Evolving cognitive and social experience in Particle Swarm Optimization through Differential Evolution

Michael G. Epitropakis, *Member, IEEE*, Vassilis P. Plagianakos and Michael N. Vrahatis

***Abstract*— In recent years, the Particle Swarm Optimization has rapidly gained increasing popularity and many variants and hybrid approaches have been proposed to improve it. Motivated by the behavior and the proximity characteristics of the social and cognitive experience of each particle in the swarm, we develop a hybrid approach that combines the Particle Swarm Optimization and the Differential Evolution algorithm. Particle Swarm Optimization has the tendency to distribute the best personal positions of the swarm near to the vicinity of problem's optima. In an attempt to efficiently guide the evolution and enhance the convergence, we evolve the personal experience of the swarm with the Differential Evolution algorithm. Extensive experimental results on twelve high dimensional multimodal benchmark functions indicate that the hybrid variants are very promising and improve the original algorithm.**

I. Introduction

The Particle Swarm Optimization (PSO) algorithm is an Evolutionary Computation method, which belongs to the broad class of Swarm Intelligence methods. PSO was introduced in [1], is inspired by the social behavior of bird flocking and fish schooling, and is based on a social-psychological model of social influence and social learning. The fundamental hypothesis to the development of PSO is that an evolutionary advantage is gained through the social sharing of information among members of the same species. Moreover, the behavior of the individuals of a flock corresponds to fundamental rules, such as nearest-neighbor velocity matching and acceleration by distance [2], [3]. The PSO algorithm is capable of handling non-differentiable, discontinuous and multimodal objective functions and has gained increasing popularity in recent years due to its ability to efficiently and effectively tackle several real-world applications [4], [5].

To improve the performance and the convergence behavior of Particle Swarm Optimization algorithm, several variations and hybrid approaches have been proposed [6]–[11]. One class of variations include hybrids that combine the PSO and the Differential Evolution (DE) algorithms [12]. These approaches aim to aggregate the advantages of both methods to efficiently tackle the optimization problem at hand. The PSO–DE hybrids usually combine the evolution schemes of both algorithms to produce a new evolutionary position scheme [13]–[17], apply one of the two algorithms as local search to evolve some pre-specified particles [18]–[20], or evolve the control parameters with one of the evolutionary approaches to produce a parameter-free hybrid [13], [21]–[23].

The current study has been motivated by the behavior and the proximity characteristics of the personal experience of each particle during the evolution process. Each particle interacts with the rest of the swarm particles. More specifically, the movement of each particle is controlled by forces; the best previous position of the particle (cognitive experience) and the position attained by the best particle in the swarm or neighborhood (social experience). Extensive simulations indicate that through the evolution process of the PSO algorithm, the cognitive experience of each particle tend to be distributed in the vicinity of the problem's optima.

To this end, we propose a hybrid evolutionary scheme to efficient evolve the social and cognitive experience of the swarm and enhance the convergence properties of the PSO algorithm. Here, we incorporate the DE algorithm, which is a simple and compact evolutionary algorithm exhibiting good convergence characteristics. Initial experimental results in a benchmark set consisting of twelve difficult high-dimensional benchmark functions demonstrate that this is a promising approach.

The rest of the paper is organized as follows: Section II and Section III briefly describe the basic operations of the canonical PSO and the DE algorithms, respectively. In Section IV, we analyze the behavior of the cognitive and social experience in the PSO algorithm that motivated the proposed approach. In Section V we propose the new hybrid evolutionary scheme, while in Section VI, we present the experimental analysis. The paper concludes with a short discussion and some pointers for future work.

II. The Particle Swarm Optimization algorithm

The PSO algorithm is a population–based stochastic algorithm that exploits a population of individuals to effectively probe promising regions of the search space. Therefore, each individual (particle) of the population (swarm) moves with an adaptable velocity within the search space and retains in its memory the best position it ever encountered. There exist two main PSO versions; namely the *global* PSO and the *local* PSO. In the *global* PSO version, the best position ever attained by all individuals of the swarm is communicated to all the particles, while in the *local* PSO version, for each particle it is assigned a neighborhood consisting of a pre-specified number of particles and the best position ever attained by

M.G. Epitropakis and M.N. Vrahatis are with with the Department of Mathematics, University of Patras, Greece. email: <mikeagn, vrahatis>@math.upatras.gr

V.P. Plagianakos is with the Department of Computer Science and Biomedical Informatics, University of Central Greece, Greece. email: vpp@ucg.gr

All authors are members of Computational Intelligence Laboratory (CILab), Department of Mathematics, University of Patras, Greece.

the particles in their neighborhood is communicated among them [3].

More specifically, each particle is a D-dimensional vector, and the swarm consists of *NP* particles. Therefore, the position of the i-th particle of the swarm can be represented as: $X_i = (x_{i1}, x_{i2}, \ldots, x_{iD})$. The velocity of each particle is also a D-dimensional vector and for the i-th particle is denoted as: $V_i = (u_{i1}, u_{i2}, \ldots, u_{iD})$. The best previous position of the i-th particle can be recorded as: $P_i = (p_{i1}, p_{i2}, \ldots, p_{iD})$ and the best particle in the swarm (i.e. in minimization problems, the particle with the smallest fitness function value) is indicated by the index best. Furthermore, the neighborhood of each particle is usually defined through its index. The majority of the PSO variants utilize the *ring* topology which is the most common topology in the literature. In the *ring* topology, the neighborhood of each particle consists of particles with neighboring indices [24], [25].

In the present investigation, we consider the version of PSO proposed by Clerc and Kennedy [26], which incorporates the parameter χ, known as the *constriction factor*. The main role of the constriction factor is to control the magnitude of the velocities and alleviate the "swarm explosion" effect that sometimes prevented the convergence of the original PSO algorithm [27]. As stated in [26], the dynamic behavior of the particles in the swarm is manipulated using the following equations:

$$\begin{aligned} V_i(t+1) &= \chi\Big(V_i(t) + c_1 r_1\big(P_i(t) - X_i(t)\big) \\ &\quad + c_2 r_2\big(P_{\mathrm{best}}(t) - X_i(t)\big)\Big), \qquad (1) \\ X_i(t+1) &= X_i(t) + V_i(t+1), \qquad (2) \end{aligned}$$

for $i = 1, 2, \ldots, NP$, where χ is the constriction factor parameter, c_1 and c_2 are positive constants referred to as *cognitive* and *social* parameters respectively, and r_1 and r_2 are randomly chosen numbers uniformly distributed in $[0, 1]$. The cognitive parameter controls the experience influence of each particle with respect to its best performance found so far, while the social parameter with respect to the best position found by its society, i.e. either the whole swarm or its neighborhood. Furthermore, in a stability analysis provided in [26], it was implied that the constriction factor is typically calculated according to the following formula: $\chi = \frac{2\kappa}{|2-\phi-\sqrt{\phi^2-4\phi}|}$, where $\phi = c_1 + c_2$ and $k = 1$, and to guarantee the quick convergence of the scheme, the value of ϕ has to satisfy $\phi > 4$. The aforementioned scheme is typically utilized for the constant $\phi = 4.1$, with $\chi = 0.72984$ and $c_1 = c_2 = 2.05$ [24], [26].

Below, we briefly describe the basic operators of the DE algorithm.

III. The Differential Evolution algorithm

The DE algorithm [12] is a stochastic parallel direct search method, which utilizes concepts borrowed from the broad class of Evolutionary Algorithms (EAs). The DE method requires few control parameters and several experimental studies have shown that DE has good convergence properties and outperforms other well known and widely used EAs [12], [28]–[30].

More specifically, DE is a population–based stochastic algorithm that exploits a population of potential solutions, *individuals*, to effectively probe the search space. Like PSO, the population of individuals is randomly initialized in the optimization domain with *NP*, D–dimensional, vectors following a uniform probability distribution. Individuals evolve over successive iterations to explore the search space and locate the minima of the objective function. Throughout the execution process, the user–defined population size, *NP*, is fixed. At each iteration, called *generation*, new vectors are derived by the combination of randomly chosen vectors from the current population. This operation in our context can be referred to as *mutation*, while the outcoming vectors as *mutant individuals*. Each mutant individual is then mixed with another, predetermined, vector – the *target* vector – through an operation called *recombination*. This operation yields the so–called *trial* vector. Finally, the trial vector undergoes the *selection* operator, according to which it is accepted as a member of the population of the next generation only if it yields a reduction in the value of the objective function f relative to that of the target vector. Otherwise, target vector is retained in the next generation. The search operators efficiently shuffle information among the individuals, enabling the search for an optimum to focus on the most promising regions of the solution space.

Here we describe the original mutation operators proposed in [12]. Specifically, for each individual x_g^i, $i = 1, \ldots,$ *NP*, where g denotes the current generation, the mutant individual v_{g+1}^i can be generated according to one of the following equations:

1) "DE/best/1"
$$v_{g+1}^i = x_g^{\mathrm{best}} + F(x_g^{r_1} - x_g^{r_2}), \qquad (3)$$
2) "DE/rand/1"
$$v_{g+1}^i = x_g^{r_1} + F(x_g^{r_2} - x_g^{r_3}), \qquad (4)$$
3) "DE/current-to-best/1"
$$v_{g+1}^i = x_g^i + F(x_g^{\mathrm{best}} - x_g^i) + F(x_g^{r_1} - x_g^{r_2}), \qquad (5)$$
4) "DE/best/2"
$$v_{g+1}^i = x_g^{\mathrm{best}} + F(x_g^{r_1} - x_g^{r_2}) + F(x_g^{r_3} - x_g^{r_4}), \qquad (6)$$
5) "DE/rand/2"
$$v_{g+1}^i = x_g^{r_1} + F(x_g^{r_2} - x_g^{r_3}) + F(x_g^{r_4} - x_g^{r_5}), \qquad (7)$$

where x_g^{best} is the best member of the previous generation, $r_1, r_2, r_3, r_4, r_5 \in \{1, 2, \ldots, i-1, i+1, \ldots, NP\}$, are random integers mutually different and not equal to the running index i, and $F > 0$ is a real parameter, called *mutation or scaling factor*. The user–defined *mutation constant* F, controls the amplification of the difference between two individuals, and is used to prevent the risk of stagnation,

of the search process. It is also mainly responsible for the convergence rate of the algorithm. Therefore, an inappropriate mutation constant value can cause deceleration of the algorithm and decrease of the population diversity.

Furthermore, here we utilize the trigonometric mutation operator [31], which performs a mutation according to the following equation, with probability τ_μ:

6) "TDE/rand/1"

$$v_{g+1}^i = (x_g^{r1} + x_g^{r2} + x_g^{r3})/3 + (p_2 - p_1)(x_g^{r1} - x_g^{r2}) + \\ + (p_3 - p_2)(x_g^{r2} - x_g^{r3}) + (p_1 - p_3)(x_g^{r3} - x_g^{r1}) \quad (8)$$

and with probability $(1 - \tau_\mu)$, the mutation is performed according to Eq. (4). τ_μ is a user defined parameter, typically set around 0.1. The values of p_m, $m = \{1, 2, 3\}$ and p' are obtained through the following equations: $p_1 = \left|f(x_g^{r1})\right| / p'$, $p_2 = \left|f(x_g^{r2})\right| / p'$, $p_3 = \left|f(x_g^{r3})\right| / p'$, and $p' = \left|f(x_g^{r1})\right| + \left|f(x_g^{r2})\right| + \left|f(x_g^{r3})\right|$.

Having performed the mutation, the recombination operator is subsequently applied to further increase the diversity of the population. To this end, the mutant individuals are combined with other predetermined individuals, called the target individuals. Specifically, for each component l ($l = 1, 2, \ldots, D$) of the mutant individual v_{g+1}^i, we randomly choose a real number r in the interval $[0, 1]$. Then, we compare this number with the user–defined *recombination constant*, CR. If $r \leqslant CR$, then we select, as the l–th component of the trial individual u_{g+1}^i, the l–th component of the mutant individual v_{g+1}^i. Otherwise, the l–th component of the target vector x_g^i becomes the l–th component of the trial vector. This operation yields the trial individual. It is evident that if the value of the recombination constant is too small (close to zero) the effect of the mutation operator is cancelled, since the target (and not the mutant) vector will become the new trial vector.

Finally, the trial individual is accepted for the next generation only if it reduces the value of the objective function (selection operator): $u_{g+1}^i = \begin{cases} v_{g+1}^i, & \text{if} f(v_{g+1}^i) < f(x_g^i) \\ x_g^i, & \text{otherwise} \end{cases}$.

IV. Studying the Cognitive and Social Experience

Numerous PSO variations have been proposed to improve the accuracy of solutions and PSO convergence behavior [7], [25], [32]. In [11], [26] has been formally proven that each particle converges to a weighted average of its personal best and neighborhood best positions. Motivated by this finding new variants have been introduced that incorporate knowledge which exploit the best personal positions [33]. Moreover, the exploitation of the best personal experience has been incorporated in several PSO variants with multiple different methodologies. Specifically, some variants adapt the best personal positions using distributions that are based on best personal positions [33], or include a weighted sum of best personal positions [25]. Other variants incorporate update schemes that utilize information of the best personal positions by means of an average of two or more best personal positions [19], [34].

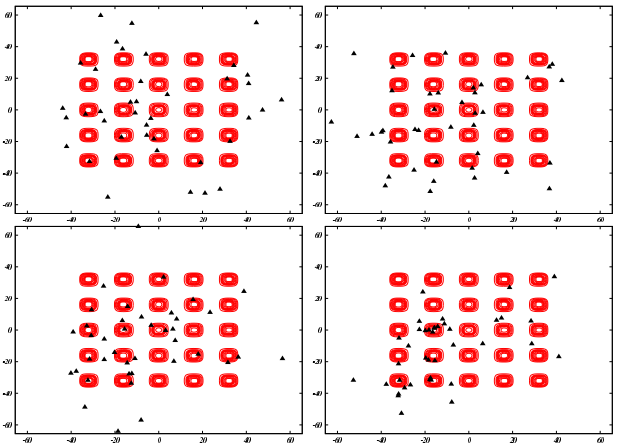

Fig. 1. local PSO population's positions after 1, 5, 10, and 20 generations

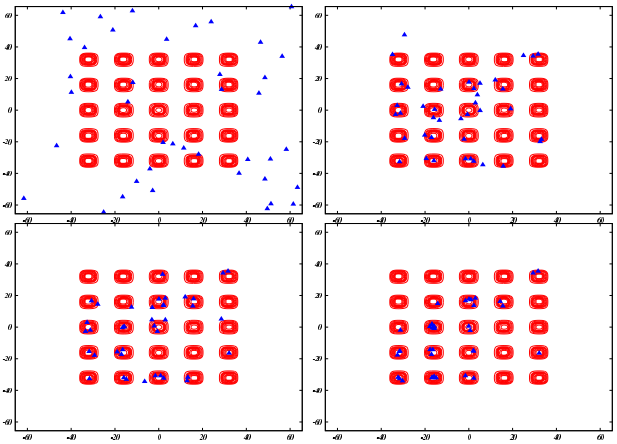

Fig. 2. local PSO population's best personal positions after 1, 5, 10, and 20 generations

The aforementioned approaches and their convergence characteristics enhance our findings. Extensive experimental simulations have verified that the PSO algorithm tends to distribute the best positions encountered by the particles in the swarm to the vicinity of a minimum. Additionally, the local version of PSO has more explorative characteristics, and tends to distribute the best personal positions to regions around many minima, while the global version of PSO exhibits more exploitive characteristics and rapidly gathers the best personal experience to the basin of attraction of a (global or local) minimum. Here, to demonstrate the evolution of the swarm, as well as the best position of each individual, we utilize the Shekel's Foxholes benchmark function, which has twenty five local minima and one global minimum ($f(-32, 32) = 0.998004$). Fig. 1 illustrates contour plots of the Shekel's Foxholes function and the positions of a swarm consisting of 40 particles that have been evolved with the local version of PSO after 1, 5, 10, and 20 generations, while Fig. 2 demonstrates the distribution of their best personal experience. It is evident that an efficient strategy to adapt or evolve the social and cognitive experience of the swarm may enhance the original PSO scheme.

Algorithm 1 The PSO algorithmic scheme utilizing Differential Evolution for evolving personal experience

```
Initialize particles in the swarm
for each time step t do
    for each particle i in the swarm do
        Update position X_i(t+1) using Eqs. (1) and (2)
        Evaluate particle X_i(t+1).
        Update social and cognitive experience
        if P_i(t) has changed position then
            /* Evolve P_i(t) utilizing one DE step in S_p*/
            Mutate P_i(t) and generate the corresponding mutant vector v_t^i through one of the mutation strategies defined by Eqs. (3) – (8)
            Recombine the mutant vector v_t^i and generate the corresponding trial vector u_t^i.
            Evaluate the trial vector u_t^i.
            Update the P_i(t) with the u_t^i, if f(u_t^i) < f(P_i(t)).
        end if
    end for
end for
```

V. The proposed approach

Motivated by the aforementioned PSO variants and our findings, we propose the evolution of the social and cognitive experience of the Particle Swarm Optimization algorithm with the Differential Evolution algorithm. In general, the evolutionary process is a very efficient procedure, but it demands a high number of function evaluations to effectively converge to an optimum. The incorporation of DE algorithm in each evolution step may result in an increase of the required function evaluations. To this end, we propose to evolve only the "promising" best personal positions of the swarm in the current generation. By "promising", we define the best personal position that has changed (improved) during the previous step and may evolve to an even better position. Extensive experimental results exhibit that the procedure of evolving only the "promising" particles is more effective than evolving the best personal positions of the whole swarm.

We define the best personal experience set as $S_p = \{P_1, P_2, \ldots, P_{NP}\}$. Hence, after each time step of the PSO algorithm, we apply one DE step to the particles that their best personal positions have improved. The DE mutation operator is using individuals from the S_p set. Specifically, the three main DE evolution steps (mutation, recombination, and selection) are applied to the promising personal bests. In the mutation procedure one of the aforementioned mutation strategies (Eqs. (3)–(8)) is utilized. A detailed algorithmic scheme of the proposed approach is illustrated in Algorithm 1.

We believe that the proposed evolutionary hybrid approach achieves a good balance between exploration and exploitation of the search space. The evolution of the personal experience will initially promote the exploration of the personal experience space while in the later time steps where the distribution of the personal best have been gathered in the vicinity of a local/global minimum, will promote exploitation of the gathered experience.

TABLE I

A brief description of the benchmark function set

	Benchmark function	n	S	$f_{\min}$
F_1	Generalized Schwefel Problem 2.26	50	$[-500, 500]^{50}$	-12569.5
F_2	Generalized Rastrigin's Function	50	$[-5.12, 5.12]^{50}$	0
F_3	Ackley's Function	50	$[-32, 32]^{50}$	0
F_4	Generalized Griewank Function	50	$[-600, 600]^{50}$	0
F_5	Generalized Penalized Function 1	50	$[-50, 50]^{50}$	0
F_6	Generalized Penalized Function 2	50	$[-50, 50]^{50}$	0
F_7	Shifted Sphere Function	n	$[-100, 100]^n$	-450
F_8	Shifted Schwefel's Problem 2.21	n	$[-100, 100]^n$	-450
F_9	Shifted Rosenbrock's Function	n	$[-100, 100]^n$	390
F_{10}	Shifted Rastrigin's Function	n	$[-5, 5]^n$	-330
F_{11}	Shifted Griewank's Function	n	$[-600, 600]^n$	-180
F_{12}	Shifted Ackley's Function	n	$[-32, 32]^n$	-140

VI. Experimental Results

This section compares the performance of the proposed PSO-DE hybrid variants with the original PSO algorithm discussed in Section II. To verify the effectiveness of the proposed approach we have used twelve widely known high dimensional benchmark functions with different characteristics. These function are from two recently proposed benchmark test sets [35], [36]. The first set of six test functions $(F_1 - F_6)$ are high dimensional multimodal functions, where the number of local minima increases exponentially with their dimensionality [35]. The remaining six test functions are high dimensional and scalable benchmark functions; two shifted unimodal (F_7 and F_8) and four shifted multimodal benchmark functions (F_9–F_{12}) [36]. A brief description of the functions is provided in Table I. More specifically, n denotes the dimensionality of the function, S is the optimization range box, and $f_{\min}$ stands for the global minimizer value. A detailed description of the benchmark functions can be found in [35], [36].

In this section, we report results averaged over 50 independent simulations. For each simulation and each PSO variant we have initialized the swarms using a uniform random number distribution with the same random seeds. Furthermore, all PSO variants have been implemented with the default parameters settings, i.e. a ring topology, $\phi = 4.1$, $\chi = 0.72984$ and $c_1 = c_2 = 2.05$ [24], [26]. Regarding the DE control parameters, the common settings of $F = 0.5$ and $CR = 0.9$ were used for all hybrid variants [12], [28].

To evaluate the efficiency and effectiveness of the hybrid PSO-DE variants against the respective original PSO variants, we utilized the $F_1 - F_6$ 50-dimensional benchmark functions to calculated two performance measures: the Success Rate (SR) and the Success Performance (SP) [28], [29], [37]. The SR is defined as the fraction of the number of times the algorithm has reached the global optimum during a pre-specified budget of function evaluations over the total number of simulations and the Success Performance measure

TABLE II

FIRST EXPERIMENTAL SET FOR THE ORIGINAL AND THE HYBRID PSO VARIANTS OVER THE SIX 50-DIMENSIONAL BENCHMARK FUNCTIONS

Algorithm	Gen. Schwefel Prob. 2.26			Gen. Rastrigin's Function			Ackley's Function		
	NFE	SR	SP	NFE	SR	SP	NFE	SR	SP
lPSO	996200	0.02	49810000	N/A	0	N/A	145637	1	145637
lPSO:DE/best/1	153914	1	153914	N/A	0	N/A	136215	1	136215
lPSO:DE/rand/1	178106	1	178106	831879	0.02	41594000	141219	1	141219
lPSO:DE/current-to-best/1	140063	1	**140063**	N/A	0	N/A	129502	1	**129502**
lPSO:DE/best/2	167644	1	167644	639785	0.12	5331540	142887	1	142887
lPSO:DE/rand/2	173306	1	173306	720542	0.3	**2401810**	145349	1	145349
lPSO:TDE/rand/1	163135	1	163135	648835	0.06	10813900	140335	1	140335
gPSO	140370	0.1	1403700	N/A	0	N/A	N/A	0	N/A
gPSO:DE/best/1	131156	0.42	312277	N/A	0	N/A	108258	0.46	**235344**
gPSO:DE/rand/1	103045	0.82	125665	N/A	0	N/A	108877	0.2	544385
gPSO:DE/current-to-best/1	112455	0.56	200813	N/A	0	N/A	110277	0.3	367589
gPSO:DE/best/2	99692	0.66	151049	N/A	0	N/A	108041	0.2	540206
gPSO:DE/rand/2	83491	0.76	**109858**	N/A	0	N/A	106611	0.2	533054
gPSO:TDE/rand/1	86529	0.64	135202	N/A	0	N/A	105347	0.24	438944
Algorithm	Gen. Griewank Function			Gen. Penalized Function 1			Gen. Penalized Function 2		
	NFE	SR	SP	NFE	SR	SP	NFE	SR	SP
lPSO	119284	0.92	129656	133156	0.96	138704	104344	0.88	118573
lPSO:DE/best/1	91622	0.82	111734	86470	0.96	90073	92763	0.96	96628
lPSO:DE/rand/1	93844	0.9	104271	90421	0.96	94188	96992	1	96992
lPSO:DE/current-to-best/1	93156	0.9	**103507**	80732	0.98	**82379**	86735	0.98	**88505**
lPSO:DE/best/2	103653	0.96	107971	92052	0.98	93931	98407	0.96	102508
lPSO:DE/rand/2	98186	0.94	104454	94726	1	94726	102005	0.98	104086
lPSO:TDE/rand/1	94641	0.86	110048	90448	1	90448	98313	0.98	100320
gPSO	60030	0.1	600300	95909.5	0.42	228356	89780	0.36	249390
gPSO:DE/best/1	67977	0.5	135956	87787	0.54	**162570**	77619	0.54	143740
gPSO:DE/rand/1	66482	0.46	144526	82307	0.44	187063	75760	0.5	151520
gPSO:DE/current-to-best/1	66871	0.56	119414	88979	0.54	164776	76605	0.58	132078
gPSO:DE/best/2	66187	0.66	**100285**	83094	0.48	173113	73441	0.64	**114752**
gPSO:DE/rand/2	65453	0.34	192509	84892	0.38	223402	75767	0.64	118387
gPSO:TDE/rand/1	65415	0.34	192398	87444	0.34	257188	73423	0.62	118426

is defined as the fraction of the mean Number of Function Evaluations (NFE) over the SR measure. In this set, we incorporate 100 particles in each swarm and a budget of $1,000,000$ function evaluations for each simulation.

Table II, for each algorithm and each benchmark function, illustrates the average number of function evaluations, and the SR and SP measures. It is evident that the proposed PSO-DE hybrid variants for both the global (gPSO) and the local (lPSO) versions of PSO exhibit superior performance with respect to the SP measure. The incorporation of the DE algorithm to evolve the personal experience speeds up convergence and in most of the cases improves the success rate (F_1, F_3, F_4, F_5, F_6). It has to be noted that in cases where the original PSO algorithm fails to converge, the hybrid variants improve the convergence rates (local version in F_2, global version in F_3 and F_4). Another interesting observation is that the lPSO:DE/current-to-best/1 variant exhibits the best success performance in almost every benchmark functions. The first experimental results show that the exploitive DE mutation strategies, i.e. DE/best/1, DE/current-to-best/1, DE/best/2, and TDE/rand/1 exhibit better success performance.

Furthermore, the second experimental set includes extensive simulations over the remaining six shifted high dimensional benchmark functions. Bellow, we report experimental results for three different dimensions, i.e. $n = 50, 100,$ and 500. Hence, for the case of 50-dimensional functions, we report results in Table III. More specifically, for each algorithm and each benchmark, the best solution achieved after a budget of $(5000 \cdot n)$ function evaluations was recorded along with its function value. The obtained function values were analyzed statistically in terms of their mean value and standard deviation averaged over the 50 independent experiments. Moreover, Fig. 3 and Fig. 4 illustrate boxplots for the 100-dimensional and 500-dimensional benchmark functions based on the best fitness value that have reached within the available budget $(5,000 \cdot n)$. In Figs. 3 and 4, the labels of the x-axis correspond to the following PSO variants: label 1 represent the lPSO, labels 2–7 the PSO hybrids with DE mutation strategies (Eqs. (3)–(8)), label 8 the gPSO and 9–14 the hybrids with DE mutation strategies (Eqs. (3)–(8)).

Table III, as well as Figs. 3 and 4 clearly illustrate the superior performance of the proposed hybrid PSO-DE variants. It has to be noticed that although the function dimensionality increases the hybrid approaches efficiently evolve the social and cognitive experience resulting in superior performance against the corresponding original PSO versions.

Furthermore, hypothesis tests were conducted for all ex-

TABLE III

EXPERIMENTAL RESULTS FOR ALL PSO VARIANTS OVER THE SIX SHIFTED 50-DIMENSIONAL BENCHMARK FUNCTIONS ($F_7 - F_{12}$)

Algorithm	F_7 Function		F_8 Function		F_9 Function	
	Mean	*St.D.*	*Mean*	*St.D.*	*Mean*	*St.D.*
lPSO local	1490.930	384.833	-445.116	1.413	403.897	29.539
lPSO:DE/best/1	870.261	524.258	-441.640	2.542	391.755	1.999
lPSO:DE/rand/1	790.364	432.002	-449.945	0.015	390.784	1.308
lPSO:DE/current-to-best/1	742.221	459.075	-444.860	1.982	**390.706**	1.449
lPSO:DE/best/2	896.027	458.625	-449.907	0.027	390.811	1.548
lPSO:DE/rand/2	796.772	398.621	-449.409	0.111	403.587	20.682
lPSO:TDE/rand/1	**722.300**	423.164	**-449.968**	0.009	391.431	2.112
gPSO global	2672.500	1105.130	-447.362	1.513	392.328	3.253
gPSO:DE/best/1	877.294	392.263	-438.602	3.068	391.276	1.878
gPSO:DE/rand/1	878.014	436.157	-449.972	0.010	**390.683**	1.483
gPSO:DE/current-to-best/1	1015.170	399.525	-442.894	2.555	392.426	9.694
gPSO:DE/best/2	932.834	459.471	-449.917	0.035	391.043	1.763
gPSO:DE/rand/2	898.407	464.747	-449.755	0.066	391.347	1.850
gPSO:TDE/rand/1	**847.355**	432.280	**-449.982**	0.007	390.817	1.613
Algorithm	F_{10} Function		F_{11} Function		F_{12} Function	
	Mean	*St.D.*	*Mean*	*St.D.*	*Mean*	*St.D.*
lPSO local	-261.306	12.841	-175.420	1.207	-137.895	0.551
lPSO:DE/best/1	-324.311	2.252	**-179.386**	0.278	**-139.998**	0.001
lPSO:DE/rand/1	-325.496	2.063	-179.270	0.282	-139.997	0.002
lPSO:DE/current-to-best/1	-322.051	3.385	-179.384	0.272	**-139.998**	0.001
lPSO:DE/best/2	-324.128	2.664	-179.154	0.234	-139.997	0.002
lPSO:DE/rand/2	-323.308	2.984	-179.133	0.239	**-139.998**	0.002
lPSO:TDE/rand/1	**-326.161**	2.008	-179.216	0.292	-139.997	0.003
gPSO local	-248.717	19.044	-172.514	3.036	-135.075	1.481
gPSO:DE/best/1	-320.930	4.453	**-179.288**	0.304	**-139.997**	0.005
gPSO:DE/rand/1	**-325.703**	2.318	-179.090	0.284	**-139.997**	0.003
gPSO:DE/current-to-best/1	-312.281	6.236	-179.275	0.235	**-139.997**	0.005
gPSO:DE/best/2	-322.159	5.431	-179.058	0.224	-139.996	0.008
gPSO:DE/rand/2	-321.910	3.382	-178.940	0.222	**-139.997**	0.003
gPSO:TDE/rand/1	-325.014	2.382	-179.129	0.257	-139.996	0.007

perimental results to evaluate and ensure their statistical significance. Therefore, for each benchmark function, each hybrid PSO-DE variant was compared against its corresponding original PSO version using the nonparametric Wilcoxon rank-sum test. The null hypothesis in each test was that the samples compared originate from the same continuous distributions with equal medians. Due to lack of space, we report that almost all simulation results reject the null hypothesis in a 95% level of significance.

Finally, to provide a cumulative comparison over all the local and global PSO variants, we have utilized the Empirical Cumulative probability Distribution Function (ECDF) [37] of the best fitness values for all 500-dimensional benchmark functions. Fig. 5 illustrates the ECDF of the best fitness values of the original global PSO versus its hybrid PSO-DE variants (top) and the ECDF of the best fitness values of the original local PSO versus its hybrid PSO-DE variants (bottom). More successful approaches have low best fitness values and large values of the empirical cumulative probability distribution. In other words, the algorithm having the maximum area under its curve exhibits the best success performance over all benchmark functions. Studying the figures, it becomes evident that the original PSO variants are significantly improved by the evolution of the social and cognitive experience over all benchmark functions. All hybrid PSO-DE variants have improved their corresponding original PSO schemes with the most promising hybrids to be the gPSO:DE/rand/1 and gPSO:TDE/rand/1 for the global PSO version and lPSO:DE/rand/1 and lPSO:TDE/rand/1 for the local PSO version.

VII. CONCLUSIONS

In the present study, a hybrid approach of the Particle Swarm Optimization and the Differential Evolution algorithm has been presented. Motivated by the tendency of PSO to distribute the best personal positions of the swarm near to the vicinity of problem's optima, we evolve the social and cognitive experience of the swarm with the DE algorithm to further enhance the original PSO convergence characteristics. To achieve this, after each PSO evolution step, we evolve the best personal experience of the particles that have changed position during the previous step, with one of the six well known DE mutation strategies. Extensive experimental results on twelve high dimensional multimodal benchmark functions, indicate that the hybrid PSO variants are very promising, since always outperform and improve the original PSO algorithm. Future investigation should employ other evolutionary algorithms, more recent DE mutation strate-

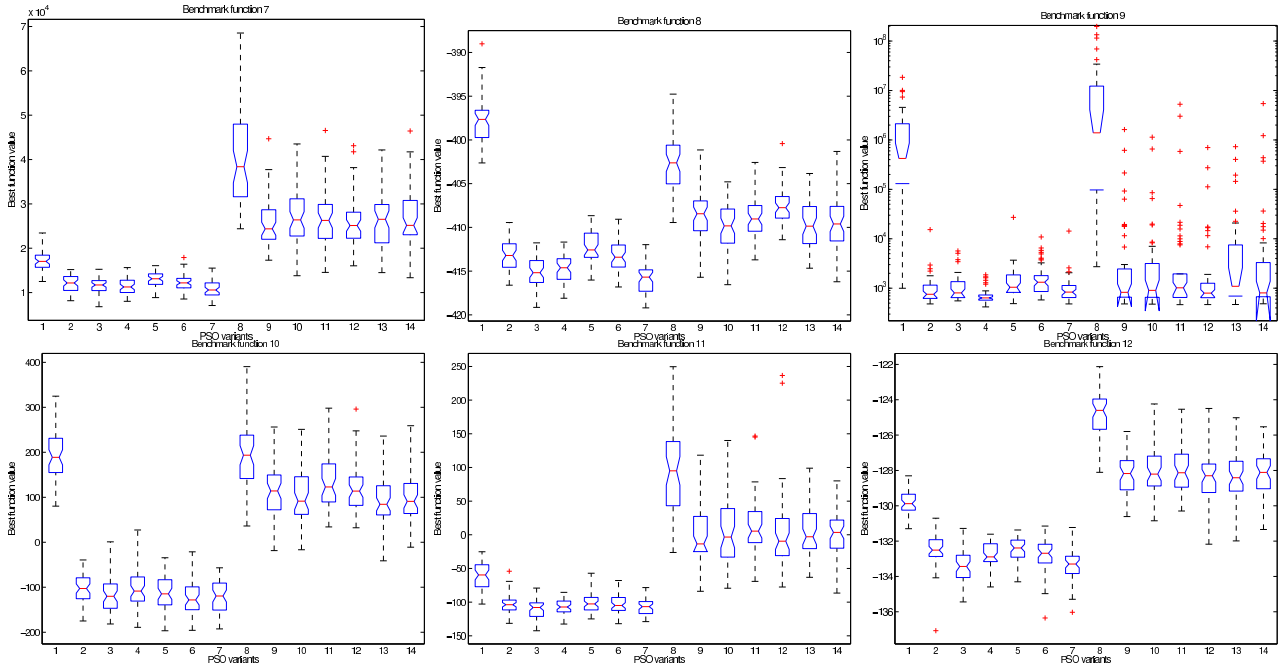

Fig. 3. Experimental results for all PSO variants over the six shifted 100-dimensional benchmark functions ($F_7 - F_{12}$)

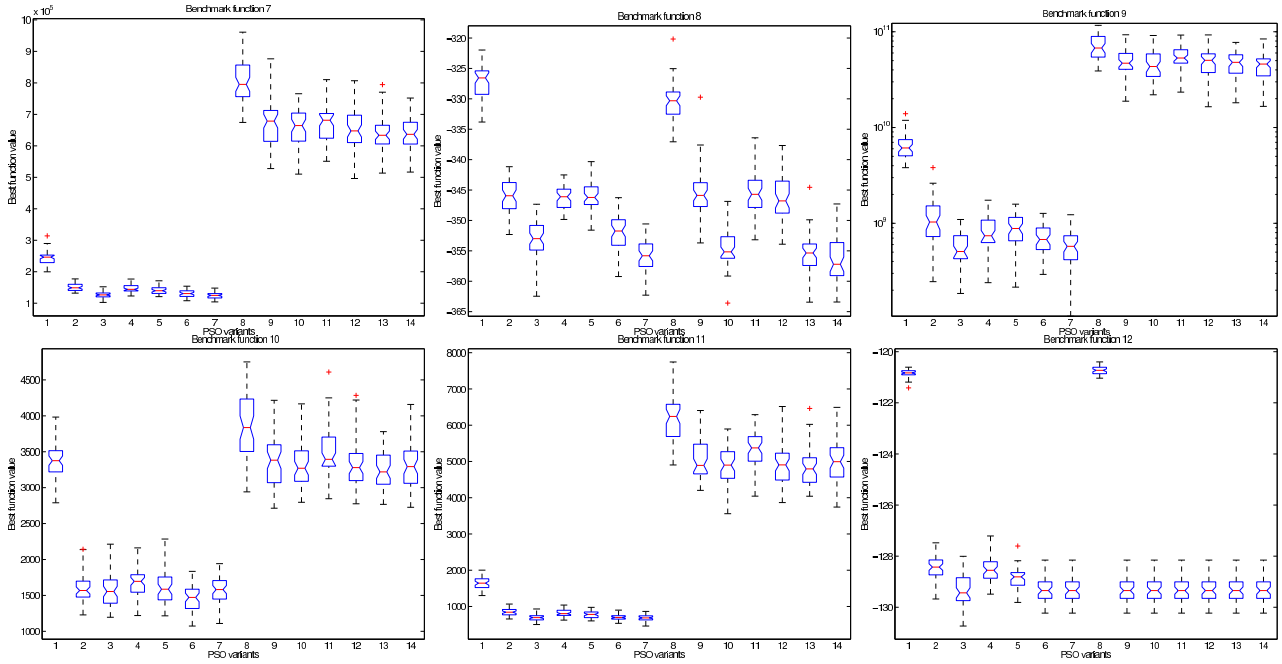

Fig. 4. Experimental results for all PSO variants over the six shifted 500-dimensional benchmark functions ($F_7 - F_{12}$)

gies [23], [38], as well as a self adaptive strategy selection technique to alleviate the hybrid approach of selecting a prespecified mutation strategy.

REFERENCES

[1] R. C. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *In Proceedings 6th Symposium on Micro Machine and Human Science*, Nagoya, Japan, 1995, pp. 39–43.

[2] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings IEEE International Conference on Neural Networks*, vol. IV. Piscataway, NJ: IEEE Service Center, 1995, pp. 1942–1948.

[3] R. Eberhart, P. Simpson, and R. Dobbins, *Computational intelligence PC tools*. San Diego, CA, USA: Academic Press Professional, Inc., 1996.

[4] M. Clerc, *Particle Swarm Optimization*. ISTE Publishing Company, 2006.

[5] A. Engelbrecht, *Computational Intelligence: An Introduction*. Halsted Press, 2002.

[6] K. E. Parsopoulos and M. N. Vrahatis, "Recent approaches to global

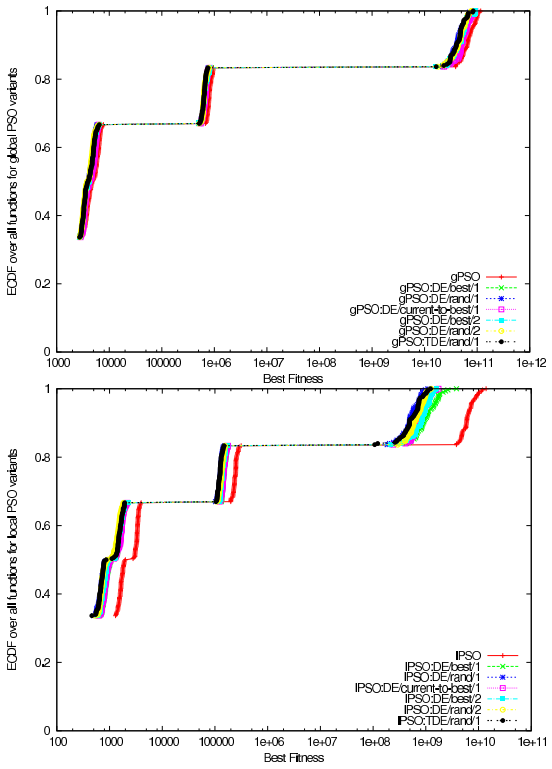

Fig. 5. Empirical cumulative probability distribution of the best fitness values of the original PSO versions against the corresponding Hybrid PSO variants over all 500–dimensional benchmark functions.

optimization problems through particle swarm optimization," *Natural Computing*, vol. 1, no. 2-3, pp. 235–306, 2002.

[7] ——, "UPSO: A unified particle swarm optimization scheme," in *Lecture Series on Computer and Computational Sciences, Proceedings of the International Conference of "Computational Methods in Sciences and Engineering" (ICCMSE 2004)*, vol. 1, 2004, pp. 868–873.

[8] T. Krink and M. Lovbjerg, "The LifeCycle model: Combining particle swarm optimisation, genetic algorithms and HillClimbers," in *Parallel Problem Solving from Nature PPSN VII*, 2002, pp. 621–630.

[9] J. Robinson, S. Sinton, and Y. Rahmat-Samii, "Particle swarm, genetic algorithm, and their hybrids: optimization of a profiled corrugated horn antenna," in *Antennas and Propagation Society International Symposium, 2002. IEEE*, vol. 1, 2002, pp. 314–317 vol.1.

[10] T. Blackwell and P. Bentley, "Improvised music with swarms," in *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, vol. 2, 2002, pp. 1462–1467.

[11] F. van den Bergh and A. Engelbrecht, "A cooperative approach to particle swarm optimization," *Evolutionary Computation, IEEE Transactions on*, vol. 8, no. 3, pp. 225–239, 2004.

[12] R. Storn and K. Price, "Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[13] M. Omran, A. Engelbrecht, and A. Salman, "Differential evolution based particle swarm optimization," in *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, 2007, pp. 112–119.

[14] S. Das, A. Abraham, and A. Konar, "Particle swarm optimization and differential evolution algorithms: Technical analysis, applications and hybridization perspectives," in *Advances of Computational Intelligence in Industrial Systems*, 2008, pp. 1–38.

[15] Z. Hao, G. Guo, and H. Huang, "A particle swarm optimization algorithm with differential evolution," in *Machine Learning and Cybernetics, 2007 Int. Conference on*, vol. 2, 2007, pp. 1031–1035.

[16] M. Pant, R. Thangaraj, C. Grosan, and A. Abraham, "Hybrid differential evolution - particle swarm optimization algorithm for solving global optimization problems," in *Digital Information Management, 2008. ICDIM 2008. Third Int. Conference on*, 2008, pp. 18–24.

[17] C. Zhang, J. Ning, S. Lu, D. Ouyang, and T. Ding, "A novel hybrid differential evolution and particle swarm optimization algorithm for unconstrained optimization," *Operations Research Letters*, vol. 37, no. 2, pp. 117–122, Mar. 2009.

[18] S. Kannan, S. M. R. Slochanal, P. Subbaraj, and N. P. Padhy, "Application of particle swarm optimization technique and its variants to generation expansion planning problem," *Electric Power Systems Research*, vol. 70, no. 3, pp. 203–210, Aug. 2004.

[19] W. Zhang and X. Xie, "DEPSO: hybrid particle swarm with differential evolution operator," in *Systems, Man and Cybernetics, 2003. IEEE International Conference on*, vol. 4, 2003, pp. 3816–3821 vol.4.

[20] T. Hendtlass, "A combined swarm differential evolution algorithm for optimization problems," in *Engineering of Intelligent Systems*, 2001, pp. 11–18.

[21] A. Salman, A. P. Engelbrecht, and M. G. Omran, "Empirical analysis of self-adaptive differential evolution," *European Journal of Operational Research*, vol. 183, no. 2, pp. 785–804, Dec. 2007.

[22] M. Omran, A. Salman, and A. Engelbrecht, "Self-adaptive differential evolution," in *Computational Intelligence and Security*, 2005, pp. 192–199.

[23] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Evolutionary adaptation of the differential evolution control parameters," in *Evolutionary Computation, 2009. CEC '09. IEEE Congress on*, 2009, pp. 1359–1366.

[24] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Swarm Intelligence Symposium, 2007. SIS 2007. IEEE*, 2007, pp. 120–127.

[25] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better," *Evolutionary Computation, IEEE Transactions on*, vol. 8, no. 3, pp. 204–210, June 2004.

[26] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *Evolutionary Computation, IEEE Transactions on*, vol. 6, no. 1, pp. 58–73, 2002.

[27] P. J. Angeline, "Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences," in *EP '98: Proceedings of the 7th International Conference on Evolutionary Programming VII*. London, UK: Springer-Verlag, 1998, pp. 601–610.

[28] U. K. Chakraborty, *Advances in Differential Evolution*. Springer Publishing Company, Incorporated, 2008.

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

[30] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *IEEE Congress on Evolutionary Computation (CEC 2004)*, vol. 2, 2004, pp. 1980–1987.

[31] H. Y. Fan and J. Lampinen, "A trigonometric mutation operation to differential evolution," *Journal of Global Optimization*, vol. 27, pp. 105–129, 2003.

[32] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 211–224, June 2004.

[33] J. Kennedy, "Bare bones particle swarms," in *Swarm Intelligence Symposium, 2003. SIS '03. IEEE*, 2003, pp. 80–87.

[34] H. Talbi and M. Batouche, "Hybrid particle swarm with differential evolution for multimodal image registration," in *Industrial Technology, 2004. IEEE ICIT '04. 2004 IEEE International Conference on*, vol. 3, 2004, pp. 1567–1572 Vol. 3.

[35] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *Evolutionary Computation, IEEE Transactions on*, vol. 3, no. 2, pp. 82–102, 1999.

[36] K. Tang et. al., "Benchmark functions for the cec'2008 special session and competition on large scale global optimization," Nature Inspired Computation and Applications Laboratory, USTC, China, Tech. Rep., 2007. [Online]. Available: http://nical.ustc.edu.cn/cec08ss.php

[37] T. Bartz-Beielstein, *Experimental Research in Evolutionary Computation: The New Experimentalism*, 1st ed. Springer, April 2006.

[38] M. G. Epitropakis, V. P. Plagianakos, and M. N. Vrahatis, "Balancing the exploration and exploitation capabilities of the differential evolution algorithm," in *Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on*, 2008, pp. 2686–2693.