Hybrid Harmony Search algorithm for Global Optimization

M. Ammar, S. Bouaziz, Adel M. Alimi
REsearch Group on Intelligent Machines (REGIM),
University of Sfax, National School of Engineers (ENIS),
BP 1173, Sfax 3038, Tunisia,
marwaa_daoued@gmail.com, souhir.bouaziz@ieee.org,
adel.alimi@ieee.org

Ajith Abraham[1,2]
[1]Machine Intelligence Research Labs, WA, USA
[2]IT4Innovations, VSB-Technical University of Ostrava,
Czech Republic
ajith.abraham@ieee.org

***Abstract*—This paper proposes two hybrid optimization methods based on Harmony Search algorithm (HS) and two different nature-inspired metaheuristic algorithms. In the first contribution, the combination was between the Improved Harmony Search (IHS) and the Particle Swarm Optimization (PSO). The second contribution merged the IHS with the Differential Evolution (DE) operators. The basic idea of hybridization was to ameliorate all the harmony memory vectors by adapting the PSO velocity or the DE operators in order to increase the convergence speed. The new algorithms (IHSPSO and IHSDE) have been compared to the IHS, DE, PSO and some other algorithms like DHS and HSDM. The DHS and HSDM are two existing algorithms, which use different hybridization concepts between HS and DE. All of these algorithms have been evaluated by different test Benchmark functions. The results demonstrated that the hybrid algorithm IHSDE have the better convergence speed into the global optimum than the IHSPSO and the standard IHS, DE and PSO.**

Keywords-component; Harmony Search; Improved Harmony Search; Differential Evolution; Particle Swarm Optimization; Benchmark functions.

I. INTRODUCTION

In recent years, several researchers have devoted their attention to develop new optimization algorithms based on analogies with natural or behavioral phenomena. The field of nature-inspired metaheuristic algorithms was principally constituted by the evolutionary algorithms like Genetic Algorithm (GA) [7] and Differential Evolution (DE) [14], as well as the swarm intelligence algorithms like Particle Swarm Optimization (PSO) [9], Bacterial Foraging Optimization (BFO) [11], and so on [19-27]. These algorithms have demonstrated their power in solving global complex optimization problems among whom learning of artificial neural networks [1, 2], Optimal Power Flow problem [16], Power Electronic Circuit Design [18], etc.

In 2001, the field extends to include the mimic algorithm, Harmony Search (HS), developed by Geem [6]. This new algorithm was inspired from jazz musical improvisation when a musician (=decision variable) plays (= generate) a note (= value) to find a perfect state of harmony (= global optimum)[6].

Since its invention, (HS) has received considerable attentions. Its effectiveness and advantages have been demonstrated in a wide range of applications [5], which directed research to further improve its performance. Moreover, in order to improve the adjusting characteristic of HS algorithm, Mahdavi *et al.* [10] suggested evolving the parameters instead of being fixed during the iterations. The Improved Harmony Search algorithm (IHS) was applied in various standard engineering optimization problems.

Harmony Search (HS) is a phenomenon imitating an algorithm inspired by the improvisation process of musicians. The HS algorithm searches the solution area as a whole to find the optimum vector, which optimizes the objective function [6]. When the HS algorithm generates a new vector, it considers all of the existing vectors in the harmony memory with fewer mathematical requirements. This feature makes the HS more flexible, the implementation easier and it is very versatile to combine HS with other metaheuristic algorithms [17] such as Differential Evolution algorithm [14] and Particle Swarm Optimization algorithm [9].

All these factors pushed some researchers like Chakraborty *et al.* to propose hybridization between the HS and the differential evolution algorithm called the improved harmony search algorithm with differential mutation operator (DHS) [3]. In addition, Qin and Forbes presented the Harmony Search with Differential Mutation Based Pitch Adjustment (HSDM) [12].

In this context, some nature inspired meta-heuristic optimization algorithms such as PSO, DE and HS are adopted in this work. In the first phase, a new idea, which approximates the vectors of IHM to the swarm concept of PSO algorithm is introduced. In this case, at each iteration, a new position vector was computed for all individuals of the swarm to converge it to the global minimum. The IHSPSO algorithm inherited a new attribute named 'Velocity' and integrated it for the computation of the new vectors of harmony memory. In the second phase, the Differential Evolution (DE) algorithm was chosen to implement their operators (mutation, crossover and selection) in IHS generation vectors. By applying these instructions, a wide variety of values were being available to guide the hybridized algorithm IHSPSO and IHSDE towards the optimal solutions with more efficiency and speed. This work considers only the single-objective optimization problems.

The remaining paper is organized as follows: Section 2 describes the original HS, the improved HS and the observed weakness of these algorithms. In section 3, the hybrid method based on the IHS and PSO is presented. The combination method between the IHS and DE algorithms is provided in Section 4. The set of some simulation results is the subject of Section 5. Finally, some concluding remarks are presented in Section 6.

II. HARMONY SEARCH (HS)

This section contains a description of the basic Harmony Search algorithm; the improved method and the weakness on witch based our hybridizations.

A. *The Harmony Search algorithm*

In order to understand the Harmony Search concept, some explications of the improvisation process by a skilled musician are the subject of this section. When a musician improvises a note usually follows one of the three rules: (1) playing a note from his memory, (2) playing a note beside a note from his memory, or (3) playing a note totally random of the sound and feasible range. Similarly, the improvisation of harmony (vector) is essentially based on these rules. The steps in the procedure of harmony search are as follows [6]:

1) Step 1. Formulation of the problem and parameter settings.

Thus, to apply the HS, the problems should be formulated in the optimization environment, with the objective function and the parameters must be defined with certain values. The HS algorithm parameters are [5, 6]:

- Harmony Memory Consideration Rate (HMCR) : the rate of randomly selected values from the memory ($0 \leq HMCR \leq 1$)
- Harmony Memory Size (HMS) (that is, equivalent to population size),
- Pitch Adjustment Rate (PAR) : the rate of altered values that was originally taken from the memory ($0 \leq PAR \leq 1$)
- Number of Improvisations (NI) (that is, the maximum number of generations).
- FW or BW: the width of the fret or bandwidth

2) Step 2. Initialize randomly the Harmony Memory (HM).

3) Step 3. Improvise a new harmony.

4) Step 4. Update the harmony memory.

5) Step 5. Repeat step 3 step 4 until the satisfaction of the termination criterion.

B. *The Improved Harmony Search (IHS)*

In the Improved Harmony Search (IHS), Mahdavi [10] suggested that PAR increase linearly and FW decrease exponentially with iterations. Therefore, mathematic expressions were adapted into these parameters to follow the iteration change:

$$PAR=(PARmax-PARmin)/(MaxItr) *currentIteration + PARmin \quad (1)$$

$$FW=fwmax*exp(coef*currentIteration) \quad (2)$$

$$coef=log(fwmin/fwmax)/MaxItr \quad (3)$$

C. *The HS weakness*

Most of the decision variables in the new harmony are selected from the other vectors stored in Harmony Memory. In addition, the new harmony vector may have the opportunity to take a place in the memory after its fitness test. Then, this vector might influence the convergence speed of the HS to the global optimum. In simulation results, we note that the HM is stable in most of the time: the memory matrix is changed one time every 75 iterations, in average. So it does not provide a large variety of values to the next improvisation. Therefore, the HS has a low probability of generating a good-quality of the new harmony vector.

To overcome this limitation in the HS, we have to incorporate a mechanism to create a wide variety of values in memory while respecting their allowable ranges. This mechanism must be dynamic, so that converges and indirectly guides the global algorithm to find its optimum.
For these reasons, we try to inspire a new hybridization idea from other nature-inspired metaheuristic algorithms like Particle Swarm Optimization (PSO) and Differential Evolution (DE) algorithms. The common idea of these algorithms is to provide a new population at each iteration not only completely different but also closer to the optimum.

The hybridization strategy is to simulate the HM vectors to the swarm particles performance of PSO and to the individuals' evolution of DE. Therefore, we try to apply a new set of instructions on all vectors in memory to have a dynamic memory that fly at each iteration towards the optimal solution. In addition, it improves the population and generates each time a better range of values for the next improvisation.

III. THE IHS HYBRIDIZED WITH PSO ALGORITHM

This section contains a description of the Particle Swarm Optimization algorithm (PSO) and the process of hybridization between IHS and PSO algorithms.

A. *The Particle Swarm Optimization (PSO)*

The particle swarm optimization (PSO) is a population-based metaheuristic algorithm. It was developed by Kennedy and Eberhart in 1995 [9]. Simulating the behaviors of bird flocking, the mean idea of PSO is that individuals, called particles, interact with one another while learning from their own experience. The system is initialized randomly with a population of solutions and searches for optimal solution by updating generations. They share the global best and gradually they move into better regions of the problem space [9]. All of particles have positions and velocities which direct the flying of the particles. They evaluated by the fitness values which computed by the optimized function.

The PSO algorithm requires primitive mathematical operators for updating the particles positions ‘p’ and velocities ‘v’ as shown below [8]:

$$v_i^{t+1} = v_i^t + c1 * rand * (pbest_i - p_i^t) + c2 * rand * (gbest_i - p_i^t) \quad (4)$$

$$p_i^{t+1} = p_i^t + v_i^{t+1} \quad (5)$$

At each iteration, every particle is updated by following two "best" values. The first one is the local best and called ‘pbest’. It is the best fitness value achieved by the particle. The second "best" is the global best and called ‘gbest’. It is the best fitness value obtained so far by any particle in the population.

B. The hybridization between IHS and PSO

From the early works on PSO, it is known that PSO algorithm have fast convergence behavior and characterized by its ability to perform very well in static and dynamic environments. The stochastic factors and the dynamic aspects of particle velocities can guide the system to the right areas of research in the workspace.

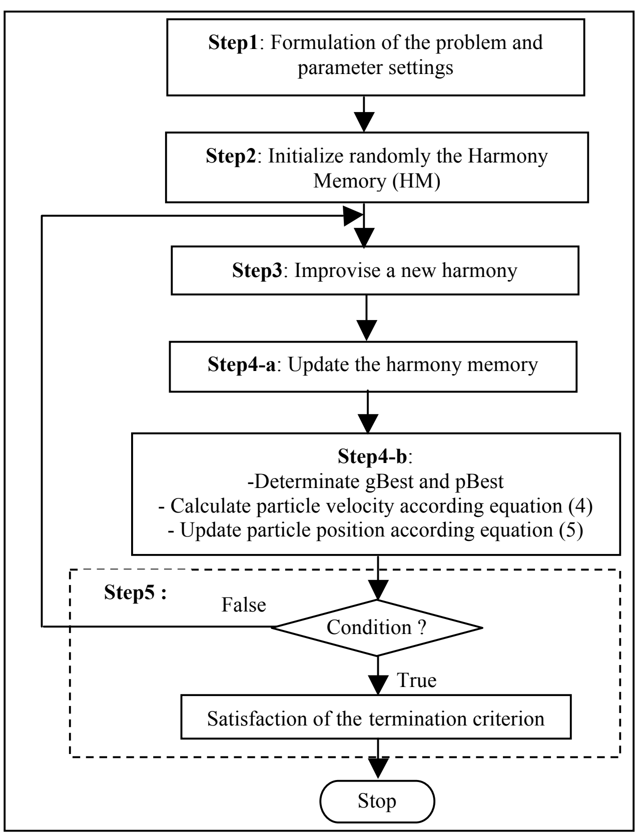

Figure 1. Flowchart of IHSPSO

In this hybridization, we integrate the terminology of the PSO algorithm in the HS in order to limit the search time for the optimum. For better results, we choose to apply the hybridization on the Improved Harmony Search algorithm instead of the basic version because of its better performance.

Indeed, we considered the memory vectors of IHS as particles of the swarm and the new memory values for new improvisation as the new positions reached by these particles. We added the ‘velocity’ parameter calculated for each particle according to the equation (4) announced in the PSO algorithm. For each iteration, we identified 'pBest' and 'gBest', which correspond to the current particles generation and calculated the new positions in relation to the calculated velocities (see Figure 1).

IV. THE IHS HYBRIDIZED WITH DE ALGORITHM

This section presents a general idea about the Differential Evolution algorithm (DE) and describes the new hybridization strategy between IHS and DE algorithm.

A. Differential Evolution (DE)

The Differential Evolution algorithm (DE) was proposed by Price and Storn in 1995 [14]. Its remarkable performance and effective approach as a global optimization algorithm on wide variety of fields of engineering has been extensively explored [1, 2]. It is a simple and straightforward strategy based on three operators: mutation, crossover and replacement [4].

- Mutation:

There exist different mutation strategies. In one of the simplest forms of DE-mutation, for each target vector of the current population, three distinct vectors are sampled randomly. Then, the vector difference of randomly sampled population members is scaled (by the control parameter F in the range [0.4, 1]) and added to the basis vector to produce a mutant vector.

- Crossover:

After the mutation, a crossover operation comes into play. The crossover is applied with certain probability controlled by the Crossover rate (Cr $\in$ [0, 1]). The crossover is a combination between the mutant vector and the target vector under consideration to generate a trial vector.

- Selection:

To keep the population size constant for future generations, the next operation of the algorithm calls selection. The goal of selection is to keep the best vector for the next generation.

B. The hybridization between IHS and DE

The Differential Evolution was the second algorithm hybridized with IHS. It has very limited number of control parameters (Cr, F, and NP in classical DE). Although it used simple adaptation formulas for F and Cr without computational burden, it was a preferment and effective technique for solving optimization problems.

In order to build an impact solution that covers the weakness found in the IHS, we applied the operators of mutation, crossover and selection on all vectors of memory. In fact, this treatment is designed to be made each iteration to change memory vectors from one generation to another and add the dynamic aspect to our algorithm. The vectors resulting from this treatment form a new range of values much closer to the global optimum for the future improvisation (see Figure 2).

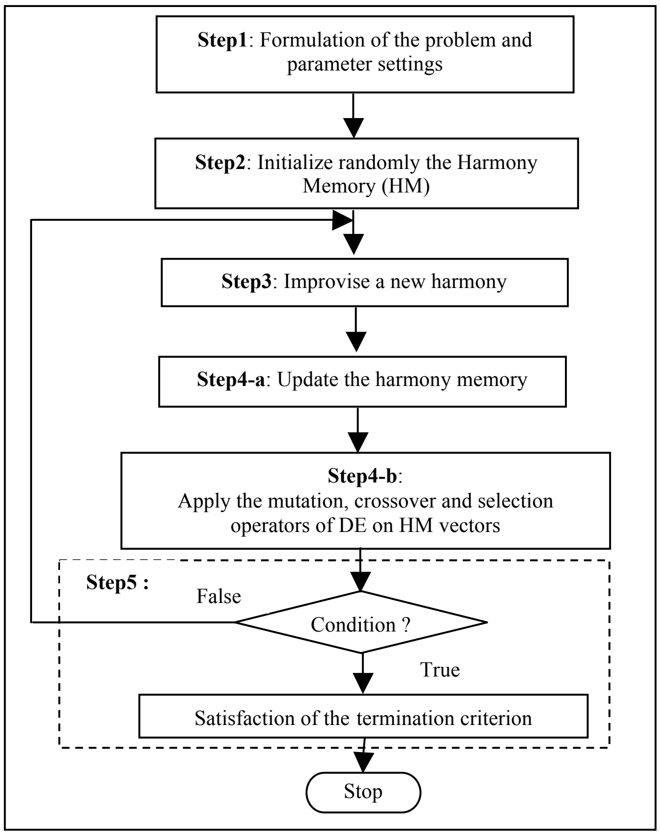

Figure 2. Flowchart of IHSDE

The IHS algorithm which is hybridized with DE (IHSDE) showed a better behavior than the one hybridized with PSO (IHSPSO) (see table II). As a consequence, the IHSDE was adapted as the best hybridized algorithm in this work.

V. EXPERIMENTS

The performance of IHSDE is evaluated and compared to the IHS, DE, PSO and IHSPSO using 25 tests Benchmark functions (table I) at 10 runs. The benchmark functions are the single objective optimization functions published in CEC 2005 [15].

A. Experimental setup

The parameters of IHS are set as: PARmin=0.0001; PARmax=1.0; bwmin=0.0001; bwmax=1.0; HMCR=0.9; HMS=10.

The parameters of DE are: CR=0.7; F=0.8; strategy = 1.

The parameters of PSO are: C1 = 0.5; C2 = 1.5.

TABLE I. list of the Benchmark functions

Benchmark function *F*	**Number**	***F* (x*)**	**Dimension**	**Range**
Sphère	1	0	2	[-5.12 5.12]
Rosenbrock	2	0	2	[-5 10]
Rastrigin	3	0	2	[-5.12 5.12]
Griewank	4	0	2	[-600 600]
Ackley	5	0	2	[-15 30]
Beale	6	0	2	[-4.5 4.5]
Booth	7	0	2	[-10 10]
Bohachevsky	8	0	2	[-100 100]
	9	0	2	[-100 100]
	10	0	2	[-100 100]
Dixon & Price	11	0	2	[-10 10]
Matyas	12	0	2	[-10 10]
Sum Squares	13	0	2	[-10 10]
Power Sum	14	0	4	[0 nbv]
Zakharov	15	0	2	[-5 10]
Perm	16	0	2	[-nbv nbv]
Powell	17	0	4	[-4 5]
Hump	18	0	2	[-5 5]
Levy	19	0	2	[-10 10]
Branin	20	0.397887	2	[-5 10;0 15]
Easom	21	-1	2	[-100 100]
Goldstein & Price	22	3	2	[-2 2]
Hartmann3	23	- 3.86278	3	[0,1]
Michalewics	24	-1.8013	2	[0 pi]
Shubert	25	-186.7309	2	[-10 10]

There are also two stopping criteria are applied:

- The maximum number of function iterations is reached. Here, it is set to 50000 times.
- The difference of objective function values between the best solution found so far and the global optimal solution (i.e., error function value is smaller than 10^{-10}).

The optimization performance is quantitatively measured by the mean value and standard deviation of the best fitness achieved when an algorithm terminates over 10 runs and the spent time or number of function evaluations. An optimization algorithm is regarded as successfully solving the problem once it achieves the closest fitness to the global optimum faster.

B. *Results*

To compare the optimization performances of HS, IHS, DE, PSO, IHSPSO and IHSDE in terms of the mean value and standard deviation of the best mean as well as the number of evaluations over 10 runs (TABLE I), 25 test Benchmark functions were tested.

TABLE II. Performances of IHS, DE, PSO, IHSPSO and IHSDE in terms of the mean value, evaluation number and standard deviation over 10 runs with benchmark functions.

F	**IHS** Mean NFEs STD	**DE** Mean NFEs STD	**PSO** Mean NFEs STD	**IHSPSO** Mean NFEs STD	**IHSDE** Mean NFEs STD
F1	4.5187e-011 27249 2.2447e-011	3.5647e-011 542 2.5598e-011	3.5471e-017 703 9.5916e-017	4.2766e-011 120 2.2177e-011	**5.4639e-011** **67** **3.2006e-011**
F2	7.8262e-011 45282 5226e-011	2.7441e-005 1592 8.6768e-005	1.5155e-012 838 4.1958e-012	6.4911e-011 520 2.6834e-011	**4.2957e-011** **100** **4.1573e-011**
F3	3.1741e-011 41472 2.4682e-011	2.3335e-011 1239 1.9295e-011	1.2896e-013 1048 3.4468e-013	**2.2750e-011** **420** **1.9708e-011**	2.9425e-011 883 1.3364e-011
F4	0.0064 46790 0.0035	0.0022 2238 0.0036	**7.9936e-016** **924** **2.1335e-015**	0.0099 50001 0.00081	0.0059 48134 0.0033
F5	1.0409e-006 50001 6.4476e-007	5.6568e-011 1145 2.7565e-011	6.1688e-012 1227 1.0185e-011	5.1547e-011 290 3.1902e-011	**1.2516e-010** **109** **1.4546e-010**
F6	5.0167e-011 41291 2.6516e-011	3.9452e-011 593 3.6209e-011	1.2750e-015 766 2.3848e-015	4.6620e-011 680 3.4312e-011	**4.4427e-011** **70** **3.0559e-011**
F7	5.0064e-011 39312 3.1562e-011	3.2402e-011 631 3.2736e-011	5.8775e-016 735 1.0494e-015	4.1209e-011 150 1.1006e-011	**6.2687e-011** **60** **1.9567e-011**
F8	4.5735e-011 37803 2.5542e-011	2.3543e-005 836 7.2478e-005	5.5511e-017 1073 1.1992e-016	3.6915e-011 191 1.5552e-011	**6.0276e-011** **62** **2.3376e-011**
F9	4.0769e-011 35207 2.5687e-011	2.9646e-011 736 2.2654e-011	2.0428e-015 829 4.9877e-015	4.2349e-011 178 1.8318e-011	**3.8629e-011** **71** **2.9354e-011**
F10	7.1071e-011 43423 2.3433e-011	4.2578e-011 748 3.1568e-011	2.7423e-015 743 5.9293e-015	5.3170e-011 210 3.1223e-011	**4.6674e-011** **78** **3.1721e-011**
F11	6.5146e-011 40304 2.6147e-011	4.3527e-011 524 2.1346e-011	4.5752e-017 1706 7.6523e-017	4.0455e-011 170 3.1845e-011	**6.6922e-011** **68** **3.1177e-011**
F12	5.7484e-011 32023 2.6410e-011	5.1415e-011 538 2.6630e-011	1.9354e-016 833 3.6419e-016	3.8983e-011 138 3.1708e-011	**5.5197e-011** **63** **2.8096e-011**
F13	4.4875e-011 25797 3.1514e-011	5.0471e-011 559 2.3580e-011	1.0713e-015 701 3.0459e-015	8.3819e-012 142 8.1335e-012	**4.4596e-011** **56** **2.3300e-011**
F14	6.2246e-004 50001 6.0806e-004	0.0231 2015 0.0356	2.2477e-004 6206 1.8162e-004	1.9530e-004 50001 1.7184e-004	**9.8412e-011** **11383** **2.5298e-012**
F15	6.2007e-011 30447 2.8584e-011	4.5618e-011 555 2.4838e-011	1.5952e-016 716 3.2267e-016	5.5227e-011 124 2.6648e-011	**3.8127e-011** **56** **3.0953e-011**
F16	3.0623e-011 42261 2.1013e-011	3.0072e-011 353 1.8266e-011	6.4035e-013 789 1.9929e-012	3.7746e-011 170 2.2669e-011	**5.3011e-011** **73** **3.2755e-011**
F17	1.0506e-006 50001 6.8446e-007	1.0435e-005 1490 3.2996e-005	5.2693e-008 1442 5.7791e-008	9.9395e-011 16350 1.1230e-012	**5.9498e-011** **150** **4.0279e-011**
F18	4.6511e-008 50001 1.5773e-012	4.6510e-008 2014 7.0217e-017	4.6510e-008 770 2.5746e-016	4.6510e-008 50001 1.2162e-016	**4.6510e-008** **50001** **1.5392e-014**
F19	4.8289e-011 2319 3.4688e-011	4.9483e-011 519 3.3277e-011	3.7053e-016 642 8.9506e-016	4.3066e-011 128 4.4756e-011	**5.5477e-011** **62** **2.7631e-011**
F20	0.3979 1508 2.6631e-005	0.3979 272 3.1654e-005	0.3979 1038 0	0.3979 56 4.4087e-005	**0.3979** **22** **3.5446e-005**
F21	-1.0000 26920 3.2156e-011	-1 2013 0	-1 934 3.5108e-016	-1.0000 7228 3.1157e-011	**-1.0000** **1097** **2.9920e-011**
F22	3.0000 45880 3.8758e-011	3.0000 637 3.0254e-011	3.0000 860 2.6089e-014	3.0000 3378 7.1556e-012	**3.0000** **97** **1.5357e-011**
F23	-3.8628 12572 5.8627e-007	-3.8628 2016 5.9212e-016	-3.8628 854 1.9918e-014	-3.8628 84 4.7866e-007	**-3.8628** **63** **6.1683e-007**
F24	-1.8013 6157 1.0107e-006	-1.8210 2013 3.9165e-016	-1.9988 1714 0.0018	-1.8136 304 0.0057	**-1.8013** **52** **8.7174e-007**
F25	-186.7309 13663 2.2569e-006	-186.7309 2014 4.2369e-014	-186.7309 1013 3.2819e-014	**-186.7309** **138** **2.0915e-006**	-186.7309 1018 1.6613e-006

For each function, bold fonts in TABLE II, show which algorithm works more efficient and gives the best results. In this comparative study, our goal is to minimize the time to reach the global optimum respecting the given margin of error. So, we considered the algorithm that is closest to the global optimum for a minimum number of evaluations as the best.

The techniques of mutation, crossover and selection of DE which adapted to IHS, accelerates the convergence of the algorithm and provides a guided sequence of steps. Therefore, the new algorithm IHSDE made the minimum number of evaluations (reduced time) to converge toward the global optimum in the most of cases (see TABLE II). The IHSDE usually demonstrates superior performances compared to IHS, DE, PSO and IHSPSO on test functions except for few cases.

In the second level, the IHSPSO was given competitive results at those of IHSDE and even sometimes better.

In the literature, there are other algorithms that have hybridization between HS and DE like DHS [3] and HSDM [12]. To evaluate IHSDE, it was compared with these algorithms with respect to each of 5 tests Benchmark functions (TABLE I). All of these algorithms run under the same conditions and parameters values: HMS = 50, HMCR = 0.98, PAR = 0.3, BW=0.01

TABLE III. Performances of HS, DHS, HSDM and IHSDE in terms of the mean value and standard deviation over 25 runs with 5 benchmark functions at 30 Dimension.

Benchmark function	HS (Mean / STD)	DHS (Mean / STD)	HSDM (Mean / STD)	IHSDE (Mean / STD)
Sphère	2.920e-05 5.389e-06	5.650e-02 2.645e-02	8.055e-06 3.508e-05	**8.0211e-011** **1.5751e-011**
Rosenbrock	2.458e+01 1.759e+01	4.460e+01 2.790e+01	2.629e+01 8.400e-01	**8.9012e-011** **4.1932e-012**
Ackley	4.001e-03 2.954e-04	6.169e-02 1.326e-02	4.395e-05 1.552e-04	**9.3415e-011** **4.3594e-012**
Griewank	1.406e-02 1.366e-02	1.205e-01 3.284e-02	7.186e-04 2.078e-03	**2.2098e-04** **0.0038**
Rastrigin	5.391e-03 9.596e-04	3.059e-02 1.298e-02	7.079e-05 2.315e-04	**8.7346e-011** **9.6772e-012**

In this table, the value in bold fonts, which corresponds to the IHSDE shows that this algorithm reaches the best result in this comparison. The IHSDE proved its superiority for all existing algorithms and showed great performances with the 5 test benchmark functions. These results emphasize the strategy allowed for hybridization in this study.

VI. CONCLUSIONS

In this paper, different metaheuristic algorithms have been studied such as Harmony Search HS, Particle Swarm Optimization PSO and Differential Evolution DE. A new hybridization search procedure inspired by evolution concept and swarm behavior was developed. This hybridization has combined The Improved HS with PSO to result IHSPSO and with DE to result IHSDE. These algorithms are tested by the benchmark functions (CEC 2005) and compared with each other and some other algorithm from the literature. The experimental results demonstrate that the hybridized Harmony Search algorithm IHSDE shows more efficiency and performing to reach the global optimum more rapidly.

ACKNOWLEDGMENTS

The authors would like to acknowledge the financial support of this work by grants from General Direction of Scientific Research (DGRST), Tunisia, under the ARUB program. This work was also supported in the framework of the IT4 Innovations Centre of Excellence project, reg. no. CZ.1.05/1.1.00/02.0070 by operational programme 'Research and Development for Innovations' funded by the Structural Funds of the European Union and state budget of the Czech Republic, EU.

REFERENCES

[1] S. Bouaziz, H. Dhahri, A.M. Alimi, "Evolving Flexible Beta Operator Neural Trees (FBONT) for Time Series Forecasting", T. Hung et al. (Eds.) : 19th International Conference in neural information Processing (ICONIP'12), Proceedings, Part III, Series: Lecture Notes in Computer Science, Doha-Qatar, vol. 7665, pp. 17-24, 2012.

[2] S. Bouaziz, H. Dhahri, A.M. Alimi, A. Abraham, "A hybrid learning algorithm for evolving Flexible Beta Basis Function Neural Tree Model", Neurocomputing, In Press-Corrected Proof, 2013.

[3] P. Chakraborty, G.G. Roy, S. Das, D. Jain, and A. Abraham, "An improved harmony search algorithm with differential mutation operator", Fundamenta Informaticae, vol. 95, pp. 1-26, 2009.

[4] S. Das and P.N. Suganthan, "Differential Evolution: A Survey of the State-of-the-Art", IEEE transactions on evolutionary computation, vol. 15, NO. 1, February 2011.

[5] Z. W. Geem, "Music-Inspired Harmony Search Algorithm:Theory and Applications", 1st edition. Springer, 2009.

[6] Z.W. Geem, J.H. Kim and G.V. Loganathan, "A new heuristic optimization algorithm: Harmony search. Simulation", Simulation, 76:60-68, 2001.

[7] J.H. Holland, "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Harbor, 1992.

[8] X. Hu, R. Eberhart and Y. Shi, "Recent advances in particle swarm", IEEE Congress on Evolutionary Computation 2004, Portland, Oregon, USA, 2004

[9] J. Kennedy and R. Eberhart, "Particle Swarm Optimization", In Proceedings of the Fourth IEEE International Conference on Neural Networks, Perth, Australia. IEEE Service Center, pp. 1942- 1948, 1995.

[10] M. Mahdavi, M. Fesanghary and E. Damangir, "An improved harmony search algorithm for solving optimization problems", Applied Mathematics and Computation, vol. 188, No. 2, pp. 1567-1579, 2007.

[11] K. M. Passino, "Biomimicry of bacterial foraging for distributed optimization and control", IEEE Control Syst. Mag., vol. 22, no. 3, pp. 52–67, Jun. 2002.

[12] A. K. Qin and F. Forbes, "Harmony Search with Differential Mutation Based Pitch Adjustment", Proc. of the 2011 Genetic and Evolutionary Computation Conference (GECCO'11), Dublin, Ireland, July, 2011.

[13] R. Storn, "On the usage of differential evolution for function optimization ", Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS), IEEE, Berkeley, 1996, pp. 519-523, 1996.

[14] R. Storn and K. Price, "Differential Evolution-A Simple and efficient adaptive scheme for global optimization over continuous spaces", Berkeley, CA, Tech. Rep. TR-95-012, 1995.

[15] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y. -P. Chen, A. Auger and S. Tiwari, "Problem Definitions and Evaluation Criteria for the CEC 2005 Special Session on Real-Parameter Optimization ", May 2005.

[16] K. Vaisakh, L. R. Srinivas, "Genetic evolving ant direction HDE for OPF with non-smooth cost functions and statistical analysis", Expert Systems with Applications, vol. 38, no. 3, pp. 2046–2062, Mar. 2011.

[17] X.S. Yang, "Harmony Search as a Metaheuristic Algorithm", in: Music-Inspired Harmony Search Algorithm: Theory and Applications", (Editor Z. W. Geem), Studies in Computational Intelligence, Springer Berlin, vol. 191, pp. 1-14, 2009.

[18] J. Zhang, H. Chung, W. L. Lo, and T. Huang, "Extended Ant Colony Optimization Algorithm for Power Electronic Circuit Design", IEEE Transactions on Power Electronic. Vol.24, No.1, pp.147-162, 2009.

[19] H. Izakian, A. Abraham, V. Snasel, Comparison of Heuristics for Scheduling Independent Tasks on Heterogeneous Distributed

Environments, The 2009 IEEE International Workshop on HPC and Grid Applications (IWHGA2009), China, IEEE Press, USA, ISBN 978-0-7695-3605-7, pp. 8-12, 2009.

[20] S. Das, A. Mukhopadhyay, A. Roy, A. Abraham and Bijaya Panigrahi, Exploratory Power of the Harmony Search Algorithm: Analysis and Improvements for Global Numerical Optimization, IEEE Transactions on Systems Man and Cybernetics - Part B, IEEE Press, USA, Volume 41, Issue 1, pp. 89-106, 2011.

[21] F. Xhafa, E. Alba, B. Dorronsoro, B. Duran and A. Abraham, Efficient Batch Job Scheduling in Grids Using Cellular Memetic Algorithms, Studies in Computational Intelligence, Springer Verlag, Germany, ISBN: 978-3-540-69260-7, pp. 273-299, 2008.

[22] S. Das, A. Biswas, S. Dasgupta and A. Abraham, Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications, Foundations of Computational Intelligence Volume 3: Global Optimization, Studies in Computational Intelligence, Springer Verlag, Germany, ISBN: 978-3-642-01084-2, pp. 23-55, 2009.

[23] Hongbo Liu, Ajith Abraham and Maurice Clerc, Chaotic Dynamic Characteristics in Swarm Intelligence, Applied Soft Computing Journal, Elsevier Science, Volume 7, Issue 3, pp. 1019-1026, 2007.

[24] A. Abraham, Intelligent Systems: Architectures and Perspectives, Recent Advances in Intelligent Paradigms and Applications, Abraham A., Jain L. and Kacprzyk J. (Eds.), Studies in Fuzziness and Soft Computing, Springer Verlag Germany, ISBN 3790815381, Chapter 1, pp. 1-35, 2002.

[25] F. Xhafa and A. Abraham, Meta-heuristics for Grid Scheduling Problems, Metaheuristics for Scheduling: Distributed Computing Environments, Studies in Computational Intelligence, Springer Verlag, Germany, ISBN: 978-3-540-69260-7, pp. 1-37, 2008.

[26] R. Thangaraj, M. Pant, A. Abraham and P. Bouvry, Particle Swarm Optimization: Hybridization Perspectives and Experimental Illustrations, Applied Maths and Computation, Elsevier Science, Netherlands, Volume 217, No. 1, pp. 5208-5226, 2011.

[27] H. Liu, A. Abraham, O.K. Choi and S.H.Moon, Variable Neighborhood Particle Swarm Optimization for Multi-objective Flexible Job-shop Scheduling Problems, The Sixth International Conference on Simulated Evolution And Learning (SEAL06), China, Springer Verlag, Germany, Lecture Notes in Computer Science, T.D.Wang et al. (Eds.): SEAL 2006, LNCS 4247, pp. 197-204, 2006.