Low Dimensional Reproduction Strategy for Real-coded Evolutionary Algorithms

Changtong Luo, Shaoliang Zhang
Department of Computational Science and Engineering
Nagoya University
Nagoya 464-8603, Japan
(c-luo, zhang)@na.cse.nagoya-u.ac.jp

Bo Yu
Department of Applied Mathematics
Dalian University of Technology
Dalian 116024, P.R. China
yubo@dlut.edu.cn

Abstract

The strategy of low dimensional reproduction (LDR) is proposed for real-coded evolutionary algorithms (REAs) in this paper. It preserves some (randomly chosen) components of the local best vector (elite individual) in the reproduction process and let the traditional reproduction operators act on the rest components. Thus it could help the search points escape from the hyperplane where the parents individuals lies, as well as keep them from getting too much decentralized and search mainly along a series of orthogonal directions (coordinate). The LDR strategy provides a universal idea to improve the performance of REAs. Four REAs are taken as examples to show the effect of the strategy. Numerical results show that the proposed strategy can accelerate the convergence speed of the applied algorithms considerably. In addition, the strategy is computational saving, easy to implement, and easy to control.

Keywords. global optimization, meta heuristics, evolutionary algorithm, real-coded, low dimensional reproduction strategy

1 Introduction

Real-coded evolutionary algorithms (REAs), such as real-coded genetic algorithm (RGA), evolutionary programming (EP), differential evolution (DE), and particle swarm optimization (PSO) have been widely used for global optimization problems arising from practical engineering, learning, and design. However, most of REAs face the dilemma of global search and fast convergence, maybe or partially because of their linear or component-wise properties. For example, complete arithmetic crossover in RGA

$$X_{i,j}'(t) = \lambda X_{i,j}(t) + (1 - \lambda) X_{r_1, j}(t),$$

and differential mutation in DE

$$X_{i,j}'(t) = X_{i,j}(t) + F \cdot (X_{r_2, j}(t) - X_{r_3, j}(t)),$$

and particle movement in PSO

$$X_{i,j}'(t) = X_{i,j}(t) + v_{i,j}(t),$$

search-region to a subspace, as can be shown in Fig.1(a) and (a'), if we take the arithmetic crossover as an example. New possible search points will distribute only along a line determined by their parents. So the algorithm may be easily trapped into non-global-optimum. On the other hand, some other evolutionary operators are component-wise, e.g., component-wise arithmetic crossover in RGA

$$X_{i,j}'(t) = \lambda X_{i,j}(t) + (1 - \lambda) X_{r_1, j}(t),$$

and normal mutation in EP

$$X_{i,j}'(t) = X_{i,j}(t) + N(0, \sigma_{x,j}^2),$$

where component-wise operators result in a uniformly random search-region, as can be shown in Fig.1(b) and (b'), if we take the component-wise arithmetic crossover as an example. New possible
search points will distribute uniformly in the box determined by their parents. Such uniform randomness may slow down the convergence rate in the end. Therefore, we need a proper search strategy that can keep a balance of global search and fast convergence.

In this paper, we will introduce a strategy for the reproduction of new individuals that is a mediate of linear and component-wise operators, which can keep a balance of global search and fast convergence. Meanwhile, the strategy can preserve some information (components) of elite, which can be regarded as a special way of elite-preservation. Elite-preservation strategy is a good technique to speed up the convergence rate of EAs, in which the individual with the highest fitness always survives to be an individual of next generation. Numerical results show that the proposed strategy considerably improves the performance of the applied algorithms. The rest of this paper is organized as follows. The details of the LDR strategy is described in section 2. In section 3, we applied the LDR strategy into different Real-coded EAs, and their performances are shown in section 4. Section 5 discusses the choices of the control parameters of REAs with LDR strategy and section 6 draws the concluding remarks.

2 Description of low dimensional reproduction (LDR) strategy

2.1 General description of real-coded evolutionary algorithms

Evolutionary Algorithm (EA) is a kind of population-based heuristics for global optimization problems, which is different from traditional point-to-point iteration. EA maintains a population (point set) of \( N \) individuals (points) \( X(t), i = 1, 2, \ldots, N \) at every iteration \( t \), where \( N \) is called population size. For a real-coded EA, an individual can be presented by a vector in \( R^n \) space (i.e., \( X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,n}) \)). Then the individuals in current population is replaced by new trial individuals \( X'_i(t) \) from generation to generation until some stopping criterion is satisfied. Usually there are three evolutionary processes: reproduction, evaluation and selection. Then the main role of searching is completed by the reproduction process. It creates new trial points from current points, usually the \( i \)-th point \( X_i(t) \) and some help points \( X_k(t), k = 1, 2, \ldots, m \). The evaluation process gives each point a fitness value according to the objective function \( f(x) \). Based on the evaluation, whether a current point can retain in the next iteration depends on the selection process, in which a better point has a greater chance to survive.

In this paper, we consider only the real-coded evolutionary algorithms (REAs). Different REAs have different reproduction operators, including the evolutionary operators mentioned above. However, we can unified them as \( X'_i(t) = G(X_i(t); X_{1}(t), X_{2}(t), \ldots, X_{m}(t)) \). Thus different REAs can be unified by a similar framework.

Let the global optimization problem be of the following form

\[
\min_{x \in \Omega} f(x),
\]

where \( \Omega = \{ x \in R^{n | l \leq x \leq u}, l = (l_1, l_2, \ldots, l_n)^T, u = (u_1, u_2, \ldots, u_n)^T \} \). Suppose \( f \) has finite global minimum on \( \Omega \). The C-style pseudocode of REA is as follows.

**procedure REA:**

- **input** evolutionary parameters: population size \( N \), initial bounds of variables \( lb, ub \), set \( t = 0 \); initialize population

\[
\bar{X}(0) = \{ X_1(0), X_2(0), \ldots, X_N(0) \},
\]

where \( X_i(0) = (x_{i,1}, x_{i,2}, \ldots, x_{i,n}) \);

**do**

- **reproduction:** generate a new population \( X'_i(t) \) based on \( \bar{X}(t) \);
- **for every index** \( i \in \{ 1, 2, \ldots, N \} \) in the population,

\[
X_i(t) = G(X_i(t); X_1(t), X_2(t), \ldots, X_m(t));
\]

- **evaluation:** evaluate populations \( \bar{X}(t) \) and \( \bar{X}'(t) \);
- **compute** \( f(X_i(t)), f(X'_i(t)) \) for each individual;
- **selection:** select \( N \) individuals among \( \bar{X}(t) \) and \( \bar{X}'(t) \) to get the next generation \( \bar{X}(t+1) \).

**}** until some stopping criterion is satisfied.

2.2 LDR strategy for REAs

As discussed above, to preserve some information of local elite and keep a balance of global search and fast convergence is necessary for an efficient REA. In this paper, this idea is realized by preserving some (randomly chosen) components of the local best vector (elite individual) in the reproduction process and let the traditional reproduction operators act on the rest components. As can be illustrated in Fig.1 (c) and (c'). The reproduction process with this strategy is called as low dimensional reproduction (LDR) because traditional operators are affective only in low dimensional search space. The LDR preserves some information of local elite. In the view of a new individual, some of its components are adsorbed to the corresponding parts of local best vector \( X_b \).

LDR strategy can be implemented in different ways. Here we provide a most simple way, to add an adsorption process before traditional reproduction process (other variants can be found in [9]). The adsorption process can be described as follows. First, determine the reproduction operator(s) \( G \) on which the LDR strategy will be used if there are several reproduction operators (e.g., crossover and mutation), choose a base point \( X_b \) which is considered as a local best individual and preset a adsorption probability \( P_a \). Then the adsorption is done component-wisely: whether the \( j \)-th component is adsorbed to the \( j \)-th component of the base point depends on a random number \( rand \), which is uniformly drawn from \([0, 1]\). The \( j \)-th component is adsorbed as a passive component if \( rand < P_a \), otherwise, it remains active and will be updated by traditional reproduction operator. A reproduction operator with LDR
strategy can be formulated as

\[
\begin{align*}
    P(X_{i,j}^r(t) = X_{i,j}(t)) &= p_u; \\
    P(X_{i,j}^r(t) = G(X_{i,j}(t)); X_{r_1,j}(t), \\
    &X_{r_2,j}(t), \ldots, X_{r_m,j}(t)) = 1 - p_u.
\end{align*}
\]

where \( b \in \{i, r_1, r_2, \ldots, r_m\} \), \( X_b \) is the base point, \( p_u \) is the adsorption probability.

2.3 Remark of LDR strategy

LDR strategy is a very easy technique for improving the existing REAs. The main difference between a REA with LDR strategy and the original REA is the reproduction process (the way of generating a new individual). In other words, to apply the LDR strategy into a REA, the only thing that one needs to do is to replace formula (1) by formula (2).

On the other hand, some techniques are necessary to get the modified form of REA with LDR strategy. The first and most important thing is the choice of the base point. If we choose a proper base point which could lead to a successful REA with LDR strategy. Improper base point might result in failure. The second key-point is the choice of reproduction operator(s) on which the LDR strategy will be used. There might be several reproduction operators in a REAs, but different REAs have different evolutionary operators, the LDR strategy is applied to their reproduction operator(s). Therefore, we will emphasize on their evolutionary operators below. Meanwhile, full description of REAs with LDR strategy is omitted here for conciseness, and it can be found in [9].

3 Some REAs with LDR strategy

In this section, LDR strategy will be used to improve the performance of some REAs, including real-coded genetic algorithm (RGA), evolutionary programming (GP), differential evolution (DE), and particle swarm optimization (PSO). As above mentioned, REAs have similar framework but different REAs have different evolutionary operators, and the LDR strategy is applied to their reproduction operator(s). Therefore, we will emphasize on their evolutionary operators below. Meanwhile, full description of REAs with LDR strategy is omitted here for conciseness, and it can be found in [9].

3.1 Strategy for RGA

In late 1980s, several real-coded genetic algorithms (REAs) were proposed [2–6]. REAs are more suitable for function (of continuous variable) optimization than binary-coded GA[10]. Various real-coded evolutionary operators have been proposed[7]: flat crossover, simple crossover, arithmetic crossover, blend crossover, BLX-\( \alpha \), discrete crossover; random mutation, non-uniform mutation, creep mutation, Mühlenbein mutation; fitness proportional selection, ranking selection, truncation selection, tournament selection, etc. The main evolutionary operators of RGA used in this paper are as follows:

1. arithmetical crossover: \( X_i'(t) = \lambda \cdot X_i(t) + (1 - \lambda) \cdot X_{r_1}(t); \)
2. non-uniform mutation: \( X_{i,j}''(t) = X_{i,j}^r(t) + (b_j - X_{i,j}^r(t)) \cdot \mu^{(1-(\lambda^2))}, \) where \( b_j \in \{l_j, u_j\}; \)
3. binary tournament selection: \( X_i(t + 1) = \begin{cases} X_{i,j}^r(t), & \text{if } f(X_{i,j}^r) < f(X_{i,j}^l) \\
X_{i,j}^l(t), & \text{else} \end{cases} \)

where \( \lambda \) and \( \mu \) are random numbers drawn uniformly from \([0, 1]\), \( r_1, r_2 \) are random integers in \([1, 2, \ldots, N]\).

Among them, the arithmetical crossover plays an important role of reproduction, so that it is chosen for applying the LDR strategy. Meanwhile, the current individual \( X_i(t) \) is chosen as the base point, that is, \( X_b(t) = X_i(t). \) The arithmetical crossover with LDR strategy can be illustrated in Fig.1 (c) and (c') and described by the following C-style pseudocode.

\[
\begin{align*}
\text{for } j=0; j<n; j++ \{} \\
\text{if rand} < P_u \text{ /* LDR strategy */ } \\
X_i'(j) &= X_i(j) \\
\text{else} \\
X_i'(j) &= X_i(j) + N(0, \sigma^2) \\
\text{\}}
\end{align*}
\]

In this case, generating a normal distributed random number \( N(0, \sigma^2) \) is relative computational demanding.

It is obvious that the LDR strategy itself is not an algorithm. To get an applicable algorithm, a traditional evolutionary algorithm (for which to use LDR strategy) is required.
3.2 Strategy for EP

Initial form of evolutionary programming (EP) was developed by L.J. Fogel et al. in 1966\cite{4} in the study of artificial intelligence, and the standard evolutionary programming procedure for real-valued optimization problems is proposed by D.B. Fogel in 1991\cite{5}. Several modified form of EPs have been proposed, such as adaptive EP\cite{1}, FEPE\cite{12}, etc. The EP investigated in this paper can be outlined as follows.

It perturbs each individual $X_i$ in current population $\hat{X}(t)$ by adding a Gaussian random vector (normal mutation): $X_{i,j}'(t) = X_{i,j}(t) + N(0, \sigma^2_i)$, $i = 1, 2, \cdots, n$, where $N(\mu, \sigma^2)$ represents a Gaussian random variable with mean $\mu$ and variance $\sigma^2$, and $\sigma^2_i$ will be updated from generation to generation by formula $\sigma^2_i(t) = \sigma^2_i(t) + \hat{\mu} \cdot N(0.1) + \tau^2 \cdot N(0, 1)$. After that, the parent population $\hat{X}(t)$ and the perturbed population $X_{i,j}'(t)$ are mixed together, each individual $X_{i,j}(t)$ gets a fitness score by tournament among the mixed population (of $2N$ individuals). The first $N$ individuals with higher scores are survived into the next generation $\hat{X}(t + 1)$, and $\hat{X}(t + 1)$ becomes current population. The above evolutionary process is repeated until some stopping criterion is satisfied.

In the above evolutionary process, normal mutation plays an role of reproduction, so it is chosen for applying the LDR strategy. Meanwhile, the current individual $X_i(t)$ is also chosen as the base point. The normal mutation with LDR strategy can be described by the following C-style pseudocode.

```c
for(i=0; i<n; i++)
  if(rand<P_a)
    X_i(t) = X_i(t)
  else
    X_i(t) = X_i(t) + N(0, \sigma_{i,j})
```

3.3 Strategy for DE

Differential evolution (DE) is a new real-coded EA for global optimization proposed by Storn and Price in 1997\cite{12}. For each individual $X_i(t)$ (a point in $R^n$) in current population set, DE reproduces a trial point by mutation and crossover. In the mutation phase DE randomly selects three distinct points $X_{r1}(t), X_{r2}(t), X_{r3}(t)$ from the current population set $\hat{X}(t)$. These points should be different from $X_i(t)$. The mutated point $X_m(t)$ is a perturbation of any of three point along with the differential variation of the other two: $X_m(t) = X_{r1}(t) + F \cdot (X_{r2}(t) - X_{r3}(t))$, where $F$ is a scaling factor. The trial point is then generated by crossover:

$$x_{trial,j}(t) = \begin{cases} x_{m,j}(t), & \text{if } \text{rand}([0,1]) < C_R \text{ or } j = j_{\text{rand}}; \\ x_{i,j}(t), & \text{otherwise.} \end{cases}$$

The trial point replaces $X_i(t)$ from the population only if it is not worse than it.

Differential mutation and crossover play a role of reproduction, while only the mutation process is chosen for applying LDR strategy. Meanwhile, $X_i(t)$ is chosen as the base point. The differential mutation with LDR strategy can be described by the following C-style pseudocode.

```c
for(j=0; j<n; j++)
  if(rand<P_a)
    X_{m,j}(t) = X_{r1,j}(t)
  else
    X_{m,j}(t) = X_{r1,j}(t) + F \cdot (X_{r2,j}(t) - X_{r3,j}(t))
```

3.4 Strategy for PSO

Particle swarm optimization (PSO) is a new population based evolutionary algorithm inspired by birds’ behavior\cite{8}, in which a new individual is generated by particle movement: $X_i(t) = X_i(t) + v_i(t)$, where $v_i(t) = w \cdot v_i(t-1) + c_1 \cdot r_1 (X_{pbest}(t) - X_i(t)) + c_2 \cdot r_2 (X_{gbest}(t) - X_i(t))$, and $w, c_1, c_2$ are three positive constants, called inertia factor, cognitive learning rate and social learning rate, respectively. Then $X_i(t)$ replaces $X_i(t)$, regardless their objective values. The LDR process is added only before the particle movement and $X_{pbest}$ is chosen as the base point. C-style pseudocode for the particle movement with LDR strategy is listed as follows.

```c
for(j=0; j<n; j++)
  if(rand<P_a)
    X_{i,j}(t+1) = X_{pbest,j}(t)
  else
    X_{i,j}(t+1) = X_{i,j}(t) + v_{i,j}(t)
```

4 Effect of LDR strategy

Our numerical results are based on a benchmark suite of 10 dimension-adjustable functions (that is, $n$ is only an adjustable parameter of the functions) with box constraints from [11], which are frequently used for testing the performance of evolutionary algorithms. The control parameters of REAs are set according to their references, and the control parameter of LDR strategy $P_a$ is found empirically. We do not claim these values to be optimal for any problem in general but they would be good values to be chosen. The stopping criterion is as follows. The algorithm will stop if any one of the following three conditions is satisfied:

1. the global minimum is attained, i.e., $f_{\text{GlobBest}}(t) - f^* < \varepsilon$, where $\varepsilon = 10^{-6}$, and $f^*$ is its known global minimum;
2. the population is matured, i.e., $f_{\text{PopWorst}}(t) < \delta$, where $\delta = 10^{-4}$;
3. the maximum number of function evaluations $E$ is reached, where $E=100n^5$. 


To reduce the influence of randomness, 100 replications are done for each problem and each algorithm, and the results presented in this section are all averaged values. A run is said to be successful iff it satisfies condition (1) and stops, otherwise we say the algorithm is failed in this run. The percentage of success \( (p_s) \) (the number of successful runs over the number of total runs) and the average number of function evaluations for which the global minima are found \( (nfe) \) (the number of function evaluations over the number of successful runs) are recorded to show the effect of LDR strategy to each algorithm.

It should be pointed out that we cite these REAs only to show the effect of our strategy, and we have no intention to compare their performance. In fact, each algorithm has its own strongpoint, and it is unfair to compare them using the test problems cited in this paper. With this in mind, the following results are given.

The comparisons of four REAs with and without LDR strategy are listed in table 1−4. It is obvious that REAs with LDR strategy are much more superior to their original versions in terms of \( nfe \) and \( p_s \). For RGA, the average reduced \( nfe \) by LDR strategy is \( (935692−372914)/935692 = 60.15\% \) and the average increased \( p_s \) by LDR strategy is 92.0% − 83.6% = 8.4%. For EP, DE, and PSO, the average reduced \( nfe \) are 53.61%, 30.24% and 66.55%, respectively, and the average increased \( p_s \) are 8.9%, 9.9% and 4.3%, respectively.

### Table 1. Comparison of RGA with and without LDR strategy

<table>
<thead>
<tr>
<th>P</th>
<th>n</th>
<th>RGA nfe</th>
<th>RGA ps</th>
<th>RGA with LDR nfe</th>
<th>RGA with LDR ps</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>10</td>
<td>47294</td>
<td>100</td>
<td>21283</td>
<td>100</td>
</tr>
<tr>
<td>CM</td>
<td>20</td>
<td>347259</td>
<td>100</td>
<td>46517</td>
<td>86</td>
</tr>
<tr>
<td>GW</td>
<td>20</td>
<td>64937</td>
<td>100</td>
<td>31882</td>
<td>100</td>
</tr>
<tr>
<td>LM1</td>
<td>10</td>
<td>8633</td>
<td>100</td>
<td>2276</td>
<td>100</td>
</tr>
<tr>
<td>LM2</td>
<td>10</td>
<td>13740</td>
<td>100</td>
<td>9737</td>
<td>100</td>
</tr>
<tr>
<td>NF3</td>
<td>5</td>
<td>44351</td>
<td>56</td>
<td>67939</td>
<td>73</td>
</tr>
<tr>
<td>RB</td>
<td>10</td>
<td>334398</td>
<td>42</td>
<td>145257</td>
<td>68</td>
</tr>
<tr>
<td>RG</td>
<td>5</td>
<td>6285</td>
<td>100</td>
<td>6177</td>
<td>100</td>
</tr>
<tr>
<td>SWF</td>
<td>10</td>
<td>40046</td>
<td>57</td>
<td>32492</td>
<td>93</td>
</tr>
<tr>
<td>SIN</td>
<td>10</td>
<td>28749</td>
<td>81</td>
<td>9354</td>
<td>100</td>
</tr>
<tr>
<td>average</td>
<td></td>
<td>93569.2</td>
<td>83.6</td>
<td>37291.4</td>
<td>92.0</td>
</tr>
</tbody>
</table>

### Table 2. Comparison of EP with and without LDR strategy

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>15</td>
<td>87563</td>
<td>100</td>
<td>37884</td>
<td>100</td>
</tr>
<tr>
<td>CM</td>
<td>15</td>
<td>22130</td>
<td>100</td>
<td>8916</td>
<td>100</td>
</tr>
<tr>
<td>GW</td>
<td>20</td>
<td>76849</td>
<td>100</td>
<td>11216</td>
<td>100</td>
</tr>
<tr>
<td>LM1</td>
<td>10</td>
<td>55840</td>
<td>66</td>
<td>49466</td>
<td>100</td>
</tr>
<tr>
<td>LM2</td>
<td>10</td>
<td>8831</td>
<td>100</td>
<td>6286</td>
<td>100</td>
</tr>
<tr>
<td>NF3</td>
<td>20</td>
<td>694733</td>
<td>71</td>
<td>30587</td>
<td>100</td>
</tr>
<tr>
<td>RB</td>
<td>10</td>
<td>147288</td>
<td>6</td>
<td>380434</td>
<td>32</td>
</tr>
<tr>
<td>RG</td>
<td>10</td>
<td>86507</td>
<td>100</td>
<td>14131</td>
<td>100</td>
</tr>
<tr>
<td>SWF</td>
<td>10</td>
<td>16471</td>
<td>100</td>
<td>8020</td>
<td>100</td>
</tr>
<tr>
<td>SIN</td>
<td>10</td>
<td>7359</td>
<td>100</td>
<td>11423</td>
<td>100</td>
</tr>
<tr>
<td>average</td>
<td></td>
<td>120357.1</td>
<td>84.3</td>
<td>55836.3</td>
<td>93.2</td>
</tr>
</tbody>
</table>

### Table 3. Comparison of DE with and without LDR strategy

<table>
<thead>
<tr>
<th>P</th>
<th>n</th>
<th>DE nfe</th>
<th>DE ps</th>
<th>DE with LDR nfe</th>
<th>DE with LDR ps</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>15</td>
<td>42234</td>
<td>100</td>
<td>17581</td>
<td>100</td>
</tr>
<tr>
<td>CM</td>
<td>25</td>
<td>13624</td>
<td>100</td>
<td>10467</td>
<td>100</td>
</tr>
<tr>
<td>GW</td>
<td>15</td>
<td>16243</td>
<td>100</td>
<td>12909</td>
<td>100</td>
</tr>
<tr>
<td>LM1</td>
<td>30</td>
<td>30172</td>
<td>100</td>
<td>19393</td>
<td>100</td>
</tr>
<tr>
<td>LM2</td>
<td>20</td>
<td>6286</td>
<td>100</td>
<td>4705</td>
<td>100</td>
</tr>
<tr>
<td>NF3</td>
<td>20</td>
<td>745813</td>
<td>83</td>
<td>324402</td>
<td>97</td>
</tr>
<tr>
<td>RB</td>
<td>10</td>
<td>257607</td>
<td>13</td>
<td>374870</td>
<td>98</td>
</tr>
<tr>
<td>RG</td>
<td>20</td>
<td>32933</td>
<td>100</td>
<td>30752</td>
<td>100</td>
</tr>
<tr>
<td>SWF</td>
<td>15</td>
<td>27141</td>
<td>100</td>
<td>13243</td>
<td>100</td>
</tr>
<tr>
<td>SIN</td>
<td>20</td>
<td>35141</td>
<td>100</td>
<td>21212</td>
<td>100</td>
</tr>
<tr>
<td>average</td>
<td></td>
<td>118919.4</td>
<td>89.6</td>
<td>82953.4</td>
<td>99.5</td>
</tr>
</tbody>
</table>

### Table 4. Comparison of PSO with and without LDR strategy

<table>
<thead>
<tr>
<th>P</th>
<th>n</th>
<th>PSO nfe</th>
<th>PSO ps</th>
<th>PSO with LDR nfe</th>
<th>PSO with LDR ps</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACK</td>
<td>10</td>
<td>494261</td>
<td>100</td>
<td>28995</td>
<td>100</td>
</tr>
<tr>
<td>CM</td>
<td>10</td>
<td>73150</td>
<td>100</td>
<td>6536</td>
<td>100</td>
</tr>
<tr>
<td>GW</td>
<td>15</td>
<td>52326</td>
<td>100</td>
<td>47001</td>
<td>100</td>
</tr>
<tr>
<td>LM1</td>
<td>20</td>
<td>116774</td>
<td>100</td>
<td>36879</td>
<td>100</td>
</tr>
<tr>
<td>LM2</td>
<td>10</td>
<td>44386</td>
<td>100</td>
<td>17623</td>
<td>100</td>
</tr>
<tr>
<td>NF3</td>
<td>5</td>
<td>7470</td>
<td>69</td>
<td>8919</td>
<td>82</td>
</tr>
<tr>
<td>RB</td>
<td>10</td>
<td>273361</td>
<td>48</td>
<td>206276</td>
<td>54</td>
</tr>
<tr>
<td>RG</td>
<td>15</td>
<td>136473</td>
<td>100</td>
<td>21634</td>
<td>100</td>
</tr>
<tr>
<td>SWF</td>
<td>5</td>
<td>7329</td>
<td>100</td>
<td>6378</td>
<td>100</td>
</tr>
<tr>
<td>SIN</td>
<td>15</td>
<td>65796</td>
<td>76</td>
<td>45023</td>
<td>100</td>
</tr>
<tr>
<td>average</td>
<td></td>
<td>127132.6</td>
<td>89.3</td>
<td>42526.4</td>
<td>93.6</td>
</tr>
</tbody>
</table>

### 5 Choice of control parameter

REAs with LDR strategy involve two kinds of control parameters: parameters of EAs and parameter of LDR strategy. Note that the LDR strategy requires only one control
parameter $P_a$ and thus it is not difficult to choose in order to obtain good result. According to our experience, a reasonable choice for $P_a$ is between 0.1 and 0.9, and a first good choice for $P_a$ is 0.8. It is interesting to point out that one can switch off the LDR strategy by setting $P_a$ less than or equal to 0.0. Easy to control is another asset of the proposed strategy.

It seems reasonable to incorporate LDR strategy after the parameters of each EA to each problem are optimized and then to adjust the parameter of the strategy. Earlier experiments have been done in this way and it shows that the effect of the strategy is not very sensitive to the parameters of EAs.

6 Conclusions

The LDR strategy have been proposed in this paper, it could help the search points to escape from the hyperplane where the parents individuals lies, as well as keep them from getting too much decentralized and search mainly along a series of orthogonal directions (coordinate).

The LDR strategy is rewarding. Traditional evolutionary algorithms face the dilemma of increasing the probability of finding global optimum solution and accelerating its convergence speed. The LDR strategy can improve the percentage of success $ps$ but also accelerate the convergence speed, as is shown by several incorporated evolutionary algorithms.

The LDR strategy is a technique of computational saving, as already described in subsection 2.3.

The LDR strategy is easy to control and to be switched off/on. It involves only one control parameter, the adsorption probability $P_a$, so it is easy to adjust the intensity of adsorption and one can switch off the strategy by setting the control parameter $P_a \leq 0.0$.

Only four evolutionary algorithms have been taken as illustrative examples to demonstrate the effect of the LDR strategy, but the idea of the strategy is universal and it may be used other stochastic algorithms, especially for population-based heuristics.

Although the strategy seems easy, its working mechanism is still not clear yet. That is, we can not tell why the strategy works well. In fact, the strategy is purely heuristic with no theoretical proof. Further research is underway in developing theoretical results and in developing more efficient method to apply the LDR strategy into special evolutionary algorithms.

Acknowledgement

This work was partially supported by a Grant-in-Aid for 21st Century COE “Frontiers of Computational Science” of Japan, and by the National Nature Science Foundation of China (10671029). Special thanks are due to professor M. Montaz Ali for providing the source code of numerical test problems. The authors would also like to thank the anonymous reviewers for their precious comments on the earlier manuscript of this work.

References