Automatically Modeling Hybrid Evolutionary Algorithms from Past Executions

Santiago Muelas, José-María Peña, and Antonio LaTorre

DATSI, Facultad de Informática
Universidad Politécnica de Madrid, Spain
{smuelas,jmpena,atorre}@fi.upm.es

Abstract. The selection of the most appropriate Evolutionary Algorithm for a given optimization problem is a difficult task. Hybrid Evolutionary Algorithms are a promising alternative to deal with this problem. By means of the combination of different heuristic optimization approaches, it is possible to profit from the benefits of the best approach, avoiding the limitations of the others. Nowadays, there is an active research in the design of dynamic or adaptive hybrid algorithms. However, little research has been done in the automatic learning of the best hybridization strategy. This paper proposes a mechanism to learn a strategy based on the analysis of the results from past executions. The proposed algorithm has been evaluated on a well-known benchmark on continuous optimization. The obtained results suggest that the proposed approach is able to learn very promising hybridization strategies.

1 Introduction

The selection of the most appropriate Evolutionary Algorithm (EA) for a given optimization problem is a difficult task, sometimes considered an optimization problem itself [2].

Even though the No Free Lunch Theorem asserts that "*any two algorithms are equivalent when their performance is averaged across all possible problems*", in practice, and being constrained to certain types of problems, the performance of some particular algorithms is better than others. In most of the cases, the selection of the most appropriate algorithm is carried out by the execution of several alternative algorithms (advised by the literature or the own experience) and then choosing the one reporting the best results.

Supported by these arguments, hybrid evolutionary techniques are a promising alternative to deal with these situations. By combining different heuristic optimization approaches, it is possible to profit from the benefits of the best approach, avoiding the limitations of the others. These hybrid algorithms also hold the hypothesis that the combination of several techniques can outperform the sole usage of its composing algorithms. This hypothesis is based on the idea that the comparative performance of the algorithms is not the same along the whole optimization process. Moreover, it is possible to identify different best performing algorithms for different phases of the optimization process.

Nowadays, there is an active research in the design of dynamic or adaptive hybrid algorithms. However, this paper introduces a different perspective, barely explored in the literature. This contribution proposes a mechanism to learn the best hybrid strategy from the analysis of the results from past executions. The idea of using past executions to induce the most appropriate hybridization technique is particularly useful in those scenarios in which an optimization problem is solved multiple times. These multiple executions could include slightly different conditions that actually have an influence in the position of the optimal value, but do not change the main characteristics of the fitness landscape in which this optimization process searches. Many industrial problems have this characteristic in which the fitness function is mainly the same, but the particular conditions or other input parameters are changed on each execution, e.g., the optimization of engineering structures evaluated under different stress conditions.

A rather naïve solution to this approach is the design of an alternating strategy, based on the generation number or the fitness values. Nevertheless, this idea does not consider that reaching a given fitness value or a particular generation number is achieved via a stochastic process. This process does not ensure that the same generations or the same fitness values reached by an algorithm actually represent the same situation of the search process in two different executions. Any successful strategy would need not only this general parameters, but also other statistical or introspective information of the evolving population, in order to identify a situation similar to one previously learned.

This paper presents a new hybrid Evolutionary Algorithm that learns the best sequence of techniques according not only to their performance but also to other statistical/informative parameters of the evolved population. This new algorithm has been evaluated using a well-known benchmark of continuous optimization functions and its results have been validated using non-parametric tests.

The rest of the paper is organized as follows: Section 2 presents an overview of several hybrid algorithms. Section 3 details the proposed algorithm. In Section 4 the experimental scenario is described in detail. Section 5 presents and comments on the results obtained and lists the most relevant facts from this analysis. Finally, Section 6 contains the concluding remarks obtained from this work.

2 Related Work

In this section, some of the most relevant work on High-level relay hybrid (HRH) algorithms will be reviewed. The HRH terminology was introduced in [7], one of the first attempts to define a complete taxonomy of hybrid metaheuristics. This taxonomy is a combination of a hierarchical and a flat classification structured into two levels. The first level defines a hierarchical classification in order to reduce the total number of classes, whereas the second level proposes a flat classification, in which the classes that define an algorithm may be chosen in an arbitrary order. From this taxonomy, four basic hybridization strategies can be derived: (a) LRH (Low-level relay hybrid): One metaheuristic is embedded into a single-solution metaheuristic. (b) HRH (High-level relay hybrid): Two

metaheuristics are executed in sequence. (c) LTH (Low-level teamwork hybrid): One metaheuristic is embedded into a population-based metaheuristic. (d) HTH (High-level teamwork hybrid): Two metaheuristics are executed in parallel. For this work, we have focused on the HRH group, the one the algorithm proposed in this paper belongs to.

There has been an intense research in HRH models in the last years combining different types of metaheuristics. In the following paragraphs some of the most recent and representative approaches will be reviewed.

The DE algorithm is one of the evolutionary algorithms that has been recently hybridized following the HRH strategy. For example, it has been combined with Evolutionary Programming (EP) in [9]. The EP algorithm is executed for each trial vector created by the DE algorithm which is worse than its associated target vector. DE has also been combined with PSO [3]. In this case, the PSO algorithm is executed as the main algorithm but, from time to time, the DE algorithm is launched to move particles from already explored areas to new positions. The particles preserve their velocity when they are moved by the DE in order to minimize the perturbation in the general behavior of the PSO algorithm.

There have also been some studies that have tried to use adaptive learning for combining the algorithms. In [6], the authors propose two adaptive strategies, one heuristic and one stochastic, to adapt the participation of several local searches when combined with a Genetic Agorithm (GA). In both strategies, there is a learning phase in which the performance of each local search is stored and used in later generations in order to select the local search to apply. In [1,10] several local searches are combined with a metaheuristic algorithm using also an adaptive scheme. The application of each algorithm is based on a population diversity measure which varies among the studies. When applied to the DE algorithm, this strategy prevents the stagnation problems of the DE by reducing the excessive difference of the best individual and the rest of the population.

Finally, as far as the authors are concerned, no other study has ever tried to focus on doing a post-execution learning of the best patterns for combining the algorithms of a HRH algorithm. Therefore, this contribution proposes a new approach, based on this idea, to try to exploit the potential synergies between different search strategies.

3 Contribution

In this study, we propose the hypothesis that it is possible, based on the behavior of previous executions, to learn a hybridization strategy for the algorithms of an HRH algorithm in order to select the most appropiate algorithm for each iteration of the execution.

For this task, a new methodology for executing HRH algorithms has been developed. Briefly, the methodology "observes" the execution of a HRH algorithm and stores some information about each state of the execution along with the information (for that state) of the performance of the algorithms involved in the hybrid algorithm. With this information, the methodology is able to construct a

model and use it as a hybridization strategy of a new algorithm which will try in future executions to select the most appropiate algorithm for each state found.

The main steps of this proposal are depicted in Figures 1.a and 1.b. As previously mentioned, the methodology starts the execution of a HRH algorithm which, as all hybrid algorithms, has a hybridization strategy that determines the combination of the algorithms involved in the hybrid. Let P_i be the population of iteration i and the *active algorithm* the one selected by the hybridization strategy for executing at that iteration. First, the *active algorithm* is executed over P_i for M evaluations (*period of evaluations*) generating the population P_{i+M}^{active}. In order to compare its performance, the remaining algorithms are also executed with the same starting population P_i, generating a new population P_{i+M}^{j} for each j algorithm. The algorithm that produces the individual with the highest score from P_{i+M}^{active} and all P_{i+M}^{j} populations is granted a *win*. Since the involved algorithms are stochastic, this process needs to be repeated several times in order to obtain a more reliable measure of the performance. After N repetitions starting with the same population P_i, the methodology generates a data record which stores the information of the state and the number of wins of each of the involved algorithms. The state of an iteration is determined by some extracted measures from both the population and the hybridization strategy. The execution continues with the population of the *active algorithm* of the last repetition and continues this process until the stop criterion is satisfied.

After a number of executions, a broad data set of data records, which store the performance of the algorithms over different states, is obtained. Then, the records are preprocessed, filtering out those which have the same number of wins for all the algorithms and adding a class attribute which contains the algorithm with the highest number of wins. The resultant data set is used as input for a machine learning algorithm (c4.5 in this study), which returns a set of rules that determine the best algorithm to apply at each state. This model is used to construct the proposed smartHRH algorithm which, at each iteration, analyzes the state and selects (according to the model) the most appropriate algorithm.

Since the proposed methodology can be applied to any HRH algorithm, the described process can also be successively applied to the smartHRH algorithm to refine the learned model. To infer the model of each version of smartHRH, the previous data sets of data records are also considered. Each data set participates with the same number of records, therefore, the maximum number which are sampled is determined by the smallest data set.

Although the proposed process can be started with any HRH algorithm, an HRH algorithm which tries to select the best combination of algorithms has been used as the initial algorithm. This algorithm, called baseHRH, uses the information of the number of wins described earlier in order to select the best algorithm for a specific state. It follows the same steps of Figure 1.a but continues the execution with the population of the last attempt of the algorithm that obtained the greatest number of wins. This way, the initial set of records used for learning come from a potentially good sequence of algorithms.

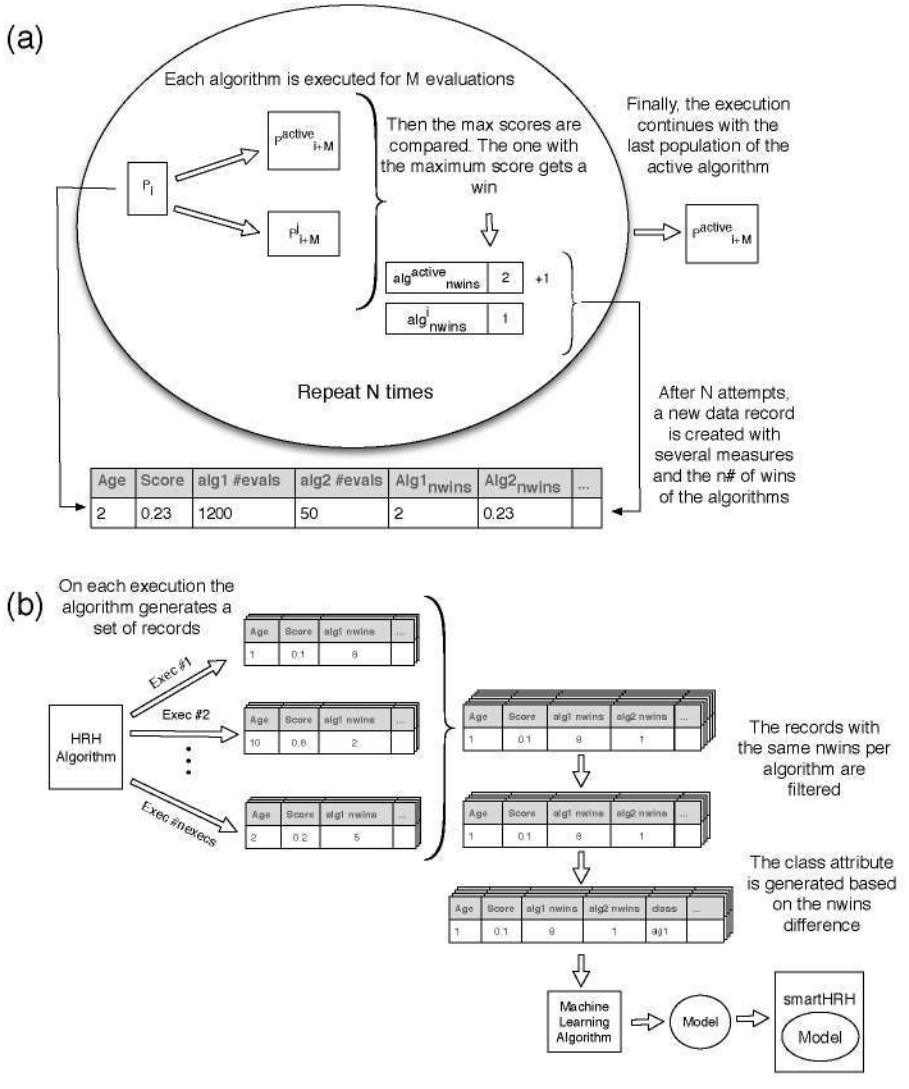

Fig. 1. Generation of the data records and learning procedure

4 Experimentation

For this experimentation, the benchmark from the workshop on *Evolutionary Algorithms and other Metaheuristics for Continuous Optimization Problems - A Scalability Test* held at the ISDA 2009 Conference has been considered. This benchmark defines 11 continuous optimization functions. The first 6 functions were originally proposed for the '*Special Session and Competition on Large Scale Global Optimization*" held at the CEC 2008 Congress [8]. The other 5 functions

Table 1. Functions

Id	Name
f1	Shifted Sphere Function
f2	Shifted Schwefel's Problem 2.21
f3	Shifted Rosenbrock's Function
f4	Shifted Rastrigin's Function
f5	Shifted Griewank's Function
f6	Shifted Ackley's Function
f7	Schwefel's Problem 2.22
f8	Schwefel's Problem 1.2
f9	Extended f_{10}
f10	Bohachevsky
f11	Schaffer

Table 2. DE Parameter Values

Parameter	Values
Population size	25
CR	0.5
F	0.5
Crossover Op.	Exponential
Selection Op.	Tournament 2

have been specially proposed for the Workshop of the ISDA 2009 Conference. These functions, presented in Table 1, have different degrees of difficulty and can scale to any dimension. Detailed information about the selected benchmark can be found at the web page of the organizers of the workshop[1].

The results reported in this section are the aggregation of 25 independent executions on 200 dimensional functions. The performance criterion is the distance (error) between the best individual found and the global optimum in terms of fitness value. Following the benchmark recommendations, the maximum number of Fitness Evaluations has been fixed to $5000 \times D$, where D is the number of dimensions. Due to the constraints of the framework employed, the maximum reachable error without loosing precision is 1E−14.

The algorithms that were used for the new HRH algorithm are two which combination obtained very competitive results on the workshop of the ISDA 2009 Conference [5]. These two are the DE algorithm and the first of the local searches of the MTS algorithm [11]. The MTS algorithm was designed for multi-objetive problems but it has also obtained very good results with large scale optimization problems. In fact, it was the best algorithm of the CEC'08 competition [8]. The DE algorithm is one of the recent algorithms that, due to its results, has quickly gained popularity on continuous optimization. In the last IEEE competitions on continuous optimization, a DE-based algorithm has always reached one of the best three positions. Nevertheless, DE is subject to stagnation problems which could heavily influence the convergence speed an the robustness of the algorithm [4]. Therefore, the idea of combining them is to assist the explorative power of DE by an exploitative local search which has proven to obtain some of the best results. The reason for selecting only the first of the three local searches of the MTS is that, in a previous study by the authors on the same set of functions, this local search was the one that achieved the best results. Besides, we have slightly modified this local search so that, at each iteration, it only explores a subset of randomly

[1] http://sci2s.ugr.es/programacion/workshop/Scalability.html

selected dimensions (75% of the total). This modification has achieved a similar performance with a 25% less of evaluations on a preliminary analysis allowing the hybrid algorithm to spend more evaluations in the DE algorithm. The parameters used for the DE are the same ones of a hybrid algorithm presented at the ISDA 2009 Conference [5] and are presented in Table 2. Any measure could be used for specifying a state but, for the experiments, the following were selected: maximum age, number of evaluations, total and average number of evaluations per algorithm, number of activations of each algorithm and the ration of one against the other, number of evaluations of the best individual without improvement and the best score.

Finally, the period of evaluations used to compare the performance of both algorithms has been set to 1250 evaluations, a value that obtained good results in a set of previous tests.

For analyzing the results, the following algorithms were executed over the benchmark: the DE algorithm, the first local search of the MTS algorithms (LS1), a randomHRH algorithm which, at each iteration, selects an algorithm based on a binomial distribution of $p = 1/2$, the baseHRH algorithm used for obtaining the first model described in Section 3 and the best smartHRH algorithm. Up to eight versions of smartHRH algorithms were obtained per function. From those eight, the one with the best average score was selected. Then, the algorithm was executed again using the same model of this best execution in order to be fair with the remaining algorithms.

5 Analysis of the Results

Table 3 presents the results of the average score of the 25 executions. The best values for each function are highlighted in the table. It can be seen that the smartHRH algorithm obtains the best results in 9 out of 11 functions, reaching the global optimum[2] in 8 functions. It can also be seen that the smartHRH algorithm outperforms the other HRH algorithms (random and base) in most of the functions. In order to obtain a better comparison, each algorithm was compared against each other using the non-parametric Wilcoxon signed-rank test. Each cell $D_{i,j}$ in Table 4 displays the functions for which the results of algorithm i were signicantly better than those of algorithm j with a $p - value < 0.05$. Here, the smartHRH algorithm is also the clear winner obtaining better results than any other algorithm in at least four functions whereas it only looses against randomHRH in f2 and against DE and randomHRH in f3. In these two functions, the DE algorithm is better than the LS1 algorithm at only certain stages of the evolution and only when allowed to execute for more evaluations than the ones used for the comparison of the algorithms (1250 in this case). If selected at these stages, the DE algorithm is able to reach better regions of the search space that could solve premature convergence problems or accelerate the convergence to the global optimum. Since the initial data does not provide any information to predict the long-term benefits of selecting an algorithm (due to the length of

[2] As mentioned earlier with a precission of 1E−14.

Table 3. Average Score Values

Function	DE	LS1	baseHRH	randomHRH	smartHRH
f1	6.78E−01	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f2	7.71E+01	5.99E+01	1.58E+01	**5.34E+00**	1.34E+01
f3	**2.46E+02**	6.98E+03	8.25E+03	1.26E+03	7.63E+03
f4	1.33E+00	**0.00E+00**	4.78E−01	1.60E+01	**0.00E+00**
f5	1.72E−01	3.25E−03	4.63E−03	9.86E−04	**0.00E+00**
f6	9.77E−02	1.00E−12	1.00E−12	1.00E−12	**0.00E+00**
f7	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
f8	1.97E+05	4.78E+00	4.54E+00	8.30E+00	**4.52E+00**
f9	**0.00E+00**	5.23E+02	3.97E−06	8.16E+01	**0.00E+00**
f10	**0.00E+00**	**0.00E+00**	**0.00E+00**	1.06E−08	**0.00E+00**
f11	**0.00E+00**	4.91E+02	8.41E−06	7.89E+01	**0.00E+00**

Table 4. Comparison between algorithms

algorithm	baseHRH	smartHRH	DE	LS1	randomHRH
baseHRH			1,2,4,8	2,9,11	4,8,9,10,11
smartHR	5,6,9,11		1,2,4,8	2,5,6,9,11	4,5,6,8,9,10,11
DE	3,5,6,9,11	3		3,5,6,9,11	3,4,5,6,9,10,11
LS1			1,2,4,8		4,8
randomHRH	2,3,5,6,8	2,3	1,2	2,3,5,9,11	

the period of evaluations), no record of the DE algorithm is generated and no transition to this algorithm is made in the smartHRH algorithm.

In the remaining functions, the smartHRH algorithm is able to detect the best strategy for obtaining the best results. For some functions, the smartHRH decides to execute only one of the algorithms (the best one for that function) whereas for others it combines them in order to obtain more stable results or to reach better solutions. For example, in f5, all the algorithms reach the global optimum in some of their executions, whereas the smartHRH algorithm is the only one to achieve this goal in all of its executions. In f6, the combination of both algorithms allow the smartHRH algorithm to reach the global optimum in all of its executions whereas none of the other algorithms is able to reach it in any of its executions. An example of the evolution of the score of a single execution of the algorithms over the f6 function is displayed in Figure 2. The DE algorithm is not displayed because it quickly converges to poor regions of the search space. It can be seen that the smartHRH algorithm has discovered a beneficial pattern (executing the DE algorithm after several evaluations of the LS1 algorithm) which allows it to reach the global optimum.

An example of the rules generated by the algorithm for functions f5 and f6 is presented in Table 5. As mentioned before, for f6, the algorithm extracts a simple pattern based on the number of evaluations which allows it to reach the global optimum in all the executions. In f5, the evolution of the algorithms is not

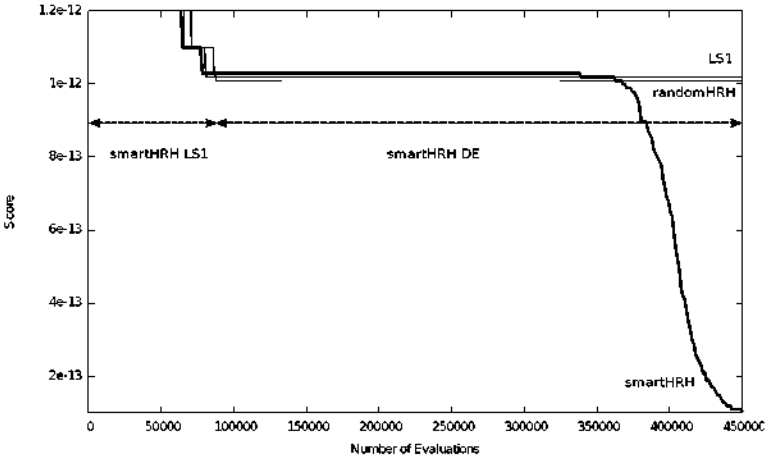

Fig. 2. Comparison of the evolution of the score

Table 5. Rules obtained for functions f5 and f6

f5				
conditions		algorithm	support	precision
if $\#activations_{de}$	<= 9 and			
age_{max}	<= 1899 and			
$avgnevals_{ls1}$	<= 68126	ls1	600	0.99
else if $avgnevals_{ls1}$	<= 37662	de	169	0.93
else if $\#activations_{ls1}$	<= 2 and			
$score_{max}$	<= 0.99	de	64	0.64
else		ls1	65	0.83
f6				
conditions		algorithm	support	precission
if $\#evaluations$	<= 88925	ls1	809	1.0
else		de	358	0.99

always the same and has different patterns. For this reason, the induced model has more rules of higher complexity.

6 Conclusions

In this work, a new hybrid algorithm that learns the best sequence of algorithms has been presented. This learning process uses the information of several parameters of the population, the hybridization and the performance of the algorithms in order to determine the future hybridization strategy. For the experimentation, the benchmark from the workshop on continuous optimization of the ISDA 2009 Conference has been considered. The results have been analyzed and compared with statistical tests. The analysis has proven that the new algorithm is able

to obtain the best overall results, reaching the global optimum in 9 out of 11 functions. This is a first study for validating that the proposed approach can learn very promising hybridization strategies for an HRH algorithm over a set of well-known functions. It must be taken into account that the objective of this study is not to compete against the best algorithms on continuous optimization, since it would not be fair due to the extra number of evaluations used in the learning phase and the per function tuning of the proposed algorithm. As future work we plan to apply this approach to scenarios in which a slightly different optimization problem needs to be solved multiple times. Therefore, the smartHRH algorithm could be trained with several instances of the problem so the resultant algorithm could obtain better results on unseen instances.

Acknowledgments. The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the Centro de Supercomputación y Visualización de Madrid (CeSViMa) and the Spanish Supercomputing Network. This work was supported by the Madrid Regional Education Ministry and the European Social Fund and financed by the Spanish Ministry of Science TIN2007- 67148.

References

1. Caponio, A., Neri, F., Tirronen, V.: Super-fit control adaptation in memetic differential evolution frameworks. Soft Computing - A Fusion of Foundations, Methodologies and Applications 13(8), 811–831 (2009)
2. Grefenstette, J.J.: Optimization of control parameters for genetic algorithms. IEEE Transactions on Systems, Man, and Cybernetics 16, 122–128 (1986)
3. Hendtlass, T.: A combined swarm differential evolution algorithm for optimization problems. In: Monostori, L., Váncza, J., Ali, M. (eds.) IEA/AIE 2001. LNCS (LNAI), vol. 2070, pp. 11–18. Springer, Heidelberg (2001)
4. Lampinen, J., Zelinka, I.: On stagnation of the differential evolution algorithm. In: Proc. of Mendel 2000, pp. 76–83 (2000)
5. Muelas, S., LaTorre, A., Peña, J.M.: A memetic differential evolution algorithm for continuous optimization. In: Proc. of ISDA 2009 (November 2009)
6. Ong, Y.-S., Keane, A.: Meta-lamarckian learning in memetic algorithms. IEEE Transactions on Evolutionary Computation 8(2), 99–110 (2004)
7. Talbi, E.-G.: A taxonomy of hybrid metaheuristics. Journal of Heuristics 8(5), 541–564 (2002)
8. Tang, K., Yao, X., Suganthan, P., MacNish, C., Chen, Y., Chen, C., Yang, Z.: Benchmark functions for the cec 2008 special session and competition on large scale global optimization. Technical report, USTC (2007)
9. Thangaraj, R., Pant, M., Abraham, A., Badr, Y.: Hybrid evolutionary algorithm for solving global optimization problems. In: Corchado, E., Wu, X., Oja, E., Herrero, Á., Baruque, B. (eds.) HAIS 2009. LNCS, vol. 5572, pp. 310–318. Springer, Heidelberg (2009)
10. Tirronen, V., Neri, F., Kärkkäinen, T., Majava, K., Rossi, T.: An enhanced memetic differential evolution in filter design for defect detection in paper production. Evolutionary Computation 16(4), 529–555 (2008)
11. Tseng, L., Chen, C.: Multiple trajectory search for large scale global optimization. In: Proc. of IEEE CEC 2008, June 2008, pp. 3052–3059 (2008)