# The Differential Ant-Stigmergy Algorithm Applied to Dynamic Optimization Problems

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

***Abstract*— Many real-world problems are dynamic, requiring an optimization algorithm which is able to continuously track a changing optimum over time. In this paper, we present a stigmergy-based algorithm for solving optimization problems with continuous variables, labeled Differential Ant-Stigmergy Algorithm (DASA). The DASA is applied to dynamic optimization problems without any modification to the algorithm. The performance of the DASA is evaluated on the set of benchmark problems provided for CEC'2009 Special Session on Evolutionary Computation in Dynamic and Uncertain Environments.**

## I. Introduction

Optimization problems whose optimal solution changes over time during the optimization are an important issue in many areas of human activities. Such dynamic optimization problems (DOPs) can be defined as follows:

$$F = f(x, \phi, t),$$

where $F$ is the optimization problem, $f$ is the cost function, $x$ is a feasible solution in the solution set $\mathbf{X}$, $t$ is time, and $\phi$ is the system control parameter, which determines the solution distribution in the fitness landscape.

In recent years Ant Colony Optimization (ACO) algorithms have been applied to more challenging and complex problem domains. One such domain are DOPs. Here, the literature shows an increasing interest for applying ACO (e.g., population based ACO (P-ACO) [1], [2], ant systems for a dynamic TSP (AS-DTSP) [3], binary ant algorithm (BAA) [4], dynamic hybrid continuous interacting ant colony (DHCIAC) [5]).

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'2009 Special Session on Evolutionary Computation in Dynamic and Uncertain Environments. Finally, Section IV concludes the paper.

## II. Stigmergy-based 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:

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

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

Here, $\delta_i$ 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, $\delta_i$, 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 $U_i = \lfloor \lg_b(\max(x_i) - \min(x_i)) \rfloor$. With the parameter $\epsilon_i$, the maximum precision of the parameter $x_i$ is set. The precision is limited by the computer's floating-point arithmetics. To enable a more flexible movement over the search space, the weight $\omega$ 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 $\Delta_i$, $1 \leq i \leq D$, 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 $\Delta_i$ 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 $\nu = (v_1 v_2 \ldots v_i \ldots v_D)$, where $v_i \in V_i$, $1 \leq i \leq D$.

978-1-4244-2959-2/09/$25.00 © 

### C. Algorithm Implementation

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

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

$$C(z) = \frac{1}{s\pi(1 + (\frac{z-l_i}{s})^2)},$$

where $l_i$ is the location offset for the $i$-th parameter and

$$s = s_{\mathrm{global}} - s_{\mathrm{local}}$$

is the scale factor. For an initial pheromone distribution the Cauchy distribution with $s_{\mathrm{global}} = 10$, $s_{\mathrm{local}} = 0$, and $l_i = 0, i = 1, 2, \ldots, D$ is used and each parameter vertices are equidistantly arranged between $z = [-4, 4]$.

There are $m$ ants in a colony, all of which begin 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:

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

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 $m^2$ for all ants) we get $\vec{p_i} = \mathbf{0}$ the search process is reset by randomly choosing new $\vec{x}^{\,\mathrm{tb}}$ 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 ($\vec{x}^{\,\mathrm{cb}}$, $\vec{p}^{\,\mathrm{cb}}$, and $y_i^{\mathrm{cb}}$).

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

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

$s_{\mathrm{local}}$ is set to

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

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

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

Pheromone evaporation is defined by some predetermined percentage $\rho$. The probability density function $C(z)$ is changed in the following way:

$$l_i \leftarrow (1 - \rho)l_i$$

and

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

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.

Figure 1 summarizes the DASA.

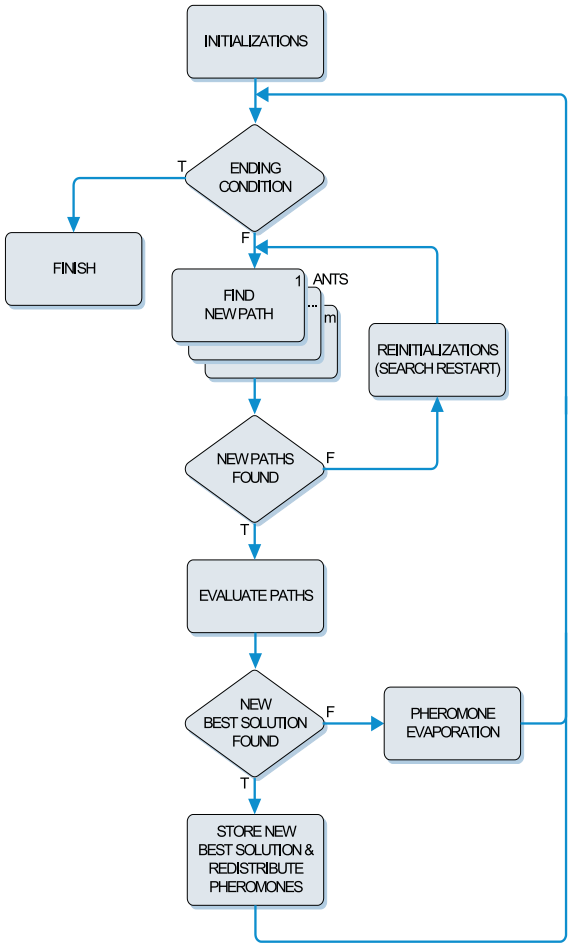


Fig. 1. The Differential Ant-Stigmergy Algorithm.

## III. Performance Evaluation

### A. The Experimental Environment

The computer platform used to perform the experiments was based on AMD Opteron™ 2.6-GHz processor, 2 GB of RAM, and the Microsoft® Windows® operating system. The DASA was implemented in Borland® Delphi™.

TABLE I

ERROR VALUES ACHIEVED FOR PROBLEM $F_1$

| Dimension($n$) | Peaks($m$) | Errors | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ |
|---|---|---|---|---|---|---|---|---|
| 10 | 10 | Avg_best | 4.17 E−13 | 3.80 E−13 | 3.80 E−13 | 6.57 E−13 | 5.56 E−13 | 7.90 E−13 |
| | | Avg_worst | 5.51 E+00 | 3.85 E+01 | 3.97 E+01 | 9.17 E+00 | 2.09 E+01 | 4.71 E+01 |
| | | Avg_mean | 1.80 E−01 | 4.18 E+00 | 6.37 E+00 | 4.82 E−01 | 2.54 E+00 | 2.34 E+00 |
| | | STD | 1.25 E+00 | 9.07 E+00 | 1.07 E+01 | 1.95 E+00 | 4.80 E+00 | 8.66 E+00 |
| | 50 | Avg_best | 5.97 E−13 | 5.03 E−13 | 3.57 E−13 | 7.73 E−13 | 8.02 E−13 | 6.73 E−13 |
| | | Avg_worst | 7.67 E+00 | 2.91 E+01 | 3.10 E+01 | 5.58 E+00 | 1.16 E+01 | 3.51 E+01 |
| | | Avg_mean | 4.42 E−01 | 4.86 E+00 | 8.42 E+00 | 5.09 E−01 | 1.18 E+00 | 2.07 E+00 |
| | | STD | 1.39 E+00 | 7.00 E+00 | 9.56 E+00 | 1.09 E+00 | 2.18 E+00 | 5.97 E+00 |
| $T_7$(5-15) | 10 | Avg_best | — | — | 3.55 E−14 | — | — | — |
| | | Avg_worst | — | — | 2.91 E+01 | — | — | — |
| | | Avg_mean | — | — | 4.84 E+00 | — | — | — |
| | | STD | — | — | 8.96 E+00 | — | — | — |
| | 50 | Avg_best | — | — | 7.39 E−14 | — | — | — |
| | | Avg_worst | — | — | 3.22 E+01 | — | — | — |
| | | Avg_mean | — | — | 7.84 E+00 | — | — | — |
| | | STD | — | — | 9.05 E+00 | — | — | — |

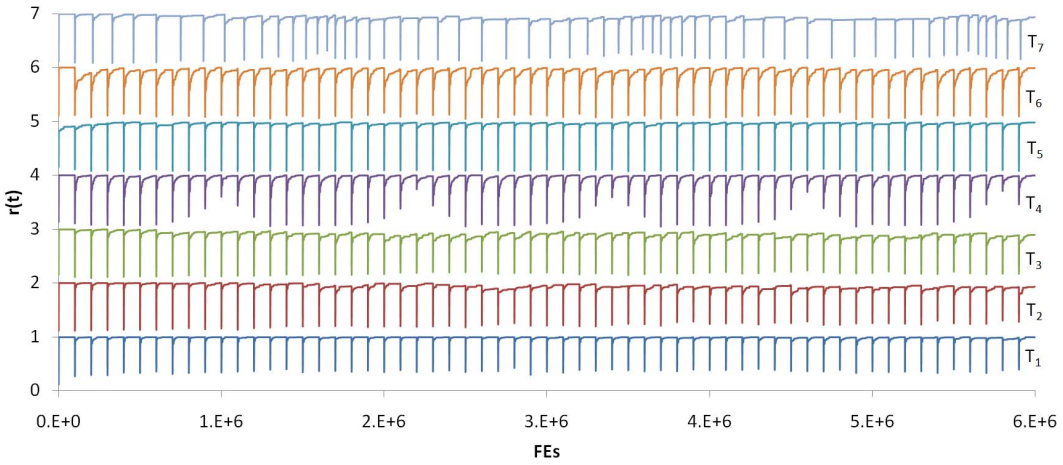


Fig. 2. Convergence graph for $F_1$ ($m = 10$).

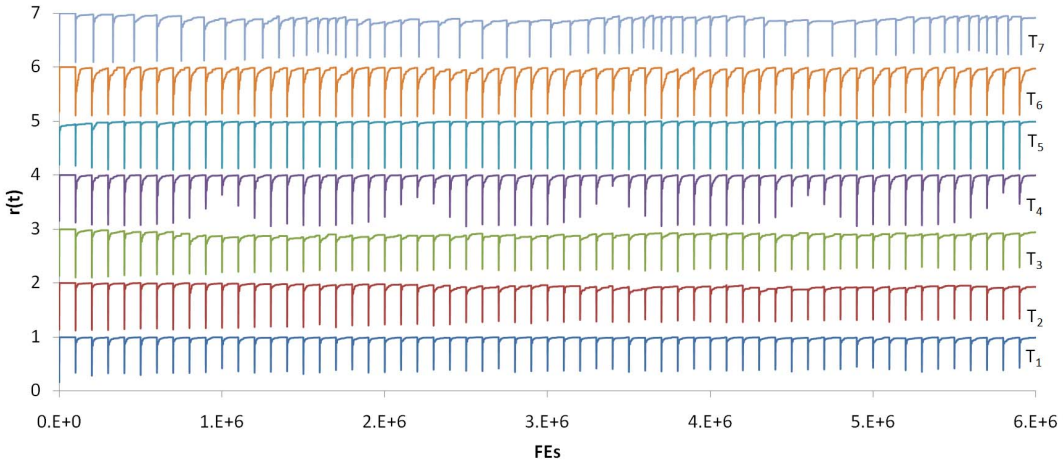


Fig. 3. Convergence graph for $F_1$ ($m = 50$).

TABLE II

ERROR VALUES ACHIEVED FOR PROBLEM $F_2$

| Dimension($n$) | Errors | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ |
|---|---|---|---|---|---|---|---|
| 10 | Avg_best | 1.97E−11 | 2.34E−11 | 2.72E−11 | 1.41E−11 | 3.59E−11 | 1.65E−11 |
| | Avg_worst | 3.39E+01 | 4.03E+02 | 3.56E+02 | 1.65E+01 | 4.33E+02 | 2.49E+01 |
| | Avg_mean | 3.30E+00 | 2.56E+01 | 1.89E+01 | 1.45E+00 | 4.96E+01 | 2.11E+00 |
| | STD | 8.78E+00 | 8.32E+01 | 6.78E+01 | 3.83E+00 | 1.12E+02 | 5.29E+00 |
| $T_7$(5-15) | Avg_best | — | — | 1.30E−12 | — | — | — |
| | Avg_worst | — | — | 3.67E+01 | — | — | — |
| | Avg_mean | — | — | 3.87E+00 | — | — | — |
| | STD | — | — | 8.12E+00 | — | — | — |

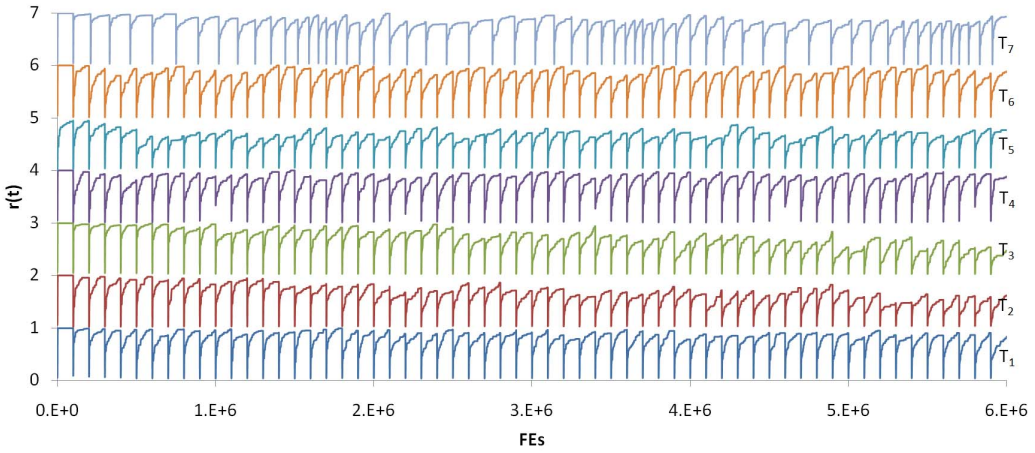


Fig. 4. Convergence graph for $F_2$.

TABLE III

ERROR VALUES ACHIEVED FOR PROBLEM $F_3$

| Dimension($n$) | Errors | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ |
|---|---|---|---|---|---|---|---|
| 10 | Avg_best | 3.39E−11 | 4.34E+01 | 1.38E+00 | 4.51E−11 | 3.08E+00 | 4.21E−11 |
| | Avg_worst | 4.35E+02 | 9.88E+02 | 9.37E+02 | 1.17E+03 | 9.23E+02 | 1.47E+03 |
| | Avg_mean | 1.57E+01 | 8.24E+02 | 6.88E+02 | 4.35E+02 | 6.97E+02 | 6.26E+02 |
| | STD | 6.71E+01 | 2.04E+02 | 2.98E+02 | 4.41E+02 | 3.15E+02 | 4.60E+02 |
| $T_7$(5-15) | Avg_best | — | — | 1.06E−01 | — | — | — |
| | Avg_worst | — | — | 9.09E+02 | — | — | — |
| | Avg_mean | — | — | 4.33E+02 | — | — | — |
| | STD | — | — | 3.80E+02 | — | — | — |

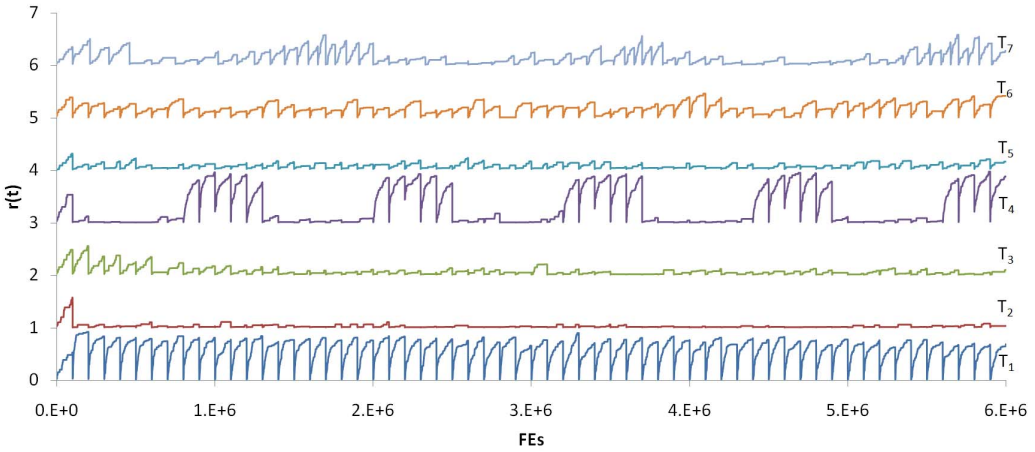


Fig. 5. Convergence graph for $F_3$.

TABLE IV

ERROR VALUES ACHIEVED FOR PROBLEM $F_4$

| Dimension($n$) | Errors | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ |
|---|---|---|---|---|---|---|---|
| 10 | Avg_best | 2.01 E−11 | 2.95 E−11 | 2.87 E−11 | 1.85 E−11 | 5.89 E−11 | 2.09 E−11 |
| | Avg_worst | 5.76 E+01 | 5.05 E+02 | 5.40 E+02 | 1.88 E+01 | 5.28 E+02 | 3.97 E+01 |
| | Avg_mean | 5.60 E+00 | 6.56 E+01 | 5.36 E+01 | 1.85 E+00 | 1.08 E+02 | 2.98 E+00 |
| | STD | 2.65 E+01 | 1.60 E+02 | 1.40 E+02 | 4.22 E+00 | 1.78 E+02 | 7.59 E+00 |
| $T_7$(5-15) | Avg_best | — | — | 7.10 E−12 | — | — | — |
| | Avg_worst | — | — | 4.51 E+02 | — | — | — |
| | Avg_mean | — | — | 2.74 E+01 | — | — | — |
| | STD | — | — | 9.00 E+01 | — | — | — |

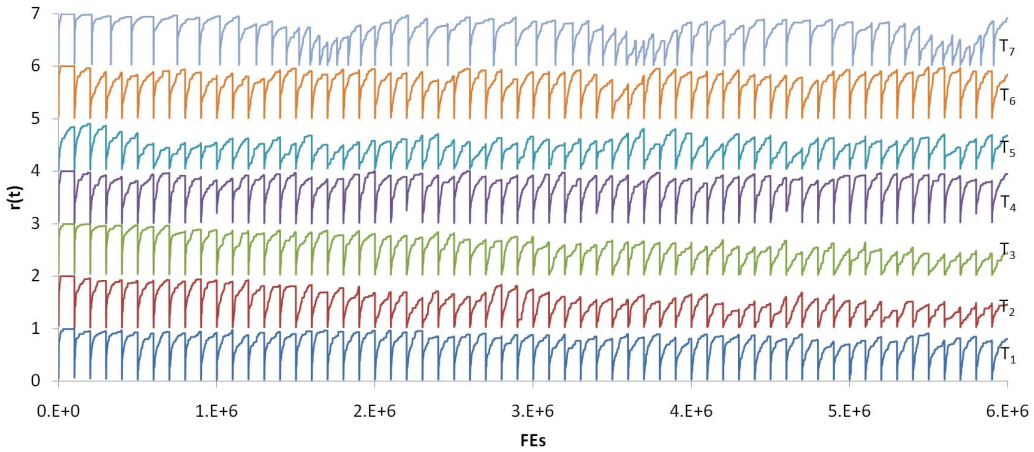


Fig. 6. Convergence graph for $F_4$.

TABLE V

ERROR VALUES ACHIEVED FOR PROBLEM $F_5$

| Dimension($n$) | Errors | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ |
|---|---|---|---|---|---|---|---|
| 10 | Avg_best | 3.22 E−11 | 3.74 E−11 | 3.86 E−11 | 2.69 E−11 | 5.99 E−11 | 2.85 E−11 |
| | Avg_worst | 1.71 E+01 | 2.22 E+01 | 1.60 E+01 | 8.10 E+00 | 2.90 E+01 | 8.75 E+00 |
| | Avg_mean | 9.55 E−01 | 9.90 E−01 | 9.49 E−01 | 3.92 E−01 | 2.30 E+00 | 4.67 E−01 |
| | STD | 3.43 E+00 | 4.05 E+00 | 3.31 E+00 | 1.61 E+00 | 6.36 E+00 | 1.73 E+00 |
| $T_7$(5-15) | Avg_best | — | — | 1.93 E−12 | — | — | — |
| | Avg_worst | — | — | 1.87 E+01 | — | — | — |
| | Avg_mean | — | — | 1.11 E+00 | — | — | — |
| | STD | — | — | 3.76 E+00 | — | — | — |

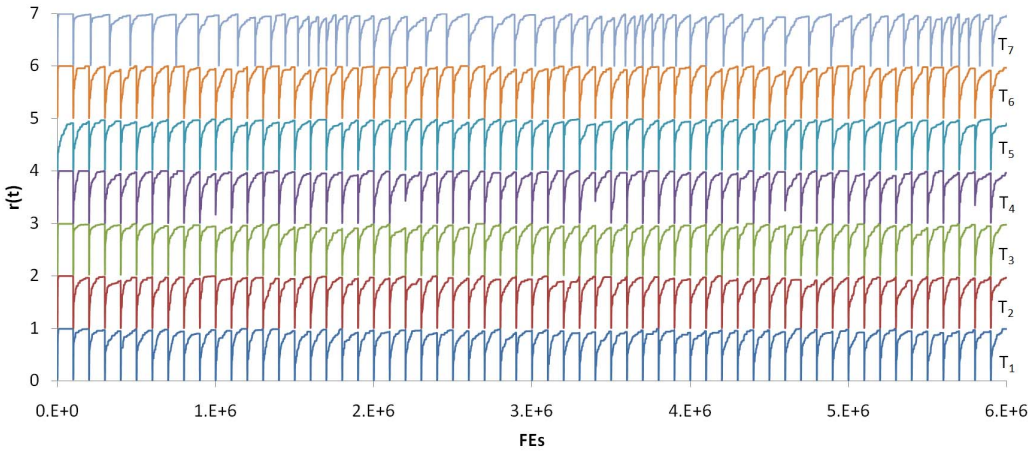


Fig. 7. Convergence graph for $F_5$.

TABLE VI
ERROR VALUES ACHIEVED FOR PROBLEM $F_6$

| Dimension($n$) | Errors | $T_1$ | $T_2$ | $T_3$ | $T_4$ | $T_5$ | $T_6$ |
|---|---|---|---|---|---|---|---|
| 10 | Avg_best | 2.36 E−11 | 3.58 E−11 | 3.69 E−11 | 2.55 E−11 | 6.37 E−11 | 2.56 E−11 |
| | Avg_worst | 4.83 E+01 | 5.54 E+02 | 5.29 E+02 | 8.16 E+01 | 4.99 E+02 | 2.49 E+02 |
| | Avg_mean | 8.87 E+00 | 3.70 E+01 | 2.67 E+01 | 9.74 E+00 | 3.79 E+01 | 1.33 E+01 |
| | STD | 1.33 E+01 | 1.22 E+02 | 9.84 E+01 | 2.20 E+01 | 1.18 E+02 | 5.74 E+01 |
| $T_7$(5-15) | Avg_best | — | — | 6.48 E−12 | — | — | — |
| | Avg_worst | — | — | 1.37 E+02 | — | — | — |
| | Avg_mean | — | — | 1.17 E+01 | — | — | — |
| | STD | — | — | 3.67 E+01 | — | — | — |

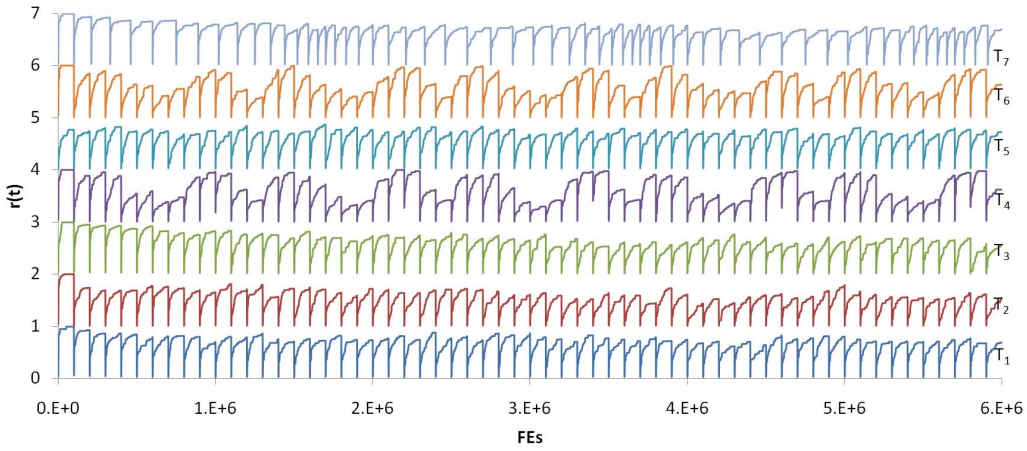


Fig. 8. Convergence graph for $F_6$.

### B. The Benchmark Suite

The DASA algorithm was tested on six benchmark problems provided for the CEC'2009 Special Session on Evolutionary Computation in Dynamic and Uncertain Environments [7]:

- $F_1$: Rotation peak function (multi-modal, scalable, rotated, the number of local optima are artificially controlled),
- $F_2$: Composition of Sphere's function (multi-modal, scalable, rotated, 10 local optima),
- $F_3$: Composition of Rastrigin's function (multi-modal, scalable, rotated, a huge number of local optima),
- $F_4$: Composition of Griewank's function (multi-modal, scalable, rotated, a huge number of local optima),
- $F_5$: Composition of Ackley's function (multi-modal, scalable, rotated, a huge number of local optima),
- $F_6$: Hybrid composition function (multi-modal, scalable, rotated, a huge number of local optima, different functions properties are mixed together, sphere functions give two flat areas for the function).

Framework of dynamic changes:

- $T_1$: small step change,
- $T_2$: large step change,
- $T_3$: random change,
- $T_4$: chaotic change,
- $T_5$: recurrent change,
- $T_6$: recurrent change with noise,
- $T_7$: random change with changed dimension.

### C. Parameter Settings

The DASA has six parameters: the number of ants, $m$, the pheromone evaporation factor, $\rho$, the maximum parameter precision, $\epsilon$, the discrete base, $b$, the global scale increase factor, $s_+$, and the global scale decrease factor, $s_-$. With respect to usual setting [6], we have made some adjustments according to problems' demands. We have decreased the number of ants to $m = 3$ for better algorithms responsiveness and with it decreased the $\rho = 0.1$ to reduce the convergence speed which was increased with smaller number of ants. The rest of the algorithm parameters are set as usual, the algorithms accuracy $\epsilon = 10^{-15}$, discrete base $b = 10$, global scale increase factor $s_+ = 0.01$, and global scale decrease factor, $s_- = 0.02$. We must note that during the experimentation we did not fine-tune the algorithms parameters, but only make a limited number of experiments to find satisfying settings.

### D. Testing procedure

For all test problems we used the following testing procedure. From the DASA's point of view, the algorithm was run for required number of evaluations and no information, other than problem evaluation value, $y = f(\vec{x})$, was given to the

algorithm. So no information regarding any changes in the problem (dynamic or dimension changes) were transferred during the algorithm run. From implementation perspective, an interface (DOP manager) was created which performed all the required changes, dynamic or dimensional. For example, for change type $T_7$, we set algorithm's number of evaluations to $6 \cdot 10^6$ and dimension to 15 (which was maximal dimension). Therefore, the DASA treated the problem as "static" 15-dimensional problem, while the DOP manager took care of all the dimensional changes.

### *E. Results*

In Tables I–VI we present the error values achieved by the DASA algorithm on all test problems.

For each change type of each function, we present average best (Avg_best), average mean (Avg_mean), average worst (Avg_worst) values and standard deviation (STD) for $x^{\mathrm{b}}(t)$ over 20 runs:

$$\text{Avg\_best} = \sum_{i=1}^{\text{runs}} \min_{j=1}^{\text{num\_change}} \frac{E_{i,j}^{\text{last}}(t)}{\text{runs}},$$

$$\text{Avg\_mean} = \sum_{i=1}^{\text{runs}} \sum_{j=1}^{\text{num\_change}} \frac{E_{i,j}^{\text{last}}(t)}{\text{runs} * \text{num\_change}},$$

$$\text{Avg\_worst} = \sum_{i=1}^{\text{runs}} \max_{j=1}^{\text{num\_change}} \frac{E_{i,j}^{\text{last}}(t)}{\text{runs}}.$$

Here, $E^{\text{last}}(t) = |f(x^{\mathrm{b}}(t)) - f(x^*(t))|$ after reaching Max_FES/change for each change.

Figures 2–8 present the convergence graphs for each problem for dimension $n = 10$. Each graph shows the median performance of the relative value $r(t)$ of $f(x^{\mathrm{b}}(t))$ and $f(x^*(t))$ for total runs with termination by the Total_FES. For maximization function $F_1$,

$$r(t) = \frac{f(x^{\mathrm{b}}(t))}{f(x^*(t))},$$

for minimization functions $F_2 - F_6$,

$$r(t) = \frac{f(x^*(t))}{f(x^{\mathrm{b}}(t))}.$$

Note that relative values depicted in the figures are presented for each $T_i$ as $r(t) + i - 1$, where $0 \leq r(t) \leq 1$, $i = 1, 2, \ldots, 7$.

Table VII presents the performance of the DASA algorithm. The $\text{Mark}_{\text{max}}$ presents the maximal mark that can be obtained by the algorithm. The sum of all $\text{Mark}_{\text{max}}$ values is 100. The mark of each problem/change_type (pct) combination is calculated by:

$$\text{mark}_{\text{pct}} = \text{percentage}_{\text{pct}} * \sum_{i=1}^{\text{runs}} \sum_{j=1}^{\text{num\_change}} \frac{r_{ij}}{\text{num\_change} * \text{runs}}$$

TABLE VII
PERFORMANCE MEASUREMENT

| Function | Dimension | Peaks | Change type | $\text{Mark}_{\text{max}}$ | $\text{mark}_{\text{pct}}$ |
|---|---|---|---|---|---|
| $F_1$ | 10 | 10 | $T_1$ | 1.5 | 1.471 |
| $F_1$ | 10 | 50 | $T_1$ | 1.5 | 1.455 |
| $F_1$ | 10 | 10 | $T_2$ | 1.5 | 1.357 |
| $F_1$ | 10 | 50 | $T_2$ | 1.5 | 1.339 |
| $F_1$ | 10 | 10 | $T_3$ | 1.5 | 1.280 |
| $F_1$ | 10 | 50 | $T_3$ | 1.5 | 1.241 |
| $F_1$ | 10 | 10 | $T_4$ | 1.5 | 1.416 |
| $F_1$ | 10 | 50 | $T_4$ | 1.5 | 1.423 |
| $F_1$ | 10 | 10 | $T_5$ | 1.5 | 1.396 |
| $F_1$ | 10 | 50 | $T_5$ | 1.5 | 1.438 |
| $F_1$ | 10 | 10 | $T_6$ | 1.5 | 1.355 |
| $F_1$ | 10 | 50 | $T_6$ | 1.5 | 1.346 |
| $F_1$ | 5-15 | 10 | $T_7$ | 1.0 | 0.885 |
| $F_1$ | 5-15 | 50 | $T_7$ | 1.0 | 0.832 |
| Sum for $F_1$ | | | | 20.0 | 18.24 |
| $F_2$ | 10 | — | $T_1$ | 2.4 | 1.865 |
| $F_2$ | 10 | — | $T_2$ | 2.4 | 1.446 |
| $F_2$ | 10 | — | $T_3$ | 2.4 | 1.583 |
| $F_2$ | 10 | — | $T_4$ | 2.4 | 1.890 |
| $F_2$ | 10 | — | $T_5$ | 2.4 | 1.420 |
| $F_2$ | 10 | — | $T_6$ | 2.4 | 1.826 |
| $F_2$ | 5-15 | — | $T_7$ | 1.6 | 1.215 |
| Sum for $F_2$ | | | | 16.0 | 11.24 |
| $F_3$ | 10 | — | $T_1$ | 2.4 | 1.413 |
| $F_3$ | 10 | — | $T_2$ | 2.4 | 0.072 |
| $F_3$ | 10 | — | $T_3$ | 2.4 | 0.174 |
| $F_3$ | 10 | — | $T_4$ | 2.4 | 0.742 |
| $F_3$ | 10 | — | $T_5$ | 2.4 | 0.223 |
| $F_3$ | 10 | — | $T_6$ | 2.4 | 0.455 |
| $F_3$ | 5-15 | — | $T_7$ | 1.6 | 0.282 |
| Sum for $F_3$ | | | | 16.0 | 3.36 |
| $F_4$ | 10 | — | $T_1$ | 2.4 | 1.759 |
| $F_4$ | 10 | — | $T_2$ | 2.4 | 1.233 |
| $F_4$ | 10 | — | $T_3$ | 2.4 | 1.327 |
| $F_4$ | 10 | — | $T_4$ | 2.4 | 1.788 |
| $F_4$ | 10 | — | $T_5$ | 2.4 | 1.091 |
| $F_4$ | 10 | — | $T_6$ | 2.4 | 1.699 |
| $F_4$ | 5-15 | — | $T_7$ | 1.6 | 1.005 |
| Sum for $F_4$ | | | | 16.0 | 9.90 |
| $F_5$ | 10 | — | $T_1$ | 2.4 | 2.021 |
| $F_5$ | 10 | — | $T_2$ | 2.4 | 2.012 |
| $F_5$ | 10 | — | $T_3$ | 2.4 | 2.030 |
| $F_5$ | 10 | — | $T_4$ | 2.4 | 2.049 |
| $F_5$ | 10 | — | $T_5$ | 2.4 | 2.019 |
| $F_5$ | 10 | — | $T_6$ | 2.4 | 2.024 |
| $F_5$ | 5-15 | — | $T_7$ | 1.6 | 1.346 |
| Sum for $F_5$ | | | | 16.0 | 13.50 |
| $F_6$ | 10 | — | $T_1$ | 2.4 | 1.478 |
| $F_6$ | 10 | — | $T_2$ | 2.4 | 1.154 |
| $F_6$ | 10 | — | $T_3$ | 2.4 | 1.335 |
| $F_6$ | 10 | — | $T_4$ | 2.4 | 1.337 |
| $F_6$ | 10 | — | $T_5$ | 2.4 | 1.367 |
| $F_6$ | 10 | — | $T_6$ | 2.4 | 1.318 |
| $F_6$ | 5-15 | — | $T_7$ | 1.6 | 0.970 |
| Sum for $F_6$ | | | | 16.0 | 8.96 |
| **Overall performance** | | | | 100 | **65.21** |

and

$$r_{ij} = \frac{r_{ij}^{\text{last}}}{1 + \sum_{s=1}^{S} \frac{1 - r_{ij}^{s}}{S}},$$

$r_{ij}^{\text{last}}$ is the relative value of the best one to the global optimum after reaching Max_FES/change for each change. $r_{ij}^{s}$ is the relative value of the best one to the global optimum at the $s$-th sampling during one change and

$$S = \frac{\text{Max\_FES/change}}{s\_f},$$

where $s_f$ is sampling frequency. In our case we have: $s_f = 100$, Max_FES/change $= 10{,}000 * n$, and $n = 10$. The percentage$_{\text{pct}}$ is defined in [7].

The overall algorithm performance is evaluated by:

$$\text{performance} = \sum_{\text{pct}=1}^{\text{Num\_test\_cases}} \text{mark}_{\text{pct}}.$$

There are totally Num_test_cases $= 49$ specific test cases.

## IV. Conclusion

This paper presented a stigmergy-based algorithm developed for numerical optimization problems. The algorithm was applied to dynamic optimization problems with continuous variables proposed for CEC'2009 Special Session on Evolutionary Computation in Dynamic and Uncertain Environments. The results showed that the proposed algorithm can find reasonable solutions for all of the problems. Only the $F_3$ problem was the one, that the DASA could not find the optimum quick enough during dynamic changes.

One obvious advantage is that was no need any changes to the original algorithm. So, it can be used as such for both cases of numerical optimization, static and dynamic. Furthermore, the algorithm is unsusceptible to different types of changes and can be used with very limited knowledge about problem, only maximal dimension and input problem parameters.

## References


[1] M. Guntsch and M. Middendorf, "A population based approach for ACO," *Lecture Notes in Coputer Science*, vol. 2279, pp. 72–81, 2002.

[2] M. Guntsch and M. Middendorf, "Applying population based ACO to dynamic optimization problems," *Lecture Notes in Coputer Science*, vol. 2463, pp. 111–122, 2002.

[3] C. J. Eyckelhof and M. Snoek, "Ant systems for a dynamic TSP," *Lecture Notes in Coputer Science*, vol. 2463, pp. 88–99, 2002.

[4] C. Fernandes, V. Ramos, and A. C. Rosa, "Stigmergic optimization in dynamic binary landscapes," *Proc. 22nd Annual ACM Symposium on Applied Computing*, Seoul, Korea, March 2007, pp. 747–748.

[5] W. Tfaili, J. Dréo, and P. Siarry, "Fitting of an ant colony approach to dynamic optimization through a new set of test functions," *International Journal of Computational Intelligence Research*, vol. 3, pp. 203–216, 2007.

[6] P. Korošec, J. Šilc, K. Oblak, and 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.

[7] C. Li, S. Yang, T. T. Nguyen, E. L. Yu, X. Yao, Y. Jin, H.-C. Beyer, and P. N. Suganthan, *Benchmark Generator for CEC'2009 Competition on Dynamic Optimization*, September 15, 2008. `http://www.cs.le.ac.uk/people/syang/ECiDUE/`


## Appendix
## Pseudocode of the DASA

1: $\vec{x}^{\,\text{tb}}$ = `Rnd_Solution()`
2: $y^{\,\text{b}} = f(\vec{x}^{\,\text{tb}})$
3: $y^{\,\text{tb}} = \inf$
4: $\mathcal{G}$ = `Graph_Initialization(`$\vec{x}^{\,\text{tb}}$, $\vec{\epsilon}$`)`
5: `Pheromone_Initialization(`$\mathcal{G}$`)`
6: **while** not ending condition met **do**
7: $k = 0$
8: **for all** $m$ ants **do**
9: **repeat**
10: $\vec{p_i}$ = `Find_Path(`$\mathcal{G}$`)`
11: $k = k + 1$
12: **if** $k > m^2$ **then**
13: $\vec{x}^{\,\text{tb}}$ = `Rnd_Solution()`
14: `Pheromone_Initialization(`$\mathcal{G}$`)`
15: **goto** line 7
16: **end if**
17: **until** ($\vec{p_i}$ = **0**)
18: $\omega = $ `Random_Integer(`$1, b-1$`)`
19: $\vec{x_i} = \vec{x}^{\,\text{tb}} + \omega\delta(\vec{p})$
20: **end for**
21: $y^{\,\text{cb}} = \inf$
22: **for all** $m$ ants **do**
23: $y = f(\vec{x_i})$
24: **if** $y < y^{\,\text{cb}}$ **then**
25: $y^{\,\text{cb}} = y$
26: $\vec{p}^{\,\text{cb}} = \vec{p_i}$
27: $\vec{x}^{\,\text{cb}} = \vec{x_i}$
28: **end if**
29: **end for**
30: **if** $y^{\,\text{cb}} < y^{\,\text{tb}}$ **then**
31: $y^{\,\text{tb}} = y^{\,\text{cb}}$
32: $\vec{x}^{\,\text{tb}} = \vec{x}^{\,\text{cb}}$
33: $s = $ `Update_Scales(`$s_{\text{global}}, s_{\text{local}}$`)`
34: `Pheromone_Redistribution(`$\vec{p}^{\,\text{cb}}, s$`)`
35: **if** $y^{\,\text{tb}} < y^{\,\text{b}}$ **then**
36: $y^{\,\text{b}} = y^{\,\text{tb}}$
37: $\vec{x}^{\,\text{b}} = \vec{x}^{\,\text{tb}}$
38: **end if**
39: **else**
40: `Update_Scale(`$s_{\text{global}}$`)`
41: **end if**
42: `Pheromone_Evaporation(`$\mathcal{G}$, $\rho$`)`
43: **end while**