Adaptive Strategy Selection in Differential Evolution for Numerical Optimization: An Empirical Study☆

Wenyin Gong[a,b], Álvaro Fialho[c], Zhihua Cai[*,a], Hui Li[a]

[a]*School of Computer Science,*
China University of Geosciences, Wuhan 430074, P.R. China
[b]*State Key Laboratory of Software Engineering,*
Wuhan University, 430072, P.R. China
[c]*Nokia Institute of Technology,*
69048-660 Manaus/AM, Brazil

Abstract

Differential evolution (DE) is a versatile and efficient evolutionary algorithm for global numerical optimization, which has been widely used in different application fields. However, different strategies have been proposed for the generation of new solutions, and the selection of which of them should be applied is critical for the DE performance, besides being problem-dependent. In this paper, we present two DE variants with adaptive strategy selection: two different techniques, namely *Probability Matching* and *Adaptive Pursuit*, are employed in DE to autonomously select the most suitable strategy while solving the problem, according to their recent impact on the optimization process. For the measurement of this impact, four credit assignment methods are assessed, which update the known performance of each strategy in different ways, based on the relative fitness improvement achieved by its recent applications. The performance of the analyzed approaches is evaluated on twenty-two benchmark functions. Experimental results confirm that they are able to adaptively choose the most suitable strategy for a specific problem in an efficient way. Compared with other state-of-the-art DE variants, better results are obtained on most of the functions in terms of quality of the final solutions and convergence speed.

Key words: Differential evolution, adaptation, strategy selection, credit assignment, numerical optimization

1. Introduction

Differential evolution (DE), proposed by Storn and Price [35], is an efficient and versatile population-based direct search algorithm that implements the evolutionary generation-and-test paradigm for global optimization, using the distance and direction informations from the current population to guide the search. Among its advantages are its simple structure, ease of use, speed, and robustness, which enables its application on many real-world applications, such as data mining, IIR design, neural network training [29], power systems [43], financial market dynamics modeling [16], data mining [4], and so on. A good survey of DE can be found in [5], where its basic concepts and major variants, as well as some theoretical studies and application examples to complex environments, are reviewed in detail.

In the seminal DE algorithm [35], a single mutation strategy was used for the generation of new solutions; later on, Price and Storn suggested nine other different strategies [29, 36]. In addition, other mutation strategies are also proposed in the DE literature [50, 3, 6, 8]. Although augmenting the robustness of the underlying algorithm, these

☆This work was partly supported by the Fundamental Research Funds for the Central Universities at China University of Geosciences (Wuhan) under Grant No. CUG100316, the Foundation of State Key Lab of Software Engineering under Grant No. SKLSE2010-08-13, the National Natural Science Foundation of China under Grant No. 61075063, and the Research Fund for the Doctoral Program of Higher Education under Grant No. 20090145110007.

*Corresponding author. Tel: +86-27-67883716.

Email addresses: cug11100304@yahoo.com.cn (Wenyin Gong), fialho@lix.polytechnique.fr (Álvaro Fialho), zhcai@cug.edu.cn (Zhihua Cai), huili@vip.sina.com (Hui Li)

many available strategies led the user to the need of defining which of them would be most suitable for the problem at hand – a difficult and crucial task for the performance of DE [31, 30, 23].

Off-line tuning techniques, such as the F-Race [1], could be used to choose the mutation strategy to be used. However, besides being computationally expensive, such techniques usually output a static setting; while, in practice, the performance of each mutation strategy does not depend on the problem itself, but rather on the characteristics of the region of the search landscape being explored by the population at each generation. Based on this, thus, in order to be more efficient, the autonomous selection of the strategy to be used should be done in a continuous way, while solving the problem, *i.e.*, dynamically adapting itself as the search goes on.

In order to contribute on remedying this drawback, in this paper, we extend our recent work [15] on the use of adaptive strategy selection within DE for global numerical optimization. To do adaptive strategy selection, *i.e.*, to be able to automatically select which is the best mutation strategy for the generation of each offspring while solving the problem, two elements need to be defined [48, 18]: (i) how to select between the available strategies based on their recent performance (strategy selection); and (ii) how to measure the performance of the strategies after their application, and consequently update the empirical quality estimates kept for each of them (credit assignment). In this work, two strategy selection techniques, namely *Probability Matching* [12] and *Adaptive Pursuit* [41], are independently analyzed in combination with each of four credit assignment techniques based on the relative fitness improvement. In addition, a parameter sensitivity analysis is conducted to investigate the impact of the hyper-parameters on the performance of the resulting adaptive strategy selection technique. Experiments have been conducted on 22 widely used benchmark problems, including 9 test functions presented in CEC-05 [37]. The results indicate that the analyzed approach is able to select the most suitable strategy, while solving a problem at hand. Compared with other state-of-the-art DE variants, better results are obtained on most of the functions in terms of quality of final solutions and convergence speed.

Compared with our previous work in [15], the main contributions of this paper are two-fold: (i) in order to pursuit the most suitable strategy at different search stages for a specific problem more rapidly, the *Adaptive Pursuit* technique is used and its performance is compared with the *Probability Matching*-based DE variant; and (ii) the comprehensive experiments are conducted to verify our approach and its performance is analyzed in detail.

The remainder of the paper is organized as follows. Section 2 briefly introduces the background and related work of this paper. In Section 3, we describe the adaptive strategy selection approaches in detail, followed by the experimental results and discussions in Section 4. Finally, Section 5 is devoted to conclusions and future work.

2. Background and Related Work

2.1. Problem Formulation

Without loss of generality, in this work, we consider the following numerical optimization problem:

$$\text{Minimize} \quad f(\mathbf{x}), \qquad \mathbf{x} \in S, \tag{1}$$

where $S \subseteq \mathbb{R}^D$ is a compact set, $\mathbf{x} = [x_1, x_2, \cdots, x_D]^T$, and D is the dimension, *i.e.*, the number of decision variables. Generally, for each variable x_j, it satisfies a boundary constraint, such that:

$$L_j \leq x_j \leq U_j, j = 1, 2, \cdots, D. \tag{2}$$

2.2. Differential Evolution

DE [35] is a simple evolutionary algorithm (EA) for global numerical optimization. It creates new candidate solutions by combining the parent individual and several other individuals of the same population. A candidate replaces the parent only if it has an equal or better fitness value. The pseudo-code of the original DE algorithm is shown in Algorithm 1, where D refers to the number of decision variables (or problem dimension); NP is the population size; F is the mutation scaling factor; CR is the crossover rate; $x_{i,j}$ is the j-th variable of the solution x_i; $\mathbf{u}_i$ is the offspring. The function rndint$(1, D)$ returns a uniformly distributed random integer number between 1 and D, while rndreal$_j[0, 1)$ gives a uniformly distributed random real number in $[0, 1)$, generated anew for each value of j. With respect to the population initialization, the widely used method is uniformly random initialization

Algorithm 1 The DE algorithm with DE/rand/1/bin strategy

1: Generate the initial population
2: Evaluate the fitness for each individual
3: **while** the termination criterion is not satisfied **do**
4: **for** $i = 1$ to NP **do**
5: Select uniform randomly $r_1 \neq r_2 \neq r_3 \neq i$
6: $j_{rand} = \text{rndint}(1, D)$
7: **for** $j = 1$ to D **do**
8: **if** $\text{rndreal}_j[0, 1) < CR$ **or** j is equal to j_{rand} **then**
9: $u_{i,j} = x_{r_1,j} + F \cdot (x_{r_2,j} - x_{r_3,j})$
10: **else**
11: $u_{i,j} = x_{i,j}$
12: **end if**
13: **end for**
14: **end for**
15: **for** $i = 1$ to NP **do**
16: Evaluate the offspring $\mathbf{u}_i$
17: **if** $f(\mathbf{u}_i)$ is better than **or** equal to $f(\mathbf{x}_i)$ **then**
18: Replace $\mathbf{x}_i$ with $\mathbf{u}_i$
19: **end if**
20: **end for**
21: **end while**

within the search space. Other initialization methods are also available, for example, orthogonal initialization [13], opposition-based initialization [32], chaotical initialization [27], etc.

From Algorithm 1, we can see that there are only three control parameters in DE. These are NP, F and CR. As for the terminal conditions, we can either fix the maximum number of fitness function evaluations (*Max_NFFEs*) or define a desired solution value to be reached (*VTR*).

In DE, many schemes have been proposed that use different mutation strategies and/or recombination operations in the reproduction stage [29, 36]. In order to distinguish among its schemes, the notation "DE/a/b/c" is used, where "DE" denotes the DE algorithm; "a" specifies the vector to be mutated; "b" is the number of difference vectors used; and "c" denotes the crossover scheme, *binomial* or *exponential*, this latter being fixed to the binomial on the remainder of this work. In line 9 of Algorithm 1, the mutation strategy is called "DE/rand/1", which is a classic strategy of DE [29]. Other well-known mutation strategies can be listed as follows.

1) "DE/best/1":
$$\mathbf{v}_i = \mathbf{x}_{best} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{3}$$

2) "DE/best/2":
$$\mathbf{v}_i = \mathbf{x}_{best} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) + F \cdot (\mathbf{x}_{r_4} - \mathbf{x}_{r_5}) \tag{4}$$

3) "DE/rand/2":
$$\mathbf{v}_i = \mathbf{x}_{r_1} + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) + F \cdot (\mathbf{x}_{r_4} - \mathbf{x}_{r_5}) \tag{5}$$

4) "DE/current-to-best/1"[1]:
$$\mathbf{v}_i = \mathbf{x}_i + F \cdot (\mathbf{x}_{best} - \mathbf{x}_i) + F \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{6}$$

where $\mathbf{x}_{best}$ represents the best individual in the current generation, r_1, r_2, r_3, r_4, and $r_5 \in \{1, \cdots, NP\}$, and $r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5 \neq i$.

[1] "DE/current-to-best" is also referred to as "DE/target-to-best/" [29, 3].

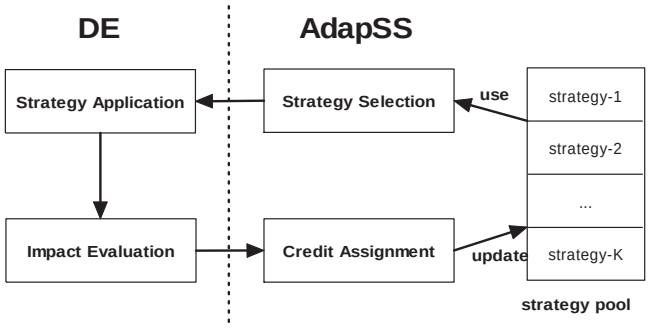

Figure 1: General framework of adaptive strategy selection within DE.

2.3. Adaptive Strategy Selection

Typically, the parameter setting in EAs is done before launching the main runs that will be used to assess the algorithm. In this case, the parameters are defined according to the user's experience, or by an *external tuning* method, which can be a standard statistical analysis, a more engineered procedure, or even another optimization algorithm. The main drawback of such *off-line* methods is that they define a static setting, what generally leads to sub-optimal performance. Intuitively, as the algorithm proceeds from a global (early) exploration of the landscape to a more focused, exploitation-like behavior, the parameters should be adjusted to take care of this new reality. Indeed, it has been empirically and theoretically demonstrated that different values of parameters might be optimal at different stages of the search process (see, *e.g.*, [7] and references therein).

Thus, to achieve better performance, the parameter setting should be done while solving the problem, adapting the behavior of the algorithm as needed. Following [7], the *internal control* of the parameters can be done in different ways. *Deterministic* methods modify the parameters values according to predefined rules; *Self-Adaptive* methods encode the parameters within the genotype, which is thus evolved in parallel with the solution; and lastly, the *Adaptive* methods use changes in some particular properties of the search process as an input signal to modify the parameter values. While the first approach introduces the extra difficulty of defining the control rules, the second defines the parameters, but the parameters space is merged with the solutions space, thus augmenting the overall complexity of the search.

This paper is focused on the latter approach, more specifically, on the *Adaptive Strategy Selection* (AdapSS) paradigm. Inspired by some recent works in the Genetic Algorithms (GAs) community (see, *e.g.*, [41, 10]), its objective is to automatically select among the available (possibly ill-known) mutation strategies, according to their performance on the search up to now. To do so, as illustrated in Figure 1, there is the need for two components: the credit assignment, that defines how the impact of the strategies on the search should be assessed and transformed into a numerical reward; and the strategy (or operator) selection mechanism that, based on the rewards received, selects which strategy should be applied at the given moment of the search.

2.4. DE with Strategy Adaptation

In the context of *on-line* selection among multiple mutation strategies within DE, some approaches can be found in the literature. Xie and Zhang [45] presented a swarm algorithm framework, in which a neural network is used to adaptively update the weights of the DE strategies. Qin *et al.* [31, 30] proposed a variant of DE, named SaDE, that updates the weights of each strategy in the search based on their previous success rate. In [2, 47], strategy adaptation techniques similar to SaDE are used to enhance DE performance. In [21], both the mutation strategies and the crossover operation are adaptively selected in DE. In [15], we proposed the use of a strategy adaptation method for DE, based on the *Probability Matching* technique being fed by relative fitness improvements; while in Gong *et al.* [14] a different family of strategy adaptation techniques was presented, where a strategy parameter η is used control the selection of different strategies, and two simple strategy adaptation mechanisms are implemented to update the parameter. Based on their previous work in [21], Mallipeddi *et al.* [20] presented a DE variant with ensemble of parameters and mutation strategies, called EPSDE, in which the strategy of each target vector is initialized randomly

and, during the evolution process, if the generated offspring is better than its target vector, the strategy of the target vector is stored in the next generation; otherwise, the strategy of the target vector is selected randomly from the pool or from the previous successful strategies stored with equal probability. In [28], Pan *et al.* presented an improved DE, referred to as SspDE, in which a strategy list, a scaling factor list, and a crossover rate list are encoded in the individual, being constantly updated during the evolution in a self-adaptive manner. Wang *et al.* [42] proposed a composite DE (CoDE). In CoDE, each strategy generated its trial vector with a parameter setting randomly selected from the parameter candidate pool.

3. Adaptive Strategy Selection in DE

In order to automatically select the most suitable strategy while solving a problem without any *prior* knowledge, in this work, we analyze the use of strategy adaptation methods in DE for numerical optimization problems. This is an extension of our recent work in [15], which has been considerably enhanced, with the major differences being listed as follows.

- In this work, two strategy selection techniques, namely *Probability Matching* (PM) [12] and *Adaptive Pursuit* (AP) [41], are analyzed and empirically compared with the baseline approaches, while in [15] only the *Probability Matching* was adopted.
- A different pool of four mutation strategies, proposed in [50, 49], is used.
- The parameter adaptation method of CR and F proposed in [50] is adopted in this work, while in [15] CR and F were set to pre-defined values.
- The sensitivity of the parameters on the performance of the *Adaptive Pursuit* technique is empirically investigated.

The main objectives of this work are two-fold. Firstly, the *Probability Matching* and *Adaptive Pursuit* strategy selection techniques are independently integrated into JADE and compared to other existing approaches. Secondly, four credit assignment techniques based on the relative fitness improvement are compared. These components for adaptive strategy selection, as well as the JADE algorithm to which they were combined to, are better described in the following, in Sections 3.1 to 3.4.

3.1. Strategy Selection

Suppose we have $K > 1$ strategies in the pool $A = \{a_1, \cdots, a_K\}$ and a probability vector $\mathbf{P}(t) = \{p_1(t), \cdots, p_K(t)\}$ ($\forall t : p_{min} \leq p_i(t) \leq 1; \sum_{i=1}^{K} p_i(t) = 1$). In this work, the *Probability Matching* (PM) and *Adaptive Pursuit* (AP) techniques are used to adaptively update the probability $p_a(t)$ of strategy a based on its known performance (frequently updated by the rewards received). Denote $r_a(t)$ as the reward that a strategy a receives after its application at time t. $q_a(t)$ is the known quality (or empirical estimate) of a strategy a, that is updated as follows [41]:

$$q_a(t+1) = q_a(t) + \alpha \cdot [r_a(t) - q_a(t)], \tag{7}$$

where $\alpha \in (0, 1]$ is the adaptation rate.

Based on this common quality empirical estimate, the PM and AP methods differ on the way they use this information to update the application probability of each strategy, as detailed in the following.

3.1.1. Probability Matching

The PM method updates the probability $p_a(t)$ as follows [12, 41]:

$$p_a(t+1) = p_{min} + (1 - K \cdot p_{min}) \frac{q_a(t+1)}{\sum_{i=1}^{K} q_i(t+1)}. \tag{8}$$

where $p_{min} \in (0, 1)$ is the minimal probability value of each strategy, used to ensure that no operator gets lost [41]. From Equation (8), we can see that when only one strategy obtains a reward during a long period of time and all other strategies receive no reward, then its selection probability $p_a(t)$ converges to $p_{max} = p_{min} + (1 - K \cdot p_{min})$. Obviously, $\sum_{a=1}^{K} p_a(t) = 1$ and $0 < p_{min} < \frac{1}{K}$

3.1.2. Adaptive Pursuit

The *Adaptive Pursuit* (AP) method is a pursuit algorithm recently introduced for adaptive operator selection in the context of GAs [41], which was originally proposed for learning automata [40]. After adapting the empirical quality estimate of each strategy in the same way than the PM method, as presented in Equation 7, the probability of each strategy is updated as follows:

$$p_{a^*}(t+1) = p_{a^*}(t) + \beta \cdot [p_{max} - p_{a^*}(t)] \tag{9}$$

and

$$\forall a \neq a^* : p_a(t+1) = p_a(t) + \beta \cdot [p_{min} - p_a(t)] \tag{10}$$

where

$$a^* = \text{argmax}_a\Big(q_a(t+1)\Big)$$

and

$$p_{max} = p_{min} + 1 - K \cdot p_{min}$$

This constraint makes sure that if $\sum_{a=1}^{K} p_a(t) = 1$, then the sum of the updated probabilities is also equal to 1, *i.e.*, $\sum_{a=1}^{K} p_a(t+1) = 1$ [41].

As in the PM method, the minimal probability p_{min} is used to ensure that no operator gets lost, and $0 < p_{min} < \frac{1}{K}$. Besides the adaptation rate α used on the update of the empirical quality estimates (Eq. 7), the AP has another hyper-parameter, the learning rate $\beta \in (0, 1]$, which basically controls how greedy the "winner-takes-all" strategy will behave.

3.2. Credit Assignment

Besides the strategy selection itself, another important issue to implement the adaptive strategy selection paradigm is the credit assignment, as shown in Figure 1. In this work, in order to assign the credit or reward for each strategy, we adopt the relative fitness improvement η_i proposed in [26] as follows:

$$\eta_i = \begin{cases} \frac{\delta}{cf_i} \cdot |pf_i - cf_i|, & \text{for minimization} \\ \frac{cf_i}{\delta} \cdot |pf_i - cf_i|, & \text{for maximization} \end{cases} \tag{11}$$

where $i = 1, \cdots, NP$. δ is the fitness of the best-so-far solution in the population. pf_i and cf_i are the fitness of the target parent and of its offspring, respectively. Note that if no improvement is achieved (*i.e.*, the offspring is worse than or equal to its target parent), the impact of the strategy application is considered as null ($\eta_i = 0$).

Denote S_a as the set of all relative fitness improvements achieved by the application of a strategy $a (a = 1, \cdots, K)$ during generation t. At the end of the generation, an unique reward is used to update the quality measure kept by the PM and AP methods (Equation 7). Following [10], to extract such reward from S_a, we analyze four different credit assignment methods as follows:

- **Average Absolute Reward** (AvgAbs):

$$r_a(t) = \frac{\sum_{i=1}^{|S_a|} S_a(i)}{|S_a|} \tag{12}$$

  where $|S_a|$ is the number of elements in S_a. If $|S_a| = 0$, $r_a(t) = 0$.

- **Average Normalized Reward** (AvgNorm):

$$r'_a(t) = \frac{\sum_{i=1}^{|S_a|} S_a(i)}{|S_a|}; \text{ and } r_a(t) = \frac{r'_a(t)}{\max\limits_{b=1,\cdots,K} r'_b(t)} \tag{13}$$

- **Extreme Absolute Reward** (ExtAbs):

$$r_a(t) = \max_{i=1,\cdots,|S_a|} S_a(i) \tag{14}$$

- **Extreme Normalized Reward** (ExtNorm):

$$r'_a(t) = \max_{i=1,\cdots,|S_a|} S_a(i); \text{ and } r_a(t) = \frac{r'_a(t)}{\max_{b=1,\cdots,K} r'_b(t)} \tag{15}$$

The reasons for adopting these four credit assignment techniques are as follows:

- Intuitively, the first technique is reasonable, and there are many approaches using the average improvement [10].
- In [44], the extreme improvement, using a statistical measure aimed at outlier detection, is considered. This method showed better performance than its competitors on a set of benchmark problems. Based on this inspiration, the third method is used herein.
- With respect to the second and fourth methods, the average and the extreme improvements are respectively normalized. The assumptions to justify such normalization can be summarized in the following manner: (i) there might be magnitude differences between rewards received in two different time instants of the search, with a later reward possibly affecting much less the update of the empirical quality estimate than it should, as rewards tend to get smaller as the search goes on; and (ii) there might also be magnitude differences in the fitness ranges of different problems, what could completely affect the behavior of the adaptive strategy selection method in case one uses the raw reward values.

3.3. Strategy Pool

As previously mentioned, there are many strategies proposed in DE [29, 36], each one presenting its own characteristics. However, to the best of our knowledge, there are no theoretical studies as of today on the choice of the optimal number of available strategies (pool size) and on the selection of strategies to form the strategy pool [30]. In this work, we consider as the strategy pool the same four strategies used in the JADE method [50, 49], which are described as follows.

1) "DE/current-to-*p*best/1 (without archive)":

$$\mathbf{v}_i = \mathbf{x}_i + F_i \cdot (\mathbf{x}^p_{best} - \mathbf{x}_i) + F_i \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{16}$$

2) "DE/current-to-*p*best/1 (with archive)":

$$\mathbf{v}_i = \mathbf{x}_i + F_i \cdot (\mathbf{x}^p_{best} - \mathbf{x}_i) + F_i \cdot (\mathbf{x}_{r_2} - \tilde{\mathbf{x}}_{r_3}) \tag{17}$$

In the latter one, an archive $\mathbf{A}$ is used to store the inferior solutions recently explored in the evolutionary search. $\mathbf{x}^p_{best}$ refers to the *p*best solution, which is randomly selected from the top $100p\%$ solutions, with $p \in (0, 1]$. $\mathbf{x}_i$, $\mathbf{x}_{r_2}$, and $\mathbf{x}^p_{best}$ are chosen from the current population $\mathbf{P}$; $\tilde{\mathbf{x}}_{r_3}$ is randomly chosen from the union between the archive and current populations ($\mathbf{P} \cup \mathbf{A}$). Later on, in order to solve the large scale problems and further increase the population diversity, the same authors proposed other two strategies [49]:

3) "DE/rand-to-*p*best/1 (without archive)":

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F_i \cdot (\mathbf{x}^p_{best} - \mathbf{x}_{r_1}) + F_i \cdot (\mathbf{x}_{r_2} - \mathbf{x}_{r_3}) \tag{18}$$

4) "DE/rand-to-*p*best/1 (with archive)":

$$\mathbf{v}_i = \mathbf{x}_{r_1} + F_i \cdot (\mathbf{x}^p_{best} - \mathbf{x}_{r_1}) + F_i \cdot (\mathbf{x}_{r_2} - \tilde{\mathbf{x}}_{r_3}) \tag{19}$$

The reasons for choosing these strategies are three-fold. Firstly, they have individually obtained good performance as shown in [50, 49]. Secondly, the two strategies without archive converge faster and are more suitable to the low-dimensional problems; on the other hand, the strategies with archive can provide higher population diversity, hence being more suitable to the high-dimensional problems. Thirdly, the JADE method is used as baseline for empirical comparison (see Section 4); the use of the same strategy pool guarantees that the performance improvements achieved by our approaches with relation to JADE are solely due to the different proposed strategy adaptation mechanisms. Note that other strategies could also be used, these 4 strategies can be seen as a test-bed for the adaptive strategy selection method.

3.4. JADE with Adaptive Strategy Selection: AdapSS-JADE

By combining the above-mentioned three aspects with the JADE algorithm [50, 49], the AdapSS-JADE method is developed. Differently from our previous work [15], the CR and F adaptation mechanism proposed in JADE[2] [50] is adopted in parallel with the adaptive strategy selection scheme. From this point of view, this approach can be regarded as an improved JADE variant. It is worth noting that our proposed strategy adaptation method can also be used in other DE variants.

The pseudo-code is illustrated in Algorithm 2; modified steps with respect to the original JADE are marked with a left arrow "$\Leftarrow$". At each generation t, for each target parent i, a strategy SI_i is selected based on the probability of each strategy. Then the offspring is generated by the application of the selected strategy. After evaluating the offspring, the relative fitness improvement η_i is calculated and stored in the set S_{SI_i}. Consequently, the reward, quality, and probability of each strategy are updated.

As previously mentioned, the study on multiple strategies adaptation in DE is scarce. Compared with the approaches proposed in [30, 14, 20, 28], in which several mutation strategies are also used within DE, the main differences between our proposed approach and theirs can be listed as follows:

- In the SaDE method [30], the *Probability Matching* strategy selection scheme is also implemented. However, our approach is completely different from SaDE in the credit assignment, using the relative fitness improvements instead of the success and failure number of trials. It turns out to be a completely different method when using the *Adaptive Pursuit* for strategy selection.

- The SaJADE [14] is also based on the JADE method [50, 49]. However, the strategy adaptation is controlled by a strategy parameter. Two adaptive mechanisms are implemented to update this parameter, which are different from the strategy selection methods (AP and PM) used in this work.

- In EPSDE [20], the strategy of each target vector is initialized randomly. In the evolution process, if the offspring is better than its target vector, the strategy of the target vector is stored in the next generation; otherwise, it is randomly selected from the pool or from the previous successful strategies stored with equal probability.

- In SspDE [28], the strategy for each target vector is selected from the strategy list of this vector, which is updated during the search in a self-adaptive manner, thus being completely different from our proposed method.

From the previous subsections, we can see that our approach is either based on the PM or on the AP strategy selection techniques. In addition, the relative fitness improvement is used to assign the reward of each strategy. In general, our proposed approach is very different from the above-mentioned variants.

It is also worth noting that the use of our proposed approaches within JADE does not significantly increase the overall computational complexity of the original algorithm. The additional complexity of AdapSS-JADE is the adaptive strategy selection, as shown in Algorithm 2, which takes $O(K \cdot NP)$ operations, where K is the total number of strategies in the pool. Since the total complexity of JADE is $O(G \cdot NP \cdot (D + \log(NP)))$ [49], where G is the maximal number of generations, AdapSS-JADE has the total complexity of $O(G \cdot NP \cdot (D + \log(NP)) + K \cdot NP)$. Generally, $K \ll G \cdot (D + \log(NP))$, hence the overall complexity of our approach is $O(G \cdot NP \cdot (D + \log(NP)))$. In general, the population size NP is set to be proportional to the problem dimension D in the DE literature. Thus, the total complexity of AdapSS-JADE is $O(G \cdot D^2)$, which is the same as the classic DE algorithm, JADE, and many other DE variants.

4. Experimental Results

In order to evaluate the performance of our approach, 22 benchmark functions were selected as the test suit. Functions $f_{01} - f_{13}$ are chosen from [48]. Functions $f_{01} - f_{04}$ are unimodal. The Rosenbrock's function f_{05} is a multimodal function when $D > 3$ [33]. Function f_{06} is the step function, which has one minimum and is discontinuous. Function f_{07} is a noisy quartic function. Functions $f_{08} - f_{13}$ are multi-modal functions where the number of local

[2]More details about JADE can be found in [50, 49].

Algorithm 2 JADE with Adaptive Strategy Selection: AdapSS-JADE

Set $\mu_{CR} = 0.5; \mu_F = 0.5$
Generate the initial population randomly
Evaluate the fitness for each individual
Set the generation counter $t = 1$
Set $K = 4$, $p_{min} = 0.05$, $\alpha = 0.3$, and $\beta = 0.8$ (if any) ⇐
For each strategy a, set $q_a(t) = 0$ and $p_a(t) = 1/K$ ⇐
while The halting criterion is not satisfied **do**
  for $i = 1$ to NP **do**
    Select the strategy SI_i based on its probability ⇐
    Select uniform randomly $r_1 \neq r_2 \neq r_3 \neq r_4 \neq r_5 \neq i$
    j_{rand} = rndint$(1, D)$
    for $j = 1$ to D **do**
      if rndreal$_j[0, 1) < CR$ **or** $j == j_{rand}$ **then**
        if $SI_i == 1$ **then**
          $u_{i,j}$ is generated by strategy (16)
        else if $SI_i == 2$ **then**
          $u_{i,j}$ is generated by strategy (17)
        else if $SI_i == 3$ **then**
          $u_{i,j}$ is generated by strategy (18)
        else if $SI_i == 4$ **then**
          $u_{i,j}$ is generated by strategy (19)
        end if
      else
        $u_{i,j} = x_{i,j}$
      end if
    end for
  end for
  for $i = 1$ to NP **do**
    Evaluate the offspring $\mathbf{u}_i$
    if $f(\mathbf{u}_i)$ is better than **or** equal to $f(\mathbf{x}_i)$ **then**
      Calculate η_i using Equation (11) ⇐
      $CR_i \rightarrow S_{CR}; F_i \rightarrow S_F$
      Replace $\mathbf{x}_i$ with $\mathbf{u}_i$
    else
      Set $\eta_i = 0$ ⇐
    end if
    $S_{SI_i} \leftarrow \eta_i$ ⇐
  end for
  Update the μ_{CR} and μ_F
  Calculate the reward $r_a(t)$ for each strategy ⇐
  Update the quality $q_a(t)$ for each strategy ⇐
  Update the probability $p_a(t)$ for each strategy by PM or AP technique ⇐
  $t = t + 1$
end while

minima increases exponentially with the problem dimension. The other 9 test functions ($F_{06} - F_{14}$) were presented in CEC-05 [37], being all multi-modal functions, with shifted and/or rotated features making them very difficult to solve. Generally, these 22 functions can be categorized into three groups: (i) basic unimodal functions ($f_{01} - f_{07}$); (ii) basic multimodal functions ($f_{08} - f_{13}$); and (iii) shifted and/or rotated multimodal functions ($F_{06} - F_{14}$). Functions $f_{01} - f_{13}$ are described in Appendix A. A detailed description of functions $F_{06} - F_{14}$ can be found in [37]. According to the main objectives of this work, the experiments are carried out with the following key aims.

1) To compare the performance of AdapSS-JADE using each of the four different credit assignment techniques described in Section 3.2.

2) With the best credit assignment technique found, to compare the performance of AdapSS-JADE using different strategy selection techniques.

3) The strategy adaptation characteristics of the best adaptive strategy selection approach are analyzed, to demonstrate that it is able of efficiently selecting the most suitable strategy to be applied, while solving the problem, without any prior knowledge.

4) The sensitivity of the parameter settings of the best strategy selection technique is studied, to indicate the effect of these parameters on its performance.

The experimental settings and performance criteria used on the empirical analysis of these four issues are presented in the following, with the results concerning each of them being presented in Sections 4.3 to 4.7.

4.1. Experimental Settings

For all experiments, we use the following parameters unless a change is mentioned.

- Dimension of each function: $D = 30$;
- Population size: $NP = 100$ [50, 49];
- $\mu_{CR} = 0.5$ and $\mu_F = 0.5$, [50, 49];
- $c = 0.1$ and $p = 0.05$ [50, 49];
- Number of strategies: $K = 4$; minimal probability: $p_{min} = 0.05$; adaptation rate: $\alpha = 0.3$; and learning rate in AP: $\beta = 0.8$ (the parameter study will be discussed in Section 4.7);
- Value to Reach (VTR): For functions $f_{01} - f_{06}$ and $f_{08} - f_{13}$, VTR = 10^{-8}; for functions $f_{07}, F_{06} - F_{14}$, VTR = 10^{-2} [37, 50];
- Maximum Number of Fitness Function Evaluations (Max_NFFEs[3]): For f_{01}, f_{06}, f_{10}, f_{12}, and f_{13}, Max_NFFEs = 150, 000; for $f_{03} - f_{05}$, Max_NFFEs = 500, 000; for f_{02} and f_{11}, Max_NFFEs = 200, 000; for $f_{07} - f_{09}$, and $F_{06} - F_{14}$, Max_NFFEs = 300, 000.

Moreover, in our experiments, each function is optimized over 50 independent runs. We also use the same set of initial random populations to evaluate different algorithms in a similar way as done in [24]. All the algorithms are implemented in standard C++.

[3]The Max_NFFEs for functions $f_{01} - f_{13}$ are mainly set as in [48], except for f_{05}, f_{08}, and f_{09}, for which the numbers used are smaller than the original ones, since our approaches are able to obtain the global optimum of these functions within the Max_NFFEs. For functions $F_{06} - F_{14}$, the Max_NFFEs are set as in [37].

4.2. Performance Criteria

Four performance criteria are selected from the literature [37] to evaluate the algorithms. These criteria are described as follows.

- **Error**: The error of a solution $\mathbf{x}$ is defined as $f(\mathbf{x}) - f(\mathbf{x}^*)$, where $\mathbf{x}^*$ is the global minimum of the function. The minimum error is recorded when the Max_NFFEs is reached in 50 runs. The average and standard deviation of the error values are calculated as well.
- Number of Fitness Function Evaluations **(NFFEs)**: The NFFEs is also recorded when the VTR is reached. The average and standard deviation of the NFFEs values are calculated.
- **Successful Rate (S_r)**: A successful run of an algorithm indicates that the algorithm can result in a function value no worse than the VTR before the Max_NFFEs condition terminates the trial. The successful rate S_r is calculated as the number of successful runs divided by the total number of runs.
- **Convergence graphs**: The convergence graphs show the *median error* performance of the best solution over the total runs, in the respective experiments.

4.3. Comparison on Different Credit Assignment Methods

In this section, the performance of different credit assignment methods described in Section 3.2 is compared. They are referred to as being different AdapSS-JADE variants, as follows:

1) AdapSS-JADE1: AdapSS-JADE with the averaged absolute reward as shown in (12).
2) AdapSS-JADE2: AdapSS-JADE with the averaged normalized reward as shown in (13).
3) AdapSS-JADE3: AdapSS-JADE with the extreme absolute reward as shown in (14).
4) AdapSS-JADE4: AdapSS-JADE with the extreme normalized reward as shown in (15).

The results are shown in Table 1 for the *Adaptive Pursuit* technique[4], all of them being averaged over 50 independent runs. The same kind of conclusions can be gathered from these results, for the AP technique, as follows: (i) all four credit assignment methods were able to provide very similar averaged successful rates; (ii) JADE with the second credit assignment method, *i.e.*, the normalized average reward, obtained the best performance in terms of the averaged ranking. This latter confirms the assumptions listed in Section 3.2, showing that the normalization indeed increases the robustness of the algorithm when tackling very different problems as the ones used in this work.

For the sake of simplicity, in the following section we only use the normalized average reward as the credit assignment method, varying just the strategy selection scheme.

4.4. Comparison on Different Strategy Selection Methods

In order to compare the performance of different strategy selection techniques, the following five JADE variants are considered:

1) Uniform-JADE: JADE with the uniform strategy selection is implemented as baseline: for the creation of each offspring, a strategy is uniformly drawn from the pool.
2) SJADE: In this approach, the strategy adaptation technique proposed in [30] is used.
3) EPS-JADE: The strategy adaptation method presented in EPSDE [20] is used in EPS-JADE. Since the parameter adaptation of CR and F is implemented in JADE, in EPS-JADE the parameter adaptation method originally proposed in EPSDE is not used.

[4]For the sake of the brevity, we omit the results of the *Probability Matching* technique. Interested reader can contact the first author for more details.

Table 1: Comparison on the performance of different credit assignment methods with *Adaptive Pursuit* for all functions at $D = 30$. "Rank" indicates the ranking of the corresponding algorithm, obtained based on the mean values shown in the table.

F	AP-AdapSS-JADE-1	AP-AdapSS-JADE-2	AP-AdapSS-JADE-3	AP-AdapSS-JADE-4
	Mean ± Std (S_r) / Rank	Mean ± Std (S_r) / Rank	Mean ± Std (S_r) / Rank	Mean ± Std (S_r) / Rank
f_{01}	2.46E+04 ± 1.05E+03 (1.0) / 2	**2.46E+04 ± 9.75E+02** (1.0) / 1	2.68E+04 ± 1.03E+03 (1.0) / 4	2.65E+04 ± 1.28E+03 (1.0) / 3
f_{02}	4.13E+04 ± 2.47E+03 (1.0) / 2	**4.01E+04 ± 1.96E+03** (1.0) / 1	4.44E+04 ± 2.63E+03 (1.0) / 4	4.35E+04 ± 2.93E+03 (1.0) / 3
f_{03}	**8.83E+04 ± 4.77E+03** (1.0) / 1	8.88E+04 ± 5.95E+03 (1.0) / 2	9.18E+04 ± 8.47E+03 (1.0) / 4	8.95E+04 ± 8.67E+03 (1.0) / 3
f_{04}	1.92E+05 ± 9.90E+03 (1.0) / 2	**1.85E+05 ± 1.05E+04** (1.0) / 1	2.20E+05 ± 1.81E+04 (1.0) / 3	2.34E+05 ± 2.12E+04 (1.0) / 4
f_{05}	1.28E+05 ± 6.68E+03 (.94) / 2	**1.26E+05 ± 6.28E+03** (.92) / 1	1.29E+05 ± 1.21E+04 (.92) / 4	1.28E+05 ± 1.33E+04 (.94) / 3
f_{06}	**9.41E+03 ± 3.46E+02** (1.0) / 1	9.47E+03 ± 3.76E+02 (1.0) / 2	1.00E+04 ± 6.15E+02 (1.0) / 3	1.01E+04 ± 5.75E+02 (1.0) / 4
f_{07}	**9.41E+03 ± 3.46E+02** (1.0) / 1	9.47E+03 ± 3.76E+02 (1.0) / 2	1.00E+04 ± 6.15E+02 (1.0) / 3	1.01E+04 ± 5.75E+02 (1.0) / 4
f_{08}	9.64E+04 ± 3.27E+03 (1.0) / 2	**9.37E+04 ± 4.06E+03** (1.0) / 1	9.64E+04 ± 6.05E+03 (1.0) / 3	9.79E+04 ± 7.34E+03 (1.0) / 4
f_{09}	1.25E+05 ± 3.22E+03 (1.0) / 3	**1.23E+05 ± 4.43E+03** (1.0) / 1	1.24E+05 ± 6.16E+03 (1.0) / 2	1.25E+05 ± 7.14E+03 (1.0) / 4
f_{10}	**3.76E+04 ± 1.40E+03** (1.0) / 1	3.76E+04 ± 1.77E+03 (1.0) / 2	4.11E+04 ± 1.92E+03 (1.0) / 4	4.09E+04 ± 2.19E+03 (1.0) / 3
f_{11}	**2.59E+04 ± 1.46E+03** (1.0) / 1	2.60E+04 ± 1.27E+03 (1.0) / 2	2.75E+04 ± 1.35E+03 (1.0) / 3	2.82E+04 ± 2.82E+03 (1.0) / 4
f_{12}	2.21E+04 ± 1.11E+03 (1.0) / 2	**2.17E+04 ± 9.74E+02** (1.0) / 1	2.39E+04 ± 1.53E+03 (1.0) / 3	2.39E+04 ± 1.54E+03 (1.0) / 4
f_{13}	2.62E+04 ± 1.64E+03 (1.0) / 2	**2.56E+04 ± 1.34E+03** (1.0) / 1	2.90E+04 ± 2.00E+03 (1.0) / 4	2.87E+04 ± 1.94E+03 (1.0) / 3
F_{06}	1.12E+05 ± 1.01E+04 (.94) / 3	1.11E+05 ± 8.56E+03 (.88) / 2	1.13E+05 ± 1.29E+04 (.90) / 4	**1.10E+05 ± 9.70E+03** (.94) / 1
F_{07}	3.64E+04 ± 6.32E+03 (.76) / 2	**3.57E+04 ± 5.72E+03** (.68) / 1	3.81E+04 ± 6.83E+03 (.84) / 4	3.81E+04 ± 6.45E+03 (.86) / 3
$F_{08}^{\star}$	2.10E+01 ± 4.54E-02 (0.0) / 4	**2.09E+01 ± 4.76E-02** (0.0) / 1	2.09E+01 ± 5.49E-02 (0.0) / 2	2.10E+01 ± 5.74E-02 (0.0) / 3
F_{09}	1.05E+05 ± 2.67E+03 (1.0) / 4	**1.02E+05 ± 5.17E+03** (1.0) / 1	1.02E+05 ± 6.33E+03 (1.0) / 2	1.02E+05 ± 8.03E+03 (1.0) / 3
$F_{10}^{\star}$	2.60E+01 ± 4.81E+00 (0.0) / 3	2.92E+01 ± 6.61E+00 (0.0) / 4	2.56E+01 ± 4.17E+00 (0.0) / 2	**2.52E+01 ± 4.66E+00** (0.0) / 1
$F_{11}^{\star}$	2.58E+01 ± 1.65E+00 (0.0) / 2	**2.56E+01 ± 2.11E+00** (0.0) / 1	2.62E+01 ± 2.01E+00 (0.0) / 3	2.63E+01 ± 1.36E+00 (0.0) / 4
$F_{12}^{\star}$	**1.12E+03 ± 1.48E+03** (.10) / 1	1.96E+03 ± 1.81E+03 (.04) / 2	2.11E+03 ± 3.31E+03 (.14) / 3	2.33E+03 ± 2.86E+03 (.08) / 4
$F_{13}^{\star}$	**2.18E+00 ± 1.72E-01** (0.0) / 1	2.19E+00 ± 1.83E-01 (0.0) / 2	2.21E+00 ± 1.76E-01 (0.0) / 3	2.23E+00 ± 1.62E-01 (0.0) / 4
$F_{14}^{\star}$	1.23E+01 ± 2.81E-01 (0.0) / 3	1.23E+01 ± 2.79E-01 (0.0) / 2	**1.22E+01 ± 2.89E-01** (0.0) / 1	1.23E+01 ± 2.98E-01 (0.0) / 4
Avg.	(S_r) / Rank	(S_r) / Rank	(S_r) / Rank	(S_r) / Rank
	(.72) / 2.04	(.71) / 1.54	(.72) / 3.09	(.72) / 3.31

* indicates that the error values of the final solutions are used, since the successful rates for all methods are lower than 50%.

4) PM-AdapSS-JADE: AdapSS-JADE with the *Probability Matching* strategy selection technique and the normalized average credit assignment scheme.

5) AP-AdapSS-JADE: AdapSS-JADE with the *Adaptive Pursuit* strategy selection technique and the normalized average credit assignment scheme.

The parameters of all algorithms are used as mentioned in Section 4.1. The results are tabulated in Tables 2, 3, and 4. In Tables 2 and 3, the paired Wilcoxon signed-rank test at $\alpha = 0.05$ is adopted to compare the significance between two algorithms. The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test, which can be used as an alternative to the paired *t*-test when the results cannot be assumed to be normally distributed [34]. In Table 3, according to the Wilcoxon's test, the results are summarized as "$w/t/l$", which means that the algorithm in the row wins in w functions, ties in t functions, and loses in l functions, compared with the algorithm in the column. Similar to the methods used in [50], the *intermediate* results are reported for the functions where several algorithms can obtain the global optimum within Max_NFFEs. In these cases, the Wilcoxon signed-rank test considers only these intermediate results. In Table 4, the best and the second best results are highlighted, respectively, in **grey boldface** and **boldface**.

With respect to the quality of the final results, from Tables 2 and 3, we can see that:

- Compared with the baseline, *i.e.*, Uniform-JADE, SJADE showed to be competitive, with the Wilcoxon's test resulting in $6/9/7$. EPS-JADE obtains similar results compared with Uniform-JADE, the Wilcoxon's test resulting in $5/12/5$. Both PM-AdapSS-JADE and AP-AdapSS-JADE approaches are significantly better than Uniform-JADE on most of the functions. On 11 test functions PM-AdapSS-JADE significantly outperforms Uniform-JADE, while on the other 11 functions there are no significant differences between these two algorithms. Uniform-JADE is significantly outperformed by AP-AdapSS-JADE on 12 functions. On 9 functions AP-AdapSS-JADE provides the similar results to Uniform-JADE. Uniform-JADE only significantly outperforms AP-AdapSS-JADE on function F_{13}.

- According to the Wilcoxon's test shown in Table 3, it can be seen that both PM-AdapSS-JADE and AP-AdapSS-JADE approaches significantly outperform SJADE on most of the test functions. PM-AdapSS-JADE is significantly better than SJADE on 12 functions. AP-AdapSS-JADE is significantly better than SJADE on 11 func-

Table 2: Comparison on the **Error** values of different strategy selection techniques for all functions at $D = 30$. When several algorithms can obtain the global optimum for a function, only the *intermediate* results for the function are reported, hereinafter.

F	NFFEs	Uniform-JADE	SJADE	EPS-JADE	PM-AdapSS-JADE	AP-AdapSS-JADE
f_{01}	150k	2.33E-60 ± 1.45E-59 (1.0)	1.48E-60 ± 6.61E-60 (1.0)	5.59E-59 ± 3.91E-58 (1.0)	1.66E-62 ± 9.17E-62 (1.0)	**2.46E-75** ± **1.42E-74** (1.0)
f_{02}	200k	3.51E-36 ± 1.63E-35 (1.0)	1.24E-28 ± 8.38E-28 (1.0)	1.40E-32 ± 8.16E-32 (1.0)	2.97E-34 ± 2.09E-33 (1.0)	**1.85E-44** ± **1.31E-43** (1.0)
f_{03}	500k	4.33E-60 ± 1.81E-59 (1.0)	**7.34E-70** ± **2.23E-69** (1.0)	2.15E-66 ± 1.52E-65 (1.0)	1.79E-67 ± 1.20E-66 (1.0)	2.50E-68 ± 8.35E-68 (1.0)
f_{04}	500k	4.79E-16 ± 2.35E-16 (1.0)	5.27E-15 ± 3.67E-15 (1.0)	1.07E-15 ± 4.91E-15 (1.0)	2.74E-16 ± 1.67E-16 (1.0)	5.14E-22 ± 5.40E-22 (1.0)
f_{05}	500k	2.39E-01 ± 9.56E-01 (.94)	**7.75E-30** ± **2.63E-29** (1.0)	1.59E-01 ± 7.89E-01 (.96)	7.97E-02 ± 5.64E-01 (.98)	3.19E-01 ± 1.09E+00 (.92)
f_{06}	50k	1.38E+00 ± 1.15E+00 (1.0)	1.78E+00 ± 1.04E+00 (1.0)	1.12E+00 ± 8.63E-01 (1.0)	1.12E+00 ± 1.05E+00 (1.0)	**4.00E-02** ± **1.96E-01** (1.0)
f_{07}	300k	5.53E-04 ± 1.52E-04 (1.0)	5.82E-04 ± 2.79E-04 (1.0)	**5.21E-04** ± **2.26E-04** (1.0)	5.28E-04 ± 1.74E-04 (1.0)	5.94E-04 ± 1.89E-04 (1.0)
f_{08}	100k	2.51E-07 ± 7.24E-07 (1.0)	1.51E-07 ± 2.04E-07 (1.0)	4.23E-06 ± 2.09E-06 (1.0)	1.29E-07 ± 1.89E-07 (1.0)	**1.82E-08** ± **1.19E-07** (1.0)
f_{09}	100k	3.56E-01 ± 3.06E-01 (1.0)	**9.09E-02** ± **7.06E-02** (1.0)	6.68E-01 ± 3.30E-01 (1.0)	3.02E-01 ± 4.52E-01 (1.0)	2.95E-01 ± 5.69E-01 (1.0)
f_{10}	50k	3.15E-10 ± 3.49E-10 (1.0)	5.56E-10 ± 6.45E-10 (1.0)	3.03E-10 ± 2.77E-10 (1.0)	1.99E-10 ± 2.01E-10 (1.0)	**1.10E-11** ± **1.86E-11** (1.0)
f_{11}	50k	1.72E-10 ± 1.21E-09 (1.0)	1.78E-17 ± 1.10E-16 (1.0)	3.93E-16 ± 2.75E-15 (1.0)	6.66E-18 ± 4.66E-17 (1.0)	**0.00E+00** ± **0.00E+00** (1.0)
f_{12}	50k	5.96E-19 ± 2.14E-18 (1.0)	1.59E-18 ± 2.59E-18 (1.0)	7.49E-19 ± 2.00E-18 (1.0)	2.39E-19 ± 7.64E-19 (1.0)	**2.24E-22** ± **7.79E-22** (1.0)
f_{13}	50k	1.91E-16 ± 3.58E-16 (1.0)	1.72E-14 ± 8.15E-14 (1.0)	7.30E-16 ± 2.34E-15 (1.0)	1.77E-16 ± 3.70E-16 (1.0)	**3.76E-20** ± **1.21E-19** (1.0)
F_{06}	300k	3.99E-01 ± 1.21E+00 (.90)	3.90E+00 ± 1.52E+01 (.88)	7.36E-01 ± 3.61E+00 (.92)	**6.92E-25** ± **4.22E-24** (1.0)	4.78E-01 ± 1.31E+00 (.88)
F_{07}	300k	1.13E-02 ± 1.03E-02 (.74)	1.08E-02 ± 9.04E-03 (.72)	**8.57E-03** ± **8.00E-03** (.84)	9.01E-03 ± 8.65E-03 (.82)	1.31E-02 ± 1.04E-02 (.68)
F_{08}	300k	2.09E+01 ± 6.79E-02 (0.0)	2.09E+01 ± 5.99E-02 (0.0)	2.09E+01 ± 5.66E-02 (0.0)	2.09E+01 ± 4.96E-02 (0.0)	2.09E+01 ± 4.76E-02 (0.0)
F_{09}	100k	2.69E-01 ± 2.61E-01 (1.0)	**6.54E-02** ± **7.23E-02** (1.0)	8.25E-01 ± 4.62E-01 (1.0)	1.83E-01 ± 1.65E-01 (1.0)	1.65E-01 ± 2.54E-01 (1.0)
F_{10}	300k	2.87E+01 ± 5.65E+00 (0.0)	3.01E+01 ± 5.02E+00 (0.0)	2.60E+01 ± 4.99E+00 (0.0)	**2.57E+01** ± **4.26E+00** (0.0)	2.92E+01 ± 6.61E+00 (0.0)
F_{11}	300k	2.83E+01 ± 1.36E+00 (0.0)	2.61E+01 ± 1.18E+00 (0.0)	2.66E+01 ± 1.38E+00 (0.0)	**2.55E+01** ± **1.22E+00** (0.0)	2.56E+01 ± 2.11E+00 (0.0)
F_{12}	300k	**1.61E+03** ± **1.98E+03** (.04)	5.44E+03 ± 5.09E+03 (.02)	3.21E+03 ± 3.69E+03 (.14)	1.74E+03 ± 1.92E+03 (.04)	1.96E+03 ± 1.81E+03 (.04)
F_{13}	300k	2.09E+00 ± 1.68E-01 (0.0)	2.45E+00 ± 2.38E-01 (0.0)	**1.35E+00** ± **1.03E-01** (0.0)	2.06E+00 ± 1.73E-01 (0.0)	2.19E+00 ± 1.83E-01 (0.0)
F_{14}	300k	1.23E+01 ± 2.94E-01 (0.0)	1.23E+01 ± 2.74E-01 (0.0)	1.23E+01 ± 2.39E-01 (0.0)	1.23E+01 ± 2.75E-01 (0.0)	**1.23E+01** ± **2.79E-01** (0.0)

Table 3: Wilcoxon's test on the **Error** values of different strategy selection techniques for all functions at $D = 30$. The algorithm in the row is compared with the algorithm in the column. The results are described as "wins/ties/losses".

	Uniform-JADE	SJADE	EPS-JADE	PM-AdapSS-JADE	AP-AdapSS-JADE
Uniform-JADE	–	–	–	–	–
SJADE	7/9/6	–	–	–	–
EPS-JADE	5/12/5	10/8/4	–	–	–
PM-AdapSS-JADE	11/11/0	12/8/2	9/11/2	–	–
AP-AdapSS-JADE	12/9/1	11/9/2	11/6/5	11/9/2	–

tions. SJADE significantly dominates PM-AdapSS-JADE and AP-AdapSS-JADE on two functions (f_{09} and F_{09})[5]. On the rest of the functions, there are no significant differences in terms of error values.

- Considering EPS-JADE, it obtains overall better results than SJADE method. Compared EPS-JADE with PM-AdapSS-JADE and AP-AdapSS-JADE, we can see that PM-AdapSS-JADE and AP-AdapSS-JADE are significantly better than EPS-JADE in most of the cases.

- Comparison on the results between PM-AdapSS-JADE and AP-AdapSS-JADE, the Wilcoxon's test result is $11/9/2$. It means that AP-AdapSS-JADE is significantly better than PM-AdapSS-JADE on 11 functions. On 9 functions, there are no significant differences between these two algorithms. Only on 2 functions (F_{10} and F_{13}), AP-AdapSS-JADE is significantly worse than PM-AdapSS-JADE. Thus, we can conclude that on most of the functions the *Adaptive Pursuit* based AdapSS-JADE approach is better than the *Probability Matching* based one, consequently being better than all the other methods used in this empirical comparison.

Generally, our proposed PM-AdapSS-JADE and AP-AdapSS-JADE approaches obtain better results than SJADE in terms of the error values and the convergence rate, which might indicate that the relative fitness improvement based credit assignment techniques are better than the method proposed in SaDE [30] (which is based on the frequency of fitness improvements). Moreover, both PM-AdapSS-JADE and AP-AdapSS-JADE approaches are better than Uniform-JADE and EPS-JADE, which means that the *Probability Matching* and *Adaptive Pursuit* techniques based AdapSS-JADE are able of efficiently adjust the probability of the most suitable strategy while solving the problem. With respect to the convergence rate, Table 4 shows that PM-AdapSS-JADE and AP-AdapSS-JADE consistently converge faster than Uniform-JADE, SJADE, and EPS-JADE on most of the functions. In addition, AP-AdapSS-JADE is capable of providing the fastest convergence rate compared with the other three methods on most of the

[5]These two functions are the Rastrigin's functions. F_{09} is the shifted version of f_{09}.

Table 4: Comparison on the **NFFEs** values of different strategy selection techniques for the successful functions at $D = 30$. The results are only reported for functions that are solved successfully within the Max_NFFEs.

F	Uniform-JADE	SJADE	EPS-JADE	PM-AdapSS-JADE	AP-AdapSS-JADE
f_{01}	2.77E+04 ± 9.04E+02	2.82E+04 ± 8.13E+02	2.77E+04 ± 8.34E+02	**2.74E+04 ± 6.00E+02**	**2.46E+04 ± 9.75E+02**
f_{02}	4.73E+04 ± 1.84E+03	4.88E+04 ± 2.16E+03	4.72E+04 ± 1.65E+03	**4.63E+04 ± 1.88E+03**	**4.01E+04 ± 1.96E+03**
f_{03}	9.02E+04 ± 6.35E+03	8.86E+04 ± 4.24E+03	**8.30E+04 ± 4.96E+03**	**8.79E+04 ± 3.49E+03**	8.88E+04 ± 5.95E+03
f_{04}	2.65E+05 ± 6.40E+03	2.85E+05 ± 7.66E+03	2.75E+05 ± 3.57E+03	**2.61E+05 ± 6.58E+03**	**1.85E+05 ± 1.05E+04**
f_{05}	1.31E+05 ± 1.02E+04	1.26E+05 ± 4.99E+03	**1.19E+05 ± 5.04E+03**	**1.25E+05 ± 3.71E+03**	1.26E+05 ± 6.28E+03
f_{06}	1.03E+04 ± 3.16E+02	1.04E+04 ± 2.93E+02	1.03E+04 ± 2.95E+02	**1.02E+04 ± 3.30E+02**	**9.47E+03 ± 3.76E+02**
f_{07}	2.31E+04 ± 5.82E+03	2.59E+04 ± 6.02E+03	**2.28E+04 ± 4.91E+03**	**2.31E+04 ± 5.18E+03**	2.33E+04 ± 5.74E+03
f_{08}	1.03E+05 ± 2.98E+03	1.04E+05 ± 2.30E+03	1.18E+05 ± 1.88E+03	**1.03E+05 ± 2.42E+03**	**9.37E+04 ± 4.06E+03**
f_{09}	1.30E+05 ± 2.36E+03	**1.29E+05 ± 2.25E+03**	1.39E+05 ± 1.93E+03	1.30E+05 ± 5.04E+03	**1.23E+05 ± 4.43E+03**
f_{10}	4.29E+04 ± 1.40E+03	4.38E+04 ± 1.47E+03	4.29E+04 ± 1.26E+03	**4.23E+04 ± 1.29E+03**	**3.76E+04 ± 1.77E+03**
f_{11}	4.26E+04 ± 3.06E+03	2.97E+04 ± 1.19E+03	2.95E+04 ± 1.29E+03	**2.89E+04 ± 1.19E+03**	**2.60E+04 ± 1.27E+03**
f_{12}	2.51E+04 ± 1.01E+03	2.59E+04 ± 9.15E+02	2.52E+04 ± 8.50E+02	**2.45E+04 ± 1.04E+03**	**2.17E+04 ± 9.74E+02**
f_{13}	3.05E+04 ± 1.26E+03	3.16E+04 ± 1.91E+03	3.03E+04 ± 1.41E+03	**3.03E+04 ± 1.32E+03**	**2.56E+04 ± 1.34E+03**
F_{06}	1.22E+05 ± 1.97E+04	**1.09E+05 ± 6.49E+03**	1.12E+05 ± 1.51E+04	1.15E+05 ± 1.19E+04	**1.11E+05 ± 8.56E+03**
F_{07}	3.64E+04 ± 4.33E+03	3.66E+04 ± 4.71E+03	3.60E+04 ± 5.11E+03	**3.48E+04 ± 4.75E+03**	**3.57E+04 ± 5.72E+03**
F_{09}	1.06E+05 ± 2.18E+03	1.03E+05 ± 2.32E+03	1.12E+05 ± 1.38E+03	1.05E+05 ± 2.83E+03	**1.02E+05 ± 5.17E+03**

functions, while also showing to be better in terms of the error values; the reason for this is that the *Adaptive Pursuit* technique is able to converge more rapidly to a strategy probability distribution that accurately reflects the quality empirical estimates, confirming what was already shown in the context of GAs [41].

For the experiments presented in the following sections, we consider only the results of the best method found here, *i.e.*, the AP-AdapSS-JADE using the normalized average reward of relative fitness improvements.

4.5. Analysis of Strategy Adaptation

The adaptation characteristics of the PM method have been analyzed in [15], with the results showing that PM-based DE is able to efficiently select the most suitable strategy while solving a given problem, globally achieving better results than the baseline adaptive methods, and than the DE using a single strategy for each of the strategies constituting the pool. In this section, the adaptation characteristics of the AP method are analyzed experimentally. Four variants of JADE are compared with AP-AdapSS-JADE, each of them using one of the strategies that constitute the pool, as follows:

1) JADE-wo: JADE with the strategy shown in Equation 16.

2) JADE-w: JADE with the strategy shown in Equation 17.

3) rJADE-wo: JADE with the strategy shown in Equation 18.

4) rJADE-w: JADE with the strategy shown in Equation 19.

The parameter settings are kept the same as described in Section 4.1. The results are shown in Table 5. In Table 5, only the intermediate results are reported for the functions where several algorithms can obtain the global optimum within the Max_NFFEs. In the last row of Table 5, according to the Wilcoxon's test, the results are summarized as "$w/t/l$", which means that AP-AdapSS-JADE wins in w functions, ties in t functions, and loses in l functions, compared with its competitors. Some typical convergence curves and the evolution of the probabilities of the selected functions are plotted in Figure 2. All results are averaged over 50 independent runs.

4.5.1. On the General Performance

From Table 5, it is clear to see that on most of the functions AP-AdapSS-JADE significantly outperforms JADE with each single strategy in the pool. AP-AdapSS-JADE is significantly better than JADE-wo, JADE-w, rJADE-wo, and rJADE-w on 16, 12, 14, and 14 out of 22 functions, respectively, while being outperformed on 0, 2, 4, and 5 functions.

Considering the convergence rate, from Figure 2, we can observe that AP-AdapSS-JADE requires less NFFEs to achieve the value-to-reach and converges faster than the four JADE variants on most of the functions.

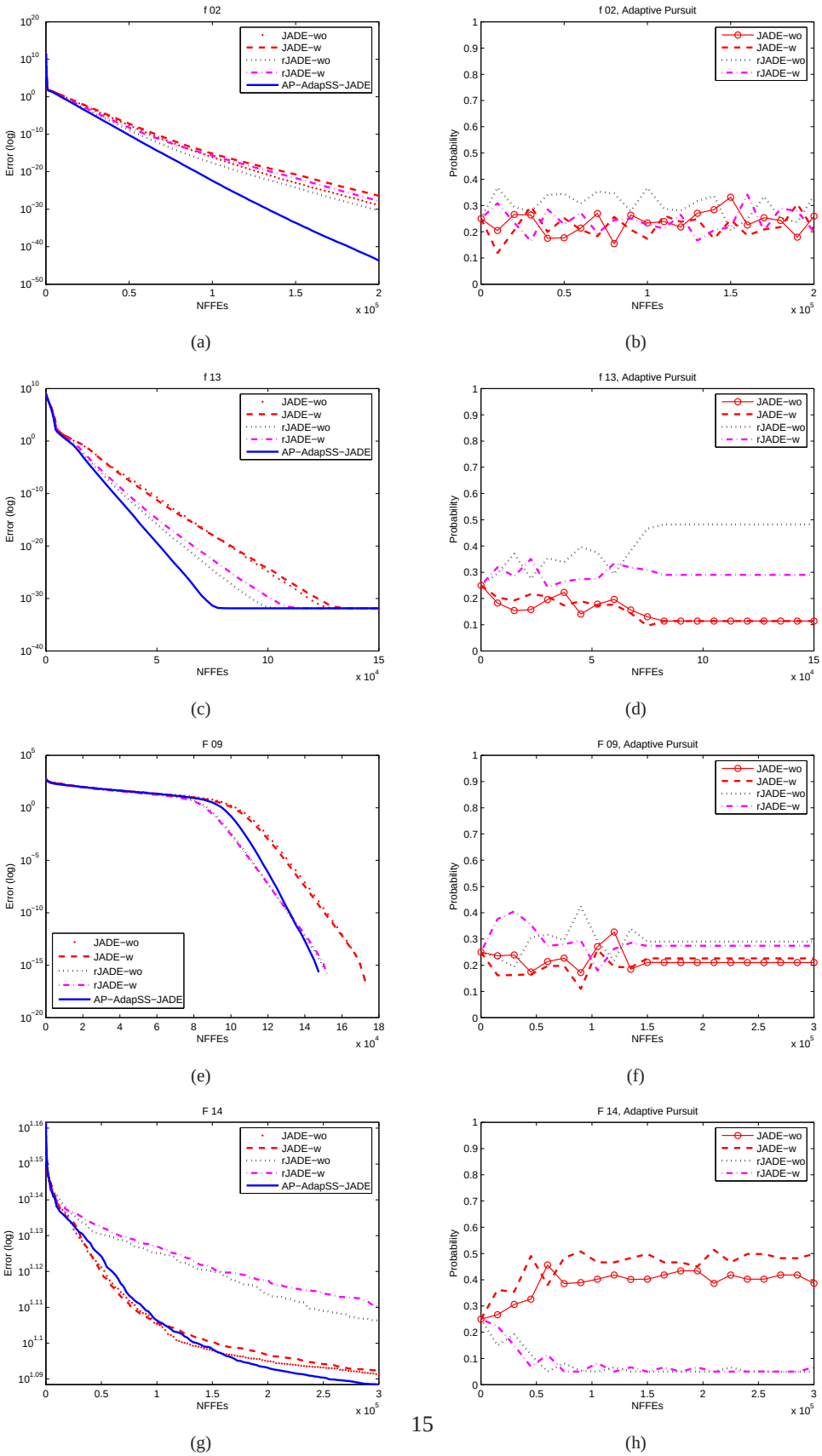

Figure 2: Adaptation analysis of AP-AdapSS-JADE on the selected functions. The convergence graphs of these functions are shown in the left column, and the evolution of the probabilities of each strategy is shown in the right column. (a, b) f_{01}; (c, d) f_{06}; (e, f) F_{09}; (g, h) F_{14}.

Table 5: Comparison on the **Error** values between JADE with single strategy and AP-AdapSS-JADE for all functions at $D = 30$.

F	NFFEs	JADE-wo	JADE-w	rJADE-wo	rJADE-w	AP-AdapSS-JADE
f_{01}	150k	5.06E-59 ± 3.18E-58 (1.0)†	7.14E-58 ± 3.36E-57 (1.0)†	2.06E-50 ± 1.46E-49 (1.0)†	1.89E-53 ± 1.33E-52 (1.0)†	**2.46E-75 ± 1.42E-74** (1.0)
f_{02}	200k	1.32E-29 ± 5.99E-29 (1.0)†	3.64E-27 ± 2.51E-26 (1.0)†	4.52E-31 ± 2.24E-30 (1.0)†	1.63E-28 ± 6.12E-28 (1.0)†	**1.85E-44 ± 1.31E-43** (1.0)
f_{03}	500k	1.71E-60 ± 6.68E-60 (1.0)†	**9.28E-80 ± 5.62E-79** (1.0)‡	1.64E+00 ± 5.30E+00 (.88)†	1.69E+00 ± 3.18E+00 (.76)†	2.50E-68 ± 8.35E-68 (1.0)
f_{04}	500k	3.17E-14 ± 1.53E-14 (1.0)†	5.02E-14 ± 3.45E-14 (1.0)†	5.42E-16 ± 3.00E-16 (1.0)†	1.15E-15 ± 4.89E-16 (1.0)†	**5.14E-22 ± 5.40E-22** (1.0)
f_{05}	500k	1.59E-01 ± 7.89E-01 (.96)	1.59E-01 ± 7.89E-01 (.96)	7.97E-02 ± 5.64E-01 (.98)	**2.25E-30 ± 4.78E-30** (1.0)	3.19E-01 ± 1.09E+00 (.92)
f_{06}	10k	3.02E+00 ± 1.24E+00 (1.0)†	5.70E+00 ± 1.57E+00 (1.0)†	1.40E-01 ± 4.00E-01 (1.0)†	1.22E+00 ± 1.20E+00 (1.0)†	**4.00E-02 ± 1.96E-01** (1.0)
f_{07}	300k	6.57E-04 ± 2.51E-04 (1.0)†	6.05E-04 ± 2.06E-04 (1.0)	4.93E-04 ± 1.64E-04 (1.0)‡	**4.81E-04 ± 1.42E-04** (1.0)‡	5.94E-04 ± 1.89E-04 (1.0)
f_{08}	100k	1.71E-04 ± 2.27E-04 (1.0)†	2.60E-04 ± 4.77E-04 (1.0)†	**2.93E-09 ± 5.12E-09** (1.0)‡	4.22E-09 ± 4.75E-09 (1.0)‡	1.82E-08 ± 1.19E-07 (1.0)
f_{09}	100k	1.90E+00 ± 7.36E-01 (1.0)†	1.63E+00 ± 7.62E-01 (1.0)†	**6.85E-03 ± 7.25E-03** (1.0)‡	1.22E-02 ± 1.70E-02 (1.0)‡	2.95E-01 ± 5.69E-01 (1.0)
f_{10}	50k	1.14E-09 ± 1.20E-09 (1.0)†	2.91E-09 ± 2.89E-09 (1.0)†	1.24E-10 ± 1.41E-10 (1.0)†	3.54E-10 ± 2.79E-10 (1.0)†	**1.10E-11 ± 1.86E-11** (1.0)
f_{11}	25k	2.71E-04 ± 1.02E-03 (1.0)†	3.39E-04 ± 1.44E-03 (1.0)†	2.24E-07 ± 1.48E-07 (1.0)	1.19E-06 ± 1.27E-06 (1.0)†	**1.49E-07 ± 5.38E-07** (1.0)
f_{12}	50k	1.44E-17 ± 3.65E-17 (1.0)†	1.68E-16 ± 4.52E-16 (1.0)†	7.14E-20 ± 2.17E-19 (1.0)†	1.81E-18 ± 5.39E-18 (1.0)†	**2.24E-22 ± 7.79E-22** (1.0)
f_{13}	50k	1.80E-11 ± 1.09E-10 (1.0)†	5.87E-12 ± 2.40E-11 (1.0)†	1.34E-16 ± 7.27E-16 (1.0)†	1.53E-15 ± 4.80E-15 (1.0)†	**3.76E-20 ± 1.21E-19** (1.0)
F_{06}	300k	5.58E+00 ± 1.58E+01 (.70)†	4.26E+00 ± 1.57E+01 (.88)	5.27E+00 ± 1.83E+01 (.88)†	2.84E+00 ± 1.15E+01 (.90)	**4.78E-01 ± 1.31E+00** (.88)
F_{07}	300k	1.40E-02 ± 1.80E-02 (.68)	8.86E-03 ± 9.58E-03 (.74)‡	1.38E-02 ± 1.13E-02 (.74)	**6.16E-03 ± 6.25E-03** (.92)‡	1.31E-02 ± 1.04E-02 (.68)
F_{08}	300k	2.09E+01 ± 4.33E-02 (0.0)	2.09E+01 ± 4.50E-02 (0.0)	2.09E+01 ± 8.20E-02 (0.0)	2.09E+01 ± 6.56E-02 (0.0)	2.09E+01 ± 4.76E-02 (0.0)
F_{09}	100k	1.66E+00 ± 7.33E-01 (1.0)†	1.21E+00 ± 6.51E-01 (1.0)†	**4.02E-03 ± 6.17E-03** (1.0)‡	2.72E-03 ± 2.81E-03 (1.0)‡	1.65E-01 ± 2.54E-01 (1.0)
F_{10}	300k	3.14E+01 ± 6.77E+00 (0.0)†	**2.90E+01 ± 4.02E+00** (0.0)	4.19E+01 ± 2.38E+01 (0.0)†	4.00E+01 ± 2.46E+01 (0.0)†	2.92E+01 ± 6.61E+00 (0.0)
F_{11}	300k	**2.49E+01 ± 1.17E+00** (0.0)	2.59E+01 ± 1.77E+00 (0.0)	3.08E+01 ± 2.50E+00 (0.0)†	3.28E+01 ± 5.29E+00 (0.0)†	2.56E+01 ± 2.11E+00 (0.0)
F_{12}	300k	4.56E+03 ± 3.92E+03 (0.0)†	4.49E+03 ± 4.19E+03 (.02)†	1.44E+04 ± 1.13E+04 (0.0)†	1.08E+04 ± 8.94E+03 (0.0)†	**1.96E+03 ± 1.81E+03** (.04)
F_{13}	300k	2.22E+00 ± 1.87E-01 (0.0)	**2.19E+00 ± 1.94E-01** (0.0)	2.62E+00 ± 2.29E-01 (0.0)†	2.70E+00 ± 2.07E-01 (0.0)†	2.19E+00 ± 1.83E-01 (0.0)
F_{14}	300k	1.23E+01 ± 2.17E-01 (0.0)	1.24E+01 ± 2.09E-01 (0.0)	1.28E+01 ± 4.90E-01 (0.0)†	1.29E+01 ± 2.41E-01 (0.0)†	**1.23E+01 ± 2.79E-01** (0.0)
w/t/l		16/6/0	12/8/2	14/4/4	14/3/5	–

† indicates AP-AdapSS-JADE is significantly better than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.
‡ means that AP-AdapSS-JADE is significantly outperformed by its competitor.

4.5.2. On the Adaptation Characteristics

In order to analyze the adaptation characteristics of AP-AdapSS-JADE, the evolution trend of the probability of each strategy for some selected functions is shown in the right columns of Figure 2. According to the results shown in Table 5 and Figure 2, it can be observed that:

- Similar to memetic algorithms, the competition and cooperation [25, 26] can also be observed in AP-AdapSS-JADE. Different strategies of JADE are working together to accomplish the shared optimization goal and resulting in a higher achievement [26]. On most of the functions, AP-AdapSS-JADE obtains the best results compared with the four JADE variants. This means that the *Adaptive Pursuit* based AdapSS-JADE is capable of enhancing the performance of JADE w.r.t. quality of final solutions and convergence speed.

- Comparing the results of JADE with each single strategy, it is difficult to say which one is the best. For example, for function f_{08} rJADE-wo and rJADE-w provide better results than JADE-wo and JADE-w. Oppositely, for function F_{11}, JADE-wo and JADE-w are better than JADE-wo and JADE-w in terms of quality of final solutions and convergence rate. This phenomenon can also be observed for other test functions.

- Figure 2 indicates that AP-AdapSS-JADE is able to adaptively select the most suitable strategy while solving the problem at hand. The *Adaptive Pursuit* technique can converge rapidly to a strategy probability distribution that results in a much higher probability of selecting the current optimal strategy. For example, for function f_{13} rJADE-wo converges fastest, followed by rJADE-w. According to Figure 2 (d), we can observe that "DE/rand-to*p*best/1 (without archive)" obtains the highest probability almost in the whole evolution process, followed by "DE/rand-to*p*best/1 (with archive)". The probabilities of "DE/current-to*p*best/1 (without archive)" and "DE/current-to*p*best/1 (without archive)" are very small for this function. On the contrary, for function F_{14}, JADE-wo and JADE-w converge faster than rJADE-wo and rJADE-w. Thus, from Figure 2 (g) and (h) we can see that strategies "DE/current-to*p*best/1 (without archive)" and "DE/current-to*p*best/1 (without archive)" get higher probabilities than "DE/rand-to*p*best/1 (without archive)" and "DE/rand-to*p*best/1 (without archive)". By carefully looking at the results in Figure 2, it can be seen that for some functions the evolution trend of the probability of each strategy is oscillatory. For function f_{02}, the reason might be that this function is relatively simple. One of the four strategies may provide a higher reward than the others in the evolution process, hence being assigned with a higher probability.

Table 6: Comparison on the **Error** values between JADE with single strategy and AP-AdapSS-JADE for functions $f_{01} - f_{13}$ at $D = 100$ and for functions $F_{06} - F_{14}$ at $D = 50$.

$D = 100$						
F	NFFEs	JADE-wo	JADE-w	rJADE-wo	rJADE-w	AP-AdapSS-JADE
f_{01}	1000k	9.50E-62 ± 1.97E-61 (1.0)†	9.67E-84 ± 1.85E-83 (1.0)†	1.77E-83 ± 4.30E-83 (1.0)†	1.53E-96 ± 1.07E-96 (1.0)†	**2.41E-109 ± 6.62E-109** (1.0)
f_{02}	1000k	7.28E-36 ± 8.44E-36 (1.0)†	1.16E-41 ± 1.44E-41 (1.0)†	1.95E-46 ± 1.36E-45 (1.0)	**1.33E-54 ± 1.68E-54** (1.0)‡	4.20E-50 ± 9.92E-50 (1.0)
f_{03}	1000k	9.13E-04 ± 7.02E-04 (0.0)†	**3.91E-05 ± 4.62E-05** (0.0)‡	3.81E+04 ± 2.81E+04 (0.0)†	4.02E+04 ± 2.61E+04 (0.0)†	2.34E-04 ± 3.90E-04 (0.0)
f_{04}	1000k	8.08E-02 ± 1.06E-01 (0.0)†	7.10E-03 ± 8.53E-03 (0.0)	2.20E-02 ± 8.87E-02 (0.0)†	**1.40E-04 ± 2.68E-04** (0.0)‡	9.86E-03 ± 1.11E-03 (0.0)
f_{05}	1000k	4.96E+01 ± 1.22E+01 (0.0)†	3.08E+01 ± 1.36E+01 (0.0)†	4.51E+01 ± 2.26E+00 (0.0)†	**1.87E+01 ± 1.98E+00** (0.0)‡	2.91E+01 ± 7.99E+00 (0.0)
f_{06}	50k	2.93E+01 ± 4.68E+00 (1.0)†	2.57E+01 ± 3.78E+00 (1.0)†	**2.38E+00 ± 1.28E+00** (1.0)‡	5.86E+00 ± 1.97E+00 (1.0)†	4.90E+00 ± 2.57E+00 (1.0)
f_{07}	1000k	1.86E-03 ± 3.58E-04 (1.0)†	1.62E-03 ± 2.90E-04 (1.0)†	8.99E-04 ± 1.76E-04 (1.0)‡	**8.83E-04 ± 1.82E-04** (1.0)‡	1.20E-03 ± 2.58E-04 (1.0)
f_{08}	1000k	9.71E+03 ± 3.53E+02 (0.0)†	9.12E+03 ± 3.31E+02 (0.0)†	9.39E+03 ± 2.92E+02 (0.0)†	8.13E+03 ± 3.65E+02 (0.0)†	**7.74E+03 ± 3.42E+02** (0.0)
f_{09}	1000k	1.91E+02 ± 7.68E+00 (0.0)†	1.84E+02 ± 7.48E+00 (0.0)†	1.62E+02 ± 8.13E+00 (0.0)†	1.59E+02 ± 7.56E+00 (0.0)†	**1.54E+02 ± 6.61E+00** (0.0)
f_{10}	100k	1.40E-02 ± 2.54E-03 (1.0)†	6.64E-03 ± 1.21E-03 (1.0)†	**9.67E-04 ± 1.30E-04** (1.0)	1.09E-03 ± 1.53E-04 (1.0)	1.01E-03 ± 3.87E-04 (1.0)
f_{11}	100k	1.27E-02 ± 7.56E-03 (.92)†	2.50E-03 ± 1.90E-03 (.98)†	4.88E-05 ± 1.62E-05 (1.0)†	4.70E-05 ± 1.52E-05 (1.0)†	**1.13E-05 ± 1.05E-05** (1.0)
f_{12}	100k	5.27E-05 ± 2.24E-05 (1.0)†	1.46E-05 ± 6.74E-06 (1.0)†	**2.88E-07 ± 9.49E-08** (1.0)‡	3.08E-07 ± 9.00E-08 (1.0)†	3.01E-07 ± 5.38E-08 (1.0)
f_{13}	100k	4.32E-01 ± 1.22E+00 (1.0)†	3.04E-02 ± 6.44E-02 (1.0)†	**1.35E-04 ± 1.04E-04** (1.0)‡	2.38E-04 ± 5.38E-04 (1.0)†	2.09E-04 ± 1.89E-04 (1.0)
$D = 50$						
F	NFFEs	JADE-wo	JADE-w	rJADE-wo	rJADE-w	AP-AdapSS-JADE
F_{06}	500k	1.60E+00 ± 1.99E+00 (.56)†	4.78E-01 ± 1.32E+00 (.88)	6.38E-01 ± 1.49E+00 (.84)†	**3.19E-01 ± 1.10E+00** (.92)‡	**3.19E-01 ± 1.10E+00** (.92)
F_{07}	500k	4.04E-03 ± 5.68E-03 (.88)†	6.20E-03 ± 1.12E-02 (.76)†	1.77E-03 ± 4.18E-03 (.92)†	**8.87E-04 ± 3.09E-03** (.92)	9.85E-04 ± 3.48E-03 (.92)
F_{08}	500k	2.11E+01 ± 3.82E-02 (0.0)	2.11E+01 ± 4.90E-02 (0.0)	2.11E+01 ± 3.51E-02 (0.0)	2.11E+01 ± 3.77E-02 (0.0)	2.11E+01 ± 3.27E-02 (0.0)
F_{09}	500k	1.62E-01 ± 1.62E-01 (.02)†	1.56E-02 ± 1.87E-02 (.68)†	5.62E-10 ± 9.73E-10 (1.0)†	1.88E-12 ± 3.06E-12 (1.0)†	**4.84E-13 ± 1.13E-12** (1.0)
F_{10}	500k	1.04E+02 ± 9.08E+00 (0.0)†	9.76E+01 ± 8.32E+00 (0.0)†	**8.40E+01 ± 7.77E+01** (0.0)	9.46E+01 ± 8.20E+01 (0.0)	8.65E+01 ± 1.14E+01 (0.0)
F_{11}	500k	**5.26E+01 ± 1.94E+00** (0.0)‡	5.32E+01 ± 1.49E+00 (0.0)	6.25E+01 ± 4.64E+00 (0.0)†	6.25E+01 ± 4.64E+00 (0.0)†	5.39E+01 ± 1.66E+00 (0.0)
F_{12}	500k	1.22E+04 ± 1.89E+04 (0.0)	1.04E+04 ± 1.66E+04 (0.0)	5.89E+04 ± 5.30E+04 (0.0)†	5.52E+04 ± 5.25E+04 (0.0)†	**7.05E+03 ± 8.22E+03** (0.0)
F_{13}	500k	7.72E+00 ± 3.99E-01 (0.0)†	7.87E+00 ± 3.39E-01 (0.0)†	7.87E+00 ± 4.79E-01 (0.0)†	8.13E+00 ± 4.38E-01 (0.0)†	**7.38E+00 ± 5.33E-01** (0.0)
F_{14}	500k	2.18E+01 ± 2.69E-01 (0.0)	2.17E+01 ± 2.47E-01 (0.0)	2.27E+01 ± 1.99E-01 (0.0)†	2.28E+01 ± 1.59E-01 (0.0)†	**2.16E+01 ± 3.29E-01** (0.0)
$w/t/l$		18/3/1	15/6/1	14/4/4	13/4/5	–

† indicates AP-AdapSS-JADE is significantly better than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.
‡ means that AP-AdapSS-JADE is significantly outperformed by its competitor.

In summary, from the above analysis we can conclude that our approach is able to efficiently select the suitable strategy for a specific problem. For each function, one of the strategies was empirically found to be the best. The AP-AdapSS-JADE was able to automatically select between them, without any external *a priori* knowledge, thus enhancing the performance of the JADE algorithm.

4.6. Scalability Study

In order to better understand the performance of our approach, in this section, the scalability study is conducted. For functions $f_{01} - f_{13}$, the dimensions are scaled at $D = 100, 200, 500$. While for functions $F_{06} - F_{14}$, $D = 50$ is used, since these functions are defined up to $D = 50$ in [37].

4.6.1. Performance on Moderate-Dimensional Problems

In this section, AP-AdapSS-JADE is compared again with the four JADE variants for all functions considering, but considering a higher dimensionality, as done in the original reference of the baseline JADE method [50]. For functions $f_{01} - f_{13}$, $D = 100$ is considered. For functions $F_{06} - F_{14}$, $D = 50$ is used, as in [37]. The population size $NP = 400$, Max_NFFEs $= D \times 10,000$. All other parameters are kept unchanged as described in Section 4.1. The results are reported in Table 6. From the results shown in these two tables, we can observe that:

- When the dimensionality of the problems increase, their complexity increase consistently, especially for the multimodal functions. Thus, the overall successful rate of each algorithm decreases for higher dimensional problems. For example, the overall S_r of JADE-wo is 9.38, and AP-AdapSS-JADE obtains the overall $S_r = 10.84$.

- For higher dimensional problems, a higher population diversity is required and, hence, the strategy which provides higher perturbation is able to obtain better performance. This is verified in Table 6, where JADE with archive (JADE-w and rJADE-w) is better than JADE without archive (JADE-wo and rJADE-wo); JADE with rand-to-*p*best/1 mutation (rJADE-wo and rJADE-w) obtains better performance than JADE with current-to-*p*best/1 mutation (JADE-wo and JADE-w).

Table 7: Comparison on the **Error** values among different JADE variants for functions $f_{01} - f_{13}$ at $D = 200$.

F	JADE-w	rJADE-w	SJADE	EPS-JADE	SaJADE	AP-AdapSS-JADE
f_{01}	3.95E-45 ± 7.65E-45†	**5.47E-65 ± 9.91E-65**‡	2.96E-54 ± 5.17E-54†	1.04E-52 ± 1.54E-52†	2.14E-63 ± 3.88E-63†	**4.87E-64 ± 6.00E-64**
f_{02}	1.24E-17 ± 3.30E-17†	**1.15E-25 ± 3.87E-25**‡	3.09E-21 ± 1.75E-20†	1.19E-20 ± 2.70E-20†	2.29E-25 ± 6.79E-25	**1.89E-25 ± 5.43E-25**
f_{03}	**1.26E+00 ± 4.36E-01**‡	3.47E+05 ± 1.07E+05†	2.80E+00 ± 9.57E-01†	6.43E+00 ± 5.34E+00†	**1.35E+00 ± 4.81E-01**‡	2.53E+00 ± 9.91E-01
f_{04}	7.23E+00 ± 6.14E-01†	**5.54E+00 ± 5.17E-01**‡	6.28E+00 ± 5.55E-01†	6.18E+00 ± 5.80E-01†	6.09E+00 ± 6.30E-01	**5.98E+00 ± 7.05E-01**
f_{05}	2.04E+02 ± 4.64E+01†	**1.49E+02 ± 1.53E+01**	1.82E+02 ± 2.98E+01†	1.84E+02 ± 3.45E+01†	1.65E+02 ± 3.14E+01†	**1.55E+02 ± 2.03E+01**
$f_{06}^{\star}$	1.17E+01 ± 4.16E+00†	2.40E+00 ± 1.80E+00	7.18E+00 ± 3.42E+00†	7.76E+00 ± 3.14E+00†	5.10E+00 ± 2.40E+00†	**2.04E+00 ± 1.26E+00**
f_{07}	1.02E-02 ± 1.41E-03†	**3.99E-03 ± 7.42E-04**‡	6.08E-03 ± 1.03E-03†	6.31E-03 ± 1.14E-03†	5.21E-03 ± 8.30E-04†	**4.97E-03 ± 1.07E-03**
f_{08}	1.66E+04 ± 5.57E+02†	**1.38E+04 ± 3.99E+02**‡	1.53E+04 ± 4.80E+02†	1.54E+04 ± 5.32E+02†	**1.42E+04 ± 4.92E+02**‡	1.44E+04 ± 5.79E+02
f_{09}	2.34E+02 ± 6.18E+00†	**1.93E+02 ± 5.86E+00**‡	2.10E+02 ± 8.63E+00†	2.13E+02 ± 6.27E+00†	2.00E+02 ± 7.19E+00	**1.98E+02 ± 8.97E+00**
$f_{10}^{\star}$	1.42E+00 ± 1.55E-01†	**2.52E-02 ± 1.58E-02**	4.40E-01 ± 2.67E-01†	6.52E-01 ± 2.62E-01†	1.34E-01 ± 1.26E-01†	**2.33E-02 ± 8.91E-03**
$f_{11}^{\star}$	6.77E-01 ± 8.58E-02†	**2.92E-02 ± 8.05E-03**‡	1.92E-01 ± 6.74E-02†	2.36E-01 ± 3.99E-02†	9.67E-02 ± 2.12E-02†	**3.22E-02 ± 7.83E-03**
$f_{12}^{\star}$	1.95E-01 ± 8.85E-02†	**1.62E-03 ± 4.00E-03**‡	2.79E-02 ± 2.54E-02†	3.41E-02 ± 2.83E-02†	8.94E-03 ± 1.25E-02†	**2.13E-03 ± 6.85E-03**
$f_{13}^{\star}$	8.77E+01 ± 4.73E+01†	**7.12E+00 ± 4.98E+00**‡	3.06E+01 ± 1.58E+01†	3.14E+01 ± 1.87E+01†	1.90E+01 ± 1.31E+01†	**9.84E+00 ± 8.52E+00**
	12/0/1	2/3/8	13/0/0	13/0/0	8/3/2	−−

* indicates that the intermediate error values at NFFEs = 100,000 of the function are used, since several algorithms can obtain the global optimum in the function.
† indicates our approach is significantly better than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.
‡ means that our approach is significantly worse than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.

Table 8: Comparison on the **Error** values among different JADE variants for functions $f_{01} - f_{13}$ at $D = 500$.

F	JADE-w	rJADE-w	SJADE	EPS-JADE	SaJADE	AP-AdapSS-JADE
f_{01}	1.40E-40 ± 1.48E-40†	**9.88E-59 ± 1.26E-58**‡	3.55E-49 ± 4.96E-49†	1.75E-47 ± 2.40E-47†	1.70E-57 ± 4.11E-57	**1.50E-57 ± 2.11E-57**
f_{02}	4.02E-08 ± 1.40E-07†	6.25E-09 ± 4.40E-08†	1.42E-09 ± 6.79E-09†	7.23E-09 ± 3.08E-08†	**3.54E-11 ± 1.54E-10**	**3.06E-11 ± 9.24E-11**
f_{03}	**1.73E+02 ± 2.86E+01**‡	1.83E+06 ± 3.96E+05†	2.57E+02 ± 3.81E+01	3.47E+02 ± 1.23E+02†	**2.17E+02 ± 4.44E+01**‡	2.44E+02 ± 3.92E+01
f_{04}	1.50E+01 ± 6.32E-01†	**1.38E+01 ± 5.41E-01**‡	1.45E+01 ± 6.76E-01†	1.44E+01 ± 6.73E-01†	1.45E+01 ± 7.05E-01†	**1.42E+01 ± 7.44E-01**
f_{05}	6.66E+02 ± 7.65E+01†	**6.02E+02 ± 7.02E+01**†	6.33E+02 ± 8.91E+01†	6.48E+02 ± 8.00E+01†	6.30E+02 ± 8.99E+01†	**5.82E+02 ± 7.69E+01**
$f_{06}^{\star}$	1.54E+03 ± 1.76E+02†	**4.18E+02 ± 4.32E+01**‡	7.94E+02 ± 8.84E+01†	8.64E+02 ± 1.09E+02†	6.72E+02 ± 6.95E+01†	**4.48E+02 ± 5.46E+01**
f_{07}	5.43E-02 ± 4.65E-03†	**2.71E-02 ± 3.00E-03**‡	3.55E-02 ± 4.03E-03†	3.75E-02 ± 4.35E-03†	**3.09E-02 ± 3.59E-03**‡	3.30E-02 ± 3.86E-03
f_{08}	5.20E+04 ± 8.83E+02†	**4.61E+04 ± 8.73E+02**‡	4.92E+04 ± 8.67E+02†	4.94E+04 ± 7.78E+02†	**4.66E+04 ± 8.63E+02**‡	4.81E+04 ± 9.88E+02
f_{09}	6.40E+02 ± 1.25E+01†	**5.37E+02 ± 1.38E+01**‡	5.83E+02 ± 1.35E+01†	5.87E+02 ± 1.32E+01†	**5.53E+02 ± 1.35E+01**‡	5.68E+02 ± 1.90E+01
f_{10}	3.03E+00 ± 1.86E-01†	**2.65E+00 ± 1.50E-01**‡	3.13E+00 ± 2.05E-01†	3.16E+00 ± 1.81E-01†	3.09E+00 ± 1.64E-01†	**2.73E+00 ± 1.29E-01**
$f_{11}^{\star}$	1.34E+01 ± 1.43E+00†	**3.47E+00 ± 2.91E-01**	7.04E+00 ± 7.08E-01†	7.69E+00 ± 9.17E-01†	5.73E+00 ± 7.24E-01†	**3.63E+00 ± 3.04E-01**
f_{12}	8.83E-03 ± 9.66E-03†	**5.60E-03 ± 7.57E-03**	8.34E-03 ± 1.26E-02	1.02E-02 ± 1.21E-02†	**6.47E-03 ± 8.70E-03**	8.21E-03 ± 1.61E-02
f_{13}	1.41E+02 ± 2.33E+01†	**3.02E+01 ± 7.21E+00**‡	6.88E+01 ± 1.59E+01†	7.49E+01 ± 1.37E+01†	5.56E+01 ± 1.26E+01†	**3.60E+01 ± 9.76E+00**
	12/0/1	3/2/8	11/2/0	13/0/0	6/3/4	−−

* indicates that the intermediate error values at NFFEs = 100,000 of the function are used, since several algorithms can obtain the global optimum in the function.
† indicates our approach is significantly better than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.
‡ means that our approach is significantly worse than its competitor by the Wilcoxon signed-rank test at $\alpha = 0.05$.

- For the basic unimodal functions ($f_{01} - f_{07}$), they are easy to be solved by all five JADE variants. rJADE-w obtains the best results, AP-AdapSS-JADE obtains the second best results, followed by rJADE-wo, JADE-w, and JADE-wo.
- For the basic multimodal functions ($f_{08} - f_{13}$), our approach obtains the best overall results, rJADE-wo is slightly worse than AP-AdapSS-JADE, followed by rJADE-w, JADE-w, and JADE-wo.
- For functions $F_{06} - F_{14}$, the shift and/or rotated features make them more difficult to be solved. AP-AdapSS-JADE gets the best overall results in terms of error values. On 5 out of 9 functions, it obtains the best results, and on two functions (F_{07}, F_{10}), it obtains the second best results.
- In general, with respect to the error value shown in Table 6, our approach is able to provide the best results in the most of the cases. It significantly performs better than JADE-wo, JADE-w, rJADE-wo, and rJADE-w on 18, 15, 14, and 13 functions (out of 22 functions), respectively.

4.6.2. Analysis on Large-scale Problems

According to the results described in Section 4.6.1, we can see that for the high-dimensional problems the performance of JADE variants benefits from the archive. Thus, in this section, we only select "current-to-p-best/1" with archive and "rand-to-$p$best/1" with archive to form the strategy pool. The population size $NP = 400$, Max_NFFEs = $D \times 5,000$. All other parameters are the same as mentioned in Section 4.1. For functions $f_{01} - f_{13}$, $D = 200$ and $D = 500$ are tested, respectively. Six JADE variants are compared, *i.e.*, JADE-w [50], rJADE-w [49], SJADE [30],

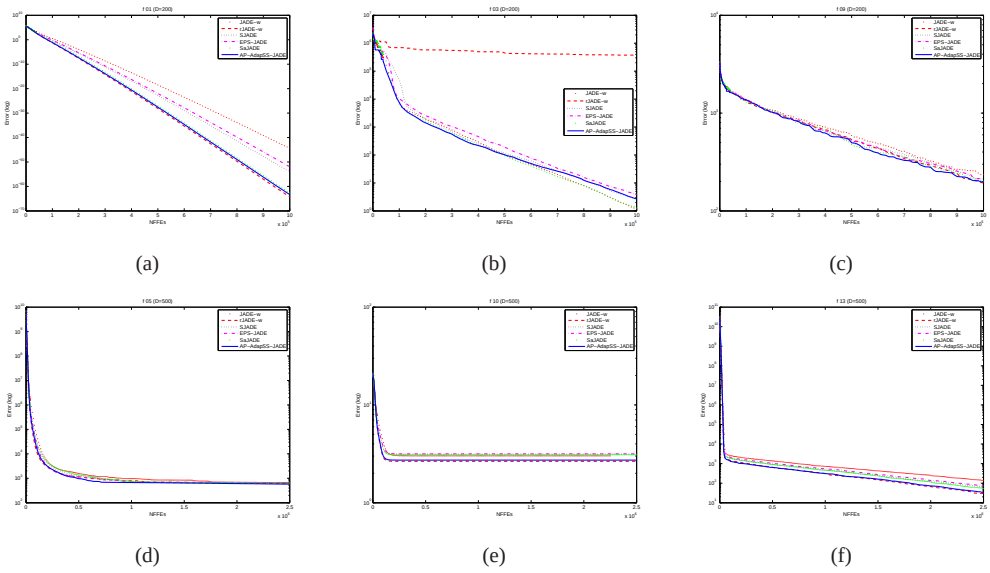

Figure 3: Convergence curves for different JADE variants. (a) $f_{01}(D = 200)$; (b) $f_{03}(D = 200)$; (c) $f_{08}(D = 200)$; (d) $f_{05}(D = 500)$; (e) $f_{10}(D = 500)$; (a) $f_{13}(D = 500)$.

EPS-JADE [20], SaJADE [14], and AP-AdapSS-JADE. For the former five algorithms, some specific parameters are set as in the original literature. The results are shown in Tables 7 and 8. The convergence curves of some selected functions are plotted in Figure 3.

The results shown in Tables 7, 8 and Figure 3 indicate that rJADE-w obtains the best overall results. Our proposed approach obtains the second best results. While JADE-w is the worst one. The reason is that "rand-to-*p*best" with archive is capable of providing higher diversity than "current-to-*p*best" with archive. When the two strategies form the strategy pool, "rand-to-*p*best" with archive is able to provide higher reward in the whole evolution process; and the synergy of these two strategies is invalid. Only one exception is for function f_{03}, where JADE-w obtains the best results. AP-AdapSS-JADE is significantly better than rJADE-w in f_{03}.

It is worth noting that although AP-AdapSS-JADE is worse than rJADE-w in the large-scale problems, it is better than other three multiple-strategy JADE variants, *i.e.*, SJADE, EPS-JADE, and SaJADE. The reason is that the *Adaptive Pursuit* technique is able to pursuit the best available strategy in the pool more quickly than other strategy adaptation techniques used in the three algorithms.

4.7. Parameter Study

In the previous experiments, the parameters (*i.e.*, p_{min}, α, and β) of the *Adaptive Pursuit* method are predefined for AP-AdapSS-JADE. In this section, the influence of different parameter settings of the *Adaptive Pursuit* method on AP-AdapSS-JADE is discussed. For each parameter, we use five values, *i.e.*, $p_{min} = \{0.0, 0.1, 0.15, 0.2, 0.24\}$[6], $\alpha, \beta = \{0.1, 0.4, 0.7, 0.9, 1.0\}$. Therefore, in this manner, following the experimental design presented in [19, 9], there are $K = 3$ factors, and each factor has $Q = 5$ levels; the resulting total number of parameter combinations is $Q^K = 125$. Since experimental design methods are capable of sampling a small number of well representative combinations for testing [9], in this section, we only employ the orthogonal design with $L_{25}(3^5)$ to sample the parameters of AP. The algorithm for constructing the orthogonal arrays can be found in [9]. The number of sampled parameter combinations is 25, and these combinations are shown in Table 9. All other parameters of AP-AdapSS-JADE are kept unchanged

[6]Note that in Section 3.1.2 we set $p_{min} \in (0, 1]$ to ensure that no strategy gets lost. However, in this experiment, to evaluate the influence $p_{min} = 0.0$ is used; this means that when the probability of a strategy equals 0.0, the strategy will be inactive in the following of the evolution process.

Table 9: Parameter sensitivity analysis: AP-AdapSS-JADE with the default parameter settings ($p_{min} = 0.05, \alpha = 0.3, \beta = 0.8$) is compared with AP-AdapSS-JADE with different parameter settings.

(p_{min}, α, β)	(0.00,0.10,0.10)	(0.00,0.40,0.40)	(0.00,0.60,0.60)	(0.00,0.80,0.80)	(0.00,1.00,1.00)
R^+	163	175	176	167	160
R^-	27	15	14	23	30
p-value	4.58E-03	5.23E-04	4.20E-04	2.40E-03	1.69E-01
significant	YES	YES	YES	YES	NO
(p_{min}, α, β)	(0.10,0.10,0.40)	(0.10,0.40,0.60)	(0.10,0.60,0.80)	(0.10,0.80,1.00)	(0.10,1.00,0.10)
R^+	82	76	96	97	83
R^-	89	95	94	74	88
p-value	8.99E-01	7.02E-01	9.84E-01	6.40E-01	9.32E-01
significant	NO	NO	NO	NO	NO
(p_{min}, α, β)	(0.15,0.10,0.60)	(0.15,0.40,0.80)	(0.15,0.60,1.00)	(0.15,0.80,0.10)	(0.15,1.00,0.40)
R^+	70	91	95	122	96
R^-	101	99	76	68	75
p-value	5.23E-01	8.91E-01	7.02E-01	2.93E-01	6.71E-01
significant	NO	NO	NO	NO	NO
(p_{min}, α, β)	(0.20,0.10,0.80)	(0.20,0.40,1.00)	(0.20,0.60,0.10)	(0.20,0.80,0.40)	(0.20,1.00,0.60)
R^+	110	108	123	97	109
R^-	80	63	67	93	62
p-value	5.68E-01	3.47E-01	2.75E-01	9.53E-01	3.25E-01
significant	NO	NO	NO	NO	NO
(p_{min}, α, β)	(0.24,0.10,1.00)	(0.24,0.40,0.10)	(0.24,0.60,0.40)	(0.24,0.80,0.60)	(0.24,1.00,0.80)
R^+	109	125	140	124	142
R^-	62	46	50	66	48
p-value	3.25E-01	8.98E-02	7.28E-02	2.58E-01	6.02E-02
significant	NO	NO	NO	NO	NO

as shown in Section 4.1. The experiments of each parameter combination are conducted over 50 independent runs. The multiple-problem Wilcoxon signed-rank test [11] is adopted to compare the performance of AP-AdapSS-JADE with default parameter settings to that of AP-AdapSS-JADE with different parameter settings. Note that when several algorithms can obtain the global optimum of a specific function within the Max_NFFEs, the NFFEs used in the second column of Table 2 are considered for the intermediate results, from which the mean values are extracted. The results are presented in Table 9; if R^+ is higher than R^-, it means that AP-AdapSS-JADE with default parameter settings is better than the compared algorithm, worst otherwise.

According to the results we can see that for 4 out of 25 parameter settings there are significant differences when different parameters are used, all the four applying minimal probability $p_{min} = 0.0$. When $p_{min} \neq 0.0$, there are no significant differences for all the other 20 parameter settings. The default parameter settings (*i.e.*, $p_{min} = 0.05, \alpha = 0.3, \beta = 0.8$) in AP-AdapSS-JADE are reasonable but not optimal, since there are other parameter settings that showed to obtain better results, such as ($p_{min} = 0.1, \alpha = 0.4, \beta = 0.6$) and ($p_{min} = 0.15, \alpha = 0.1, \beta = 0.6$). From this analysis, we can conclude that: (i) the *Adaptive Pursuit* strategy selection technique, being rewarded by the normalized average of relative fitness improvements, is not sensitive to its parameter setting when $p_{min} \neq 0.0$; and (ii) the cooperation of the multiple strategies is important to the performance of AP-AdapSS-JADE, since the "loss" of one strategy ($p_{min} = 0.0$) significantly deteriorates the results.

5. Conclusions and Future Work

Many mutation strategies have been proposed for generating new solutions within DE in different ways. Although allowing a very wide use of DE on many different fields of application, this number of available strategies creates an extra difficulty to the user: it is not trivial to define which strategy should be used on a given problem in order to achieve good performance. Besides, the strategies are not simply problem-dependent; indeed, their performance tends to vary as the search goes on, according to the characteristics of the region of the search space that is being explored by the current population.

Motivated by this difficulty, in this paper we extend our recent work [15] on investigating the use of the adaptive strategy selection approach within DE, *i.e.*, a method capable of automatically selecting which strategy should be applied at each instant of the search, while solving the problem, according to how well each of the available strategies have recently performed in the same search/optimization process, without any prior knowledge. In order to verify the

performance of such approach, some different adaptive strategy selection combinations were coupled with JADE [50, 49], a recently proposed DE variant; the final method being referred to as AdapSS-JADE. Experiments were conducted on 22 widely used benchmark functions. From the results and analysis previously mentioned, we can summarize that:

- To implement adaptive strategy selection, credit assignment, *i.e.*, how to assign a reward to a strategy after its application, is an important issue that needs to be addressed. In this work, four credit assignment methods based on the relative fitness improvement were presented and their performance was analyzed. Although there were no significant differences among these four methods on most of the functions, the normalized average reward method was able to obtain the best results in terms of the averaged ranking. The reason might be that the average reward is able to provide the reward for each strategy more exactly than the extreme reward in DE for continuous problems, while the normalization can efficiently eliminate the magnitude differences between the previous estimate $q_a(t)$ and the reward $r_a(t)$, and also between the very different fitness ranges (consequently rewards) that might be found in the considered problems.

- The second and not less important issue in adaptive strategy selection is the strategy selection technique itself, *i.e.*, how to select the next strategy to be applied based on the rewards recently received by each of the available ones. Two strategy selection techniques, the *Probability Matching* (PM) and the *Adaptive Pursuit* (AP), were analyzed to address this issue. According to their performance using the normalized average reward of relative fitness improvements as credit assignment, we can conclude that both PM and AP based AdapSS-JADE approaches are able to enhance the performance of DE and obtain better results than the baseline methods. In addition, the AP technique converges more rapidly and accurately to an efficient strategy probability distribution compared with the PM method, which is consistent with the conclusions presented in [41].

- According to the analysis of the strategy adaptation of the best resulting method, referred to as AP-AdapSS-JADE, our approach is capable of efficiently selecting the most suitable strategy while solving the problem, without any prior knowledge, performing consistently better than the JADE using a single strategy, for each of the four available strategies. This latter conclusion confirms the assumption that the *competition* and *cooperation* between the available strategies are important to provide a higher achievement.

- Besides, AP-AdapSS-JADE obtains highly competitive results when compared to other recent advanced DEs, presenting a better performance on most of the functions, while not considerably augmenting the total computational complexity of the base JADE algorithm.

- Based on the analysis on large-scale problems, when the synergy of different strategies in the pool is invalid, AP-AdapSS-JADE only obtains the second best results. It is reasonable because the *Adaptive Pursuit* technique needs some generations to pursuit the most suitable strategy for a specific problem.

- Finally, the parameter analysis done shows that the AP-AdapSS-JADE approach is not sensitive to the setting of its 3 hyper-parameters, the minimal probability p_{min}, the adaptation rate α, and the learning rate β; several different parameter settings were able to achieve similar good performance.

For further work, when tackling multimodal problems, the maintenance of a minimal level of diversity is also important for the search process, and thus should also be taken into account for the rewarding of the strategies; the Compass or the Pareto-based approaches, proposed in [22], could be analyzed for this purpose in the future. Additionally, using other strategy selection techniques, such as the approaches based on Multi-Armed Bandits [10], could also be an interesting direction for the strategy adaptation within DE. In addition, experimental results indicate that synergy of strategies is very important for multiple-strategy DE. Thus, another direction of future work is performing comprehensively empirical study to investigate how to select strategy to form the pool. Furthermore, in the future, we will try to improve the performance of our approach on large-scale problems [46, 39, 17, 38].

Acknowledgments

The authors would like to sincerely thank the Editor and the anonymous reviewers for their constructive comments, which improved the original paper significantly.

A. Benchmark Functions

The details of the first 13 functions ($f_{01} - f_{13}$) used in this work are the following:

- f_{01} Sphere function:

$$f_{01} = \sum_{i=1}^{D} x_i^2, \quad x_i \in [-100, 100]$$

Minimize: $f_{01}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{02} Schwefel's problem 2.22:

$$f_{02} = \sum_{i=1}^{D} |x_i| + \prod_{i=1}^{D} |x_i|, \quad x_i \in [-10, 10]$$

Minimize: $f_{02}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{03} Schwefel's problem 1.2:

$$f_{03} = \sum_{i=1}^{D} \Big(\sum_{j=1}^{i} x_j \Big)^2, \quad x_i \in [-100, 100]$$

Minimize: $f_{03}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{04} Schwefel's problem 2.21:

$$f_{04} = \max_i \{|x_i|, 1 \le i \le D\}, \quad x_i \in [-100, 100]$$

Minimize: $f_{04}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{05} Generalized Rosenbrock's function:

$$f_{05} = \sum_{i=1}^{D-1} \Big[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \Big], \quad x_i \in [-30, 30]$$

Minimize: $f_{05}(\mathbf{x}) = f(1, \cdots, 1) = 0$.

- f_{06} Step function:

$$f_{06} = \sum_{i=1}^{D-1} \Big(\lfloor x_i + 0.5 \rfloor \Big)^2, \quad x_i \in [-100, 100]$$

Minimize: $f_{06}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{07} Quartic function with noise:

$$f_{07} = \sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1), \quad x_i \in [-1.28, 1.28]$$

Minimize: $f_{07}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{08} Generalized Schwefel's problem 2.26:

$$f_{08} = \sum_{i=1}^{D} \Big(- x_i \sin(\sqrt{|x_i|}) \Big) + 418.98288727243369 \times D, \quad x_i \in [-500, 500]$$

Minimize: $f_{08}(\mathbf{x}) = f(420.9687, \cdots, 420.9687) = 0$.

- f_{09} Generalized Rastrigin's function:

$$f_{09} = \sum_{i=1}^{D} \Big(x_i^2 - 10 \cos(2\pi x_i) + 10 \Big), \quad x_i \in [-5.12, 5.12]$$

Minimize: $f_{09}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{10} Ackley's function:

$$f_{10} = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + \exp(1), \quad x_i \in [-32, 32]$$

Minimize: $f_{10}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{11} Generalized Griewank function:

$$f_{11} = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1, \quad x_i \in [-600, 600]$$

Minimize: $f_{11}(\mathbf{x}) = f(0, \cdots, 0) = 0$.

- f_{12}, f_{13} Generalized penalized functions:

$$\begin{aligned} f_{12} \quad = \quad & \frac{\pi}{D}\left\{10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2 \cdot [1 + 10\sin^2(\pi y_{i+1})] + (y_D - 1)^2\right\} \\ & + \sum_{i=1}^{D} u(x_i, 10, 100, 4) \end{aligned}, \quad x_i \in [-50, 50]$$

Minimize: $f_{12}(\mathbf{x}) = f(1, \cdots, 1) = 0$.

$$\begin{aligned} f_{13} \quad = \quad & \frac{1}{10}\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + (x_D - 1)^2[1 + \sin^2(2\pi x_D)]\right\} \\ & + \sum_{i=1}^{D} u(x_i, 5, 100, 4) \end{aligned}, \quad x_i \in [-50, 50]$$

Minimize: $f_{13}(\mathbf{x}) = f(1, \cdots, 1) = 0$.

where

$$y_i = 1 + \frac{1}{4}(x_i + 1)$$

and

$$u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m, & x_i > a \\ 0, & -a \le x_i \le a \\ k(-x_i - a)^m, & x_i < a \end{cases}$$

References

[1] Birattari, M., Stützle, T., Paquete, L., Varrentrapp, K., 2002. A racing algorithm for configuring metaheuristics. In: W. B. Langdon et al. (Ed.), Proc. Genetic and Evolutionary Computation Conference. Morgan Kaufmann, pp. 11–18.

[2] Brest, J., Bošković, B., Greiner, S., Žumer, V., Maučec, M. S., May 2007. Performance comparison of self-adaptive and adaptive differential evolution algorithms. Soft Comput. 11 (7), 617–629.

[3] Das, S., Abraham, A., Chakraborty, U. K., Konar, A., 2009. Differential evolution using a neighborhood-based mutation operator. IEEE Trans. on Evol. Comput. 13 (3), 526–553.

[4] Das, S., Sil, S., 2010. Kernel-induced fuzzy clustering of image pixels with an improved differential evolution algorithm. Information Sciences 180 (8), 1237 – 1256.

[5] Das, S., Suganthan, P. N., 2011. Differential evolution: A survey of the state-of-the-art. IEEE Trans. on Evol. Comput. 15 (1), 4–31.

[6] Dorronsoro, B., Bouvry, P., 2011. Improving classical and decentralized differential evolution with new mutation operator and population topologies. Evolutionary Computation, IEEE Transactions on 15 (1), 67 –98.

[7] Eiben, A. E., Michalewicz, Z., Schoenauer, M., Smith, J. E., 2007. Parameter control in evolutionary algorithms. In: Lobo, F.G. et al. (Ed.), Parameter Setting in Evolutionary Algorithms. Springer, pp. 19–46.

[8] Epitropakis, M. G., Tasoulis, D. K., Pavlidis, N. G., Plagianakos, V. P., Vrahatis, M. N., Feb. 2011. Enhancing differential evolution utilizing proximity-based mutation operators. IEEE Trans. on Evol. Comput. 15 (1), 99–119.

[9] Fang, K., Ma, C., 2001. Orthogonal and Uniform Design. Science Press, Beijing, China, in Chinese.

[10] Fialho, Á., Schoenauer, M., Sebag, M., 2009. Analysis of adaptive operator selection techniques on the royal road and long k-path problems. In: Proc. Genetic Evol. Comput. Conf. pp. 779–786.

[11] García, S., Molina, D., Lozano, M., Herrera, F., 2009. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 special session on real parameter optimization. Journal of Heuristics 15 (6), 617 – 644.
[12] Goldberg, D. E., 1990. Probability matching, the magnitude of reinforcement, and classifier system bidding. Mach. Learn. 5 (4), 407–425.
[13] Gong, W., Cai, Z., Jiang, L., 2008. Enhancing the performance of differential evolution using orthogonal design method. Applied Mathematics and Computation 206 (1), 56–69.
[14] Gong, W., Cai, Z., Ling, C. X., Li, H., 2011. Enhanced differential evolution with adaptive strategies for numerical optimization. IEEE Transactions on Systems, Man, and Cybernetics: Part B – Cybernetics 41 (2), 397–413.
[15] Gong, W., Fialho, A., Cai, Z., July 2010. Adaptive strategy selection in differential evolution. In: Branke, J. (Ed.), Genetic and Evolutionary Computation Conference (GECCO 2010). ACM Press, pp. 409–416.
[16] Hachicha, N., Jarboui, B., Siarry, P., 2011. A fuzzy logic control using a differential evolution algorithm aimed at modelling the financial market dynamics. Information Sciences 181 (1), 79 – 91.
[17] Herrera, F., Lozano, M., D.Molina, 2010. Special issue on the scalability of evolutionary algorithms and other metaheuristics for large scale continuous optimization problems.
URL http://sci2s.ugr.es/eamhco/CFP.php
[18] Lee, C.-Y., Yao, X., 2004. Evolutionary programming using mutations based on the Lévy probability distribution. IEEE Trans. on Evol. Comput. 8 (1), 1–13.
[19] Leung, Y.-W., Wang, Y., Feb 2001. An orthogonal genetic algorithm with quantization for global numerical optimization. IEEE Trans. on Evol. Comput. 5 (1), 41–53.
[20] Mallipeddi, R., Suganthan, P., Pan, Q., Tasgetiren, M., 2011. Differential evolution algorithm with ensemble of parameters and mutation strategies. Applied Soft Computing 11 (2), 1679–1696, in press.
[21] Mallipeddi, R., Suganthan, P. N., 2010. Differential evolution algorithm with ensemble of parameters and mutation and crossover strategies. In: Swarm, Evolutionary, and Memetic Computing. Vol. LNCS 6466. pp. 71–78.
[22] Maturana, J., Lardeux, F., Saubion, F., 2010. Autonomous operator management for evolutionary algorithms. Journal of Heuristics 16 (6), 881–909.
[23] Mezura-Montes, E., Miranda-Varela, M. E., del Carmen Gómez-Ramón, R., November 2010. Differential evolution in constrained numerical optimization: An empirical study. Information Sciences 10 (22), 4223–4262.
[24] Noman, N., Iba, H., Feb 2008. Accelerating differential evolution using an adaptive local search. IEEE Trans. on Evol. Comput. 12 (1), 107–125.
[25] Norman, M., Moscato, P., Plata, L., 1991. A competitive-cooperative approach to complex combinatorial search.
URL http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.44.776&rep=rep1&type=pdf
[26] Ong, Y.-S., Keane, A. J., Apr 2004. Meta-Lamarckian learning in memetic algorithms. IEEE Trans. on Evol. Comput. 8 (2), 99–110.
[27] Ozer, A. B., 2010. Cide: Chaotically initialized differential evolution. Expert Systems with Applications 37 (6), 4632 – 4641.
[28] Pan, Q.-K., Suganthan, P., Wang, L., Gao, L., Mallipeddi, R., Jan. 2011. A differential evolution algorithm with self-adapting strategy and control parameters. Computers and Operations Research 38 (1), 394 – 408.
[29] Price, K., Storn, R., Lampinen, J., 2005. Differential Evolution: A Practical Approach to Global Optimization. Springer-Verlag, Berlin.
[30] Qin, A. K., Huang, V. L., Suganthan, P. N., Apr 2009. Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans. on Evol. Comput. 13 (2), 398–417.
[31] Qin, A. K., Suganthan, P. N., 2005. Self-adaptive differential evolution algorithm for numerical optimization. In: IEEE Congr. on Evol. Comput. pp. 1785–1791.
[32] Rahnamayan, S., Tizhoosh, H., Salama, M., Feb 2008. Opposition-based differential evolution. IEEE Trans. on Evol. Comput. 12 (1), 64–79.
[33] Shang, Y.-W., Qiu, Y.-H., 2006. A note on the extended rosenbrock function. Evol. Comput. 14 (1), 119–126.
[34] Shaw, C., Williams, K., Assassa, R., 2003. Patients' views of a new nurse-led continence service. Journal of Clinical Nursing 9 (4), 574–584.
[35] Storn, R., Price, K., Dec 1997. Differential evolution–A simple and efficient heuristic for global optimization over continuous spaces. J. of Global Optim. 11 (4), 341–359.
[36] Storn, R., Price, K., 2010. Home page of differential evolution.
URL http://www.ICSI.Berkeley.edu/~storn/code.html
[37] Suganthan, P. N., Hansen, N., Liang, J. J., Deb, K., Chen, Y.-P., Auger, A., Tiwari, S., 2005. Problem definitions and evaluation criteria for the CEC2005 special session on real-parameter optimization.
URL http://www.ntu.edu.sg/home/EPNSugan
[38] Tang, K., Li, X., Suganthan, P. N., Yang, Z., Weis, T., 2009. Benchmark functions for the CEC'2010 special session and competition on large scale global optimization. Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China & Nanyang Technological University, Singapore.
URL http://nical.ustc.edu.cn/cec10ss.php
[39] Tang, K., Yao, X., Suganthan, P. N., MacNish, C., Chen, Y. P., Chen, C. M., Yang, Z., 2007. Benchmark functions for the CEC'2008 special session and competition on large scale global optimization. Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China.
URL http://nical.ustc.edu.cn/cec08ss.php
[40] Thathachar, M., Sastry., P., 1985. A class of rapidly converging algorithms for learning automata. IEEE Transactions on Systems, Man and Cybernetics SMC-15, 168–175.
[41] Thierens, D., 2005. An adaptive pursuit strategy for allocating operator probabilities. In: Proc. Genetic Evol. Comput. Conf. pp. 1539–1546.
[42] Wang, Y., Cai, Z., Zhang, Q., 2011. Differential evolution with composite trial vector generation strategies and control parameters. IEEE Trans. on Evol. Comput. 15 (1), 55–66.
[43] Wang, Y., Li, B., Weise, T., 2010. Estimation of distribution and differential evolution cooperation for large scale economic load dispatch optimization of power systems. Information Sciences 180 (12), 2405 – 2420.
[44] Whitacre, J., Pham, T., Sarker, R., 2006. Use of statistical outlier detection method in adaptive evolutionary algorithms. In: Cattolico, M.

(Ed.), Proc. Genetic Evol. Comput. Conf. (GECCO). pp. 1345–1352.
[45] Xie, X.-F., Zhang, W.-J., 2004. SWAF: Swarm algorithm framework for numerical optimization. In: Proc. Genetic Evol. Comput. Conf., Part I. Vol. 3102 of LNCS. pp. 238–250.
[46] Yang, Z., Tang, K., Yao, X., Aug. 2008. Large scale evolutionary optimization using cooperative coevolution. Inf. Sci. 178, 2985–2999.
[47] Yang, Z., Tang, K., Yao, X., 2008. Self-adaptive differential evolution with neighborhood search. In: Proc. IEEE Congress on Evol. Comput. pp. 1110–1116.
[48] Yao, X., Liu, Y., Lin, G., Jul 1999. Evolutionary programming made faster. IEEE Trans. on Evol. Comput. 3 (2), 82–102.
[49] Zhang, J., Sanderson, A. C., 2009. Adaptive Differential Evolution: A Robust Approach to Multimodal Problem Optimization. Springer-Verlag, Berlin.
[50] Zhang, J., Sanderson, A. C., Oct 2009. JADE: Adaptive differential evolution with optional external archive. IEEE Trans. on Evol. Comput. 13 (5), 945–958.