Applied Soft Computing 11 (2011) 3385–3401

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Compact Genetic Algorithms using belief vectors

Joon-Yong Lee [a,*], Min-Soeng Kim [b], Ju-Jang Lee [a]

[a] Department of EE, KAIST, 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Republic of Korea
[b] SK Telecom, SK T-Tower, 11, Euljiro-2ga, Jung-gu, Seoul 100-999, Republic of Korea

ARTICLE INFO

Article history:
Received 3 July 2009
Received in revised form 26 February 2010
Accepted 3 January 2011
Available online 14 January 2011

Keywords:
Compact Genetic Algorithms (cGAs)
Belief vectors (BVs)
Genetic diversity
Entropy

ABSTRACT

Instead of the genetic operators such as crossover and mutation, compact Genetic Algorithms (cGAs) use a probability vector (PV) for the current population to reproduce offsprings of the next generation. Therefore, the original cGA can be easily implemented with no parameter tuning of the genetic operators and with reducing memory requirements. Many researchers have suggested their own schemes to improve the performance of the cGA, such as quality of solutions and convergence speed. However, these researches mainly have given fast convergence to the original cGA. They still have the premature convergence problem resulting in the low quality of solutions. Besides, the additional control parameters such as η of ne-cGA are even required for several cGAs. We propose two new schemes, called cGABV (an acronym for cGA using belief vectors) and cGABVE (an acronym for cGABV with elitism), in order to improve the performance of conventional cGAs by maintaining the diversity of individuals. For this purpose, the proposed algorithms use a belief vector (BV) instead of a PV. Each element of the BV has a probability distribution with a mean and a variance, whereas each element of a PV has a singular probability value. Accordingly, the proposed BV enables to affect the performances by controlling the genetic diversity of each generation. In addition, we propose two variants of the proposed cGABV and cGABVE, Var1 and Var2, employing the entropy-driven parameter control scheme in order to avoid the difficulty of designing the control parameter (λ). Experimental results show that the proposed variants of cGAs outperform the conventional cGAs. For investigating the diversity of each cGA, the entropy is employed and calculated at each generation. Finally, we discuss the effect of λ related to the variances of the BV through the additional experiment.

1. Introduction

The original compact Genetic Algorithm (cGA) [1] is one of estimation of distribution algorithms (EDAs), which use probability density functions estimated from the current population to reproduce offsprings of the next generation instead of the genetic operators such as crossover and mutation. The original cGA firstly introduced in [1] has mainly two advantages in comparison with the simple GA (sGA) which employs uniform crossover. First, the original cGA is easily implemented without tuning parameters of the genetic operators according to a given problem. In other words, the original cGA basically requires no specified knowledge to tune the parameters so that common users may easily implement the original cGA to solve their own problems without prior knowledge about the GA. In addition, the cGA is more useful to solve the problems with memory constraints because the cGA reduces the memory requirements from $L \cdot N$ of the sGA to $L \cdot \log_2(N+1)$ (L: the length of a chromosome, N: the population size) [1,2]. Therefore, some researchers have already applied the cGAs to the practical problems such as the intrinsic evolvable hardware [3], partitioning and placement of the multi-FPGA systems [4,5], the traveling salesman problems (TSPs) [6], motor control [7], and the hierarchical sensor network management [8].

The performance of the cGA is commonly evaluated by two measures: quality of solutions and convergence speed [1,9–11]. Moreover, these two measures generally have a trade-off relationship with each other in the conventional cGAs. The original cGA shows comparable performances to the sGA in terms of both measures for easy problems involving the low order building blocks (BBs) [1]. However, the original cGA shows poor performances for solving difficult continuous multimodal problems. In [1], a parameter, called a 'tournament size', is suggested to improve this drawback by controlling the selection pressure. A 'tournament size' represents the number of individuals participating in the selection of a winner. Therefore, if a tournament size gets larger, the selection pressure becomes higher so that the convergence speed of the cGA is improved. At the same time, however, the quality of solutions gets worse than the sGA under the same population size because a local optima problem (i.e., the premature convergence) tends to occur due to the high selection pressure. Furthermore, this scheme

* Corresponding author. Tel.: +82 42 350 5432; fax: +82 42 350 5432.
E-mail addresses: leejy79@kaist.ac.kr (J.-Y. Lee), minsoeng.kim@gmail.com (M.-S. Kim), jjlee@ee.kaist.ac.kr (J.-J. Lee).

1568-4946/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2011.01.010

requires the additional memory in proportion to the tournament size.

Since then, there have been various related works to improve the performances of the cGA and to keep the benefits of the original cGA, such as efficient memory requirements and the simplicity for the implementation. Conventional methods are roughly categorized into four kinds of cGAs: the cGAs combined with a local search method [4,6,12], the cGAs combined with a genetic strategy (e.g., mutation) [3,10,13], the cGAs employing elitism [3,9], and the cGAs using a different updating strategy of the PV [14], respectively.

Firstly, Harik et al. [12] presented an extended cGA (EcGA) for solving complex problems by using the marginal product model (MPM) search algorithm. MPMs are formed as a product of marginal distributions on a partition of the genes. Unlike the models used in the cGA, MPMs can represent probability distributions for more than one gene at a time. Thus, the EcGA can find the better solutions for the difficult problems, whereas the EcGA requires additional memory and more computational costs per a function evaluation. Baraglia et al. [6] tried to combine the cGA with a Lin–Kernighan (LK) local search algorithm for solving the TSPs. Their algorithm can achieve the better solutions for the TSPs than other conventional methods. Hidalgo et al. [4,5] also proposed a hybrid parallelized cGA-LK for solving the partitioning and placement of the multi-FPGA systems.

Secondly, Zhou et al. [10] proposed the mutated by bit cGA (MBBCGA) in which every bit of an individual is mutated one by one and every newly mutated individual is evaluated to compete with the previous individual at each generation. Anh et al. [13] proposed an augmented cGA (acGA) using tournament selection without replacement (TS/R). In the original cGA, if same individuals begin to appear in a generation because of convergence of the PV to a solution (i.e., when many elements of the PV nearly equal to 1 or 0), evolution process slows down until different individuals appear. The acGA tried to solve the slowdown problem in the steady state phase of evolution by flipping non-convergent genes with a probability. Since flipping non-convergent genes helps make individuals different from each other, the acGA slightly improves the performance of the cGA.

Thirdly, Anh et al. [9] proposed two elitism-based cGAs: the persistent elitist cGA (pe-cGA) and the nonpersistent elitist cGA (ne-cGA). The pe-cGA and ne-cGA combine the original cGA with the elitist selection method, where the fittest individual of each generation is guaranteed to be selected as the parent of the offsprings in the next generation. The elitism of the pe-cGA leads to the faster convergence because the pe-cGA keeps the best solution in the next generation until the better solution appears. However, the pe-cGA shows poor quality of solutions due to the premature convergence. On the other hand, since the ne-cGA does not persistently keep the best and ne-cGA reinitializes the best individual in an effective manner, the selection pressure of the ne-cGA is reduced to avoid the premature convergence (i.e., convergence to a local solution). The parameter η, called the allowable length of the elite chromosome's inheritance, should be also preset properly according to the problems. (In [9], $\eta = 0.1N$, where N is the population size.) However, the addition of elitism still leads to the loss of quality of solutions in comparison with the original cGA, even if the convergence speed is improved as faster.

Finally, Rimcharoen et al. [14] presented the moving average technique to modify the update rate of the PV in the cGAs. The moving average technique prevents the PV from rapidly converging to an unreliable solution. The presented cGAs can improve the quality of solutions with a smaller number of function evaluations in compared with the original cGA. However, this scheme relatively has the slow convergence in compared with the elitism-based cGAs (pe-cGA and ne-cGA). Furthermore, since the moving average window size (M), denoted as a control parameter of this schemes, has few effects on the quality of solutions and the number of function evaluations, the tournament size is applied to improve the converging speed as in [1].

The aim of this paper is to propose new schemes which can easily control the performances (the quality of solutions and the convergence speed) without any complex additional methods and without the loss of advantages of the original cGA. For this purpose, firstly, it is required that the advantages of the original cGA, such as the less memory requirement and simpler implementation than other EDAs and GAs [1,2,7], are carried in the proposed cGAs. Next, the proposed cGAs in this paper improve the quality of solutions as well as the convergence speed, while the conventional cGAs mainly focused on the fast convergence. Accordingly we recall that the genetic diversity in GAs is a pretty important factor to decide the performances [15]. It is inferred that the genetic diversity maintenance leads to explore with avoiding premature convergence also in cGAs. Moreover, the quality of solutions in the proposed cGAs is less affected by the population size unlike in the conventional cGAs. It describes that the proposed cGAs can produce the high quality solutions with the low population size by relevantly controlling the genetic diversity.

The key idea for these requirements is to apply the concept of 'probability of probability' to the PV of the original cGA. For this purpose, we propose that each element of the PV has a probability distribution (e.g., a normal distribution) with a mean and a variance, instead of a singular value. The similar assumption was presented by the stochastic hill climbing with learning by vectors of normal distributions (SHCLVND) [16]. SHCLVND also use a normal distribution instead of a singular value to represent a real-coded variable. This representation using a probability distribution has been extended to the other EDAs in order to concern the dependencies among the variables [17]. In the same manner, we can use a probability distribution instead of a single probability value as the element of the PV in the cGA. Therefore, this probability distribution (belief function) implies the certainty of the PV with which genes of offsprings are defined as 1. In other words, the higher 'degree of belief' of the probability means that the probability is more reliable and vice versa. Finally, a belief vector (BV), similar to a vector of normal distributions [16], is proposed instead of the conventional PV of the cGA. The BV employs the concept of 'degree of belief' to the PV by using a probability distribution model for each element. Each element of the BV is represented as a probability distribution with a center and width which, respectively, represent the mean and uncertainty of the probability value. In the beginning of evolution, all the elements of the BV (i.e., probability models) are unreliable due to insufficient information from scanty samples of the early generations. Thus, a variance of each element is assigned as high to get the high uncertainty. As evolution proceeds, a variance of each element of the BV is proposed to decay in accordance with the predesigned control parameter (λ) because it is commonly assumed that the probability models of the BV are reliably adapted during the succession of generations. Finally, we propose two cGAs using the BV: cGABV and cGABVE. The former uses the BV without elitism, while the later uses the BV with elitism. The proposed cGAs are defined without any complex additional local search methods. Also, they can be easily implemented without additional memory requirements. However, there still remains the difficulty of appropriately designing additional parameter λ according to the given problem. To avoid this difficulty, the entropy-driven parameter control algorithm is employed to automatically adjust the control parameter λ regardless of the given problems. 'Var1' means the variant applying the entropy-driven scheme to the cGABV. 'Var2' represents the variant applying the entropy-driven scheme to the cGABVE.

To evaluate the performance of the proposed cGAs, we compare the proposed cGAs with the original cGA, pe-cGA, and ne-cGA

N: Population size, L: Chromosome length. j: jth generation.

Step 1. Initialize the PV. ($PV^j = \{p_1^j, p_2^j, \cdots, p_L^j\}$).
$j = 1$;
for($i = 0; i < L; i$++)
$p_i^j = 0.5$;

Step 2. Generate two chromosomes ($indv_1^j, indv_2^j$) from the PV.
for($k = 0; k < 2; k$++){
for($i = 0; i < L; i$++){
if(rand( ) < p_i^j)
$indv_k^j.gene[i] = 1$;
else
$indv_k^j.gene[i] = 0$;
}
}

Step 3. Let them evaluate and compete.
$[indv_{winner}^j, indv_{loser}^j]$ = compete($indv_1^j, indv_2^j$);

Step 4. Update the PV.
for($i = 0; i < L; i$++){
if($indv_{winner}^j.gene[i] != indv_{loser}^j.gene[i]$){
if($indv_{winner}^j.gene[i] == 1$)
p_i^j+=1/N;
else p_i^j-=1/N;
}
}
$j = j + 1$;

Step 5. Check if the PV has converged. Go to **Step 2**, if it is not satisfied.

Step 6. The PV represents the final solution.

Fig. 1. Pseudocode of the original cGA. 'rand( )' represents that it generates the uniform random number in the range of (0,1).

which have the most reliable performances for various problems among the existing cGAs. Accordingly, as similar to these conventional cGAs, numerical experiments are performed with various problems involving the continuous unimodality and multimodalities. Moreover, the quantitative diversity measure of the cGAs is investigated by using the entropy of probability models. To investigate the effect of the elitism in the propose cGAs, we compare the proposed cGABV with the proposed cGABVE. We also compare Var1 and Var2 with the cGABV and cGABVE to verify the effect of the entropy-driven parameter control scheme in the propose cGAs. Furthermore, we also discuss the effect of the control parameter (λ) with the additional experimental results.

The remainder of this paper is as follows. In Section 2, we briefly review the original cGAs. After that, we describe the proposed cGAs using the BV in compared with the original cGA using the PV in Section 3. In Section 3, the elitism is also considered for the fast convergence due to the higher selection pressures. Moreover, two variants employing the entropy-driven λ control scheme are proposed in this section. Finally, after the experimental results show the superiority of the proposed cGAs in comparison with the conventional cGAs in Section 4, we conclude this paper in Section 5.

2. The original cGA

In this section, we briefly review the original cGA and the PV, before we present the proposed algorithms.

Fig. 1 describes a pseudo-code of the original cGA [1], where L denotes the bit length of one chromosome. The i th element value of the PV means the i th gene's probability. In the framework of the original cGA, firstly, each element of the PV is initialized as 0.5. Thus, each gene is randomly generated by 0 or 1 with the same probability in the first generation. After two individuals are evaluated and competed with one another, each probability is updated to favor the winner individual (See Step 4 in Fig. 1).

In the original cGAs, each element of the PV denotes a value of probability to assign 1 to the i th gene of an individual. In each generation, p_i^j is updated as much as $1/N$ (i.e., an update rate), where N denotes the population size. Hence, increasing the population size (N) results in increasing the number of steps needed for the elements of the PV to converge to 0 or 1 and leading to exploration. In other words, if the chances that the genetic diversity is maintained increase with increasing N, the quality of solutions also gets higher. However, the slowdown of convergence occurs at the same time. Furthermore, if some elements of the PV begin to converge to 0 or 1, the expressive range (i.e., diversity) of the variable generated by this PV becomes too small so that the evolution of a population is stagnant until two different individuals appear. If the population size (N) becomes smaller, this stagnant phase is likely to occur in an early stage of evolution. Hereby, the solutions tends not only to have premature convergence (i.e., converge to a locally optimized solutions), but also to keep on staying around local solutions.

3. The proposed cGAs

In this section, we present the proposed algorithms in detail. We also describe the difference between the PV of the original cGA and the BV of the proposed cGAs. Furthermore, the elitism is considered for improving the convergence speed of the proposed cGAs. In addition, the entropy measure is applied to reasonably analysis the diversity maintenance of a population in the cGAs. Finally, we introduce the entropy-driven parameter control scheme to compensate the defect due to the additional control parameter λ.

3.1. The proposed cGA using BVs (cGABV)

Fig. 2 describes a pseudo-code of the proposed cGA. As previously mentioned, the concept of a BV is mainly distinguishable for the original cGA. We enhance the PV of the cGA by employing a probability distribution for each element. The original cGA is chosen as a parent algorithm and the proposed BV is embedded in the cGA to improve its performance and to prevent the algorithm from falling into the premature convergence. A variance of probability distribution models, called a belief function, represents how the singular value of each probability (element) used to generate each gene (0 or 1) is unknown and untrustworthy. Thus, each probability value used to generate each gene is obtained from this belief function. Therefore, the BV is defined as a set of real-coded values including a mean and a standard deviation (i.e., a center and a width). The belief vector of the j th generation BV^j is defined as $[\mu_1^j, \sigma_1^j, \mu_2^j, \sigma_2^j, \cdots, \mu_L^j, \sigma_L^j]$, where L denotes the length of a chromosome. μ_i^j and σ_i^j, respectively, describe a mean and a standard deviation of a belief function with regard to the i th gene in the j th generation. In our work, all the σ_i^j are referred as identical regardless of the genes to minimize the additional memory requirements. Besides, a normal distribution (Gaussian distribution) is applied to define a belief function for i th gene as follows:

$$f_{\mathrm{bel},i}^j(x, \mu_i^j, \sigma_i^j) = \frac{1}{\sqrt{2\pi}\sigma_i^j} e^{((-(x-\mu_i^j)^2)/2(\sigma_i^j)^2)}. \quad (1)$$

As shown in Fig. 3, a variance produces the probability (uncertainty) of a probability value that a gene becomes 1. Hence the PV becomes a special case of $\sigma_i^j \to 0$ in this definition of the BV. As the uncertainty of a probability value increases with increasing a variance (i.e., a width of a distribution), the diversity of a population is also maintained, and vice versa. Furthermore, the shape of $f_{\mathrm{bel},i}^j$ needs to properly change as evolution proceeds (e.g., as a → b

N: Population size, L: Chromosome length. j: jth generation.
N_{eval}: The number of function evaluations. N_{max}: The maximum number of function evaluations.

Step 1. Initialize the BV. $BV^j = \{\mu_1^j, \sigma_1^j, \mu_2^j, \sigma_2^j, \cdots, \mu_L^j, \sigma_L^j\}$.

```
j = 1;
for(i = 0; i < L; i++){
    μ_i^j = 0.5;
    σ_i^j = σ_0;
    p_i^j = 0.5;
}
```

Step 2. Generate two chromosomes ($indv_1^j$, $indv_2^j$).

```
for(k = 0; k < 2; k++){
    for(i = 0; i < L; i++){
        if(rand( ) < p_i^j)
            indv_k^j.gene[i] = 1;
        else
            indv_k^j.gene[i] = 0;
    }
}
```

Step 3. Evaluate them and compete with each other.

$[indv_{winner}^j, indv_{loser}^j]$ = compete($indv_1^j, indv_2^j$);

Step 4. Update the BV.

```
for(i = 0; i < L; i++){
    if(indv_winner^j.gene[i]!=indv_loser^j.gene[i]){
        if(indv_winner^j.gene[i]==1)
            μ_i^j+=1/N;
        else      μ_i^j-=1/N;
    }
}
```

Step 5. Compute the variance (σ_i^j) of the BV via Eq. (2).

Step 6. Generate the probabilities (p_i^j) of every gene from $N(\mu_i^j, \sigma_i^j)$.

```
for(i = 0; i < L; i++)
    p_i^j ~ N(μ_i^j, σ_i^j);
j = j + 1;
```

Step 7. Check if termination condition ($N_{eval} \geq N_{max}$) is satisfied. Go to **Step 2**, otherwise.

Step 8. $indv_{winner}^j$ represents the final solution.

Fig. 2. Pseudocode of the proposed cGABV. 'rand( )' represents that it generates the uniform random number in the range of (0,1).

→ c in Fig. 3). For this purpose, the diversity should be guaranteed enough that the exploration of populations in the early generations vigorously occurs to avoid a premature convergence. In the end of generations, the selection pressure becomes higher as decreasing uncertainties of the belief functions for the exploitation in the vicinity of a global solution. For this purpose, the variance in the j th generation is defined as follows:

$$\sigma_i^j = \sigma_0 \cdot \left(1.0 + \tanh\left(-\lambda \frac{j}{N_{\max}}\right)\right). \tag{2}$$

In Eq. (2), $N_{\max}$ denotes the maximum number of function evaluations and this function is monotonically decreasing as generations go by. The full width at half maximum (FWHM), considered as a simple measure of the width of a distribution, should be less than 0.5 for a probability value randomly generated from the belief function (normal distribution) to nearly stay in the range of [0, 1]. Hence, $\sigma_0 = 0.5/(2\sqrt{2\ln 2}) \simeq 0.2123$ according to the relation between the FWHM and σ_i^j for a Gaussian function. Furthermore, the variance curves change according to λ, as shown in Fig. 4. The larger λ results in the faster reduction of the variance which describes the degree of certainty, and vice versa. Experimental results show that λ noticeably affects the performance of the proposed cGAs. In particular, when the variance is greater than the update rate ($1/N$) of the mean value, the variance affects to generate the probability value for each element (See Step 4 in Fig. 2). As evolution proceeds, the affects of a variance gradually become insignificant due to continuous decrease of a variance. Therefore, in the same manner to calculate σ_0 by using the FWHM, an inequality can be formulated as follows:

$$\frac{1}{2}\left(1 + \tanh\left(-\lambda \frac{j_e}{N_{\max}}\right)\right) \geq \frac{1}{N}, \tag{3}$$

where j_e means a boundary of generations in which the variance is more effectible to the probability (p_i) of elements than the update rate of the mean value. Eq. (3) can be rewritten as

$$\tanh^{-1}\left(\frac{2}{N} - 1\right) \leq -\lambda \frac{j_e}{N_{\max}}. \tag{4}$$

Here

$$\tanh^{-1} x = \frac{1}{2}\left[\ln(1+x) - \ln(1-x)\right]. \tag{5}$$

From this, Eq. (4) can be rewritten as follows:

$$\frac{1}{2\lambda}\ln(N-1) \geq \frac{j_e}{N_{\max}}. \tag{6}$$

Therefore, if the population size and λ are, respectively, given as 100 and 10, $j_e \leq 0.2298 N_{\max}$. In other words, if the maximum number of

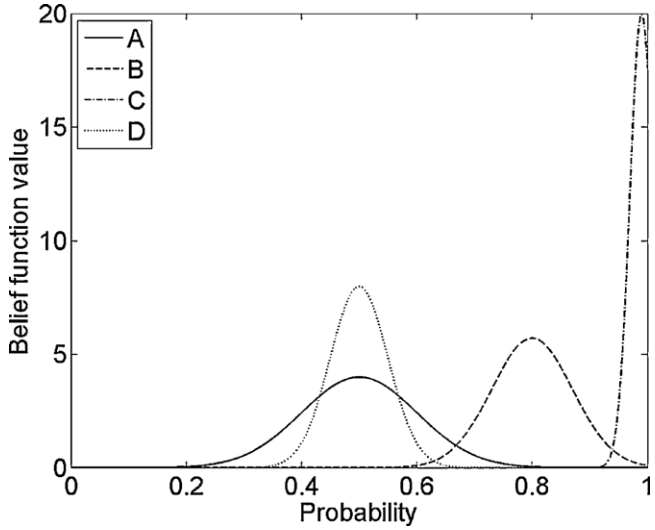

Fig. 3. The belief functions in various conditions. Each condition is as follows: (A) $\mu_i = 0.5, \sigma_i = 0.1$, (B) $\mu_i = 0.8, \sigma_i = 0.07$, (C) $\mu_i = 0.99, \sigma_i = 0.02$ and (D) $\mu_i = 0.5, \sigma_i = 0.05$. As shown in this figure, if mean values of two belief functions are identical to each other but variances are different from each other (as between A and D), the probabilities generated from two belief function may differ from each other with various degrees of uncertainties.

function evaluations is 4000, the variance loses effect after about 919th generation so that cGAs using the BV may become almost similar to cGA using the PV afterwards.

3.2. Use of the elitism

Use of the elitism is a simple way to obtain the higher selection pressure. The elitism leads the fast convergence rate also in the proposed cGA using the BV. To combine the elitism with the previously proposed cGABV, the procedure of Fig. 5 is added to the peudocode of the proposed cGA, which is called the proposed cGABVE.

3.3. Diversity measure for the cGAs

In this paper, the concept of entropy is applied to evaluate the diversity in the cGAs. The Shannon entropy is a measure of the uncertainty associated with a random variable. Actually, this entropy has been presented in evolutionary computation as a diversity measure of a population modeled by a probability distribution [15,18,19]. In order to investigate the diversity of the cGA, entropy of the j th generation is calculated as follows:

$$D^j = -\sum_{i=1}^{L} p_i^j \cdot \log p_i^j. \tag{7}$$

p_i^j means a singular probability of each element in case of the PV, whereas p_i^j in the BV means a random number generated from the distribution $f_{bel,i}^j(x, \mu_i^j, \sigma_i^j)$ using Box–Muller transformation [20]. As investigating the diversity of each cGA, we show that there are different behaviors between the conventional cGAs and the proposed cGAs in the next section.

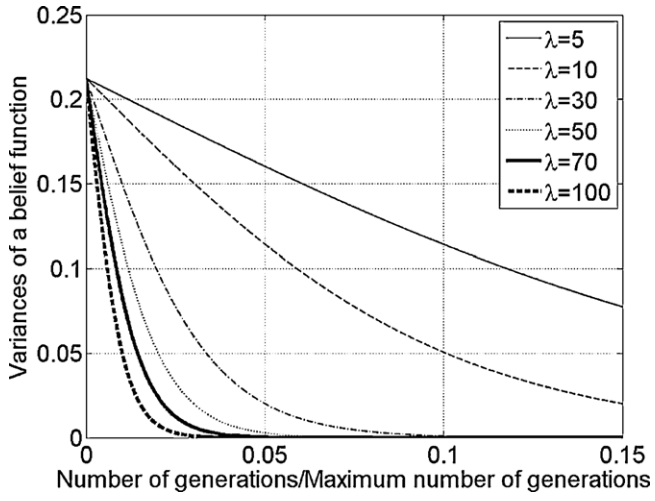

Fig. 4. The change of the variance in accordance with λ. $\sigma_0 = 0.5/(2\sqrt{2 \ln 2}) \simeq 0.2123$.

$indv_{elite}^j$: The elite individual.

Step 2*. Generate two chromosomes $(indv_1^j, indv_2^j)$.

```
for(i = 0; i < L; i++){
    if(rand( ) < p_i^j)
        indv_1^j.gene[i] = 1;
    else
        indv_1^j.gene[i] = 0;
}
if(j == 0)
    for(i = 0; i < L; i++){
        if(rand( ) < p_i^j)
            indv_2^j.gene[i] = 1;
        else
            indv_2^j.gene[i] = 0;
    }
else
    indv_2^j = indv_elite^j;
```

Step 3*. Evaluate them and compete with each other.

$[indv_{winner}^j, indv_{loser}^j] = \text{compete}(indv_1^j, indv_2^j)$;

$indv_{elite}^j = indv_{winner}^j$;

Fig. 5. Additional pseudocodes for the proposed cGABVE.

3.4. Entropy-driven λ control for cGABV and cGABVE

Entropy-driven parameter control for evolutionary algorithms (EAs) are proposed in [21]. In [21], the control parameters such as the crossover rate and mutation rate are adjusted according to the entropy value. In the same manner, we modify the entropy-driven control written in programmable parameter control for EA (PPC_{EA}) [21] in order to avoid the difficulty of tuning λ according to the given problems. Algorithm 3.1 describes the pseudocode of the entropy-driven scheme used in this paper. Algorithm 3.1 is performed between Step 4 and Step 5 in the cGABV and between Step 6 and Step 7 in the cGABVE.

Algorithm 3.1 *(Entropy-driven λ control scheme for cGABV and cGABVE).*

After calculating the entropy D^j at the j th generation, adjust λ as follows:

```
if D^j > 0.6 then
  λ^{j+1} = λ^j × 1.22
else
  if D^j < 0.4 then
    λ^{j+1} = λ^j × 0.82
  end if
else
  λ^{j+1} = λ^j
end if.
```

Table 1
Test functions.

Problems	Name	Function
f_1	OneMax problem	$f_1(\mathbf{x}) = \sum_{i=1}^{m} x_i$ ($x_i = 0$ or 1)
f_2	MDP problem	$f_2(\mathbf{x}) = \sum_{i=1}^{m/3} g_2(x_{2i})$ [a]
f_3	3-Bit trap problem	$f_3(\mathbf{x}) = \sum_{i=1}^{m/3} g_3(x_{3i})$ [b]
f_4	De Jong's function 1	$f_4(\mathbf{x}) = \sum_{i=1}^{m} x_i^2$
f_5	Circle function	$f_5(\mathbf{x}) = \left(\sum_{i=1}^{m} x_i^2\right)^{1/4} \cdot \left[\sin^2\left(50 \cdot \left(\sum_{i=1}^{m} x_i^2\right)^{1/10}\right) + 1.0\right]$
f_6	Schaffer's binary function	$f_6(\mathbf{x}) = 0.5 + \frac{\sin^2\left(\sqrt{\sum_{i=1}^{m} x_i^2}\right) - 0.5}{\left(1.0 + 10^{-3} \cdot \left(\sum_{i=1}^{m} x_i^2\right)\right)^2}$
f_7	Ackley's path function	$f_7(\mathbf{x}) = -a \cdot m + e^{-b \cdot \sqrt{\frac{\sum_{i=1}^{m} x_i^2}{m}}} - e^{\frac{\sum_{i=1}^{m} \cos(c \cdot x_i)}{m}} + a + e^1$

[a] $g_2(x_{2i})$ and $g_3(x_{3i})$ are fully described in Eqs. (8) and (9), respectively.
[b] m denotes the number of variables (see Table 2). $\mathbf{x} = \{x_1, x_2, \cdots, x_m\}$.

As shown in Algorithm 3.1, as entropy is greater than 0.6, λ is increased to facilitate the exploitation phase. As entropy is smaller than 0.4, more exploration is needed to avoid the premature convergence into the local optima. Consequently, we propose two additional variants of the cGABV and cGABVE, 'Var1' and 'Var2', which represent the cGABV and cGABVE with entropy-driven parameter control, respectively.

4. Numerical studies

In this section, to evaluate the performance of the proposed cGAs, we have numerical studies with various test functions which deal not only with the problems involving the BBs, but also with the real-valued problems with more than one local optimum point. The four proposed cGAs, cGABV, cGABVE, Var1, and Var2, are compared with the original cGAs, pe-cGA, and ne-cGA for each problems under the same conditions. The fitness function values and the number of function evaluations averagely obtained over 50 independent runs are taken as performance measures according to the population size as in [1]. All the problems have been widely used for evaluating the performance of optimization algorithms. The brief descriptions of all the test functions can be offered from the original references [9,22]. Table 1 presents the test functions.

4.1. Performance criteria and experimental conditions

For investigating the performances, fitness function error values, the number of successful runs, and the number of function evaluations for successful runs are considered as the criteria of 'quality of solutions' and 'convergence speed', respectively. Mean and standard deviation of fitness function error values are calculated from the fitness function error values of the winner individual in population at the last generation according to various population sizes. Generally speaking, the time required to evaluate the fitness function occupies a major portion of the total time of a generation in practical real world problems. Therefore, the number of function evaluations is considered as a measure for evaluating the convergence property. Genetic diversity is also investigated in each independent run by calculating entropy as in Eq. (7). Therefore, three features are mainly considered for evaluating the algorithms as follows:

1. *Fitness function error values versus the population size:* The fitness function error values ($f(x) - f(x^*)$) for each algorithm are averagely reported from 50 independent runs. As the fitness function error values get smaller, it indicates that the quality of solutions becomes higher. Therefore, this feature is considered as a criterion for quality of solutions. Termination condition for calculating the fitness function error values is the maximum number of function evaluations.
2. *Number of successful runs:* A successful run indicates the run during which the algorithm achieves the fixed accuracy level within the maximum number of fitness evaluations. In other words, according to the population size, this number means how often the algorithm succeeds in finding the globally optimal solution with overcoming the local optima problem. Therefore, this feature is also considered as the measure to evaluate quality of solutions. The number of successful runs is counted among 50 independent runs.
3. *Number of function evaluations versus the population size:* The number of function evaluations needed in each run to achieve the corresponding fixed accuracy level is recorded. The maximum number of function evaluations applies. In the previous works, the convergence speed is commonly related to how fast the probabilities of the PV or BV are converged to 0 or 1. However, even if the probabilities of the PV or BV are quickly converged, the cGAs are useless if the cGAs just provide not the global optimum solution but the local optimum solution. Therefore, we consider the convergence speed meaningfully, only if the cGAs achieve the successful run in this paper.

Table 2 shows experimental conditions used to evaluate the conventional cGAs and the proposed cGAs, according to each problem. For the numerical studies, we define η as $0.1N$ suggested from [9] and λ as 10 from experience. For Var1 and Var2, λ^1 is also initialized as 10. After the experimental results are reported, we will discuss the effect of λ in the last subsection.

4.2. Results for the low-BBs problems

A 100-bit one max problem and a minimum deceptive problem (MDP) [9,23] are considered as representative test problems involving the lower-order BBs. Tables 3 and 4 show the results of the cGAs for the OneMax problem (f_1). Table 3 shows the mean and standard deviation of the fitness function error values obtained from the winner individual of the last generation in 50 independent runs during $N_{\max}$. As the mean value is smaller, it is likely that the solutions are closer to the optimal solution. Table 4 shows the number of function evaluations taken for the fitness function error value of elite to achieve the given accuracy level as well as the number of successful runs. Accordingly, the larger number of successful runs indicates that quality of solutions is higher. As mentioned previously, the number of function evaluations averagely used for successful runs of each cGA is considered as the convergence speed in this paper. As shown in Table 4, Var1 gen-

Table 2
Experimental conditions.

Test problems	Experimental conditions [a]						Properties [a]
	f^*	$[x_i^{\min}, x_i^{\max}]$	$L(m)$	$N_{\max}$	P_m	P_g	Max/min
f_1	1E−6	·	100 (100)	4000	100	10	Maximization
f_2	1E−6	·	20 (10)	4000	200	20	Minimization
f_3	1E−6	·	30 (10)	100000	3000	300	Minimization
f_4	1E−6	[−5.12,5.12]	60 (3)	10000	200	20	Minimization
f_5	1E−2	[−32.767,32.768]	40 (2)	10000	100	10	Minimization
f_6	1E−6	[−100.0,100.0]	40 (2)	30000	300	30	Minimization
f_7	1E−2	[−32.767,32.768]	100 (5)	40000	300	30	Minimization

[a] f^*: fixed accuracy level for each function, $[x_i^{\min}, x_i^{\max}]$: range of each variable, L: chromosome's length, m: number of variables, $N_{\max}$: maximum number of function evaluations, P_m: maximum population size, P_g: Gap between population sizes.

Table 3
Results for f_1. Mean and standard deviation of the fitness error values are reported according to the population size.

Pop. size	Org. cGA		pe-cGA		ne-cGA		cGABV		cGABVE		Var1		Var2	
10	13.18	2.14	24.62	3.52	35.4	2.66	2.48	1.25	11.86	2.46	0.2	0.45	0	0.00
20	3.64	1.61	20.08	3.22	25.84	2.51	0.54	0.85	11.06	2.35	1.68	1.12	0	0.00
30	1.22	1.10	17.16	3.18	23.34	3.48	0.12	0.32	9.98	2.63	1.02	1.10	0	0.00
40	0.28	0.49	17.76	2.78	21.46	3.32	0.04	0.20	9.94	2.12	0.84	0.90	0	0.00
50	0.02	0.14	15.8	2.76	18.64	3.20	0	0.00	8.98	2.15	0.8	0.98	0	0.00
60	0.04	0.20	13.04	2.76	17.82	2.74	0	0.00	9.76	2.61	0.78	1.03	0.02	0.14
70	0	0.00	13.22	2.82	16.78	3.47	0	0.00	8.56	2.08	0.88	1.05	0.02	0.14
80	0	0.00	11.9	2.50	15.34	3.36	0	0.00	9.28	2.23	0.28	0.60	0.02	0.14
90	0	0.00	11.5	2.71	14.08	2.70	0	0.00	8.64	2.68	0	0.00	0.02	0.14
100	0	0.00	11.02	2.46	14.44	3.48	0	0.00	8.52	2.44	0	0.00	0.02	0.14

erally outperforms the other cGAs regardless of a population size. Especially, Var1 with the population size of 40 provides the best performance in terms of quality of solutions as well as the convergence speed for the successful runs. However, since Var1 throws away the elite at each generation, the average value of the fitness error values of Var1 at the last generation are not converged to zero in spite of a one-hundred percent success. In other words, the cGAs without the elitism is likely to keep doing the exploitation and exploration if the probability for each gene is not converged into 0 or 1, although the global optimal solution is appeared. In the population size of more than 70, the original cGA also gives a one-hundred percent success with the relatively small number of function evaluation. On the other hand, the cGAs combined with the elitism such as pe-cGA, ne-cGA, and cGABVE give no successful run. It is shown that the elitism is likely to give the negative affects due to the higher selection pressure as cGAs are applied to the OneMax problem. Var2 overcomes the locally optimized problem caused by the elitism with adjusting λ according to the genetic diversity, compared with the other cGAs with elitism. However, Var2 records one failure for each in the population size of more than 60. Besides, Var2 is relatively slower than the original cGA, cGABV, and Var1 to achieve the successful runs. Fig. 6 represents the change of the degree of diversity versus the number of function evaluations on the OneMax problem with the population size of 10.

Table 4
Results for f_1. The number of successful runs and the average of the number of function evaluations called for successful runs are reported according to the population size. '–' represents that the corresponding algorithm fails to obtain the global optimum.

Pop. size	Org. cGA	pe-cGA	ne-cGA	cGABV	cGABVE	Var1	Var2
10	–	–	–	1	–	50	50
	–	–	–	638	–	1252.56	1422.74
20	–	–	–	34	–	50	50
	–	–	–	744.5882	–	1519.56	1667.82
30	14	–	–	45	–	50	50
	923.2857	–	–	909.9556	–	1452.4	1902.08
40	37	–	–	48	–	**50**	50
	940.7027	–	–	1096.917	–	**1242.96**	2033.9
50	49	–	–	50	–	50	50
	1132.49	–	–	1264.24	–	1273.48	2229.54
60	48	–	–	50	–	50	49
	1344.167	–	–	1429.56	–	1352.96	2409.367
70	50	–	–	50	–	50	49
	1520.24	–	–	1608.32	–	1538.92	2566.347
80	50	–	–	50	–	50	49
	1710.76	–	–	1776.68	–	1747.68	2692.347
90	50	–	–	50	–	50	49
	1891.68	–	–	2006.64	–	1919.04	2875.837
100	50	–	–	50	–	50	49
	2101.68	–	–	2193.48	–	2111.88	3048.041

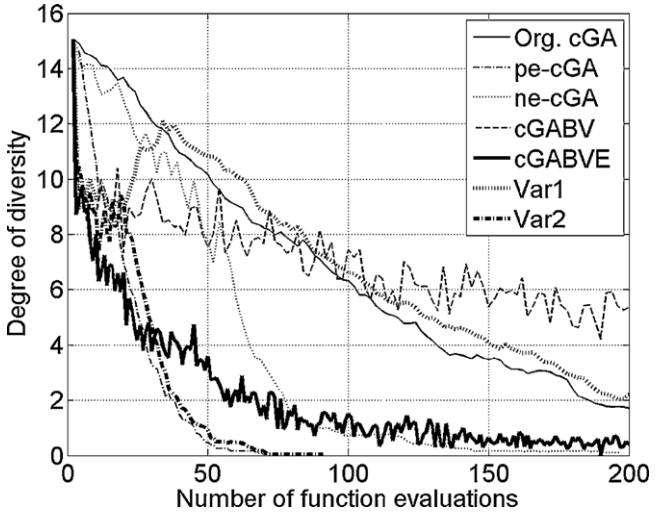

Fig. 6. (Population size = 10) The degree of diversity versus number of function evaluations on the OneMax problem (f_1). $\eta = 0.1$ and $\lambda = 10$ are applied to the ne-cGA and the proposed cGAs, respectively.

Fig. 6 shows the differences of the genetic diversity calculated by Eq. (7) among the conventional cGAs and the proposed cGAs. It is easy to show that the diversity curve of pe-cGA decreases most rapidly. Unlike the conventional cGAs, the diversity curves of the proposed cGAs have many oscillations in the beginning of the evolution. The diversity curves of Var1 and Var2 become similar with the curves of the original cGA and pe-cGA, respectively, as evolution proceeds. However, in the curves of Var1 and Var2, an increase of the diversity occasionally occurs in the middle of evolutionary process unlike in the original cGA and pe-cGA. These oscillations intuitively indicate the genetic diversity is maintained to attempt the exploitation. As a result, diversity maintenance leads the algorithms to overcome the local optima problems (i.e., premature convergence) and to find the global optimal solution. On the other hand, in the conventional cGAs, diversity curves consistently decrease without oscillations. Therefore, once individuals converge toward wrong directions, it may be hard for these individuals to escape the locally optimal solutions and to continuously progress toward the direction of global solutions. The decreasing rate is commonly proportional to the convergence speed and the selection pressure. For these reasons, it is shown that the original cGA, the proposed cGABV, the proposed Var1 without the elitism have high quality of solutions.

The next problem, called MDP (f_2), is formed by concatenating 10 copies of minimum deceptive function [9,13] (see Table 2) as in Eq. (8).

$$g_2(x_{2i}) = \begin{cases} 0.7, & \text{if} x_{2i} = 00 \\ 0.4, & \text{if} x_{2i} = 01 \\ 0.0, & \text{if} x_{2i} = 10 \\ 1.0, & \text{if} x_{2i} = 11 \end{cases} \tag{8}$$

As shown in Tables 5 and 6, the cGAs combined with the elitism have the poor performance in terms of quality of solutions as on the OneMax problem. The cGAs with the elitism require the less number of function evaluations for successful runs, whereas a success rate of the cGAs with the elitism is lower than the other cGAs without the elitism. In general, the proposed cGABV shows better performance than the others and shows the best performance especially in the population size of 60. As the population size increases, Var1 is also competitive with cGABV. In the diversity properties of the proposed cGABV and cGABVE (see Fig. 7), there are some oscillations as in the OneMax problem. The curves of the cGAs without the elitism display the analogous trend after about 1000 function evaluations. In the curve of Var2, it is shown that the diversity increases due to the adjustment of λ at about 1500 function evaluations. Diversity maintenance caused by the adjustment of λ leads Var2 to achieve more successful runs than the other elitism-based cGAs.

In summary, for solving the simple problems involving the lower BBs, the proposed cGAs without the elitism generally have the best performance in the aspect of quality of solutions. In other words, it indicates that high selection pressure due to the elitism provides the negative affect to solve f_1 and f_2. Var1 is superior among the cGAs without the elitism. Var2 also tries to overcome the negative affect due to the elitism by using the entropy-driven parameter control scheme. Consequently, the proposed cGAs employing the BV are competitive in compared with the conventional cGAs.

4.3. Results for the high-BBs problems

We consider 3-bit trap problem (f_3), one of the fully deceptive problems, as involving the high order BBs. In addition to the equation of Table 1, the problems is formulated by

$$g_3(x_{3i}) = \begin{cases} 0.35(2 - x_{3i}), & \text{if} x_{3i} \leq 2 \\ (x_{3i} - 2), & \text{otherwise.} \end{cases} \tag{9}$$

Tables 7 and 8 describe the results of the cGAs for f_3. As in the low-BBs problems, the cGAs without the elitism generally pro-

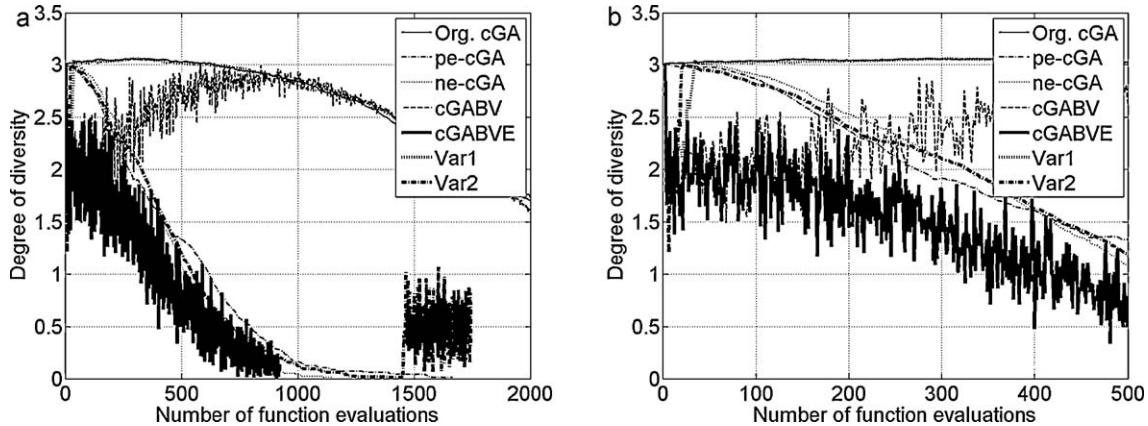

Fig. 7. (Population size = 200) The degree of diversity versus number of function evaluations on the MDP problem (f_2). $\eta = 0.1$ and $\lambda = 10$ are applied to the ne-cGA and the proposed cGAs, respectively. (b) depicts the enlargement of (a)).

Table 5
Results for f_2. Mean and standard deviation of the fitness error values are reported according to the population size.

Pop. size	Org. cGA		pe-cGA		ne-cGA		cGABV		cGABVE		Var1		Var2	
20	0.616	0.46	1.326	0.60	0.906	0.42	0.36	0.33	0.856	0.49	0.62	0.53	0.744	0.47
40	0.232	0.31	0.976	0.49	0.822	0.36	0.216	0.28	0.808	0.62	0.308	0.33	0.488	0.37
60	0.096	0.20	0.776	0.42	0.79	0.42	0.032	0.11	0.616	0.45	0.45	0.51	0.328	0.28
80	0.024	0.09	0.792	0.42	0.692	0.45	0.064	0.15	0.616	0.39	0.536	0.68	0.192	0.26
100	0.008	0.06	0.672	0.57	0.552	0.40	0.08	0.18	0.528	0.45	0.912	0.68	0.112	0.20
120	0.008	0.06	0.552	0.46	0.632	0.42	0	0.00	0.592	0.39	0.734	0.75	0.136	0.19
140	0	0.00	0.528	0.37	0.656	0.49	0	0.00	0.48	0.47	0.38	0.55	0.096	0.19
160	0.008	0.06	0.504	0.33	0.344	0.35	0	0.00	0.448	0.48	0.014	0.10	0.064	0.15
180	0	0.00	0.416	0.33	0.496	0.35	0	0.00	0.456	0.43	0	0.00	0.072	0.17
200	0	0.00	0.416	0.30	0.44	0.37	0	0.00	0.36	0.32	0	0.00	0.072	0.15

Table 6
Results for f_2. The number of successful runs and the average of the number of function evaluations called for successful runs are reported according to the population size.

Pop. size	Org. cGA	pe-cGA	ne-cGA	cGABV	cGABVE	Var1	Var2
20	14	1	1	31	3	25	7
	205.8571	77	56	351.5484	94.33333	387.36	354
40	39	2	2	36	7	47	11
	386.4615	129.5	161.5	503.5556	186.7143	701.4894	1061.818
60	45	2	5	**50**	10	48	18
	558.8	216.5	171	**660.4**	218	633.9167	886.7222
80	50	3	5	50	8	50	30
	689.12	207.3333	241	817.64	352.875	733.88	1194.5
100	50	9	10	50	14	50	37
	809.88	213.2222	281.5	1000.32	381.9286	861.52	1512.459
120	50	12	8	50	8	50	33
	983.72	334.5	265.75	1068.4	401	1001.8	1182.667
140	50	11	11	50	15	50	39
	1099.32	337.4545	324.2727	1198.96	398.4	1136.84	1496.333
160	50	10	19	50	17	50	42
	1261.96	407.4	343.9474	1412.56	404.1176	1211.36	1420.905
180	50	14	10	50	16	50	42
	1369.48	467.6429	334.1	1485.92	530.6875	1366.48	1636.905
200	50	12	16	50	17	50	41
	1561.84	476.0833	468.4375	1581.12	551.1765	1476.48	1670.683

vide better performance than the elitism-based cGAs. In particular, the original cGA and Var1 show the best performance in terms of quality of solutions and the number of function evaluations for a successful run. The proposed cGABV also shows a one-hundred percent success, but averagely requires the more number of function evaluations for a successful run in comparison with the original cGA and Var1. Accordingly, it indicates that the entropy-driven parameter control of Var1 lead cGABV to achieve the optimal solution more quickly. As shown in Fig. 8, the diversity curves of the proposed cGABV and cGABVE have many oscillation and slowly decreases as in the previous results. The curves of the cGAs with elitism and without elitism are, respectively, analogous to each other.

4.4. Results for the continuous and unimodal problems

In this subsection, we deal with continuous, convex and unimodal problems, representatively De Jong's function 1 problems (f_4). The results for this problems are reported in Tables 9 and 10. The proposed cGAs generally shows better performance than the conventional cGAs. The cGABVE and Var2 among the proposed cGAs provide the best performance in particular in terms of the convergence speed for successful runs as well as quality of solutions. It is shown that the elitism leads the proposed cGAs to improve the convergence speed without loss of quality of solutions. The proposed cGABVE gives fast convergence property in comparison with Var2, whereas a failure occurs sometimes in the

Table 7
Results for f_3. Mean and standard deviation of the fitness error values are reported according to the population size.

Pop. size	Org. cGA		pe-cGA		ne-cGA		cGABV		cGABVE		Var1		Var2	
300	0.091	0.154	0.658	0.435	0.833	0.499	0.091	0.169	0.357	0.346	0.306	0.788	0.007	0.049
600	0.028	0.118	0.609	0.362	0.532	0.404	0.007	0.049	0.14	0.210	0.014	0.069	0	0.000
900	0.007	0.049	0.511	0.409	0.476	0.369	0	0.000	0.084	0.179	0	0.000	0.525	0.379
1200	0	0.000	0.399	0.321	0.378	0.349	0	0.000	0.175	0.213	0	0.000	0.385	0.323
1500	0	0.000	0.336	0.232	0.315	0.307	0	0.000	0.112	0.191	0	0.000	0.308	0.286
1800	0	0.000	0.224	0.278	0.301	0.262	0	0.000	0.105	0.188	0	0.000	0.266	0.302
2100	0	0.000	0.231	0.285	0.231	0.294	0	0.000	0.112	0.191	0	0.000	0.231	0.285
2400	0	0.000	0.189	0.245	0.224	0.260	0	0.000	0.154	0.223	0	0.000	0.147	0.233
2700	0	0.000	0.133	0.209	0.231	0.228	0	0.000	0.098	0.172	0	0.000	0.168	0.213
3000	0	0.000	0.175	0.201	0.203	0.253	0	0.000	0.126	0.219	0	0.000	0.147	0.199

Table 8
Results for f_3. The number of successful runs and the average of the number of function evaluations called for successful runs are reported according to the population size.

Pop. size	Org. cGA	pe-cGA	ne-cGA	cGABV	cGABVE	Var1	Var2
300	**50**	7	5	50	17	50	49
	4767.72	936.1429	751.4	8327.32	2782.824	4959.84	7655.388
600	50	5	10	50	33	50	50
	8735.2	2259.2	1663.8	12990.04	4170.485	8566.36	9899
900	50	11	10	50	40	50	8
	12344.04	2973.182	2048.6	17296	4964.075	12646.84	2312.5
1200	50	13	16	50	28	50	15
	15984.72	3650.538	3442.25	20960.72	7060.643	16329.44	3079.733
1500	50	12	19	50	36	50	18
	20556.96	4377.583	4337.158	24901.2	6635.694	20229.56	3993.167
1800	50	27	17	50	37	50	24
	23977.4	5138.407	4902.882	27432.6	7835.757	24180.44	5256.417
2100	50	26	26	50	36	50	26
	26982.8	5963.808	5277.962	31756.2	8561.028	27872.44	5984.923
2400	50	29	24	50	32	50	33
	30925.84	6231.345	6354.708	35038.16	8650.938	31045.56	6126.818
2700	50	34	21	50	37	50	29
	34306.44	6650.647	7833.476	37690.08	9362.622	34865.6	7714.69
3000	50	27	27	50	35	50	31
	38840.92	7236.556	8415.519	40805.12	9809.343	38095.92	7962.452

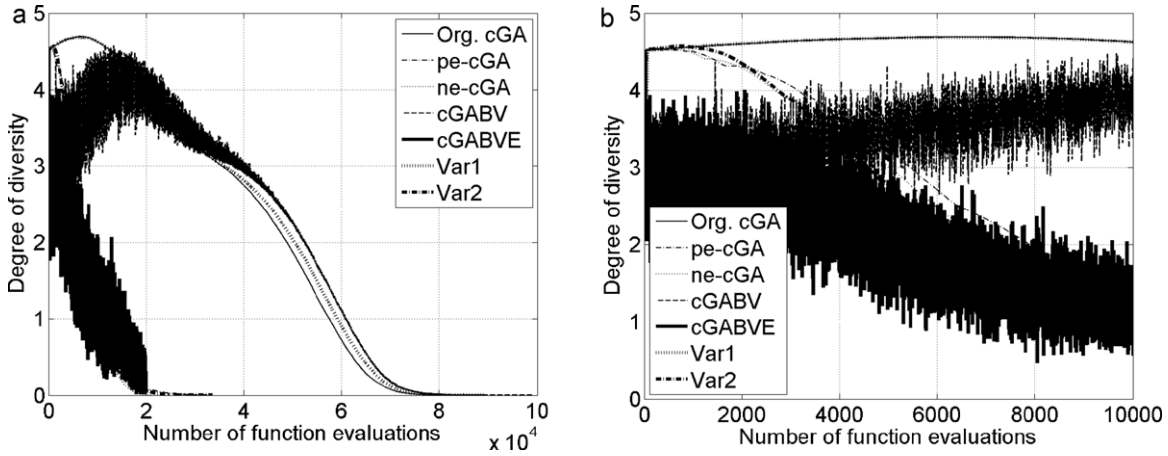

Fig. 8. (Population size = 3000) The degree of diversity versus number of function evaluations on the 3-bit trap problem (f_3). $\eta = 0.1$ and $\lambda = 10$ are applied to the ne-cGA and the proposed cGAs, respectively. (b) illustrates the enlargement of (a).

proposed cGABVE. Fig. 9 illustrates the diversity curves of the cGAs.

4.5. Results for the continuous and multimodal problems

Finally, we experiment the proposed cGAs with the continuous and multimodal functions in this subsection. All the functions used in this subsection have many local minima, namely highly multimodal, so that the cGA is likely to converge to the wrong directions.

Tables 11 and 12 shows the results for the Circle function problem (f_5). As shown in Table 12, there are no cGAs with the success rate of a one-hundred percent to achieve the fixed accuracy level. The success rates of the proposed Var1 and Var2 are relatively

Table 9
Results for f_4. Mean and standard deviation of the fitness error values are reported according to the population size.

Pop. size	Org. cGA		pe-cGA		ne-cGA		cGABV		cGABVE		Var1		Var2	
20	0.296	0.551	0.470	0.688	0.094	0.077	0.000	0.000	0.000	0.000	0.001	0.004	0.000	0.000
40	0.101	0.342	0.061	0.097	0.060	0.064	0.002	0.014	0.000	0.000	0.512	3.227	0.000	0.000
60	0.002	0.005	0.049	0.134	0.102	0.160	0.000	0.000	0.000	0.000	0.766	3.913	0.000	0.000
80	0.000	0.001	0.006	0.011	0.038	0.074	0.000	0.001	0.000	0.000	0.295	2.064	0.000	0.000
100	0.000	0.000	0.006	0.023	0.029	0.058	0.000	0.000	0.000	0.000	0.132	0.917	0.000	0.000
120	0.000	0.000	0.004	0.015	0.032	0.044	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
140	0.000	0.000	0.001	0.003	0.018	0.029	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
160	0.000	0.000	0.000	0.002	0.011	0.024	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
180	0.000	0.000	0.002	0.008	0.007	0.018	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
200	0.000	0.000	0.001	0.004	0.004	0.007	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Table 10
Results for f_4. The number of successful runs and the average of the number of function evaluations called for successful runs are reported according to the population size. '–' represents that the corresponding algorithm fails to obtain the global optimum.

Pop. size	Org. cGA	pe-cGA	ne-cGA	cGABV	cGABVE	Var1	Var2
20	1	–	–	34	49	50	**50**
	478	–	–	1444.588	655	1925.84	**753.36**
40	8	–	–	29	50	50	50
	1211.75	–	–	2141.862	899.2	3872.28	1169.14
60	16	–	–	36	50	50	50
	1964.5	–	–	2816.889	1023.28	4340.92	1592.78
80	32	–	–	37	50	50	50
	2729.813	–	–	3390.541	1202.42	4137.24	1908.86
100	44	–	–	43	50	40	50
	3298.045	–	–	4018.977	1325.3	3512.25	2287.08
120	44	1	1	47	49	45	50
	3910.273	1104	600	4585.021	1467.551	3913.022	2646.52
140	50	7	1	47	50	48	50
	4543.68	3594.714	614	5064.511	1624.68	4670.75	3071.7
160	46	7	1	49	50	50	50
	5096.565	4984.857	851	5774.245	1733.6	5072.96	3104.84
180	49	13	–	49	50	49	50
	5875.184	3591.154	–	6342.245	1870.34	5794.122	3369.26
200	50	8	–	50	49	50	50
	6578.12	1309	–	7072.28	1988.735	6480.6	3448.78

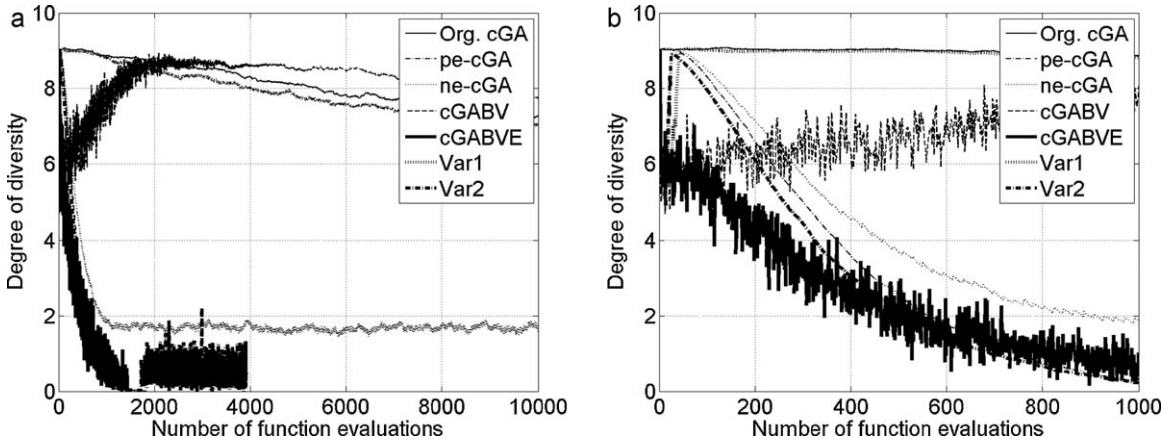

Fig. 9. (Population size = 200) The degree of diversity versus number of function evaluations on the De Jong's function 1 problem (f_4). $\eta = 0.1$ and $\lambda = 10$ are applied to the ne-cGA and the proposed cGAs, respectively. (b) illustrates the enlargement of (a).

higher than the others. In addition, Var2 averagely provides the minimum fitness error value in comparison with the others as shown in Table 11. The conventional pe-cGA and ne-cGA have no successful runs in all the population sizes, since the premature convergence tends to occur due to high selection pressure as shown in Fig. 10. On the other hand, the proposed cGABVE has a few successful runs in spite of the use of the elitism because of the diversity maintenance due to the oscillations of the diversity change. As shown in Fig. 10, the diversity curve shows the similar trend to the curve for f_4.

Tables 13 and 14 show the results for the Schaffer's binary function problem (f_6). As for the Circle function problem, there are no

Table 11
Results for f_5. Mean and standard deviation of the fitness error values are reported according to the population size.

Pop. size	Org. cGA		pe-cGA		ne-cGA		cGABV		cGABVE		Var1		Var2	
10	2.060	1.210	2.108	1.000	2.373	1.033	1.312	0.904	1.073	0.935	1.216	1.017	1.033	1.057
20	1.613	1.090	1.630	0.803	0.775	0.263	0.785	0.686	0.748	0.622	1.059	0.880	0.347	0.419
30	1.073	0.962	1.211	0.668	0.674	0.251	0.515	0.510	0.441	0.391	0.677	0.812	0.316	0.416
40	0.657	0.639	1.044	0.701	0.732	0.302	0.391	0.398	0.366	0.418	0.631	1.094	0.198	0.330
50	0.485	0.577	0.898	0.513	0.766	0.318	0.415	0.414	0.265	0.262	0.473	0.848	0.146	0.277
60	0.389	0.455	0.804	0.580	0.803	0.335	0.377	0.394	0.221	0.228	0.712	1.098	0.136	0.236
70	0.239	0.320	0.731	0.501	0.746	0.343	0.160	0.155	0.269	0.381	1.019	1.594	0.097	0.151
80	0.186	0.280	0.700	0.499	0.669	0.324	0.150	0.168	0.165	0.140	0.295	0.418	0.063	0.075
90	0.145	0.198	0.624	0.538	0.599	0.311	0.132	0.152	0.261	0.414	0.192	0.350	**0.049**	**0.039**
100	0.100	0.106	0.473	0.353	0.565	0.351	0.116	0.128	0.159	0.174	0.157	0.326	0.061	0.064

Table 12
Results for f_5. The number of successful runs and the average of the number of function evaluations called for successful runs are reported according to the population size. '–' represents that the corresponding algorithm fails to obtain the global optimum.

Pop. size	Org. cGA	pe-cGA	ne-cGA	cGABV	cGABVE	Var1	Var2
10	–	–	–	–	–	2	2
	–	–	–	–	–	7795	507.5
20	–	–	–	–	–	3	1
	–	–	–	–	–	3785.3333	641
30	–	–	–	1	–	1	4
	–	–	–	2146	–	2322	2071.25
40	–	–	–	1	–	5	5
	–	–	–	2154	–	5685.6	2264.2
50	–	–	–	3	1	4	1
	–	–	–	2202	767	3962.5	9847
60	2	–	–	–	1	3	4
	4189	–	–	–	1183	6404	2752.75
70	3	–	–	2	3	8	7
	3284.667	–	–	3687	1112	5195	3742.8571
80	6	–	–	3	1	4	4
	3294	–	–	3672.667	905	6075	5058
90	5	–	–	4	3	3	7
	3876.4	–	–	4256	1445	3775.3333	5009.5714
100	3	–	–	2	4	3	5
	2965.333	–	–	4853	1181.75	4593.3333	4345.2

Table 13
Results for f_6. Mean and standard deviation of the fitness error values are reported according to the population size.

Pop. size	Org. cGA		pe-cGA		ne-cGA		cGABV		cGABVE		Var1		Var2	
30	0.276	0.166	0.091	0.097	0.014	0.010	0.226	0.178	0.042	0.045	0.169	0.167	0.028	0.036
60	0.155	0.170	0.080	0.087	0.016	0.012	0.149	0.142	0.030	0.039	0.162	0.201	0.022	0.034
90	0.212	0.176	0.032	0.040	0.015	0.014	0.131	0.155	0.019	0.036	0.122	0.142	0.011	0.016
120	0.122	0.147	0.043	0.058	0.018	0.019	0.102	0.134	0.018	0.023	0.102	0.134	0.012	0.017
150	0.084	0.104	0.020	0.026	0.016	0.014	0.072	0.097	0.013	0.014	0.080	0.120	0.011	0.014
180	0.049	0.075	0.030	0.035	0.015	0.015	0.092	0.131	0.012	0.013	0.093	0.121	0.010	0.013
210	0.087	0.132	0.013	0.012	0.014	0.014	0.071	0.095	0.010	0.009	0.057	0.103	0.011	0.013
240	0.082	0.117	0.017	0.023	0.013	0.012	0.064	0.119	0.009	0.009	0.073	0.110	0.010	0.012
270	0.065	0.085	0.015	0.017	0.017	0.021	0.048	0.086	0.012	0.015	0.096	0.151	0.008	0.010
300	0.070	0.117	0.014	0.018	0.023	0.011	0.107	0.148	**0.008**	**0.006**	0.072	0.123	0.009	0.007

Table 14
Results for f_6. The number of successful runs and the average of the number of function evaluations called for successful runs are reported according to the population size. '–' represents that the corresponding algorithm fails to obtain the global optimum.

Pop. size	Org. cGA	pe-cGA	ne-cGA	cGABV	cGABVE	Var1	Var2
30	–	–	–	2	5	11	15
	–	–	–	3551	882.4	4659.0909	1533.2
60	1	–	–	–	8	6	17
	25936	–	–	–	920.125	9231	2089.8235
90	2	–	–	5	12	7	13
	8467	–	–	7600	1371.583	15660.571	3050.4615
120	2	4	–	2	11	3	14
	6087	14485.5	–	9017	1309.091	6904	2970.5714
150	–	1	–	5	8	1	13
	–	953	–	12077.6	1638.75	7610	3534.3077
180	3	3	–	2	9	2	14
	12012	16957.67	–	18683	1730.444	13469	4400.7143
210	1	6	1	1	12	4	13
	22706	7953.667	1096	17780	1802.333	13135.5	4760.0769
240	5	3	–	7	14	2	11
	18775.2	5646	–	18755.43	2174.286	16226	3418.0909
270	3	7	–	4	10	4	19
	21984.67	7688.429	–	22408.5	2369	17208	4438.6842
300	5	6	2	3	10	2	11
	23057.6	5767.667	1546.5	24009.33	2915.4	24530	3447.4545

Table 15
Results for f_7. Mean and standard deviation of the fitness error values are reported according to the population size.

Pop. size	Org. cGA		pe-cGA		ne-cGA		cGABV		cGABVE		Var1		Var2	
30	3.933	2.702	4.735	2.332	9.090	2.121	2.038	1.261	1.065	1.373	2.298	2.737	1.366	1.285
60	2.842	2.439	4.880	2.161	7.836	2.224	1.847	1.404	0.579	1.030	1.947	3.490	0.887	1.244
90	1.309	1.304	3.333	1.788	6.765	1.934	1.741	1.344	0.718	1.161	1.699	3.466	0.557	1.062
120	0.895	1.189	3.177	1.538	6.209	2.271	1.061	1.227	0.661	1.130	0.923	1.114	0.495	0.992
150	0.698	1.789	2.749	1.611	4.949	2.111	0.744	1.205	0.732	1.176	0.465	0.813	0.148	0.551
180	0.367	0.837	2.264	1.243	4.740	2.033	0.589	1.028	0.628	1.056	0.408	0.901	0.124	0.494
210	0.049	0.324	1.828	1.312	3.977	1.583	0.421	0.915	0.362	0.856	0.318	0.757	0.058	0.394
240	0.254	0.703	1.542	1.228	3.444	1.646	0.127	0.503	0.405	0.897	0.110	0.440	**0.001**	**0.000**
270	0.095	0.456	1.909	1.236	3.505	1.631	0.338	0.808	0.455	0.956	0.002	0.001	0.174	0.589
300	0.056	0.328	1.719	1.157	3.052	1.635	0.141	0.552	0.324	0.861	0.239	0.667	0.260	0.708

Table 16
Results for f_7. The number of successful runs and the average of the number of function evaluations called for successful runs are reported according to the population size. '–' represents that the corresponding algorithm fails to obtain the global optimum.

Pop. size	Org. cGA	pe-cGA	ne-cGA	cGABV	cGABVE	Var1	Var2
30	–	–	–	11	31	14	23
	–	–	–	5376.545	1711.839	5715	3769.7391
60	2	–	–	16	37	25	33
	16077	–	–	7060.875	2273.459	9393.6	6097.8485
90	9	–	–	18	36	30	39
	12681.78	–	–	9008.111	2978.278	14078.933	6872.5128
120	14	1	–	24	33	23	40
	12470.43	39322	–	10278.42	3302.909	9669.7391	11244.95
150	34	–	–	33	33	28	46
	8825	–	–	11882.85	3835.576	8703.8571	10547.674
180	41	2	–	35	34	37	47
	10316.54	30319	–	13258.57	4120.618	10348.216	12434.936
210	47	3	–	39	38	41	49
	11611.7	29466	–	14729.28	4447.026	11906.732	15349.224
240	43	1	–	47	36	47	**50**
	13326.33	27181	–	16106.72	4862.333	13092.085	**13210.7**
270	48	1	–	42	31	50	46
	14472.46	2517	–	17476.76	4836.194	14434.36	13600.435
300	48	–	–	47	30	44	39
	16128.42	–	–	18905.96	5463.5	15736.818	13259

cGAs with a one-hundred percent success. The proposed cGAs combined with the elitism, cGABVE and Var2, are relatively superior to the others as shown in Table 14. In general, Var2 shows the most competitive performance in terms of the number of successful runs and mean of the fitness error values. As shown in Table 13, ne-cGA and cGABVE generally have better performance than the other conventional cGAs. As shown in Fig. 11, the diversity curve shows the similar trend to the curves for f_4 and f_5.

Tables 15 and 16 show the results for the Ackley's path function problem (f_7). Var1 with the population size of 270 and Var2 with

Table 17
Results of Wilcoxon rank-sum tests with the solutions obtained with ten kinds of population sizes for each problem. '–' represents the target for the Wilcoxon rank-sum test. Wilcoxon rank-sum test is performed between each algorithm and the target algorithm. $H = 1$ indicates a rejection of the null hypothesis at the 1% significance level, and vice versa.

		Org. cGA	pe-cGA	ne-cGA	cGABV	cGABVE	Var1	Var2
f1	Mean/stdv	1.838/4.047	15.610/5.062	20.314/6.972	0.318/0.889	9.658/2.610	0.648/0.975	0.010/0.100
	p-Value/*H*	0/1	0/1	0/1	0/1	0/1	0/1	–
f2	Mean/stdv	0.099/0.268	0.696/0.517	0.633/0.439	0.075/0.196	0.576/0.478	0.395/0.581	0.230/0.340
	p-Value/*H*	0.586/0	0/1	0/1	–	0/1	0/1	0/1
f3	Mean/stdv	0.013/0.069	0.347/0.357	0.372/0.381	0.001/0.062	0.146/0.231	0.032/0.266	0.218/0.297
	p-Value/*H*	0.460/0	0/1	0/1	–	0/1	0.158/0	0/1
f4	Mean/stdv	0.040/0.224	0.060/0.263	0.039/0.077	0.000/0.005	0.000/0.000	0.171/1.776	0/0
	p-Value/*H*	0/1	0/1	0/1	0/1	0.025/0	0/1	–
f5	Mean/stdv	0.695/0.943	1.022/0.800	0.870/0.673	0.435/0.582	0.397/0.535	0.643/0.989	0.245/0.503
	p-Value/*H*	0/1	0/1	0/1	0/1	0/1	0/1	–
f6	Mean/stdv	0.120/0.150	0.035/0.057	0.015/0.015	0.106/0.141	0.017/0.027	0.103/0.145	0.013/0.021
	p-Value/*H*	0/1	0/1	0/1	0/1	0.008/1	0/1	–
f7	Mean/stdv	1.050/1.923	2.814/1.981	5.357/2.743	0.905/1.274	0.593/1.083	0.841/2.049	0.407/0.926
	p-Value/*H*	0/1	0/1	0/1	0/1	0/1	0/1	–

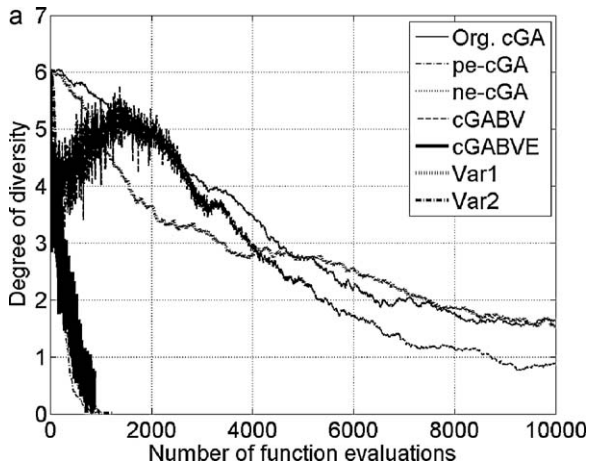

Fig. 10. (Population size = 100) The degree of diversity versus number of function evaluations on the Circle function problem (f_5). $\eta = 0.1$ and $\lambda = 10$ are applied to the ne-cGA and the proposed cGAs, respectively.

the population size of 240 provide a one-hundred percent success. The latter has faster convergence property and smaller mean value than the former. Fig. 12 illustrates the diversity curves of the cGAs

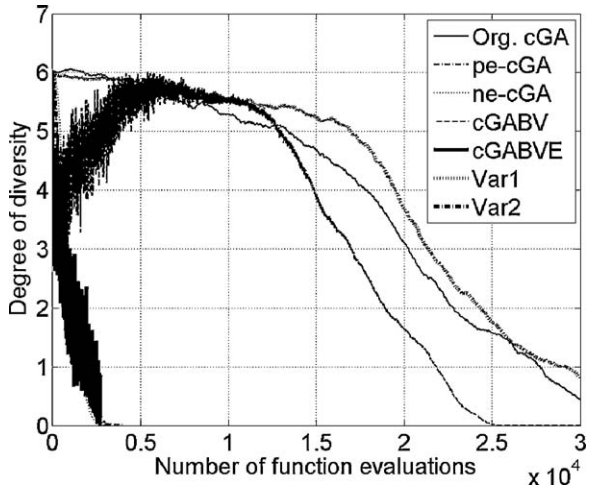

Fig. 11. (Population size = 300) The degree of diversity versus number of function evaluations on the Schaffer's binary function problem (f_6). $\eta = 0.1$ and $\lambda = 10$ are applied to the ne-cGA and the proposed cGAs, respectively.

according to the number of function evaluations. This figure also shows the similar trend to the curves for f_4 and f_5. However, it is shown that ne-cGA attempts the exploitation as the diversity increases during evolution. Since the elite individual is removed at this moment in ne-cGA, the poor performance is obtained for f_7 in which many local optima exist. For this reason, η of ne-cGA should be tuned for f_7 additionally. For investigating the effect of λ, the additional experiments with f_7 are performed in the next subsection with changing λ.

4.6. Summary of the results

As shown in the results, the proposed cGAs outperform the conventional cGAs in general. To verify the results, the hypothesis tests are carried out with the fitness function error values of all the solutions obtained from each cGA on each problem. Since solutions of each cGA being compared follow a non-normal distribution, the Wilcoxon rank-sum test, which is one of the best-known non-parametric significance tests, is utilized. This test is a non-parametric test for assessing whether two independent samples of observations come from the same distribution [24]. This test is virtually identical to performing an ordinary parametric two-sample *t*-test on the data after ranking over the combined samples. α for tests is 0.01. Table 17 shows the results of the Wilcoxon rank-sum tests. Consequently, Var2 is superior to the others in the problems except f_2 and f_3. For f_2 and f_3, the original cGA and cGABV are superior to the others.

4.7. Effects of λ

As mentioned above, we investigate the effects of λ through the additional experiments in this subsection. The additional experiments are performed for the Ackley's path function problem (f_7).

4.7.1. Changes in the diversity

Firstly, Fig. 13 shows that the diversity curve is changed according to changing λ. As mentioned previously, if λ is increasing, the diversity curves are more steeply decreasing with less oscillations. In other words, it means that the proposed cGAs have the similar behavior of the original cGA or pe-cGA in terms of genetic diversity. As mentioned previously, the diversity property of the proposed cGA is affected by the variation of λ related with the change of the variance determining the shape of a belief function (see Fig. 4). Accordingly, it describes that, since λ directly affects the diversity of the cGAs, a relevant λ is required for defining the belief function of the proposed cGAs in order to control the algorithm according to

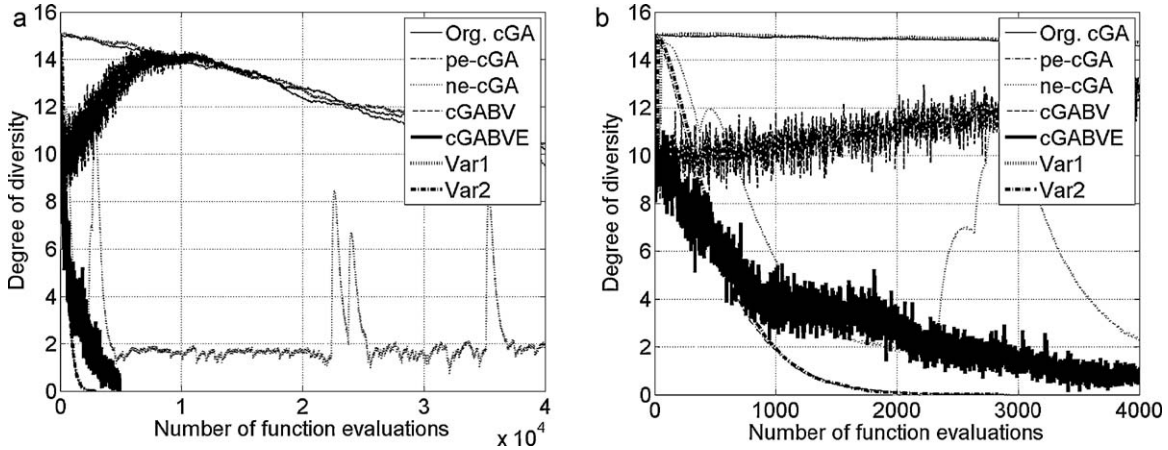

Fig. 12. (Population size = 300) The degree of diversity versus number of function evaluations on the Ackley's path function problem (f_7). $\eta = 0.1$ and $\lambda = 10$ are applied to the ne-cGA and the proposed cGAs, respectively. (b) illustrates the enlargement of (a).

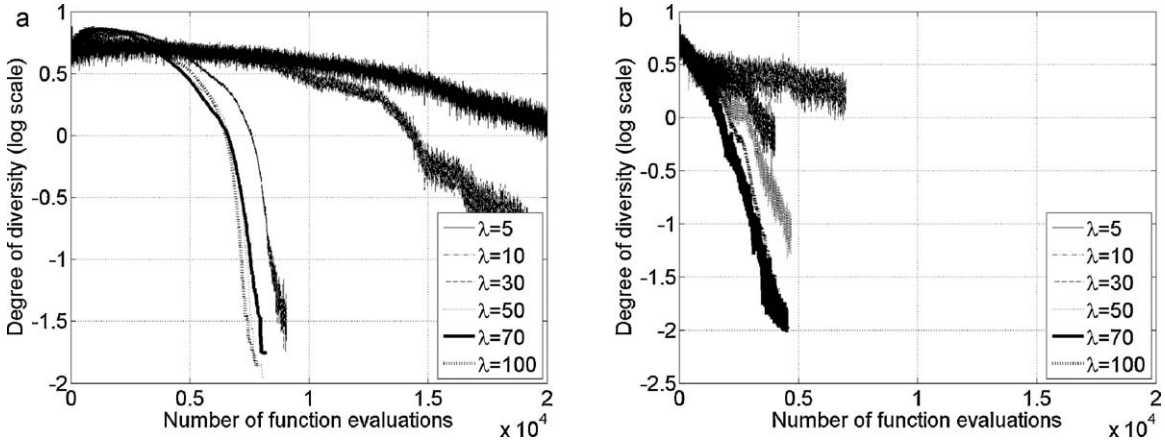

Fig. 13. The degree of diversity versus number of function evaluations on the Ackley's path function problem (f_7) according to the λ in case of population size 300: (a) cGABV and (b) cGABVE.

a certain problem. It also indicate that λ of the proposed cGAs using the BV indirectly affect the algorithm by controlling the diversity of each generation. For the Ackley's path function problem (f_7), when λ = 10 or 5, diversity curves are too slowly decreasing as shown in Fig. 13 so that it is likely to fail to find the optimal solution in the search space because of the complexity of the problem.

4.7.2. Changes in the performance

Fig. 14 describes the performances of the proposed cGAs according to λ for the Ackley's path function problem (f_7). Here, Figs. and show the results of the proposed cGABV. Figs. and show the results of the proposed cGABVE. In case of the proposed cGABV, if λ are decreasing as 5, the proposed cGABV produces poor quality of solutions. Furthermore, as mentioned above, the diversity curve is too slowly converging as shown in Fig. 13 so that the property of convergence also becomes quite worse. On the other hand, if λ is increasing, the proposed cGABV becomes similar to the original cGA, because the effect caused by the uncertainty of the BV is early reduced as in the original cGA. However, when the population size is less than about 100, the proposed cGABV is superior to the any other cGAs. Unlike the proposed cGABV, if λ are decreasing in the proposed cGABVE, the fast convergence is obtained with loss of

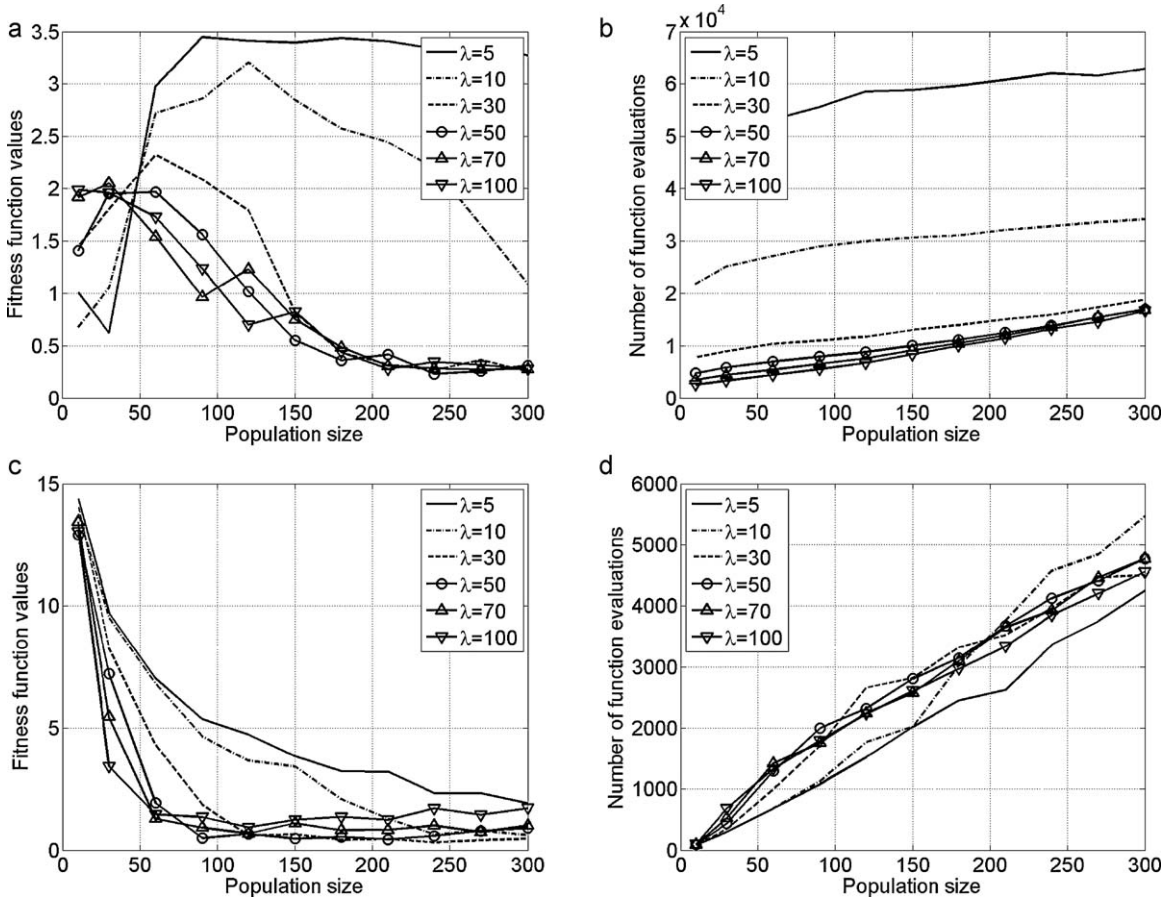

Fig. 14. The performances of the proposed cGAs on the Ackley's Path function problem (f_7). ((a) and (b)) The proposed cGABV. ((c) and (d)) The proposed cGABVE. ((a) and (c)) Fitness function values versus the population size. ((b) and (d)) Number of function evaluations versus the population size.

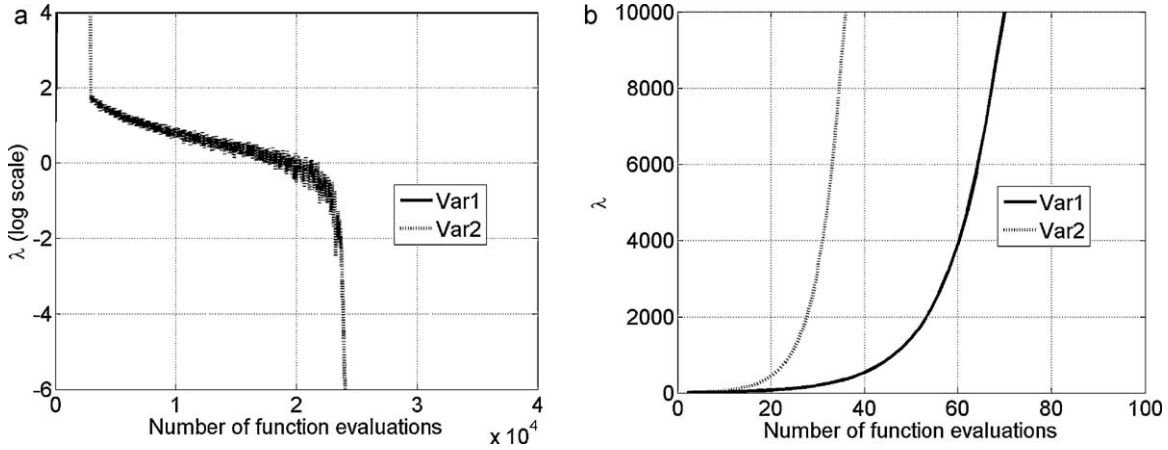

Fig. 15. Change of λ in the proposed variants of cGAs for the Ackley's Path function problem (f_7): (a) change of λ and (b) change of λ at the beginning.

quality of solutions. On the other hand, if λ is increasing, the proposed cGABVE becomes similar to the pe-cGA, because the effect caused by the uncertainty of the BV is early reduced as in the proposed cGABV. If the λ is predefined as about 50 or 70, the proposed cGAs may have the best and the most well-balanced performance, compared with the other cGAs, because of the highest quality of solutions with the minimum loss of the convergence speed.

4.7.3. Changes of λ in Var1 and Var2

As shown in the results, the entropy-driven parameter control scheme for λ leads the proposed cGAs to overcome the negative affects due to the elitism and to improve the convergence speed. For the Ackley's path function problem (f_7), Fig. 15 shows the change of λ in Var1 and Var2 with the population size of 300. At the beginning of evolution, λ of both the proposed variants increases exponentially. The difference between Var1 and Var2 appears after about 2000 function evaluations. λ of Var1 is maintained as the large value, whereas λ of Var2 starts to decrease.

5. Conclusion

This paper presents the combination of the probability distribution with the PV of the cGA. We focus on the diversity maintenance to improve the original cGA using PVs. The BV is newly proposed instead of the PV to control generic diversity and we also propose the cGABV and cGABVE using the BV. For evaluating the genetic diversity, the concept of entropy is employed. However, the additional control parameter λ should be appropriately predefined by users according to the given problems for the proposed cGABV and cGABVE. In order to avoid the difficulty of parameter tuning, we additionally propose two variants, Var1 and Var2, employing the entropy-driven control scheme for cGABV and cGABVE, respectively. The experimental results show that the proposed cGAs commonly has the highest quality of solutions regardless of the benchmark problems. For the problems involving the building blocks, Var1 generally give the best performance in comparison with the others. For the continuous problems, Var2 provides the better performance than the other cGAs in terms of the quality of solutions and the convergence property for the successful runs. To verify the experimental results, the Wilcoxon rank-sum tests are performed with the fitness function error values of all the solutions obtained from each cGA on each problem. From the results of the hypothesis tests, it is shown that Var2 is the most competitive on except for f_2 and f_3 in comparison with the other cGAs. For f_2 and f_3, the proposed cGABV and original cGA are superior to the others. The additional experiment is carried out for investigating the effects of λ and these results show that λ can directly control the diversity maintenance. In addition, the change of λ in Var1 and Var2 employing the entropy-driven parameter control scheme is also investigated.

References

[1] G.R. Harik, F.G. Lobo, D.E. Goldberg, The compact genetic algorithm, IEEE Transactions on Evolutionary Computation 3 (4) (1999) 287–297.
[2] C.W. Ahn, K.P. Kim, R.S. Ramakrishna, A memory-efficient elitist genetic algorithm, Lecture Notes in Computer Science (2004) 552–559.
[3] J.C. Gallagher, S. Vigraham, G. Kramer, A family of compact genetic algorithms for intrinsic evolvable hardware, IEEE Transactions on Evolutionary Computation 8 (2) (2004) 367–385.
[4] J.I. Hidalgo, J. Lanchares, A. Ibarra, R. Hermida, A hybrid evolutionary algorithm for multi-FPGA systems design, in: Euromicro Symp. on Digital System Design (DSD), Dortmund, Germany, 2002, pp. 60–67.
[5] J.I. Hidalgo, M. Prieto, J. Lanchares, R. Baraglia, F. Tirado, O. Garnica, Hybrid parallelization of a compact genetic algorithm, in: Proceedings. 11th Euromicro Conference on Parallel, Distributed and Network-Based Processing, Genova, Italy, 2003, pp. 449–455.
[6] R. Baraglia, J.I. Hidalgo, R. Perego, A hybrid heuristic for the traveling salesman problem, IEEE Transactions on Evolutionary Computation 5 (6) (2001) 613–622.
[7] E. Mininno, F. Cupertino, D. Naso, Real-valued compact genetic algorithms for embedded microcontroller optimization, IEEE Transactions on Evolutionary Computation 12 (2) (2008) 203–219, doi:10.1109/TEVC.2007.896689.
[8] M.H. Jin, W.Z. Liu, D.F. Hsu, C.Y. Kao, Compact genetic algorithm for performance improvement in hierarchical sensor networks management, in: ISPAN '05: Proceedings of the 8th International Symposium on Parallel Architectures, Algorithms and Networks, Las Vegas, USA, 2005, pp. 406–413.
[9] C.W. Ahn, R.S. Ramakrishna, Elitism-based compact genetic algorithms, IEEE Transactions on Evolutionary Computation 7 (4) (2003) 367–385.
[10] C. Zhou, K. Meng, Z. Qiu, Compact genetic algorithm mutated by bit, in: Proceedings of the 4th World Congress on Intelligent Control and Automation, Shanghai, China, 2002, pp. 1836–1839.
[11] J.Y. Lee, S.M. Im, J.J. Lee, Bayesian network-based non-parametric compact genetic algorithm, in: INDIN 2008. 6th IEEE International Conference on Industrial Informatics, 2008, pp. 359–364.
[12] G.R. Harik, Linkage learning via probabilistic modeling in the ecga, IlliGAL Report No. 99010, 1999.
[13] C.W. Ahn, R.S. Ramakrishna, Augmented compact genetic algorithm, Lecture Notes in Computer Science 3019 (2004) 560–565.
[14] S. Rimcharoen, D. Sutivong, P. Chongstitvatana, Updating strategy in compact genetic algorithm using moving average approach, in: in:2006 IEEE Conference on Cybernetics and Intelligent Systems (CIS2006), Bangkok, Thailand, 2006, pp. 1–6.
[15] A. Ekart, S. Nemeth, Maintaining the diversity of genetic programs, in: Proceedings of the 5th European Conference on Genetic Programming, London, UK, 2002, pp. 162–171.
[16] S. Rudlof, M. Koppen, Stochastic hill climbing with learning by vectors of normal distributions, in: The first Online Workshop on Soft Computing, Nagoya, Japan, 1996.
[17] M. Pelikan (Ed.), Hierarchical Bayesian Optimization Algorithm, Springer Berlin, Heidelberg/St. Louis, USA, 2005.
[18] J.P. Rosca, Entropy-driven adaptive representation, in: Workshop Genetic Programming: From Theory to Real-World Applications, Tahoe City, CA, 1995, pp. 23–32.

[19] D. Chen, C. Zhao, Particle swarm optimization with adaptive population size and its application, Applied Soft Computing 9 (1) (2009) 39–48, doi:10.1016/j.asoc.2008.03.001.
[20] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery (Eds.), Numerical Recipes in C++: The Art of Scientific Computing, 2nd ed. February, Cambridge University Press, 2002.
[21] S.-H. Liu, M. Mernik, B.R. Bryant, Entropy-driven parameter control for evolutionary algorithms, Informatica 31 (1) (2007) 41–50.
[22] H. Pohlheim, Geatbx: Example functions (single and multi-objective functions) 2 parametric optimization (October, 2006). http://www.geatbx.com/docu/fcnindex-01.html.
[23] D.E. Goldberg (Ed.), Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, 1989.
[24] J. Gibbons, S. Chakraborti, Nonparametric Statistical Inference, CRC, 2003.