A Novel Crossover-First Differential Evolution Algorithm with Explicitly Tunable Mutation Rates for Evolutionary-Based Global Optimization

JASON TEO
Faculty of Computing and Informatics
Universiti Malaysia Sabah
Jalan UMS, 88450 Kota Kinabalu, Sabah
MALAYSIA
jtwteo@ums.edu.my

KIM-ON CHIN, SHALIZA WAHAB,
AZALI SAUDI, SITI HASNAH TANALOL
Faculty of Computing and Informatics
Universiti Malaysia Sabah
Jalan UMS, 88450 Kota Kinabalu, Sabah
MALAYSIA

Abstract: Differential Evolution (DE) is currently one of the most popular evolutionary-based global optimization algorithms being simple to understand and implement as well as having fast convergence and robustness across a wide range of problems. Although it is classed as an evolutionary algorithm (EA), its genetic operations are atypical of such classes of algorithms. EAs typically perform crossover followed by mutation where both operations have an explicitly tunable rate of operation. However in DE, the mutation operation is conducted before the crossover operation. Moreover, although DE has a crossover rate, it does not have a mutation rate; rather it mandatorily mutates every gene in its chromosome essentially performing a 100% rate of mutation. Following this line of observation, we proceeded to experiment with a novel version of DE where the crossover and mutation operations are reversed to mimic typical EAs as well as to add in an explicitly tunable mutation rate. We have found that this simple and intuitive yet previously unexplored modification to DE is able to improve its performance, particularly in more complex search spaces with highly non-uniform fitness landscapes. Non-parametric tests show that the improvements are statistically significant.

Key–Words: Global Optimization; Differential Evolution; Heuristic Search; Evolutionary Optimization; Genetic Operations.

1 Introduction

Evolutionary-based optimization methods have become one of the most powerful and popular methods for solving highly non-linear and complex numerical optimization problems [15, 14, 16, 17]. Storn and Price first proposed the Differential Evolution (DE) algorithm in 1995 [19] and has since become one of the the most widely used evolutionary-based global optimization algorithms by real-world practitioners [3, 13]. Its popularity and widespread adoption stems from its following advantageous characteristics [3, 7, 9, 11]:

1. very simple to understand and implement;
2. highly competitive all-round global optimizer that performs well across synthetic as well as real-world problems, including non-linear constraints, non-convex, non-differentiable, multi-objective, and dynamic components;
3. fast convergence speed and robust;
4. very few parameters that need to be tuned;
5. extremely scalable to large dimensions and expensive problems due to its very low computational overhead.

There is currently a very large interest in DE with an average of 190 papers being published monthly [1]. Among all of its advantageous properties, the most commonly cited reason for its high adoption rate among practitioners is its simplicity factor, where its critical routines can be implemented in 4-5 lines code [3]. This is indeed a very important factor when it comes to biasing non-experts' decisions in attempting to implement an intelligent heuristic to solve their optimization problems. This is indeed true when one observes that the most commonly implemented DE version when applied in solving real-world problems is actually the original and most basic version known as "rand/1/bin" [3, 2, 18]. Simplicity appears to be key in

[1]Scopus database search for articles with title/abstract/keywords "Differential Evolution" within physical sciences category only for the period of Jan-Feb 2015 returned 382 results.

real-world adoption of DE where more involved versions of DE, though providing superior performance, is much less preferred compared to the basic, original version of DE.

However, although there is a large body of work that furthers the state-of-the-art in DE research, practically all of the recent advancements have diverged from the original motivations of DE: a "simple yet powerful" optimization algorithm [18]. Although these efforts have advanced DE academically, the numerous modifications, refashioning, tweaking and variations have added a highly significant level of complexity to the basic DE algorithm. As notable from the reviews mentioned above, practically all of the improved algorithms have added sub-routines (e.g. self-adaptation of parameters [7]), global population adjustments [1, 21], specialized functions (e.g. surrogate models [11]) and/or more complex operators (e.g. aggregation methods [12]). This has made DE comparatively more difficult to understand and subsequently more challenging to implement.

Our work returns to the original motivations of DE by keeping simple yet attempting to improve its power in global optimization. Firstly, typical EAs conduct the crossover operation before they conduct the mutation operation; however in DE, the reverse is true. Secondly, typical EAs have two explicitly tunable parameters, that is the crossover rate and the mutation rate; however in DE, there is only one explicitly tunable parameter in the form of its crossover rate. Thus, our study explores two basic questions:

1. Does reversing the order of genetic operations benefit DE?

2. Does having an explicitly tunable mutation operator benefit DE?

The motivation for attempting to study these effects is that firstly, macro-operations in the form of crossover typically conducts larger scale rearrangements of the genetic material of the child chromosomes before micro-operations in the form of mutation performs smaller scale adjustments of the crossover-ed genetic material in the child chromosomes. We call our novel implementation of DE which implements crossover first with explicit mutation as XDEM. We test XDEM using the CEC 2005 global optimization competition suite of 25 benchmark test problems comprising unimodal, multimodal, expanded and hybrid composition functions and compare its results against the standard DE as well as three other commonly used global search heuristics. A non-parametric statistical test is conducted followed by 8 related post-hoc procedures.

This paper is presented as follows. Section 2 explains the basic DE and current methods of augmenting DE. Section 3 presents our simple approach to XDEM. Section 4 provides the experimental results that compares XDEM to the basic DE as well as three other heuristics. Finally, some conclusions and future avenues of exploration are given in Section 5.

2 Background

The basic DE algorithm is a population-based, real-valued, stochastic global optimizer that conducts the following operations in the following order after initialization: 1. mutation; 2. crossover; 3. selection; 4. repeat until termination. It also requires three user-defined parameters to be set prior to the optimization run: 1. F: scaling factor; 2. CR: crossover rate; and 3. NP: population size. The reader may refer to [18] for a detailed treatment of DE. In brief, given a minimization problem *f*:

$$f(x)^* = \min_{x_i \in \Omega} f(x_i) \tag{1}$$

where x_i is a vector with *D* dimensions, x^* is the global solution and $\Omega \subseteq R^D$, DE will attempt to optimize the vector of variables $x = \{x_1, x_2, ..., x_D\}$, where $x_i^G = \{x_{i,1}^G, x_{i.2}^G, ..., x_{i,D}^G\}$ represents the *i*th individual in the population of solutions of the *G*th generation of optimization iteration.

Importantly, a new trial solution is generated in the following order:

(1) Mutation: for each parent x_i^G, a new vector is created as follows:

$$v_i^{G+1} = x_{r1}^G + F \cdot (x_{r2}^G - x_{r3}^G), \tag{2}$$

where $r1, r2$, and $r3$ are randomly chosen from $[1, NP]$ and $i \neq r1 \neq r2 \neq r3$.

(2) Crossover: for each parent x_i^G, a trial solution is created as follows:

$$u_{i,j}^{G+1} = \begin{cases} v_{i,j}^{G+1} & \text{if } R_j \leq CR \text{ or } j = j_{rand}. \\ x_{i,j}^G & \text{otherwise.} \end{cases} \tag{3}$$

where R_j is a uniform random $[0, 1]$ and j_{rand} is random integer $[1, D]$.

(3) Selection: the new trial solution competes with the parent for survival to the next optimization iteration:

$$x_i^{G+1} = \begin{cases} u_i^{G+1} & \text{if } f(u_i^{G+1}) \leq f(x_i^G). \\ x_i^G & \text{otherwise.} \end{cases} \tag{4}$$

Next, we present the proposed modification to the basic DE algorithm.

3 XDEM: The Crossover-First Differential Evolution Algorithm with Explicit Mutation

Here we explain the simple proposed modification to the basic DE algorithm presented above. The only difference compared to DE is the reversal in the order of operations between mutation and crossover in the generation of the trial solution as well as the introduction of an explicit mutation rate parameter.

In XDEM, a new trial solution is generated in the following order:

(1) Crossover: for each parent x_i^G, a new vector is created as follows:

$$v_{i,j}^{G+1} = \begin{cases} x_{r1,j}^G & \text{if } R_j \leq CR \text{ or } j = j_{rand}. \\ x_{i,j}^G & \text{otherwise.} \end{cases} \tag{5}$$

where where $r1$ is randomly chosen from $[1, NP]$, R_j is a uniform random $[0, 1]$, j_{rand} is random integer $[1, D]$ and $i \neq r1$.

(2) Mutation: for each parent x_i^G, a trial solution is created as follows:

$$u_{i,j}^{G+1} = \begin{cases} x_{R2,j}^G + F \cdot (x_{R3,j}^G - x_{R4,j}^G) & \text{if } R_j \leq MR. \\ v_{i,j}^{G+1} & \text{otherwise.} \end{cases} \tag{6}$$

where $r2, r3$, and $r4$ are randomly chosen from $[1, NP]$, R_j is a uniform random $[0, 1]$ and $i \neq r1 \neq r2 \neq r3 \neq r4$. MR denotes the explicit mutation rate parameter that is tunable.

(3) Selection: the new trial solution competes with the parent for survival to the next optimization iteration:

$$x_i^{G+1} = \begin{cases} u_i^{G+1}, & \text{if } f(u_i^{G+1}) \leq f(x_i^G). \\ x_i^G, & \text{otherwise.} \end{cases} \tag{7}$$

In this proposed approach, XDEM functions more conventionally as an evolutionary optimization algorithm by first crossing-over the target parent vector with a randomly chosen individual from the existing population in Eq. 5. It then takes this crossover-ed vector and mutates it using the standard DE mutation operator (as described in Section 2 by Eq. 3) whereby a scaled differential between two randomly chosen individuals are added to a third randomly chosen individual, which is implemented as described in Eq. 6 to create a new trial solution. In the mutation operation, there is now an explicitly tunable mutation rate parameter which we denote as MR. The survivor selection method remains unchanged.

4 Methodology

In order to benchmark the performance of XDEM, we have chosen the CEC 2005 global optimization competition suite of 25 benchmark test problems [20]. It contains 25 minimization problems with diverse characteristics ranging from simple unimodal functions (F1-F5) to straightforward multimodal functions (F6-F12) as well as expanded functions (F13 & F14) to the highly complex and non-uniform hybrid composition functions (F15-F25). XDEM is run three times separately to test the effects of setting the new mutation rate parameter at low, medium and high settings of 01, 0.5 and 0.9 (denoted as XDEM1, XDEM5 and XDEM9) respectively.

Summarized below are the experimental settings used in this study, which is similar to those of [4]:

- **Optimization parameters:** $D = 10$, repeats = 50, termination =$f(x) < 10^{-8}$ or $G = 100,000$.
- **DE**: rand/1/bin, NP=100, CR=0.9, F=0.5.
- **XDEM**: rand/1/bin, NP=100, CR=0.9, F=0.5, MR=0.1,0.5,0.9.
- **Classic Particle Swarm Optimization (PSO) [10]:** $c1 = 2.8, c2 = 1.3$, and w from 0.9 to 0.4, individuals = 100.
- **CHC [5]:** BLX-α crossover operator, $\alpha = 0.5$, individuals = 50.
- **Steady-State Genetic Algorithm [6, 8]:** BLX-α crossover operator, negative assortative mating, BGA mutation, $\alpha = 0.5$, individuals = 50.

5 Results

Table 1: Average Best Solutions Found Over 50 Repeated Runs

Data-set	XDEM1	XDEM5	XDEM9	DE-Bin	PSO	CHC	SSGA
F1	3.47E+01	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.23E-04	2.46E+00	**0.00E+00**
F2	7.70E+02	**0.00E+00**	8.72E-05	**0.00E+00**	2.59E-02	1.18E+02	8.72E-05
F3	3.30E+06	3.40E+05	7.95E+04	**0.00E+00**	5.17E+04	2.70E+05	7.95E+04
F4	7.31E+02	**0.00E+00**	2.59E-03	**0.00E+00**	2.49E+00	9.19E+01	2.59E-03
F5	8.07E+02	6.07E+01	1.34E+02	**1.10E-04**	4.10E+02	2.64E+02	1.34E+02
F6	7.21E+05	7.47E+00	6.17E+00	**2.40E-01**	7.31E+02	1.42E+06	6.17E+00
F7	1.27E+02	**3.98E-02**	1.27E+03	3.46E-01	2.68E+01	1.27E+03	1.27E+03
F8	2.06E+01	2.04E+01	2.04E+01	**2.03E+01**	2.04E+01	**2.03E+01**	2.04E+01
F9	5.27E+00	**0.00E+00**	**0.00E+00**	1.88E+01	1.44E+01	5.89E+00	**0.00E+00**
F10	2.91E+01	1.60E+01	1.71E+01	2.64E+01	1.40E+01	**7.12E+00**	1.71E+01
F11	9.65E+00	8.98E+00	3.26E+00	8.82E+00	5.59E+00	**1.60E+00**	3.26E+00
F12	**2.17E+03**	3.76E+02	2.79E+02	9.47E+03	6.36E+02	7.06E+02	2.79E+02
F13	1.22E+00	**6.69E-01**	6.71E+01	2.02E+00	1.50E+00	8.30E+01	6.71E+01
F14	3.84E+00	3.53E+00	2.26E+00	3.61E+00	3.30E+00	**2.07E+00**	2.26E+00
F15	**2.21E+02**	2.55E+02	2.92E+02	3.51E+02	3.40E+02	2.75E+02	2.92E+02
F16	1.59E+02	1.10E+02	1.05E+02	1.48E+02	1.33E+02	**9.73E+01**	1.05E+02
F17	1.58E+02	1.33E+02	1.19E+02	1.64E+02	1.50E+02	**1.05E+02**	1.19E+02
F18	9.13E+02	**6.36E+02**	8.06E+02	8.26E+02	8.51E+02	8.80E+02	8.06E+02
F19	9.19E+02	**6.31E+02**	8.90E+02	8.26E+02	8.50E+02	8.80E+02	8.90E+02
F20	9.16E+02	**6.25E+02**	8.89E+02	8.28E+02	8.51E+02	8.96E+02	8.89E+02
F21	8.73E+02	**6.42E+02**	8.52E+02	1.02E+03	9.14E+02	8.16E+02	8.52E+02
F22	8.25E+02	7.72E+02	7.52E+02	**6.07E+02**	8.07E+02	7.74E+02	7.52E+02
F23	9.90E+02	**7.27E+02**	1.00E+03	1.09E+03	1.03E+03	1.08E+03	1.00E+03
F24	4.36E+02	4.09E+02	**2.36E+02**	4.08E+02	4.12E+02	2.96E+02	**2.36E+02**
F25	1.39E+03	4.09E+02	1.75E+03	**4.08E+02**	5.10E+02	1.76E+03	1.75E+03

Table 1 shows the average best solution found by all the algorithms over 50 repeated trials over the 25 benchmark test problems. The best results obtained for each problem is highlighted in bold. XDEM5 has the highest number of best results with 11 followed by DE with 9, CHC with 6, XDEM9 and SSG with 3 each and XDEM1 with 2. PSO was not able to obtain any best solution in this particular test setup. It is interesting to note that although DE came second with 9 best results, 5 of these came from the simplest class of problems comprising unimodal functions, two from the next simplest class of problems comprising multimodal functions with only two coming from the most difficult class of hybrid composition functions. From these results, it appears that having the crossover-first operation in the novel XDEM algorithm with a medium mutation rate setting (XDEM5) is able to improve DE's performance, particularly in the hardest class of problems which are the expanded (F13 & F14) and hybrid composition (F15-F25) functions. Inclusion of the tunable mutation rate parameter was also beneficial but only in certain conditions such as when it was set to a low setting in XDEM1, it was able to obtain the best results in F12 and F15, and then on the high setting in XDEM9 in F1, F9 and F24. Combining all the runs of the XDEM with mutation rates would yield a total 14 best results against the original DE of only 9. Therefore, these results do appear to support an initial positive answer to both questions posed earlier in the introduction section of this study, that is the crossover-first order of genetic operation and tunable mutation rates can benefit DE. Next, some statistical procedures will be conducted to test the significance of the differences obtained above.

Table 2: Aligned Friedman Average Rankings (F1-F25)

Algorithm	Ranking
XDEM5	54.84
XDEM9	80.70
SSGA	80.70
DE-Bin	85.42
PSO	90.16
CHC	95.66
XDEM1	128.52

The Aligned Friedman procedure [4]was conducted on all the algorithms and the average rankings are shown in Table 2. The best ranked algorithm was XDEM5, followed by XDEM9 and SSGA. The worst ranked algorithm was XDEM1, which is an indication that although a low mutation rate setting was beneficial in a limited number of cases, it was not ideal in the majority of the problems. The standard DE algorithm was ranked exactly in the middle of the pack but importantly, the two of the novel XDEM algorithms were ranked better than DE.

Next, we proceeded to conduct a non-parametric statistical test on the results obtained using the Aligned Friedman procedure. The statistics obtained are as follows:

- Aligned Friedman statistic (distributed according to chi-square with 6 degrees of freedom: 21.036870. P-value computed by Aligned Friedman Test: 0.001806.

As the test statistic shows statistically significant differences in the results obtained, we then proceeded to conduct 8 post-hoc test based on the results obtained from the Aligned Friedman procedure, namely the Bonferroni-Dunn, Holm, Hochberg, Hommel, Holland, Rom, Finner and Li post-hoc tests [4].

Table 3: Adjusted p-values (F1-F25) (ALIGNED FRIEDMAN)

i	algorithm	unadjusted p	p_{Bonf}	p_{Holm}	p_{Hoch}	p_{Homm}
1	XDEM1	2.720379E-7	$1.632227E-6^{\dagger}$	$1.632227E-6^{\dagger}$	$1.632227E-6^{\dagger}$	$1.632227E-6^{\dagger}$
2	CHC	0.004390	$0.026340^{\dagger}$	$0.021950^{\dagger}$	$0.021950^{\dagger}$	$0.021950^{\dagger}$
3	PSO	0.013706	0.082241	0.054827	0.054827	0.054827
4	DE-Bin	0.032837	0.197024	0.098512	0.071125	0.071125
5	XDEM9	0.0711256	0.426754	0.142251	0.071125	0.071125
6	SSGA	0.071125	0.426754	0.142251	0.071125	0.071125

Table 4: Adjusted p-values (F1-F25) (ALIGNED FRIEDMAN)

i	algorithm	unadjusted p	p_{Holl}	p_{Rom}	p_{Finn}	p_{Li}
1	XDEM1	2.7203795E-7	$1.632226E-6^{\dagger}$	$1.551990E-6^{\dagger}$	$1.632226E-6^{\dagger}$	$2.928683E-7^{\dagger}$
2	CHC	0.004390	$0.021758^{\dagger}$	$0.020874^{\dagger}$	$0.013112^{\dagger}$	$0.004704^{\dagger}$
3	PSO	0.013706	0.053710	0.052279	$0.027226^{\dagger}$	$0.014541^{\dagger}$
4	DE-Bin	0.032837	0.095312	0.071125	$0.048849^{\dagger}$	$0.034144^{\dagger}$
5	XDEM9	0.071125	0.137192	0.071125	0.084731	0.071125
6	SSGA	0.071125	0.137192	0.071125	0.084731	0.071125

Table 5: Adjusted p-values (F13-F25) (ALIGNED FRIEDMAN)

i	algorithm	unadjusted p	p_{Bonf}	p_{Holm}	p_{Hoch}	p_{Homm}
1	PSO	0.003168	$0.012673^{\dagger}$	$0.012673^{\dagger}$	$0.012673^{\dagger}$	$0.010706^{\dagger}$
2	CHC	0.005353	$0.021413^{\dagger}$	$0.016060^{\dagger}$	$0.016060^{\dagger}$	$0.016060^{\dagger}$
3	DE-Bin	0.012797	0.051190	$0.025595^{\dagger}$	$0.018546^{\dagger}$	$0.018546^{\dagger}$
4	SSGA	0.018546	0.074187	$0.025595^{\dagger}$	$0.018546^{\dagger}$	$0.018546^{\dagger}$

Table 6: Adjusted p-values (F13-F25) (ALIGNED FRIEDMAN)

i	algorithm	unadjusted p	p_{Holl}	p_{Rom}	p_{Finn}	p_{Li}
1	PSO	0.003168	$0.012613^{\dagger}$	$0.012084^{\dagger}$	$0.012613^{\dagger}$	$0.003217^{\dagger}$
2	CHC	0.005353	$0.015974^{\dagger}$	$0.016060^{\dagger}$	$0.012613^{\dagger}$	$0.005424^{\dagger}$
3	DE-Bin	0.012797	$0.025431^{\dagger}$	$0.018546^{\dagger}$	$0.017026^{\dagger}$	$0.012871^{\dagger}$
4	SSGA	0.018546	$0.025431^{\dagger}$	$0.018546^{\dagger}$	$0.018546^{\dagger}$	$0.018546^{\dagger}$

Taking XDEM5 as the control, a 1xN post-hoc comparison is conducted among the algorithms. The unadjusted p-values and p-values adjusted according to the eight post-hoc procedures based on the Aligned Friedman test are shown in Tables 3 and 4. Statistically significant differences at the $\alpha < 0.05$ level are denoted with †. Over the 25 test problems, XDEM5 is only statistically better than CHC and XDEM9 on all tests and better than PSO and DE-Bin using the more powerful Finner and Li post-hoc tests. In order to observe the significance of the apparently better performance of XDEM5 in the more difficult classes of problems (i.e. expanded and hybrid composition functions), we re-conduct the Aligned Friedman test using only the subset of results obtained from F13-F25. The results are presented below.

Tables 5 and 6 show the post-hoc tests conducted using XDEM5 as the control again and this time comparing against non-XDEM runs only on the subset of F13-F25 test problems representing the two most challenging classes of problems in the test suite. All of the post-hoc tests show that XDEM5's superior performance was statistically significant except for the Bonferroni-Dunn test against DE-Bin and SSGA. As such, the raw average results obtained supported by the non-parametric statistical testing conducted shows a strong indication that reversing the crossover and mutation operations in DE in combination with a suitably set mutation rate can be beneficial for the standard DE algorithm, particularly for the more difficult classes of problems in which the standard DE typically displays inferior performance.

6 Conclusion

In this study, we have reversed the order of the mutation and crossover operations in the standard DE algorithm to design a novel DE algorithm called XDEM where crossover is conducted first before mutation and also to introduce an explicitly tunable mutation rate parameter. Testing against the standard DE algorithm as well as three other commonly used global optimization heuristics over a set of 25 benchmark test problems showed that the novel XDEM algorithm was the top-ranked algorithm among all the algorithms tested and that XDEM was able to improve DE's performance in solving the most difficult classes of test problems.

Observing that having a suitable mutation rate influences the performance of XDEM, our next line of investigation will be to adapt the new mutation rate operator. It would also be informative to investigate the performance of XDEM when solving problems with higher dimensions as well as to investigate its suitability when porting to multi-objective optimization problems.

References:

[1] Janez Brest and Mirjam Sepesy Maučec. Self-adaptive differential evolution algorithm using population size reduction and three strategies. *Soft Computing*, 15(11):2157–2174, 2011.

[2] Uday K. Chakraborty. *Advances in Differential Evolution*. Springer Publishing Company, Incorporated, 1 edition, 2008.

[3] S. Das and P.N. Suganthan. Differential evolution: A survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15(1):831–836, 2011.

[4] Joaquin Derrac, Salvador Garcia, Daniel Molina, and Francisco Herrera. A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, 1(1):3 – 18, 2011.

[5] Larry J. Eshelman. The {CHC} adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination. volume 1 of *Foundations of Genetic Algorithms*, pages 265 – 283. Elsevier, 1991.

[6] Larry J. Eshelman and J. David Schaffer. Real-coded genetic algorithms and interval-schemata. In L. DARRELL WHITLEY, editor, *Foundations of Genetic Algorithms*, volume 2 of *Foundations of Genetic Algorithms*, pages 187 – 202. Elsevier, 1993.

[7] Qinqin Fan and Xuefeng Yan. Self-adaptive differential evolution algorithm with discrete mutation control parameters. *Expert Systems with Applications*, 42(3):1551 – 1572, 2015.

[8] C. Fernandes and A. Rosa. A study of non-random matching and varying population size in genetic algorithm using a royal road function. In *Proceedings of the 2001 Congress on Evolutionary Computation*, pages 60–66, 2001.

[9] Shu-Mei Guo, Chin-Chang Yang, Hsin-Yu Chang, and Jason Sheng-Hong Tsai. Constraint-activated differential evolution for constrained min-max optimization problems: Theory and methodology. *Expert Systems with Applications*, 42(3):1626 – 1636, 2015.

[10] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, volume 4, pages 1942–1948, Nov 1995.

[11] Xiaofen Lu, Ke Tang, Bernhard Sendhoff, and Xin Yao. A new self-adaptation scheme for differential evolution. *Neurocomputing*, 146(0):2 – 16, 2014. Bridging Machine learning and Evolutionary Computation (BMLEC) Computational Collective Intelligence.

[12] R. Mallipeddi, P.N. Suganthan, Q.K. Pan, and M.F. Tasgetiren. Differential evolution algorithm with ensemble of parameters and mutation strategies. *Applied Soft Computing*, 11(2):1679 – 1696, 2011. The Impact of Soft Computing for the Progress of Artificial Intelligence.

[13] Ferrante Neri and Ville Tirronen. Recent advances in differential evolution: a survey and experimental analysis. *Artificial Intelligence Review*, 33(1-2):61–106, 2010.

[14] Filippo Neri. Cooperative concept learning by means of A distributed GA. In *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference, New York, USA, 9-13 July 2002*, pages 953–956, 2002.

[15] Filippo Neri. Relational concept learning by cooperative evolution. *ACM Journal of Experimental Algorithmics*, 7:12, 2002.

[16] Filippo Neri and Lorenza Saitta. Analysis of genetic algorithms evolution under pure selection. In *Proceedings of the 6th International Conference on Genetic Algorithms, Pittsburgh, PA, USA, July 15-19, 1995*, pages 32–41, 1995.

[17] Filippo Neri and Lorenza Saitta. Exploring the power of genetic search in learning symbolic classifiers. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(11):1135–1141, 1996.

[18] Kenneth Price, Rainer M. Storn, and Jouni A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[19] R. Storn and K. Price. Differential evolution: A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, International Computer Science Institute, Berkeley, 1995.

[20] Ponnuthurai N Suganthan, Nikolaus Hansen, Jing J Liang, Kalyanmoy Deb, YP Chen, Anne Auger, and S Tiwari. Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization. *Technical Report, Nanyang Technological University, Singapore, May 2005 AND KanGAL Report 2005005, IIT Kanpur, India*, 2005.

[21] Ming Yang, Zhihua Cai, Changhe Li, and Jing Guan. An improved adaptive differential evolution algorithm with population adaptation. In *Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 145–152. ACM, 2013.