OptBees – A Bee-inspired Algorithm for Solving Continuous Optimization Problems

Renato Dourado Maia  
Computer Science Department  
State University of Montes Claros  
Montes Claros, MG, Brazil  
renato.dourado@unimontes.br

Leandro Nunes de Castro  
Natural Computing Laboratory (LCoN)  
Graduate Program in Electrical Engineering  
Mackenzie University, São Paulo, Brazil  
luneses@mackenzie.br

Walmir Matos Caminhas  
Computational Intelligence Laboratory  
Electronic Engineering Department  
Federal University of Minas Gerais, MG, Brazil  
caminhas@cpdee.ufmg.br

Abstract — OptBees is an algorithm inspired by the processes of collective decision-making by bee colonies designed with the objective of generating and maintaining diversity, trading off exploitation (diversification) and exploration (intensification) and promoting a multimodal search, so that a broader coverage of promising regions of the search space can be achieved, allowing the determination of locally optimal solutions and/or multiple global optimal solutions. In this paper, the OptBees is presented in detail and its performance is evaluated, in terms of global search, in all twenty-five minimization problems proposed for the Optimization Competition of Real Parameters of the CEC 2005 Special Session on Real-Parameter Optimization. The results obtained show that OptBees is competitive when the goal is just to obtain the best possible solution, without being necessary to determine locally optimal solutions and/or multiple global optimal solutions.

Keywords — sociobiology, insects, bees, collective decision-making, optimization, multimodality, diversity.

1. 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.

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.

In this context, the main purposes of this paper are: 1) to present OptBees, a new bee-inspired algorithm for optimization in continuous spaces; 2) and to evaluate its performance, specifically in terms of global search capabilities, by applying it to all twenty-five minimization problems proposed for the
IEEE 2005 Congress on Evolutionary Computation Competition (CEC 2005), considering ten and thirty-dimensional spaces. Although a performance evaluation of OptBees was presented in [5], it was very preliminary. The goal of the current performance evaluation is not to compare OptBees with other bee-inspired algorithms for continuous optimization. Instead, it is benchmarked against the best performances known to date for the CEC 2005 competition problems, which represent the state-of-the-art in continuous search and optimization techniques. 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 new bee-inspired algorithm for continuous optimization problem solving, 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. 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 one 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:
- $n_{\text{min}}$: initial number of active bees.
- $n_{\text{max}}$: maximum number of active bees.
- $\rho$: inhibition radius.
- $n_{\text{mean}}$: average foraging effort.
- $p_{\text{rec}}$: minimum probability of a bee being a recruiter.
- $p_{\text{rec}}$: 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.

\begin{algorithm}
   \textbf{while} (stopping criterion is not attained) \textbf{do}
   \begin{enumerate}
      \item Evaluate the quality of the sites being explored by the active bees.
      \item Apply local search.
      \item Determine the recruiter bees.
      \item Update the number of active bees.
      \item Determine the recruited and scout bees.
      \item Perform the recruitment process.
      \item Perform the exploration process.
   \end{enumerate}
   \textbf{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 $n_1$ 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 $\max f(x) = -f(x)$).

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, for thirty iterations, using half of the domain of the variable being changed as the search range.

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 $q_{\text{min}}$ and $q_{\text{max}}$ 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 $p_{\text{min}}$ defines the minimum probability of a bee to be a recruiter.

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

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 $n_{\text{random}}$ belonging to the interval $[0, 1]$ is generated for each bee $i$ so that, the higher the $p_i$ value, the more likely the bee $i$ is classified as recruiter: if $n_{\text{random}}$ is smaller than $p_i$, then 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 $\rho$, are inhibited, i.e., they become classified as non-recruiters. Let $d(i, j)$ be the Euclidean 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 $n_{\text{mean}}$ determines the desired number of non-recruiter bees for each recruiter bee, i.e., in a given iteration, the number $n_d = (r + 1)n_{\text{mean}}$ determines the desired number of active bees. If this number $n_d$ is greater than the current number of active bees $n_{\text{active}}$, $n_{\text{adjust}} = n_d - n_{\text{active}}$ 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 $n_{\text{active}}$, $n_{\text{adjust}} = n_{\text{active}} - n_d$ is the necessary number of bees that have to become inactive in order to achieve $n_d$ active bees. This process respects the maximum ($n_{\text{max}}$) and minimum ($n_{\text{min}}$) number of active bees (the minimum number of active bees $n_{\text{min}}$ is equal to the initial number of active bees $n_1$): if $n_d > n_{\text{max}}$, then $n_d$ is forced to $n_{\text{max}}$; if $n_d < n_{\text{min}}$, then $n_d$ is forced to $n_{\text{min}}$. 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 $n_{\text{rec}} = n_{\text{active}} - r$ ($r$ 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 \( n_r = \left\lfloor p_{rec} n_r \right\rfloor \) (\( \left\lfloor \cdot \right\rfloor \) 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 \( n_{ri} \) 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 \( n_{ri} (i = 1, 2, \ldots, r) \) are calculated using the expression \( n_{ri} = \left\lfloor (q_i/Q_{recruiters}) n_r \right\rfloor \) (\( \left\lfloor \cdot \right\rfloor \) denotes the nearest integer function and \( Q_{recruiters} = \sum_{k=1}^{r} q_k \))

With the number \( n_{ri} \) 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 \( n_{nr} \) 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.

\[
x_i = x_i + 2 \cdot U \otimes (y - x_i) \quad (2)
\]

\[
x_i = x_i + 2 \cdot u \cdot (y - x_i) \quad (3)
\]

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 \( x_i + 2(y - x_i) \), while using Equation 2 the recruited bee will be positioned in any point inside the dashed rectangle.

\[
x_i + 2 \cdot (y - x_i)
\]

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, in terms of global search, experiments were performed based on the set of test problems proposed for the Competition on Real Parameters Optimization of the CEC 2005 Special Session on Real-Parameter Optimization (CEC 2005 Competition), which occurred in the IEEE Congress on Evolutionary Computation (CEC) in 2005 [9]. The following four reasons justify the choice of this problem set for the experiments [10]: 1) the problems include many features that may occur in real optimization situations, which allow the algorithm to have its performance evaluated in a comprehensive manner, without being favored by a specific class of problems; 2) the fact that the problem set is associated with a competition ensures that the results reported for each of the algorithms were generated using a well-defined and standardized experimental methodology, which allows a direct comparison of these results with those obtained by other algorithms using the same experimental methodology to solve the problems without requiring the implementation and execution of tests for the algorithms that participate in the competition; 3) although the competition took place in 2005, some of the algorithms that presented the best results are still considered state-of-the-art and new algorithms are constantly being proposed and used for assessment, which makes them relevant for performance comparison; and 4) the problems are scalable, allowing, in a simple way, the execution of tests for different numbers of dimensions.

The CEC 2005 Competition problem set consists of twenty five minimization mono-objective problems in continuous spaces, which may include the following characteristics: mono or multimodality, large number of local optima, presence or not of noise in the objective-function value, dependence or not between variables, presence of plateaus, global optimum located within the search space boundaries, non-differentiability in some search space points. Details about these problems can be found in [9].

For the experiments, the Matlab® version of the original implementation for all CEC 2005 Competition problems, available at the following address, was used:

http://www3.ntu.edu.sg/home/EPNSugan/index_files/CEC-05/CEC05.htm
(last accessed on May 31, 2012).

The experiments aimed to evaluate the performance of OptBees in high-dimensional spaces through the comparison of its results with those obtained by several other tools (the state of the art) available in the literature. For this, the twenty-five problems of the CEC 2005 Competition set were considered, in spaces of ten (\( D = 10 \)) and thirty (\( D = 30 \)) dimensions, using the standard methodology defined by the competition organization. So, the results obtained by OptBees could be compared directly with the results presented by all other algorithms that participated in the competition. In addition to the
algorithms that participated in the competition, three other algorithms were considered for comparison – cob-aiNet [10] [11], opt-aiNet [12] and dop-aiNet [13], because in [10] results were reported for the twenty-five CEC’2005 competition problems, for these three algorithms, considering the standard methodology defined by the competition organization, and because cob-aiNet is a more recent tool, proposed in 2010.

A. Results Presentation and Discussion

The experimental methodology used followed the guidelines of the competition [9], which are presented below, so that results can be directly compared with those reported by the competition organization and the other more recent algorithms also applied to these problems:

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

The parameterization of OptBees considered the fact that, in the CEC 2005 Competition, the goal is to find the global optimum and not the location of the maximum number of local optima solutions. Thus, the values used for the parameters, defined in preliminary tests, which were the same for $D = 10$ and $D = 30$, were: minimum number of active bees $n_{\text{min}} = 200; \$ maximum number of active bees $n_{\text{max}} = 1500; \$ average foraging effort $n_{\text{mean}} = 20; \$ social inhibition radius $\rho$ 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_{\text{rec}} = 0.01; \$ percentage of non-recruiter bees that will be actually recruited $p_{\text{rec}}$ varying linearly with the number of function evaluations between 0.5 and 1.

It is important to highlight that the organizing committee of the competition discouraged participants to seek a separate set of parameters for each problem [9]. Therefore, in this paper the same set of parameters was used in all experiments performed, even though some competitors have not followed strictly this guideline. Moreover, by using a single set of parameters, the results obtained can be used as a means to indirectly assess the robustness of the algorithm in relation to its own parameters.

Table I shows de values\(^1\) for the mean and standard deviation of the absolute error between the best found solution and the global optimum of the problem (AAE) for the OptBees algorithm, for each problem, for ten and thirty dimensions. Due to space limitations, this paper will not present the results for all the algorithms. Tables II and III present references in which these results can be consulted, besides details about the corresponding algorithms and their parameterizations.

Tables II and III show the ranks of each algorithm, for each problem, obtained by sorting the values of the average absolute error (AAE) in ascending order, in addition to the average ranks, the standard deviation of the ranks and the final rank for each algorithm (final ranks were determined from the average ranks), for ten and thirty dimensions, respectively. As one of the stopping criteria used in the competition was to obtain an absolute error between the best solution and the global optimum less than or equal to $1 \cdot 10^{-8}$, all values of AAE less than or equal to this value was taken as zero, so that the algorithms that ended its run due to this stopping criterion were not harm in the determination of their ranks. It is worth mentioning the fact that no results were reported for problems $F_{21}$ to $F_{25}$, with $D = 30$, for L-DMS-PSO\(^2\), nor for problems $F_{16}$ to $F_{25}$, for L-SaDE, also in case of thirty dimensions: in these cases, both algorithms were classified as rank 15, corresponding to the worst rank.

\[\begin{array}{|c|c|c|}
\hline
\text{Algorithm} & \text{Absolute Average Error} & \text{D = 10} \\
\hline
F_1 & 3.4107 \cdot 10^{-11} \pm 2.8422 \cdot 10^{-18} & 1.5234 \cdot 10^{-12} \pm 3.9245 \cdot 10^{-14} \\
F_2 & 2.9559 \cdot 10^{-12} \pm 1.8419 \cdot 10^{-12} & 6.3346 \cdot 10^{-12} \pm 2.0107 \cdot 10^{-11} \\
F_3 & 4.5755 \cdot 10^{-4} \pm 4.7478 \cdot 10^{-3} & 2.7438 \cdot 10^{-5} \pm 1.5701 \cdot 10^{-4} \\
F_4 & 1.8305 \cdot 10^{-4} \pm 1.5951 \cdot 10^{-1} & 1.1500 \cdot 10^{-6} \pm 3.7215 \cdot 10^{-5} \\
F_5 & 3.0777 \cdot 10^{-11} \pm 8.8556 \cdot 10^{-11} & 5.8429 \cdot 10^{-5} \pm 1.9718 \cdot 10^{-4} \\
F_6 & 2.5068 \cdot 10^{-12} \pm 1.2266 \cdot 10^{-9} & 1.1644 \cdot 10^{-13} \pm 1.8610 \cdot 10^{-11} \\
F_7 & 1.2171 \cdot 10^{-8} \pm 7.9280 \cdot 10^{-1} & 1.3588 \cdot 10^{-2} \pm 1.2878 \cdot 10^{-2} \\
F_8 & 2.0041 \cdot 10^{-12} \pm 2.9824 \cdot 10^{-2} & 2.0074 \cdot 10^{-4} \pm 4.0015 \cdot 10^{-2} \\
F_9 & 4.7479 \cdot 10^{-14} \pm 3.5499 \cdot 10^{-14} & 1.6598 \cdot 10^{-13} \pm 4.9009 \cdot 10^{-14} \\
\hline
\end{array}\]

\(^1\) The results were presented in scientific notation, using four decimal digits. It is important to emphasize, however, that the results reported in the literature for algorithms DE, G-CMA-ES, K-PCX and L-CMA-ES have only two decimal digits, a fact that may influence the ordering of results and determination of the ranks of each algorithm.

\(^2\) In this work, the same acronym used by the CEC’2005 competition organization in the document that reported the results to assign each of the algorithms that participated in the competition will be adopted.
The memetic algorithm BLX-MA achieved a worse performance than OptBees for ten and also for thirty dimensions. The two algorithms based on differential evolution – L-SaDE and DE – achieved a better performance than OptBees for ten dimensions, but worse for thirty.

DMS-L-PSO, an algorithm that, as OptBees, belongs to the Swarm Intelligence paradigm, achieved a better performance than OptBees for \( D = 10 \), but worse for \( D = 30 \). It is important to highlight again that DMS-L-PSO failed to converge for five problems with \( D = 30 (F_{21} \text{ to } F_{25}) \), which reflects its low robustness in relation to an increase in dimensionality.

Table V also shows the final general rank, obtained from the average rank for \( D = 10 \) and \( D = 30 \). OptBees reached rank three, surpassed by G-CMA-ES, an estimation of distribution algorithm, and BLX-GL50, a generational genetic algorithm, the only algorithms that have achieved better average performance than OptBees for \( D = 10 \) and \( D = 30 \). This fact shows that OptBees is also a competitive tool in the context where the goal is just to obtain the best possible solution, without being necessary to determine locally optima solutions and/or multiple global optimal solutions.

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 twenty-five minimization problems proposed for the Optimization Competition of Real Parameters of the CEC 2005 Special Session on Real-Parameter Optimization, considering 10 and 30 dimensions.

The performance of OptBees was compared with that of fourteen algorithms: the eleven CEC’2005 Competition competitors, plus opt-aiNet, dopt-aiNet and cob-aiNet. For ten dimensions, OptBees achieved rank six, overcoming the algorithms SPC-PNX, cob-aiNet, L-CMA-ES, EDA, BLX-MA, K-PCX, CoEVO, opt-aiNet and dopt-aiNet. For thirty dimensions, OptBees achieved rank five, overcoming the algorithms BLX-MA, cob-aiNet, DE, SPC-PNX, DMS-L-PSO, K-PCX, L-SaDE, CoEVO, opt-aiNet and dopt-aiNet. OptBees reached the final general rank three, surpassed by G-CMA-ES, an estimation of distribution algorithm, and BLX-GL50, a generational genetic algorithm, the only algorithms that have achieved better average performance than OptBees for ten and thirty dimensions.

These results show that OptBees is also a competitive tool in the context where the goal is just to obtain the best possible solution, without being necessary to determine locally optimal solutions and/or multiple global optimal solutions.

For future research, the authors plan to do a quantitative analysis of the diversity generation and maintenance, and the capacity of determining locally optimal solutions and/or multiple global optima. Another future research is to adapt OptBees for solving clustering and classification problems. Finally, OptBees will be extended for solving constrained, multi-objective and combinatorial optimization problems.
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>14</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>F2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>14</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>F3</td>
<td>8</td>
<td>12</td>
<td>1</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>9</td>
<td>5</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>6,64</td>
<td>14</td>
<td>9</td>
<td>1</td>
</tr>
<tr>
<td>F4</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>1</td>
<td>10</td>
<td>1</td>
<td>8</td>
<td>15</td>
<td>9</td>
<td>1</td>
<td>13</td>
<td>1</td>
<td>14</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>F5</td>
<td>1</td>
<td>10</td>
<td>12</td>
<td>1</td>
<td>8</td>
<td>1</td>
<td>13</td>
<td>1</td>
<td>9</td>
<td>1</td>
<td>14</td>
<td>1</td>
<td>15</td>
<td>11</td>
<td>1</td>
</tr>
<tr>
<td>F6</td>
<td>1</td>
<td>9</td>
<td>10</td>
<td>8</td>
<td>6</td>
<td>7</td>
<td>12</td>
<td>1</td>
<td>5</td>
<td>14</td>
<td>15</td>
<td>14</td>
<td>15</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>F7</td>
<td>3</td>
<td>10</td>
<td>5</td>
<td>9</td>
<td>6</td>
<td>12</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>11</td>
<td>15</td>
<td>14</td>
<td>15</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>F8</td>
<td>12</td>
<td>8</td>
<td>9</td>
<td>14</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>14</td>
<td>10</td>
<td>15</td>
<td>13</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>F9</td>
<td>9</td>
<td>7</td>
<td>12</td>
<td>8</td>
<td>1</td>
<td>11</td>
<td>6</td>
<td>5</td>
<td>14</td>
<td>1</td>
<td>10</td>
<td>15</td>
<td>15</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>F10</td>
<td>5</td>
<td>7</td>
<td>11</td>
<td>9</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>13</td>
<td>4</td>
<td>8</td>
<td>14</td>
<td>15</td>
<td>15</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>F11</td>
<td>4</td>
<td>7</td>
<td>14</td>
<td>1</td>
<td>8</td>
<td>6</td>
<td>2</td>
<td>15</td>
<td>5</td>
<td>10</td>
<td>13</td>
<td>12</td>
<td>11</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>F12</td>
<td>10</td>
<td>6</td>
<td>12</td>
<td>5</td>
<td>2</td>
<td>11</td>
<td>4</td>
<td>15</td>
<td>7</td>
<td>8</td>
<td>13</td>
<td>14</td>
<td>13</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>F13</td>
<td>7</td>
<td>8</td>
<td>12</td>
<td>11</td>
<td>2</td>
<td>13</td>
<td>6</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>14</td>
<td>15</td>
<td>14</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>F14</td>
<td>2</td>
<td>1</td>
<td>12</td>
<td>10</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>3</td>
<td>14</td>
<td>6</td>
<td>8</td>
<td>13</td>
<td>15</td>
<td>11</td>
<td>9</td>
</tr>
<tr>
<td>F15</td>
<td>14</td>
<td>9</td>
<td>10</td>
<td>8</td>
<td>2</td>
<td>12</td>
<td>6</td>
<td>15</td>
<td>5</td>
<td>4</td>
<td>7</td>
<td>13</td>
<td>11</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>F16</td>
<td>2</td>
<td>6</td>
<td>13</td>
<td>9</td>
<td>3</td>
<td>10</td>
<td>1</td>
<td>4</td>
<td>7</td>
<td>5</td>
<td>8</td>
<td>14</td>
<td>15</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>F17</td>
<td>2</td>
<td>8</td>
<td>12</td>
<td>5</td>
<td>3</td>
<td>9</td>
<td>7</td>
<td>1</td>
<td>15</td>
<td>4</td>
<td>6</td>
<td>13</td>
<td>14</td>
<td>10</td>
<td>11</td>
</tr>
<tr>
<td>F18</td>
<td>3</td>
<td>13</td>
<td>14</td>
<td>2</td>
<td>12</td>
<td>7</td>
<td>11</td>
<td>9</td>
<td>10</td>
<td>5</td>
<td>6</td>
<td>15</td>
<td>4</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>F19</td>
<td>4</td>
<td>13</td>
<td>14</td>
<td>3</td>
<td>11</td>
<td>9</td>
<td>12</td>
<td>7</td>
<td>10</td>
<td>2</td>
<td>6</td>
<td>15</td>
<td>5</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>F20</td>
<td>4</td>
<td>11</td>
<td>14</td>
<td>5</td>
<td>13</td>
<td>9</td>
<td>12</td>
<td>3</td>
<td>10</td>
<td>2</td>
<td>7</td>
<td>15</td>
<td>6</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>F21</td>
<td>12</td>
<td>13</td>
<td>10</td>
<td>7</td>
<td>9</td>
<td>6</td>
<td>8</td>
<td>14</td>
<td>4</td>
<td>5</td>
<td>11</td>
<td>3</td>
<td>15</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>F22</td>
<td>10</td>
<td>2</td>
<td>14</td>
<td>4</td>
<td>3</td>
<td>12</td>
<td>5</td>
<td>11</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>15</td>
<td>6</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>F23</td>
<td>7</td>
<td>13</td>
<td>12</td>
<td>5</td>
<td>10</td>
<td>8</td>
<td>4</td>
<td>14</td>
<td>11</td>
<td>6</td>
<td>1</td>
<td>15</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>F24</td>
<td>1</td>
<td>9</td>
<td>12</td>
<td>1</td>
<td>9</td>
<td>1</td>
<td>13</td>
<td>14</td>
<td>1</td>
<td>11</td>
<td>14</td>
<td>15</td>
<td>15</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>F25</td>
<td>10</td>
<td>8</td>
<td>1</td>
<td>14</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>11</td>
<td>13</td>
<td>6</td>
<td>12</td>
<td>15</td>
<td>13</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>Average</td>
<td>5,36</td>
<td>7,96</td>
<td>9,56</td>
<td>5,92</td>
<td>5,24</td>
<td>7,2</td>
<td>2,96</td>
<td>8,2</td>
<td>7</td>
<td>5,16</td>
<td>6,64</td>
<td>11,24</td>
<td>14,28</td>
<td>6,68</td>
<td>6,6</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>4,17</td>
<td>3,71</td>
<td>4,78</td>
<td>4,04</td>
<td>3,99</td>
<td>3,96</td>
<td>2,51</td>
<td>5,31</td>
<td>5,25</td>
<td>3,35</td>
<td>4,17</td>
<td>4,21</td>
<td>1,37</td>
<td>4,35</td>
<td>4,40</td>
</tr>
<tr>
<td>Final Rank</td>
<td>4</td>
<td>11</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>10</td>
<td>1</td>
<td>12</td>
<td>9</td>
<td>2</td>
<td>7</td>
<td>14</td>
<td>15</td>
<td>8</td>
<td>6</td>
</tr>
</tbody>
</table>
### TABLE III.

**Rank of each algorithm for the CEC’2005 test problems ($D = 30$). The last three rows show, respectively, the average rank, the standard deviation of the rank and final rank, obtained from the average rank. The best results are marked in bold.**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$F_1$</td>
<td>1</td>
<td>1</td>
<td>13</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$F_2$</td>
<td>1</td>
<td>10</td>
<td>13</td>
<td>12</td>
<td>8</td>
<td>1</td>
<td>1</td>
<td>11</td>
<td>1</td>
<td>7</td>
<td>9</td>
<td>1</td>
<td>16</td>
<td>15</td>
<td>1</td>
</tr>
<tr>
<td>$F_3$</td>
<td>6</td>
<td>11</td>
<td>5</td>
<td>10</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>9</td>
<td>1</td>
<td>8</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>12</td>
<td>7</td>
</tr>
<tr>
<td>$F_4$</td>
<td>5</td>
<td>6</td>
<td>9</td>
<td>4</td>
<td>8</td>
<td>1</td>
<td>10</td>
<td>7</td>
<td>15</td>
<td>3</td>
<td>2</td>
<td>13</td>
<td>14</td>
<td>14</td>
<td>11</td>
</tr>
<tr>
<td>$F_5$</td>
<td>5</td>
<td>8</td>
<td>13</td>
<td>4</td>
<td>9</td>
<td>3</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>6</td>
<td>10</td>
<td>14</td>
<td>15</td>
<td>12</td>
<td>11</td>
</tr>
<tr>
<td>$F_6$</td>
<td>4</td>
<td>10</td>
<td>13</td>
<td>9</td>
<td>5</td>
<td>7</td>
<td>1</td>
<td>12</td>
<td>8</td>
<td>6</td>
<td>14</td>
<td>15</td>
<td>11</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$F_7$</td>
<td>1</td>
<td>9</td>
<td>13</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>12</td>
<td>7</td>
<td>11</td>
<td>14</td>
<td>15</td>
<td>8</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>$F_8$</td>
<td>13</td>
<td>8</td>
<td>9</td>
<td>15</td>
<td>1</td>
<td>12</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>11</td>
<td>10</td>
<td>14</td>
<td>6</td>
<td>4</td>
</tr>
<tr>
<td>$F_9$</td>
<td>7</td>
<td>5</td>
<td>11</td>
<td>9</td>
<td>8</td>
<td>12</td>
<td>6</td>
<td>4</td>
<td>13</td>
<td>1</td>
<td>10</td>
<td>15</td>
<td>14</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$F_{10}$</td>
<td>3</td>
<td>7</td>
<td>12</td>
<td>8</td>
<td>5</td>
<td>10</td>
<td>2</td>
<td>1</td>
<td>14</td>
<td>4</td>
<td>6</td>
<td>13</td>
<td>15</td>
<td>9</td>
<td>11</td>
</tr>
<tr>
<td>$F_{11}$</td>
<td>4</td>
<td>10</td>
<td>13</td>
<td>12</td>
<td>6</td>
<td>15</td>
<td>1</td>
<td>9</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>14</td>
<td>11</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>$F_{12}$</td>
<td>7</td>
<td>4</td>
<td>12</td>
<td>3</td>
<td>1</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>9</td>
<td>2</td>
<td>8</td>
<td>13</td>
<td>14</td>
<td>11</td>
<td>6</td>
</tr>
<tr>
<td>$F_{13}$</td>
<td>10</td>
<td>9</td>
<td>11</td>
<td>6</td>
<td>4</td>
<td>13</td>
<td>5</td>
<td>12</td>
<td>3</td>
<td>1</td>
<td>8</td>
<td>14</td>
<td>15</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>$F_{14}$</td>
<td>2</td>
<td>4</td>
<td>9</td>
<td>10</td>
<td>1</td>
<td>10</td>
<td>5</td>
<td>13</td>
<td>15</td>
<td>3</td>
<td>7</td>
<td>12</td>
<td>14</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>$F_{15}$</td>
<td>4</td>
<td>8</td>
<td>12</td>
<td>9</td>
<td>6</td>
<td>11</td>
<td>2</td>
<td>15</td>
<td>3</td>
<td>5</td>
<td>10</td>
<td>13</td>
<td>14</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>$F_{16}$</td>
<td>6</td>
<td>11</td>
<td>13</td>
<td>1</td>
<td>7</td>
<td>9</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>15</td>
<td>5</td>
<td>12</td>
<td>14</td>
<td>8</td>
<td>10</td>
</tr>
<tr>
<td>$F_{17}$</td>
<td>2</td>
<td>9</td>
<td>12</td>
<td>5</td>
<td>3</td>
<td>6</td>
<td>10</td>
<td>4</td>
<td>14</td>
<td>15</td>
<td>1</td>
<td>11</td>
<td>13</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>$F_{18}$</td>
<td>6</td>
<td>4</td>
<td>13</td>
<td>7</td>
<td>10</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>5</td>
<td>15</td>
<td>9</td>
<td>12</td>
<td>14</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td>$F_{19}$</td>
<td>6</td>
<td>4</td>
<td>13</td>
<td>7</td>
<td>11</td>
<td>2</td>
<td>7</td>
<td>1</td>
<td>5</td>
<td>15</td>
<td>10</td>
<td>12</td>
<td>14</td>
<td>9</td>
<td>3</td>
</tr>
<tr>
<td>$F_{20}$</td>
<td>8</td>
<td>4</td>
<td>13</td>
<td>9</td>
<td>7</td>
<td>3</td>
<td>9</td>
<td>1</td>
<td>5</td>
<td>15</td>
<td>11</td>
<td>12</td>
<td>14</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>$F_{21}$</td>
<td>3</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>15</td>
<td>3</td>
<td>3</td>
<td>12</td>
<td>1</td>
<td>15</td>
<td>3</td>
<td>10</td>
<td>13</td>
<td>2</td>
<td>11</td>
</tr>
<tr>
<td>$F_{22}$</td>
<td>4</td>
<td>7</td>
<td>11</td>
<td>6</td>
<td>15</td>
<td>3</td>
<td>1</td>
<td>13</td>
<td>2</td>
<td>15</td>
<td>5</td>
<td>10</td>
<td>12</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>$F_{23}$</td>
<td>8</td>
<td>7</td>
<td>12</td>
<td>2</td>
<td>15</td>
<td>4</td>
<td>2</td>
<td>11</td>
<td>6</td>
<td>15</td>
<td>4</td>
<td>9</td>
<td>13</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>$F_{24}$</td>
<td>9</td>
<td>1</td>
<td>11</td>
<td>1</td>
<td>15</td>
<td>1</td>
<td>10</td>
<td>6</td>
<td>12</td>
<td>15</td>
<td>1</td>
<td>8</td>
<td>13</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>$F_{25}$</td>
<td>2</td>
<td>4</td>
<td>12</td>
<td>11</td>
<td>15</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>10</td>
<td>15</td>
<td>6</td>
<td>8</td>
<td>13</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>Average</td>
<td>5.08</td>
<td>6.56</td>
<td>11.48</td>
<td>6.76</td>
<td>7.44</td>
<td>5.48</td>
<td>4.24</td>
<td>7.48</td>
<td>5.8</td>
<td>8.72</td>
<td>6.76</td>
<td>12.2</td>
<td>13.92</td>
<td>6.72</td>
<td>6.32</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>3.04</td>
<td>3.00</td>
<td>1.96</td>
<td>3.85</td>
<td>4.73</td>
<td>4.61</td>
<td>3.43</td>
<td>4.93</td>
<td>5.22</td>
<td>5.59</td>
<td>3.67</td>
<td>2.02</td>
<td>1.04</td>
<td>3.97</td>
<td>3.67</td>
</tr>
<tr>
<td>Final Rank</td>
<td>2</td>
<td>6</td>
<td>13</td>
<td>8</td>
<td>10</td>
<td>3</td>
<td>1</td>
<td>11</td>
<td>4</td>
<td>12</td>
<td>8</td>
<td>14</td>
<td>15</td>
<td>15</td>
<td>7</td>
</tr>
</tbody>
</table>
### TABLE IV. RANK OF EACH ALGORITHM FOR THE CEC’2005 TEST PROBLEMS ($D = 30$). The last three rows show, respectively, the average rank, the standard deviation of the rank and final rank, obtained from the average rank. The best results are marked in bold.

<table>
<thead>
<tr>
<th>Problem</th>
<th>$F_1$</th>
<th>$F_2$</th>
<th>$F_3$</th>
<th>$F_4$</th>
<th>$F_5$</th>
<th>$F_6$</th>
<th>$F_7$</th>
<th>$F_8$</th>
<th>$F_{11}$</th>
<th>$F_{12}$</th>
<th>$F_{14}$</th>
<th>$F_{15}$</th>
<th>$F_{16}$</th>
<th>$F_{17}$</th>
<th>$F_{18}$</th>
<th>$F_{21}$</th>
<th>$F_{22}$</th>
<th>$F_{23}$</th>
<th>$F_{24}$</th>
<th>$F_{25}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank ($D = 10$)</td>
<td>1</td>
<td>1</td>
<td>9</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>13</td>
<td>7</td>
<td>1</td>
<td>12</td>
<td>9</td>
<td>9</td>
<td>3</td>
<td>9</td>
<td>3</td>
<td>12</td>
<td>11</td>
<td>9</td>
<td>9</td>
<td>8</td>
</tr>
<tr>
<td>Rank ($D = 30$)</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>11</td>
<td>11</td>
<td>1</td>
<td>10</td>
<td>4</td>
<td>1</td>
<td>11</td>
<td>8</td>
<td>6</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>10</td>
<td>8</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>Difference</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>-10</td>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>-4</td>
<td>2</td>
<td>3</td>
<td>6</td>
<td>6</td>
<td>6</td>
</tr>
</tbody>
</table>

### TABLE V. RANK OF EACH ALGORITHM FOR THE CEC’2005 TEST PROBLEMS ($D = 30$). The last three rows show, respectively, the average rank, the standard deviation of the rank and final rank, obtained from the average rank. The best results are marked in bold.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Rank ($D = 10$)</td>
<td>4</td>
<td>11</td>
<td>13</td>
<td>5</td>
<td>3</td>
<td>10</td>
<td>1</td>
<td>12</td>
<td>9</td>
<td>2</td>
<td>7</td>
<td>14</td>
<td>15</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td>Rank ($D = 30$)</td>
<td>2</td>
<td>6</td>
<td>13</td>
<td>8</td>
<td>10</td>
<td>3</td>
<td>1</td>
<td>11</td>
<td>4</td>
<td>12</td>
<td>8</td>
<td>14</td>
<td>15</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>Difference</td>
<td>2</td>
<td>5</td>
<td>0</td>
<td>-3</td>
<td>-7</td>
<td>7</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>-10</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Average</td>
<td>3</td>
<td>8.5</td>
<td>13</td>
<td>6.5</td>
<td>6.5</td>
<td>6.5</td>
<td>1</td>
<td>11.5</td>
<td>6.5</td>
<td>7</td>
<td>7.5</td>
<td>14</td>
<td>15</td>
<td>7.5</td>
<td>5.5</td>
</tr>
<tr>
<td>Final General Rank</td>
<td>2</td>
<td>11</td>
<td>13</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>12</td>
<td>4</td>
<td>8</td>
<td>9</td>
<td>14</td>
<td>15</td>
<td>9</td>
<td>3</td>
</tr>
</tbody>
</table>
ACKNOWLEDGMENT

The authors thank Capes, CNPq, MackPesquisa, Fapesp and Fapemig for the financial support.

REFERENCES


