2014 IEEE Congress on Evolutionary Computation (CEC)
July 6-11, 2014, Beijing, China

Real-Parameter Optimization with OptBees

Renato Dourado Maia, Leandro Nunes de Castro, and Walmir Matos Caminhas

***Abstract*—This paper reports how OptBees, an algorithm inspired by the collective decision-making of bee colonies, performed in the test bed developed for the Special Session & Competition on Real-Parameter Single Objective Optimization at CEC-2014. The test bed includes 30 scalable functions, many of which are both non-separable and highly multi-modal. Results include OptBees' performance on the 10, 30, 50 and 100-dimensional versions of each function.**

I. INTRODUCTION

SWARM Intelligence has attracted the interest of many researchers over the past years. It can be defined as any attempt to design algorithms or distributed problem-solving techniques inspired by the collective behavior of social insect colonies and other animal societies [1]. This definition is focused on social insects, such as termites, bees, wasps, as well as different ant species. The classical example of a swarm is bees swarming around their hive, but the metaphor can be easily extended to other systems with a similar architecture. For instance, an ant colony can be thought of as a swarm whose individual agents are ants; a flock of birds is a swarm of birds; an immune system [2] is a swarm of cells; and a crowd is a swarm of people [3]. The individual agents of a swarm behave without supervision and each of these agents has a stochastic behavior that takes into account its perception of the neighborhood. Local rules, without any relation to the global pattern, and interactions between agents lead to the emergence of a collective intelligence, called swarm intelligence.

Self-organization in swarms presents four main characteristics [1]:

1. *Positive feedback:* simple behavioral rules promote the creation of convenient structures. Recruitment and reinforcement, such as trail laying and following in some ant species or dances in bees, are examples of positive feedback;
2. *Negative feedback:* this type of feedback counterbalances positive feedback and helps to stabilize the collective pattern and emergent behaviors. In order to avoid the saturation which might occur in terms of available foragers, a negative feedback mechanism, such as food source exhaustion, crowding or competition at the food sources, is needed;
3. *Fluctuations:* random walks, errors or random task switching among swarm individuals are vital for creativity and innovation. Randomness is often crucial for emergent structures, since it enables the discovery of new solutions;
4. *Multiple interactions:* an important feature of swarms is that agents use information coming from other agents and the environment for decision making.

Renato Dourado Maia is with the Computer Sciences Department, State University of Montes Claros, MG, Brazil (phone: +55 38 3229 8273; fax: +55 38 3229 8270; e-mail: renato.dourado@unimontes.br).

Leandro Nunes de Castro is with Natural Computing Laboratory (LCoN), Graduate Program in Electrical Engineering, Mackenzie University, São Paulo, Brazil (e-mail: lnunes@mackenzie.br).

Walmir Matos Caminhas is with the Computational Intelligence Laboratory, Electronic Engineering Department, Federal University of Minas Gerais, MG, Brazil (e-mail: caminhas@cpdee.ufmg.br).

According to [4], five principles have to be satisfied by a swarm so that intelligent behaviors emerge:

1. *Proximity*: The swarm should be able to do simple space and time computations;
2. *Quality*: The swarm should be able to respond to quality factors in the environment, such as the quality of food sources or the safety of their location;
3. *Diverse response*: The swarm should not allocate all its resources along excessively narrow channels and should distribute resources into many nodes;
4. *Stability*: The swarm should not change its mode of behavior upon every fluctuation of the environment;
5. *Adaptability*: The swarm must be able to change behavior mode when the investment in energy is worth.

Although the self-organization and division of labor features defined by [1] and the satisfaction principles stated in [4] for swarm intelligence are strongly and clearly seen in bee colonies, problem solving techniques based on bee swarm intelligence have begun to be introduced only very recently, from the early to the mid 2000 and have shown promising results in various domains.

The main purposes of this paper are: 1) to present OptBees [5], a bee-inspired algorithm for optimization in continuous spaces; 2) and to evaluate its performance by applying it to all thirty minimization problems proposed for the IEEE 2014 Congress on Evolutionary Computation Competition on Real-Parameter Single Objective Optimization (CEC 2014), considering spaces of 10, 30, 50 and 100 dimensions. A distinguishing feature of OptBees, when compared with other bioinspired techniques such as most evolutionary algorithms, is the use of different types of agents with different roles that may change for each agent according to the features of the problem and the dynamics of the algorithm.

The remainder of this paper is organized as follows. Section II presents OptBees, a bee-inspired algorithm for continuous optimization, and Section III reports and discusses experimental results. Section IV outlines concluding remarks and avenues for future research.

II. OptBees – A Bee-inspired Algorithm For Solving Continuous Optimization Problems

Ants and bees have provided some of the best described mechanisms of collective decision-making. In many insect societies, the essence of these mechanisms is the same, even if some details remain particular to each society. This section formalizes an optimization algorithm inspired by the collective decision-making process of insect societies, specifically bee colonies, targeting continuous optimization. It also provides a brief review of related works from the literature.

Some of the most important features of collective decision-making by bee colonies for the design of algorithms for solving optimization problems are [6]:

1. Bees dance to recruit nestmates to a food source;
2. Bees adjust the exploration and recovery of food according to the colony state;
3. Bees, unlike ants, exploit multiple food sources simultaneously, but almost invariably converge to the same new construction site of the nest;
4. There is a positive linear relationship between the number of bees dancing and the number of bees recruited to a food source: the linear system of recruitment means that workers are evenly distributed among similar options;
5. The dance communicates the distance and direction of new sites for nests. Recruitment for the new site continues until a threshold number of bees is reached;
6. The quality of the food source influences the bee dance;
7. All bees retire after some time, which means that regardless of the quality of the new site, bees stop recruiting other bees. This retirement depends on the quality of the site: the larger, the later the retirement.

By seeking inspiration in these collective decision-making features in bee colonies, the OptBees algorithm, whose main steps are presented below, was proposed [5]. A distinguishing feature of OptBees when compared with other bioinspired techniques, such as most evolutionary algorithms, is the use of different types of agents with different roles. In OptBees, there are three main types of agent bees: 1) recruiters, responsible for recruiting bees for exploiting a certain (promising) region of the space; 2) scouts, that randomly search for new promising regions of the space; and 3) recruited, that are recruited by recruiters to exploit its corresponding (promising) region of the space. These three types of bees represent the active ones, i.e., the bees involved in the foraging activity (task). The other bees represent the inactive bees that stay at the hive with no specific task to perform.

The active bees fly around the space, searching for high quality food sources (promising regions in the search space). According to the qualities of the food sources being explored by active bees, each is classified as recruiter or non-recruiter: this means that multiple food sources (promising regions) can be exploited simultaneously. The recruiter bees attract some of the non-recruiters to exploit their corresponding food source (as in the natural phenomena, the number of bees that each recruiter recruits is proportional to the quality of the food source being explored) and the other non-recruiter bees, the scouts, randomly search for new promising regions (the recruitment process simulates the dance). If the active bees discover a large number of high quality food sources, some of the inactive bees become active and engage in the foraging activity: this process mimics the bees' capability of adjusting the exploration and recovery of food according to the colony state. A high-level pseudocode of the proposed OptBees algorithm is presented below, and a detailed discussion of each step follows in the sequence.

OptBees Algorithm

Input Parameters:

- n_{min}: initial number of active bees.
- n_{max}: maximum number of active bees.
- ρ: inhibition radius.
- n_{mean}: average foraging effort.
- p_{min}: minimum probability of a bee being a recruiter.
- p_{rec}: percentage of non-recruiter bees that will be actually recruited.

Output Parameters:

- Active bees and the respective values of the objective function.

1. Randomly generate a swarm of N bees.

while (stopping criterion is not attained) **do**

2. Evaluate the quality of the sites being explored by the active bees.
3. Apply local search.
4. Determine the recruiter bees.
5. Update the number of active bees.
6. Determine the recruited and scout bees.
7. Perform the recruitment process.
8. Perform the exploration process.

end while

9. Evaluate the quality of the sites being explored by the active bees.
10. Apply local search.

A. *Initialization (Step 1)*

The OptBees algorithm was designed to solve continuous optimization problems. Thus, the natural choice of representation for the bees is to use real-valued vectors. The n_i active bees are initialized by randomly creating real-valued vectors in a space of dimension L, using uniform distribution, where L is the number of coordinates (dimension) of the problem being solved, according to the search space limits.

B. *Evaluation of Bees (Steps 2 and 9)*

The target application of the OptBees algorithm in this paper is optimization in continuous spaces. Thus, some knowledge (information) about the function to be optimized is available (e.g., the function itself, $f(x) = x^3 + x + 3$), and the objective is to determine the values of x that optimize this function. Therefore, in the present paper if the objective is to minimize (or maximize) function $f(x) = x^3 + x + 3$, then the objective can be stated as follows: $\min f(x)$ (or $\max f(x)$). In OptBees, the quality of food sources being exploited by active bees are determined using the values of the objective function $f(x)$ corresponding to the vector of real numbers represented by each one of them. As conceptually quality is a feature to be maximized, for minimization problems it is necessary to perform an appropriate treatment for mapping the values of the objective function to the corresponding values of quality (an example of such treatment is to replace $\min f(x)$ by $\max f_1(x) = -f(x)$).

C. Local Search Operator (Steps 3 and 10)

The local search operator is not inspired by the behavior of bees and was used only to improve the performance of OptBees. Any local search algorithm for continuous spaces can be used in these steps. The algorithm used as the local search operator was LocalSearch1, proposed as part of the MTS algorithm [7] (the winner of the 2008 CEC Special Session and Competition on Large Scale Global Optimization [8]), because it performed well over many problems. This algorithm works in each dimension (variable) of the candidate solutions sequentially and independently, by decreasing or increasing the values of such variables according to a search range (which can be reduced during the execution of the algorithm every time this increase/decrease does not lead to improvements of the candidate solution). A more detailed explanation of this local search algorithm can be found in [7]. This operator is applied at each iteration of OptBees, considering only the bee that represents the current best solution, for thirty iterations, using half of the domain of the variable being changed as the search range: for example, if the variable x_i is such that $a \le x_i \le b$, the correspondent search range for the local search operator is $(b - a)/2$.

D. Determination of the Recruiter Bees (Steps 4)

Determining the recruiter bees involves three steps. In the first step, a probability p_i of being a recruiter bee is associated with each active bee. These probabilities are calculated by Equation 1, in which q_i represents the quality of the food source being explored by bee i and q_{min} and q_{max} represent, respectively, the minimum and maximum qualities among the food sources being explored by each active bee in the current iteration (these quality values are determined using the objective-function values, as explained in Section 2.1.2) and p_{min} defines the minimum probability of a bee to be a recruiter.

$$p_i = \left(\frac{1 - p_{min}}{q_{max} - q_{min}}\right) \cdot \left(q_i - q_{min}\right) + p_{min} \qquad (1)$$

Equation 1 performs a linear scaling between the quality of the food source being explored by a bee and the probability of this bee to be a recruiter. In the second step, the bees are processed and, according to the probabilities calculated in the previous step, are now classified as recruiters or non-recruiters. A random number n_{random} uniformly distributed onver the interval [0, 1] is generated, using a uniform distribution, for each bee i so that, the higher the p_i value, the more likely the bee i is classified as recruiter: if n_{random} is smaller than p_i, than the bee i is classified as recruiter. In the third step, the recruiter bees are processed, in accordance with the corresponding food sources qualities, from best to worst and, for each recruiter bee, the other recruiters, which are at a distance less than or equal to the social inhibition radius ρ, are inhibited, i.e., they become classified as non-recruiters. Let $d(i, j)$ be the Euclidian distance between bees i and j, and consider the set of recruiter always sorted in descending order of qualities. The social inhibition process can be formulated as follows, for each recruiter bee j: for all the other recruiter bees i $(i \neq j)$, if $d(i, j) < \rho$, the bee i is classified as non-recruiter. The motivation for this inhibition process is to avoid the presence of more than one recruiter bee at the same region of the search space.

E. Number of Active Bees Update (Step 5)

After the determination of the recruiter bees, let r be the number of recruiter bees. The average foraging effort n_{mean} determines the desired number of non-recruiter bees for each recruiter bee, i.e., in a given iteration, the number $n_d = (r + 1)\cdot n_{mean}$ determines the desired number of active bees. If this number n_d is greater than the current number of active bees n_{active}, $n_{adjust} = n_d - n_{active}$ is the necessary number of bees that have to become active in order to achieve n_d active bees; if this number is less than the current number of active bees, $n_{adjust} = n_{active} - n_d$ is the necessary number of bees that have to become inactive in order to achieve n_d active bees. This process respects the maximum (n_{max}) and minimum (n_{min}) number of active bees (the minimum number of active bees n_{min} is equal to the initial number of active bees n_i): if $n_d > n_{max}$, then n_d is forced to n_{max}; if $n_d < n_{min}$, then n_d is forced to n_{min}. When an inactive bee becomes active, it is inserted in a random position in the search space. For the inactivation process, the bees are selected according to the corresponding food source quality they explore, from the worse to the best. When a bee is inactivated, it is removed from the swarm and, when a bee is activated, it is inserted into the swarm, i.e., the swarm size varies dynamically. Through this procedure, the foraging effort (computational effort) adapts in accordance with the number of recruiter bees and the maximum number of active bees.

F. Determination of the Recruited and Scout Bees (Step 6)

The number of non-recruiter bees is determined by $n_{nr} = n_{active} - r$ (r is the number of recruiter bees determined in Step 4). Only a percentage of the non-recruiters will be recruited and the others will become scout bees. Thus, the number of recruited bees is $n_r = [p_{rec}\cdot n_{nr}]$ ($[\cdot]$ denotes the nearest integer function) and the number of scout bees is $n_s = n_{nr} - n_r$. The process for determining the recruited bees involves three steps. In the first step, the number of recruited bees to be associated with each recruiter is determined. The relative quality of the food source being explored by each recruiter in relation to the other determines this number: each recruiter recruits a number of bees proportional to the quality of the food source it explores. Let nr_i be the number of recruited bees to be associated with the recruiter bee i, $Q_{recruiters}$ the sum of the qualities of the food sources being explored by all the recruiter bees and q_i the quality of the food source being explored by bee i. The values of nr_i $(i = 1, 2, \cdots, r)$ are calculated using the expression $nr_i = [(q_i/Q_{recruiters})\cdot n_r]$ ($[\cdot]$ denotes the nearest integer function and $Q_{recruiters} = \sum_{k=1}^{r} q_k$)

With the number nr_i of recruited bees to be associated with each recruiter already determined, the non-recruiter bees are processed and associated with the nearest recruiter among those who do not have associated with them a number of bees recruited equal to the corresponding number nr_i determined in the first step. After these procedures, the remaining n_s non-recruiter bees are considered scout bees.

G. *Recruitment Process (Step 7)*

In the recruitment process, the recruiter bees attract the recruited bees to the food sources (search space region) they explore. This recruitment process is implemented by Equation 2 or 3, each with 50% probability, where $\mathbf{x}_i$ is the recruited bee, $\mathbf{y}$ is the recruiter bee, u is a random number with uniform distribution in the interval [0, 1], $\mathbf{U}$ is a vector whose elements are random numbers with uniform distribution in the interval [0, 1] ($\mathbf{U}$ has the same dimension as $\mathbf{x}_i$ and $\mathbf{y}$) and the symbol $\otimes$ denotes the element-wise product.

$$\mathbf{x}_i = \mathbf{x}_i + 2 \cdot \mathbf{U} \otimes (\mathbf{y} - \mathbf{x}_i) \tag{2}$$

$$\mathbf{x}_i = \mathbf{x}_i + 2 \cdot u \cdot (\mathbf{y} - \mathbf{x}_i) \tag{3}$$

Fig. 1 shows the difference between the two recruitment processes, considering a two-dimensional problem: using Equation 3, the recruited bee, after recruitment, will be positioned in any point in the vector that connects the recruiter bee and the point $\mathbf{x}_i + 2\cdot(\mathbf{y} - \mathbf{x}_i)$, while using Equation 2 the recruited bee will be positioned in any point inside the dashed rectangle.

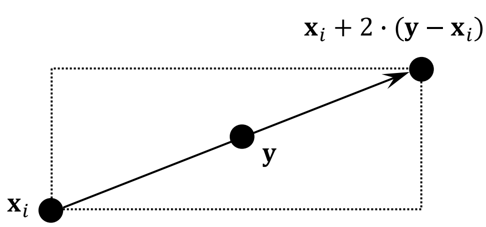

Fig. 1. Differences between the two recruitment processes, considering a two-dimensional problem: $\mathbf{x}_i$ and $\mathbf{y}$ are, respectively, the recruited and the recruiter bee.

H. *Exploration Process (Step 8)*

In the exploration process, each one of the n_s scout bees is moved to a random position in the search space.

III. Experimental Results

To evaluate the performance of OptBees, experiments were performed based on the set of test problems proposed for the Special Session & Competition on Real-Parameter Single Objective Optimization at CEC-2014 (CEC-2014 Competition), which occurred in the IEEE Congress on Evolutionary Computation (CEC) in 2014 [9].

The CEC-2014 Competition problem set consists of thirty minimization mono-objective problems in continuous spaces, which may include several characteristics, such as: uni or multimodality, large number of local optima, dependence or not between variables, non-differentiability in some search space points, and presence of plateaus. Details about these problems can be found in [9].

For the experiments, the Matlab® version of the original implementation for all CEC-2014 Competition problems, available at the following address (CEC-2014 folder), was used:

http://web.mysites.ntu.edu.sg/epnsugan/PublicSite/Shared Documents/Forms/AllItems.aspx
(last accessed on January 09, 2014).

A. *Algorithm Complexity*

The experiments were carried out using a Matlab® implementation of OptBees in a Windows 7 64 bits machine with Intel(R) Core (TM) i7-3630QM CPU @ 2.40 GHz processor and 16 GB of RAM, running Matlab® 8.1 R2013a. The statistics that describe the complexity of the algorithm are presented in Table I. $T0$ is the time of the test program described in [9] and does not depend on the dimension. $T1$ corresponds to the computing time for 200,000 evaluations of the function 18 of [9] and is calculated for the corresponding dimension. Finally, $T2$ is the complete computing time of the algorithm, for a given dimension, with 200,000 evaluations as stopping criterion when tackling benchmark function 18. $T2$ is recorded 5 times and $\hat{T}2$ is the mean of these five values.

TABLE I. Algorithm Complexity.

D	$T0$	$T1$	$\hat{T}2$	$(\hat{T}2 - T1)/T0$
10	0.1293 s	1.4264 s	54.8171 s	412.9211
30	0.1293 s	1.5837 s	54.4716 s	409.0325
50	0.1293 s	1.9408 s	54.0424 s	402.7644
100	0.1293 s	3.4185 s	39.8139 s	281.4803

The $T2$ values presented in Table I seem odd, specially the cost reduction with the increase in dimension (the smaller value was obtained for the greater dimension). This fact is justified by the local search operator used in OptBees, which was implemented using the MTS algorithm [7]. This operator represents, in the worst case, an increase of $40\cdot D$ function evaluations per iteration. As one of the stopping criteria is the number of function evaluations (see Section III.B), the use of the local search operator can considerably reduce the total number of iterations performed by the algorithm, because this operator demands, in the worst case, $2 \cdot D$ functions evaluations reducing the total execution time compared to the version without the local search step. To better demonstrate the relation of the algorithm's complexity with the dimension, Table II presents the same information of Table I, but using the OptBess without the local search step.

TABLE II. Algorithm Complexity.

D	$T0$	$T1$	$\hat{T}2$	$(\hat{T}2 - T1)/T0$
10	0.1293 s	1.4264 s	86.5571 s	658.3968
30	0.1293 s	1.5837 s	148.7507 s	1138.1825
50	0.1293 s	1.9408 s	162.9534 s	1245.2637
100	0.1293 s	3.4185 s	196.2801 s	1491.5824

B. *Results Presentation and Discussion*

The experimental methodology used followed the competition guidelines [9], as presented below:

- The stopping criterion is a maximum number of evaluations of the objective function equals to $D \cdot 10^4$ (D is the dimension of the problem) or an absolute error between the best solution found and the global optimum less than $1 \cdot 10^{-8}$ (in all experiments, only the first criterion was considered).
- Solutions must be initialized randomly, using uniform distributions and considering each problem range.

The values used for the parameters, defined in preliminary tests and which were the same for D = 10, 30, 50 and 100, were: minimum number of active bees n_{min} = 200; maximum number of active bees n_{max} = 1500; average foraging effort n_{mean} = 20; social inhibition radius ρ varying linearly with the number of function evaluations between 0.1 and 0.4 (these numbers correspond to a percentage of the maximum possible distance between two points in the problem's search space); minimum probability of a bee to be a recruiter p_{min} = 0.01; percentage of non-recruiter bees that will be actually recruited p_{rec} varying linearly with the number of function evaluations between 0.5 and 1.

For each pair function/dimension, 51 executions have been performed. Tables III to VI present, for each function, the best, worst, median, mean and standard deviation of the 51 error values between the best solutions obtained in each execution and the global optimum of each problem, for D = 10, 30, 50 and 100, respectively.

Considering the threshold of $1 \cdot 10^{-8}$ [9] and analyzing the mean and median of the error values (Tables III to VI), it is possible to note that the algorithm succeeded only for function 8, for 10, 30, 50 and 100 dimensions.

Comparing the results obtained for functions 8 (Shifted Rastrigin's Function) and 9 (Shifted and Rotated Rastrigin's Function), it is possible to suggest that rotation degenerates the performance of OptBees. This is only a speculation, because there is no more pairs for comparison in the set of functions.

Another fact to be emphasized is that the algorithm did not performed well for functions 1-3, although they are unimodal. The mean and median of the error values for function 1were the worst for all dimension values.

IV. Conclusions and Future Research

This paper presented OptBees, an algorithm for solving continuous optimization problems inspired by the processes of collective decision-making by bee colonies, and evaluated it, in terms of global search, in all thirty minimization problems proposed for the Special Session & Competition on Real-Parameter Single Objective Optimization at CEC-2014 (CEC-2014 Competition), which occurred in the IEEE Congress on Evolutionary Computation (CEC) in 2014, considering 10, 30, 50 and 100 dimensions. It is important to highlight that the same set of parameters was used in all experiments. By doing this, the results obtained can be used as a means to indirectly assess the robustness of the algorithm in relation to its own parameters. Moreover, the values used may serve as a reference for tuning the parameters of the algorithm.

For future research, the authors plan to adapt OptBees for solving clustering and classification problems. Finally, OptBees will be extended for solving constrained, multiobjective and combinatorial optimization problems.

Acknowledgment

The authors thank Capes (Proc. n. 9315/13-6), CNPq, MackPesquisa, Fapesp and Fapemig for the financial support.

References

[1] E. Bonabeau, M. Dorigo, G. Theraulaz, "Swarm Intelligence: From Natural to Artificial Intelligence", NY: Oxford University Press, NewYork, 1999.

[2] L. N. De Castro and J. Timmis, "Artificial Immune Systems: A New Computational Intelligence Approach", Springer-Verlag, 2002.

[3] J. Vesterstrøm, J. Riget, "Particle swarms extensions for improved local, multimodal and dynamic search in numerical optimization", M.Sc. Thesis, May 2002.

[4] M. M. Millonas, "Swarms, phase transitions, and collective intelligence". In: Artificial life III. Addison-Wesley, Reading, pp. 417–445, 1994.

[5] R. D. Maia, L. N. de Castro, and W. M. Caminhas, "Bee colonies as model for multimodal continuous optimization: the OptBees algorithm", In: Proc. of the 2012 IEEE World Congress on Computational Intelligence (WCCI), pp. 3316–3323, 2012.

[6] R. D. Maia, L. N. de Castro, and W. M. Caminhas, "Collective decision-making by bee colonies as model for optimization - the OptBees algorithm". Applied Mathematical Sciences, Vol. 7, 2013, no. 87, 4327–4351. http://dx.doi.org/10.12988/ams.2013.35271

[7] J. Gadau, and J. Fewell (eds.), "Organization of insect societies: from genome to sociocomplexity", Cambridge: Harward Univesity Press, 2009.

[8] Y. Tseng and C. Chen, "Multiple trajectory search for large scale global optimization," in Proc. of the 2008 IEEE Congress on Evol. Computation, CEC (IEEE World Congress on Computational Intelligence), 2008, pp. 3052–3059.

[9] K. Tang, X. Yao, P. N. Suganthan, C. MacNish, Y. -P Chen, C. M. Chen, Z. Yang, "Benchmark functions for the CEC 2008 special session and competition on large scale global optimization". Tech report, Nature Inspired Computation and Appl. Lab., USTC, China, 2007.

[10] J. J. Liang, B-Y. Qu, P. N. Suganthan, "Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization", Technical Report 201311, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore, December 2013.

TABLE III. RESULTS FOR $D = 10$.

Function	Best	Worst	Median	Mean	Standard Deviation
1	1.261675e+00	2.242969e+03	5.015910e+02	7.841906e+02	6.959889e+02
2	8.526513e-14	4.573520e-01	9.254413e-07	9.882615e-03	6.399495e-02
3	1.261924e-11	1.427346e+01	9.693412e-03	9.213070e-01	2.940135e+00
4	1.136868e-13	6.659546e+00	7.387894e-04	2.690817e+00	2.878214e+00
5	1.999909e+01	2.000000e+01	2.000000e+01	1.999997e+01	1.285257e-04
6	1.758454e-01	5.678854e+00	2.927143e+00	3.016623e+00	1.279700e+00
7	1.136868e-13	5.166064e-01	9.836176e-02	1.561593e-01	1.399044e-01
8	0.000000e+00	2.273737e-13	1.136868e-13	1.159160e-13	5.792541e-14
9	1.989918e+00	3.482345e+01	2.089412e+01	2.083557e+01	7.575815e+00
10	3.539870e+00	3.799565e+02	2.404790e+02	2.192256e+02	1.044594e+02
11	1.253307e+02	7.754740e+02	3.741770e+02	3.927480e+02	1.632871e+02
12	1.738923e-02	3.525246e-01	1.100401e-01	1.303986e-01	7.850468e-02
13	8.961890e-02	8.979782e-01	3.895233e-01	4.161576e-01	1.827992e-01
14	8.280050e-02	9.999562e-01	3.206315e-01	3.686513e-01	1.941139e-01
15	6.309924e-01	5.859628e+00	2.369609e+00	2.438871e+00	1.245568e+00
16	1.646939e+00	3.433115e+00	2.694207e+00	2.639589e+00	3.936085e-01
17	7.459763e+00	5.101045e+03	3.413958e+02	6.843999e+02	9.403682e+02
18	1.346650e+00	2.870701e+02	2.117447e+01	3.350430e+01	4.599888e+01
19	6.103675e-02	2.514376e+00	1.034284e+00	9.330416e-01	3.776804e-01
20	1.479826e-01	1.549413e+02	3.219025e+00	8.957556e+00	2.236439e+01
21	5.223295e-03	2.174918e+02	1.945200e+01	5.705615e+01	6.260637e+01
22	3.326307e-01	2.206911e+01	2.048040e+01	1.702468e+01	7.475871e+00
23	9.094947e-13	3.294575e+02	3.294575e+02	2.723515e+02	1.022859e+02
24	1.213682e+02	1.595922e+02	1.364543e+02	1.373780e+02	9.916332e+00
25	1.155083e+02	1.783876e+02	1.466656e+02	1.459907e+02	1.307333e+01
26	1.001178e+02	1.008855e+02	1.003349e+02	1.003964e+02	1.928558e-01
27	3.277668e+00	1.446173e+01	7.019261e+00	7.422917e+00	2.415321e+00
28	3.063329e+02	3.071682e+02	3.066547e+02	3.066715e+02	1.960580e-01
29	2.035722e+02	3.158630e+02	2.115400e+02	2.199635e+02	2.217210e+01
30	2.568144e+02	5.751023e+02	3.876258e+02	3.891652e+02	7.953363e+01

TABLE IV. RESULTS FOR $D = 30$.

Function	Best	Worst	Median	Mean	Standard Deviation
1	2.128259e+03	2.015254e+06	1.449797e+04	8.568108e+04	3.042808e+05
2	1.421086e-13	7.335643e-11	3.979039e-13	3.317538e-12	1.140735e-11
3	2.074785e-11	2.661897e-01	3.280730e-06	8.413625e-03	3.770659e-02
4	1.591616e-12	7.923307e+01	1.780862e+01	1.256348e+01	1.370719e+01
5	1.999993e+01	2.000002e+01	2.000000e+01	2.000000e+01	1.020580e-05
6	9.971452e+00	2.540501e+01	1.610041e+01	1.637620e+01	3.440187e+00
7	5.684342e-13	1.840124e-01	2.712538e-02	3.745587e-02	3.819787e-02
8	2.273737e-13	5.684342e-13	3.410605e-13	3.633520e-13	1.042443e-13
9	7.362678e+01	2.367972e+02	1.353126e+02	1.371473e+02	3.241943e+01
10	5.937099e+02	1.659468e+03	1.067921e+03	1.041298e+03	2.521947e+02
11	1.425681e+03	4.058494e+03	2.755178e+03	2.716635e+03	5.683141e+02
12	6.907802e-02	3.457577e-01	1.727159e-01	1.812459e-01	6.117389e-02
13	2.248237e-01	9.068504e-01	5.580158e-01	5.608331e-01	1.474928e-01
14	1.939904e-01	1.159490e+00	3.196832e-01	3.995474e-01	2.311388e-01
15	4.593759e+00	4.694168e+01	1.086428e+01	1.271035e+01	6.923406e+00
16	9.349669e+00	1.224433e+01	1.090429e+01	1.090813e+01	6.906300e-01
17	6.958538e+02	2.444064e+05	1.265433e+04	2.740173e+04	4.041942e+04
18	4.689211e+01	3.507524e+03	1.220316e+02	1.955904e+02	4.767984e+02
19	4.532170e+00	1.293697e+01	7.862976e+00	7.898369e+00	1.878753e+00
20	4.311325e+01	3.273664e+03	6.701585e+02	8.526656e+02	7.788849e+02
21	1.494755e+03	8.526872e+04	1.065338e+04	1.743190e+04	1.821236e+04
22	2.336580e+01	5.001664e+02	2.450420e+02	2.320181e+02	9.249958e+01
23	3.145566e+02	3.148715e+02	3.147871e+02	3.147704e+02	6.655945e-02
24	2.274264e+02	2.462545e+02	2.367150e+02	2.361550e+02	5.471342e+00
25	2.007298e+02	2.014085e+02	2.009381e+02	2.009750e+02	1.686304e-01
26	1.003022e+02	1.011311e+02	1.005128e+02	1.005510e+02	1.723199e-01
27	4.012412e+02	4.048401e+02	4.020528e+02	4.024000e+02	9.765229e-01
28	4.036819e+02	4.586777e+02	4.277527e+02	4.313996e+02	1.524469e+01
29	2.125023e+02	2.175793e+02	2.160027e+02	2.159010e+02	1.176235e+00
30	3.539336e+02	8.591254e+02	5.785674e+02	5.928338e+02	9.868715e+01

TABLE V. RESULTS FOR $D = 50$.

Function	Best	Worst	Median	Mean	Standard Deviation
1	2.617628e+04	5.495306e+05	1.093822e+05	1.399895e+05	9.700890e+04
2	1.258329e-08	1.363155e-02	2.539062e-05	1.007515e-03	2.951449e-03
3	5.179866e-01	7.584468e+02	1.003823e+02	1.710623e+02	1.799747e+02
4	4.661160e-12	1.399486e+02	3.307430e+01	3.887269e+01	3.450702e+01
5	1.999999e+01	2.000016e+01	2.000000e+01	2.000001e+01	2.596923e-05
6	1.863885e+01	4.078708e+01	3.002901e+01	3.038208e+01	4.527025e+00
7	1.250555e-12	2.192783e-01	9.857285e-03	2.382861e-02	3.930165e-02
8	3.410605e-13	1.364242e-12	6.821210e-13	6.709753e-13	1.985519e-13
9	1.452637e+02	3.532047e+02	2.467485e+02	2.470609e+02	4.863315e+01
10	1.069273e+03	2.843562e+03	1.780572e+03	1.839321e+03	3.424270e+02
11	3.375450e+03	6.602790e+03	5.024166e+03	5.132490e+03	6.832365e+02
12	7.808634e-02	3.196506e-01	1.540429e-01	1.607579e-01	4.687016e-02
13	3.561553e-01	8.149059e-01	5.944466e-01	6.046181e-01	1.159827e-01
14	2.312876e-01	1.073320e+00	3.812685e-01	4.731432e-01	2.341573e-01
15	1.213786e+01	6.204396e+01	2.405014e+01	2.622750e+01	9.193322e+00
16	1.734696e+01	2.027107e+01	1.897125e+01	1.891172e+01	8.794893e-01
17	3.878289e+03	1.653589e+05	2.027864e+04	3.259546e+04	3.515002e+04
18	8.293306e+01	3.798608e+03	2.370893e+02	6.842287e+02	1.085504e+03
19	9.513997e+00	2.184479e+01	1.516925e+01	1.521910e+01	2.781164e+00
20	5.367037e+02	8.095507e+03	1.486798e+03	1.928721e+03	1.457190e+03
21	2.813547e+03	1.712816e+05	5.459435e+04	6.008911e+04	4.130791e+04
22	2.677056e+02	1.232272e+03	7.527340e+02	7.583594e+02	2.006227e+02
23	3.375432e+02	3.387941e+02	3.381861e+02	3.381523e+02	2.832484e-01
24	2.594103e+02	2.788712e+02	2.700276e+02	2.696632e+02	3.255931e+00
25	2.012042e+02	2.043458e+02	2.020610e+02	2.021852e+02	6.086605e-01
26	1.003582e+02	1.008554e+02	1.005482e+02	1.005664e+02	1.212595e-01
27	4.160825e+02	1.492730e+03	1.164281e+03	1.138903e+03	2.158727e+02
28	4.024549e+02	5.730176e+02	4.314004e+02	4.446973e+02	3.813471e+01
29	2.258692e+02	2.369583e+02	2.318918e+02	2.319770e+02	2.384018e+00
30	5.884245e+02	1.061502e+03	8.277122e+02	8.182176e+02	1.292997e+02

TABLE VI. RESULTS FOR $D = 100$.

Function	Best	Worst	Median	Mean	Standard Deviation
1	1.329004e+05	5.777974e+05	2.821950e+05	2.998119e+05	1.002993e+05
2	5.345442e-03	1.460913e+02	1.395706e+00	1.089401e+01	2.921521e+01
3	2.965819e+01	3.794404e+03	4.760710e+02	7.501311e+02	7.863212e+02
4	3.994568e+00	2.915990e+02	1.445005e+02	1.423510e+02	5.475983e+01
5	2.000000e+01	2.000010e+01	2.000000e+01	2.000001e+01	2.493669e-05
6	5.710250e+01	9.014743e+01	7.093542e+01	7.146720e+01	7.729229e+00
7	3.069545e-12	2.712538e-02	6.593837e-12	5.890736e-03	8.046122e-03
8	9.094947e-13	2.273737e-12	1.364242e-12	1.424429e-12	3.045050e-13
9	4.745929e+02	9.302746e+02	6.467167e+02	6.656778e+02	9.791608e+01
10	2.847064e+03	4.978270e+03	4.382945e+03	4.256601e+03	4.299952e+02
11	9.494610e+03	1.484734e+04	1.226856e+04	1.236003e+04	1.110371e+03
12	1.456756e-01	3.511581e-01	2.254343e-01	2.316516e-01	5.078015e-02
13	4.002629e-01	7.983901e-01	5.850623e-01	5.896901e-01	8.163608e-02
14	1.807082e-01	2.710252e-01	2.278829e-01	2.269719e-01	2.270765e-02
15	4.080465e+01	1.253867e+02	6.288085e+01	6.593136e+01	1.829844e+01
16	3.801807e+01	4.342127e+01	4.099111e+01	4.090655e+01	1.173099e+00
17	2.170690e+04	2.755429e+05	8.997073e+04	1.094500e+05	6.739481e+04
18	2.738600e+02	6.594198e+03	5.666371e+02	1.593554e+03	2.110345e+03
19	2.888583e+01	1.118899e+02	5.210050e+01	5.283433e+01	1.564400e+01
20	3.430015e+03	2.011230e+04	9.705619e+03	1.062428e+04	4.235490e+03
21	5.585455e+04	9.653459e+05	2.698928e+05	3.107895e+05	1.596048e+05
22	1.238000e+03	2.595178e+03	2.007131e+03	2.027456e+03	3.309572e+02
23	3.451288e+02	3.490721e+02	3.459302e+02	3.462065e+02	9.282852e-01
24	3.313020e+02	3.622777e+02	3.554554e+02	3.489227e+02	1.049998e+01
25	2.060476e+02	2.119307e+02	2.080475e+02	2.081813e+02	1.124885e+00
26	1.004603e+02	1.007565e+02	1.006335e+02	1.006235e+02	6.835259e-02
27	1.728359e+03	2.559276e+03	2.175416e+03	2.160226e+03	1.671294e+02
28	5.478686e+02	7.346008e+02	6.056181e+02	6.137071e+02	4.041927e+01
29	2.671713e+02	2.821926e+02	2.748931e+02	2.748131e+02	3.274910e+00
30	2.174755e+03	3.280027e+03	2.890718e+03	2.855360e+03	2.420633e+0