FACTA UNIVERSITATIS (NIŠ)
SER.: ELEC. ENERG. vol. 19, no. 2, August 2006, 247-260

The Multilevel Ant Stigmergy Algorithm for Numerical Optimization

Peter Korošec and Jurij Šilc

Abstract: The Multilevel Ant Stigmergy Algorithm (MASA) is a new approach to solving multi-parameter problems based on stigmergy, a type of collective work that can be observed in nature. In this paper we evaluate the performance of MASA regarding its applicability as numerical optimization techniques. The evaluation is performed with several widely used benchmarks functions, as well as on an industrial case study. We also compare the MASA with Differential Evolution, well-known numerical optimization algorithm. The average solution obtained with the MASA was better than a solution recently found using Differential Evolution.

Keywords: Ant-based algorithm, multilevel approach, numerical optimization, stigmergy.

1 Introduction

Numerical optimization is important in decision science and in the analysis of physical systems. An important step in optimization is the identification of some objective, i.e., a quantitative measure of the performance of the system. This objective can be any quantity or combination of quantities that can be represented by a single number. The objective depends on certain characteristics of the system called parameters, which are often restricted or constrained in some way. Furthermore, the parameters can have either continuous or discrete values. Our goal is to find values of the parameters that optimize the objective. Depending on the type of parameters, we distinguish between *continuous* optimization [1] and *discrete* optimization [2].

There is no universal optimization algorithm to solve such optimization problems. Many of the problems arising in real-life applications are NP-hard. Hence,

Manuscript received April 10, 2006. An earlier version of this paper was presented at the 7th International Conference on Artificial Evolution EA'05, October 26-28, 2005, Lille, France.

The authors are with Jožef Stefan Institute, Computer Systems Department, Jamova 19, SI-1000 Ljubljana, Slovenia. (e-mail: [peter.korosec, jurij.silc]@ijs.si).

one usually solves large instances with the use of approximate methods that return near-optimal solutions in a relatively short time. Algorithms of this type are called *heuristics*. The upgrade of a heuristic is a *metaheuristic* [3]: a set of algorithmic concepts that can be used to define a heuristic method applicable to a wider set of different problems. A particularly successful metaheuristic is based on stigmergy.

Stigmergy is a method of communication in decentralized systems in which the individual parts of the system communicate with one another by modifying their local environment [4]. It provides a new paradigm for developing decentralized complex applications such as autonomous and collective robotics [5], communication in computer networks [6], multi-agent systems [7], optimization algorithms, etc.

In this paper, we introduce a new stigmergy-based approach inspired by colony of real ants. Ants communicate with one another by laying down pheromones along their trails, so an ant colony is a stigmergic system. Ant colony metaheuristic is usually used for solving (discrete) combinatorial optimization problems [8], but we will show a successful implementation on numerical multi-parameter optimization problem which is often solved by algorithms for continuous optimization.

Multi-parameter optimization is the process of finding the point in the parameter space $P = \{p_1, p_2, \ldots, p_D\}$ where a cost function $f(P)$ is minimized according to the feasible set Ω of parameters p_i, $i = 1, 2, \ldots, D$, that satisfy the constraints. Very often this cost function contains information about the problem target and the constraints that the solution has to meet (constrained optimization). These constraints define the region of the design space where the solution has to be comprised – called the feasibility region. Optimizing a multi-parameter function is usually a continuous problem.

2 Multilevel Ant Stigmergy Algorithm

The *Multilevel Ant Stigmergy Algorithm* (MASA) [9] is a new approach to solving multi-parameter optimization problems. It is based on stigmergy, a type of collective work that can be observed in ant colonies. MASA consists of five phases (see Fig. 1):

1. *Search graph construction*. We first had to discretize the continuous, multi-parameter problem by discretizing all the parameters of the multi-parameter function. For example, if a parameter p_i has a range from L_i to U_i and the discrete step is Δ then a discrete parameter p_i has $\lceil \frac{U_i - L_i}{\Delta} \rceil + 1$ discrete values. A search graph is defined as a connected, directed, non-weighted, acyclic graph. It is also rooted and ordered. So what we do is translate all the discrete parameter values into a search graph. For this purpose we define a search

Multilevel Ant Stigmergy Algorithm

1. Construct the search graph from all parameters.
2. Coarsen the graph in L levels.
3. Initialize vertices with initial amount of pheromone.
4. For all levels ℓ from L down to 1, do:
   4.1. While current level ℓ stopping criterion not met, do:
      (a) For all ants find (using probability rule) the cheapest path.
      (b) Update pheromone amounts in all vertices visited by the ants.
      (c) Additionally increase the pheromone amounts on currently best path (Daemon action).
      (d) Evaporate pheromone in all vertices.
   4.2. Refine the graph by one level.
5. Optionally perform LO of the currently best solution.

Fig. 1. Outline of the multilevel ant stigmergy algorithm.

graph $\mathscr{G} = (V,E)$ with a set of vertices

$$V = \bigcup_{d=1}^{D} V_d, \quad V_d = \{v_{\langle d,1\rangle}, \ldots, v_{\langle d,n_d\rangle}\}$$

and set of edges between the vertices

$$E = \bigcup_{d=1}^{D} E_d,$$
$$E_d = \{e_{\langle d-1,i\rangle,\langle d,j\rangle} = (v_{\langle d-1,i\rangle}, v_{\langle d,j\rangle}) \mid v_{\langle d-1,i\rangle} \in V_{d-1} \wedge v_{\langle d,j\rangle} \in V_d\},$$

where D represents the length of the longest path in the search graph, which equals the number of parameters, and n_d represents the number of discrete values of a parameter p_d. In this way, the multi-parameter optimization problem is transformed into a problem of finding the cheapest path in the search graph. Physical limitation of the graph size is determined by computer's RAM size; in 32-bit environment this means 10^6 vertices and in 64-bit environment 10^{15} vertices.

2. *Coarsening*. The search graph is coarsened to a predetermined size. Coarsening is done by merging two or more neighboring vertices into one vertex; this is done in L iterations (we call them levels $\ell = 1,2,\ldots,L$). Let us consider coarsening from level ℓ to level $\ell+1$ at a distance d. Here

$$V_d^\ell = \{v_{\langle d,1\rangle}^\ell, \ldots, v_{\langle d,n_d^\ell\rangle}^\ell\}$$

is a set of vertices at level ℓ and distance d of the search graph $\mathscr{G}$, where $1 \leq d \leq D$. If n_d^1 is the number of vertices at a starting level of coarsening

and a distance d, then for every level ℓ the equation $n_d^{\ell+1} = \lceil \frac{n_d^\ell}{s_d^\ell} \rceil$ is true, where s_d^ℓ is the number of vertices at level ℓ, which merge into one vertex at level $\ell+1$. So what we do is we divide V_d^ℓ into $n_d^{\ell+1}$ subsets, where

$$V_d^\ell = \bigcup_{k=1}^{n_d^{\ell+1}} V_{\langle d,k\rangle}^\ell, \forall i,j \in \{1,\ldots,n_d^{\ell+1}\} \wedge i \neq j : V_{\langle d,i\rangle}^\ell \cap V_{\langle d,j\rangle}^\ell = \emptyset.$$

Each subset is defined as follows:

$$V_{\langle d,1\rangle}^\ell = \{v_{\langle d,1\rangle}^\ell, \ldots, v_{\langle d,s_d^\ell\rangle}^\ell\},$$

$$V_{\langle d,2\rangle}^\ell = \{v_{\langle d,s_d^\ell+1\rangle}^\ell, \ldots, v_{\langle d,2s_d^\ell\rangle}^\ell\},$$

$$\vdots$$

$$V_{\langle d,n_d^{\ell+1}\rangle}^\ell = \{v_{\langle d,(n_d^{\ell+1}-1)s_d^\ell+1\rangle}^\ell, \ldots, v_{\langle d,n_d^\ell\rangle}^\ell\}.$$

Set $V_d^{\ell+1} = \{v_{\langle d,1\rangle}^{\ell+1}, \ldots, v_{\langle d,n_d^{\ell+1}\rangle}^{\ell+1}\}$ is the set of vertices at distance d at level $\ell+1$, where $v_{\langle d,k\rangle}^{\ell+1} \in V_{\langle d,k\rangle}^\ell$ is selected on some predetermined principle. For example, random pick, the most left/right/centered vertex in the subset, etc. The outline of the coarsening pseudo code from V_d^ℓ to $V_d^{\ell+1}$ is as follows:

For $k = 1$ to $n_d^{\ell+1}$ do
  $v_{\langle d,k\rangle}^{\ell+1}$ = SelectOneVertex($V_{\langle d,k\rangle}^\ell$)
EndFor

3. *Optimization*. Here the algorithm applies the optimization procedure based on ant-colony optimization [10] (loop 4.1 in Fig. 1). All ants simultaneously start from the starting vertex. The probability of choosing the next vertex depends on the amount of pheromone in the vertices. More specifically, ant α in step d moves from vertex $v_{\langle d-1,i\rangle} \in \{v_{\langle d-1,1\rangle}, \ldots, v_{\langle d-1,n_{d-1}\rangle}\}$ to vertex $v_{\langle d,j\rangle} \in \{v_{\langle d,1\rangle}, \ldots, v_{\langle d,n_d\rangle}\}$ with the probability given by:

$$\text{prob}_{ij,\alpha}(d) = \frac{\tau_{\langle d,j\rangle}}{\sum_{1\leq k\leq n_d} \tau_{\langle d,k\rangle}},$$

where $\tau_{\langle d,k\rangle}$ is the amount of pheromone on vertex $v_{\langle d,k\rangle}$. Ants repeat this action until reaching the ending vertex. The parameter values gathered on each ant's path represent a candidate solution which is then evaluated according to the given objective function. Afterwards, each ant returns to the starting vertex, on its way depositing pheromone in the vertices according to

the evaluation result: the better the result, the more pheromone is deposited. If the gathered parameter values form an infeasible solution, the amount of pheromone in the parameter vertices is slightly decreased. When the ants return to the starting vertex, two additional actions are performed. First, following ant colony optimization, "daemon action" is applied as a type of elitism, i.e., additional increase of the pheromone amount on the currently best path. Second, the pheromone in all vertices evaporates, therefore, in each vertex the amount of pheromone is decreased by some predetermined percentage ρ on each vertex $v_{\langle d,k\rangle}$ in the search graph $\mathscr{G}$:

$$\tau_{\langle d,k\rangle} \leftarrow (1-\rho)\tau_{\langle d,k\rangle}.$$

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

4. *Refinement*. The coarsened graph is refined by one level. Because of the simplicity of the coarsening, the refinement itself is very trivial. Let us consider refinement from level l to level $l-1$ at distance d. The outline of the refinement pseudo code is as follows:

```
For k = 1 to n_d^ℓ do
    For each v_⟨d,i⟩^(ℓ−1) ∈ V_⟨d,k⟩^(ℓ−1) do
        τ_⟨d,i⟩^(ℓ−1) = τ_⟨d,k⟩^ℓ
    EndFor
EndFor
```

All vertices created from one vertex have the same amount of pheromone as the original one. When refinement is done, the optimization phase continues. These two phases are repeated until the graph is expanded to its original size and the optimization performed on every level of the expansion.

5. *Local optimization*. This phase performs the steepest-descent local optimization (LO) on the currently best solution until a local minimum is found. Local optimization is optional, but usually significantly contributes to the quality of solutions.

3 Differential Evolution

Evolutionary Algorithms (e.g., CMA-ES [11]), the Particle Swarm Optimization [12], and *Differential Evolution* (DE) [13] are very popular numerical optimization procedures. The results reported in [14, 15] show that DE generally outperforms the other algorithms. Therefore, we decided to compare the MASA with DE.

Differential evolution is the stochastic, population-based optimization algorithm. It was introduced by Storn and Price [13] and was developed to optimize the real (float) parameters of a real-valued function. DE resembles the structure of an evolutionary algorithm, but differs from traditional evolutionary algorithms in its generation of new candidate solutions and by its use of a "greedy" selection scheme. The basic idea of DE is outlined in Fig. 2.

The candidate is calculated as a weighted sum of three randomly chosen individuals, that are different from the parent. Only then does the parent participates in the creation of the candidate – the candidate is modified by a crossover with its parent. Finally, the candidate is evaluated and compared to the parent. The candidate replaces the parent in the population, only if it is better than the parent. The described procedure (loop 2.1 in Fig. 2) is repeated for all the parent individuals from the population. When it is finished the individuals from the population are randomly enumerated and the procedure is repeated.

Differential Evolution

1. Evaluate the initial population S of random individuals.
2. While stopping criterion not met, do:
   2.1. For each parent $\mathbf{s}_i$ ($i = 1, \ldots,$ *popSize*) from S repeat:
      (a) Randomly select three individuals $\mathbf{s}_{i_1}, \mathbf{s}_{i_2}, \mathbf{s}_{i_3}$ from S, where i, i_1, i_2 and i_3 are pairwise different.
      (b) Calculate candidate $\mathbf{c}$ as $\mathbf{c} = \mathbf{s}_{i_1} + F \cdot (\mathbf{s}_{i_2} - \mathbf{s}_{i_3})$, where F is a scaling factor.
      (c) Modify the candidate by binomial crossover with the parent $\mathbf{s}_i$ using crossover constant CR.
      (d) Evaluate the candidate.
      (e) If the candidate is better than the parent, replace the parent with the candidate.
   2.2. Randomly enumerate the individuals in S.

Fig. 2. Outline of differential evolution.

4 Performance Evaluation

In this section we analyze the performance of the MASA and compare the MASA to DE. The evaluation is performed on a set of numerical benchmark functions and in designing optimal universal electric motor rotor and stator geometries where the primary objective is to minimize the motor power losses.

4.1 Benchmark functions

For the benchmark functions we have decided to use *Sphere* $f_{Sp}(\vec{x})$, *Griewangk* $f_{Gr}(\vec{x})$, *Rastrigin* $f_{Ra}(\vec{x})$, *Rosenbrock* $f_{Ro}(\vec{x})$, *Krink* $f_{Kr}(\vec{x})$, and *negative Krink*

$f_{\overline{Kr}}(\vec{x})$. For evaluation purposes we used three different function dimensions $D = 5$, 25, and 50. The function definitions are as follows:

$$f_{Sp}(\vec{x}) = \sum_{i=1}^{D} x_i^2,$$

$$f_{Gr}(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{D}(x_i - 100)^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i - 100}{\sqrt{i}}\right) + 1,$$

$$f_{Ra}(\vec{x}) = \sum_{i=1}^{D}\left(10 + x_i^2 - 10\cos(2\pi x_i)\right),$$

$$f_{Ro}(\vec{x}) = \sum_{i=1}^{D-1}\left(100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right),$$

$$f_{Kr}(\vec{x}) = \sum_{i=1}^{D}\left(-37.816415 - |x_i - 50| + 40\sin\left(\frac{5\pi x_i}{18}\right)\right),$$

$$f_{\overline{Kr}}(\vec{x}) = \sum_{i=1}^{D}\left(-89.016293 + |x_i - 50| - 40\sin\left(\frac{5\pi x_i}{18}\right)\right).$$

The global minimum of functions f_{Sp}, f_{Gr}, f_{Ra}, and f_{Ro} is exactly zero, while for f_{Kr} and $f_{\overline{Kr}}$ we set constants so that the global minimum is as close to zero possible, see Table 1.

Table 1. Function constraints and minimum values

Function	L_i	U_i	Δ	Minimum value
$f_{Sp}(\vec{x})$	-100	100	10^{-3}	$f_{Sp}(\overrightarrow{0}) = 0$
$f_{Gr}(\vec{x})$	-600	600	10^{-2}	$f_{Gr}(\overrightarrow{100}) = 0$
$f_{Ra}(\vec{x})$	-5.12	5.12	10^{-4}	$f_{Ra}(\overrightarrow{0}) = 0$
$f_{Ro}(\vec{x})$	-50	50	10^{-3}	$f_{Ro}(\overrightarrow{1}) = 0$
$f_{Kr}(\vec{x})$	0	100	10^{-3}	$f_{Kr}(\approx \overrightarrow{52.167}) \approx 0$
$f_{\overline{Kr}}(\vec{x})$	0	100	10^{-3}	$f_{\overline{Kr}}(\approx \overrightarrow{99.031}) \approx 0$

We ran the MASA and DE 30 times on each experiment. The number of maximum function evaluations per experiment was set to 500,000.

For the MASA the number of ants was 10, the ending condition for each level was “no best solution found for last 50 iterations” and the coarsening was implemented by merging two vertices into one. The DE algorithm has three parameters, which were set to the following values: the population size was 50, the crossover constant was 0.8, and the scaling factor was 0.5.

The evaluation results of the MASA and DE are presented in Table 2, where the best and the average solutions obtained in 30 runs are shown for each experiment. The standard deviation of the solutions and the average number of function

evaluations per experiment are also included in the table. We can see that on almost all functions and dimensions the MASA found an optimal or near-optimal solution. The only function where the MASA performed worse was f_{Ro}. The main reason for this is that the first expression $(100(x_{i+1}-x_i^2)^2$ has two global minima at $x_i = 0$ and $x_i = 1$, $i = 1,2,\ldots,D$, while the second expression $(x_i - 1)^2$ has only one global minimum at $x_i = 1$. The x_i^2 in the first expression prefers $x_i = 0$ over $x_i = 1$. Since the first expression dominates over the second, the MASA is at first mislead into solution $x_i = 0$, from where it can slowly move toward the global minimum at $x_i = 1$, $i = 1,2,\ldots,D$. Furthermore, for DE we can see that for f_{Sp} and f_{Gr} return near optimal results for all dimensions; for f_{Ra} and f_{Kr} return near optimal results only for $N = 5$; for f_{Ro} return near optimal results for $N < 50$; for $f_{\overline{Kr}}$ DE do not produce any good results in 500,000 evaluations.

A convergence comparison of the MASA and DE on high-dimensional functions can be seen in Fig. 3 and Fig. 4. We observe, with exception of f_{Ro}, that the MASA outperforms DE.

4.2 Electric motor design

The efficiency of an universal electric motor (UM) can be improved by reducing the power losses in the motor that originate in the iron and the copper. An approach to reducing the power losses is to optimize the geometry of the rotor and the stator. Due to the high magnetic saturation of the iron in a UM, this optimization task is non-linear. In our case, 10 mutually independent variable parameters defining the rotor and stator geometry are subject to optimization ($D = 10$). Predefined discrete step for all parameters is $\Delta = 0.1$ mm. The optimization task is to find the geometry parameter values that would generate the rotor and stator geometry with minimum power losses.

The MASA was run 20 times. On each run the algorithm coarsened the graph to level $L = 7$, and on each level of optimization we let 200 ants down the graph. So, overall the algorithm made 1400 evaluations (calculated with the ANSYS finite-element program) on each run. At the end, LO was applied, which required, on average, an additional 116.25 evaluations. The time required for a run was approximately one day on an Athlon XP 1800+ processor. Almost all the time is consumed by the solution evaluations.

The DE starts with a $popSize = 20$ initial solutions that are random perturbations of a predefined engineering solution. The stopping criterion was set to 1400 solutions, parameters were set to the following values: $CR = 0.9$, and $F = 0.5$.

The results (see Table 3) show that the MASA outperforms the DE and significantly improves the engineering design of the UM rotor and stator. In six out of 20 runs, MASA was able to find geometry parameter values resulting in power losses

Table 2. Experimental results for benchmark functions

Function	D	Algorithm	Best	Mean	Std	Avg iterations
f_{Sp}	5	DE	0	0	0	8785
	25		0	0	0	52230
	50		0	0	0	105560
	5	MASA	0	0	0	9703
	25		0	0	0	22852
	50		0	0	0	27562
f_{Gr}	5	DE	0	0	0	39697
	25		0	$0.986\ 10^{-3}$	$0.308\ 10^{-2}$	100140
	50		0	$0.493\ 10^{-3}$	$0.188\ 10^{-2}$	132061
	5	MASA	0	$0.616\ 10^{-1}$	$0.598\ 10^{-1}$	11347
	25		0	$0.148\ 10^{-1}$	$0.140\ 10^{-1}$	30761
	50		0	$0.328\ 10^{-2}$	$0.608\ 10^{-2}$	46472
f_{Ra}	5	DE	0	0	0	18222
	25		0.995	14.307	14.083	500000
	50		11.940	98.290	43.517	500000
	5	MASA	0	0	0	8885
	25		0	0.696	0.911	32084
	50		0	0.663	1.149	55824
f_{Ro}	5	DE	0	$0.315\ 10^{-7}$	$0.131\ 10^{-6}$	84646
	25		0	$0.139\ 10^{-1}$	$0.745\ 10^{-1}$	476097
	50		15.188	37.273	14.522	500000
	5	MASA	$0.133\ 10^{-1}$	$0.280\ 10^{-1}$	$0.102\ 10^{-1}$	80246
	25		$0.174\ 10^{-1}$	0.949	2.636	500000
	50		$0.744\ 10^{-1}$	5.126	18.595	500000
f_{Kr}	5	DE	$0.742\ 10^{-4}$	$0.742\ 10^{-4}$	0	18498
	25		14.000	209.900	86.522	500000
	50		600.382	921.951	156.130	500000
	5	MASA	$0.136\ 10^{-5}$	4.733	3.514	15751
	25		$0.681\ 10^{-5}$	3.547	3.955	59069
	50		$0.136\ 10^{-4}$	3.827	4.502	88073
$f_{\overline{Kr}}$	5	DE	0.418	8.100	8.466	72800
	25		19.795	87.365	39.688	272312
	50		97.104	228.734	65.940	500000
	5	MASA	$-0.609\ 10^{-3}$	5.613	5.334	21626
	25		$-0.304\ 10^{-2}$	4.690	5.997	56639
	50		$-0.609\ 10^{-2}$	3.221	4.465	86784

under 120 W. In the remaining 14 runs it performed comparably to DE.

The best rotor and stator geometry obtained in numerical optimization experiments generates power losses of 111.1 W. It was found by MASA with LO, and is presented in Fig. 5. These laminations have a large rotor and stator slots, and therefore low copper losses and overall power losses. The difficulty with this design is however in the strange dimensions of the stator pole. Its narrow middle part makes it unacceptable for production. This outcome is due to the settings in

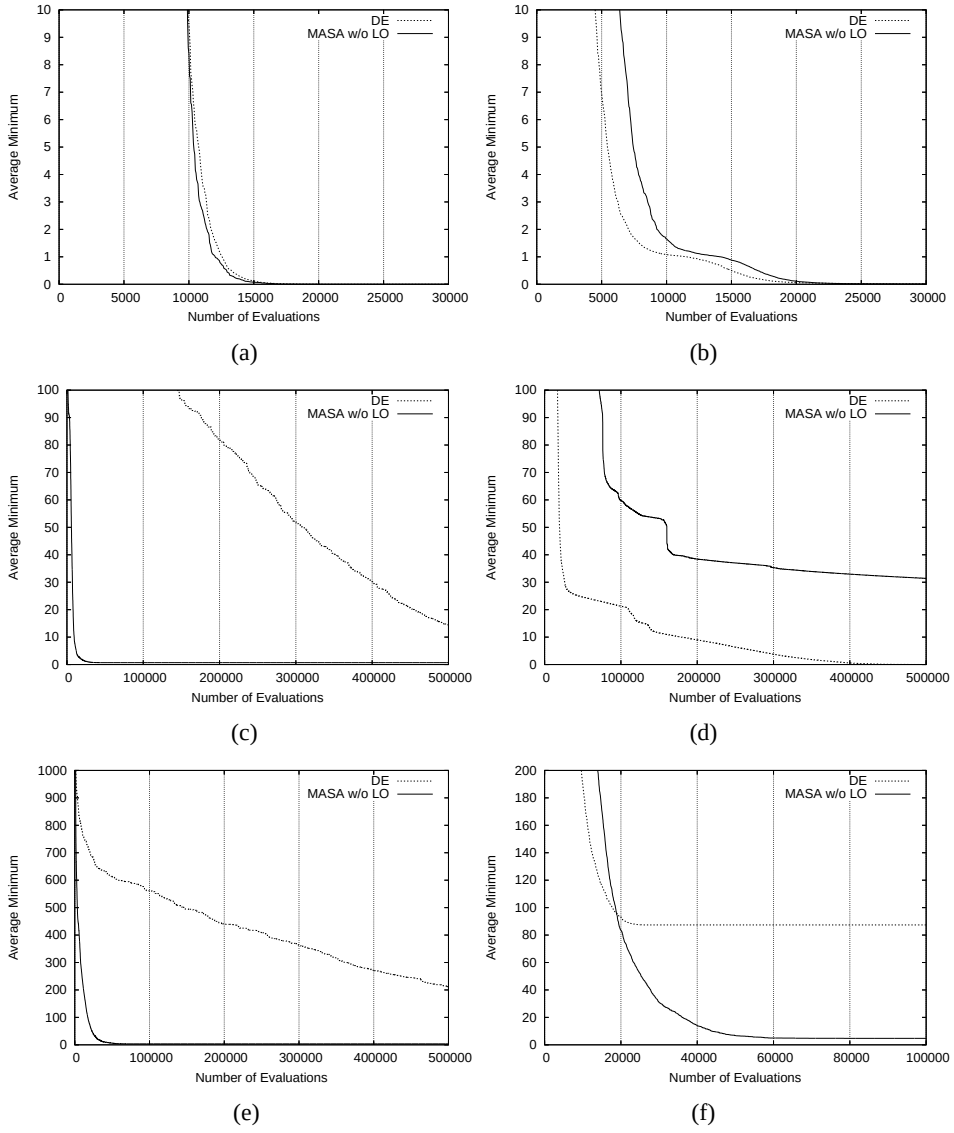

Fig. 3. Average minimum of (a) *Sphere*, (b) *Griewangk*, (c) *Rastrigin*, (d) *Rosenbrock*, (e) *Krink*, and (f) *negative Krink* functions with $D = 25$.

the simulation script used. It could be modified by inserting additional constraints on the stator geometry. Once they are specified, a new optimization cycle will be necessary. At this stage, however, the goal of our study is to check for possible improvements of the engineering UM design and compare the optimization methods on this problem [16].

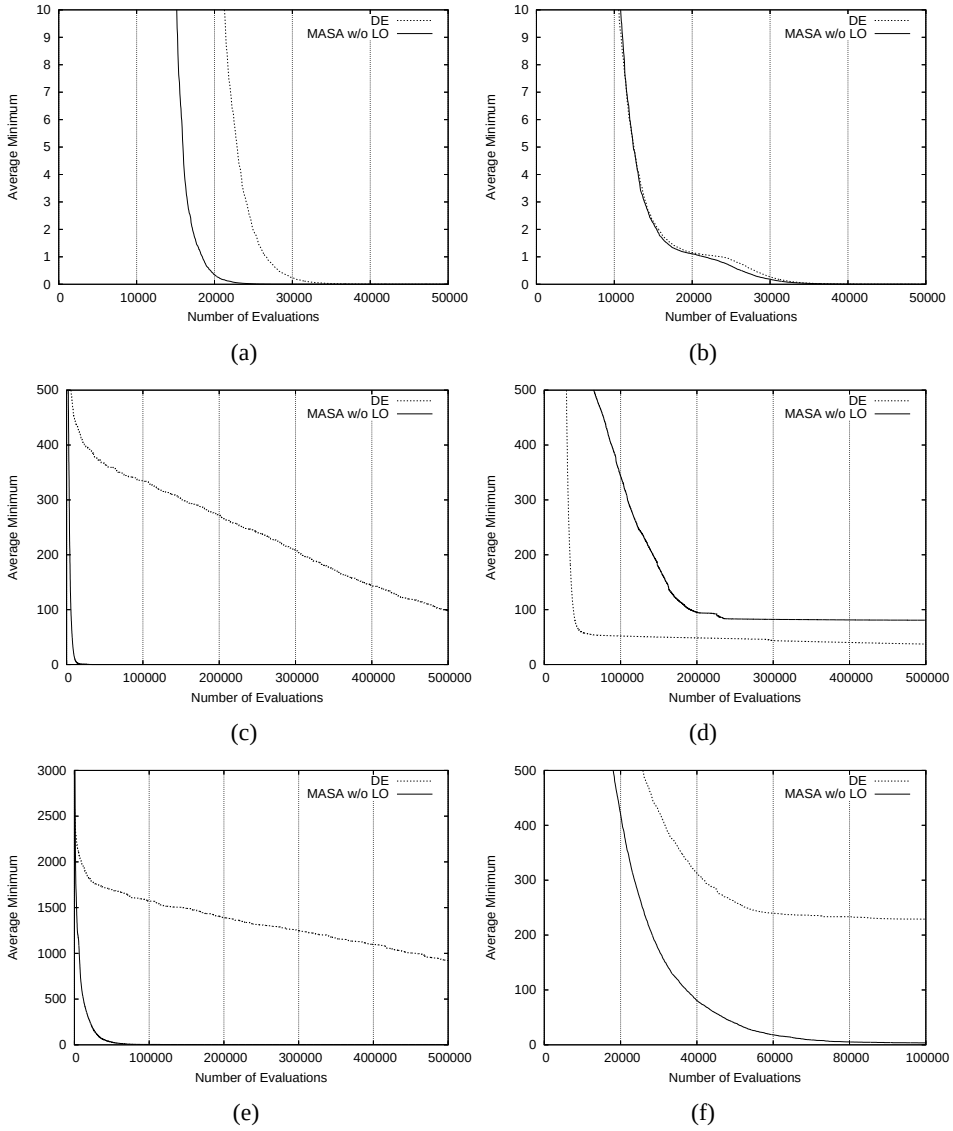

Fig. 4. Average minimum of (a) *Sphere*, (b) *Griewangk*, (c) *Rastrigin*, (d) *Rosenbrock*, (e) *Krink*, and (f) *negative Krink* functions with $D = 50$.

5 Conclusions

In this paper, we presented a new approach to solving numerical problems based on stigmergy, a type of collective work that can be observed in an ant colony. We proposed a general approach for the translation of a multi-parameter problem into

Table 3. Result of the electric motor optimization (power losses in watts)

Method	Best	Mean	Worst	Std
Engineering solution	177.9			
DE	129.1	132.9	139.9	3.3
MASA w/o LO	114.2	128.9	135.9	7.8
MASA	111.1	126.2	135.0	9.7

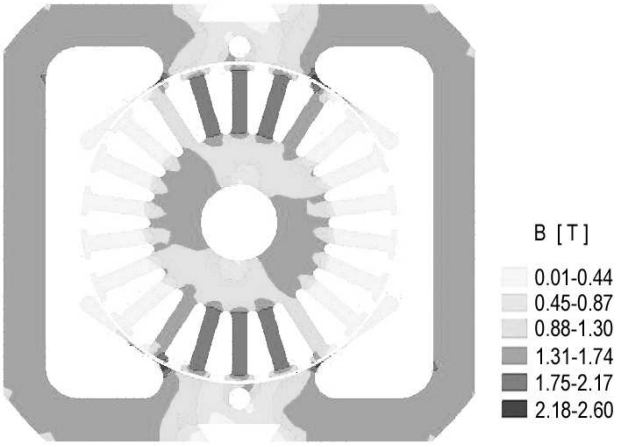

Fig. 5. Laminations of the rotor and stator design with minimum power losses (111.1 W) as found in the optimization experiments by the MASA. The figure shows maximum magnetic flux density *B* in tesla.

a search graph representation. Each longest path in the graph represents one possible solution, and all longest paths together represents the whole solution space of the multi-parameter problem. For an efficient search of the solution space we used a multilevel approach. We call this approach the Multilevel Ant Stigmergy Algorithm.

We evaluated the performance of the Multilevel Ant Stigmergy Algorithm and Differential Evolution in terms of their applicability as numerical optimization techniques. First, the comparison is performed with several widely used benchmarks functions (*Sphere*, *Griewangk*, *Rastrigin*, *Rosenbrock*, and *Krink*). It was determined that for lower-dimension functions the performance was comparable, while for higher dimensions the Multilevel Ant Stigmergy Algorithm outperformed Differential Evolution in all functions with the exception of one. Second, both methods was tested on a real industrial problem: the minimization of the power losses in a universal electric motor. The average solution obtained with the Multilevel Ant Stigmergy Algorithm was better than a solution recently found using Differential Evolution.

References

[1] J. Nocedal and S. J. Wright, *Numerical Optimization*. New York, Berlin, Heidelberg: Springer, 1999.

[2] W. J. Cook, W. H. Cunningham, W. R. Pulleyblank, and A. Schrijver, *Combinatorial Optimization*. New York: John Wiley & Sons, 1997.

[3] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison," *ACM Computing Surveys*, vol. 35, no. 3, pp. 268–308, Sept. 2003.

[4] P.-P. Grassé, "La reconstruction du nid et les coordinations inter-individuelle chez bellicositermes natalensis et cubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs," *Insectes Sociaux*, vol. 6, no. 41-81, 1959.

[5] O. Holland and C. Melhuish, "Stimergy, self-organization, and sorting in collective robotics," *Artificial Life*, vol. 5, no. 2, pp. 173–202, 1999.

[6] M. Roth and S. Wicker, "Termite: Ad-hoc networking with stigmergy," in *Proc. of the IEEE Global Telecommunications Conference (GLOBECOM 2003)*, vol. 22, no. 1, San Francisco, CA, Dec. 2003, pp. 2937–2941.

[7] K. P. Hadeli, P. Valckenaers, M. Kollingbaum, and H. Van Brussel, "Multi-agent coordination and control using stigmergy," *Computers in Industry*, vol. 53, no. 1, pp. 75–96, Jan. 2004.

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

[9] P. Korošec and J. Šilc, "The multilevel ant stigmergy algorithm: an industrial case study," in *Proc. of the 8th Joint Conference on Information Sciences*, Salt Lake City, UT, July 21–25, 2005, pp. 475–478.

[10] M. Dorigo, G. D. Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artificial Life*, vol. 5, no. 2, pp. 137–172, 1999.

[11] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distribution in evolution strategies: The covariance matrix adaptation," in *Proc. of 1996 IEEE International Conference on Evolution Computation (ICEC '96)*, Nagoya, Japan, May 20–22, 1996, pp. 312–317.

[12] R. C. Eberhart and J. Kennedy, "A new optimizer using particles swarm theory," in *Proc. of the 6th International Symposium on Micro Machine and Human Science*, Nagoya, Japan, Oct. 1995, pp. 39–43.

[13] R. Storn and K. Price, "Differential evolution - a simple and efficien adaptive scheme for global optimization over continuous spaces," International Computer Science Institute, Berkley, CA, Tech. Rep. TR 95-012, Mar. 1995.

[14] T. Krink, B. Filipič, G. B. Fogel, and R. Thomsen, "Noisy optimization problems - a particular challenge for differential evolution?" in *Proc. of the 2004 Congress on Evolutionary Computation (CEC2004)*, vol. 1, Portland Marriott Downtown, Portland, OR, June 19–23, 2004, pp. 332–339.

[15] J. Vesterstrøm and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proc. of the 2004 Congress on Evolutionary Computation (CEC2004)*, vol. 1, Portland Marriott Downtown, Portland, OR, June 19–23, 2004, pp. 1980–1987.

[16] T. Tušar, P. Korošec, G. Papa, B. Filipič, and J. Šilc, "A comparative study of stochastic optimization methods in electric motor design," Jožef Stafan Institute, Ljubljana, Slovenia, Tech. Rep. DP–9349, 2006.