Neurocomputing 149 (2015) 1038–1047

Contents lists available at ScienceDirect

Neurocomputing

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

A binary differential evolution algorithm learning from explored solutions

Yu Chen [a,b,*], Weicheng Xie [c], Xiufen Zou [b]

[a] School of Science, Wuhan University of Technology, Wuhan 430070, China
[b] School of Mathematics and Statistics, Wuhan University, Wuhan 430072, China
[c] Shenzhen Key Laboratory of Spatial Smart Sensing and Services, School of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518052, China

ARTICLE INFO

Article history:
Received 3 October 2013
Received in revised form
23 May 2014
Accepted 23 July 2014
Communicated by Swagatam Das
Available online 2 August 2014

Keywords:
Binary differential evolution algorithm
Convergence in probability
Renewal metric
Refinement metric

ABSTRACT

Although real-coded differential evolution (DE) algorithms can perform well on continuous optimization problems (CoOPs), designing an efficient binary-coded DE algorithm is still a challenging task. Inspired by the learning mechanism in particle swarm optimization (PSO) algorithms, we propose a binary learning differential evolution (BLDE) algorithm that can efficiently locate the global optimal solutions by learning from the last population. Then, we theoretically prove the global convergence of BLDE, and compare it with some existing binary-coded evolutionary algorithms (EAs) via numerical experiments. Numerical results show that BLDE is competitive with the compared EAs. Further study is performed via the change curves of a renewal metric and a refinement metric to investigate why BLDE cannot outperform some compared EAs for several selected benchmark problems. Finally, we employ BLDE in solving the unit commitment problem (UCP) in power systems to show its applicability to practical problems.

1. Introduction

1.1. Background

Differential evolution (DE) [26], a competitive evolutionary algorithm emerging more than a decade ago, has been widely utilized in the science and engineering fields [24,4]. The simple and straightforward evolving mechanisms of DE endow it with the powerful capability to solve continuous optimization problems (CoOPs), but hamper its applications to discrete optimization problems (DOPs).

To take full advantage of the superiority of mutations in classic DE algorithms, Pampará and Engelbrecht [21] introduced a trigonometric generating function to transform the real-coded individuals of DE into binary strings, and proposed an angle modulated differential evolution (AMDE) algorithm for DOPs. Compared with the binary differential evolution (BDE) algorithms that directly manipulate binary strings, AMDE was much slower, but outperformed BDE algorithms with respect to accuracy of the obtained solutions [7]. Meanwhile, Gong and Tuson proposed a binary DE algorithm by forma analysis [9], but it cannot perform well on binary constraint satisfaction problems due to its weak exploration ability [31]. Attempting to simulate the operation mode of the continuous DE mutation, Kashan et al. [14] designed a dissimilarity based differential evolution (DisDE) algorithm incorporating a measure of dissimilarity in mutation. Numerical results show that DisDE is competitive with some existing binary-coded evolutionary algorithms (EAs).

In addition, the performances of BDE algorithms can also be improved by incorporating recombination operators of other EAs. Hota and Pat [12] proposed an adaptive quantum-inspired differential evolution algorithm (AQDE) applying quantum computing techniques, while He and Han [10] introduced the negative selection in artificial immune systems to obtain an artificial immune system based differential evolution (AIS-DE) algorithm. With respect to the fact that the logical operations introduced in AIS-DE tends to produce "1" bits with increasing probability, Wu and Tseng [29] proposed a modified binary differential evolution strategy to improve the performance of BDE algorithms on topology optimization of structures.

1.2. Motivation and contribution

Existing research efforts tried to incorporate the recombination strategies of various EAs to obtain efficient BDEs for DOPs, however, there are still some points to be improved:

- AMDE [21] has to transform real values into binary strings, which leads to an explosion of the computational cost for

* Corresponding author at: School of Science, Wuhan University of Technology, Wuhan 430070, China. Tel.: +86 27 87108027.
E-mail address: chymath@gmail.com (Y. Chen).

http://dx.doi.org/10.1016/j.neucom.2014.07.030
0925-2312/

function evaluations. Meanwhile, the mathematical properties of the transformation function can also influence its performances on various DOPs;

- BDE algorithms directly manipulating bit-strings, such as binDE [9], AIS-DE [10] and MBDE [29]cannot effectively imitate the mutation mechanism of continuous DE algorithms. Thus, they cannot perform well on high-dimensional DOPs due to their weak exploration abilities;
- DisDE [14], which incorporates a dissimilarity metric in the mutation operator, has to solve a minimization problem during the mutation process. As a consequence, the computation complexity of DisDE is considerably high.

Generally, it is a challenging task to design an efficient BDE algorithm perfectly addressing the aforementioned points. Recently, variants of the particle swarm optimization (PSO) algorithm [15] have been successfully utilized in real applications [6,1,23,2,17]. Although DE algorithms perform better than PSO algorithms in some real world applications [28,25,22], it is still promising to improve DE by incorporating PSO in the evolutionary process [3,18,19]. Considering that the learning mechanism of PSO can accelerate the convergence of populations, we propose a hybrid binary-coded evolutionary algorithm learning from the last population, named as the binary learning differential evolution (BLDE) algorithm. In BLDE, the searching process of population is guided by the renewed information of individuals, the dissimilarity between individuals and the best explored solution in the population. Using these, BLDE can perform well on DOPs.

The remainder of the paper is structured as follows. Section 2 presents a description of BLDE, and its global convergence is theoretically proved in Section 3. Then, in Section 4, BLDE is compared with some existing algorithms using numerical results. To test the performance of BLDE on real-life problems, we employ it to solve the unit commitment problem (UCP) in Section 5. Finally, discussions and conclusions are presented in Section 6.

2. The binary learning differential evolution algorithm

2.1. Framework of the binary learning differential evolution algorithm

Algorithm 1. The binary learning differential evolution (BLDE) algorithm.

Randomly generate two populations $\mathbf{X}^{(1)}$ and $\mathbf{A}^{(1)}$ of μ individuals; Set $t:=1$;
whilethe stop criterion is no satisfied **do**
  Let $\mathbf{x}_{gb}=(x_{gb,1},\dots,x_{gb,n}) \triangleq \arg\max_{\mathbf{x}\in\mathbf{X}^{(t)}}\{f(\mathbf{x})\}$;
  for all $\mathbf{w}\in\mathbf{X}^{(t)}$ **do**
    Randomly select $\mathbf{x}=(x_1,\dots,x_n)$ and $\mathbf{y}=(y_1,\dots,y_n)$ from $\mathbf{X}^{(t)}$, as well as $\mathbf{z}=(z_1,\dots,z_n)$ from $\mathbf{A}^{(t)}$;
    $\mathbf{tx}=(tx_1,\dots,tx_n)\triangleq \arg\max\{f(\mathbf{y}),f(\mathbf{z})\}$;
    for $j=1,2,\dots,n$ **do**
      if $y_j=z_j$ **then**
        if $x_{gb,j}\neq x_j$ **then**
          $tx_j=x_{gb,j}$;
        else
          if $rand(0,1)\le p$ **then**

$$tx_j=\begin{cases}0 & \text{with probability } \dfrac{1}{2};\\ 1 & \text{otherwise.}\end{cases}$$

          end if
        end if
      end if
    end for
    if $f(\mathbf{tx})\ge f(\mathbf{w})$ **then**
      $\mathbf{w}=\mathbf{tx}$;
    end if
  end for
  $t:=t+1$;
  $\mathbf{A}^{(t)}=\mathbf{X}^{(t-1)}$;
end while

For a binary optimization problem (BOP)[1]

$$\max_{\mathbf{x}\in S}\ f(\mathbf{x})=f(x_1,\dots,x_n),\quad S\subset\{0,1\}^n, \tag{1}$$

the BLDE algorithm illustrated by Algorithm 1 possesses two collections of μ solutions, the population $\mathbf{X}^{(t)}$ and the archive $\mathbf{A}^{(t)}$. At the first generation, the population $\mathbf{X}^{(1)}$ and the archive $\mathbf{A}^{(1)}$ are generated randomly. Then, repeat the following operations until the stopping criterion is satisfied.

For each individual $\mathbf{w}\in\mathbf{X}^{(t)}$ a trial solution is generated by three randomly selected individuals $\mathbf{x}$, $\mathbf{y}\in\mathbf{X}^{(t)}$ and $\mathbf{z}\in\mathbf{A}^{(t)}$. At first, initialize the trial individual $\mathbf{tx}=\{tx_1,\dots,tx_n\}$ as the winner of two individuals $\mathbf{y}\in\mathbf{X}^{(t)}$ and $\mathbf{z}\in\mathbf{A}^{(t)}$. $\forall\ j\in\{1,2,\dots,n\}$, if $\mathbf{y}$ and $\mathbf{z}$ coincide on the jth bit, the jth bit of $\mathbf{tx}$ is changed as follows.

- If the jth bit of $\mathbf{x}$ differs from that of $\mathbf{x}_{gb}$, tx_j is set to be $x_{gb,j}$, the jth bit of $\mathbf{x}_{gb}$;
- otherwise, tx_j is randomly mutated with a preset probability p.

Then, replace $\mathbf{w}$ with $\mathbf{tx}$ if $f(\mathbf{tx})\ge f(\mathbf{w})$. After the update of population $\mathbf{X}^{(t)}$ is completed, set $t=t+1$ and $\mathbf{A}^{(t)}=\mathbf{X}^{(t-1)}$.

2.2. The positive functions of the learning scheme

Generally speaking, the trial solution $\mathbf{tx}$ is generated by three randomly selected individuals. Meanwhile, it also incorporates conditional learning strategies in the mutation process.

- By randomly selecting $\mathbf{y}\in\mathbf{X}^{(t)}$, BLDE can learn from any member in the present population. Because the elitism strategy is employed in the BLDE algorithm, BLDE could learn from any *pbest* solution in the population, unlike PSO, where particles can only learn from their own *pbest* individuals.
- By randomly selecting $\mathbf{z}\in\mathbf{A}^{(t)}$, BLDE can learn from any member in the last population. In the early stages of the iteration process, individuals in the population $\mathbf{X}^{(t)}$ are usually different with those in $\mathbf{A}^{(t)}=\mathbf{X}^{(t-1)}$. Combined with the first strategy, this scheme actually enhances the exploration ability of the population and to some extent, accelerates convergence of the population.
- When bits of $\mathbf{y}$ coincide with the corresponding bits of $\mathbf{z}$, trial solutions learn from the *gbest* on the condition that randomly selected $\mathbf{x}\in\mathbf{X}^{(t)}$ differs from $\mathbf{x}_{gb}$ on these bits. This scheme imitates the learning strategy of PSO. The scheme can also prevent the population from being governed by dominating patterns because the increased probability $\mathbf{P}\{x_{gb,j}=x_j\}$ will lead to a random mutation performed on $\mathbf{tx}$, thus preventing duplicates of the dominating patterns in the population.

In PSO algorithms, each particle learns from the *pbest* (the best solution it has obtained so far) and the *gbest* (the best solution the swarm has obtained so far), and particles in the swarm exchange

[1] When a CoOP is considered, the real-value variables can be coded as bit-strings, and consequently, a binary optimization problem is constructed to be solved by binary-coded evolutionary algorithms.

information via the *gbest* solution only. The simple and unconditional learning strategy of PSO usually results in its fast convergence rate, however, it sometimes leads to its premature convergence to local optima. The BLDE algorithm, learning from $\mathbf{X}^{(t)}$ as well as $\mathbf{A}^{(t)}$, can explore the feasible region in a better way, and by conditionally learning from $\mathbf{x}_{gb}$, it will not be attracted by local optimal solutions.

3. Convergence analysis of BLDE

Denote $\mathbf{x}^*$ to be an optimal solution of BOP (1), the global convergence of BLDE can be defined as follows.

Definition 1. Let $\{\mathbf{X}^{(t)},\ t=1,2,\ldots\}$ be the population sequence of BLDE. It is said to converge in probability to the optimal solution $\mathbf{x}^*$ of BOP (1), if it holds that

$$\lim_{t\to\infty}\mathbf{P}\{\mathbf{x}^*\in\mathbf{X}^{(t)}\}=1.$$

To confirm the global convergence of the proposed BLDE algorithm, we first show that any feasible solution can be generated with a positive probability.

Lemma 1. *In two generations, BLDE can generate any feasible solution of BOP (1) with a probability greater than or equal to a positive constant c.*

Proof. Denote $\mathbf{x}^{(t)}(i)=(x_1^{(t)}(i),\ldots,x_n^{(t)}(i))$ and $\mathbf{a}^{(t)}(i)=(a_1^{(t)}(i),\ldots,a_n^{(t)}(i))$ to be the ith individuals of $\mathbf{X}^{(t)}$ and $\mathbf{A}^{(t)}$, respectively. Let $\mathbf{tx}^{(t)}(i)=(tx_1^{(t)}(i),\ldots,tx_n^{(t)}(i))$ be the ith trial individual generated at the tth generation. There are two different cases to be investigated.

1. If $\mathbf{X}^{(t)}$ and $\mathbf{A}^{(t)}$ include at least one common individual, the probability $\mathbf{P}\{\mathbf{y}=\mathbf{z}\}$ is greater than or equal to $1/\mu^2$, where $\mathbf{y}\in\mathbf{X}^{(t)}$ and $\mathbf{z}\in\mathbf{A}^{(t)}$ are selected randomly from $\mathbf{X}^{(t)}$ and $\mathbf{A}^{(t)}$, respectively. Then, the random mutation illustrated by Lines 12–14 of Algorithm 1 will be activated with probability $1/\mu$, which is the minimum probability of selecting $\mathbf{x}$ to be $\mathbf{x}_{gb}^{(t)}$, the best individual in the present population $\mathbf{X}^{(t)}$. For this case, both $\mathbf{P}\{tx_j=0\}$ and $\mathbf{P}\{tx_j=1\}$ are greater than or equal to $p/2\mu^3$. Then, any feasible solution can be generated with a positive probability greater than or equal to $(p/2\mu^3)^n$.
2. If all individuals in $\mathbf{X}^{(t)}$ differ from those in $\mathbf{A}^{(t)}$, two different solutions $\mathbf{y}\in\mathbf{X}^{(t)}$ and $\mathbf{z}\in\mathbf{A}^{(t)}$ are located at the same index i_0 with probability

   $$\mathbf{P}\left\{\mathbf{y}=\mathbf{x}^{(t)}(i_0),\mathbf{z}=\mathbf{a}^{(t)}(i_0)\right\}=\frac{1}{\mu^2}.$$

   Because $\mathbf{y}\neq\mathbf{z}$, $I_1=\{j;y_j\neq z_j\}$ is not empty. Moreover, the elitism update strategy ensures that the trial individual $\mathbf{tx}^{(t)}(i_0)$ is initialized to be $\mathbf{tx}^{(t)}(i_0)=\mathbf{y}$. Then,

   $$tx_j^{(t)}(i_0)=y_j=x_j^{(t)}(i_0),\quad \forall\ j\in I_1,$$

   and $\forall\ j\notin I_1$, $\mathbf{tx}^{(t)}(i_0)$ will remain unchanged with a probability greater than $(1-p)/\mu$, the probability of selecting $\mathbf{x}=\mathbf{x}_{gb}$ and not activating the mutation illustrated by Lines 12–14 of Algorithm 1. That is to say, the probability of generating a trial individual $\mathbf{tx}^{(t)}(i_0)=\mathbf{y}=\mathbf{x}^{(t)}(i_0)$ is greater than or equal to $(1-p)/\mu^3$.

   For this case, the i_0th individual of the population will remain unchanged at the tth generation, and at the next generation (generation $t+1$), $\mathbf{x}^{(t+1)}(i_0)$ will coincide with $\mathbf{a}^{(t+1)}(i_0)$. Then, it comes to the first case, and consequently, the trial individual $\mathbf{tx}^{(t+1)}(i)$ can reach any feasible solution with a positive probability greater than or equal to $(p/2\mu^3)^n$. For this case, any feasible solution can be generated with a probability greater than $(1-p)/\mu^3(p/2\mu^3)^n$.

In conclusion, in two generations the trial individual $\mathbf{tx}$ will reach any feasible solution with a probability greater than or equal to a positive constant c, where $c=(1-p)/\mu^3(p/2\mu^3)^n$. □

Theorem 1. *BLDE converges in probability to the optimal solution* $\mathbf{x}^*$ *of OP (1).*

Proof. Lemma 1 shows that there exists a positive number $c>0$ such that

$$\mathbf{P}\{\mathbf{x}^*\in\mathbf{X}^{(t+2)}|\mathbf{x}^*\notin\mathbf{X}^{(t)}\}\geq c,\quad \forall t\geq 1.$$

Denoting

$$P=\mathbf{P}\{\mathbf{x}^*\in\mathbf{X}^{(t+2)}|\mathbf{x}^*\notin\mathbf{X}^{(t)}\},$$

we know that

$$\mathbf{P}\{\mathbf{x}^*\notin\mathbf{X}^{(t+2)}|\mathbf{x}^*\notin\mathbf{X}^{(t)}\}=1-P.$$

Thus,

$$\mathbf{P}\{\mathbf{x}^*\in X^{(t)}\}=1-\mathbf{P}\{\mathbf{x}^*\notin X^{(t)}\}=1-\mathbf{P}\{\mathbf{x}^*\notin\mathbf{X}^{(t)}|\mathbf{x}^*\notin\mathbf{X}^{(t-2)}\}.$$

If t is even,

$$\begin{aligned}\lim_{t\to\infty}\mathbf{P}\{\mathbf{x}^*\in X^{(t)}\}&=1-\lim_{t\to\infty}\mathbf{P}\{\mathbf{x}^*\notin X^{(t)}\}\\&=1-\lim_{t\to\infty}(1-p)^{t/2}\mathbf{P}\{\mathbf{x}^*\notin\mathbf{X}^{(0)}\}\\&=1;\end{aligned}$$

otherwise,

$$\begin{aligned}\lim_{t\to\infty}\mathbf{P}\{\mathbf{x}^*\in X^{(t)}\}&=1-\lim_{t\to\infty}\mathbf{P}\{\mathbf{x}^*\notin X^{(t)}\}\\&=1-\lim_{t\to\infty}(1-p)^{(t-1)/2}\mathbf{P}\{\mathbf{x}^*\notin\mathbf{X}^{(1)}\}\\&=1.\end{aligned}$$

In conclusion, BLDE converges in probability to the optimal solution $\mathbf{x}^*$ of BOP (1). □

4. Numerical experiments

Although Theorem 1 validates the global convergence of the BLDE algorithm, its convergence characteristics have not been investigated. In this section, we try to show its competitiveness through numerical experiments.

4.1. Benchmark problems

Table 1 illustrates the selected benchmark problems, the properties and settings of which are listed in Table 2. As for the continuous problems P_3-P_7, all real variables are coded by bit-strings. For the multiple knapsack problem (MKP) P_8, we test BLDE via five test instances characterized by data files "weing6.dat, sent02.dat, weish14.dat, weish22.dat and weish30.dat" [30], termed as P_{8-1}, P_{8-2}, P_{8-3}, P_{8-4} and P_{8-5}, respectively. When a candidate solution is evaluated, it is penalized by $PT(\mathbf{x})=1+\max_j p_i/\min_{i,j}w_{i,j}\cdot\max_i\{\max_j(w_{i,j}x_j-W_i),0\}$ [27].

4.2. Parameter settings

For numerical comparisons, BLDE is compared with the angle modulated particle swarm optimization (AMPSO) [20], the angle modulated differential evolution (AMDE) [21], the dissimilarity artificial bee colony (DisABC) algorithm [13], the binary particle swarm optimization (BPSO) algorithm [16], the binary differential evolution (binDE) [9] algorithm and the self-adaptive quantum-inspired differential evolution (AQDE) algorithm [12]. As is suggested

Table 1
Descriptions of the selected benchmark problems.

Problems	Descriptions.
P_1:	$\max f_1(\mathbf{x}) = \sum_{i=1}^{n} \prod_{j=1}^{i} x_j,\ x_j \in \{0,1\},\ j=1,\ldots,n.$
P_2:	*Long Path Problem* : *Root2path* [11]
P_3:	$\max f_3(\mathbf{x}) = -\max_{i=1,\ldots,m} \lvert x_i \rvert,\ x_i \in [-10,10], i=1,\ldots,D.$
P_4:	$\max f_4(\mathbf{x}) = -\frac{1}{4000}\sum_{i=1}^{D}(x_i-100)^2 + \prod_{i=1}^{D} \cos\left(\frac{x_i-100}{\sqrt{i}}\right) - 1,\ x_i \in [-300,300],\ i=1,\ldots,D.$
P_5:	$\max f_5(\mathbf{x}) = -\sum_{i=1}^{D} i x_i^4 - rand[0,1),\ x_i \in [-1.28,1.28],\ i=1,\ldots,D.$
P_6:	$\max f_6(\mathbf{x}) = -\sum_{i=1}^{D-1}(100(x_{i+1}-x_i^2)^2 + (1-x_i)^2),\quad x_i \in [-2.048,2.048],\ i=1,\ldots,D.$
P_7:	$\max f_7(\mathbf{x}) = -20 + 20\exp(-0.2\sqrt{\frac{1}{m}\sum_{i=1}^{m} x_i^2}) + \exp\left(\frac{1}{m}\sum_{i=1}^{m}\cos(2\pi x_i)\right) - e,\ x_i \in [-30,30],\ i=1,\ldots,D.$
P_8 :	$\max f_8(\mathbf{x}) = \max \sum_{j=1}^{n} p_j x_j, s.t.\ \sum_{j=1}^{n} w_{i,j} x_j \le W_i,\ i=1,\ldots,m, x_j \in \{0,1\},\ j=1,\ldots,n.$

Table 2
Properties and settings of the benchmark problems.

Problem	Binary/ Real	Dimension	Bit-length	Constraints	Maximum objective value
P_1	Binary	30	30	–	30
P_2	Binary	29	29	–	49 992
P_3	Real	30	180	–	0
P_4	Real	30	480	–	0
P_5	Real	30	240	–	0
P_6	Real	30	300	–	0
P_7	Real	30	300	–	0
P_{8-1}	Binary	28	28	2	130 623
P_{8-2}	Binary	60	60	30	8722
P_{8-3}	Binary	60	60	5	6954
P_{8-4}	Binary	80	80	5	8947
P_{8-5}	Binary	90	90	5	11 191

by the designers of these algorithms, the parameters of AMPSO, AMDE, DisABC, BPSO, binDE and AQDE are listed in Table 3. Prerun for BLDE shows that when the mutation ability p is less than 0.05, its weak exploration ability leads it prematurely to the local optima of multi-modal problems; while when p is greater than $\min\{0.15, 10/n\}$, it cannot efficiently exploit the local region of global optima. Thus, in this paper we set $p=\max\{0.05, \min\{0.15, 10/n\}\}$ to keep a balance between exploration and exploitation. All compared algorithms are tested with a population of size 50, and the results are compared after $300 \times n$ FEs, except that numerical results are compared after $300 \times n \times m$ function evaluations (FEs) for MKPs, where n is the bitstring length, m is the number of constraints for MKP.

4.3. Numerical comparisons

Implemented by the MATLAB package, the compared algorithms are run on a PC with an INTEL(R) CORE(R) CPU, running at 2.8 GHZ with 4 GB RAM. After 50 independent runs for each problem, the results are compared in Table 4 via the average best fitness (AveFit), the standard deviation of best fitness (StdDev), the success rate (SR) and the expected runtime (RunTime). Taking AveFit and StdDev as the sorting indexes, the overall ranks of the compared algorithms are listed in Table 5.

Numerical results in Table 4 show that BLDE is generally competitive with the compared algorithms for the selected benchmark problems, which is also illustrated by Table 5, where BLDE, on an average, ranks first for the benchmark problems. Additionally, because it contains no time-consuming operations, for most cases, BLDE spends less CPU time for the selected benchmark problems. Considering that AveFit and StdDev are two overall statistical indexes of the numerical results, we also perform a Wilcoxon rank sum test [8] with a significance level of 0.05 to

Table 3
Parameter settings for the tested algorithms.

Algorithm	Parameter settings
AMPSO	$c_1 = 1.496180,\ c_2 = 1.496180,\ \phi = 0.729844,\ V_{max} = 4.0.$
AMDE	$CR = 0.25, F = 1.$
DisABC	$\phi_{max} = 0.9,\ \phi_{min} = 0.5,\ p_s = 0.5,\ N_{local} = 50,\ p_{local} = 0.01.$
BPSO	$C = 2,\ V_{max} = 6.0.$
binDE	$F = 0.8,\ CR = 0.5.$
AQDE	$F = 0.1{*}r_1{*}r_2,\ CR = 0.5 + 0.0375{*}r_3,\ r_1, r_2 \sim U(0,1),\ r_3 \sim N(0,1).$
BLDE	$p = \max(0.05,\ \min(0.15, 10/n)).$

compare performances of the tested algorithms, and the results are listed in Table 6.

The results of Wilcoxon rank sum tests demonstrate that BPSO performs significantly better on P_5 and P_7, the noisy quadric problem and the maximization problem of Ackley's function, respectively. Because BPSO imitates the evolving mechanisms of PSO by simultaneously changing all bits of the individuals, it can quickly converge to the global optimal solutions. However, BLDE sometimes mutates bit by bit; consequently, its evolutionary process is more vulnerable to noise and the multimodal landscapes of benchmark problems. Thus, BPSO also performs better than BLDE on P_5 and P_7. For similar reasons, BPSO outperforms BLDE on P_{8-1}, a low-dimensional MKP.

Meanwhile, binDE obtains better results than BLDE on the low-dimensional MKPs $P_{8-1}-P_{8-3}$, but performs worse than BLDE on the other problems; this is attributed to the fact that the exploitation ability of binDE descends with the expansion of the search space. Consequently, binDE cannot perform well on the high-dimensional problems. Similarly, AQDE, which is specially designed for Knapsack problems, outperforms BLDE only for the low-dimensional MKP P_{8-1} and cannot perform better than BLDE for other selected benchmark problems.

4.4. Further comparison on the exploration and exploitation abilities

To further explore the underlying causes for BLDE performing worse than BPSO, binDE and AQDE on the given test problems, we try to investigate how their exploration and exploitation abilities change during the evolutionary process. Thus, a renewal metric and a refinement metric are defined to respectively quantify the exploration and exploitation abilities.

Definition 2. Denote the population of an EA at the tth generation to be $\mathbf{X}^{(t)}$, which consists of μ n-bit individuals. Let $HammDist(\mathbf{x}, \mathbf{y})$ to be the Hamming distance between two binary vectors $\mathbf{x}$ and $\mathbf{y}$. The *renewal metric* of an EA at the tth generation is defined as

$$\alpha(t) \triangleq \frac{1}{\mu \cdot n} \sum_{i=1}^{\mu} Ham(\mathbf{x}^{(t)}(i), \mathbf{tx}^{(t)}(i)), \tag{2}$$

Table 4
Numerical results of AMPSO, AMDE, DisABC and BLDE on the 12 test problems. The best results for each problem are highlighted by boldface type.

Problem	AMPSO AveFit ± StdDev (SR, Runtime)	AMDE AveFit ± StdDev (SR, Runtime)	DisABC AveFit ± StdDev (SR, Runtime)	BPSO AveFit ± StdDev (SR, Runtime)	binDE AveFit ± StdDev (SR, Runtime)	AQDE AveFit ± StdDev (SR,Runtime)	BLDE AveFit ± StdDev (SR,Runtime)
P_1	**3.00E+01 ± 0.00E+00** (**100**, 3.01E−01)	**3.00E+01 ± 0.00E+00** (**100**, 2.78E−01)	**3.00E+01 ± 0.00E+00** (**100**, 1.60E+01)	**3.00E+01 ± 0.00E+00** (**100**, 2.95E−01)	2.94E+01 ± 3.14E-01 (96, 4.07E−01)	2.34E+01 ± 2.88E+00 (4, 2.44E−01)	**3.00E+01 ± 0.00E+00** (**100**, **2.15E−01**)
P_2	**5.0E+04 ± 0.00E+00** (**100**, 2.34E+02)	5.0E+04 ± 1.54E+02 (**88**, 2.03E+02)	4.53E+04 ± 7.19E+03 (34, 2.92E+02)	3.96E+04 ± 1.65E+04 (66, 2.81E+02)	4.52E+04 ± 8.92E+03 (40, 2.79E+02)	3.46E+04 ± 1.37E+04 (16, 2.96E+02)	5.00E+04 ± 6.09E+01 (96, 3.07E+02)
P_3	−8.92E+00 ± 2.15E+00 (0, **3.45E+02**)	−5.48E+00 ± 3.21E+00 (**2**, 3.47E+02)	−6.88E+00 ± 2.86E−01 (0, 3.75E+02)	−4.88E+00 ± 7.39E−01 (0, 3.53E+02)	−6.34E+00 ± 3.04E−01 (0, 3.53E+02)	−6.55E+00 ± 3.68Ev01 (0, 3.55E+02)	−**3.22E+00 ± 8.74E−01** (0, 3.53E+02)
P_4	−4.55E+01 ± 3.53E+01 (0, **1.03E+03**)	−1.12E+01 ± 1.99E+01 (**48**, 1.04E+03)	−5.70E+01 ± 5.48E+00 (0, 1.21E+03)	−6.18E+00 ± 2.40E+00 (0, 1.06E+03)	−4.07E+01 ± 4.27E+00 (0, 1.04E+03)	−1.57E+01 ± 3.79E+00 (0, 1.05E+03)	−**1.12E+00 ± 1.10E−01** (0, 1.05E+03)
P_5	−1.13E+01 ± 1.15E+01 (**22**, **4.72E+02**)	−1.27E+00 ± 3.67E+00 (**22**, 4.76E+02)	−3.11E+01 ± 5.55E+00 (0, 5.21E+02)	−**1.90E−02 ± 8.20E−03** (10, 4.82E+02)	−2.35E+01 ± 3.91E+00 (0, 4.83E+02)	−2.32E+01 ± 4.37E+00 (0, 4.85E+02)	−5.79E−02 ± 2.24E−02 (0, 4.84E+02)
P_6	−2.94E+03 ± 9.26E+02 (0, **6.37E+02**)	−1.18E+02 ± 3.51E+02 (**8**, 6.41E+02)	−4.23E+03 ± 4.05E+02 (0, 7.00E+02)	−5.54E+02 ± 2.82E+02 (0, 6.49E+02)	−3.58E+03 ± 2.92E+02 (0, 6.48E+02)	−2.02E+03 ± 4.17E+02 (0, 6.51E+02)	−**4.55E+01 ± 9.68E+01** (0, 6.45E+02)
P_7	−7.87E+00 ± 3.29E+00 (0, **6.02E+02**)	−4.57E+00 ± 2.84E+00 (0, 6.08E+02)	−1.10E+01 ± 3.19E−01 (0, 6.72E+02)	−**1.67E+00 ± 5.40E−03** (0, 6.22E+02)	−1.06E+01 ± 2.74E−01 (0, 6.20E+02)	−1.00E+01 ± 6.43E−01 (0, 6.19E+02)	−1.93E+00 ± 3.84E−02 (0, 6.20E+02)
P_{8-1}	1.21E+05 ± 4.61E+03 (0, 7.35E−01)	1.23E+05 ± 2.70E+03 (0, **6.85E−01**)	1.28E+05 ± 1.14E+03 (2, 3.28E+00)	1.29E+05 ± 2.99E+03 (18, 9.82E−01)	**1.30E+05 ± 2.04E+02** (**52**, 1.25E+00)	1.30E+05 ± 2.89E+02 (20, 9.39E−01)	1.28E+05 ± 2.66E+03 (10, 8.97E−01)
P_{8-2}	7.62E+03 ± 4.80E+02 (0, 2.71E+01)	8.02E+03 ± 1.19E+02 (0, **2.61E+01**)	8.49E+03 ± 4.21E+01 (0, 1.20E+02)	8.66E+03 ± 3.56E+01 (0, 3.65E+01)	**8.72E+03 ± 4.45E+00** (**84**, 4.41E+01)	8.70E+03 ± 1.47E+01 (4, 3.43E+01)	8.70E+03 ± 1.62E+01 (4, 3.25E+01)
P_{8-3}	5.30E+03 ± 2.12E+02 (0, 4.29E+00)	5.24E+03 ± 1.83E+02 (0, **4.13E+00**)	6.01E+03 ± 1.19E+01 (0, 1.92E+01)	6.87E+03 ± 7.85E+01 (26, 5.88E+00)	**6.95E+03 ± 0.00E+00** (**100**, 7.16E+00)	6.84E+03 ± 7.11E+01 (2, 5.51E+00)	6.93E+03 ± 3.66E+01 (58, 5.23E+00)
P_{8-4}	6.52E+03 ± 4.14E+02 (0, 6.04E+00)	6.43E+03 ± 2.22E+02 (0, **5.90E+00**)	7.19E+03 ± 1.89E+02 (0, 2.75E+01)	8.81E+03 ± 1.02E+02 (**8**, 8.31E+00)	8.71E+03 ± 1.06E+02 (0, 1.01E+01)	8.70E+03 ± 9.21E+01 (0, 7.73E+00)	**8.87E+03 ± 5.43E+01** (4, 7.28E+00)
P_{8-5}	8.10E+03 ± 5.96E+02 (0, 7.09E+00)	8.37E+03 ± 2.87E+02 (0, **6.91E+00**)	9.33E+03 ± 2.29E+02 (0, 3.28E+01)	1.11E+04 ± 4.40E+01 (2, 9.64E+00)	1.09E+04 ± 7.01E+01 (0, 1.17E+01)	1.10E+04 ± 8.22E+01 (0, 8.87E+00)	**1.12E+04 ± 1.86E+01** (**6**, 8.29E+00)

Table 5
Ranked performances of the compared algorithms for the selected benchmark problems.

Problem	AMPSO	AMDE	DisABC	BPSO	binDE	AQDE	BLDE
p_1	1	1	1	1	6	7	1
p_2	1	3	4	6	5	7	2
p_3	7	3	6	2	4	5	1
p_4	5	2	6	7	4	3	1
p_5	4	3	7	1	6	5	2
p_6	5	2	7	3	6	4	1
p_7	7	3	6	1	5	4	2
p_{8-1}	7	6	5	3	1	2	4
p_{8-2}	7	6	5	4	1	2	3
p_{8-3}	6	7	5	3	1	4	2
p_{8-4}	6	7	5	2	3	4	1
p_{8-5}	7	6	5	2	4	3	1
Average	5.3	4.1	5.2	2.9	3.8	4.2	1.8

Table 6
Wilcoxon rank sum tests of the compared algorithms on the benchmark problems. The notation $+(-)$ means the algorithm for comparison is significantly superior to (inferior to) BLDE with significance level 0.05; $\approx$ means the compared algorithm is not significantly different from x BLDE.

Algorithm	HBPD	Algorithm	HBPD
AMPSO	$+:\varnothing$ $\approx: P_1, P_2$ $-: P_3, P_4, P_5, P_6, P_7, P_{8-1}, P_{8-2}, P_{8-3}, P_{8-4}, P_{8-5}$	AMDE	$+:\varnothing$ $\approx: P_1, P_2, P_4, P_5$ $-: P_3, P_6, P_7, P_{8-1}, P_{8-2}, P_{8-3}, P_{8-4}, P_{8-5}$
DisABC	$+:\varnothing$ $\approx: P_1, P_{8-1},$ $-: P_2, P_3, P_4, P_5, P_6, P_7, P_{8-2}, P_{8-3}, P_{8-4}, P_{8-5}$	BPSO	$+: P_5, P_7$ $\approx: P_1, P_{8-1}$ $-: P_2, P_3, P_4, P_6, P_{8-2}, P_{8-3}, P_{8-4}, P_{8-5}$
binDE	$+: P_{8-1}, P_{8-2}, P_{8-3}$ $\approx: P_1$ $-: P_2, P_3, P_4, P_5, P_6, P_7, P_{8-4}, P_{8-5}$	AQDE	$+: P_{8-1}$ $\approx: P_{8-2}$ $-: P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_{8-3}, P_{8-4}, P_{8-5}$

where $\mathbf{x}^{(t)}(i)$ is the ith individual in $\mathbf{X}^{(t)}$, and $\mathbf{tx}^{(t)}(i)$ is the corresponding candidate solution. The *refinement metric* of an EA at the tth generation is defined as

$$\beta(t) \triangleq \frac{1}{\mu \cdot n} \sum_{i=1}^{\mu} (n - Ham(\mathbf{x}^{(t)}(i), \mathbf{x}_{gb}(t))), \tag{3}$$

where $\mathbf{x}_{gb}(t)$ is the best explored solution before the tth generation.

The Hamming distance between $\mathbf{x}^{(t)}(i)$ and the corresponding trial vector $\mathbf{tx}(i)$ denotes the overall changes that is performed on the bit-string by the variation strategies. Accordingly, the average value over the whole population can indicate the overall changes of the population. Then, $\alpha(t)$ properly reveals the exploration abilities of EAs at generation t. An EA which harbors a big value of $\beta(t)$ can intensely exploit the local region around the best explored solution $\mathbf{x}_{gb}$; therefore, it harbors a powerful exploitation ability.

For the comparison, we illustrate the changing curves of the renewal metric and the refinement metric for BLDE, BPSO, AQDE and binDE in Fig. 1. Fig. 1(a) and (b) shows that when BPSO is employed to solve P_5 and P_7, the renewal metric quickly descends to about zero, and the refinement metric ascends to a high level, which demonstrates that the population of PSO quickly converges. Meanwhile, the diversity of the population rapidly descends to a low level, and the population focuses on local search around the obtained best solution. Because the intensity of noise in P_5 is small, the convergence of BPSO is not significantly influenced. For P_7, the massive local optimal solutions are regularly distributed in the feasible region, BPSO can also quickly locate the global optimal solution. However, BLDE tries to keep a balance between exploration and exploitation, and the bit-by-bit variation strategies make it more vulnerable to be frustrated by the noise of P_5 as well as the multi-modal landscape of P_7. As a consequence, BPSO performs better than BLDE on P_5 and P_7.

However, the local optimal solutions of MKPs are not regularly distributed. Thus, to efficiently explore the feasible regions, it is vital to keep a balance between exploration and exploitation. Fig. 1 (c–f) demonstrates binDE and AQDE can keep a balance between exploration and exploitation for the compared algorithms. Thus, AQDE performs better than BLDE on the test problem P_{8-1}, and binDE performs better than BLDE on P_{8-1}, P_{8-2} and P_{8-3}.

5. Performance of BLDE on the unit commitment problem

In this section, we employ BLDE for solving the unit commitment problem (UCP) in power systems. To minimize the production cost over a daily to weekly time horizon, UCP involves the optimum scheduling of power generating units as well as the determination of the optimum amounts of power to be generated by committed units[2] [5]. Thus, UCP is a mixed integer optimization problem, the decision variables of which are the binary string representing the on/off statuses of units and the real variables indicating the generated power of units.

5.1. Objective function of UCP

The objective of UCP is to minimize the total production cost

$$F = \sum_{t=1}^{T} \sum_{i=1}^{N} [\phi_i(P_{it}) \cdot u_{it} + \psi_{it} \cdot (1 - u_{i,t-1}) \cdot u_{i,t}] \tag{4}$$

[2] To compare with the work reported in [5], we employ similar notations and descriptions in this section.

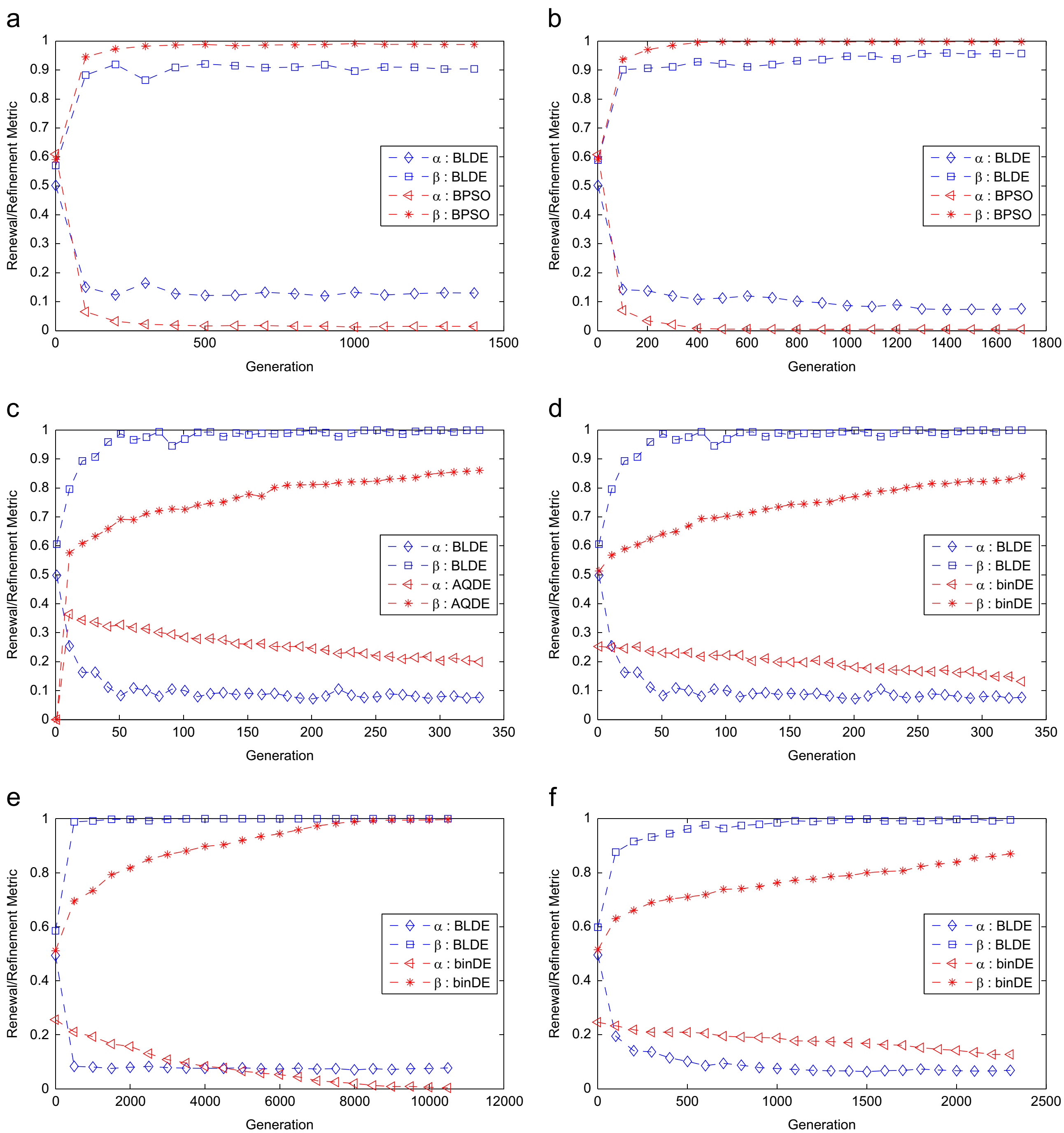

Fig. 1. Comparisons of the renewal and refinement metrics for test problems P_5, P_7, P_{8-1}, P_{8-2}, P_{8-3}. (a) P_5 : BLDE vs. BPSO. (b) P_7: BLDE vs. BPSO. (c) P_{8-1}: BLDE vs. AQDE. (d) P_{8-1}: BLDE vs. binDE. (e) P_{8-2}: BLDE vs. binDE. (f) P_{8-3}: BLDE vs binDE.

where N is the number of units to be scheduled and T is the time horizon. When the ith unit is committed to generating power P_{it} at time t, the binary variable u_{it} is set to be 1; otherwise, $u_{it}=0$. The function $\phi_i(P_{it})$ represents the fuel cost of unit i at time t, which is frequently approximated by

$$\phi_i(P_{it}) = a_i + b_i p_{it} + c_i P_{it}^2 \tag{5}$$

where a_i, b_i and c_i are known coefficients of unit i. If the ith unit has been off prior to start-up, there is a start-off cost

$$\psi_{it} = \begin{cases} d_i, & \text{if } \Gamma_i^{down} \le \tau_{it}^{off} \le \Gamma_i^{down} + f_i \\ e_i, & \text{if } \tau_{it}^{off} > \Gamma_i^{down} + f_i \end{cases} \tag{6}$$

where d_i, e_i, f_i and Γ_i^{down} are the hot start cost, cold start cost, cold start time and minimum down time of unit i, respectively. τ_{it}^{off}, the continuously off time of unit i, is determined by

$$\tau_{it}^{off} = \begin{cases} 0 & \text{if } u_{it} = 1 \\ 1 & \text{if } u_{it} = 0,\ t = 1 \text{ and } \sigma_i > 0 \\ 1 - \sigma_i & \text{if } u_{it} = 0,\ t = 1 \text{ and } \sigma_i < 0 \\ 1 + \tau_{i,t-1}^{off} & \text{if } u_{it} = 1 \text{ and } t > 1 \end{cases} \tag{7}$$

where σ_i is the initial status of unit i, which shows for how long the unit was on/off prior to the start of the time horizon.

5.2. Constraints in UCP

The minimization of the total production cost is subject to the following constraints.

Power balance constraints: The total generated power at time t must meet the power demand at that time instant, i.e.,

$$\sum_{i=1}^{N} u_{it}P_{it} = D_t, \quad t = 1, 2, \ldots, T \tag{8}$$

where D_t is the power demand at time t. In practice, it is hardly possible to exactly meet the power demand; therefore, an error ϵ is allowed for the generated power, i.e.,

$$\left|\frac{\sum_{i=1}^{N} u_{it}P_{it}}{D_t} - 1\right| \le \epsilon, \quad t = 1, 2, \ldots, T. \tag{9}$$

Spinning reserve constraints: Due to the possible equipment outages, it is necessary for power systems to satisfy the spinning reserve constraints. Thus, the sum of the maximum power generating capacities of all committed units should be greater than or equal to the power demand plus the minimum spinning reserve requirement, i.e.,

$$\sum_{i=1}^{N} u_{it}P_i^{max} \ge D_t + R_t, \quad t = 1, 2, \ldots, T \tag{10}$$

where P_i^{max} is the maximum power generating capacity of unit i, and R_t is the minimum spinning reserve requirement at time t.

Minimum up time constraints: If unit i is on at time t and switched off at time $t+1$, the continuous up time τ_{it}^{on} should be greater than or equal to the minimum up time Γ_i^{up} of unit i, i.e.,

$$\tau_{it}^{on} \ge \Gamma_i^{up} \quad \text{if } u_{it} = 1, u_{i,t+1} = 0 \text{ and } t < T, \quad i = 1, \ldots, N \tag{11}$$

where the continuously up time is

$$\tau_{it}^{on} = \begin{cases} 0 & \text{if } u_{it} = 0 \\ 1 & \text{if } u_{it} = 1,\ t = 1 \text{ and } \sigma_i < 0 \\ 1 + \sigma_i & \text{if } u_{it} = 1,\ t = 1 \text{ and } \sigma_i > 0 \\ 1 + \tau_{i,t-1}^{on} & \text{if } u_{it} = 1 \text{ and } t > 1. \end{cases} \tag{12}$$

Minimum down time constraints: If unit i is off at time t and switched on at time $t+1$, the continuous up time τ_{it}^{off} should be greater than or equal to the minimum off time Γ_i^{down} of unit i, i.e.,

$$\tau_{it}^{off} \ge \Gamma_i^{down}, \quad \text{if } u_{it} = 0, u_{i,t+1} = 1 \text{ and } t < T, \quad i = 1, \ldots, N \tag{13}$$

Range of generated power: The generated power of a unit is limited in an interval, i.e.,

$$P_i^{min} \le P_{it} \le P_i^{max}, \quad i = 1, 2, \ldots, N \text{ and } t = 1, 2, \ldots, T \tag{14}$$

where P_i^{min} and P_i^{max} is the minimum power output and the maximum power output of unit i, respectively.

5.3. Implement of BLDE for UCP

The optimal commitment of power units in UCP is obtained by combining BLDE with real-coded DE operations. In BLDE, each binary individual represents an on/off scheduling plan of units, accompanied with a real-coded individual representing the specific power outputs of units. When the binary individuals are recombined during the iteration process, the real-coded individuals are recombined via the *DE/rand/1* mutation and binary crossover strategies of the real-coded DE. Then, binary individuals and the corresponding real individuals are integrated together for evaluation. If the combined mixed-integer individuals violate the constraints in UCP, they are repaired via the repairing mechanisms proposed in [5].

The performance of BLDE is tested via a 10-unit power system, the parameters and forecasted power demands of which are respectively listed in Tables 7 and 8. To fairly compare BLDE with the method proposed in [5], we also set the population size to be 100, and the results are compared after 30 independent runs of 2500 iterations, where the scalar factor F is set to be 0.8. The statistical results are listed in Table 9.

The comparison results show that when the power balance error ϵ is small, performance of BLDE is a bit worse than that of the binary-real-coded differential evolution (BRCDE) algorithm proposed in [5]. However, when the power balance is relaxed to a relatively great extent, BLDE outperforms BRDE for UCP of the

Table 7
Unit parameters for the 10-unit power system.

Unit (i)	P_i^{max}(MW)	P_i^{min}(MW)	a_i (\$/h)	b_i (\$/MWh)	c_i (\$/MW^{2}h)	d_i (\$)	e_i (\$)	f_i (h)	Γ_i^{up}(h)	Γ_i^{down}(h)	σ_i (h)
1	455	150	1000	16.19	0.00048	4500	9000	5	8	8	8
2	455	150	970	17.26	0.00031	5000	10 000	5	8	8	8
3	130	20	700	16.60	0.00200	550	1100	4	5	5	−5
4	130	20	680	16.50	0.00211	560	1120	4	5	5	−5
5	162	25	450	19.70	0.00398	900	1800	4	6	6	−6
6	80	20	370	22.26	0.00712	170	340	2	3	3	−3
7	85	25	480	27.74	0.00079	260	520	2	3	3	−3
8	55	10	660	25.92	0.00413	30	60	0	1	1	−1
9	55	10	665	27.27	0.00222	30	60	0	1	1	−1
10	55	10	670	27.79	0.00173	30	60	0	1	1	−1

Table 8
Forecasted power demands for the 10-unit system over 14-h time horizon.

Hour	1	2	3	4	5	6	7	8	9	10	11	12
Demand (MW)	700	700	850	950	1000	1100	1150	1200	1300	1400	1450	1500
Hour	13	14	15	16	17	18	19	20	21	22	23	24
Demand (MW)	1400	1300	1200	1050	1000	1100	1200	1400	1300	1100	900	800

Table 9
Results comparison between BLDE and BRCDE [5] for the 10-unit power system."-" means that the corresponding item was not presented in the literature.

Method	Power balance error ϵ (%)	Best cost	Average cost	Worst cost	Standard deviation
BRCDE	0.0	563 938	–	–	–
	0.1	563 446	563 514	563 563	30
	0.5	561 876	–	–	–
	1	559 357	–	–	–
BLDE	0.0	563 977	564 005	564 088	24
	0.1	563 552	563 636	563 745	49
	0.5	561 677	561 847	–	50
	1	559 155	559 207	559 426	48

10-unit power system. The reason could be that crossover operation for real variables is not appropriately regulated for UCP, and accordingly, simultaneous variations on all real variables usually lead to violations of constraints. Thus, BLDE can only outperform BRCDE when the constraints are relaxed greatly.

6. Discussions

In this paper, we propose a BLDE algorithm appropriately incorporating the mutation strategy of binary DE and the learning mechanism of binary PSO. For the majority of the selected benchmark problems, BLDE can outperform the compared algorithms, which indicate that BLDE is competitive with the compared algorithms. However, statistical test results show that BPSO performs better than BLDE on P_5 and P_7, AQDE is more efficient for P_{8-1}, and binDE obtains better results on P_{8-1}, P_{8-12} as well as P_{8-3}. When generating a candidate solution, BLDE first initiates it as the winner of two obtained solutions and then regulates it by learning from the best individual in the population. This strategy simultaneously incorporates the synchronously changing strategy and the bitwise mutation strategy of candidate generation. Thus, BLDE can perform well on most of the high-dimensional benchmark problems. However, when BLDE is employed to solve P_5 and P_7, the global optimal solutions of which are easy to be locate, it performs worse than BPSO; when it is implemented to solve the low-dimensional problems P_{8-1}, P_{8-2} and P_{8-3}, the local optimal solutions of which are irregularly distributed in the feasible regions, it cannot perform better than binDE.

7. Conclusions

Generally, the proposed BLDE is competitive with the existing binary evolutionary algorithms. However, its performance can be improved. Thus, future work will focus on designing an adaptive strategy appropriately managing the synchronously changing strategy and the bitwise mutation strategy employed in BLDE. Meanwhile, we will try to further improve its performances on mixed-integer optimization problems by efficiently incorporating it with real-coded recombination strategies.

Acknowledgments

This work was partially supported by the Natural Science Foundation of China under Grant No. 51039005, 61173060, 91230118 and 61303028, as well as the Fundamental Research Funds for the Central Universities (WUT: 2013-Ia-001).

References

[1] A. Banks, J. Vincent, C. Anyakoha, A review of particle swarm optimization, Part I: background and development, Nat. Comput. 6 (4) (2007) 467–484.
[2] A. Banks, J. Vincent, C. Anyakoha, A review of particle swarm optimization, Part II: hybridisation, combinatorial, multicriterial, and constrained optimization and indicative applications, Nat. Comput. 7 (1) (2008) 109–124.
[3] S. Das, A. Konar, U.K. Chakraborty, Improving particle swarm optimization with differentially perturbed velocity, in: Proceedings of 2005 Conference on Genetic and Evolutionary Computation (GECCO'05), ACM, 2005, pp. 177–184.
[4] S. Das, P.N. Suganthan, Differential evolution: a survey of the state-of-the-art, IEEE Trans. Evol. Comput. 15 (2011) 4–31.
[5] D. Datta, S. Dutta, A binary-real-coded differential evolution for unit commitment problem, Elect. Power Energy Syst. 42 (1) (2012) 517–524.
[6] R.C. Eberhart, Y. Shi, Particle swarm optimization: developments, applications and resources, in: Proceedings of 2001 IEEE Conference on Evolutionary Computation (CEC 2001), IEEE, 2001, pp. 81–86.
[7] A.P. Engelbrecht, G. Pampará, Binary differential evolution strategies, in: Proceedings of 2007 IEEE Congress on Evolutionary Computation, IEEE, 2007, pp. 1942–1947.
[8] J. Gibbons, S. Chakraborti, Nonparametric Statistical Inference, Fifth ed., CRC Press, Tayor & Francis Group, Boca Ranton, Florida, USA, 2011.
[9] T. Gong, A.L. Tuson, Differential evolution for binary encoding, in: Soft Computing in Industrial Applications, Springer, 2007, pp. 251–262.
[10] X. He, L. Han, A novel binary differential evolution algorithm based on artificial immune system, in: Proceedings of 2007 IEEE Congress on Evolutionary Computation, IEEE, 2007, pp. 2267–2272.
[11] J. Horn., D.E. Goldber, K. Deb, Long path problems, in: Proceedings of 1994 International Conference on Parallel Problem Solving from Nature (PPSN III), Springer, 1994, pp. 149–158.
[12] A.R. Hota, A. Pat, An adaptive quantum-inspired differential evolution algorithm for 0–1 knapsack problem, in: Proceedings of 2010 Second World Congress on Nature and Biologically Inspired Computing (NaBIC), IEEE, 2010, pp. 703–708.
[13] M.H. Kashan, N. Nahavandi, A.H. Kashan, DisABC: a new artificial bee colony algorithm for binary optimization, Appl. Soft Comput. 12 (1) (2012) 342–352.
[14] M.H. Kashan, A.H. Kashan, N. Nahavandi, A novel differential evolution algorithm for binary optimization, Comput. Optim. Appl. 55 (2) (2013) 481–513.
[15] J. Kennedy, R.C. Eberhart, Particle swarm optimization, in: Proceedings of 1995 IEEE International Conference on Neural Networks, IEEE, 1995, pp. 1942–1948.
[16] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, IEEE, 1997, pp. 4104–4108.
[17] J. Kennedy, Particle swarm optimization, in: Encyclopedia of Machine Learning, Springer, 2010, pp. 760–766.
[18] P.W. Moore, G.K. Venayagamoorthy, Evolving digital circuit using hybrid particle swarm optimization and differential evolution, Int. J. Neural Syst. 16 (3) (2006) 163–177.
[19] M.G.H. Omran, A.P. Engelbrecht, A. Salman, Bare bones differential evolution, Eur. J. Oper. Res. 196 (1) (2009) 128–139.
[20] G. Pampara, N. Franken, A.P. Engelbrecht, Combining particle swarm optimisation with angle modulation to solve binary problems, in: Proceedings of 2005 IEEE Congress on Evolutionary Computation, IEEE, 2005, pp. 89–96.
[21] G. Pampara, A.P. Engelbrecht, N. Franken, Binary differential evolution, in: Proceedings of 2006 IEEE Congress on Evolutionary Computation, IEEE, 2006, pp. 1873–1879.
[22] A. Ponsich, C.A. Coello Coello, Differential evolution performances for the solution of mixed-integer constrained process engineering problems, Appl. Soft Comput. 11 (2011) 399–409.
[23] R. Poli, J. Kennedy, T. Blackwell, Particle swarm optimization: an overview, Swarm Intell. 1 (2007) 33–57.
[24] K. Price, R. Storn, J. Lampinen, Differential Evolution: A Practical Approach to Global Optimization, Springer, Berlin, Heidelberg, 2005.
[25] I.T. Rekanos, Shape reconstruction of a perfectly conducting scatterer using differential evolution and particle swarm optimization, IEEE Trans. Geosci. Remote Sens. 46 (7) (2008) 1967–1974.
[26] R. Storn, K. Price, Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces, J. Glob. Optim. 11 (1997) 341–359.
[27] Ş. Uyar, G. Eryiğit, Improvements to penalty-based evolutionary algorithms for the multi-dimensional knapsack problem using a gene-based adaptive mutation approach, in: Proceedings of 2005 Conference on Genetic and Evolutionary Computation (GECCO '05), ACM, 2005, pp. 1257–1264.
[28] J. Vesterstrom, R. Thomsen, A comparative study of differential evolution, particle swarm optimization and evolutionary algorithms on numerical benchmark problems, in: Proceedings of 2004 IEEE Conference on Evolutionary Computation (CEC'04), IEEE, 2004, pp. 1980–1987.
[29] C.-Y. Wu, K.-Y. Tseng, Topology optimization of structures using binary differential evolution, Struct. Multidiscipl. Optim. 42 (2010) 939–953.
[30] URL: ⟨http://www.zib.de/index.php?id=921&no_cache=1&L=0&cHash=fbd4ff9555f8714ac6238261e3963432&type=98⟩.
[31] Q. Yang, A comparative study of discrete differential evolution on binary constraint satisfaction problems, in: Proceedings of 2008 IEEE Congress on Evolutionary Computation, IEEE, 2008, pp. 330–335.

Yu Chen received the BS degree from Sichuan University, China in 2001, the MS and PhD degrees from Wuhan University, China in 2007 and 2011, respectively. He is currently an associate professor at the School of Science, Wuhan University of Technology, China. His research interests include evolutionary computation and its applications in system biology.

Weicheng Xie received his PhD degree in Computational Mathematics from Wuhan University, China in 2013. He is now a postdoctoral researcher at Shenzhen Key Laboratory of Spatial Smart Sensing and Services, School of Computer Science and Software Engineering, Shenzhen University. His current researches focus on the computer vision and the intelligent recognition of human face.

Xiufen Zou received the BS and MS degrees in computational mathematics and the PhD degree in computer science from Wuhan University, China, in 1986, 1991, and 2003, respectively. She is currently a full professor at the School of Mathematics and Statistics, Wuhan University. Her current research interests include computational intelligence and its applications in complex biological systems. She has published more than 40 academic papers in the aforementioned areas. She was a recipient of the First-Class Prize of the Natural Science Award of Hubei Province, China, in 2005 for her paper "Parallel Evolutionary Optimization and Modeling Algorithms" and the Third-Class Prize of the Science and Technology Progress Award of the National Educational Ministry of China in 1998 for her paper "Parallel Computational Model and Algorithms for Complex System."