Adaptive Plan System of Swarm Intelligent using Differential Evolution with Genetic Algorithm*

Hieu PHAM** and Hiroshi HASEGAWA**
** College of Systems Engineering and Science, Shibaura Institute of Technology
307 Fukasaku, Minuma-ku, Saitama-shi, Saitama 337-8570, Japan
E-mail: pnh112@gmail.com

Abstract
This paper describes a new proposed strategy for Adaptive Plan System of Swarm Intelligent – Particle Swarm Optimization (PSO) using Differential Evolution (DE) with Genetic Algorithm (GA) called DE/PSOGA to solve large scale optimization problems, to reduce calculation cost, and to improve convergence towards the optimal solution. This is an approach that combines the global search ability of DE, GA and the local search ability of Adaptive plan (AP). The proposed strategy incorporates concepts from DE and PSO, updating particles not only by DE operators but also by mechanism of PSO for Adaptive System (AS) with GA. To evaluate its performance, the DE/PSOGA is applied to various benchmark tests with multi-dimensions. It is shown to be statistically significantly superior to other Evolutionary Algorithms (EAs), and Memetic Algorithms (MAs). We confirmed satisfactory performance through various benchmark tests.

Key words: Adaptive Plan, Differential Evolution, Genetic Algorithm, Multi-Peak Problems, Particle Swarm Optimization

1. Introduction
Evolutionary Algorithms (EAs) have been widely applied to solve complex numerical optimization problems, especially the multi-peak problems with multi-dimensions. The most popular EA, Genetic Algorithm (GA)(1),(2) has been applied to various multi-peak optimization problems, and its validity has been reported by many researchers. However, it requires a huge computational cost to obtain stability in convergence towards an optimal solution. To reduce the cost and to improve the stability, a strategy that combines global and local search methods becomes necessary. As for this strategy, current research has proposed various methods. For instance, Memetic Algorithms (MAs)(3)-(8) are a class of stochastic global search heuristics in which EAs-based approaches are combined with local search techniques to improve the quality of the solutions created by evolution. MAs have proven very successful across the search ability for multi-peak functions with multi-dimensions(3). These methodologies need to choose suitably a best local search method from various local search methods for combining with a global search method within the optimization process. Furthermore, since genetic operators are employed for a global search method within these algorithms, design variable vectors (DV) which are renewed via a local search are encoded into its genes many times at its GA process. These certainly have the potential to break its improved chromosomes via gene manipulation by GA operators, even if these approaches choose a proper survival strategy.

To solve these problems and maintain the stability of the convergence towards an optimal solution for multi-peak optimization problems with multiple dimensions, Hieu Pham et al. proposed new evolutionary strategies of Adaptive Plan System with Genetic Algorithm (APGAs) - simple APGA, APGA using the Variable Neighborhood Range Control (APGA/VNC),
and Hybrid neighborhood control with APGA (H-APGA)\(^9\). It is shown to be statistically significantly superior to other EAs and MAs.

Among the modern meta-heuristic algorithms, a well-known branch is Particle Swarm Optimization (PSO)\(^{10,11}\). PSO is a robust stochastic optimization algorithm which is defined by the behavior of a swarm of particles in a multidimensional search space looking for the best solution. It has been developed through simulation of a simplified social system, and has been found to be robust in solving optimization problems. PSO is the method using simple iterative calculations, thus it is easy to create the program source. Therefore, PSO is applicable to wide-ranging optimization problems. Nevertheless, the performance of the PSO greatly depends on its parameters and it often suffers from the problem of being trapped in the local optimum. It might be difficult to find the global optimal solution when it comes to complex objective functions which have a lot of local optimal solutions. The main problem of PSO is that it prematurely converges to stable point which is not necessary optimum. To resolve this problem, various improvement algorithms are proposed to be a successful in solving a variety of optimal problems\(^{12–14}\). Another active research trend in PSO is hybrid PSO, which combines PSO with other evolutionary paradigms such as Particle Swarm Inspired Evolutionary Algorithm (PS-EA)\(^{15}\), Hybrid Genetic Algorithm and Particle Swarm Optimization (GA-PSO)\(^{16}\), and Particle Swarm Ant Colony Optimization (PSACO)\(^{17}\) etc. All techniques have also been hybridized with traditional PSO to enhance performance and to prevent the swarm from crowding too closely and to locate as many optimal solutions as possible.

A new evolutionary algorithm known as Differential Evolutionary (DE) was recently introduced and has garnered significant attention in the research literature\(^{18}\). DE has many advantages including simplicity of implementation, reliable, robust, and in general is considered as an effective global optimization algorithm\(^{19}\). DE operates through similar computational steps as employed by a standard EA. However, unlike traditional EAs, the DE variants perturb the current generation population members with the scaled differences of randomly selected and distinct population members. Therefore, no separate probability distribution has to be used for generating the offspring\(^{20}\). Recently, DE has drawn the attention of many researchers all over the world resulting in a lot of variants of the basic algorithm with improved performance such as Self-adaptive DE (SaDE)\(^{21}\), DE with Neighborhood-Based Mutation (DEGL)\(^{22}\), Self-adaptive control parameters DE (jDE)\(^{23}\) and Advanced DE (ADE)\(^{24}\) etc. Compared with PSO technique and other forms of EAs\(^{25}\), it hardly requires any parameter tuning and is very efficient and reliable.

As PSO has memory, knowledge of good solutions is retained by all particles, whereas in DE, previous knowledge of the problem is discarded once the population changes. Moreover PSO and DE both work with an initial population of solutions. Therefore, combining the searching ability of these methods seems to be a reasonable approach\(^{26}\).

In this paper, we purposed a new strategy for Adaptive Plan System of PSO using DE with GA to solve large scale optimization problems, to reduce a large amount of calculation cost, and to improve the convergence towards the optimal solution called DE/PSOGA.

The remainder of this paper is organized as following manner. The concept of DE is described in Sect. 2, Sect. 3 explains the algorithm of proposed strategy (DE/PSOGA), and Sect. 4 discusses about the convergence to the optimal solution using five benchmark test functions. Finally, Sect. 5 includes some brief conclusions.

2. Differential Evolution

Differential Evolution (DE) is an EA proposed by Storn and Price\(^{18}\), also a population-based heuristic algorithm, which is simple to implement, requires little or no parameter tuning and is known for its remarkable performance for combinatorial optimization.
2.1. DE Basic Concepts

DE is similar to other EAs particularly GA in the sense that it uses the same evolutionary operators such as selection, recombination, and mutation. However the significant difference is that DE uses distance and direction information from the current population to guide the search process. The performance of DE depends on the manipulation of target vector and difference vector in order to obtain a trial vector.

2.1.1. Mutation

The main scheme in DE becomes mutation operator. For each target vector \( X_i,G \) in a \( D \)-dimensional search space, the most useful strategies of a mutant vector are as follows:

**DE/rand/1:**

\[
V_{i,G} = X_{r_1,G} + F \cdot (X_{r_2,G} - X_{r_3,G})
\]

**DE/best/1:**

\[
V_{i,G} = X_{best,G} + F \cdot (X_{r_2,G} - X_{r_3,G})
\]

**DE/target-to-best/1:**

\[
V_{i,G} = X_{i,G} + F \cdot (X_{best,G} - X_{r_3,G}) + F \cdot (X_{r_2,G} - X_{r_3,G})
\]

**DE/best/2:**

\[
V_{i,G} = X_{best,G} + F \cdot (X_{r_2,G} - X_{r_3,G}) + F \cdot (X_{r_1,G} - X_{r_3,G})
\]

**DE/rand/2:**

\[
V_{i,G} = X_{r_1,G} + F \cdot (X_{r_2,G} - X_{r_3,G}) + F \cdot (X_{r_4,G} - X_{r_3,G})
\]

where \( r_1, r_2, r_3, r_4, r_5 \) ∈ \([1, 2, \ldots, NP]\) are mutually exclusive randomly chosen integers with an initiated population of \( NP \), and all are different from the base index \( i \). \( G \) denotes subsequent generations, and \( F > 0 \) is a scaling factor which controls the amplification of differential evolution. \( X_{best,G} \) is the best individual vector with the best fitness (lowest objective function value for a minimization) in the population.

2.1.2. Crossover

To enhance the potential diversity of the population, a crossover operation is introduced. The donor vector exchanges its components with the target vector to form the trial vector:

\[
U_{i,G+1} = \begin{cases} 
V_{i,G+1}, (rand_j \leq CR) \text{ or } (j = j_{rand}) \\
X_{i,G+1}, (rand_j \geq CR) \text{ and } (j \neq j_{rand}) 
\end{cases}
\]

where \( j \in [1, 2, \ldots, D] \); \( rand_j \) ∈ [0.0, 1.0]; \( CR \) is the crossover probability takes value in the range [0.0,1.0], and \( j_{rand} \in [1, 2, \ldots, D] \) is the randomly chosen index.

2.1.3. Selection

To determine whether the target vector or the trial vector survives to the next generation, selection is performed. The selection operation is described as:

\[
X_{i,G+1} = \begin{cases} 
U_{i,G}, & f(U_{i,G}) \leq f(X_{i,G}) \\
X_{i,G}, & f(U_{i,G}) > f(X_{i,G}) 
\end{cases}
\]

2.2. DE Variants

In this section, we discuss about an introduction of the most prominent DE variants that were developed and appeared to be competitive against the existing best-known real parameter optimizers.

460
2.2.1. DE using Arithmetic Recombination

To make the recombination process of DE rotationally invariant, Price proposed a new trial vector generation strategy "DE/current-to-rand/1"(27), which replaces the binominal crossover operator with the arithmetic recombination operator to generate the trial vector as follows:

\[ U_{i,G} = X_{i,G} + k_i \cdot (X_{r1,G} - X_{r2,G}) \]  

Now incorporating (1) in (9) and further simplifying we have:

\[ U_{i,G} = X_{i,G} + k_i \cdot (X_{r1,G} - X_{r2,G}) + F' \cdot (X_{r2,G} - X_{r3,G}) \]  

where \( k_i \) is the combination coefficient, which can be a constant or a random variable distribution from \([0.0,1.0]\), and \( F' = k_i \cdot F \) is a new constant parameter.

2.2.2. DE with Adaptive Selection

Qin et al. proposed a self-adaptive variant of DE (SaDE), where along with the control parameter values the trial vector generation strategies are also gradually self-adapted by learning from their previous experiences in generating promising solutions. Consequently, it is possible to determine a more suitable generation strategy along with its parameter settings adaptively to match different phases of the search process.

In the SaDE algorithm, for each target vector in the current population, one trial vector generation strategy is selected from the candidate pool according to the probability learned from its success rate in generating improved solution within a certain number of previous generations. The selected strategy is subsequently applied to the corresponding target vector to generate a trial vector. More specifically, at each generation, the probabilities of choosing each strategy in the candidate pool are initially equal \( 1/K \) for \( K \) total of strategies and are gradually adapted during evolution.

3. New Evolutionary Computation – DE/PSOGA

With a view to global search, we proposed the new algorithm using DE for Adaptive Plan system of PSO with GA named DE/PSOGA. The DE/PSOGA aims at getting the direction from PSO operator to adjust into adaptive system of APGA using alternative operator of DE scheme. In addition, for a verification of APGA search process, refer to Ref. (9).

3.1. DE/PSOGA Algorithm

The proposed DE/PSOGA starts like the usual DE algorithm up to the point where the trial vector is generated. If the trial vector with binomial crossover operation (6) satisfies the conditions given by (7), then the algorithm enters the PSO operator to get the direction and generates a new candidate solution with adaptive system of APGA. The inclusion of APGA process turns helps in maintaining diversity of the population and reaching a global optimal solution. The flow-chart and pseudocode of DE/PSOGA algorithm are respectively shown in Fig. 1 and Fig. 2.

DE update strategies using adaptive selection of three effective strategies:
- DE/best/1 (2)
- DE/target-to-best/1 (3).
- DE using arithmetic recombination (9).

3.2. DE Control Parameters

In this paper, a self-adaption DE control parameters has been adapted for the efficiency of the DE/PSOGA(23). The control parameters \( F \) and \( CR \) are encoded into the individual and adjusted by introducing new parameters \( \tau_1, \tau_2 \). The new control parameters for next generation are computed as:

\[ F_{i,G+1} = \begin{cases} F_{i} + \text{rand} \cdot F_u & \text{with probability } \tau_1 \\ F_{i,G} & \text{else} \end{cases} \]  

(10)
Begin
  Initialize population;
  Generate initial DVs;
  Evaluate individuals with initial DVs;
  while (TERMINATION CONDITION) do
    Generate DVs by DE update strategies;
    Evaluate individuals with DVs;
    Update velocity by PSO operator;
    Generate new DVs via AP with GA;
  end while
  Renew population;
End

Fig. 1 Flow-chart of DE/PSOGA algorithm

Begin
  Initialize population;
  Generate initial DVs;
  Evaluate individuals with initial DVs;
  while (TERMINATION CONDITION) do
    Generate DVs by DE update strategies;
    Evaluate individuals with DVs;
    Update velocity by PSO operator;
    Generate new DVs via AP with GA;
  end while
  Renew population;
End

Fig. 2 Pseudocode of DE/PSOGA algorithm

\[ CR_{i,G+1} = \begin{cases} 
  \text{rand}_3 & \text{with probability } \tau_2 \\
  CR_{i,G} & \text{else}
\end{cases} \] (11)

where \( F_l \) and \( F_u \) are the lower and upper limits of \( F \), \( F_l = 0.1 \) and \( F_u = 0.9 \), and \( \tau_1 = \tau_2 = 0.1 \). The new \( F \) and \( CR \), obtained before the mutation is performed, take value from \([0.0, 1.0]\).

3.3. PSO Operator

We are concerned here with conventional basic model of PSO\(^{(10)}\). In this model, each particle which make up a swarm has information of its position \( x_i \) and velocity \( v_i \) (where \( i \) is the index of the particle) at the present in the search space. Each particle aims at the global optimal solution by updating next velocity making use of the position at the present, based on its best solution has been achieved so far \( pbest_{ij} \) and the best solution of all particles \( gbest_j \) (where \( j = [1, 2, \ldots, D] \), \( D \) is the dimension of the solution vector), as following equation:

\[ v_{ij,G+1} = w v_{ij,G} + c_1 r_1 \left( pbest_{ij,G} - X_{ij,G} \right) + c_2 r_2 \left( gbest_{j,G} - X_{ij,G} \right) \] (12)

where \( w \) is inertia weight; \( c_1 \) and \( c_2 \) are cognitive acceleration and social acceleration, respectively; \( r_1 \) and \( r_2 \) are random numbers uniformly distributed in the range \([0.0, 1.0]\). After a number of iterations, PSO is going to get the global optimal solution as conclusive \( gbest \).

In our strategy, the concept of time-varying has been adapted\(^{(28)}\). The inertia weight \( w \) in
(12) linearly decreasing with the iterative generation as below:

\[ w = \left( w_{\text{max}} - w_{\text{min}} \right) \frac{\text{iter} - \text{iter}_{\text{max}}}{\text{iter}_{\text{max}}} + w_{\text{min}} \]  

(13)

where \( \text{iter} \) is the current iteration number while \( \text{iter}_{\text{max}} \) is the maximum number of iterations, the maximal and minimal weights \( w_{\text{max}} \) and \( w_{\text{min}} \) are respectively set 0.9, 0.4 known from experience.

The concept of diversification and intensification is quite important in PSO algorithm, because it decides the characteristic of the swarm and the search performance. By using (13), the particles can be transformed from diversification to intensification by decreasing the inertia weight linearly as the search proceeds.

The acceleration coefficients \( c_1 \) and \( c_2 \) are also important parameters in PSO. Both acceleration coefficients are essential to the success of PSO. The idea behind time-varying acceleration coefficients is to enhance the global search in early part of the optimization and to encourage the particles to converge towards the global optima at the end of the search proceeds. With a large cognitive component and small social component at the beginning, particles are allowed to move around the search space instead of moving toward the population best during early stages. On the other hand, a small cognitive component and a large social component allow the particles to converge to the global optima in the latter part of the optimization process. The acceleration coefficients are expressed as:

\[ c_1 = (c_{1f} - c_{1i}) \frac{\text{iter} - \text{iter}_{\text{max}}}{\text{iter}_{\text{max}}} + c_{1i} \]  

(14)

\[ c_2 = (c_{2f} - c_{2i}) \frac{\text{iter} - \text{iter}_{\text{max}}}{\text{iter}_{\text{max}}} + c_{2i} \]  

(15)

where \( c_{1i}, c_{1f}, c_{2i} \) and \( c_{2f} \) are initial and final values of the acceleration coefficient factors respectively. The most effective values are set 2.5 for \( c_{1i} \) and \( c_{2f} \) and 0.5 for \( c_{1f} \) and \( c_{2i} \) as in(29).

3.4. Adaptive Plan – AP

Adaptive Plan with Genetic Algorithm (APGA)(9) that combines the global search ability of a GA and an Adaptive Plan with excellent local search ability is superior to other EAs, MAs(3). The APGA concept differs in handling Design variable vectors (DVs) from general EAs based on GAs. EAs generally encode DVs into the genes of a chromosome, and handle them through GA operators. However, APGA completely separates DVs of global search and local search methods. It encodes Control variable vectors (CVs) of AP into its genes on Adaptive system (AS). Moreover, this separation strategy for DVs and chromosomes can solve MA problem of breaking chromosomes. The control variable vectors (CVs) steer the behavior of adaptive plan (AP) for a global search, and are renewed via genetic operations by estimating fitness value. For a local search, AP generates next values of DVs by using CVs, response value vectors (RVs) and current values of DVs according to the formula:

\[ X_{i,G+1} = X_{i,G} + \text{AP}(C_G, R_G) \]  

(16)

where \( \text{AP}() \), \( X \), \( C \), \( R \), and \( G \) denote a function of AP, DVs, CVs, RVs and subsequent generation, respectively.

It is necessary that the AP realizes a local search process by applying various heuristics rules. The plan introduces a DV generation formula using velocity update from PSO operator that is effective in the convex function problem as a heuristic rule, because a multi-peak problem is combined of convex functions.

\[ \text{AP}(C_G, R_G) = \text{scale} \cdot SP \cdot PSO \cdot (\nabla R) \]  

(17)
where $\nabla R$ denote sensitivity of RVs, constriction factor scale randomly selected from a uniform distribution in [0.1,1.0], step size $SP$ and velocity update $PSO$ by (12).

A step size $SP$ is defined by CVs for controlling a global behavior to prevent it falling into the local optimum. $C = [c_{i,j}, \ldots, c_{i,p}]; (0.0 \leq c_{i,j} \leq 1.0)$ that is encoded into a chromosome by 10 bit strings (as shown in Fig. 3) is used so that it can change the direction to improve or worsen the objective function. In addition, $i$, $j$ and $p$ are the individual number, design variable number and its size, respectively.

### 3.5. GA Operators

#### 3.5.1. Selection
To maintain the diverseness of individuals with a goal of keeping off an early convergence, selection is performed using a tournament strategy, and a tournament size of 2 is used.

#### 3.5.2. Elite Strategy
An elite strategy, where the best individual survives in the next generation, is adopted during each generation process. We assume that the best individual, i.e., as for the elite individual, generates two behaviors of AP by updating DVs with AP, not GA operators. Therefore, its strategy replicates the best individual to two elite individuals, and keeps them to next generation. As shown in Fig. 4, one of them (◦ symbol) is that both DVs and CVs are not renewed, and are kept to next generation as an elite individual at the same search point. DVs of another one (∆ symbol) is renewed by AP, and its CVs which are coded into chromosome arent changed by GA operators.

#### 3.5.3. Crossover and Mutation
In order to pick up the best values of each CV, a single point crossover operator is used for the string of each CV. This can be considered to be a uniform crossover operation for the string of the chromosome as shown in Fig. 5(a). To maintain the strings diverse, mutation operator is performed and set for each string at mutation ratio on each generation as shown in Fig. 5(b).

#### 3.5.4. Recombination of Genes
At following conditions, the genetic information on chromosome of individual is recombined by uniform random function.

1. One fitness value occupies 80% of the fitness of all individuals
2. One chromosome occupies 80% of the population
4. Numerical Experiments

In this section, the numerical experiments were performed to compare among strategies. Next, the new algorithm were compared with other techniques for the robustness of the optimization approach. These experiments involved 25 independent trials for each function. In the experiments, we set the parameters as specified in Table 1. The initial seed number was randomly varied during every trial. The lower and upper limits of scaling factor $F$, $F_l = 0.1$ and $F_u = 0.9$, and the initial control parameters $F$ and $CR$ were respectively set by 0.1 and 0.5 for the performance of DE. For PSO operator, the maximal and minimal weights $w_{\text{max}}$ and $w_{\text{min}}$ were respectively set 0.9, 0.4, the acceleration coefficients were set 2.5 each for $c_{1i}$ and $c_{2f}$ and 0.5 each for $c_{1f}$ and $c_{2i}$. The GA parameters used in solving benchmark functions, selection ratio, crossover ratio and mutation ratio were set 1.0, 0.8 and 0.1 respectively. The population size had 100 individuals and the terminal generation was $1500^{th}$ generation.

### 4.1. Benchmark Tests

For the DE/PSOGA, we estimated the stability of the convergence to the optimal solution by using five benchmarks with 30, and 100 dimensions - Ridge ($f_3$), Rosenbrock ($f_5$), Rastrigin ($f_9$), Ackley ($f_{10}$), and Griewank ($f_{11}$) (see Appendix A).

Appendix B lists their characteristics, including the terms epistasis, multi-peaks, and steepness. $D$ denotes the dimensionality of the test problem, design range variables and the global optimum value are summarized. A more detailed description of each function is given in Ref. (30).
4.2. Experiment Results by DE/PSOGA

The experiment results, averaged over 25 trials with Rosenbrock function using scaling factor tuning are given in Table 2 (crossover probability $CR = 0.5$). When the success rate of optimal solution is not 100%, "-" is described. From the results via optimization experiments, we employed the best value of scaling factor $F$ for the DE/PSOGA is 0.1.

Table 3 shows the average results of all benchmark functions with 30 dimensions by the DE/PSOGA using the DE control parameters (10), (11). In addition, the experiment results with fixed total evaluation times are given in Table 4. The solution of all benchmark functions reach their global optimum solutions, and the success rate of optimal solution is 100%.

Next, Fig. 6-10 shows diagrams for average fitness values of all individuals by the best trial until the DE/PSOGA algorithm reach the global optimum solutions with 30 dimensions, again to confirm above mentioned results.

The effect of population size on the performance of algorithm is reported in Table 5. From this table, it can be concluded that as the population size increases, the performance of the DE/PSOGA rapidly deteriorates. Additionally, the results show that the DE/PSOGA algorithm is effective in all benchmarks for various population sizes. The performance of the DE/PSOGA show relatively deterioration with the growth of the population size, which suggests that the DE/PSOGA is more stable and robust on population size.

As a result, we confirmed that the DE/PSOGA algorithm could reduced the calculation cost and improved the stability of the convergence to the optimal solution.

4.3. Comparison for Robustness

To evaluate the performance of DE/PSOGA, we compared to other EAs such as GA, PSO, PS-EA in Ref. (25), ABC(31), DE(18), jDE(23), and ADE(24). Maximum number of generation and the population size, i.e. 100, as in the study presented in Ref. (15), (25). The mean and the standard deviations of the function values obtained by these methods are given in Table 6 and Table 7.

By means of the comparison with other methodologies, the DE/PSOGA could certainly achieve optimal solution with low calculation cost. Additionally, the results show that the proposed DE/PSOGA algorithm outperformed other techniques in all function. The conver-
Table 4  Experiment results, mean and standard deviations obtained by DE/PSOGA over 25 runs in terms of 150,000 FES (population size 100)

| Function | Dimension | Mean   | Std Dev |
|----------|-----------|--------|---------|
| f3       | 30        | 0.00E+00 | 0.00E+00 |
|          | 100       | 0.00E+00 | 0.00E+00 |
| f5       | 30        | 0.00E+00 | 0.00E+00 |
|          | 100       | 0.00E+00 | 0.00E+00 |
| f9       | 30        | 0.00E+00 | 0.00E+00 |
|          | 100       | 0.00E+00 | 0.00E+00 |
| f10      | 30        | 4.44E-16 | 0.00E+00 |
|          | 100       | 4.44E-16 | 0.00E+00 |
| f11      | 30        | 0.00E+00 | 0.00E+00 |
|          | 100       | 0.00E+00 | 0.00E+00 |

Table 5  Experiment results, averaged over 25 runs by DE/PSOGA with different population size \((D = 30)\)

**Population size 50**

| Function | Gen. No | Func. call | Mean   | Std Dev |
|----------|---------|------------|--------|---------|
| f3       | 240     | 12,000     | 0.00E+00 | 0.00E+00 |
| f5       | 372     | 18,100     | 0.00E+00 | 0.00E+00 |
| f9       | 278     | 13,900     | 0.00E+00 | 0.00E+00 |
| f10      | 391     | 19,550     | 4.44E-16 | 0.00E+00 |
| f11      | 336     | 16,800     | 0.00E+00 | 0.00E+00 |

**Population size 200**

| Function | Gen. No | Func. call | Mean   | Std Dev |
|----------|---------|------------|--------|---------|
| f3       | 1500    | 300,000    | 3.04E+01 | 4.56E+05 |
| f5       | 1500    | 300,000    | 1.13E+02 | 5.56E+06 |
| f9       | 1500    | 300,000    | 1.16E+00 | 2.77E+01 |
| f10      | 1500    | 300,000    | 6.89E-01 | 2.67E-01 |
| f11      | 1500    | 300,000    | 8.11E-01 | 2.68E+01 |

Fig. 6  Average fitness of all individuals in the population with Ridge function \((D = 30,\) population size 100)
Fig. 7 Average fitness of all individuals in the population with Rosenbrock function ($D = 30$, population size 100)

Fig. 8 Average fitness of all individuals in the population with Rastrigin function ($D = 30$, population size 100)

Fig. 9 Average fitness of all individuals in the population with Ackley function ($D = 30$, population size 100)
Average fitness of all individuals in the population with Griewank function ($D = 30$, population size 100)

Table 6 Comparison of GA, PSO, PS-EA, ABC and DE/PSOGA algorithm ($D = 30$, population size=100)

| Function | Gen. No | GA (25) | PSO (25) | PS-EA (25) | ABC (31) | DE/PSOGA |
|----------|---------|---------|----------|------------|----------|----------|
|          |         | Mean best (Std Dev) | Mean best (Std Dev) | Mean best (Std Dev) | Mean best (Std Dev) | Mean best (Std Dev) |
| $f_3$    | 1000    | -       | -        | -          | -        | 0.00E+00  |
|          |         | (59.5102) | (63.635) | (35.5791) | (0.152742) | (0.00E+00) |
| $f_5$    | 1000    | 166.283 | 402.54   | 98.407     | 0.219626 | 0.00E+00  |
|          |         | (2.6386) | (6.9521) | (0.9988)   | (0.181557) | (0.00E+00) |
| $f_9$    | 1000    | 10.4388 | 32.476   | 3.0527     | 0.033874 | 0.00E+00  |
|          |         | (2.6386) | (6.9521) | (0.9988)   | (0.181557) | (0.00E+00) |
| $f_{10}$ | 1000    | 1.0989  | 1.49E-6  | 0.3771     | 3E-12    | 4.4E-16   |
|          |         | (0.42956) | (1.86E-6) | (0.009762) | (5E-12)  | (0.00E+00) |
| $f_{11}$ | 1000    | 1.2342  | 0.011151 | 0.8211     | 2.87E-09 | 0.00E+00  |
|          |         | (0.11045) | (0.014209) | (0.1394)   | (8.45E-10) | (0.00E+00) |

DE/PSOGA in term of 100,000 FES; Gen. No 1000

gence of the optimal solution could be improved more significantly in the DE/PSOGA than that in other methods for the same calculation cost. Therefore, it is desirable to introduce this strategy for global optimization.

4.4. Discussion for Improved DE/PSOGA

We shall discuss some points for improving the optimization process and overall performance of the DE/PSOGA algorithm. Proposing new mutations and adjusting control parameters are an open challenge direction of research.

Former, using a new mutation rule to enhance the local exploitation ability and to improve the convergence rate. Furthermore, a dynamic crossover probability scheme is proposed to balance the exploration and exploitation abilities.

Latter, a new approach is how to adapt automatically the control parameters using an Artificial Neural Network (ANN), or neuro-fuzzy system can be used for finding them to solve different optimization problems.

5. Conclusion

In this paper, to overcome the computational complexity, a new strategy using DE for Adaptive Plan system of PSO with GA called DE/PSOGA has been proposed to solve large scale optimization problems, and to improve the convergence to the optimal solution. Then, we verified the effectiveness of DE/PSOGA by the numerical experiments performed five benchmark functions.

We confirmed that the DE/PSOGA reduced the calculation cost and dramatically im-
Table 7  Comparison of DE, jDE, ADE and DE/PSOGA algorithm (D = 30, population size=100)

| Function | Gen. No | DE(18) | jDE(23) | ADE(24) | DE/PSOGA |
|----------|---------|---------|---------|---------|----------|
|          |         | Mean best (Std Dev) | Mean best (Std Dev) | Mean best (Std Dev) | Mean best (Std Dev) |
| f_3      | 1500    | 1.630860 (0.886153) | 0.090075 (0.080178) | - (0.00E+00) (0.00E+00) |
| f_5      | 1500    | 7.8E-09 (5.8E-09) | 3.1E-15 (8.3E-15) | 3.75E-05 (8.90E-05) | 0.00E+00 (0.00E+00) |
| f_9      | 1500    | 173.405 (13.841) | 1.5E-15 (4.8E-15) | 0.0E+00 (0.0E+00) | 0.00E+00 (0.00E+00) |
| f_10     | 1500    | 9.7E-08 (4.2E-08) | 7.7E-15 (1.4E-15) | 6.93E-11 (3.10E-11) | 4.44E-16 (0.00E+00) |
| f_11     | 1500    | 2.9E-13 (4.2E-13) | 0 (0.0E+00) | 0.0E+00 (0.0E+00) | 0.00E+00 (0.00E+00) |

DE/PSOGA in term of 150,000 FES; Gen. No 1500
(1) Gen. No 3000; (2) Gen. No 5000; (3) Gen. No 2000

proved the stability of the convergence to the optimal solution.

Overall, the DE/PSOGA was capable of attaining robustness, high quality, low calculation cost and efficient performance on many benchmark problems.

Finally, this study plans to do a comparison with the sensitivity plan of the AP by applying other optimization methods and optimizing the benchmark functions, constrained real-parameters, and dynamic optimization problems.

Appendix A. Benchmark Tests

Ridge : \( f_3 = \sum_{i=1}^{D} \left( \sum_{j=1}^{i} x_j \right)^2 \)  \hspace{1cm} (A.1)

Rosenbrock : \( f_5 = \sum_{i=1}^{D} [100(x_{i+1} + 1 - (x_i + 1)^2) + x_i^2] \)  \hspace{1cm} (A.2)

Rastrigin : \( f_9 = 10D + \sum_{i=1}^{D} [x_i^2 - 10\cos(2\pi x_i)] \)  \hspace{1cm} (A.3)

Ackley : \( f_{10} = -20 \exp \left( -0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right) - \exp \left( \frac{1}{n} \sum_{i=1}^{n} \cos (2\pi x_i) \right) + 20 + e \)  \hspace{1cm} (A.4)

Griewank : \( f_{11} = 1 + \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos \left( \frac{x_i}{\sqrt{i}} \right) \)  \hspace{1cm} (A.5)

Appendix B.

Table 8  Characteristics of Benchmark tests

| Function | Epistasis | Multi-peaks | Steepness | Design range | Optimum |
|----------|-----------|-------------|-----------|--------------|---------|
| f_3      | Yes       | No          | Average   | \([-100, 100]^D\] | f(0) = 0 |
| f_5      | Yes       | No          | Big       | \([-30, 30]^D\] | f(0) = 0 |
| f_9      | No        | Yes         | Average   | \([-5.12, 5.12]^D\] | f(0) = 0 |
| f_{10}   | No        | Yes         | Average   | \([-32, 32]^D\] | f(0) = 0 |
| f_{11}   | Yes       | Yes         | Small     | \([-600, 600]^D\] | f(0) = 0 |

\( D \) denotes the dimensionality of the test problem
Fig. 11  Benchmarks: (a) Ridge; (b) Rosenbrock; (c) Rastrigin; (d) Ackley; (e) Griewank
References

(1) Holland, J., Adaptation in Natural and Artificial Systems, *The University of Michigan 1975, MIT Press*, (1992).
(2) Goldberg, D.E., Genetic Algorithms in Search Optimization and Machine Learning, *Addison - Wesley*, (1989).
(3) Smith, J.E., Hart, W.E. and Krasnogor, N., Recent Advances in Memetic Algorithms, (2005), Springer.
(4) Ong, Y.S. and Keane, A.J., Meta-Lamarckian Learning in Memetic Algorithms, *IEEE Transactions on Evolutionary Computation*, Vol.8, No.2, (2004), pp.99–110.
(5) Ong, Y.S., Lim, M.H., Zhu, N. and Wong, K.W., Classification of Adaptive Memetic Algorithms: A Comparative Study, *IEEE Transactions on Systems, Man and Cybernetics Part B*, Vol.36, No.1, (2006), pp.141–152.
(6) Neri, F., Tirronen, V., Kärkkäinen, T. and Rossi, T., Fitness Diversity Based Adaptation in Multimeme Algorithms: A Comparative Study, *IEEE Congress on Evolutionary Computation (CEC)*, (2007), pp. 2374–2381.
(7) Caponio, A., Cascella, G.L., Neri, F., Salvatore, N. and Sumner, M., A Fast Adaptive Memetic Algorithm for Online and Offline Control Design of PMSM Drives, *IEEE Transactions on Systems, Man and Cybernetics Part B, Special Issue on Memetic Algorithms*, Vol.37, No.1, (2007), pp.28–41.
(8) Tirronen, V., Neri, F., Kärkkäinen, T., Majava, K. and Rossi, T., An Enhanced Memetic Differential Evolution in Filter Design for Defect Detection in Paper Production, *Evolutionary Computation Journal, MIT Press*, Vol.16, No.4, (2008), pp.529–555.
(9) Hieu Pham, Tooyama, S. and Hasegawa, H., Evolutionary Strategies of Adaptive Plan System with Genetic Algorithm, *JSME Journal of Computational Science and Technology*, Vol.6, No.3, (2012), pp.129–146.
(10) Kennedy, J. and Eberhart, R., *Swarm Intelligence*, (2001), Morgan Kaufmann Publishers.
(11) Clerc, M., *Particle swarm optimization*, (2005), ISTE.
(12) Clerc, M. and Kennedy, J., The Particle Swarm-Explosion, Stability, and Convergence in a Multidimensional Complex Space, *IEEE Transactions on Evolutionary Computation*, Vol.6, No.1, (2002), pp.58–73.
(13) Shi, Y. and Eberhart, R.C., A Modified Particle Swarm Optimizer, *Proceedings of the IEEE Congress on Evolutionary Computation*, (1998), pp.69–73.
(14) Eberhart, R.C. and Shi, Y., Tracking and Optimizing Dynamic Systems with Particle Swarms, *Proceedings of the IEEE Congress on Evolutionary Computation*, (2001), pp.94–100.
(15) Srinivasan, D. and Seow, T.H., Evolutionary Computation, *IEEE Congress on Evolutionary Computation*, (2003), pp.2292–2297.
(16) Kao, Y.T. and Zahara, E., A hybrid genetic algorithm and particle swarm optimization for multimodal functions, *Applied Soft Computing*, Vol.8, No.2, (2008), pp.849–857.
(17) Shelokar, P.S., Patrick Siarry, Jayaraman, V.K. and Kulkarni, B.D., Particle swarm and ant colony algorithms hybridized for improved continuous optimization, *Applied Mathematics and Computation*, Vol.188, No.1, (2007), pp.129–142.
(18) Storn, R. and Price, K., Differential Evolution - a simple and efficient Heuristic for global optimization over continuous spaces, *Journal Global Optimization*, Vol.11, No.4, (1997), pp.341–357.
(19) Price, K., Storn, R. and Lampinen, J., *Differential Evolution: A Practical Approach to Global Optimization*, (2005), Springer-Verlag, Berlin, Germany.
(20) Das, S. and Suganthan, P.N., Differential evolution - A survey of the State-of-the-Art, *IEEE Transactions on Evolutionary Computation*, Vol.15, No.1, (2011), pp.4–31.
(21) Qin, A.K., Huang, V.L. and P. N. S., Differential evolution algorithm with strategy adaptation for global numerical optimization, *IEEE Trans. Evol. Comput.*, (2009).
(22) Das, S., Abraham, A., Chakraborty, U.K. and Konar, A., Differential evolution using a neighborhood based mutation operator, *IEEE Trans. Evol. Comput.*, Vol.13, No.3, (2009), pp.526–553.

(23) Brest, J., Greiner, S., Bošković, B., Mernik, M. and Žumer, V., Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, *IEEE Tras. Evol. Comput.*, Vol.10, No.6, (2006), pp.646–657.

(24) Wagdy Mohamed, A., Sabry, H.Z. and Farhat, A., Advanced Differential Evolution algorithm for global numerical optimization. *IEEE International Conference on Computer Applications and Industrial Electronics (ICCAIE)*, (2011), pp.156–161.

(25) Vesterstrom, J. and Thomsen, R., A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems, *Proceedings of the 2004 Congress on Evolutionary Computation*, No.2, (2004), pp.1980–1987.

(26) Das, S., Abraham, A. and Konar, A., Particle Swarm Optimization and Differential Evolution Algorithms: Technical Analysis, Applications and Hybridization Perspectives, *Studies in Computational Intelligence (SCI)*, Vol.116, (2008), pp.1–38.

(27) Price, K.V., An introduction to differential evolution, *Corne, D., Dorigo, M. and Glover, F. (Eds.), New Ideas in Optimization*, (1999), pp.79–108. Uk, McGraw-Hill.

(28) Shi, Y. and Eberhart, RC., Empirical study of particle swarm optimization, *Proc. IEEE Int. Congr. Evol. Comput.*, Vol.3, (1999), pp.101–6.

(29) Eberhart, RC and Shi, Y., Comparing inertia weights and constriction factors in particle swarm optimization, *Proc. Congr. Evol. Comput.*, Vol.1, (2000), pp.84–8.

(30) Yao, X., Liu, Y. and Lin, G., Evolutionary programming made faster. *IEEE Trans. Evol. Comput.*, Vol.3, No.2, (1999), pp.82–102.

(31) Karaboga, D., and Basturk, B., A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm, *Journal Global Optimization*, Vol.39, (2006) pp.459–471.