The Differential Ant-Stigmergy Algorithm for Large-Scale Global Optimization

Peter Korošec, *Member, IEEE,* Katerina Tashkova, Jurij Šilc, *Member, IEEE*

***Abstract*— Ant-colony optimization (ACO) is a popular swarm intelligence metaheuristic scheme that can be applied to almost any optimization problem. In this paper, we address a performance evaluation of an ACO-based algorithm for solving large-scale global optimization problems with continuous variables, labeled Differential Ant-Stigmergy Algorithm (DASA). The DASA transforms a real-parameter optimization problem into a graph-search problem. The parameters' differences assigned to the graph vertices are used to navigate through the search space. The performance of the DASA is evaluated on the set of benchmark problems provided for CEC'2010 Special Session and Competition on Large-Scale Global Optimization.**

I. Introduction

The global numerical optimization problem can be expressed as:

$$\min F(\vec{x})$$

$$\text{subject to} \quad \vec{x}^{\,\min} \leq \vec{x} \leq \vec{x}^{\,\max}$$

where $\vec{x} = [x_1, x_2, \ldots, x_D]$ is a parameter vector in $\mathbb{R}^D$ space, $F(\vec{x})$ is the objective function, and D is the problem dimension, i.e., the number of parameters to be optimized. $\vec{x}^{\,\min} = [x_1^{\min}, x_2^{\min}, \ldots, x_D^{\min}]$ and $\vec{x}^{\,\max} = [x_1^{\max}, x_2^{\max}, \ldots, x_D^{\max}]$ define the feasible solution space, i.e., a feasible solution $\vec{x} = [x_1, x_2, \ldots, x_D]$ must satisfies $x_i^{\min} \leq x_i \leq x_i^{\max}$ for $i = 1, 2, \ldots, D$. The feasible solution space is denoted by $[\vec{x}^{\,\min}, \vec{x}^{\,\max}]$.

Many meta-heuristic algorithms are designed to solve numerical optimization problems. A major challenge is to solve high-dimensional problems of this kind (i.e., $D = 1000$), which arise in many recent real-world domains such as bio-computing, automatic frequency planning in telecommunication domain, and data or web mining [1], [2].

Continuous ant-colony optimization is an emerging field in numerical optimization. Although ant-colony optimization (ACO) has been proven to be one of the best meta-heuristics in many combinatorial optimization problems [3], its application to real-parameter optimizations appears more challenging, since the pheromone-laying method is not straightforward. So far, there have been developed a few adaptations of the ACO algorithm for continuous optimization problems (cited in [12], [5]):

- The simplified direct simulation of real ants' behaviour (ACO with discrete encoding [6], API (named after *Pachycondyla apicalis*) [7], and continuous interacting ant colony [8]).
- The extended ACO metaheuristic to explore continuous space with discretization (adaptive ant-colony algorithm [9], binary ant system [10], and multilevel ant-stigmergy algorithm [11]).
- The extended ACO metaheuristic to explore continuous space with probabilistic sampling (extended ACO [12], continuous ant-colony system [13], aggregation pheromone system [14], direct ACO [15], multivariate ant-colony algorithm for continuous optimization [16], and differential ant-stigmergy algorithm [5]).
- The hybrid methods (ACO with Levenberg-Marquard algorithm [17], ACO with Powell search method [18], ACO with genetic algorithm [19], ACO with tabu-search method [20], ACO with immune algorithm [21], and ACO with orthogonal design scheme [22]).

Peter Korošec, peter.korosec@ijs.si (corresponding author), Katerina Tashkova, katerina.taskova@ijs.si, and Jurij Šilc, jurij.silc@ijs.si, are with the Jožef Stefan Institute, Ljubljana, Slovenia.

The remainder of this paper is organized as follows: Section II introduces the optimization algorithm called the Differential Ant-Stigmergy Algorithm. Section III presents the experimental evaluation on the set of benchmark problems provided for CEC'2010 Special Session and Competition on Large-Scale Global Optimization. Section IV gives a comparison of our algorithm to some recently proposed evolutionary algorithms. Finally, Section V concludes the paper.

II. Differential Ant-Stigmergy Algorithm

A. Parameter Differences

Let x_i' be the current value of the i-th parameter. During the searching for the optimal parameter value, the new value, x_i, is assigned to the i-th parameter as follows:

$$x_i = x_i' + \delta_i. \tag{1}$$

Here, δ_i is the so-called *parameter difference* and is chosen from the set

$$\Delta_i = \Delta_i^- \cup \{0\} \cup \Delta_i^+,$$

where

$$\Delta_i^- = \{\delta_{i,k}^-|\ \delta_{i,k}^- = -b^{k+L_i-1}, k = 1, 2, \ldots, d_i\}$$

and

$$\Delta_i^+ = \{\delta_{i,k}^+|\ \delta_{i,k}^+ = b^{k+L_i-1}, k = 1, 2, \ldots, d_i\}.$$

Here, $d_i = U_i - L_i + 1$. Therefore, for each parameter x_i, the parameter difference, δ_i, has a range from b^{L_i} to b^{U_i}, where b is the so-called *discrete base*, $L_i = \lfloor \lg_b(\epsilon_i) \rfloor$, and $U_i = \lfloor \lg_b(x_i^{\max} - x_i^{\min}) \rfloor$. With the parameter ϵ_i, the maximum precision of the parameter x_i is set. The precision is limited by the computer's floating-point arithmetics. To

978-1-4244-8126-2/10/$26.00 ©2010 IEEE

enable a more flexible movement over the search space, the weight ω is added to Eq. 1:

$$x_i = x'_i + \omega\delta_i \tag{2}$$

where $\omega = \text{RandomInteger}(1, b-1)$.

B. Graph Representation

From all the sets Δ_i, $i = 1, 2, \ldots, D$, where D represents the number of parameters, the so-called *differential graph* $\mathcal{G} = (V, E)$ with a set of vertices, V, and a set of edges, E, between the vertices is constructed. Each set Δ_i is represented by the set of vertices, $V_i = \{v_{i,1}, v_{i,2}, \ldots, v_{i,2d_i+1}\}$, and $V = \bigcup_{i=1}^{D} V_i$. Then we have that

$$\Delta_i = \{\delta^-_{i,d_i}, \ldots, \delta^-_{i,d_i-j+1}, \ldots, \delta^-_{i,1}, 0, \delta^+_{i,1}, \ldots, \delta^+_{i,j}, \ldots, \delta^+_{i,d_i}\}$$

corresponds to

$$V_i = \{v_{i,1}, \ldots, v_{i,j}, \ldots, v_{i,d_i+1}, \ldots, v_{i,d_i+1+j}, \ldots, v_{i,2d_i+1}\},$$

where

$$\begin{aligned} v_{i,j} &\xrightarrow{\delta} \delta^-_{i,d_i-j+1}, \\ v_{i,d_i+1} &\xrightarrow{\delta} 0, \\ v_{i,d_i+1+j} &\xrightarrow{\delta} \delta^+_{i,j} \end{aligned}$$

and $j = 1, 2, \ldots, d_i$. Each vertex of the set V_i is connected to all the vertices that belong to the set V_{i+1}. Therefore, this is a directed graph, where each path $\vec{p}$ from the starting vertex, $v_1 \in V_1$, to any of the ending vertices, $v_D \in V_D$, is of equal length and can be defined with v_i as $\nu = (v_1 v_2 \ldots v_i \ldots v_D)$.

C. Algorithm Implementation

The optimization consists of an iterative improvement of the temporary best solution, $\vec{x}^{\,\text{tb}}$, by constructing an appropriate path $\vec{p}$. New solutions are produced by applying $\vec{p}$ to $\vec{x}^{\,\text{tb}}$ (Eq. 2).

First a solution $\vec{x}^{\,\text{tb}}$ is randomly chosen (by uniform sampling) and evaluated. Then a search graph is created and an initial amount of pheromone is deposited on the search graph according to the Cauchy probability density function:

$$C(z; z_{0_i}, s) = \frac{1}{\pi}\left(\frac{s}{(z - z_{0_i})^2 + s^2}\right),$$

where z_{0_i} is the location offset for the i-th parameter and s is the scale factor, defined as:

$$s = s_{\text{global}} - s_{\text{local}}.$$

For an initial pheromone distribution the Cauchy distribution with $s_{\text{global}} = 1$, $s_{\text{local}} = 0$, and $z_{0_i} = 0$, for $i = 1, 2, \ldots, D$ is used and each parameter vertices are equidistantly arranged between $z = [-4, 4]$.

There are n ants in a colony, all of which search simultaneously from the starting vertex. Ants use a probability rule to determine which vertex will be chosen next. The rule is based on a simple ACO. More specifically, ant a in step i moves from a vertex in set V_{i-1} to vertex $v_{i,j} \in V_i$ with a probability given by:

$$\text{prob}(a, v_{i,j}) = \frac{\tau(v_{i,j})}{\sum_{1 \le k \le 2d_i+1} \tau(v_{i,k})},$$

where $\tau(v_{i,k})$ is the amount of pheromone in vertex $v_{i,k}$.

The ants repeat this action until they reach the ending vertex. For each ant i, path $\vec{p_i}$ is constructed. If for some predetermined number of tries (in our case n^2 for all ants) we get $\vec{p_i} = \mathbf{0}$ the search process is reset by randomly choosing new $\vec{x}^{\,\text{tb}}$ and pheromone re-initialization. Otherwise, a new solution $\vec{x_i}$ is constructed.

After all ants have created solutions, they are being evaluated with a calculation of $y_i = F(\vec{x_i})$. The information about the best among them is stored as currently best information ($\vec{x}^{\,\text{cb}}$, $\vec{p}^{\,\text{cb}}$, and $y_i^{\,\text{cb}}$).

The current best solution, $\vec{x}^{\,\text{cb}}$ is compared to the temporary best solution $\vec{x}^{\,\text{tb}}$. If $y^{\,\text{cb}}$ is better than $y^{\,\text{tb}}$, then temporally best information is replaced with currently best information. In this case s_{global} is increased according to the global scale increase factor, s_+:

$$s_{\text{global}} \leftarrow (1 + s_+) s_{\text{global}},$$

s_{local} is set to

$$s_{\text{local}} = \frac{1}{2} s_{\text{global}}$$

and pheromone amount is redistributed according to the associated path $\vec{p}^{\,\text{cb}}$, where $z_{0_i} = z(p_i^{\,\text{cb}})$, so that the peak of Cauchy distribution is over the vertex of the selected path. Furthermore, if new $y^{\,\text{tb}}$ is better then global best $y^{\,\text{b}} = f(x^{\,\text{b}})$, then globally best information is replaced with temporally best information. So, global best solution is stored. If no better solution is found s_{global} is decreased according to the global scale decrease factor, s_-:

$$s_{\text{global}} \leftarrow (1 - s_-) s_{\text{global}}.$$

Pheromone evaporation is defined by some predetermined percentage ρ. The probability density function $C(z; z_{0_i}, s)$ is changed in the following way:

$$z_{0_i} \leftarrow (1 - \rho) z_{0_i}$$

and

$$s_{\text{local}} \leftarrow (1 - \rho) s_{\text{local}}.$$

Here we must emphasize that $\rho > s_-$, because otherwise we might get negative scale factor.

The whole procedure is then repeated until some ending condition is met.

The pseudocode of the algorithm is presented as follows:

Algorithm DASA

```
1:  x^tb = RndSolution()
2:  y^b = F(x^tb)
3:  y^tb = inf
4:  G = GraphInitialization(x^tb, ε)
5:  PheromoneInitialization(G)
6:  while not ending condition met do
7:      k = 0
8:      for all n ants do
9:          repeat
10:             p_i = FindPath(G)
```

```
        k = k + 1
        if k < n^2 then
          x⃗^tb = RndSolution()
          y^tb = F(x⃗^tb)
          PheromoneInitialization(G)
          goto line 7
        end if
      until (p⃗_i = 0)
      ω = RandomInteger(1, b − 1)
      x⃗_i = x⃗^tb + ωδ(p⃗)
    end for
    y^cb = inf
    for all n ants do
      y = F(x⃗_i)
      if y < y^cb then
        y^cb = y
        p⃗^cb = p⃗_i
        x⃗^cb = x⃗_i
      end if
    end for
    if y^cb < y^tb then
      y^tb = y^cb
      x⃗^tb = x⃗^cb
      s = UpadeteScales(s_global, s_local)
      PheromoneRedistribution(p⃗^cb, s)
      if y^tb < y^b then
        y^b = y^tb
        x⃗^b = x⃗^tb
      end if
    else
      UpadeteScale(s_global)
      PheromoneEvaporation(G, ρ)
    end if
  end while
```

III. Performance Evaluation

A. The Experimental Environment

The computer platform used to perform the experiments is based on AMD Opteron 2427 2.20-GHz six core processor, 8 GB of RAM, and the Microsoft Windows operating system. The DASA is implemented in C language.

B. The Benchmark Suite

The DASA algorithm is tested on 20 benchmark problems provided for the CEC'2010 Special Session and Competition on Large-Scale Global Optimization [23]. The test suite includes separable, partially-separable and nonseparable functions, where the sphere function, the rotated elliptic function, Schwefels problem 1.2, Rosenbrocks function, the rotated Rastrigins function, and the rotated Ackleys function are used as the basic functions. The test suite provides an improved platform for investigating the behavior of algorithms on high-dimensional problems in different scenarios.

a) Separable functions

- F_1: shifted elliptic function
- F_2: shifted Rastrigins function
- F_3: shifted Ackleys function

b) Single-group m-nonseparable functions

- F_4: single-group shifted and m-rotated elliptic function
- F_5: single-group shifted and m-rotated Rastrigins function
- F_6: single-group shifted and m-rotated Ackleys function
- F_7: single-group shifted m-dimensional Schwefels problem 1.2
- F_8: single-group shifted m-dimensional Rosenbrocks function

c) $\frac{D}{2m}$-group m-nonseparable functions

- F_9: $\frac{D}{2m}$-group shifted and m-rotated elliptic function
- F_{10}: $\frac{D}{2m}$-group shifted and m-rotated Rastrigins function
- F_{11}: $\frac{D}{2m}$-group shifted and m-rotated Ackleys function
- F_{12}: $\frac{D}{2m}$-group shifted m-dimensional Schwefels problem 1.2
- F_{13}: $\frac{D}{2m}$m-group shifted m-dimensional Rosenbrocks function

d) $\frac{D}{m}$-group m-nonseparable functions

- F_{14}: $\frac{D}{m}$-group shifted and m-rotated elliptic function
- F_{15}: $\frac{D}{m}$-group shifted and m-rotated Rastrigins function
- F_{16}: $\frac{D}{m}$-group shifted and m-rotated Ackleys function
- F_{17}: $\frac{D}{m}$-group shifted m-dimensional Schwefels problem 1.2
- F_{18}: $\frac{D}{m}$m-group shifted m-dimensional Rosenbrocks function

e) Nonseparable functions

- F_{19}: shifted Schwefels problem 1.2
- F_{20}: shifted Rosenbrocks function

The control parameter used to define the degree of separability of a given function, in the given test suite is set as $m = 50$. Moreover, all functions are given for the special case of dimension $D = 1000$.

C. Parameter Settings

The DASA has six parameters: the number of ants, n, the pheromone evaporation factor, ρ, the maximum parameter precision, ϵ, the discrete base, b, the global scale increase factor, s_+, and the global scale decrease factor, s_-. The algorithm parameters are set as usual [24]: $n = 10$, $\rho = 0.2$, $\epsilon = 10^{-15}$, $b = 10$, $s_+ = 0.01$, and $s_- = 0.02$. We must note that during the experimentation we did not fine-tune the algorithms parameters.

D. Testing Procedure

For each problem, the DASA is applied 25 times. The function error, $\text{Error} = F(\vec{x}) - F(\vec{x}^*)$, where $\vec{x}^*$ is the optimum, is recorded after 1.2e+05, 6.0e+05, and 3.0e+06 function evaluations (FEs). The Error is collected for $n = 25$ runs, after which the trials are ordered from best to worst. The results of the 1st (`Best`), 13th (`Median`), and 25th (`Worst`) trial, as well as the trial mean (`Mean`), and standard deviation (`Std`) are presented.

TABLE I

EXPERIMENTAL RESULTS

		F_1	F_2	F_3	F_4	F_5	F_6	F_7
FEs = 1.2e+05	Best	1.15e+05	4.65e+02	2.10e+00	2.05e+12	5.51e+08	1.98e+07	1.78e+07
	Median	1.50e+05	5.24e+02	2.34e+00	6.61e+12	8.06e+08	1.99e+07	4.14e+07
	Worst	1.94e+05	5.95e+02	2.68e+00	1.04e+13	1.05e+09	2.00e+07	1.02e+08
	Mean	1.50e+05	5.18e+02	2.35e+00	7.00e+12	8.32e+08	1.99e+07	4.53e+07
	Std	1.76e+04	2.77e+01	1.29e-01	2.21e+12	1.30e+08	5.50e+04	1.91e+07
FEs = 6.0e+05	Best	9.30e-08	5.37e+01	2.82e-07	8.01e+11	5.03e+08	1.97e+07	1.23e+05
	Median	1.73e-07	6.77e+01	4.41e-07	2.05e+12	7.45e+08	1.98e+07	1.73e+05
	Worst	3.27e-07	9.75e+01	5.77e-07	4.29e+12	9.72e+08	1.99e+07	3.00e+05
	Mean	1.86e-07	6.87e+01	4.34e-07	2.04e+12	7.52e+08	1.98e+07	1.81e+05
	Std	6.15e-08	1.05e+01	8.39e-08	8.79e+11	1.22e+08	5.02e+04	4.46e+04
FEs = 3.0e+06	Best	5.59e-23	3.98e+00	5.54e-11	2.26e+11	4.41e+08	1.96e+07	2.57e+00
	Median	5.42e-22	7.96e+00	7.37e-11	4.94e+11	6.36e+08	1.97e+07	7.18e+00
	Worst	8.44e-21	1.29e+01	8.56e-11	1.10e+12	7.45e+08	1.98e+07	1.49e+01
	Mean	1.52e-21	8.48e+00	7.20e-11	5.05e+11	6.20e+08	1.97e+07	7.78e+00
	Std	2.33e-21	2.52e+00	8.27e-12	2.22e+11	7.87e+07	4.45e+04	3.10e+00

		F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}
FEs = 1.2e+05	Best	1.00e+07	6.52e+08	7.02e+03	2.02e+02	6.40e+05	1.14e+04	1.64e+09
	Median	1.05e+08	7.58e+08	7.91e+03	2.03e+02	6.82e+05	3.04e+04	2.00e+09
	Worst	1.17e+09	9.34e+08	8.63e+03	2.03e+02	7.62e+05	4.87e+04	2.19e+09
	Mean	2.36e+08	7.67e+08	7.87e+03	2.03e+02	6.92e+05	3.11e+04	1.97e+09
	Std	3.27e+08	7.30e+07	4.55e+02	3.69e-01	3.93e+04	8.29e+03	1.39e+08
FEs = 6.0e+05	Best	1.49e+05	1.34e+08	6.81e+03	1.98e+02	1.06e+05	1.58e+03	4.23e+08
	Median	3.54e+07	1.62e+08	7.71e+03	1.99e+02	1.27e+05	1.35e+04	4.89e+08
	Worst	3.77e+08	2.17e+08	8.43e+03	1.99e+02	1.51e+05	2.50e+04	5.78e+08
	Mean	7.69e+07	1.62e+08	7.66e+03	1.99e+02	1.28e+05	1.39e+04	4.96e+08
	Std	9.40e+07	2.06e+07	4.56e+02	2.27e-01	1.03e+04	5.79e+03	3.62e+07
FEs = 3.0e+06	Best	2.84e+03	2.83e+07	6.78e+03	1.98e+02	1.40e+03	5.16e+02	8.35e+07
	Median	1.21e+06	3.58e+07	7.33e+03	1.98e+02	1.75e+03	9.72e+02	1.00e+08
	Worst	3.48e+08	5.18e+07	7.75e+03	1.99e+02	2.22e+03	3.65e+03	1.14e+08
	Mean	4.98e+07	3.60e+07	7.29e+03	1.98e+02	1.78e+03	1.21e+03	1.00e+08
	Std	8.95e+07	4.78e+06	2.69e+02	1.57e-01	2.25e+02	7.39e+02	7.88e+06

		F_{15}	F_{16}	F_{17}	F_{18}	F_{19}	F_{20}	
FEs = 1.2e+05	Best	1.43e+04	3.97e+02	1.55e+06	2.16e+05	5.01e+06	1.93e+05	
	Median	1.56e+04	3.97e+02	1.67e+06	2.80e+05	5.78e+06	2.49e+05	
	Worst	1.63e+04	3.98e+02	1.79e+06	4.41e+05	6.58e+06	3.67e+05	
	Mean	1.54e+04	3.97e+02	1.67e+06	2.90e+05	5.91e+06	2.57e+05	
	Std	5.30e+02	2.96e-01	6.06e+04	5.39e+04	4.05e+05	4.80e+04	
FEs = 6.0e+05	Best	1.40e+04	3.97e+02	3.49e+05	1.93e+04	2.28e+06	1.60e+03	
	Median	1.51e+04	3.97e+02	3.77e+05	4.20e+04	2.64e+06	2.11e+03	
	Worst	1.60e+04	3.98e+02	4.11e+05	6.53e+04	2.90e+06	4.02e+03	
	Mean	1.51e+04	3.97e+02	3.82e+05	4.30e+04	2.63e+06	2.17e+03	
	Std	5.08e+02	2.57e-01	1.99e+04	1.35e+04	1.27e+05	5.68e+02	
FEs = 3.0e+06	Best	1.38e+04	3.96e+02	8.52e+03	2.37e+03	7.18e+05	7.75e+02	
	Median	1.45e+04	3.97e+02	1.03e+04	3.95e+03	8.28e+05	1.19e+03	
	Worst	1.51e+04	3.97e+02	1.23e+04	1.09e+04	9.43e+05	1.49e+03	
	Mean	1.45e+04	3.97e+02	1.03e+04	4.92e+03	8.34e+05	1.13e+03	
	Std	3.69e+02	2.18e-01	8.99e+02	2.28e+03	5.26e+04	1.79e+02	

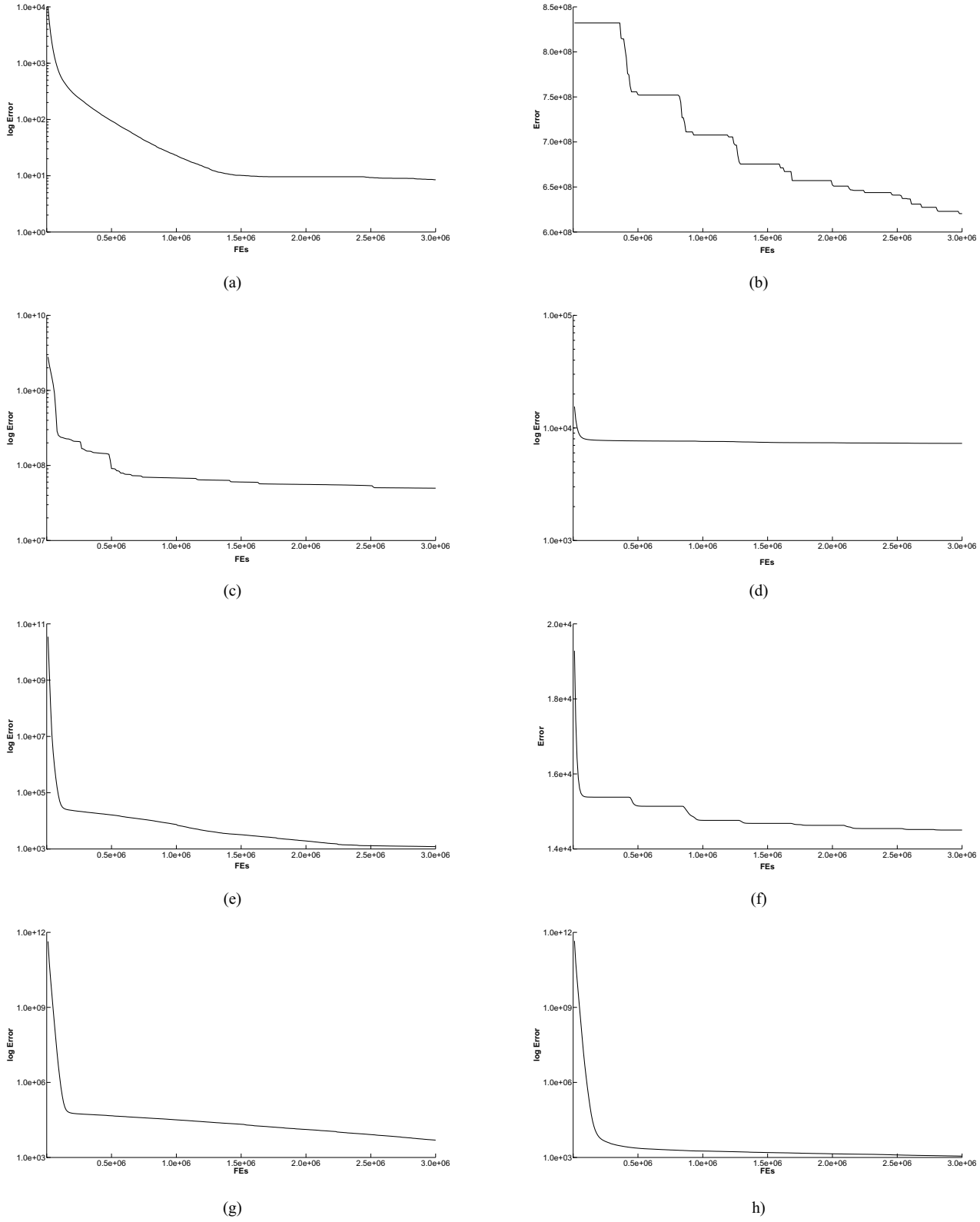

Fig. 1. Convergence graph for functions: (a) F_2; (b) F_5; (c) F_8; (d) F_{10}; (e) F_{13}; (f) F_{15}; (g) F_{18}; and (h) F_{20}.

E. Results

In Table I we present the error values achieved by the DASA algorithm on all test problems. Additionally, the convergence curves of the DASA for the following eight selected problems: F_2, F_5, F_8, F_{10}, F_{13}, F_{15}, F_{18}, and F_{20} are presented in Figure 1.

A single convergence curve captures the function error behavior with the number of function evaluations (FEs) and it is plotted using an average error results for a specific test case over all 25 runs. The convergence curves are plotted on a semi-log plots with logarithmic scale for the y-axis in order to be able to capture the convergence trend over a wide range of values.

One can notice that the DASA solved with high consistency the separable F_1 (unimodal) and F_3 (multimodal) functions, and got close to the optimum of the separable multimodal function F_2 and single-group 50-nonseparable unimodal function F_7.

Obtained solutions for Rastrigins-based nonseparable multimodal functions (F_5, F_{10}, F_{15}) and Ackleys-based nonseparable multimodal functions (F_6, F_{11}, F_{16}) are far from the optimum as result of a very slow convergence.

The results on the standard deviation reveal that overall the DASA finds consistent solutions in the space even in the cases when it is far away from the optimum, regardless of the number of function evaluations.

The convergence performance for the selected function presented in Figure 1 shows fast convergence of the DASA in the first 1.0e+05 evaluations. For few of the plotted functions this turns out to be premature, meaning that, in the rest of the given time interval, the algorithm is slowly moving in the neighborhood of a local optimum. The graph for F_{10} function is an example where the early stagnation of the DASA is clearly visible. Exceptionally slow convergence across the whole inspected interval of 3.0e+06 evaluations can be noticed for the F_2 and F_5 functions. In order to capture the function error behavior in these two cases we used linear scale for its representation as illustrated in Figure 1b and Figure 1f.

IV. Comparison with Other Algorithms

In this section, we compare the DASA with the following algorithms:

- Cooperative Coevolution (DECC-G) proposed in [25], is a framework which incorporates a random grouping scheme and adaptive weighting for problem decomposition and coevolution, and a novel differential evolution as main search algorithm. The parameter group size is set to $s = 100$ in the experimental evaluation.
- Multilevel Cooperative Coevolution (MLCC) is a framework for large-scale optimization problems proposed in [26]. It is an improved grouping based cooperative coevolution regrading the grouping strategy in the objective problem decomposition, and it is designed to self-adapt between different decomposers according to the captured historical performance.
- Differential Evolution (DE) is a simple and efficient evolutionary population-based heuristic for numerical optimization, which combines differential mutation strategy and uniform crossover operation over candidate solutions [27]. Based on a manual tuning, the algorithm parameters are set as follows: "`rand-to-best/1/bin`" strategy is used, population size is set to $NP = 60$, the weight factor to $F = 0.6$, and crossover factor to $CR = 0.3$. The tuning included testing DE with the following parameters: $F = [0.5, 0.6, 0.7, 0.85]$, $CR = [0.3, 0.5, 0.7, 1.0]$, strategy = ["`bin/1/exp`", "`rand/1/exp`", "`best/2/exp`", "`rand/2/exp`", "`best/1/bin`", "`rand/1/bin`", "`rand-to-best/1/bin`", "`rand-to-best/1/exp`", "`best/2/bin`", "`rand/2/bin`"]. In the notation "`x/y/z`", `x` represents a string denoting the vector to be perturbed (`rand`: random vector; `best`: best vector), `y` is the number of difference vectors considered for perturbation of `x` and `z` stands for the type of crossover being used (`exp`: exponential; `bin`: binomial). Population size is fixed to 60 in all cases.
  For every parameter combination the algorithm is run two times and the run is limited to 3.0e+06 function evaluations.

Compared to the preliminary median results (Table II) obtained by the algorithms DECC-G, MLCC, and DE, the DASA obtains better solutions for the single-group 50-nonseparable functions F_4 and F_8 and nonseparable functions F_{19}, F_{20}. Additionally, it obtains better solutions on the 10-group 50-nonseparable functions F_9, F_{12}, F_{13} and on the 20-group 50-nonseparable functions F_{14}, F_{17}, F_{18}. As evident in Table II, the DASA is the best ranked algorithm with average rank of 1.975.

Based on the median results given in Table II, a statistical hypothesis testing is carried out by applying the non-parametric test presented in [28]. Here, we are trying to reject the null hypothesis H_0 that two algorithms perform equally well. In general, H_0 is a statistical hypothesis that states that there are no differences between two data sets. With the significance level α a criterion for rejecting H_0 is given. For a chosen α, H_0 is rejected if p-value is less than or equal to α, where the p-value is determined according to the probability of difference between these two sets. When a p-value is observed in regard to multiple comparisons it reflects the probability error of a certain comparison, but it does not take into account the remaining comparisons. One way to solve this problem is to report adjusted p-values, which take into account that multiple test were conducted. An adjusted p-value can be compared directly with any chosen α. We applied the following methods to acquire adjusted p-values: Nemenyis test, Holm's procedure, Shaffers static procedure, and Bergmann-Hommel's dynamic procedure. The values of the obtained p-values are given in Table III.

The statistical multiple-comparison tests show that for significance level $\alpha = 0.05$ we can not reject hypothesis H_0 for

TABLE II
ALGORITHM COMPARISON

	Error (Median) at FEs = 3.0e+06				Rank			
	DECC-G	MLCC	DE	DASA	DECC-G	MLCC	DE	DASA
F_1	2.86e-07	**0.00e+00**	2.24e-20	5.42e-22	4	1	3	2
F_2	1.31e+03	**6.43e-11**	3.14e+03	7.96e+00	3	1	4	2
F_3	1.39e+00	**1.46e-13**	7.94e+00	7.37e-11	3	1	4	2
F_4	1.51e+13	1.03e+13	8.21e+13	**4.94e+11**	3	2	4	1
F_5	**2.38e+08**	3.92e+08	3.09e+08	6.36e+08	1	3	2	4
F_6	4.80e+06	1.95e+07	**2.15e+01**	1.97e+07	2	3	1	4
F_7	1.07e+08	5.15e+05	**1.08e+00**	7.18e+00	4	3	1	2
F_8	6.70e+07	4.67e+07	2.91e+07	**1.21e+06**	4	3	2	1
F_9	3.18e+08	1.24e+08	6.06e+07	**3.58e+07**	4	3	2	1
F_{10}	1.07e+04	**3.16e+03**	1.15e+04	7.33e+03	3	1	4	2
F_{11}	**2.33e+01**	1.98e+02	1.34e+02	1.98e+02	1	3.5	2	3.5
F_{12}	8.87e+04	3.47e+04	7.59e+04	**1.75e+03**	4	2	3	1
F_{13}	3.00e+03	1.91e+03	1.73e+03	**9.72e+02**	4	3	2	1
F_{14}	8.07e+08	3.16e+08	1.66e+08	**1.00e+08**	4	3	2	1
F_{15}	1.18e+04	**6.89e+03**	1.30e+04	1.45e+04	2	1	3	4
F_{16}	**7.51e+01**	3.95e+02	1.87e+02	3.97e+02	1	3	2	4
F_{17}	2.89e+05	1.59e+05	5.45e+05	**1.03e+04**	3	2	4	1
F_{18}	2.30e+04	4.17e+03	4.47e+03	**3.95e+03**	4	2	3	1
F_{19}	1.11e+06	1.36e+06	2.73e+07	**8.28e+05**	2	3	4	1
F_{20}	3.98e+03	2.04e+03	2.01e+03	**1.19e+03**	4	3	2	1
				Average rank	3.000	2.325	2.700	1.975

TABLE III
ADJUSTED p-VALUES

	unadjusted	adjusted p-value			
hypothesis	p-value	Nemenyi	Holm	Shaffer	Bergmann
DECC-G vs. DASA	0.01205	0.07229	0.07229	0.07229	0.07229
DE vs. DASA	0.07575	0.45452	0.37876	0.22726	0.22726
DECC-G vs. MLCC	0.09825	0.58949	0.39299	0.29475	0.29475
MLCC vs. DE	0.35833	2.14996	1.07498	1.07498	0.35833
MLCC vs. DASA	0.39127	2.34760	1.07498	1.07498	0.78253
DECC-G vs. DE	0.46243	2.77460	1.07498	1.07498	0.78253

any of the tests, i.e., there is no significant difference in the quality of generated solutions by the compared algorithms. On the other hand, for $\alpha = 0.10$ the DASA is significantly better than DECC-G, while we can not say the same for the other comparisons.

V. CONCLUSION

This paper presented an experimental evaluation of the Differential Ant-Stigmergy Algorithm on high-dimensional numerical optimization problems. The algorithm was applied to optimization problems with continuous variables proposed for CEC'2010 Special Session and Competition on Large-Scale Global Optimization. The results show that the proposed algorithm with the proposed setting solves sparable large-scale problems with high consistency. Overall, DASA was not able to solve the nonseparable functions, but even in those cases the obtained solutions showed good consistency and were better than the ones obtained by the evolutionary algorithms DECC-G, MLCC and DE. The statistical non-parametric comparison showed that for 5% significance level there is no significant difference in the quality of generated solutions by the compared algorithms.

REFERENCES

[1] G.W. Weber, S. Ozögür-Akyüz, E. Kropat, "A review on data mining and continuous optimization applications in computational biology and medicine," *Birth Defects Research. Part C: Embryo Today*, vol. 87, pp. 165–181, 2009.

[2] F. Luna, A.J. Nebro, E. Alba, J.J. Durillo, "Solving large-scale real-world telecommunication problems using a grid-based genetic algorithm," *Engineering Optimization*, vol. 40, pp. 1067–1084, 2008.

[3] M. Dorigo and T. Stützle, *Ant Colony Optimization,* Cambridge, Massachusetts: The MIT Press, 2004.

[4] K. Socha, M. Dorigo, "Ant colony optimization for continuous domains," *European Journal of Operational Research*, vol. 185, pp. 1155–1173, 2008.

[5] P. Korošec, J. Šilc, "High-dimensional real-parameter optimization using the differential ant-stigmergy algorithm," *International Journal of Intelligent Computing and Cybernetics*, vol. 2, pp. 34–51, 2009.

[6] G. Bilchev, I.C. Parmee, "The ant colony metaphor for searching continuous design spaces", *Lecture Notes in Computer Science*, vol. 993, pp. 25–39, 1995.

[7] N. Monmarché, G. Venturini, M. Slimane, "On how pachycondyla apicalis ants suggest a new search algorithm", *Future Generation Computer Systems*, vol. 16, pp. 937–946, 2000.

[8] J. Dréo, P. Siarry, "A new ant colony algorithm using the heterarchical concept aimed at optimization of multiminima continuous functions", *Lecture Notes in Computer Science*, vol. 2463, pp. 216–221, 2002.

[9] Y.J. Li, T.J. Wu, "An adaptive ant colony system algorithm for continuous-space optimization problems", *Journal of Zhejiang University - Science A*, vol. 4, pp. 40–46, 2003.

[10] M. Kong, P. Tian, "A binary ant colony optimization for the unconstrained function optimization problem", *Lecture Notes in Computer Science*, vol. 3801, pp. 682–687, 2005.

[11] P. Korošec, J. Šilc, "Using stigmergy to solve numerical optimization problems," *Computing and Informatics*, vol. 27, pp. 377–402, 2008.

[12] K. Socha, M. Dorigo, "Ant colony optimization for continuous domains", *European Journal of Operational Research*, vol. 185, pp. 1155–1173, 2008.

[13] S.H. Pourtakdoust, H. Nobahari, "An extension of ant colony system to continuos optimization problems", *Lecture Notes in Computer Science*, vol. 3172, pp. 294-301, 2004.

[14] S. Tsutsui, "Aggregation pheromone system: A real-parameter optimization algorithm using aggregation pheromones as the base metaphor", *Transactions of the Japanese Society for Artificial Intelligence*, vol. 20, pp. 76–83, 2005.

[15] M. Kong, P. Tian, "A direct application of ant colony optimization to function optimization problem in continuous domain", *Lecture Notes in Computer Science*, vol. 4150, pp. 324–331, 2006.

[16] F.O. de França, G.P. Coelho, F.J. Von Zuben, R.R. de Faissol, "Multivariate ant colony optimization in continuous search spaces," *Proc. 10th Annual Conference on Genetic and Evolutionary Computation*, Atlanta, Georgia, USA, July 2008, pp. 9–16.

[17] K. Socha, C. Blum, "An ant colony optimization algorithm for continuous optimization: application to feed-forward neural network training", *Neural Computing and Applications*, vol. 16, pp. 235–247, 2007.

[18] Y. Ge, Q.C. Meng, C.J. Yan, J. Xu, "A hybrid ant colony algorithm for global optimization of continuous multi-extreme functions", *Proc. 3rd International Conference on Machine Learning and Cybernetics*, Shanghai, China, August 2004 pp. 2427–2432.

[19] C. Chen, X.Y. Tian, X.Y. Zou, P.X. Cai, J.Y. Mo, "A hybrid ant colony optimization for the prediction of protein secondary structure", *Chinese Chemical Letters*, vol. 16, pp. 1551–1554, 2005.

[20] S.L. Ho, S. Yang, G. Ni, J.M. Machado, "A modified ant colony optimization algorithm modeled on tabu-search methods", *IEEE Transaction on Magnetics*, vol. 42, pp. 1195–1198, 2006.

[21] W. Gao, "Immunized continuous ant colony algorithm", *Proc. 26th Chinese Control Conference*, Zhangjiajie, China, July 2007, pp. 705–709.

[22] X.M. Hu, J. Zhang, Y. Li, "Orthogonal methods based ant colony search for solving continuous optimization problems", *Journal of Computer Science and Technology*, vol. 23, pp. 2–18, 2008.

[23] K. Tang, X. Li, P.N. Suganthan, Z. Yang, T. Weise, "Benchmark Functions for the CEC'2010 Special Session and Competition on Large Scale Global Optimization," Technical Report, Nature Inspired Computation and Applications Laboratory, USTC, China, 2010.

[24] P. Korošec, J. Šilc, K. Oblak, F. Kosel, "The differential ant-stigmergy algorithm: An experimental evaluation and a real-world application," *Proc. IEEE Congress on Evolutionary Computation*, Singapore, September 2007, pp. 157–164.

[25] Z. Yang, K. Tang, X. Yao, "Large Scale Evolutionary Optimization Using Cooperative Coevolution," *Information Sciences*, vol. 178, pp. 2985–2999, 2008.

[26] Z. Yang, K. Tang, X. Yao, "Multilevel Cooperative Coevolution for Large Scale Optimization," *Proc. IEEE Congress on Evolutionary Computation*, Hongkong, China, July 2008, pp. 1663–1670.

[27] R. Storn, K. Price, "Differential evolution – A Simple and efficient heuristic for global optimization over continuous spaces", *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[28] S. García, F. Herrera, "An extension on Statistical comparisons of classifiers over multiple data sets for all pairwise comparisons", *Journal of Machine Learning Research*, vol. 9, pp. 2677–2694, 2008.