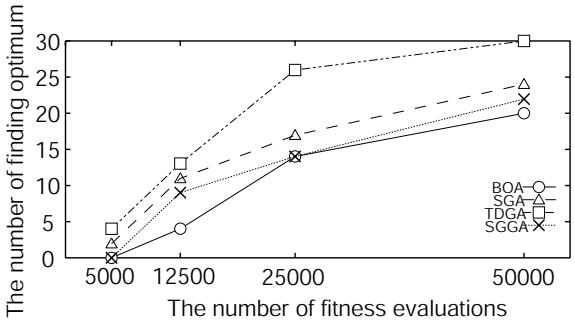


**Figure 5: Performance of each algorithm**

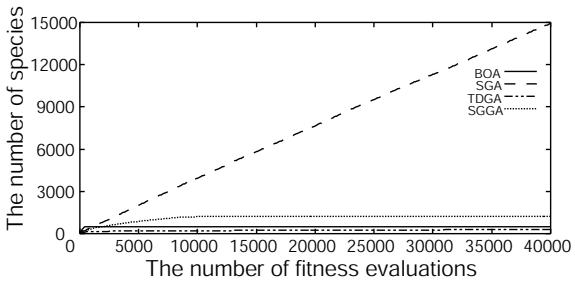


**Figure 6: The number of cumulative species**

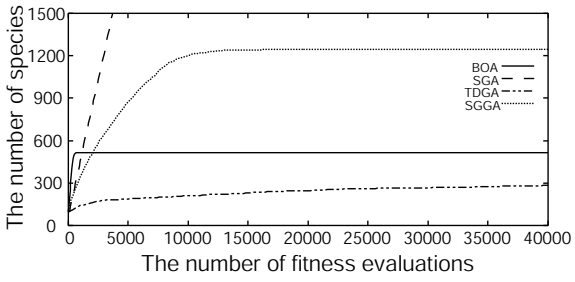


**Figure 7: The number of cumulative species (enlarged)**

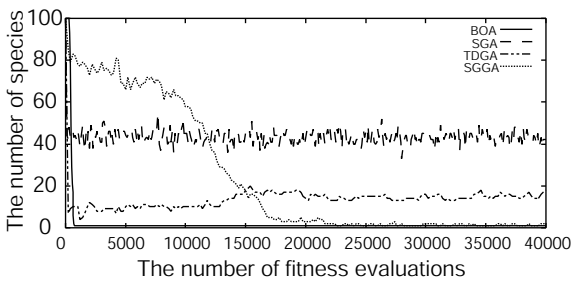


**Figure 8: The number of current species**

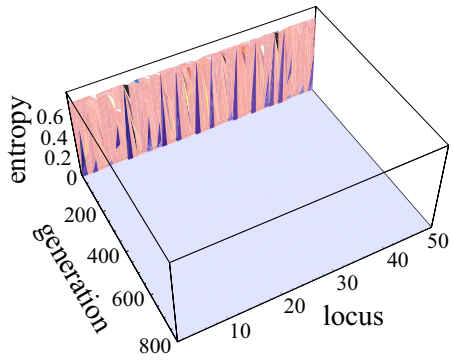


**Figure 9: Entropy of BOA in knapsack problem**

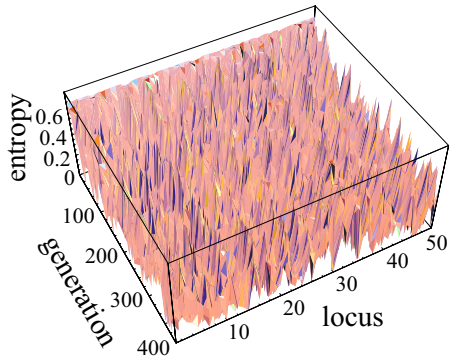


**Figure 10: Entropy of SGA in knapsack problem**

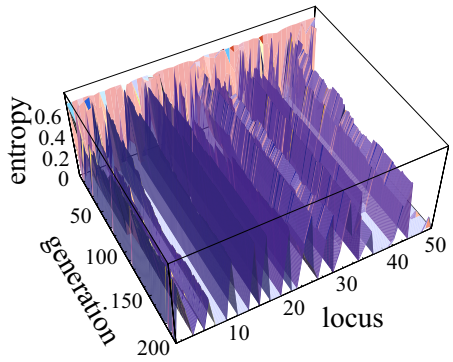


**Figure 11: Entropy of TDGA in knapsack problem**

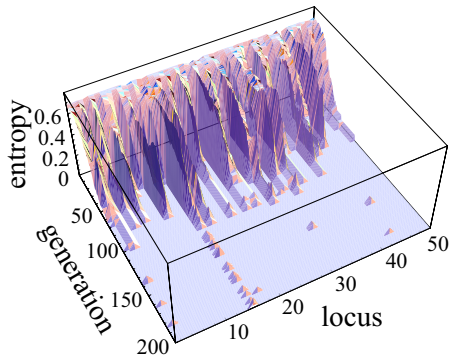


**Figure 12: Entropy of SGGA in knapsack problem**

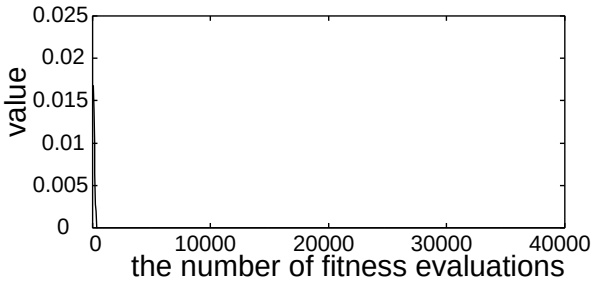


**Figure 13: GMR of BOA in knapsack problem**

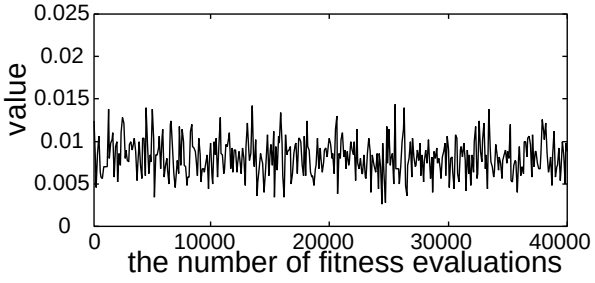


**Figure 14: GMR of SGA in knapsack problem**

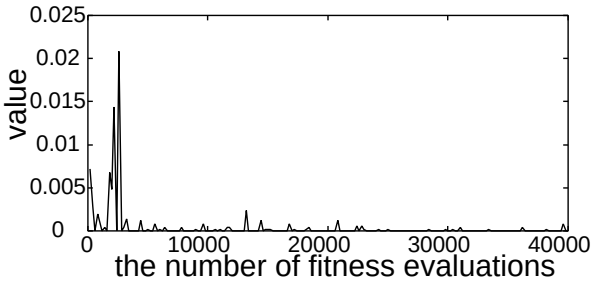


**Figure 15: GMR of TDGA in knapsack problem**

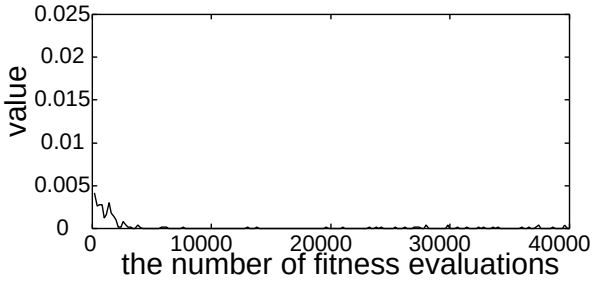


**Figure 16: GMR of SGGA in knapsack problem**

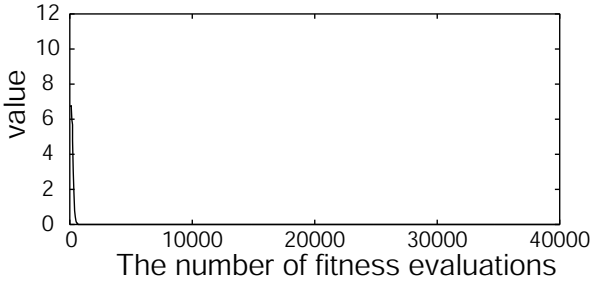


**Figure 17: TF of BOA in knapsack problem**

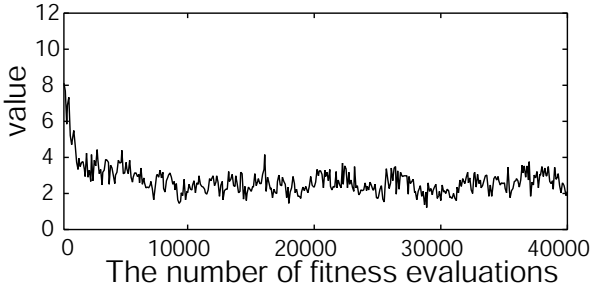


**Figure 18: TF of SGA in knapsack problem**

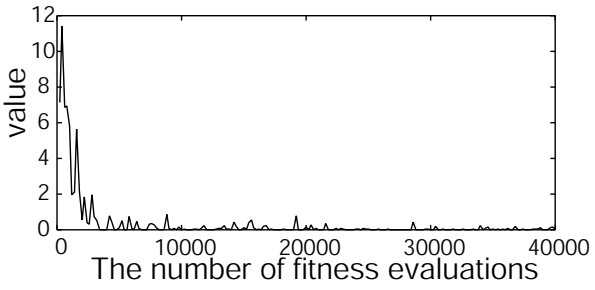


**Figure 19: TF of TDGA in knapsack problem**

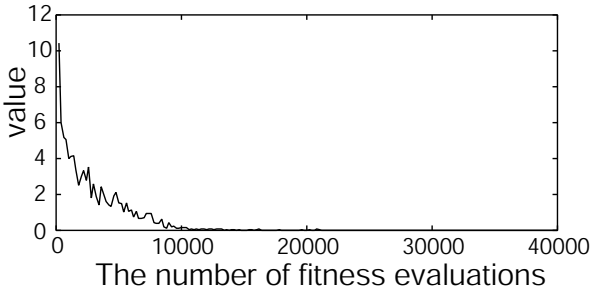


**Figure 20: TF of SGGA in knapsack problem**