Investigation of the Exponential Population Scheme for Genetic Algorithms

Yuen-Jen Lin
Taiwan Evolutionary Intelligence Laboratory
Department of Electrical Engineering
National Taiwan University
r05921045@ntu.edu.tw

Tian-Li Yu
Taiwan Evolutionary Intelligence Laboratory
Department of Electrical Engineering
National Taiwan University
tianliyu@ntu.edu.tw

ABSTRACT

Early development of GAs requires many parameters to be tuned. The tuning process increases the difficulty for inexperienced practitioners. Modern GAs have most of these parameters pre-determined, and therefore recent research concerning parameterless schemes has focused on population size. The techniques developed in this paper are mainly based on Harik and Lobo's work and the exponential population scheme (EPS), which double the population until the solution is satisfactory. In this paper, we modify EPS based on theoretical analyses. Specifically, we propose a new termination criterion and an optimized population multiplier. The experiment results show that our scheme reduces 33.4%, 19.1% and 29.6% number of function evaluations (NFE) on hBOA (the parameter-less hBOA), LT-GOMEA and DSMGA-II respectively when compared to Harik-Lobo scheme, and reduces 28.5%, 4.7% and 11.0% NFE on hBOA, LT-GOMEA and DSMGA-II respectively when compared to EPS. In addition, compared to EPS, our scheme empirically reduces the number of failures when using LT-GOMEA to solve the folded trap and MAX-SAT problems.

CCS CONCEPTS

• **Computing methodologies** → *Artificial intelligence*;

KEYWORDS

Parameter-less Genetic Algorithm, Population Sizing

ACM Reference Format:
Yuen-Jen Lin and Tian-Li Yu. 2018. Investigation of the Exponential Population Scheme for Genetic Algorithms. In *Proceedings of Genetic and Evolutionary Computation Conference (GECCO '18)*, Jennifer B. Sartor, Theo D'Hondt, and Wolfgang De Meuter (Eds.). ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205551

1 INTRODUCTION

Genetic algorithms are well known for their applicability and ability to solve problems efficiently. However, in traditional GA procedures such as crossover and selection, many control parameters have to be determined by users. These parameters might highly impact to the efficiency of GA and the impact might be problem dependent. Several methods like meta-GA [9] have been proposed to conquer the inconvenience of fine-tuning these parameters when using GA in practice.

In recent years, the crossover in traditional GA has been replaced by recombinative operators such as optimal mixing [21], restrict mixing and back mixing [12]. The research on selection also suggests that the selection pressure should be fixed on a small value like 2 [7]. These works help to determine most of the control parameters. However, there is the last and maybe the most important parameter – population size, which decides whether the information in population is enough for GA to solve problems. There have been some research working on population sizing of GA, such as decision making models [5], supply of building blocks [3] and entropy-based model building [26]. In practical usage, however, the actual needed population size cannot be directly derived since the problem structure is unknown. Usually, GA users need prior knowledge or experiences to set an appropriate population size.

There are two branches of research on getting rid of the population size. One is to use existing GAs with an external mechanism. For example, in the parameter-less GA scheme proposed by Harik and Lobo [11, 13, 19], several GAs with different population sizes are initialized and each GA runs independently. This method has also been adopted on some other GAs like the parameter-less hBOA [16]. The other branch is embedding modification, either to adapt population size during GA process [13, 25], or to change other GA procedures [2]. The parameter-less population pyramid (P3) [8] is one of the representative algorithms belonging to this branch.

The exponential population scheme (EPS), which belongs to the former branch, restarts GA with a doubled population size when the previous population converges. EPS was not adopted on the simple GA because of the long convergence time [11]. However, due to fast convergence of modern GAs, EPS becomes promising. Research has shown that EPS outperforms Harik-Lobo scheme on LT-GOMEA [1]. In this paper, we modify EPS to improve efficiency and show that our modified scheme is universally applicable on modern GAs.

The rest of this paper is organized as follows. The background is first introduced in Section 2. Our modification of the scheme is proposed in Section 3. Experiment results are shown in Section 4. The conclusion follows.

2 BACKGROUND

In this section, we introduce two parameterless schemes for GAs – Harik-Lobo scheme and EPS. We also address the GAs on which we test the parameterless schemes, including the hierarchical Bayesian

GECCO '18, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5618-3/18/07…$15.00
https://doi.org/10.1145/3205455.3205551

optimization algorithm (hBOA), the linkage tree gene-pool optimal mixing evolutionary algorithm (LT-GOMEA) and the dependency structure matrix genetic algorithm II (DSMGA-II).

2.1 Harik-Lobo Scheme

In Harik-Lobo scheme, several populations with different sizes are maintained and evolved in a parallel way. It starts with a small population of size n_0, runs a GA for m generations, then runs the GA on a population with a doubled size of $2n_0$ for a generation. This process repeats until the $2n_0$ population has evolved for m generations, then the scheme runs the GA on a population of size $4n_0$ for a generation, and so on. During the process, the populations that have converged are terminated since they cannot generate new solutions. In addition, if the average fitness of a certain population catches up with smaller populations, the smaller populations are terminated. Algorithm 1 shows the pseudocode with $m = 4$ [19].

The conditions that Harik-Lobo scheme stops are predefined by the users, similar to some other parameterless schemes like P3. The scheme might stop when a solution of good enough quality is found, or its time or memory limit is reached. In this paper, the target for the parameterless schemes is set to finding the optimal solution.

Algorithm 1: Harik-Lobo scheme

```
𝒫: populations, GA: the GA used in the scheme, n0: initial
  population size, n: size of the next population, r: population
  multiplier, UpNext: index of next population to evolve,
  BasePop: index of smallest unterminated population,
  Generations: list of generations of each population in 𝒫

n ← n0
Initialize 𝒫[0] with a size of n
BasePop ← 0
UpNext ← 0
while ¬ShouldTerminate do
    if UpNext = Size(𝒫) then
        n ← rn
        Initialize 𝒫[UpNext] with a size of n
    Run GA for a generation with 𝒫[UpNext]
    Generations[UpNext] ← Generations[UpNext] + 1
    if HasConverged(𝒫[UpNext]) then
        BasePop ← UpNext + 1
        UpNext ← UpNext + 1
    else
        if CatchUpWithSmallerPop(𝒫[UpNext]) then
            BasePop ← UpNext
        if Generations[UpNext] mod 4 = 0 then
            UpNext ← UpNext + 1
        else
            UpNext ← BasePop
return the best solution in 𝒫
```

2.2 Exponential Population Scheme

Unlike Harik-Lobo scheme, which maintains several populations at the same time, EPS restarts the GA with a doubled population size when the current population converges. The concept was first discussed by Harik and Lobo [11]. This scheme is straightforward for users: if the current population is not large enough to solve the problem, simply try a larger population. However, EPS was not adopted by Harik and Lobo due to slow convergence of the simple GA. Other termination criterion can be defined by the users, but choosing the moment to terminate a population before convergence might impact efficiency [17].

Nevertheless, using EPS might be practical on modern GAs, which converges faster than the simple GA. Taking LT-GOMEA as an example, the premature convergence of a population is determined when the population has not changed between two generations. Research shows that EPS outperforms Harik-Lobo scheme on LT-GOMEA when using premature convergence as termination criterion [1]. Algorithm 2 shows the pseudocode of EPS.

Algorithm 2: Exponential population scheme

```
𝒫: population, GA: the GA used in the scheme, n0: initial
  population size, n: size of the next population, r:
  population multiplier

n ← n0
Initialize 𝒫 with a size of n
while ¬ShouldTerminate do
    Run GA for a generation with 𝒫
    if PrematureConvergence(𝒫) then
        n ← rn
        Initialize NewPop with a size of n
        𝒫 ← NewPop
return the best solution in 𝒫
```

2.3 Hierarchical BOA

hBOA is a probabilistic model-building GA [14, 15]. In each generation, hBOA selects better chromosomes from the population and learns the probabilistic model from the selected chromosomes. Then the model is used for the recombination of new chromosomes. hBOA uses restricted tournament replacement (RTR) [10] to replace the old chromosomes in the population. For a new chromosome, RTR finds the closest chromosome with it in the population and compares their fitness. Replacement occurs only when the fitness of the new chromosome is greater than the old one. RTR ensures diversity in the population, which is necessary for building a good probabilistic model.

2.4 LT-GOMEA

In LT-GOMEA, the recombinative operator used to generate new chromosomes is optimal mixing (OM) [21], in which the grouped genes to be exchanged between two chromosomes are referred as masks. In each OM operation, a donor and a receiver are chosen from the population, then the donor gives its pattern in the mask to the receiver. The receiver only takes the change if its fitness does not decrease after receiving the pattern.

The set of masks used to perform OM in LT-GOMEA is modeled by clusters in a linkage tree [20, 22]. A linkage tree is computed with a distance measure between two genes based on mutual information. Before the mixing process, half of the population is chosen using binary tournament selection to build a linkage tree [22]. Each generation, every chromosome in the population acts as a receiver, on which OM is performed with all the masks in the linkage tree.

2.5 DSMGA-II

In DSMGA-II, an incremental linkage set (ILS) is used in place of masks in the mixing process [12]. ILS use mutual information as the dependency measure between each pair of gene positions. DSMGA-II uses two modified versions of OM operator – restrict mixing (RM) and back mixing (BM). RM flips the bits in the receiver with the masks in the ILS if the corresponding pattern exists in the population. If the fitness of the receiver does not decrease after flipping, the receiver takes the change and RM stops. Once an RM succeeds, BM is launched. The pattern changed in the receiver is used as a donor and pasted into other chromosomes in the population. The change is taken if the fitness of the other chromosomes that receive the pattern improves.

3 OUR MODIFIED SCHEME

EPS is potentially applicable on modern GAs. One reason is that fast convergence makes it easier to determine whether a population is able to produce better solutions. Additionally, since EPS only maintains one population at a time, it avoids unnecessary function evaluations that come from maintaining larger populations. Because Harik-Lobo scheme increases population size exponentially, evolving larger populations for a generation costs several times more in terms of number of function evaluations (NFE) compared to evolving smaller populations.

There is still room for improvement in EPS. In this section, we propose our new termination criterion and population multiplier to improve the efficiency of the scheme. The reasons behind our modifications are also discussed.

3.1 New Termination Criterion

In previous research [1], the population is terminated upon premature convergence, which is decided by whether the population remains the same between two generations. Once premature convergence happens, we assume that the population cannot produce better solutions since no new solution is produced after a generation of mixing. This termination criterion seems reasonable, but it has some drawbacks. For those problems with large plateaus in the problem landscape, GAs may converge slowly, continuously generate new solutions and reach the premature convergence criterion too late. Therefore, we need a more robust criterion to determine the moment when a population should be terminated.

We derive the new termination criterion by optimal mixing (OM) since it is one of the most powerful recombinative operators in modern GA research. During the OM process, the receiver takes the pattern from the donor only if its fitness does not decrease after the exchange. For the patterns in a certain mask, there are two situations where the contribution of these patterns to fitness does not increase after a generation. One is if the corresponding patterns in all chromosomes in the population are the same, no successful OM was performed. The other is that all successful OMs exchange patterns that contribute equally to fitness. In both situations, the contribution of the patterns in the mask to fitness is not likely to improve further. If the average fitness of the population does not increase after a generation, then the patterns in all masks did not improve. We believe the population cannot produce better solutions and should be terminated if the average fitness does not improve after a number of generations.

Considering cases where the problem is composed of nonoverlapping subproblems, here we make the following assumption: whenever no improvement in fitness is accomplished during a generation for a single mask, there exists a big cluster in the current population of corresponding patterns that contribute equally to fitness, and the rest of the patterns are superior to those in the cluster. Our assumption is based on the following reasoning: if such a cluster does not exist, during OM the fitness contributions from the mask in the donor and the receiver are unequal, hence fitness improvement is highly likely to occur. In addition, if some patterns inferior to those in the cluster exist, when corresponding chromosomes become receivers during OM, their fitness is highly likely to improve.

If the patterns in the mask do not improve after a generation, the patterns in the cluster were not replaced by superior ones and all successful OMs exchange the patterns in the cluster. So the probability for no improvement is

$$p_{not\ improve} = {p_{eq}}^{np_{eq}},$$

where p_{eq} is the proportion of the patterns in the cluster, n is population size and np_{eq} is the number of chromosomes that have these patterns. Thus the probability that the patterns do not improve for t consecutive generations is $({p_{eq}}^{np_{eq}})^t$.

However, since the proportion p_{eq} is unknown, the probability cannot be estimated before we observe that the fitness does not improve. In practice, the event is observed as either having happened or not, so the corresponding maximum likelihood of probability is either 1 or 0. Once it occurs, we believe that the event tends to happen, so the probability is closer to 1 rather than 0. Thus we assume that the probability is greater than 0.5, expressed as the inequality

$$({p_{eq}}^{np_{eq}})^t > 0.5.$$

On the other hand, we say that the patterns in the cluster have taken over the population if $p_{eq} > \frac{n-1}{n}$. To ensure the condition is fulfilled when the event is observed, t should satisfy the formula

$$({p_{eq}}^{np_{eq}})^t > 0.5 > (\frac{n-1}{n})^{n\frac{n-1}{n}t}.$$

From the right hand side of the inequality we get

$$t > \frac{\ln 2}{(n-1)\ln\frac{n}{n-1}}.$$

Usually, the population size n is greater than 2, so the right-hand side of the formula is less than 1. In other words, if average fitness does not improve after a generation, there is a low probability the population will produce better solutions, so we should terminate the current population and switch to the next larger population.

3.2 The Population Multiplier

In both EPS and Harik-Lobo scheme, population size is doubled when a new population is initialized. However, no solid reason

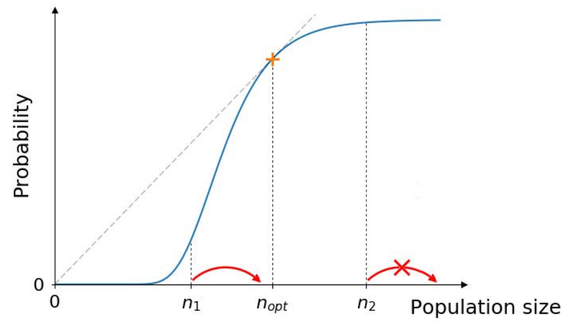

Figure 1: The blue line is the *n*-*p* curve. The population should be enlarged if the GA fails with the population size $< n_{opt}$ and should not be enlarged if the GA fails with the population size $> n_{opt}$. n_{opt} appears at where $\frac{n}{p(n)}$, which is the reciprocal of the slope of the gray dashed line, is minimized.

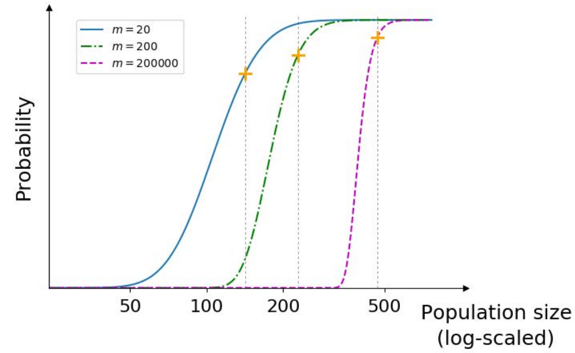

Figure 2: The probability $(1-(1-2^{-k})^n)^m$ with different numbers of building blocks m. Size of building blocks k is set to 5. The optimal population sizes n_{opt} of different m are marked by orange crosses.

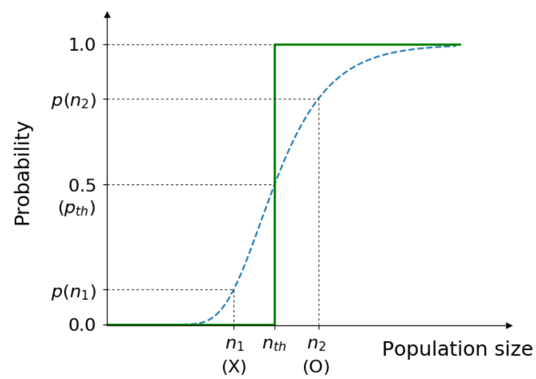

Figure 3: Modeling the *n*-*p* curve by step function. Suppose that the GA fails with population size n_1 and succeeds with population size n_2. The blue dashed line is the original *n*-*p* curve, and the green line is the modeled step function.

is given in the literature for doubling the population size instead of taking another population multiplier like 3 or 5. In the following paragraphs, we discuss the reasonable value of the population multiplier.

3.2.1 Whether to enlarge population. When GAs successfully find the optimal solution, we do not need to enlarge the population. However, the question remains whether to enlarge the population when GAs fail. Assuming we keep trying the GA with the same population size, we define the optimal population size n_{opt} such that the expectation value of the NFE consumed is at a minimum. Here we consider the optimal mixing evolutionary algorithms (OMEAs) [21], whose complexity in terms of NFE is $\Theta(n)$, where n is the population size [23]. If we keep trying the GA with the same population size as $n, n, \ldots$, the expectation value of the NFE consumed is proportional to the expectation value of the total size of used populations, which is $\frac{n}{p(n)}$, where $p(n)$ is the probability of successful convergence (*n*-*p* curve). By definition, $\frac{n}{p(n)}$ is of minimum value at $n = n_{opt}$, as illustrated in Figure 1. When the GA fails, we should enlarge the population if the population size is less than n_{opt}, since the expectation value of the NFE decreases. Otherwise, we should not enlarge the population.

From another perspective, if the problem is formed by m building blocks of size k, the probability that all optimal subsolutions of each building block are in the population is $(1-(1-2^{-k})^n)^m$ [23]. Since the population size increase in EPS grows exponentially, it is reasonable to illustrate the population size using a logarithmic scale. When the population size is log-scaled, the probability is close to a step function when m approaches infinity, as shown in Figure 2. Thus, we model the probability as a step function with the threshold population size n_{th}. In the step function model, the optimal population size n_{opt} mentioned above can be approximated by n_{th}. If the GA fails, we believe that the population size is less than n_{th}, so the population should be enlarged since the maximum likelihood of the probability is 0.

It is still necessary to explain how the threshold population size n_{th} is chosen. When enlarging the population, consider the situation where the GA fails with population size n_1 and succeeds with population size n_2, where $n_1 < n_2$. The probability of such a situation occurring, denoted as p_e, is $(1-p(n_1))p(n_2)$. Assuming a probability p_{th} such that $p(n_1) < p_{th} < p(n_2)$, therefore $p_e \geq (1-p_{th})p_{th}$. Since the above situation has occurred, its maximum likelihood is 1. To maximize the lower-bound value of p_e, p_{th} should be 0.5. Then n_{th} can be defined corresponding to p_{th}, and $p(n)$ can be modeled as a step function with n_{th}, as illustrated in Figure 3.

3.2.2 Strategies for enlarging population. When a GA with a small population fails, it is straightforward to enlarge the population and try again. However, it is difficult to obtain new information from the GA besides the failure itself. Without extra information, it is reasonable to repeat the same action until the GA succeeds in finding the optimal solution. EPS, in which population size is multiplied by the same multiplier $(n, rn, r^2n\ldots)$, is one such strategy. Another strategy is to increase population size by the same amount as $n, n+N, n+2N\ldots$. Both strategies stop when the population size is larger than or equal to the threshold population size n_{th}.

Population multiplier	Threshold population size			
	4		5	
	Pop. size seq.[a]	Sum	Pop. size seq.	Sum
2	1, 2, 4	7	1, 2, 4, 8	15
3	1, 3, 9	13	1, 3, 9	13

[a] Pop. size seq. = Population size sequence.

Table 1: Example showing that the threshold population size highly impacts the total population size in EPS. Suppose the initial population size is 1. Setting the population multiplier to 2 is better than 3 when the threshold population size is 4. However, setting the population multiplier to 3 is better when the threshold population size is 5.

As described in Section 3.2.1, the total NFE consumed is proportional to the total size of used populations. We assume that the size of the last population is n_{th}. Considering the two strategies mentioned above, the total population size is $\Theta(n_{th})$ for the first strategy (EPS) as a summation of the geometric sequence and $\Theta({n_{th}}^2)$ for the second strategy as a summation of the arithmetic sequence. In this case, EPS requires a lower NFE than the other strategy which increases population size by the same amount.

3.2.3 Deriving the population multiplier. EPS starts with a small population of size n_0 and a population multiplier r. As we try populations of size n_0, rn_0, r^2n_0, ... sequentially, we can always find an i such that $r^{i-1}n_0 < n_{th} \le r^i n_0$ since n_{th} is finite. So the last population size is $r^i n_0$, and the total NFE of EPS is proportional to total population size, which is $n_0 + rn0 + r^2 n_0 + \ldots + r^i n_0$. The summation varies with different population multipliers r and threshold population sizes n_{th}. The example in Table 1 shows that the better r depends on the value of n_{th}, but we want to find the optimal r for the average case.

Let n_{th} be expressed as $xr^{i-1}n_0$, where x is between 1 and r. The total population size can be approximated to

$$\sum_{j=0}^{i} r^j n_0 \approx \frac{r^{i+1} n_0}{r-1}.$$

Then, replace n_0 with n_{th} and we get the formula expressed as

$$\frac{r^2}{r-1}\frac{n_{th}}{x}.$$

The value depends on r and n_{th}. Since our goal is to reduce the NFE in an average case, we compare the value to the minimum required NFE, which is proportional to n_{th} because it is the minimum population size the GA needs to find the optimal solution. We thus calculate the ratio between total population size and the threshold population size n_{th}, and the value is

$$\frac{r^2}{r-1}\frac{1}{x}.$$

The threshold population size n_{th} can be any value, but it is not realistic to assume a probability distribution with infinite range. According to the previous assumption, there must exist i such that $r^{i-1}n_0 < n_{th} \le r^i n_0$. As i is confirmed, we assume that n_{th} is uniformly distributed within $(r^{i-1}n_0, r^i n_0]$. In other words, x is uniformly distributed in $(1, r]$. So the expectation value of the ratio for a certain r is

$$\int_1^r \frac{1}{r-1}\frac{r^2}{(r-1)x}dx = (\frac{r}{r-1})^2 \ln r.$$

The formula has a minimum value when $r \approx 3.5$. As the result, we suggest using 3.5 instead of 2 as the population multiplier.

4 EXPERIMENT RESULTS

In this section, we first detail the setup of experiments. Then we show the comparison between the parameterless schemes. Verification of the population multiplier follows.

4.1 Experiment Setup

In this paper, we compare our scheme with EPS and Harik-Lobo scheme. The experimental settings are described as follows. Since we suggest multiplying the population size by 3.5 rather than 2, both values are used as population multipliers in the experiment for comparison. The parameterless schemes start with an initial population size of 10 as the setting in [16] and [19]. EPS is implemented with premature convergence as the termination criterion. Our scheme is implemented with the termination criterion proposed in Section 3. For Harik-Lobo scheme on hBOA, we compare our scheme with the parameter-less hBOA directly. For Harik-Lobo scheme on LT-GOMEA and DSMGA-II, the scheme we use is mainly based on the latest implementation [19]. In Harik-Lobo scheme, there is still a parameter m that controls how often to evolve larger populations. The works [11, 17, 19] suggest using 4 as m in the average case. Thus, we set m to 4 in the experiment. The implementations of the parameterless schemes for hBOA, LT-GOMEA and DSMGA-II are based on the implementation in [15], [8] and [12], respectively.

We use six different types of benchmark problems, including four linkage-underlying problems and two real-world problems. Respectively, they are concatenated trap [4], cyclic trap [24], folded trap [6], NK-landscape [18], Ising spin-glass and maximum satisfiability (MAX-SAT) problems. The formula definitions of the six types of problems are shown in Table 2. For the concatenated trap and the cyclic trap problems, the size of subproblems k is set to 5. For the folded trap problem, the size of subproblems k is set to 6. We choose three different NK-landscape problem sets: NK-S1, NK-S3, NK-S5, in which the step size s is 1, 3 and 5, respectively. These problem sets represent problems with different overlapping degrees. For NK-landscape, Ising spin-glass and MAX-SAT problems, we use 100 instances for each problem set. For the concatenated trap, the cyclic trap and the folded trap problems, we do 100 independent runs. For NK-landscape, Ising spin-glass and MAX-SAT problems, we do ten independent runs on each instance, and the results of 100 instances are averaged. In each run, the parameterless schemes are executed until the optimal solution for the problem is found.

4.2 Results and Discussion

Now we compare our scheme with EPS and Harik-Lobo scheme. We use these parameterless schemes on hBOA, LT-GOMEA and DSMGA-II. The results are discussed sequentially. The average improvement of our scheme and the comparison with P3 follow.

Problem	Definition
Concatenated trap	$f_{k,m}^{trap}(\mathbf{x}) = \Sigma_{i=1}^{m} f_{k}^{trap}\left(\Sigma_{j=i\cdot k-k+1}^{i\cdot k}\mathbf{x}_j\right)$, where $f_{k}^{trap}(u) = \begin{cases} 1 & \text{if } u = k, \\ \frac{k-1-u}{k} & \text{otherwise.} \end{cases}$
Cyclic trap	$f_{k,m}^{cyclic}(\mathbf{x}) = \Sigma_{i=1}^{m} f_{k}^{trap}\left(\Sigma_{j=i\cdot k-i-k+2}^{i\cdot k-i+1}\mathbf{x}_j\right)$, where $f_{k}^{trap}(u) = \begin{cases} 1 & \text{if } u = k, \\ \frac{k-1-u}{k} & \text{otherwise.} \end{cases}$ and $x_{\ell+1} = x_1$.
Folded trap	$f_{k=6,m}^{folded}(\mathbf{x}) = \Sigma_{i=1}^{m} f_{k=6}^{folded}\left(\Sigma_{j=i\cdot k-k+1}^{i\cdot k}\mathbf{x}_j\right)$, where $f_{k=6}^{folded}(u) = \begin{cases} 1 & \text{if } \lvert u-3\rvert = 3, \\ 0 & \text{if } \lvert u-3\rvert = 2, \\ 0.4 & \text{if } \lvert u-3\rvert = 1, \\ 0.8 & \text{if } \lvert u-3\rvert = 0. \end{cases}$
NK-landscape	$f_{l,k,s}^{NK}(\mathbf{x}) = \Sigma_{i=1}^{\frac{(l-k-1)}{s}} f_{i,k}^{subNK}(\mathbf{x}_{i\cdot s+1}, \mathbf{x}_{i\cdot s+2}, ..., \mathbf{x}_{i\cdot s+k})$, where $f_{i,k}^{subNK} \in [0,1]$ for any $\mathbf{x}$.
Ising spin-glass	$f_{n}^{spin}(\mathbf{x}) = -\Sigma_{i,j=1}^{n}\mathbf{x}_i\mathbf{x}_j J_{ij}$, where n is the number of total pairs.
MAX-SAT	$f(\mathbf{x}) = \cap_{i=1}^{m}(\cup_{j=1}^{k_i}\mathbf{x}_{ij})$, where m is the number of clauses, k_i is the number of literals in the i-th clause.

Table 2: Definitions of test problems. We denote the problem size as ℓ, a chromosome as a vector $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_\ell)$, and the number of 1's of the subfunctions in the trap problems as u.

4.2.1 Results of the schemes on hBOA. Since hBOA uses RTR, the population changes if and only if average fitness of the population becomes better. In this case, our termination criterion is identical to premature convergence in EPS, so EPS is omitted in this comparison. The results of our scheme and the parameter-less hBOA are shown in Table 3. Our scheme outperforms the parameter-less hBOA on the folded trap, NK-landscape, and two real-world problems. The results also show that our scheme requires a lower NFE when the population multiplier is set to 3.5.

Problems	**Ours**		**Parameter-less hBOA**
	Population multiplier		
	2	3.5	
Concatenated trap	92.8	61.5	56.3
Cyclic trap	170	88.8	75.1
Folded trap	57.1	41.2	91.4
NK-S1	67.5	48.5	90.9
NK-S3	67.4	46.3	84.3
NK-S5	34.8	25.5	54.5
Ising spin-glass	6.3	4.5	18.5
MAX-SAT	22.8	21.9	27.0
			Unit: 10^4 NFE

Table 3: Results of our scheme on hBOA and the parameter-less hBOA. The problem size is 60 in the folded trap problem, 50 in the MAX-SAT problem, and 100 in other problems.

4.2.2 Results of the schemes on LT-GOMEA. The results are shown in Table 4. Our scheme outperforms Harik-Lobo scheme on most of the test problems. Our scheme perform worse on the folded trap problem, on which LT-GOMEA converges late. Changing the population multiplier from 2 to 3.5 benefits our scheme on the folded trap, NK-S1 and NK-S3 problems. In most cases, Harik-Lobo scheme performs worse after the change, since larger populations in Harik-Lobo scheme consume higher NFE when the population multiplier increases.

There are no obvious performance differences between our scheme and EPS in most cases, but EPS performs worse on problems with large plateaus in the landscape, such as Ising spin-glass problems. EPS even fails in almost half of the runs with the population multiplier set to 2 when solving the 240-bit folded trap problem, on which LT-GOMEA continuously generates new chromosomes and reaches premature convergence late. EPS also fails more when solving 100-bit MAX-SAT problems. For these problems, our scheme empirically reduces the number of failures, since the new termination criterion stops the population from useless exploration.

4.2.3 Results of the schemes on DSMGA-II. The results are shown in Table 5. On DSMGA-II, using 3.5 as the population multiplier also reduces the NFE in our scheme and EPS in most cases. Our scheme does not outperform Harik-Lobo scheme on the concatenated trap and the cyclic trap problem, because DSMGA-II usually solves these problems in one or two generations with sufficient population size, so larger populations are not initialized in Harik-Lobo scheme. It is worth to mention that changing the population multiplier benefits Harik-Lobo scheme on these two problems. Nevertheless, our scheme outperforms Harik-Lobo scheme on the other problems.

Similar to the LT-GOMEA results, there are no obvious performance differences between our scheme and EPS in most cases, and our scheme outperforms EPS on Ising spin-glass problems. However, our scheme performs worse than EPS on MAX-SAT problems. When solving MAX-SAT problems, our scheme tends to try larger populations because our termination criterion terminates populations too early.

4.2.4 Average improvement of our scheme. Table 6 shows the average improvement of our scheme compared to the other two schemes on the eight test problems. We compare with the original version of EPS and Harik-Lobo scheme, in which the population multiplier is set to 2. The results show that our scheme outperforms the other two schemes on average, and the improvement increases in most cases when the population multiplier is set to 3.5.

Problems	Ours		EPS		Harik-Lobo	
	Population multiplier					
	2	3.5	2	3.5	2	3.5
Concatenated trap	30.9	36.7	33.0	34.6	33.9	40.0
Cyclic trap	84.9	84.5	80.8	85.7	112	162
Folded trap	2860	1676	22594[a]	7879	2348	1118
NK-S1	230	199	227	207	289	336
NK-S3	229	202	230	195	286	322
NK-S5	64.4	70.9	65.1	71.3	79.1	80.6
Ising spin-glass	74.7	82.7	116	120	101	127
MAX-SAT	255	356[b]	918[c]	1260[c]	333[b]	350[d]
Unit: 10^4 NFE						

[a] There are 47 failures in 100 runs.
[b] The scheme fails several times in 10 runs on 1 instance.
[c] The scheme fails several times in 10 runs on 9 instances.
[d] The scheme fails several times in 10 runs on 2 instances.

Table 4: Results of the parameterless schemes on LT-GOMEA. The problem size is 240 in the folded trap problem, 100 in the MAX-SAT problem, and 400 in other problems.

Problems	Ours		EPS		Harik-Lobo	
	Population multiplier					
	2	3.5	2	3.5	2	3.5
Concatenated trap	11.1	13.8	11.6	9.8	10.8	8.9
Cyclic trap	24.7	23.3	25.6	24.4	23.5	20.7
Folded trap	31.8	28.8	31.8	33.2	67.2	92.0
NK-S1	178	154	179	151	286	636
NK-S3	140	112	137	109	198	375
NK-S5	20.3	16.6	20.6	16.7	27.1	39.0
Ising spin-glass	59.4	41.8	98.5	65.6	86.4	79.5
MAX-SAT	48.1	51.9	43.3	57.1	71.3	207
Unit: 10^4 NFE						

Table 5: Results of the parameterless schemes on DSMGA-II. The problem size is 240 in the folded trap problem, 100 in the MAX-SAT problem, and 400 in other problems.

Comparison		hBOA	LT-GOMEA	DSMGA-II
Ours (r = 2) [a]	**EPS**	0% [b]	6.2% [c]	4.5%
	Harik-Lobo	1.2%	13.8% [d]	25.0%
Ours (r = 3.5) [a]	**EPS**	28.5%	4.7% [c]	11.0%
	Harik-Lobo	33.4%	19.1% [d]	29.6%

[a] r = Population Multiplier.
[b] On hBOA, our scheme is identical to EPS when the population multiplier is 2.
[c] The folded trap and MAX-SAT problems are not counted in the average.
[d] MAX-SAT problems are not counted in the average.

Table 6: The average improvement of our scheme compared with the other two parameterless schemes.

4.2.5 Comparison with P3. Here we compare our scheme (r = 3.5) with P3, one of the most efficient parameterless GAs. P3 requires 71, 143, 6831, 1900, 2103, 400, 183 and 151 thousand function evaluations for the concatenated trap, the cyclic trap, the folded trap, NK-S1, NK-S3, NK-S5, Ising spin-glass and MAX-SAT problems, respectively (Problem sizes are as described in Table 4 and Table 5). P3 outperforms our scheme on LT-GOMEA in all test problems except NK-S3. On DSMGA2, our scheme outperforms P3 on folded trap and NK-landscape problems and performs worse on the other problems. However, P3 modifies the process of LT-GOMEA [8], and our scheme does not change the GA process.

4.3 Verification for Population Multiplier

The experiment results shows that our scheme requires a lower NFE when the population multiplier is set to 3.5 in the average case. However, we do not know if taking other population multipliers is better in practice. To verify our analysis in Section 3.2, we test our scheme on several problems and see if the average result is consistent with the analysis.

Three different problem sets (NK-S1, NK-S3, NK-S5) are used for verification. Since these problem sets are randomly generated and consist of subproblems which overlap in different degrees, we believe that these problems represent most problems that are composed of subproblems. Moreover, since the problems consist of building blocks of the same size, we can estimate the needed population size by supply models. For each problem set, we use four different problem sizes: 50, 100, 200, 400. According to supply model theory, the needed population size is $\Theta(2^k \ln m) = \Theta(2^k \ln \ell)$, where m is the number of building blocks, k is the order of each building block, and ℓ is the problem size. So as problem size grows exponentially, the needed population size increases linearly. We therefore assume that the needed population size is uniformly distributed in some range for each problem set.

We use LT-GOMEA and DSMGA-II for the verification. In the experiment, the population multiplier r is scanned from 2 to 5, increasing by 0.1. To calculate the average ratio for each population multiplier, we use the average NFE of our scheme from the results in Section 4.2, and the results from previous research for the optimal NFE needed by the GAs.

Figure 4 shows the verification results. For LT-GOMEA, the minimum average ratio appears in the interval between r = 2.5 and r = 4.1. For DSMGA-II, the minimum average ratio appears in the interval between r = 2.4 and r = 3.6. The variances are high since the problems have different optimal population multipliers. Nevertheless, the results show that setting the population multiplier to 3.5 is better than 2 for our scheme.

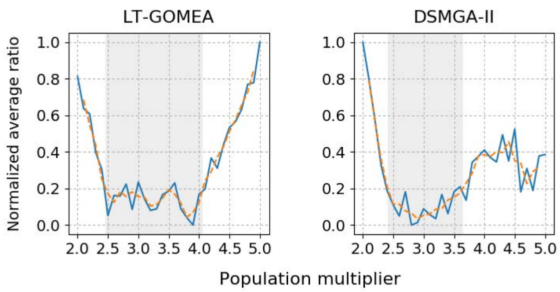

Figure 4: The average ratios between the NFE of our scheme and the optimal NFE while using LT-GOMEA and DSMGA-II. The blue lines are the normalized average ratios, and the orange dashed lines are moving averages of the normalized average ratios. The ranges where the moving averages < 0.2 are marked with gray shading.

5 CONCLUSIONS

In this paper, we modify EPS by using a new termination criterion and an optimized population multiplier. The new termination criterion decides more robust timing for terminating a population, and the optimized population multiplier makes our modified scheme more efficient in terms of NFE. We also provide theoretical analyses of our modifications. Our scheme is applicable on modern GAs such as hBOA, LT-GOMEA and DSMGA-II. The experiment results show that our scheme reduces NFE by 33.4%, 19.1% and 29.6% on hBOA, LT-GOMEA and DSMGA-II respectively when compared to Harik-Lobo scheme, and reduces NFE by 28.5%, 4.7% and 11.0% on hBOA, LT-GOMEA and DSMGA-II respectively when compared to EPS. In addition, on problems with large plateaus in the landscape, such as the folded trap and Ising spin-glass problems, our scheme considerably outperforms EPS, since our termination criterion avoids unnecessary exploration when the population is not large enough to find the optimal solution.

As for future work, we would like to research methods for determining suitable population size to solve the problem. In our scheme, we simply try a larger population if the current population seems unpromising. If we can tell the population size is large enough or not, we can keep trying the GA with sufficient population size until success, reducing the NFE needed for maintaining a larger population. Since the information from studies of fitness might not be sufficient, further investigation is needed.

ACKNOWLEDGMENTS

The authors would like to thank the Ministry of Science and Technology in Taiwan for their support under Grant No. MOST 106-2628-E-002-011-MY2.

REFERENCES

[1] W. den Besten. 2015. *Parameter-less GOMEA.* Master's thesis. Utrecht, Utrecht, Netherlands.

[2] C.-H. Chang and T.-L. Yu. 2016. Investigation on Parameterless Schemes for DSMGA-II. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion (GECCO '16 Companion).* ACM, New York, NY, USA, 85–86.

[3] D. E. Goldberg, K. Sastry, and T. Latoza. 2001. On the supply of building blocks. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation (GECCO '01).* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 336–342.

[4] K. Deb and D. E. Goldberg. 1994. Sufficient conditions for deceptive and easy binary functions. *Annals of Mathematics and Artificial Intelligence* 10, 4 (1994), 385–408.

[5] D. E. Goldberg, K. Deb, and J. H. Clark. 1992. Genetic Algorithms, Noise, and the Sizing of Populations. *Complex Systems* 6 (1992), 333–362.

[6] D. E. Goldberg, K. Deb, and J. Horn. 1992. *Massive Multimodality, Deception, and Genetic Algorithms.* Technical Report.

[7] D. E. Goldberg, K. Deb, and D. Thierens. 1993. Toward a Better Understanding of Mixing in Genetic Algorithms. *Journal of the Society of Instrument and Control Engineers* 32, 1 (1993), 10–16.

[8] B. W. Goldman and W. F. Punch. 2015. Fast and Efficient Black Box Optimization Using the Parameter-less Population Pyramid. *Evolutionary Computation* 23, 3 (2015), 451–479.

[9] J. Grefenstette. 1986. Optimization of Control Parameters for Genetic Algorithms. *IEEE Trans. Syst. Man Cybern.* 16, 1 (1986), 122–128.

[10] G. R. Harik. 1995. Finding Multimodal Solutions Using Restricted Tournament Selection. In *Proceedings of the Sixth International Conference on Genetic Algorithms.* Morgan Kaufmann, 24–31.

[11] G. R. Harik and F. G. Lobo. 1999. A Parameter-less Genetic Algorithm. In *Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation - Volume 1 (GECCO '99).* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 258–265.

[12] S.-H. Hsu and T.-L. Yu. 2015. Optimization by Pairwise Linkage Detection, Incremental Linkage Set, and Restricted / Back Mixing: DSMGA-II. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO '15).* ACM, New York, NY, USA, 519–526.

[13] F. G. Lobo and C. F. Lima. 2007. Adaptive Population Sizing Schemes in Genetic Algorithms. In *Parameter Setting in Evolutionary Algorithms.* Springer Berlin Heidelberg, Berlin, Heidelberg, 185–204.

[14] M. Pelikan. 2005. Probabilistic Model-Building Genetic Algorithms. In *Hierarchical Bayesian Optimization Algorithm: Toward a new Generation of Evolutionary Algorithms.* Springer Berlin Heidelberg, Berlin, Heidelberg, 13–30.

[15] M. Pelikan and D. E. Goldberg. 2001. Escaping Hierarchical Traps with Competent Genetic Algorithms. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation (GECCO '01).* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 511–518.

[16] M. Pelikan and T.-K. Lin. 2004. Parameter-Less Hierarchical BOA. In *Genetic and Evolutionary Computation – GECCO 2004: Genetic and Evolutionary Computation Conference, Seattle, WA, USA, June 26-30, 2004. Proceedings, Part II*, K. Deb (Ed.). Springer Berlin Heidelberg, Berlin, Heidelberg, 24–35.

[17] M. Pelikan and F. Lobo. 1999. *Parameter-less Genetic Algorithm: A Worst-case Time and Space Complexity Analysis.* Technical Report.

[18] M. Pelikan, K. Sastry, D. E. Goldberg, M. V. Butz, and M. Hauschild. 2009. Performance of Evolutionary Algorithms on NK Landscapes with Nearest Neighbor Interactions and Tunable Overlap. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation (GECCO '09).* ACM, New York, NY, USA, 851–858.

[19] J. C. Pereira and F. G. Lobo. 2015. A Java Implementation of Parameter-less Evolutionary Algorithms. (2015). arXiv:1506.08694 http://arxiv.org/abs/1506.08694

[20] D. Thierens. 2010. The Linkage Tree Genetic Algorithm. In *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature: Part I (PPSN'10).* Springer-Verlag, Berlin, Heidelberg, 264–273.

[21] D. Thierens and P. A. Bosman. 2011. Optimal Mixing Evolutionary Algorithms. In *Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation (GECCO '11).* ACM, New York, NY, USA, 617–624.

[22] D. Thierens and P. A. Bosman. 2013. Hierarchical Problem Solving with the Linkage Tree Genetic Algorithm. In *Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation (GECCO '13).* ACM, New York, NY, USA, 877–884.

[23] Y.-F. Tung and T.-L. Yu. 2015. Theoretical Perspective of Convergence Complexity of Evolutionary Algorithms Adopting Optimal Mixing. In *Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation (GECCO '15).* ACM, New York, NY, USA, 535–542.

[24] T.-L. Yu, K. Sastry, and D. E. Goldberg. 2005. Linkage Learning, Overlapping Building Blocks, and Systematic Strategy for Scalable Recombination. In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation (GECCO '05).* ACM, New York, NY, USA, 1217–1224.

[25] T.-L. Yu, K. Sastry, and D. E. Goldberg. 2005. Online population size adjusting using noise and substructural measurements. In *2005 IEEE Congress on Evolutionary Computation*, Vol. 3. 2491–2498.

[26] T.-L. Yu, K. Sastry, D. E. Goldberg, and M. Pelikan. 2007. Population Sizing for Entropy-based Model Building in Discrete Estimation of Distribution Algorithms. In *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation (GECCO '07).* ACM, New York, NY, USA, 601–608.