Elitism-Based Compact Genetic Algorithms

Chang Wook Ahn, *Student Member, IEEE*, and R. S. Ramakrishna, *Senior Member, IEEE*

***Abstract*—This paper describes two elitism-based compact genetic algorithms (cGAs)—*persistent elitist compact genetic algorithm* (pe-cGA), and *nonpersistent elitist compact genetic algorithm* (ne-cGA). The aim is to design efficient compact-type GAs by treating them as estimation of distribution algorithms (EDAs) for solving difficult optimization problems without compromising on memory and computation costs. The idea is to deal with issues connected with lack of memory—inherent disadvantage of cGAs—by allowing a selection pressure that is high enough to offset the disruptive effect of uniform crossover. The point is to properly reconcile the cGA with elitism. The pe-cGA finds a near optimal solution (i.e., a winner) that is maintained as long as other solutions (i.e., competitors) generated from probability vectors are no better. It attempts to adaptively alter the selection pressure according to the degree of problem difficulty by employing only the pair-wise tournament selection strategy. Moreover, it incorporates the equivalent model of the $(1 + 1)$ evolution strategy (ES) with self-adaptive mutation. The pe-cGA, apart from providing a high performance, also reveals the hidden connection between EDAs (e.g., cGA) and ESs (e.g., $(1 + 1)$-ES). On the other hand, the ne-cGA further improves the performance of the pe-cGA by avoiding strong elitism that may lead to premature convergence. The ne-cGA comes with all the benefits of the pe-cGA. In addition, it maintains genetic diversity as a bonus. This paper also proposes an analytic model for investigating convergence enhancement (i.e., *speedup*).**

Experimental results show that the proposed algorithms, ne-cGA in particular, generally exhibit a better quality of solution and a higher rate of convergence for most of the problems than do the existing cGA, sGA, and $(1 + 1)$-ES. The speedup model has been verified by experiments. The results also show that an adequate alleviation of elitism further improves the solution quality, as well as the convergence speed.

***Index Terms*—Compact genetic algorithms, elitism, genetic diversity, selection pressure, speedup.**

I. Introduction

GENETIC ALGORITHMS (GAs) are stochastic search mechanisms. They are inspired by the mechanics of (Darwinian) natural selection and genetics [1]. They have been successfully used in a wide variety of applications in business, engineering, and science [1], [2]. Of all the issues connected with GAs—such as population size, genetic operators (e.g., selection, crossover, and mutation), and encoding methods, etc.,—the population size that guarantees an optimal solution quickly enough has been a topic of intense research [3]–[8]. This is because large populations generally result in better solutions, but at increased computational costs and memory requirements. Goldberg *et al.* [3] developed the first population-sizing equation based on the variance of fitness. They further enhanced the equation that permits accurate statistical decision making among competing building blocks (BBs) [4]. However, if wrong BBs are chosen in the first generation, the GAs will never recover [5], [6]. Extending the decision model in [4], Harik *et al.* [5] exploited the similarity between the gambler's ruin problem and the selection mechanism of GAs for determining an adequate population size that guarantees a solution with the desired quality. Furthermore, the analytic model started from the assumption that the fitness values of a pair of chromosomes can be ordered. This effectively implies tournament selection without replacement. Moreover, Ahn and Ramakrishna [7] further enhanced and generalized the population sizing equation [5] so as to dispense with any (problem dependent) stochastic information such as signal or collateral noise of competing BBs. In an attempt to understand the real importance of population in evolutionary algorithms (EAs), He and Yao [8] showed that the introduction of population increases the first hitting probability, so that the mean first hitting time is shortened.

Based on the results in [5], on the other hand, Harik *et al.* [9] proposed the compact GA (cGA) as an estimation of distribution algorithm (EDA) that generates offspring population according to the estimated probabilistic model of parent population instead of using traditional recombination and mutation operators [10], [11]. The cGA represents the population as a probability (distribution) vector (PV) over the set of solutions and operationally mimics the order-one behavior of simple GA (sGA) with uniform crossover using a small amount of memory. Therefore, the cGA may be quite useful in memory-constrained applications such as multicast routing and resource allocation problems in the emerging field of wireless networks. When confronted with easy problems (e.g., continuous-unimodal problems) involving lower order BBs, the cGA can achieve solutions of comparable quality with approximately the same number of fitness evaluations as the sGA with uniform crossover [9]. However, the cGA does not provide acceptable solutions to difficult problems (e.g., deceptive problems or multimodal problems), because it does not have the memory to retain the required knowledge (e.g., decision error, linkage information of genes) about nonlinearity of the problems [9]. These problems involve higher order BBs. It is noted that most practical applications may come within the purview of difficult optimization problems because they usually have many local optima (i.e., multimodal) and the genes are interdependent in general (i.e., it is difficult to model the problems as the combination of lower order BBs). It follows that the cGA may not be effective in solving real-world problems. In order to obtain better solutions to such difficult problems, the cGA should exert a higher selection pressure. This, in turn, increases

Manuscript received October 30, 2002; revised February 10, 2003 and April 12, 2003. This work was supported in part by the International Research Program of the Kwang-Ju Institute of Science and Technology (K-JIST).

The authors are with the Department of Information and Communications, Kwang-Ju Institute of Science and Technology (K-JIST), Gwang-Ju 500–712, Korea (e-mail: cwan@kjist.ac.kr; rsr@kjist.ac.kr).

Digital Object Identifier 10.1109/TEVC.2003.814633

the survival probability of higher order BBs, thereby preventing loss of the best solution found so far. In other words, higher selection pressure may play the role of memory. Therefore, it can take care of a finite number of decision errors and some linkage information of genes. Selection pressure of the cGA can be increased by creating a larger tournament size in a simple manner [9]. However, this scheme requires additional (but insignificant) memory that is proportional to tournament size s. Furthermore, it is difficult to precompute the (average) order of BBs in practical applications. Even if the order of BBs can be found (or is known) in advance, the tournament size that provides the selection pressure high enough to compensate for the highly disruptive effects of uniform crossover cannot be determined precisely. Although Harik *et al.* [9] investigated a relation between tournament size and selection pressure (in the cGA) by employing a global schema theorem, the relation was verified only in the context of a specific problem involving concatenation of ten copies of a three-bit, fully deceptive function with deceptive-to-optimal ratio of 0.7. This implies that the relation may only be partially satisfied. This is because the tournament size is closely related to not only the order of BBs but also to other factors (such as deceptive-to-optimal ratio and collateral noise). The problem with Harik's result is demonstrated through experiments in Section IV-B, but a detailed investigation is beyond the scope of this paper.

Furthermore, Harik *et al.* [12] proposed an extended compact GA (ecGA) for solving difficult problems such as fully deceptive problems by combining a greedy marginal product model (MPM) search algorithm with a minimal description length (MDL) search model. Although the MPMs are similar to the PV of cGA with regard to the products of marginal distributions on a partition (i.e., a BB) of the genes, they can provide a direct linkage map with each partition separating tightly linked genes [13]. Thus, the ecGA can find better solutions to difficult problems with a smaller number of function evaluations (than the sGA). However, it requires more memory and carries higher computational costs per function evaluation.

Baraglia *et al.* [14] proposed a hybrid heuristic algorithm that combines cGA with an efficient Lin-Kernighan (LK) local search algorithm; it is called cGA-LK. The aim of cGA-LK is to deal with difficult order-k $(k > 1)$ optimization problems such as the traveling salesman problem (TSP) without requiring larger memory than the existing cGA. The cGA-LK exploits the cGA in order to generate high-quality solutions (to TSP), which are then refined with the LK local search algorithm. The refined solutions are in turn exploited further with a view to improving the quality of the simulated population (i.e., the probabilities of PV). Thereby, it achieves a performance that is better than is possible with sGA and cGA in terms of quality of solutions. However, the algorithm may incur an unacceptably high-computational cost because it employs the complex LK local search algorithm.

In the same context, Hidalgo *et al.* [15] devised a hybrid algorithm for multi-FPGA partitioning. The mechanism that combines the existing cGA and a random local search algorithm is quite similar to that of cGA-LK. Every time a certain number of epochs elapses, the best individual competes with a new individual found by local search. If the new individual has a higher fitness, then the PV (of cGA) is updated by its traits.

In this paper, we propose two new elitism-based compact GAs that belong to a class of EDAs—the persistent elitist compact GA (pe-cGA), and the nonpersistent elitist compact GA (ne-cGA). The main objective is to efficiently solve difficult optimization problems using the cGA without unduly compromising on memory and computational costs. In this paper, difficult problems have the following characteristics: 1) full deception; 2) interdependence (of decision variables); 3) multimodality; and 4) symmetry. The goal can be accomplished by correcting inherent defects of the cGA such as deficient memory by employing elitism in an ingenious manner. The pe-cGA deals with the problem of lack of memory by simply retaining the best solution found so far, thereby mitigating the disruptive effects of uniform crossover. That is, the pe-cGA can adequately increase its selection pressure by evolving a simulated population (i.e., the PV) by means of only two competing chromosomes. It is interesting to note that the pe-cGA operates as if the tournament size s (that is related closely to the order of BBs, the deceptive-to-optimal ratio, and the number of collateral noise) is automatically altered in tune with the degree of difficulty of the problem. It is noteworthy that the pe-cGA does not require any problem dependent information that may not be available in practice. Moreover, the pe-cGA obeys a model that is equivalent to the $(1+1)$ evolution strategy (ES) with self-adaptive mutation. The result can be interpreted as a revelation of the relationship between EDAs and ESs. In addition, the pe-cGA offers some advantages over the $(1+1)$-ES to GA practitioners and designers. On the other hand, it can be further improved by controlling the strength of elitism. The scheme is named ne-cGA. Since the ne-cGA also incorporates elitism (in a restricted manner), it carries nearly all the characteristics of the pe-cGA. Moreover, the restricted elitism plays a role in arresting the rapid degeneration of genetic diversity, thereby improving the quality of solution. Of course, the genetic diversity in EDAs can be defined as the average entropy of elements of PV; a detailed investigation of this issue is beyond the scope of this paper, however. The only difference from the pe-cGA is in the operational mechanism, whereby a chromosome that is randomly generated replaces the elite chromosome when a certain criterion indicating the allowable length of the elite chromosome's inheritance is not satisfied. The reason why the ne-cGA can outperform the pe-cGA is because (strong) elitism may incur high selection pressure, thereby leading to premature convergence [16]. Moreover, an analytic model with regard to convergence improvement is also suggested and verified in this paper. The improvement is measured by the ratio of the number of function evaluations of cGA to that of pe-cGA. This is the *speedup*.

The rest of the paper is organized as follows. Section II briefly describes the existing cGA and elitism. In Section III, the proposed elitism-based compact GAs for efficiently solving difficult problems are described. The speedup model is also presented in this section. Section IV presents the results obtained with the algorithms on several test functions/problems and Ising spin-glasses (ISG) systems (a real-world application), thus providing a basis for a comparative study. The paper concludes with a summary of the results in Section V.

Parameters. n : population size, l : chromosome length

Step 1. Initialize probability vector
for $i := 1$ *to* l *do* $p[i] := 0.5$;

Step 2. Generate two chromosomes from the probability vector
$a :=$ **generate** (p) ; $b :=$ **generate** (p) ;

Step 3. Let them compete
$winner, loser :=$ **compete** (a, b) ;

Step 4. Update the probability vector
for $i := 1$ *to* l *do*
if $winner[i] \neq loser[i]$ *then*
if $winner[i] == 1$ *then* $p[i] := p[i] + 1/n$;
else $p[i] := p[i] - 1/n$;

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

Step 6. The probability vector represents the final solution.

Fig. 1. Pseudocode of the cGA.

II. Compact Genetic Algorithm and Elitism

This section provides background information on cGA and elitism.

A. Compact Genetic Algorithm (cGA)

The cGA manages its population as a PV over the set of solutions (i.e., only models its existence), thereby mimicking the order-one behavior of the sGA with uniform crossover using a small amount of memory [9], [14].

Fig. 1 describes pseudocode of the cGA. The values of PV $p_i \in [0, 1], \forall i = 1, \ldots, l$, where l is the number of genes (i.e., the length of the chromosome), measures the proportion of "1" alleles in the ith locus of the simulated population [9], [14]. The PV is initially assigned 0.5 to represent a randomly generated population. In every generation (i.e., iteration), competing chromosomes are generated on the basis of the current PV, and their probabilities are updated to favor a better chromosome (i.e., winner). It is noted that the generation of chromosomes from PV simulates the effects of crossover that leads to a decorrelation of the population's genes. In a simulated population of size n, the probability p_i is increased (decreased) by $1/n$ when the ith locus of the winner has an allele of "1" ("0") and the ith locus of the loser has an allele of "0" ("1"). If both the winner and the loser have the same allele in each locus, then the probability remains the same. This scheme is equivalent to (steady-state) pair-wise tournament selection [9]. The cGA is terminated when all the probabilities converge to zero or one. The convergent PV itself represents the final solution. It is seen that the cGA requires $l * \log_2(n + 1)$ bits of memory while the sGA requires $l * n$ bits [9]. Thus, a large population size can be effectively exploited without unduly compromising on memory requirements [14].

On the other hand, the cGA simulates higher selection pressure to solve problems with higher order BBs. Selection pressure of the cGA can be increased by replacing ***Steps 2–4*** in Fig. 1 with the procedures described as follows [9].

First, s chromosomes are generated from the PV and the best one is found. Second, the best chromosome competes with the other $(s - 1)$ chromosomes and the PV is updated on the way. However, it requires bothersome information such as the order of BBs and the tournament size (that closely adjusts the selection pressure that is sufficient to combat the highly disruptive effects of uniform crossover). Since such information may not be available in practice, the cGA may not be that useful.

B. Elitism

As an operational characteristic of GAs, elitism provides a means for reducing genetic drift by ensuring that the best chromosome(s) is allowed to pass/copy their traits to the next generation [16], [17]. Genetic drift is used to explain/measure stochastic changes in gene frequency through random sampling of the finite population [18]. Some genes of chromosomes may turn out to be more important to the final solution than others [17]. When the chromosomes representing decision variables that have a reduced "salience" to the final solution do not experience sufficient selection pressure, genetic drift may be stalled. Therefore, it is important to maintain adequate selection pressure, as demanded by the application, in order to avoid this [17]. In other words, the arrest of genetic drift reflects the failure to exert adequate selection pressure by increasing the tournament size or by some form of elitism.

Since elitism can increase the selection pressure by preventing the loss of low "salience" genes of chromosomes due to deficient selection pressure, it improves the performance with regard to optimality and convergence of GAs in many cases. However, the degree of elitism should be adjusted properly and carefully because high selection pressure may lead to premature convergence [16].

III. Proposed Elitism-Based Compact Genetic Algorithms

This section describes the persistent elitist compact GA (pe-cGA), and the nonpersistent elitist compact GA (ne-cGA). A speedup model is also suggested.

A. Proposed GAs

The proposed pe-cGA and ne-cGA combine the existing cGA with elitism in an effective manner. The major objective is to improve the quality of solution and the rate of convergence (to the global optimum) with acceptable memory and computational costs. Since the cGA operates on each gene independently, it may lose linkage information. As a consequence, the cGA may not be able to solve difficult problems, especially those involving higher order BBs (e.g., deceptive problems).

In Section II, we found that the selection pressure of cGA should be proportional to the degree of difficulty of problems for efficiently solving them. In other words, a more difficult problem requires a higher selection pressure for finding a better solution. This is because higher selection pressure offsets the disruptive effects of uniform crossover (i.e., it carries partial knowledge about the gene's correlation such as the linkage information), thereby encouraging convergence to a better solution. Although the selection pressure of the cGA can be

Parameters. E_{chrom}: elite chromosome, N_{chrom}: new chromosome

Step 2*. Generate one chromosome from the probability vector
if the first generation *then*
$E_{chrom} := \textbf{generate}(p)$; /* *initialize the elite chromosome* */
$N_{chrom} := \textbf{generate}(p)$; /* *generate a new chromosome* */

Step 3*. Let them compete and let the winner inherit persistently
$winner, loser := \textbf{compete}(E_{chrom}, N_{chrom})$;
$E_{chrom} := winner$; /* *update the elite chromosome* */

Fig. 2. Modification of the cGA that realizes the pe-cGA.

increased by creating a larger tournament size, it requires additional (by no means significant) memory and problem-dependent information that is not generally available in real-world problems. Even if such information is available, computation of the necessary tournament size that builds a selection pressure that is high enough (to offset the crossover disruption) is not easy. As a result, the selection pressure should be adaptively adjusted in response to the degree of difficulty of the problems without actually varying the tournament size.

Since pair-wise tournament selection has been employed, the selection pressure should be adaptively increased by evolving only two competing chromosomes. It has already been shown that the selection pressure is also increased by passing the best chromosome(s) onto the next generation (i.e., elitism in Section II-B). Therefore, the idea is to adequately increase the selection pressure in accordance with the difficulty of the problems by employing elitism in an appropriate manner. In order to accomplish this, ***Steps 2–3*** of the cGA in Fig. 1 should be replaced by the ones described in Fig. 2.

The procedures (of Fig. 2) are being added (by the authors) with a view to simulating elitism in the cGA. Of the two competing chromosomes, only the loser is replaced by the new one that is generated from the PV. In other words, the winner is never eliminated in so far as a better chromosome has not yet been produced from the PV. This scheme is called *persistent elitist compact GA* (pe-cGA).

The following proposition is important in this regard.

Proposition 1: The pe-cGA is equivalent to the $(1+1)$-Evolutionary Strategy (ES) with self-adaptive mutation.

Proof: Let g: $\mathbf{R}^m \rightarrow \mathbf{R}$ be the objective function to be maximized. Consider the Markovian process $(\mathbf{X}_k)_{k\geq 0}$ generated by the stochastic algorithm [19]

$$\mathbf{X}_{k+1} = \begin{cases} \mathbf{X}_k + l_k\mathbf{Z}_k, & \text{if } g(\mathbf{X}_k + l_k\mathbf{Z}_k) > g(\mathbf{X}_k) \\ \mathbf{X}_k, & \text{otherwise} \end{cases} \quad (1)$$

where l_k is the step length control parameter that is increased as far as mutation improves solutions. Each random vector $\mathbf{Z}_k$ (of the sequence of independent and identically distributed random vectors) has a joint probability density function (pdf) with independent marginal densities.

The model of (1) falls within the purview of the $(1+1)$-ES with self-adaptive mutation, if the step length control parameter is changed when the relative frequency of improving mutations is below or above some threshold within τ trials [19]. In other words, (1) can exactly model the $(1+1)$-ES with self-adaptive mutation if it considers the τ trials as an elementary event of stage k.

Now, let $\mathbf{Y}_k$ be a random vector generated from the PV. The probability distribution of $\mathbf{Y}_k$ is given by

$$F_{\mathbf{Y}_k}(y_1, \ldots, y_m) = \prod_{i=1}^{m} \mathbf{P}_k(i) \quad (2)$$

where $\mathbf{P}_k(i)$ represents the ith element of the PV in the kth generation.

By employing $\mathbf{Y}_k$, on the other hand, the pe-cGA can be described by

$$\mathbf{X}_{k+1} = \begin{cases} \mathbf{Y}_k, & \text{if } g(\mathbf{Y}_k) > g(\mathbf{X}_k) \\ \mathbf{X}_k, & \text{otherwise} \end{cases}. \quad (3)$$

Here, $\mathbf{Y}_k$ describes a new chromosome generated from the PV. Moreover, $\mathbf{X}_k$ represents a winner (i.e., the elite chromosome) in the $(k-1)$th generation, (viz., a chromosome in the kth generation that is inherited from $(k-1)$th generation). Then, (3) can naturally be rewritten as

$$\mathbf{X}_{k+1} = \begin{cases} \mathbf{X}_k + (\mathbf{Y}_k - \mathbf{X}_k), & \text{if } g(\mathbf{Y}_k) > g(\mathbf{X}_k) \\ \mathbf{X}_k, & \text{otherwise} \end{cases}. \quad (4)$$

The random vector $\mathbf{Z}_k$ and its scaling constant l_k in (1) are employed. Since $\mathbf{Y}_k$ is also a random vector and $\mathbf{X}_k$ is a constant vector in the kth generation, one can relate $\mathbf{Z}_k$ with $\mathbf{Y}_k$ as follows:

$$l_k\mathbf{Z}_k = \mathbf{Y}_k - \mathbf{X}_k. \quad (5a)$$

Here, the pdf of $\mathbf{Z}_k$ can be computed easily as

$$f_{\mathbf{Z}_k}(z_1, \ldots, z_m) = f_{\mathbf{Z}_k}(\mathbf{z}) = |l_k| f_{\mathbf{Y}_k}(l_k\mathbf{z} + \mathbf{X}_k). \quad (5b)$$

Employing (4) and (5), (3) can be rewritten as

$$\mathbf{X}_{k+1} = \begin{cases} \mathbf{X}_k + l_k\mathbf{Z}_k, & \text{if } g(\mathbf{X}_k + l_k\mathbf{Z}_k) > g(\mathbf{X}_k) \\ \mathbf{X}_k, & \text{otherwise} \end{cases}. \quad (6)$$

Comparing (1) and (6), we can conclude that the two algorithms (i.e., the pe-cGA, the $(1+1)$-ES with self-adaptive mutation) follow an identical model. ■

The general tendency is to treat EDAs and ESs as belonging to different realms of evolutionary algorithms. *Proposition 1* is interesting in that it reduces an EDA with elitism to a certain ES. Thus, the pe-cGA not only achieves a higher performance but also opens up an avenue for examining the unsuspected relationship between EDAs and ESs.

Instead of gaining in selection pressure, on the other hand, the pe-cGA may lose genetic diversity owing to implied elitism. From *Proposition 1*, however, we see that the pe-cGA maintains genetic diversity that is comparable with that of $(1+1)$-ES with self-adaptive mutation. Furthermore, the pe-cGA offers some additional benefits over $(1+1)$-ES to GA practitioners and designers. First, the step length control parameter l_k is adjusted in a dynamic manner. Second, the pe-cGA does not have to select and fix the multivariate probability distribution for generating the random vector $\mathbf{Z}_k$. In other words, the mutation distribution of pe-cGA can also be adaptively adjusted.

In the $(1+1)$-ES with self-adaptive mutation, however, the update rule for l_k should be effective and the probability distribution for $\mathbf{Z}_k$ should be selected with care. This is because

Parameters.
θ : the present length of inheritance, η : the allowable length of inheritance

Step 2**. Generate one chromosome from the probability vector
if the first generation *then*
$\theta := 0$; /* *initialize the control parameter* */
$E_{chrom} :=$ **generate** (p) ; /* *initialize the elite chromosome* */
$N_{chrom} :=$ **generate** (p) ; /* *generate a new chromosome* */

Step 3**. Let them compete and let the winner inherit nonpersistently
$winner, loser :=$ **compete** (E_{chrom}, N_{chrom}) ;
if $\theta \le \eta$ & $winner == E_{chrom}$ *then*
$E_{chrom} := winner$; /* *update the elite chromosome by a winner* */
$\theta++$; /* *increment the control parameter* */
/* *replace the winner as a new randomly generated chromosome* */
else $E_{chrom} :=$ **generate** $(prob. := 0.5)$;
$\theta := 0$; /* *reinitialize the control parameter* */

Fig. 3. Modification of the cGA that realizes the ne-cGA.

they directly and critically affect the performance of the algorithm. Popular choices for mutation distribution are Gaussian and Cauchy distributions [19], [20].

Lee and Yao [21] developed an EA using (stable) Lévy distribution with different values of parameters in this regard. The objective is to adaptively alter the mutation distribution in the environment. However, the parameter of Lévy distribution is not self-adaptive and the adjusted mechanism is applied only to the variation of the decision variables, not to the self-adaptive deviation (i.e., l_k) [21].

On the other hand, strong elitism may lead to premature convergence (to a suboptimal solution). This is because a high selection pressure (brought about by strong elitism) results in the population's reaching equilibrium very fast, but it inevitably sacrifices genetic diversity. Thus, a parameter η that indicates an allowable length of the elite chromosome's inheritance is introduced to control the strength of elitism. This parameter restrains the length of inheritance (i.e., the number of generations) of the winner, thereby playing a role in retrieving genetic diversity to some extent. This scheme is called *nonpersistent elitist compact GA* (ne-cGA). In order to employ elitism in a nonpersistent manner, the ***Steps 2–3*** in Fig. 1 should be modified by the procedures described in Fig. 3.

As in pe-cGA, the loser is always replaced by a new candidate generated from the current PV. However, the winner can be passed on to the next generation only when the present length of inheritance, denoted by θ, does not exceed the allowable length of inheritance (i.e., η). In other words, a new randomly generated chromosome can replace the winner if $\theta > \eta$. Physically, this is very similar to mutation or random immigrant mechanism in GAs. Hence, this strategy may gently nudge the simulated population toward restoration of genetic diversity. Moreover, the ne-cGA carries almost all the characteristics of the pe-cGA because it also employs elitism (the strength is restricted, though).

The following proposition relates to the length of inheritance η.

Proposition 2: The allowable length of inheritance (η) should not exceed the simulated population size n. That is, $\eta < n$.

Proof: The letters $\mathbf{X}_k$ and $\mathbf{Y}_k$ are as in *Proposition 1*. Define $\mathbf{W}_k$ as a random vector generated from a random PV set to 0.5. Let $\mathbf{V}_k$ be another random vector defined as $(\mathbf{P}_{k+1} - \mathbf{P}_k)$. This vector represents the changes between intergeneration PVs. Let us assume that the winner takes the PV to convergence regardless of optimality of the solution. In other words, it is assumed that the (current) winner always defeats its competitor when the PV converges, irrespective of whether the solution is optimal or suboptimal.

The evolution of PV can be described by

$$\mathbf{P}_{k+1}(i) = \begin{cases} \mathbf{P}_k(i) + E[\mathbf{V}_k(i) \,|\, \mathbf{X}_k(i) = 1], & \text{if } \mathbf{X}(i) = 1 \\ \mathbf{P}_k(i) + E[\mathbf{V}_k(i) \,|\, \mathbf{X}_k(i) = 0], & \text{if } \mathbf{X}_k(i) = 0 \end{cases}. \tag{7}$$

Each conditional expectation on $\mathbf{V}_k(i)$ can be computed as follows:

$$E[\mathbf{V}_k(i) \,|\, \mathbf{X}_k(i) = 1] = (1/n) \cdot p[\mathbf{V}_k(i) = 1/n \,|\, \mathbf{X}_k(i) = 1] + 0 \cdot p[\mathbf{V}_k(i) = 0 \,|\, \mathbf{X}_k(i) = 1] \tag{8a}$$

$$E[\mathbf{V}_k(i) \,|\, \mathbf{X}_k(i) = 0] = (-1/n) \times p[\mathbf{V}_k(i) = -1/n \,|\, \mathbf{X}_k(i) = 0] + 0 \cdot p[\mathbf{V}_k(i) = 0 \,|\, \mathbf{X}_k(i) = 0]. \tag{8b}$$

On the other hand, each conditional probability is seen to be

$$p[\mathbf{V}_k(i) = 1/n \,|\, \mathbf{X}_k(i) = 1] = p[\mathbf{Y}_k(i) = 0] = 1 - \mathbf{P}_k(i) \tag{9a}$$

$$p[\mathbf{V}_k(i) = 0 \,|\, \mathbf{X}_k(i) = 1] = p[\mathbf{Y}_k(i) = 1] = \mathbf{P}_k(i) \tag{9b}$$

$$p[\mathbf{V}_k(i) = -1/n \,|\, \mathbf{X}_k(i) = 0] = p[\mathbf{Y}_k(i) = 1] = \mathbf{P}_k(i) \tag{9c}$$

$$p[\mathbf{V}_k(i) = 0 \,|\, \mathbf{X}_k(i) = 0] = p[\mathbf{Y}_k(i) = 0] = 1 - \mathbf{P}_k(i). \tag{9d}$$

Let us consider (9a) first. To increase the ith element of PV by $1/n$ (i.e., $\mathbf{V}_k(i) = 1/n$) given that ith gene of the winner [i.e., $\mathbf{X}_k(i)$] has "1," the ith gene of its competitor [i.e., $\mathbf{Y}_k(i)$] should generate "0" because the winner in $(k-1)$th generation (i.e., $\mathbf{X}_k$) becomes a winner in the present kth generation [i.e., the chromosome is passed to the next, i.e., $(k+1)$th generation]. Since $\mathbf{P}_k(i)$ is t he probability that $\mathbf{Y}_k(i) = 1$, it is computed as $1 - \mathbf{P}_k(i)$. The rest of (9) can be explained in a similar manner.

Using (8) and (9), (7) can be rewritten as

$$\mathbf{P}_{k+1}(i) = \begin{cases} \mathbf{P}_k(i) + (1/n) \cdot \{1 - \mathbf{P}_k(i)\}, & \text{if } \mathbf{X}_k(i) = 1 \\ \mathbf{P}_k(i) - (1/n) \cdot \mathbf{P}_k(i), & \text{if } \mathbf{X}_k(i) = 0 \end{cases}. \tag{10}$$

Define $\mathbf{P}_k^M$ and $\mathbf{P}_k^m$ as $\max_{\forall j} \mathbf{P}_k(j)$ and $\min_{\forall j} \mathbf{P}_k(j)$, respectively. None of the elements in the PV should be brought to convergence by the same winner because there is no guarantee that the chromosome leads to an optimal solution. It means that $\mathbf{P}_k^M$ or $\mathbf{P}_k^m$ should not be taken to convergence to "1.0" or "0.0" by the same winner. Hence, follows (11a) and (11b):

$$\mathbf{P}_{k+\eta}^M < 1 \tag{11a}$$

$$\mathbf{P}_{k+\eta}^m > 0. \tag{11b}$$

Here, η is the allowable length of inheritance of the winner.

By employing (10), (11a) can be rewritten as follows:

$$\begin{aligned}\mathbf{P}_{k+\eta}^{M} &= (1-1/n)\cdot\mathbf{P}_{k+\eta-1}^{M} + 1/n \\ &= (1-1/n)^2\cdot\mathbf{P}_{k+\eta-2}^{M} + \{(1-1/n)+1\}\cdot(1/n) \\ &= \cdots \\ &= (1-1/n)^{\eta}\cdot\mathbf{P}_{k}^{M} + \{1-(1-1/n)^{\eta}\} \\ &\quad \text{(since the simulated population size } n \gg 1) \\ &\approx (1-\eta/n)\cdot\mathbf{P}_{k}^{M} + \eta/n < 1.\end{aligned}$$

Thus, the allowable length η is seen to satisfy

$$\eta < n. \tag{12a}$$

By a similar method, (11b) can be rewritten as

$$\begin{aligned}\mathbf{P}_{k+\eta}^{m} &= (1-1/n)\cdot\mathbf{P}_{k+\eta-1}^{m} \\ &= (1-1/n)^2\cdot\mathbf{P}_{k+\eta-2}^{m} \\ &= \cdots \\ &= (1-1/n)^{\eta}\cdot\mathbf{P}_{k}^{m} \\ &\approx (1-\eta/n)\cdot\mathbf{P}_{k}^{m} > 0.\end{aligned}$$

The allowable length η is also found to obey the inequality

$$\eta < n. \tag{12b}$$

By considering (12a) and (12b), it is seen that the allowable length η should not exceed the simulated population size n. ■

It follows from *Proposition 2* that the quality of solution found by ne-cGA is identical to that found by pe-cGA when $\eta = n$. This is empirically confirmed later (Section IV-E). It is obvious that the ne-cGA does not require any extra memory (as in the pe-cGA). Moreover, the ne-cGA is not demanding on computational costs (see Fig. 3).

B. Speedup Model

A *speedup* model (i.e., a gain in convergence speed) is presented. Due to the similarity of stochastic mechanisms of sGA and cGA, the respective convergence speeds would not be very different (especially as population size increases). At the very least, the speed of cGA is slightly higher than that of sGA. Furthermore, the convergence speed of ne-cGA is approximately the same as that of pe-cGA from a statistical point of view because the ne-cGA imposes a somewhat relaxed elitism on the pe-cGA. Therefore, speedup is defined as the ratio of the number of fitness evaluations of cGA (i.e., T_{cGA}) to that of pe-cGA (i.e., $T_{\mathrm{pe-cGA}}$).

That is

$$\text{Speedup} = \frac{T_{\mathrm{cGA}}}{T_{\mathrm{pe-cGA}}}. \tag{13}$$

Consider a well-known convergence model of population based GAs in the context of problems in which BBs are of equal salience, genes converge uniformly, and the fitness is distributed binomially. One-max problem is a typical example. It is formulated as follows:

$$f_{\mathrm{OneMax}} = \sum_{i=1}^{m} x_i \tag{14}$$

where m is the number of BBs (i.e., bits) and x_i is the value of ith gene.

From the characteristics of the problem, we can approximate the mean and variance of fitness of the population as a normal distribution with mean μ_t, and variance σ_t^2. Here, $\mu_t = mp_t$ and $\sigma_t^2 = mp_t(1-p_t)$, and p_t represents the proportion of correct BBs (of the population) in generation t. Mülenbein and Schlierkamp–Voosen [22] proposed a convergence equation for the problem and ordinal selection schemes as follows:

$$\mu_{t+1} = \mu_t + I\sigma_t. \tag{15}$$

Here, I is the selection intensity that is defined as the expected increase in the average fitness of a population after the selection operation.

Equation (15) leads to [22]

$$p_t = \frac{1}{2}\left[1+\sin\left(\frac{I}{\sqrt{m}}t + \arcsin(2p_0-1)\right)\right]. \tag{16}$$

The convergence time t_{conv} (i.e., the number of generations encountered before convergence occurs) can be shown to be [22]

$$t_{\mathrm{conv}} = \left(\frac{\pi}{2} - \arcsin(2p_0-1)\right)\frac{\sqrt{m}}{I}. \tag{17}$$

If the population size is n, the number T_{conv} of function evaluations performed before convergence is clearly given by

$$T_{\mathrm{conv}} = \left(\frac{\pi}{2} - \arcsin(2p_0-1)\right)\frac{n\sqrt{m}}{I}. \tag{18}$$

From (13) and (18), the speedup is seen to be

$$\text{Speedup} = \frac{T_{\mathrm{cGA}}}{T_{\mathrm{pe-cGA}}} = \frac{I_{\mathrm{pe-cGA}}}{I_{\mathrm{cGA}}}. \tag{19}$$

Hence, the speedup can also be computed as the ratio of selection intensity of pe-cGA to that of cGA. This is quite reasonable because the selection intensity is inversely proportional to the speed of convergence. Note that the selection intensity is required for computing the speedup. Unfortunately, it is not the same for different selection schemes. For a tournament selection of size s (generally used in simple GA), for example, the selection intensity I is given by [23]

$$I = \mu_{s:s} = s\int_{-\infty}^{\infty} x\phi(x)(\Phi(x))^{s-1}\,dx. \tag{20}$$

Here, $\Phi(x)$ $(\phi(x))$ is the normal distribution (density) function with zero mean and unit variance.

To find the selection intensity of cGA, let us consider the selection method that randomly chooses two individuals and reflects two copies of the better one on the population. This is equivalent to a steady-state pair-wise tournament selection [9]. Since there is no population of selected parents for a steady-state scheme, the selection intensity can be redefined as the expected fitness increase after n offsprings have been generated [24]. In the case of selection (of cGA) with a tournament size s, the best individual competes with the other $(s-1)$ individuals, updating the PV along the way [9]. It means that all the parents in these $(s-1)$ competitions (i.e., $2(s-1)$ individuals) are replaced by the copies of the best individual out of s individuals. Thus, the creation of n offsprings is equivalent to the execution

of $n/\{2(s-1)\}$ selections. Since one execution of the selection with tournament size s increases the expected fitness (for population size n) by $(1/n)\sum_{i=1}^{s-1}(\mu_{s:s}-\mu_{i:s})$, the selection intensity of cGA can be defined by

$$I_{\mathrm{cGA}} = \frac{1}{2(s-1)}\sum_{i=1}^{s-1}(\mu_{s:s}-\mu_{i:s}) \tag{21}$$

where, $\mu_{i:s}$ represents the expected fitness value of the ith ranked individual of a random sample of size s of a population.

The expected value of the ith-order statistic $\mu_{i:s}$ can be computed by [23], [24]

$$\mu_{i:s} = s\binom{s-1}{i-1}\int_{-\infty}^{\infty} x\phi(x)\Phi(x)^{i-1}(1-\Phi(x))^{s-i}\,dx. \tag{22}$$

When $s=2$ (i.e., a steady-state pair-wise tournament selection), the selection intensity is given as

$$\begin{aligned} I_{\mathrm{cGA}} &= \frac{\mu_{2:2}-\mu_{1:2}}{2} \\ &= \frac{2\int_{-\infty}^{\infty} x\phi(x)\Phi(x)\,dx - 2\int_{-\infty}^{\infty} x\phi(x)\{1-\Phi(x)\}\,dx}{2} \\ &= \int_{-\infty}^{\infty} x\phi(x)\{2\Phi(x)-1\}\,dx \\ &= 2\int_{-\infty}^{\infty} x\phi(x)\Phi(x)\,dx \approx 0.56. \end{aligned} \tag{23}$$

It is interesting to note that the selection intensity for the tournament selection (of sGA) with $s=2$ [see (20)] is exactly the same as that of cGA with $s=2$ (23). That is, convergence performances of sGA and cGA are identical when $s=2$. However, it is observed that the convergence speed of sGA is slightly lower than that of cGA [see Fig. 4(b)]. The discrepancy seems to be due to the inherent nature (a sort of hitchhiking) of the process of mixing of BBs.

On the other hand, the pe-cGA reflects two copies of a tournament winner on the population (i.e., it is equal to the steady-state pair-wise tournament selection of cGA). However, the winner is deterministically chosen as one of the competitors. In the context of a population, the selection mechanism can be approximated by a truncation selection. The corresponding truncation selection takes the top (1/5) individuals in a population as parents because a selected individual whose fitness value (of Gaussian distribution) is higher than the fitness of the top (1/5) individuals would be a winner with $\mathrm{prob.} \geq 0.98$ from the long-run behavior point of view.

Moreover, truncation selection that picks the top $(1/\delta)$ portion of the population as parents is equivalent to a (μ,λ) selection with $\mu=\lambda/\delta$ [25]. Therefore, the selection mechanism of pe-cGA amounts to the (μ,λ) selection with $\mu=\lambda/5$ (i.e., $\delta=5$). Since the selection intensity of (μ,λ)-selection can be given by $(\lambda/\mu)\cdot\phi(\Phi^{-1}(1-(\mu/\lambda)))$, the pe-cGA has the following intensity:

$$I_{\mathrm{pe-cGA}} = 5\cdot\phi(\Phi^{-1}(0.8)) \approx 1.46. \tag{24}$$

By substituting (23) and (24) in (19), the speedup is found to be

$$\mathrm{Speedup} = \frac{T_{\mathrm{cGA}}}{T_{\mathrm{pe-cGA}}} = \frac{I_{\mathrm{pe-cGA}}}{I_{\mathrm{cGA}}} \approx 2.61. \tag{25}$$

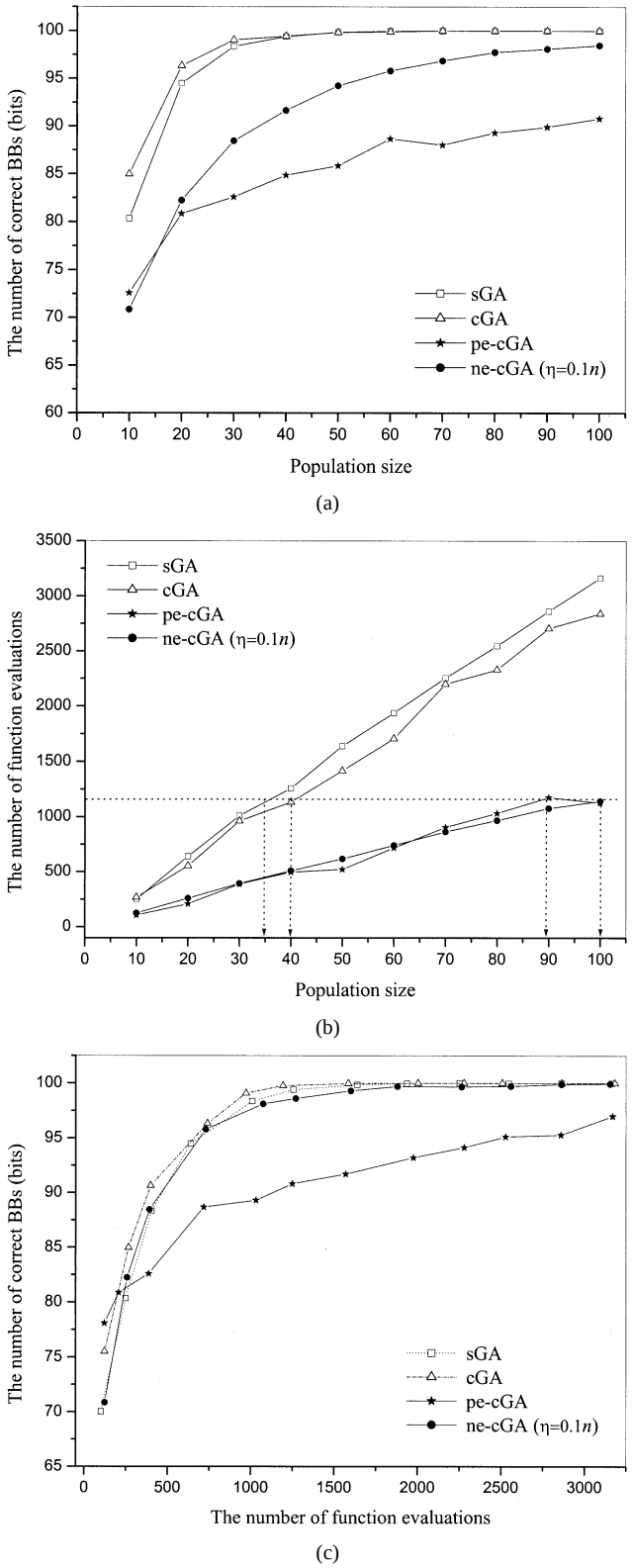


Fig. 4. Performance of the algorithms on f_{OneMax}. (a) Number of correct BBs versus population size. (b) Number of function evaluations versus population size. (c) Number of function evaluations versus number of correct BBs.

It is noted that the indicated assumption invalidates this model for the problems whose genes (i.e., BBs) are highly correlated and are of unequal salience. However, such characteristics carry a tendency that prevents the cGA from steadily converging to a

TABLE I
STATISTICAL COMPARISON OF ALGORITHMS ($n = 100$) WORKING ON f_{OneMax}, WHERE "STD" STANDS FOR THE STANDARD DEVIATION

GAs	sGA		cGA		pe-cGA		ne-cGA	
Statistic	Mean	STD	Mean	STD	Mean	STD	Mean	STD
	1936.2	74.76	1713.76	74.76	715.41	95.14	744.03	55.89

Cases	sGA – cGA	cGA – ne-cGA	ne-cGA – pe-cGA
t-test	21.039†	82.512†	2.594

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.01$ by a two-tailed test.

solution because of the absence of a strong reference for convergence. That is, the speedup plays a role in providing a lower bound on such problems (i.e., $\mathrm{Speedup} \geq 2.61$).

IV. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, the performance of pe-cGA and ne-cGA are compared with that of cGA, sGA, and $(1+1)$-ES on various test functions and ISG systems (a real-world application) through computer simulations. Fitness value (e.g., the number of correct BBs) and the number of (fitness) function evaluations are taken to be performance measures. The former considers solution quality (i.e., optimality) and the latter indicates the convergence performance.

In all the experiments, the sGA uses tournament selection without replacement and uniform crossover with exchange probability 0.5 [9]. The crossover is applied with probability one and the mutation probability is set to zero. The parameter η directly influencing the strength of elitism (i.e., selection pressure) is set to $0.1n$ as a default value. All the results were averaged over 100 runs. Each experiment is terminated when the PV converges to a solution.

A. Experimental Results for the Problems Involving Lower Order BBs

A 100-bit one-max problem (i.e., the counting ones problem) and a minimum deceptive problem (MDP) (formed by concatenating ten copies of minimum deceptive function) [1] are considered for evaluating the proposed algorithms on problems involving lower order BBs. The one-max problem and the MDP are representative problems with the order-one BBs and the order-two BBs, respectively.

A 100-bit one-max problem (i.e., f_{OneMax}) that is specified by (14) is considered first. Fig. 4(a)–(b) compares the number of correct BBs (i.e., bits) and the number of function evaluations returned by each algorithm as applied to one-max problem. On the face of it, the proposed algorithms do not find high quality solutions while convergence speeds are admittedly far higher (than those of sGA and cGA). Table I supports the claim on the improvement of convergence speed of the proposed algorithms over sGA and cGA. Moreover, the validity of the proposed speedup can also be concluded from Table VII.

In the interest of fair comparison of the algorithms on the basis of optimality and convergence performance, we investigate the number of correct BBs (solution quality) obtained by each population size that performs the same number of function evaluations [7], [26]. Since the population sizes of 35 (sGA), 40 (cGA), 90 (pe-cGA), and 100 (ne-cGA) perform approximately 1200 function evaluations [see Fig. 4(b)], for example, the solutions returned by the populations should be compared for investigating the superiority of the algorithms. Unfortunately, finding the exact population size for a particular execution for each GA is very difficult in practice. However, determining the population size with certain constraints is relatively easy. We can determine the population size for each GA by exhaustive search so as to achieve almost the same number of function evaluations. From this perspective, the number of correct BBs is plotted against function evaluations in Fig. 4(c). The figure shows that the sGA, cGA, and ne-cGA achieve similar quality of solution while the pe-cGA computes a worse solution. The reason is discussed below.

The pe-cGA may suffer from lack of genetic diversity leading to premature convergence since an elite chromosome is never lost unless a superior chromosome appears. This is the reason why the ne-cGA restricts the length of inheritance of the elite chromosome (i.e., the strength of elitism). In the figures, we may note that ne-cGA augments genetic diversity to some extent, thereby maintaining the quality of solution and the convergence speed at levels comparable with those of the reference algorithms. The pe-cGA, however, returns a poor overall performance. Note that the unsatisfactory performance of the pe-cGA may not be critical from a practical point of view because most real-world problems such as multicast routing, and (adaptive) equalizer design in fading channels, ISG systems, and maximum satisfiability (MAXSAT) problem, etc., cannot be modeled as combinations of order-one BBs.

The next test problem is a MDP defined by

$$f_{\mathrm{MDP}} = \sum_{i=1}^{m} f(x_{2i}), \quad \text{where } f(x_{2i}) = \begin{cases} 0.7, & \text{if } x_{2i} = 00 \\ 0.4, & \text{if } x_{2i} = 01 \\ 0.0, & \text{if } x_{2i} = 10 \\ 1.0, & \text{otherwise} \end{cases}. \tag{26}$$

Here, x_{2i} presents the values (i.e., alleles) of a 2-bit long substring (i.e., BB).

Fig. 5(a) and (b) depicts the quality of solution and the speed of convergence of the algorithms when applied to the MDP with $m = 10$ (i.e., 10-BBs). Trends similar to those found in Fig. 4(a) and (b) can be seen here as well. Table II also shows the higher convergence speed of the proposed algorithms over sGA and cGA. Moreover, accuracy of the model of speedup can be observed in Table VII. With regard to fair comparison, Fig. 5(c)

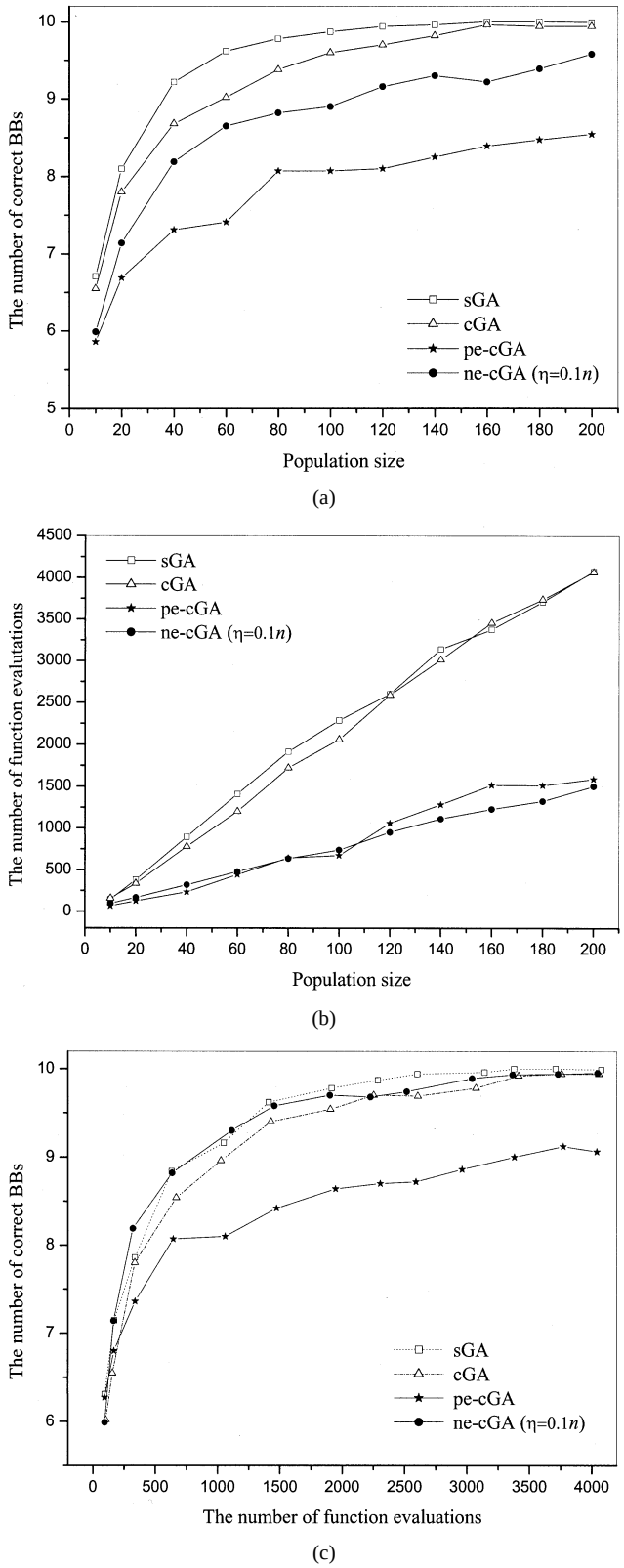


Fig. 5. Performance of the algorithms on f_{MDP}. (a) Number of correct BBs versus population size. (b) Number of function evaluations versus population size. (c) Number of correct BBs versus number of function evaluations.

shows the quality of solution versus the number of function evaluations. In the figure, it is seen that ne-cGA outperforms cGA and performs as well as sGA.

However, there is a difference with regard to pe-cGA. That is, the pe-cGA may be less effective than sGA and cGA on real-world problems because some of them can be modeled as combinations of order-two BBs (especially, the deceptive ones).

From Figs. 4 and 5, we may conclude that ne-cGA is a promising candidate for solving relatively simple problems as compared with sGA, cGA, and pe-cGA, even though a significant improvement of overall performance is not evident.

B. Experimental Results for the Problems Involving Higher Order BBs

Fully deceptive problems [27] are considered for testing pe-cGA and ne-cGA on the problems involving higher order BBs. Before investigating the performance, we define a trap function f_{trap} (a constituent of deceptive problems [27]) by

$$f_{\mathrm{trap}}(u,a,b,z,k)=\begin{cases}(a/z)\cdot(z-u), & \text{if } u\le z\\ \{b/(k-z)\}\cdot(u-z), & \text{otherwise}\end{cases} \tag{27}$$

where u is the unitation that is defined as the number of ones of a (sub)string, a and b are the local (i.e., deceptive) and the global optimum, respectively, z is the slope-change location, and k is the problem size.

The first deceptive problem is based on a three-bit trap function. The test problem is formed by concatenating ten copies of the three-bit trap function for a total chromosome length of 30 bits. Each three-bit trap function has a deceptive-to-optimal ratio of 0.7. That is, the problem is formulated by

$$f_{3-\mathrm{bit}}=\sum_{i=1}^{10} f_{\mathrm{trap}}(u_{3i},0.7,1,2,3) \tag{28}$$

where u_{3i} is the unitation of a 3-bit long substring.

Fig. 6(a) and (b) depicts the results of each algorithm as applied to the first deceptive problem. From Fig. 6(a), it is observed that pe-cGA achieves a better solution (than does cGA) with $s=8$ and ne-cGA finds a quality of solution that is comparable with that found by cGA with $s=16$. At the same time, their convergence speeds are higher than those of all the cGAs. From Fig. 6(b), it is also clear that the quality of solution found by the proposed algorithms is no better than that found by the sGA with $s=4$ while their convergence speeds are higher than those of sGAs except when $s=16$ for pe-cGA. Convergence performance is clearly seen in the statistical test in Table III. Moreover, Table VII also supports the accuracy of the speedup model.

The reason why the proposed algorithms do not find a better solution (than the sGA does with $s=4$) is because they lack the memory (comparing to the sGA $s\ge 4$) to retain the knowledge about nonlinearity of the problems. However, this is an inherent characteristic of all the compact-type GAs. It is also seen that the solution quality returned by the proposed algorithms improves as the population size grows. Thus, if a larger population is employed, a higher quality of solution is obtained. Note the important fact that the algorithms (i.e., pe-cGA, ne-cGA) exert a selection pressure that is high enough to combat the disruptive effects of crossover without altering the tournament size.

TABLE II
STATISTICAL COMPARISON OF ALGORITHMS ($n = 200$) WORKING ON f_{MDP}, WHERE "STD" STANDS FOR THE STANDARD DEVIATION

GAs	sGA		cGA		pe-cGA		ne-cGA	
Statistic	Mean	STD	Mean	STD	Mean	STD	Mean	STD
	4076.0	313.70	4066.4	395.16	1531.3	262.05	1505.2	225.90

Cases	sGA – cGA	cGA – pe-cGA	pe-cGA – ne-cGA
t-test	0.191	52.162†	2.463

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.01$ by a two-tailed test.

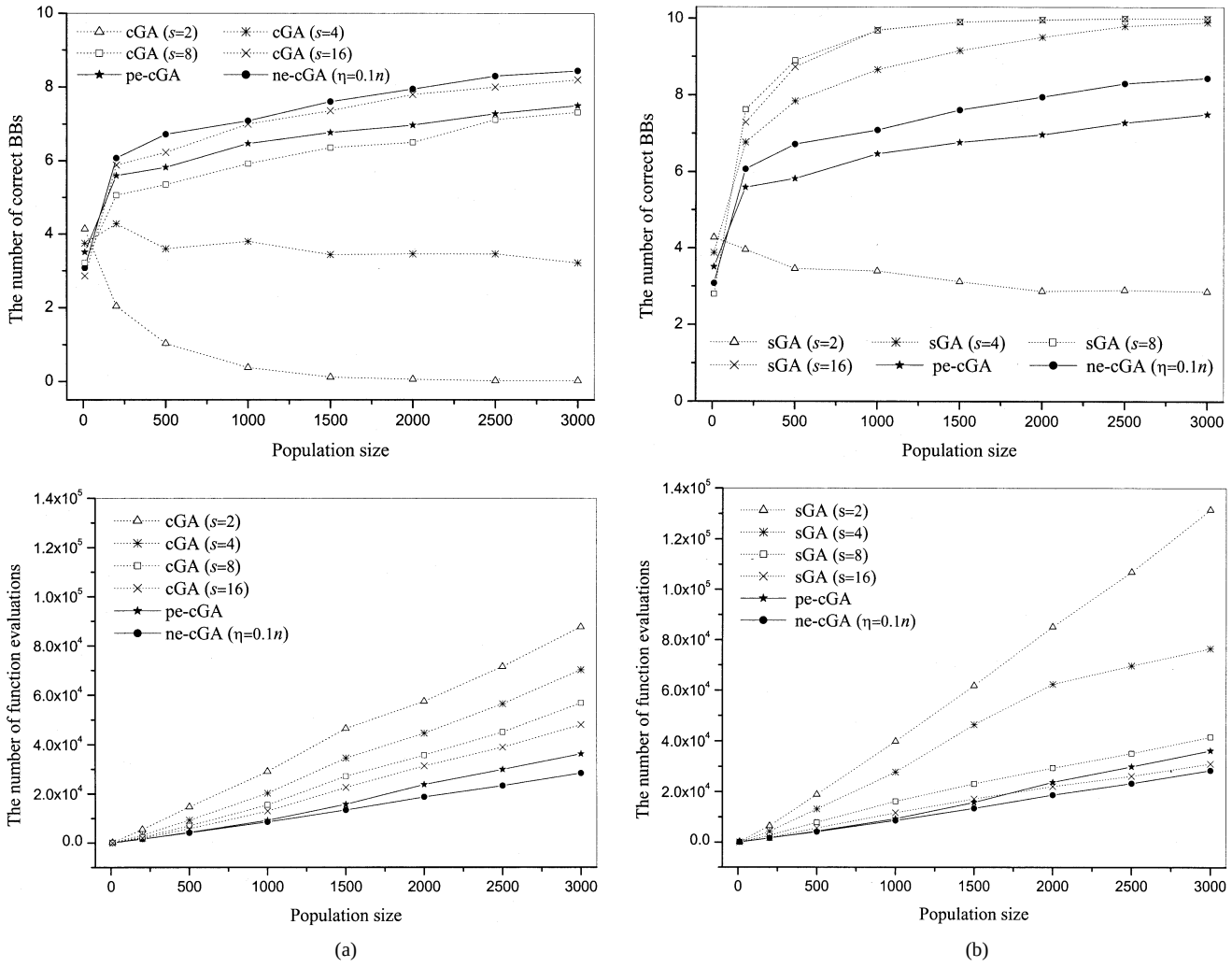


Fig. 6. Performance of the solution quality and the convergence speed for the algorithms as applied to $f_{3-\mathrm{bit}}$. (a) Performance of the cGA, pe-cGA, and ne-cGA on $f_{3-\mathrm{bit}}$. (b) Performance of the sGA, pe-cGA, and ne-cGA on $f_{3-\mathrm{bit}}$.

Furthermore, these benefits accrue without any compromise on memory requirements and computational costs. In the context of this test problem, on the other hand, it appears that the relation (investigated by Harik *et al.* [9]) between selection pressure and tournament size of the sGA and cGA is satisfied. The relation developed from a global schema theorem dictates the tournament size (of the sGA and cGA) that provides a higher selection pressure that is sufficient to grow the correct BBs as the population size increases. The tournament sizes (in order-k BB) should be greater than or equal to 2^{k-1} and 2^k, respectively, in the case of sGA and cGA [9].

The second deceptive problem is formed by concatenating ten copies of the four-bit trap function for a total chromosome length of 40 bits. Each four-bit trap function has a deceptive-to-optimal ratio of 0.7. That is, the problem is specified by

$$f_{4-\mathrm{bit}} = \sum_{i=1}^{10} f_{\mathrm{trap}}(u_{4i}, 0.7, 1, 3, 4). \tag{29}$$

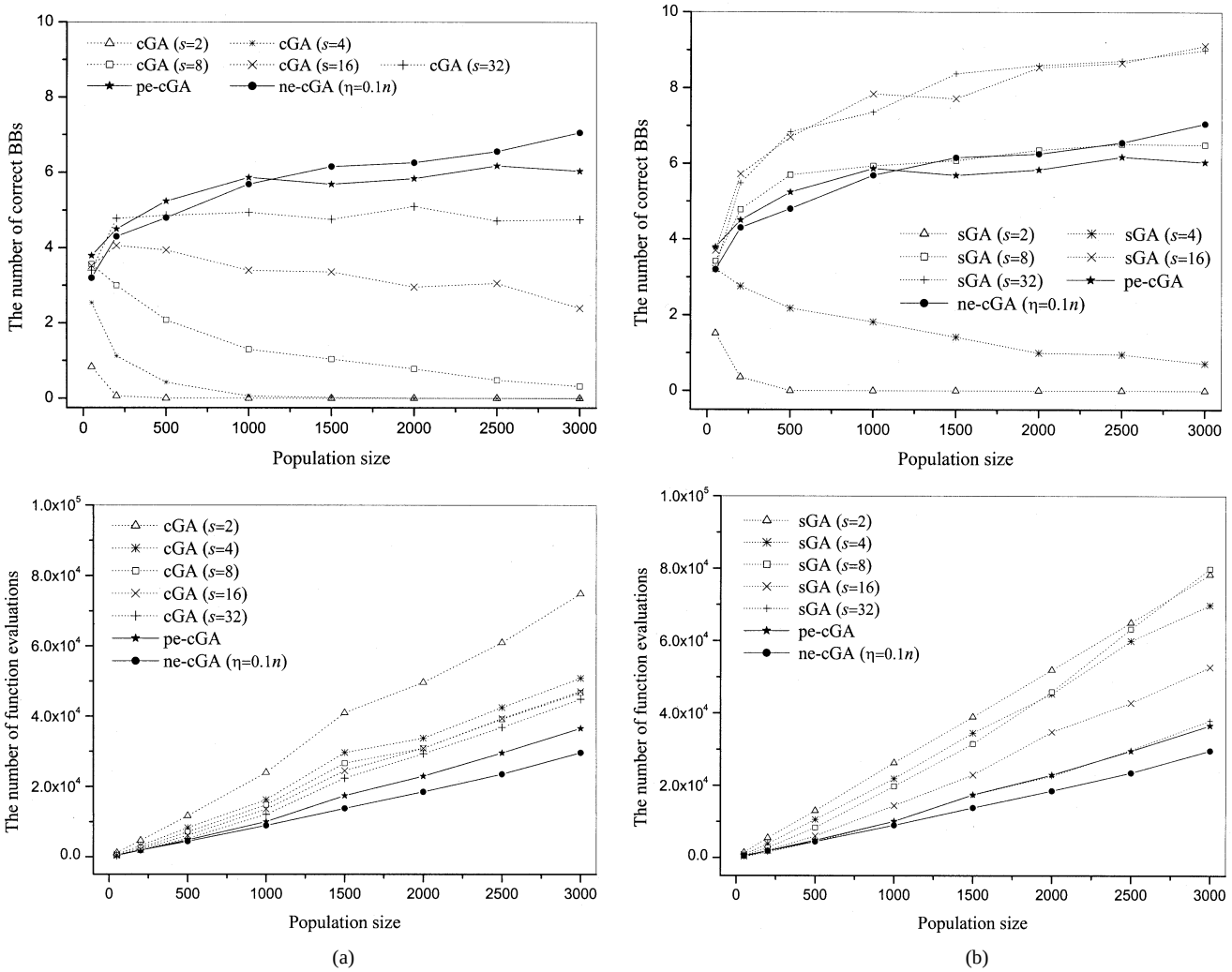
The results are compared in Fig. 7(a)–(b). From Fig. 7(a), the proposed algorithms generally outperform all the cGAs that choose a tournament size s from 2 to 32. Moreover, the

TABLE III
STATISTICAL COMPARISON OF ALGORITHMS ($s = 16$ AND $n = 3000$) WORKING ON $f_{3-\mathrm{bit}}$, WHERE "STD" STANDS FOR THE STANDARD DEVIATION

GAs	sGA		cGA		pe-cGA		ne-cGA	
Statistic	Mean	STD	Mean	STD	Mean	STD	Mean	STD
	31050.0	2872.3	48716.30	4518.37	35896.73	3765.04	28415.41	2967.25

Cases	cGA – pe-GA	pe-cGA – sGA	sGA – ne-cGA
t-test	21.797†	10.234†	6.381†

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.01$ by a two-tailed test.



(a) (b)

Fig. 7. Performance of the solution quality and the convergence speed for the algorithms as applied to $f_{4-\mathrm{bit}}$. (a) Performance of the cGA, pe-cGA, and ne-cGA on $f_{4-\mathrm{bit}}$. (b) Performance of the sGA, pe-cGA, and ne-cGA on $f_{4-\mathrm{bit}}$.

convergence speeds of pe-cGA and ne-cGA are higher than those of all the cGAs. In Fig. 7(b), it is seen that the proposed algorithms achieve a quality of solution that is similar to that obtained by sGA with $s = 8$ and their convergence speeds are higher than those of all the sGAs except when $s = 32$ for pe-cGA. The results are similar to those in Fig. 6. Fast convergence is supported by the statistical test of Table IV. Moreover, Table VII also attests to the accuracy of the speedup model for the four-bit deceptive problem.

It is also clear that the relation (proposed by Harik *et al.* [9]) between selection pressure and tournament size is not satisfied for the cGA in this test problem. According to the relation, the correct BB will grow when the tournament size is greater than or equal to 16 (i.e., 2^k where $k = 4$). However, the cGA is not able to propagate the BB as long as the tournament size is less than 32 (i.e., $s < 32$) in this problem. It appears that there is some discrepancy in the relation. The tournament size may be intricately related to not only the deceptive-to-optimal ratio, but also to the order of BBs and the number of collateral noise sources, etc., in the cGA. The rule [9] takes only the order of BBs into account. However, further investigation on these issues is beyond the scope of this paper. Furthermore, such information

TABLE IV
STATISTICAL COMPARISON OF ALGORITHMS ($s = 32$ AND $n = 3000$) WORKING ON $f_{4-\text{bit}}$, WHERE "STD" STANDS FOR THE STANDARD DEVIATION

GAs	sGA		cGA		pe-cGA		ne-cGA	
Statistic	Mean	STD	Mean	STD	Mean	STD	Mean	STD
	37770.0	5925.20	45014.17	4711.80	36433.04	4027.80	28737.64	2435.51

Cases	cGA – sGA	sGA – pe-sGA	pe-GA – ne-cGA
t-test	9.570†	1.866	16.351†

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.01$ by a two-tailed test.

may not be available in any case. It follows that the existing cGA may not be very useful in practice even if a perfect relation is discovered.

Figs. 6 and 7 show that the proposed algorithms seem to adaptively adjust their selection pressures according to the difficulty of the problems. It is seen that the proposed algorithms are always able to provide a selection pressure that is enough to steadily grow the correct BBs as the population size increases. Therefore, they can effectively solve the difficult problems (especially, deceptive problems involving higher order BBs) without any knowledge about the problem dependent information such as the degree of deception, the (average) order of BBs, and the selection pressure needed to combat disruptive effects of crossover.

C. Experimental Results for Continuous and Multimodal Problems/Functions

Most real-world problems do not involve the concatenation of distinct order-k BBs in a simple manner since their solution/search spaces are continuous and multimodal. The problems can be modeled as an intricate combination of lower and higher-order BBs. In order to investigate the performance on such problems, a circle function [28] and Schaffer's binary function [29] are employed. The functions may be used for modeling several real-world problems, especially those arising in the emerging areas of wireless networks (such as the ultra-wide band antenna design and fading channel estimation problems).

The circle function is investigated first. It is defined in (30) and plotted in Fig. 8

$$\text{minimize } f_C(x) = \left(\sum_{i=1}^{n} x_i^2\right)^{1/4} \cdot \left[\sin^2\left(50 \cdot \left(\sum_{i=1}^{n} x_i^2\right)^{1/10}\right) + 1.0\right] \quad x_i \in [-32.767 \quad 32.768], \quad n = 2 \quad (30)$$

This multimodal function has many local optima (i.e., minima) that are located on concentric circles near the global optimum (i.e., the origin) [28].

Fig. 9(a) and (b) compares the objective function values and the function evaluations achieved by the algorithms as applied to the circle function. It is seen that the proposed algorithms significantly outperform sGA and cGA with regard to convergence speed (see Table V), without unduly compromising on the solution quality. Under fair comparison as depicted in Fig. 9(c), the overall performance of ne-cGA is better than that of the rest of the algorithms as the number of function evaluations increases while pe-cGA achieves a performance that is similar to that of sGA. As noted in Section III-B, the model of speedup does not accurately estimate the extent of convergence improvement (see Table VII) due to interdependency and unequal salience of BBs in this problem. Instead, it provides a marginal speedup for this kind of problem.

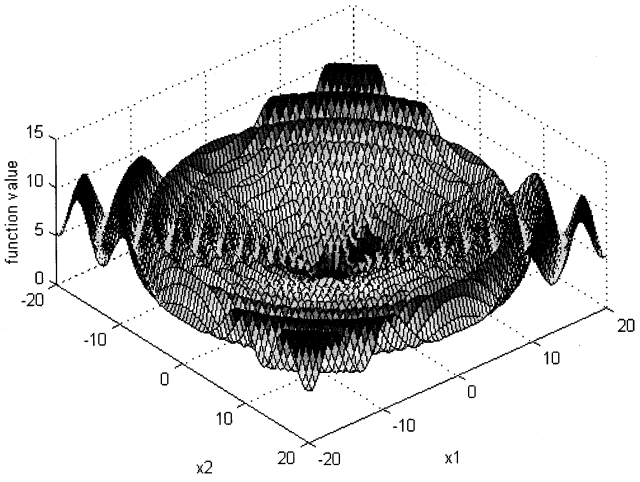


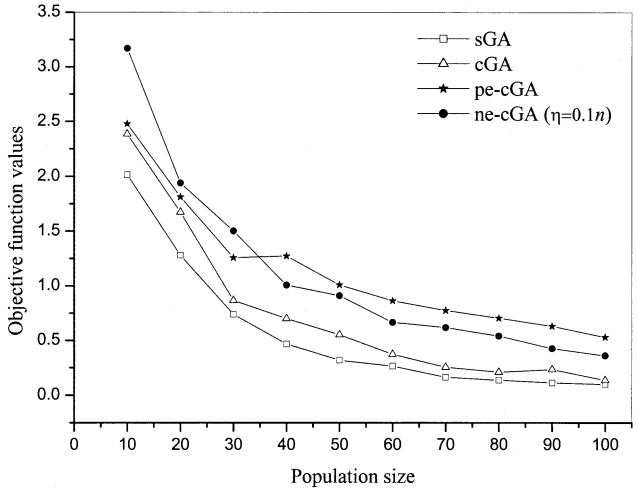
Fig. 8. Plot of f_C (the global optimum $f_C^* = 0.0$).

Schaffer's binary function is considered next. The function is presented in (31)

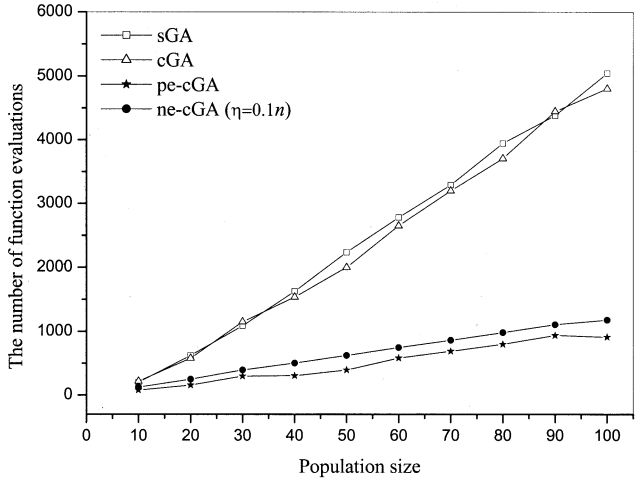
$$\text{maximize } f_{S_6}(x) = \frac{\sin^2\left(\sqrt{\sum_{i=1}^{n} x_i^2}\right)}{1.0 + 10^{-3} \cdot \left(\sum_{i=1}^{n} x_i^2\right)^2} \quad x_i \in [-16.383 \quad 16.384], \quad n = 5. \quad (31)$$

The characteristics (e.g., landscape) of this function are easily grasped from its two-dimensional (2-D) case shown in Fig. 10. The function is degenerate in the sense that many points share the same global optimal function value ($f_{S_6}^* = 0.994\,006\,93$). As can be seen in Fig. 10, the points are located on the highest circle in the crown near the origin [29].

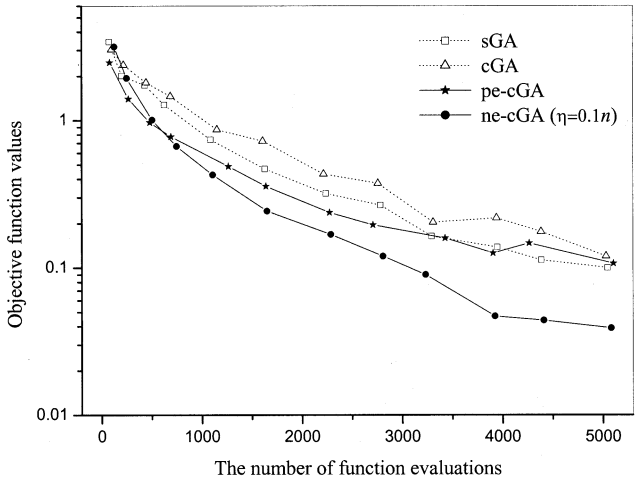
Fig. 11 compares the algorithms as applied to Schaffer's binary function. Fig. 11(a) shows that the solution found by ne-cGA is better than those computed by sGA and cGA, while pe-cGA finds a solution that is similar to that returned by the cGA. Although the result is invalid when the population size is small, such populations are not regarded as feasible candidates in practice. Fig. 11(b) and Table VI show that their convergence speeds are higher than those of sGA and cGA. Furthermore, the extent of (convergence) improvement is about 8.38 times (see Table VII). It also supports the model of speedup as a



(a)



(b)



(c)

Fig. 9. Performance of the algorithms on f_C. (a) Objective function values versus population size. (b) Number of function evaluations versus population size. (c) Objective function values versus number of function evaluations.

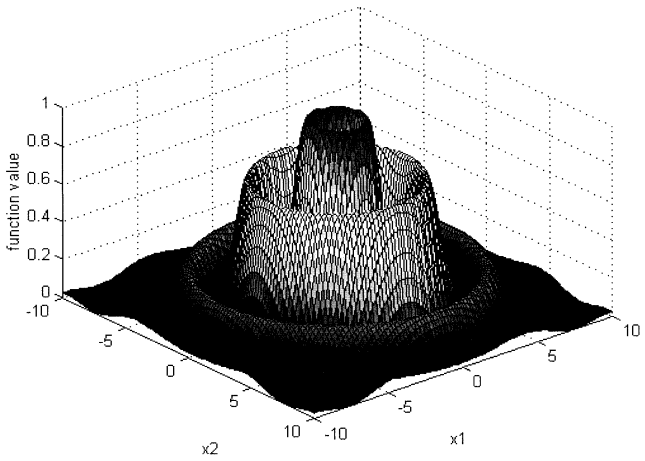


Fig. 10. Plot of the 2-D f_{S_6}.

lower bound; accuracy is not high, however. Note that a larger discrepancy of speedup implies a higher intercorrelation and unequal salience of BBs.

On the basis of comparison studies (Figs. 9 and 11), the proposed algorithms are found to be suitable for solving those types of problems. On the other hand, Figs. 9 and 11 imply that sGA and cGA with $s = 2$ already have a selection pressure that is high enough to overcome crossover disruption because the degree of optimality improves as the population size increases.

D. Comparison Results With ESs

ESs are among the main cutting-edge evolutionary algorithms. However, comparison of ESs with EDAs is quite unusual because they are considered to be different schemes. In other words, any criterion for comparing them in an unbiased way has not yet been suggested. This paper has investigated a relation between pe-cGA and $(1+1)$-ES in Section III-A. More intense comparative studies of the two schemes is imperative. Although ESs maintain at each step a set of solution candidates (i.e., a population), $(1 + 1)$-ES does not involve any concept of population. Without loss of generality, this paper employs the average quality of solutions returned after (almost) the same execution time (i.e., the number of function evaluations) as the comparison criterion. All the test problems have been investigated. For discrete test problems, $(1 + 1)$-ES with Bernoulli distribution as a mutation distribution is taken as a reference. This is the same as $(1 + 1)$-EA proposed by He and Yao [8]. For an objective function g to be maximized, it can be defined as follows, in (32) at the bottom of the page, where Z_i is the Bernoulli random variable with a flipping probability of 0.1.

The solution quality for each algorithm is presented in Table VIII. The $(1 + 1)$-ES is terminated at a specified epoch that is close to the convergence time of all the algorithms. From Figs. 6 and 7, tournament sizes of sGA and cGA are seen to be $s = 4$ and $s = 8$ for the three-bit deceptive problem, respectively; and they are $s = 8$ and $s = 32$ for the

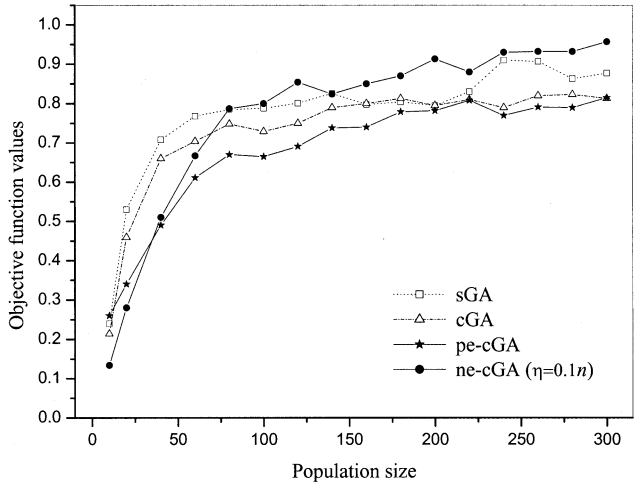
$$\mathbf{X}_{k+1} = \begin{cases} \mathbf{X}_k + (Z_1, Z_2, \ldots, Z_m), & \text{if } g(\mathbf{X}_k + (Z_1, Z_2, \ldots, Z_m)) > g(\mathbf{X}_k) \\ \mathbf{X}_k, & \text{otherwise} \end{cases} \tag{32}$$

TABLE V
STATISTICAL COMPARISON OF ALGORITHMS ($n = 100$) WORKING ON f_C, WHERE "STD" STANDS FOR THE STANDARD DEVIATION

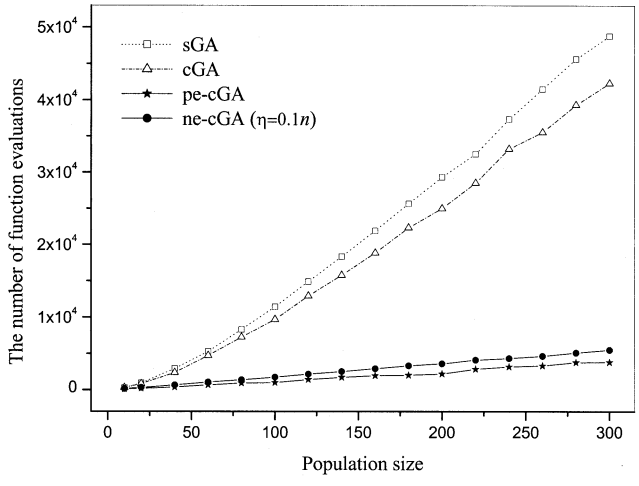
GAs	sGA		cGA		pe-cGA		ne-cGA	
Statistic	Mean	STD	Mean	STD	Mean	STD	Mean	STD
	5047.0	833.21	4806.97	600.65	910.95	119.22	1174.30	153.89

Cases	sGA – cGA	cGA – ne-sGA	ne-GA – pe-cGA
t-test	2.338	58.586†	13.527†

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.01$ by a two-tailed test.



(a)



(b)

Fig. 11. Performance of the algorithms on f_{S_6}. (a) Objective function values versus population size. (b) Number of function evaluations versus population size.

four-bit deceptive problem. This is because the values provide a selection pressure that is enough to offset the disruptive effect of crossover. From the Table VIII, it is seen that the pe-cGA generally returns a solution that is better than or similar to that of $(1 + 1)$-ES except for f_{MDP}. Moreover, the ne-cGA outperforms $(1 + 1)$-ES on most of the test problems.

For continuous test problems, $(1 + 1)$-ES with self-adaptive mutation is employed as a reference. The $(1 + 1)$-ES is defined by (1). Gaussian distribution with zero mean and unit variance (i.e., $N(0, 1)$) is employed for mutation. The initial step length control parameter l_0 is assigned the value 100 and the period τ that is an instance for adjusting l_k is set to one. The step-length rule suggested in [19] (for avoiding premature convergence) is employed. Evolution of $(1 + 1)$-ES is terminated when l_k is smaller than 0.1.

Performance of all the algorithms is exhibited in Table IX. In this table, it is seen that the pe-cGA is similar in performance to $(1 + 1)$-ES while the performance of ne-cGA is better. By observing the standard deviations, one can also conclude that $(1 + 1)$-ES is quite unstable with regard to solution quality, as well as convergence speed. Note that the performance of $(1+1)$-ES would be the same as that of pe-cGA if the step length control parameter l_k and the mutation distribution $f_{\mathbf{z}_k}(Z)$ are adjusted to satisfy (5b) at every instance (*Proposition 1*). However, this is not possible in practice.

As a consequence (of comparison studies), the proposed algorithms (especially, ne-cGA) are also seen to be quite promising candidates for solving various problems (that can be approximated/modeled by the tested types of problems) vis-a-vis the corresponding ESs.

E. Effects of the Length of Inheritance

The ne-cGA is characterized by a parameter η that controls the length of elite chromosome's inheritance. That is, it controls the strength of elitism. Hence, the parameter may affect the algorithm's performance. This experiment focuses only on examining the influence of this parameter on the performance. The algorithm is investigated on three test problems/functions: the one-max problem, the deceptive problem based on three-bit deceptive function, and Schaffer's binary function.

Fig. 12 compares the number of correct BBs (i.e., bits) and the number of function evaluations returned by the pe-cGA and the ne-cGA with different values of η as applied to the one-max problem (i.e., a simple problem).

It is observed that the quality of solution improves as η decreases (i.e., elitism becomes weaker) except when the population size is small. This is due to increased genetic diversity brought about by weakening elitism. Moreover, this improvement does not have to pay any price in terms of reduced convergence speed (i.e., the number of function evaluations).

The solution quality (of ne-cGA) with $\eta = 0.05n$ is slightly better than that of ne-cGA with $\eta = 0.1n$ when the population size is larger than 60. However, their performances are nearly the same under a fair comparison. Thus, the performance will not be improved any more when a value that is smaller than $0.05n$ is assigned to the parameter η. It implies that too low a value of elitism incurs degradation in algorithm's performance.

TABLE VI
STATISTICAL COMPARISON OF ALGORITHMS ($n = 300$) WORKING ON f_{S_6}, WHERE "STD" STANDS FOR THE STANDARD DEVIATION

GAs	sGA		cGA		pe-cGA		ne-cGA	
Statistic	Mean	STD	Mean	STD	Mean	STD	Mean	STD
	48798.0	5406.12	42680.33	3508.60	3721.83	485.74	5451.0	557.04

Cases	sGA – cGA	cGA – ne-sGA	ne-GA – pe-cGA
t-test	9.493†	104.794†	23.396†

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.01$ by a two-tailed test.

TABLE VII
COMPARISON OF SPEEDUP ON THE TEST PROBLEMS

	Theory	f_{OneMax}	f_{MDP}	f_{3-bit}	f_{4-bit}	f_C	f_{S_6}
Speedup	2.61	2.55	2.63	2.73	2.47	4.21	8.38

TABLE VIII
PERFORMANCE COMPARISON OF ALGORITHMS FOR DISCRETE TEST PROBLEMS, WHERE "QOS" AND "CONV" STAND FOR QUALITY OF SOLUTION AND CONVERGENCE, RESPECTIVELY

Problems	Measures	Statistic	sGA	cGA	pe-cGA	ne-cGA	(1+1)-ES
One-Max: $f^*_{OneMax} = 100$	QoS	Mean	99.94	99.97	92.97	99.61	88.31
		STD	0.24	0.17	2.40	0.57	1.67
	CONV	Mean	1996.47	1988.93	1994.25	1988.54	2000
		STD	73.20	50.98	228.97	64.58	-
MDP: $f^*_{MDP} = 10$	QoS	Mean	9.76	9.71	8.56	9.73	9.80
		STD	0.50	0.48	1.10	0.51	0.50
	CONV	Mean	1951.95	1985.11	1919.41	1985.90	2000
		STD	456.61	247.30	262.34	166.06	-
3-bit deceptive: $f^*_{3-bit} = 10$	QoS	Mean	7.51	5.58	6.74	7.26	6.93
		STD	0.98	1.39	1.45	1.24	1.39
	CONV	Mean	9930.40	9927.71	9659.63	9848.04	10000
		STD	2520.62	543.70	878.89	1284.92	-
4-bit deceptive: $f^*_{4-bit} = 10$	QoS	Mean	5.97	5.02	6.08	7.05	5.04
		STD	1.17	1.42	1.32	1.05	1.32
	CONV	Mean	29876.4	29187.3	29583.1	29950.7	30000
		STD	7556.91	2822.7	2838.26	2042.85	-

Problems	f_{OneMax}		f_{MDP}	
Cases	pe-cGA – ES	ne-cGA – ES	pe-cGA – ES	ne-cGA – ES
t-test	15.919†	63.995†	-13.344†	-1.185
Problems	f_{3-bit}		f_{4-bit}	
Cases	pe-cGA – ES	ne-cGA – ES	pe-cGA – ES	ne-cGA – ES
t-test	-0.946	1.770	5.375†	10.398†

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.05$ by a two-tailed test.

This is the reason why the ne-cGA with $\eta = 0.05n$ cannot achieve a better performance with respect to optimality, as well as convergence speed when the population size is smaller than 60. We can see that elitism is quite weak when population is in the above range since the length of inheritance is at most three generations. Thus, it is clear that a proper adjustment of elitism in one-max type problems leads to better quality (of solution) without unduly compromising on convergence performance.

Fig. 13 compares the algorithms when applied to the deceptive problem constructed by a three-bit deceptive function with the deceptive-to-optimal ratio of 0.7. The quality of solution improves as η decreases as long as elitism is not too weak. Unlike the convergence tendency shown in Fig. 12, however, the convergence performance also improves but slightly as long as a smaller value is assigned to η before it reaches $0.1n$.

Therefore, it is noted that an apt adjustment of elitism in deceptive-type problems (i.e., difficult problems) leads to a better solution without any compromising on the rate of convergence.

Fig. 14 presents the effects of the parameter η on ne-cGA as applied to Schaffer's binary function. In order to highlight the effects of restricted elitism, a 2-D function is considered. As can be seen in the figures, the convergence performance is getting

TABLE IX
PERFORMANCE COMPARISON OF ALGORITHMS FOR CONTINUOUS TEST PROBLEMS, WHERE "QOS" AND "CONV" STAND FOR QUALITY OF SOLUTION AND CONVERGENCE, RESPECTIVELY

Problems	Measures	Statistic	sGA	cGA	pe-cGA	ne-cGA	(1+1)-ES
Circle (2-D): $f_C^* = 0.0$	QoS	Mean	0.2347	0.3051	0.1917	0.097	0.2788
		STD	0.2621	0.4104	0.1843	0.096	0.8766
	CONV	Mean	3016.14	3083.92	3020.27	3011.15	3094.22
		STD	501.80	384.28	508.33	334.68	1496.71
Schaffer (5-D): $f_{S_6}^* = 0.99400693$	QoS	Mean	0.7180	0.6818	0.779	0.828	0.7711
		STD	0.2009	0.2059	0.1449	0.1703	0.2151
	CONV	Mean	2303.4	2341.5	2271.28	2340.20	2352.43
		STD	340.47	257.01	187.89	256.12	1127.77

Problems	f_C		f_{S_6}	
Cases	pe-cGA – ES	ne-cGA – ES	pe-cGA – ES	ne-cGA – ES
t-test	-0.553	-1.575	0.305	2.069†

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.05$ by a two-tailed test.

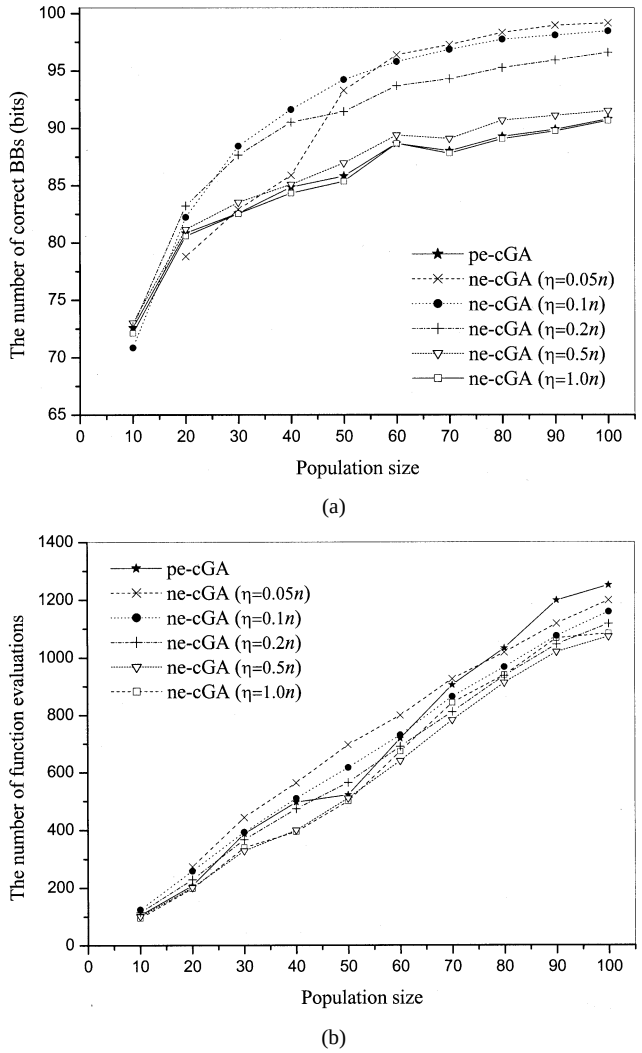


Fig. 12. Comparison of the pe-cGA and ne-cGA with various η on f_{OneMax}. (a) Number of correct BBs (bits) versus population size. (b) Number of function evaluations versus population size.

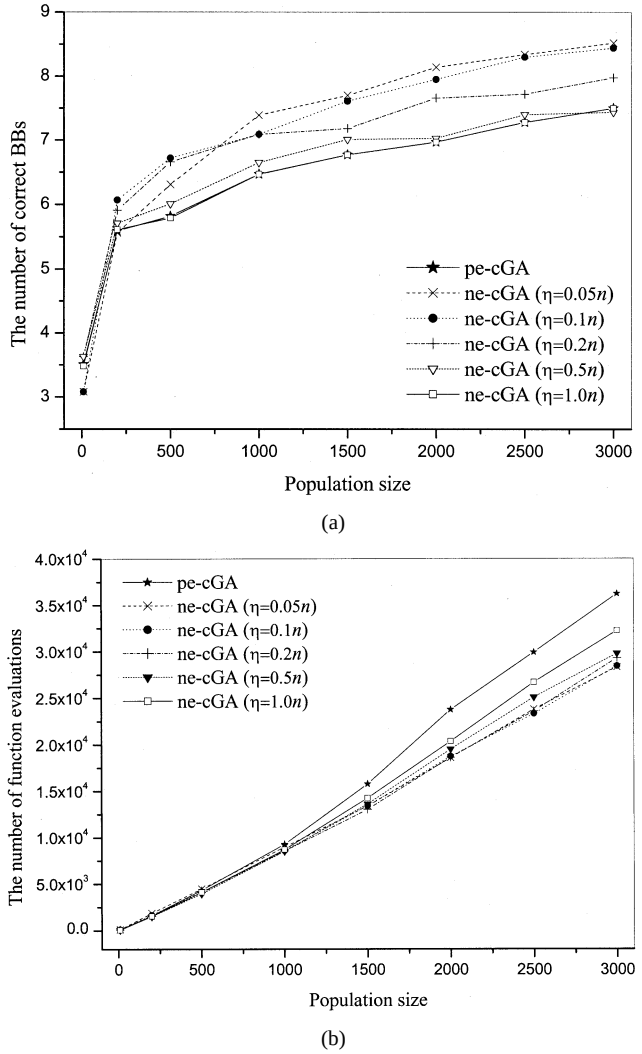


Fig. 13. Comparison of the pe-cGA and ne-cGA with various η on $f_{3-\mathrm{bit}}$. (a) Number of correct BBs versus population size. (b) Number of function evaluations versus population size.

worse without improving the solution quality as a value less than $0.5n$ is assigned to the parameter. It is seen that the ne-cGA with $\eta = 0.5n$ returns the best performance when the solution quality and convergence speed are considered at the same time.

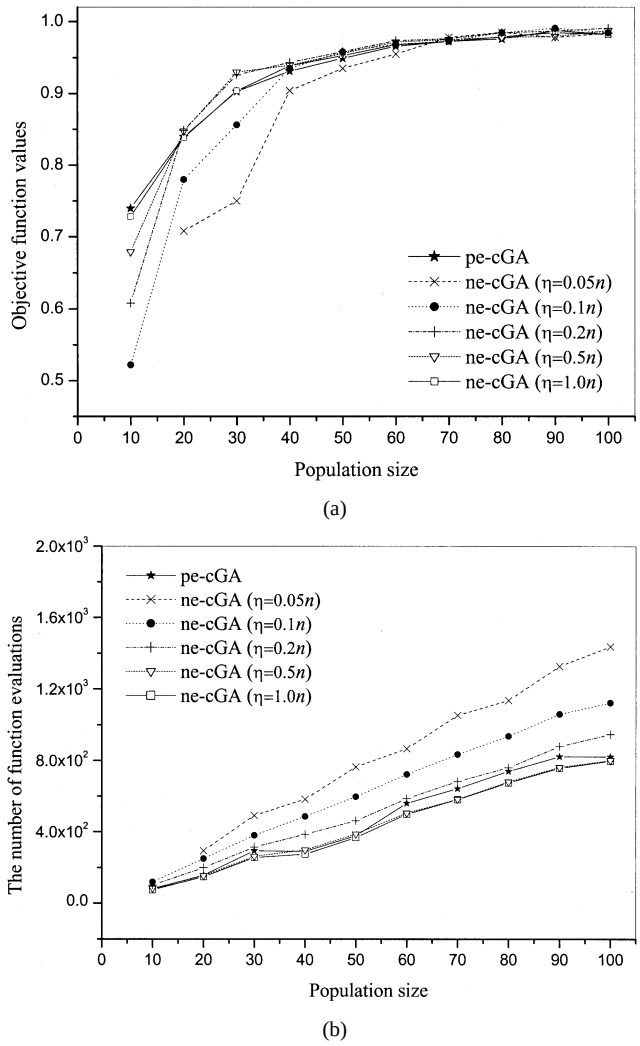


Fig. 14. Comparison of the pe-cGA and ne-cGA with various η on f_{S_6}. (a) Objective function values versus population size. (b) Number of function evaluations versus population size.

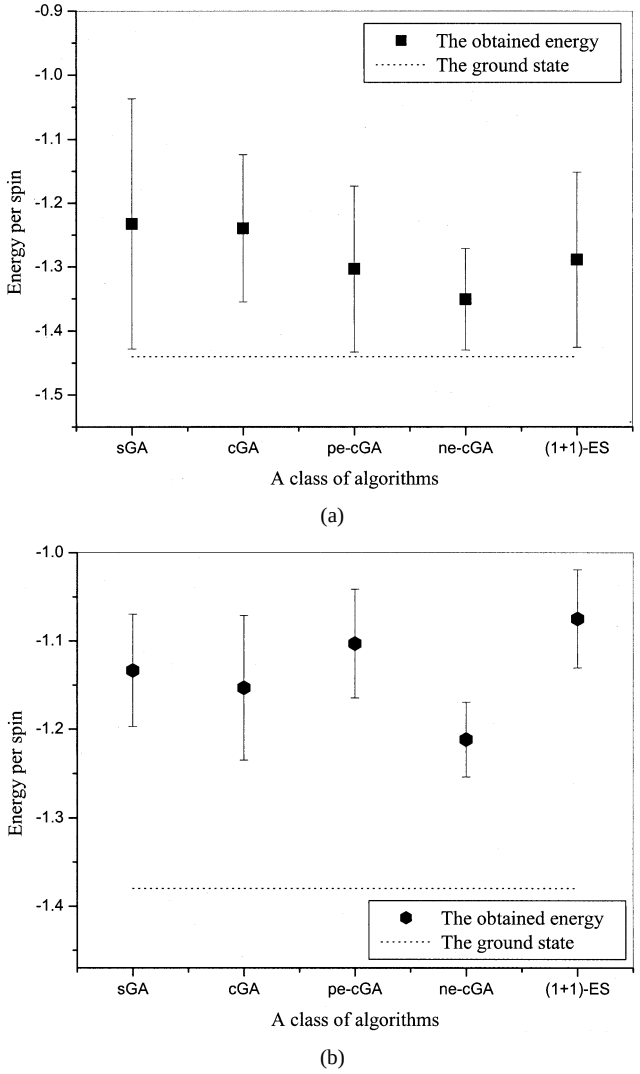


Fig. 15. Performance of Algorithms on ISG systems. (a) Energy per spin for an ISG system with 25 spins versus a class of algorithms. (b) Energy per spin for an ISG system with 100 spins versus a class of algorithms.

It follows that the reduction of η below half the simulated population size (i.e., $0.5n$) results in the degradation of both the solution quality, as well as the convergence speed in this type of problems. However, this degradation is not of any major concern in so far as the parameter η is above $0.1n$.

As shown in Figs. 12–14, all the objective function values of ne-cGA with $\eta = 1.0n$ are comparable with that of the pe-cGA. Incidentally, it verifies an assertion that arises from *Proposition 2*. The assertion requires that the solution quality of ne-cGA with $\eta = n$ be the same as that of the pe-cGA.

As a result, it follows that an adequately controlled elitism (i.e., the length of the elite chromosome's inheritance) imparts the genetic diversity, thereby improving the performance. Note that the adjustment of elitism would be problem-dependent in practice; thus, it is difficult to properly adapt it. Based on the comparative studies, however, $\eta = 0.1n$ can be considered as a promising value.

F. Real-World Applications: Ising Spin-Glasses (ISG) Systems

Ising spin-glasses (ISG) systems are considered in order to examine the feasibility and usefulness of the proposed algorithms in solving real-world problems. Finding the ground state of a given ISG system is a well known problem in statistical physics. In the context of GAs, ISG systems are frequently employed as test cases in the study of GAs because they exhibit symmetry and a large number of plateaus [30], [31]. The physical state of an ISG system is defined by a Hamiltonian H that specifies the energy of the system by

$$H(\sigma) = -\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} J_{ij}\sigma_i\sigma_j \tag{33}$$

where a set of spins $\sigma = (\sigma_0, \sigma_1, \ldots, \sigma_{n-1})$ taking a value from $\{-1, +1\}$ represents the physical state of spins, and J_{ij} specifies a coupling coefficient from ith spin to jth spin. The objective is to find the state of spins so that the energy is minimized (i.e., the ground state is achieved).

In our investigations, spins were arranged on a 2-D grid. The spins interact with only their nearest neighbors. We assume periodic boundary conditions. There exist several polynomial time algorithms for solving this special case of ISG problems. The optimal solution of such a system is provided by an online

TABLE X
STATISTICAL COMPARISON AMONG ALGORITHMS ON ISG SYSTEMS, WHERE "E/S" STANDS FOR ENERGY PER SPIN

Problems	Measure	Statistic	sGA	cGA	pe-cGA	ne-cGA	(1+1)-ES
25-spins	E/S	Mean	-1.2325	-1.2394	-1.3031	-1.3504	-1.2883
		STD	0.1956	0.1155	0.1298	0.0794	0.1371
100-spins	E/S	Mean	-1.1334	-1.1530	-1.1029	-1.2116	-1.0748
		STD	0.0637	0.0819	0.0616	0.0423	0.0556

	t-test			
25-spins	sGA – cGA	cGA – ES	ES – pe-cGA	pe-cGA – ne-cGA
	0.270	2.730†	0.783	3.110†
100-spins	ES – pe-cGA	pe-cGA – sGA	sGA – cGA	cGA – ne-cGA
	3.392†	3.263†	1.890	5.923†

† The value of t with 99 degrees of freedom is significant at $\alpha = 0.05$ by a two-tailed test.

server [32]. Moreover, J_{ij} is chosen from $\{-1, +1\}$ in a random fashion (uniform distribution). All the algorithms were tested on two ISG systems with 25 and 100 spins (arranged on 5×5 and 10×10 toroids, respectively).

Fig. 15 and Table X exhibit the energy (per spin) found by each algorithm, measured after almost equal number of function evaluations. That is, the results of ISG systems with 25 and 100 spins are collected after approximately 10^4 and $3 * 10^4$ evaluations, respectively. It is observed that the ne-cGA finds the smallest energy for both systems although the pe-cGA can not find a solution that is better than that of sGA and cGA for the 100-spins system.

Based on the comparative studies presented in this section, we may conclude that the proposed algorithms, especially ne-cGA, are quite promising candidates for solving various types of problems from simple to difficult problems including real-world applications.

V. CONCLUSION

This paper has presented two elitism-based cGAs in an EDA framework. The aim is to address the problems associated with inadequate memory in the cGA by employing elitism in a proactive manner. The cGA is likely to lose—irretrievably—the best current solution due to lack of memory. Thereby, it cannot speedily solve many difficult optimization problems in an efficient manner. The pe-cGA and the ne-cGA have been proposed in this regard. The pe-cGA compensates for the lack of memory (of the cGA) by keeping the current best solution until, hopefully a better solution is found. It was shown that the pe-cGA is identical to $(1 + 1)$-ES with self-adaptive mutation. The ne-cGA relaxes selection pressure (i.e., elitism) of the pe-cGA by restricting the length of elite chromosome's inheritance, thereby mitigating the possibility of premature convergence (i.e., convergence to a local optimum). The allowable length of inheritance was shown to be bounded by the simulated population size (i.e., $\eta < n$).

Furthermore, an analytic model of speedup for quantifying convergence improvement has been proposed.

Simulation studies showed that the proposed algorithms generally provide a better solution and a higher convergence speed than those of the sGA and cGA. The overall performance was shown to be better than that of sGA, cGA, and $(1+1)$-ES, especially as the problem becomes more and more difficult. It was also noted that most real-world problems cannot be modeled as simple problems. The experiments have also shown that the pe-cGA and ne-cGA do not have to adjust the tournament size in an attempt to exert a selection pressure high enough to compensate for crossover disruption. The algorithms always offer a proper selection pressure as if the tournament size (i.e., selection pressure) is automatically regulated in accordance with the degree of difficulty of the problems. Moreover, the experiments have shown that the speedup model accurately estimates the convergence improvement if the problems involve equally salient and uncorrelated BBs; it provides a marginal speedup, otherwise.

On the other hand, it was demonstrated that the quality of the solution improves as the length of inheritance is decreased, while the convergence performance depends on the problem, in general. However, a proper adjustment of elitism improves the solution quality and enhances the convergence speed.

Furthermore, a feasibility of the proposed algorithms for real-world applications has been demonstrated by investigating their performance on ISG systems.

The proposed algorithms can search the solution space effectively and speedily without compromising on memory and computational requirements. Moreover, the search capability may not depend overly on the shape of the search/solution space (i.e., difficulty of problems).

ACKNOWLEDGMENT

The authors would like to thank X. Yao and the reviewers for their constructive comments and suggestions, and D. E. Goldberg for his insightful comments on estimation of distribution algorithms.

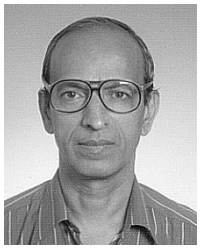
REFERENCES

[1] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, MA: Addison-Wesley, 1989.
[2] T. Bäck, D. B. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*. New York: Inst. Physics Publishing and Oxford Univ. Press, 1997.
[3] D. E. Goldberg and M. Rundnick, "Genetic algorithms and the variance of fitness," *Complex Syst.*, vol. 5, no. 3, pp. 265–278, 1991.
[4] D. E. Goldberg, K. Deb, and J. H. Clark, "Genetic algorithms, noise, and the sizing of populations," *Complex Syst.*, vol. 6, no. 4, pp. 333–362, 1992.

[5] G. Harik, E. Cantŭ-Paz, D. E. Goldberg, and B. L. Miller, "The Gambler's ruin problem, genetic algorithms, and the sizing of populations," *Evol. Comput.*, vol. 7, pp. 231–253, 1999.
[6] K. Sastry and D. E. Goldberg, "Modeling tournament selection with replacement using apparent added noise," in *Intelligent Engineering Systems Through Artificial Neural Networks*. New York: ASME, 2001, vol. 11, pp. 129–134.
[7] C. W. Ahn and R. S. Ramakrishna, "A genetic algorithm for shortest path routing problem and the sizing of populations," *IEEE Trans. Evol. Comput.*, vol. 6, pp. 566–579, Dec. 2002.
[8] J. He and X. Yao, "From an individual to a population: An analysis of the first hitting time of population-based evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 6, pp. 495–511, Oct. 2002.
[9] G. Harik, F. G. Lobo, and D. E. Goldberg, "The compact genetic algorithm," *IEEE Trans. Evol. Comput.*, vol. 3, pp. 287–297, Nov. 1999.
[10] S. Tsutsui, "Probabilistic model-building genetic algorithms in permutation representation domain using edge histogram," in *Parallel Problem Solving from Nature—PPSN VII (Lecture Notes in Computer Science, Vol. 2439)*, J. J. M. Guervós, P. Adamidis, H.-G. Beyer, J.-L. Fernández-Villacañas, and H.-P. Schwefel, Eds. Berlin, Germany: Springer-Verlag, 2002, pp. 224–233.
[11] P. Larrañaga and J. A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. Boston, MA: Kluwer, 2002.
[12] G. Harik, "Linkage learning via probabilistic modeling in the ECGA," Univ. Illinois at Urbana–Champaign, Illinois Genetic Algorithms Lab., Urbana, IL, IlliGAL Rep. 99 010, 1999.
[13] K. Sastry and D. E. Goldberg, "On extended compact genetic algorithm," in *Proc. Late Breaking Papers in Genetic and Evolutionary Computation Conf. (GECCO)*. San Francisco, CA, 2000, pp. 352–359.
[14] R. Baraglia, J. I. Hidalgo, and R. Perego, "A hybrid heuristic for the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 5, pp. 613–622, Dec. 2001.
[15] J. I. Hidalgo, J. Lanchars, A. Ibarra, and R. Hermida, "A hybrid evolutionary algorithm for multi-FPGA systems design," in *Proc. Euromicro Symp. Digital System Design (DSD)*, 2002, pp. 60–67.
[16] D. Dumitrescu, B. Lazzerini, L. C. Jain, and A. Dumitrescu, *Evolutionary Computation*. Boca Raton, FL: CRC Press, 2000.
[17] P. M. Reed, B. S. Minsker, and D. E. Goldberg, "The practitioner's role in competent search and optimization using genetic algorithms," presented at the *World Water and Environmental Resources Congress*, Washington, DC, 2001. ISBN 0-7844-0569-7.
[18] A. Rogers and A. P. Bennett, "Genetic drift in genetic algorithm selection schemes," *IEEE Trans. Evol. Comput.*, vol. 3, pp. 298–303, Nov. 1999.
[19] G. Rudolph, "Self-adaptive mutations may lead to premature convergence," *IEEE Trans. Evol. Comput.*, vol. 5, pp. 410–414, Aug. 2001.
[20] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, pp. 82–102, July 1999.
[21] C. Y. Lee and X. Yao, "Evolutionary algorithms with adaptive lévy mutations," *Proc. IEEE Congress on Evol. Comput. (CEC)*, pp. 568–575, 2001.
[22] H. Mülenbein and D. Schlierkamp-Voosen, "Predictive models for the breeder genetic algorithm: I. Continuous parameter optimization," *Evol. Comput.*, vol. 1, no. 1, pp. 25–49, 1993.
[23] T. Bäck, "Generalized convergence models for tournament- and (μ, λ)-selection," in *Proc. 6th Int. Conf. Genetic Algorithms*. San Francisco, CA, 1995, pp. 2–8.
[24] D. Thierens, "Selection schemes, elitist recombination, and selection intensity," in *Proc. 7th Int. Conf. Genetic Algorithms*. San Francisco, CA, 1997, pp. 152–159.
[25] H. Mülenbein and D. Schlierkamp-Voosen, "The science of breeding and its application to the breeder genetic algorithm (BGA)," *Evol. Comput.*, vol. 1, no. 4, pp. 335–360, 1993.
[26] E. Cantŭ-Paz, *Efficient and Accurate Parallel Genetic Algorithms*. Norwell, MA: Kluwer, 2000.
[27] K. Deb and D. E. Goldberg, "Analyzing deception in trap functions," in *Foundations of Genetic Algorithms*, L. D. Whitley, Ed. San Mateo, CA: Morgan Kaufmann, 1993, pp. 93–108.
[28] S. D. Müller, J. Marchetto, S. Airaghi, and P. Koumoutsakos, "Optimization based on bacterial chemotaxis," *IEEE Trans. Evol. Comput.*, vol. 6, pp. 16–29, Feb. 2002.
[29] J. D. Schaffer, R. A. Caruana, L. J. Eshelman, and R. Das, "A study of control parameters affecting online performance of genetic algorithms for function optimization," in *Proc. 3rd Int. Conf. Genetic Algorithms*, J. D. Schaffer, Ed.. San Francisco, CA, 1989, pp. 51–59.
[30] M. Pelikan, D. E. Goldberg, and E. Cantŭ-Paz, "Linkage Problem, Distribution Estimation, and Bayesian Networks," Univ. Illinois at Urbana–Champaign, Illinois Genetic Algorithms Lab., Urbana, IL, IlliGAL Rep. 98 013, 1998.
[31] C. V. Hoyweghen, "Detecting spin-flip symmetry in optimization problems," in *Theoretical Aspects of Evolutionary Computing (Natural Computing Series)*, L. Kallel, B. Naudts, and A. Rogers, Eds, Berlin, Germany: Springer-Verlag, 2001, pp. 423–437.
[32] [Online]. Available: http://www.informatik.uni-koeln.de/ls-juenger/projects/sgs.html

Chang Wook Ahn (S'02) was born in Hae-Je, Chonnam, Korea, in 1977. He received the B.S. and M.S. degrees in electrical engineering from Korea University, Seoul, Korea, in 1998 and 2000, respectively. He is currently working toward the Ph.D. degree in the Department of Information and Communications, Kwang-Ju Institute of Science and Technology (K-JIST), Gwang-Ju, Korea.

From January 2003 to August 2003, he was a Visiting Scholar in the Illinois Genetic Algorithms Laboratory (IlliGAL), University of Illinois at Urbana–Champaign (UIUC). His current research interests include genetic and evolutionary algorithms, neural networks, and the applications of evolutionary techniques to wireless networks.



R. S. Ramakrishna (M'83–SM'98) received the Ph.D. degree from the Indian Institute of Technology, Kanpur, India, in 1979.

From 1980 to 1996, he worked as an Assistant Professor, Associate Professor, and Professor in the Department of Computer Science and Engineering, Indian Institute of Technology, Bombay, India. He is currently a Professor in the Department of Information and Communications at the Kwang-Ju Institute of Science and Technology (K-JIST), Gwang-Ju, Korea. His current research interests include evolutionary algorithms, parallel and distributed computing, computer graphics, and quantum computing.