We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

11

Applications of the Differential Ant-Stigmergy Algorithm on Real-World Continuous Optimization Problems

Peter Korošec and Jurij Šilc
Jožef Stefan Institute, Ljubljana
Slovenia

1. Introduction

Ants have always fascinated human beings. What particularly strikes the occasional observer as well as the scientist is the high degree of societal organization that these insects can achieve in spite of very limited individual capabilities (Dorigo et al., 2000). Ants have inspired also a number of optimization algorithms. These algorithms are increasingly successful among researches in computer science and operational research (Blum, 2005; Cordón et al., 2002; Dorigo & Stützle, 2004).

A particular successful metaheuristic—Ant Colony Optimization (ACO)—as a common framework for the existing applications and algorithmic variants of a variety of ant algorithms has been proposed in the early nineties by Marco Dorigo (Dorigo, 1992). ACO takes inspiration from the foraging behavior of some ant species. These ants deposit pheromone on the ground in order to mark some favorable path that should be followed by other members of the colony. ACO exploits a similar mechanism for solving combinatorial optimization problems.

In recent years ACO algorithms have been applied to more challenging and complex problem domains. One such domain is continuous optimization. However, a direct application of the ACO for solving continuous optimization problem is difficult.

The first algorithm designed for continuous function optimization was continuous ant colony optimization (Bilchev & Parmee, 1995) which comprises two levels: global and local; it uses the ant colony framework to perform local searches, whereas global search is handled by a genetic algorithm. Up to now, there are few other adaptations of ACO algorithm to continuous optimization problems: continuous interacting ant colony (Dréo & Siarry, 2002), ACO for continuous and mixed-variable (Socha, 2004), aggregation pheromone system (Tsutsui, 2006), and multilevel ant-stigmergy algorithm (Korošec & Šilc, 2008).

In this chapter we will present so-called Differential Ant-Stigmergy Algorithm (DASA), a new approach to the continuous optimization problem (Korošec, 2006). We start with the DASA description followed by three case studies which show real-world application of the proposed optimization approach. Finally, we conclude with discussion of the obtained results.

Source: Evolutionary Computation, Book edited by: Wellington Pinheiro dos Santos,
ISBN 978-953-307-008-7, pp. 572, October 2009, I-Tech, Vienna, Austria

Fig. 1. Stigmergy is an organizing principle in emergent systems in which the individual parts of the system communicate with one another indirectly by modifying their local environment. Ant colonies are a classic example. The ants communicate indirectly. Information is exchanged through modifications of the environment (local gradients of pheromone).

2. A differential ant-stigmergy approach

2.1 Continuous optimization

The general continuous optimization problem is to find a set of parameter values, $\boldsymbol{p} = (p_1, p_2, \ldots, p_D)$, that minimizes a function, $f_{\text{cost}}(\boldsymbol{p})$, of D real variables, i.e.,

$$\text{Find:} \quad \boldsymbol{p}^* \mid f_{\text{cost}}(\boldsymbol{p}^*) \leq f_{\text{cost}}(\boldsymbol{p}), \forall \boldsymbol{p} \in \Re^D. \tag{1}$$

To solve this problem, we created a fine-grained discrete form of continuous domain. With it we were able to represent this problem as a graph. This enabled us to use ant-based approach for solving numerical optimization problems.

2.2 The fine-grained discrete form of continuous domain

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

$$p_i = p'_i + \delta_i. \tag{2}$$

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

$$\Delta_i = \Delta_i^- \cup \mathbf{0} \cup \Delta_i^+, \tag{3}$$

where

$$\Delta_i^- = \{\delta_{i,k}^- \mid \delta_{i,k}^- = -b^{k+L_i-1}, k = 1,2,\ldots d_i\} \tag{4}$$

and

$$\Delta_i^+ = \{\delta_{i,k}^+ \mid \delta_{i,k}^+ = +b^{k+L_i-1}, k = 1,2,\ldots d_i\}. \tag{5}$$

Here

$$d_i = U_i - L_i + 1. \tag{6}$$

Therefore, for each parameter p_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*,

$$L_i = \lfloor \lg_b(\varepsilon_i) \rfloor, \tag{7}$$

and

$$U_i = \lfloor \lg_b(\max(p_i) - \min(p_i)) \rfloor. \tag{8}$$

With the parameter ε_i, the maximum precision of the parameter p_i is set. The precision is limited by the computer's floating-point arithmetic. To enable a more flexible movement over the search space, the weight ω is added to Eq. 2:

$$p_i = p'_i + \omega\delta_i. \tag{9}$$

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

2.3 Graph representation

From all the sets Δ_i, $1 \le i \le D$, where D represents the number of parameters, a so-called *search graph* $G = (V, E)$ with a set of vertices, V, and a set of edges, E, between the vertices is constructed (see Fig. 2). Each set Δ_i is represented by the set of vertices,

$$V_i = \{v_{i,1}, v_{i,2}, \ldots, v_{i,2d_i+1}\}, \tag{10}$$

and

$$V = \bigcup_{i=1}^{D} V_i. \tag{11}$$

Then we have that

$$\Delta_i = \underbrace{\{\delta_{i,d_i}^-, \ldots, \delta_{i,d_i-j+1}^-, \ldots, \delta_{i,1}^-\}}_{\Delta_i^-}, 0, \underbrace{\{\delta_{i,1}^+, \ldots, \delta_{i,j}^+, \ldots, \delta_{i,d_i}^+\}}_{\Delta_i^+} \tag{12}$$

is equal to

$$V_i = \{v_{i,1}, \ldots, v_{i,j}, \ldots, \underbrace{v_{i,d_i+1}}_{0}, \ldots, v_{i,d_i+1+j}, \ldots, v_{i,2d_i+1}\}, \tag{12}$$

where $j = 1,2,\ldots,d_i$.

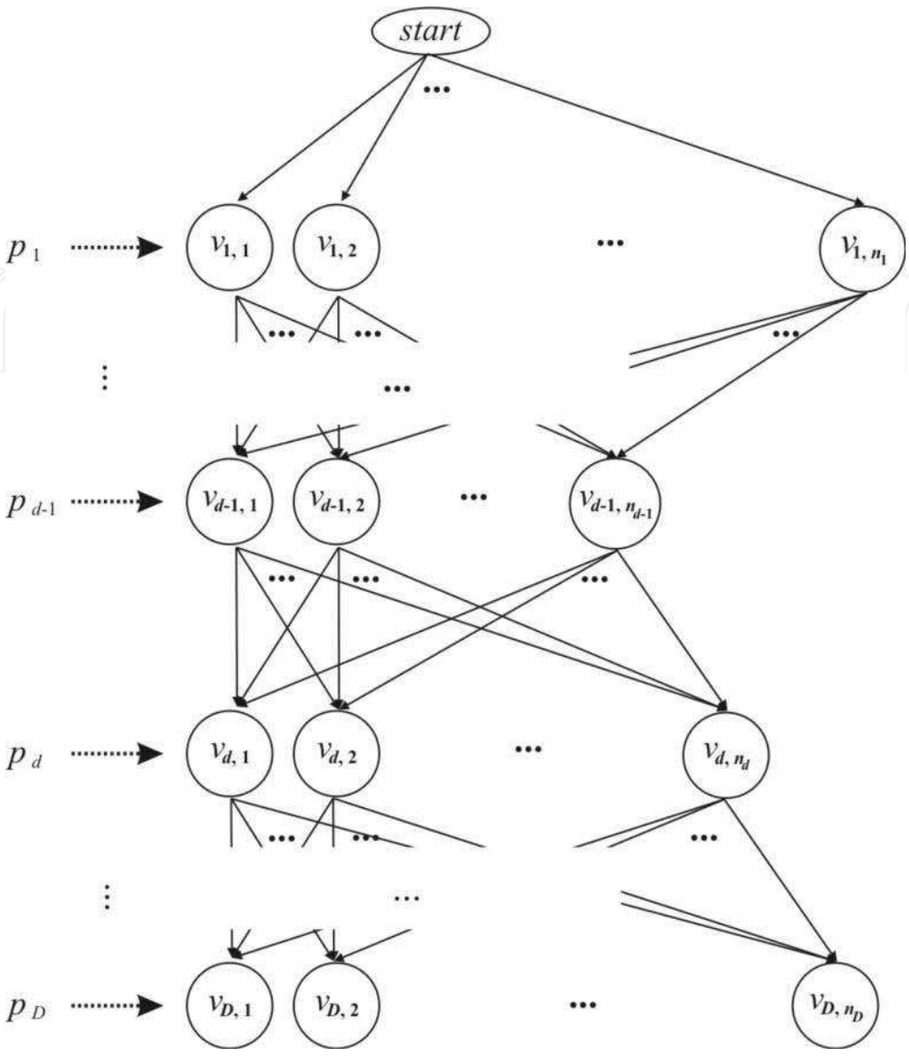

Fig. 2. Search graph representation of a discrete multi-parameter optimization problem

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 ν from start vertex to any of the ending vertices is of equal length and can be defined with v_i as:

$$\nu = (v_1 v_2 \ldots v_D), \tag{13}$$

where $v_i \in V_i$, $1 \le i \le D$.

The optimization task is to find a path ν, such that $f_{\text{cost}}(p) \le f_{\text{cost}}(p')$, where p' is currently the best solution, and $p = p' + \Delta(\nu)$ (using Eq. 9). Additionally, if the objective function $f_{\text{cost}}(p)$ is smaller than $f_{\text{cost}}(p')$, then the p' values are replaced with p values.

2.4 The differential ant stigmergy algorithm

The optimization consists of an iterative improvement of the currently best solution, p', by constructing an appropriate path ν, that uses Eq. 9 and returns a new best solution. This is done as follows (see Fig. 3):

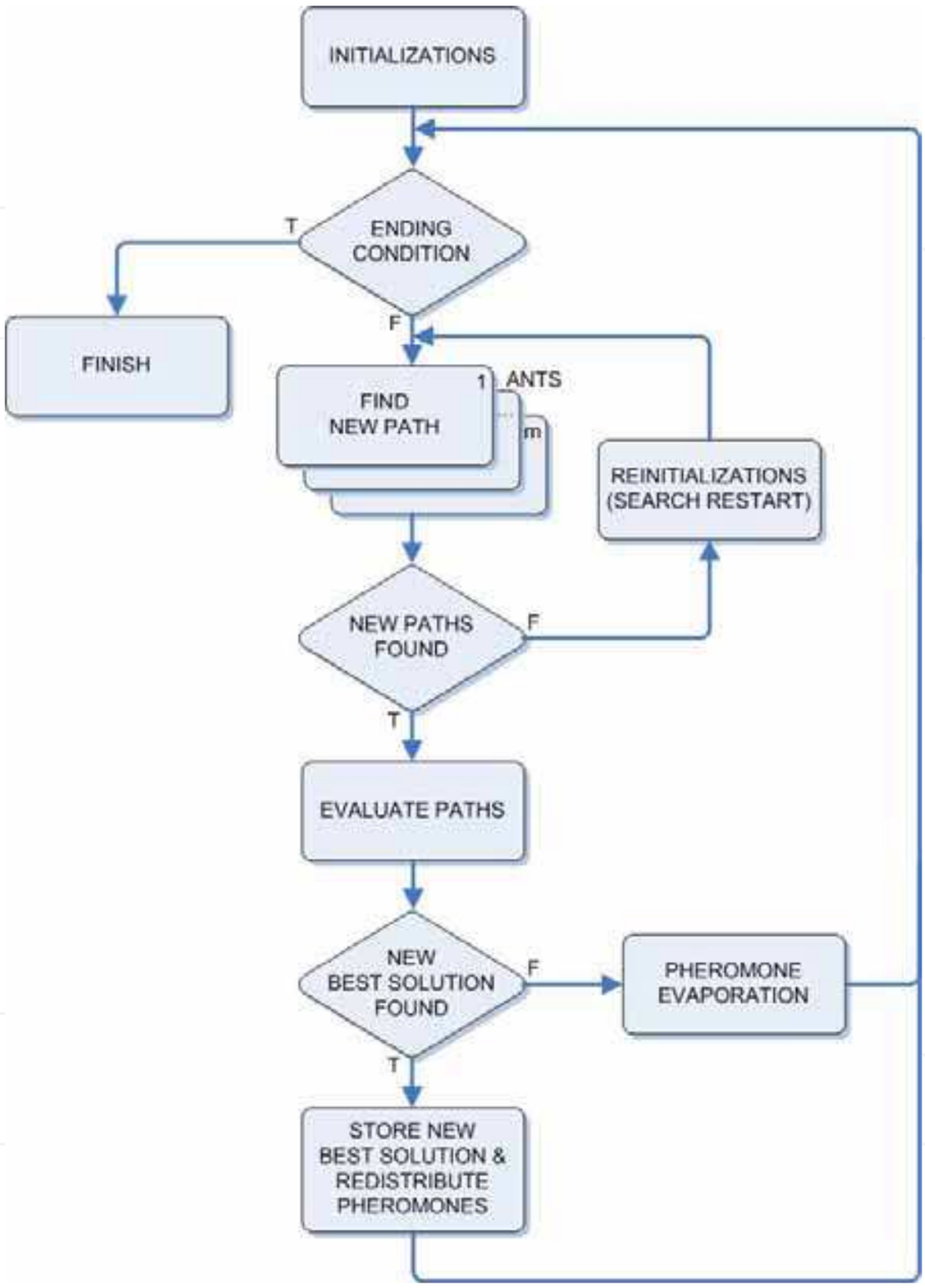

Fig. 3. The Differential Ant-Stigmergy Algorithm (DASA)

Step 1. A solution p' is manually set or randomly chosen.

Step 2. A search graph is created and an initial amount of pheromone, $\tau_{V_i}^0$, is deposited on all the vertices from the set $V_i \subset V$, $1 \le i \le D$, according to the Cauchy probability density function

$$C(x) = \frac{1}{s\pi + \frac{\pi}{s}(x-l)^2}, \tag{14}$$

where l is the location offset and $s = s_{\text{global}} - s_{\text{local}}$ is the scale factor. For an initial pheromone distribution the standard Cauchy distribution ($l = 0,\ s_{\text{global}} = 1,$ and $s_{\text{local}} = 0$) is used and each parameter vertices are equidistantly arranged between $z = [-4,4]$.

Step 3. There are m ants in a colony, all of which begin simultaneously from the *start* vertex. Ants use a probability rule to determine which vertex will be chosen next. More specifically, ant α in step i moves from a vertex in set V_i to vertex $v_{i,j} \in \{v_{i,1}, v_{i,2}, \ldots, v_{i,2d_i+1}\}$ with a probability, *prob*, given by:

$$prob_j(\alpha, i) = \frac{\tau(v_{i,j})}{\sum_{1 \le k \le 2d_i+1} \tau(v_{i,k})}, \tag{15}$$

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

The ants repeat this action until they reach the ending vertex. For each ant, path $\boldsymbol{\nu}$ is constructed. If for some predetermined number of tries we get $\boldsymbol{\nu} = \boldsymbol{0}$ the search process is reset by randomly choosing new $\boldsymbol{p}^{\text{temporary\_best}}$ and pheromone re-initialization. New solution $\boldsymbol{p}$ is constructed (see Eq. 9) and evaluated with a calculation of $f_{\text{cost}}(\boldsymbol{p})$.

The current best solution, $\boldsymbol{p}^{\text{current\_best}}$, out of m solutions is compared to the temporary best solution $\boldsymbol{p}^{\text{temporary\_best}}$. If $\boldsymbol{p}^{\text{current\_best}}$ is better than $\boldsymbol{p}^{\text{temporary\_best}}$, then $\boldsymbol{p}^{\text{temporary\_best}}$ values are replaced with $\boldsymbol{p}^{\text{current\_best}}$ values. In this case s_{global} is increased (in our case for 1 %) and pheromone amount is redistributed according to the associated path $\boldsymbol{\nu}^{\text{best}}$. Furthermore, if new $\boldsymbol{p}^{\text{temporary\_best}}$ is better then $\boldsymbol{p}^{\text{best}}$, then $\boldsymbol{p}^{\text{best}}$ values are replaced with $\boldsymbol{p}^{\text{temporary\_best}}$ values. So, global best solution is stored. If no better solution is found s_{local} is decreased (in our case for 3 %).

Step 4. Pheromone dispersion is defined by some predetermined percentage χ. The probability density function $C(x)$ is changed in the following way:

$$l \leftarrow (1-\chi)l \tag{16}$$

and

$$s_{\text{local}} \leftarrow (1-\chi)s_{\text{local}}. \tag{17}$$

Pheromone dispersion has a similar effect as pheromone evaporation in classical ACO algorithm.

Step 5. The whole procedure is then repeated until some ending condition is met.

It is a well known that ant-based algorithms have problems with convergence. This happens when on each step of the walk there is a large number of possible vertices from which ant can choose from. But this is not the case with the DASA where Cauchy distribution of pheromone over each parameter was used. Namely, such distribution reduces the width of the graph to only few dominant parameter values (i.e., vertices). On the other hand, with proper selection of the discrete base, *b*, we can also improve the algorithm's convergence (larger *b* reduces the search graph size).

2.5 Software implementation and parameter settings

The DASA was implemented in the Borland® Delphi™ programming language (see Fig. 4). The computer platform used to perform the optimizations was based on AMD Opteron™ 2.6-GHz processors, 2 GB of RAM, and the Microsoft® Windows® XP 32-bit operating system.

The DASA parameters were set as follows: the number of ants, m, was set to 10, the pheromone evaporation factor, χ, was set to 0.2, and the maximum parameter precision, ε, dependent on the discrete step of each parameter.

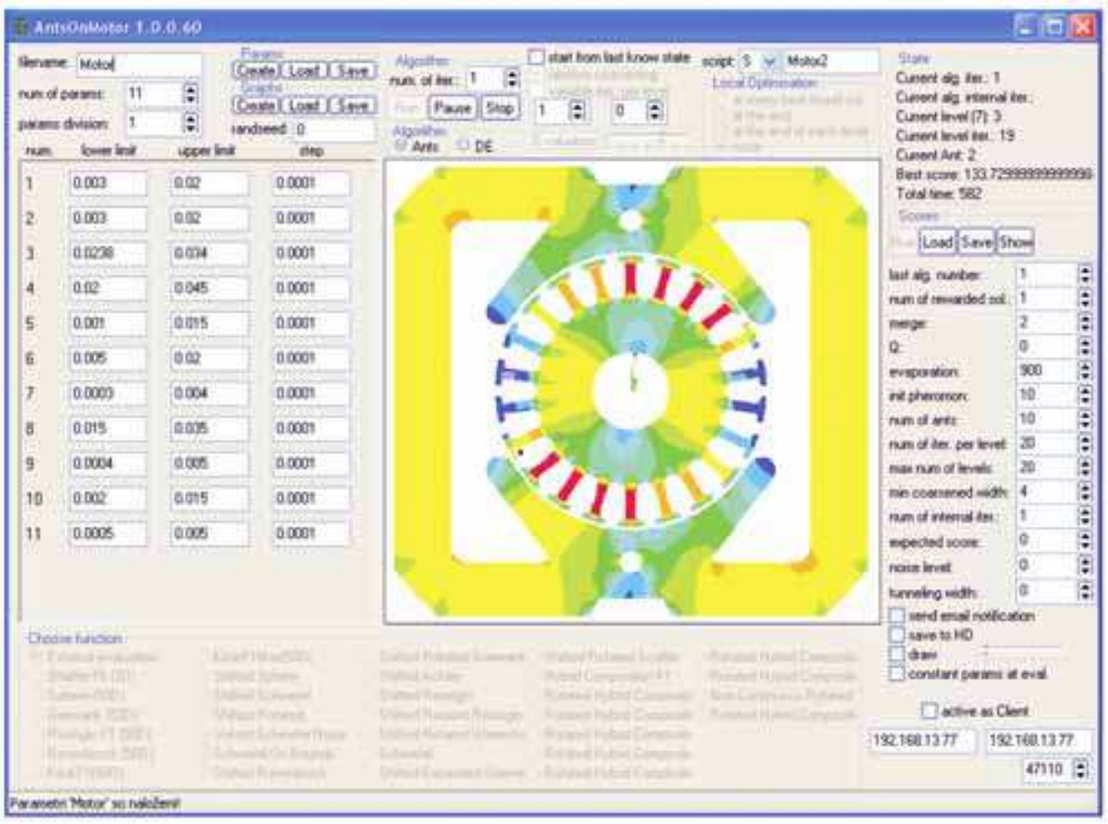

Fig. 4. GUI used for setting the DASA options

3. Case studies

Case studies presented in this section are related to the development of a new dry vacuum cleaner (Korošec et al., 2007; Tušar et al., 2007). In particular, the efficiency of the turbo-compressor unit (see Fig. 5) was improved.

3.1 An electric motor power losses minimization

Home appliances, such as vacuum cleaners and mixers, are generally powered by a universal electric motor (UM). These appliances need as low as energy consumption, that is, input power, as possible, while still satisfying the needs of the user by providing sufficient output power. The optimization task is to find the geometrical parameter values that will generate the rotor and the stator geometries resulting in the minimum power losses.

Fig. 5. Turbo-compressor unit of a dry vacuum cleaner

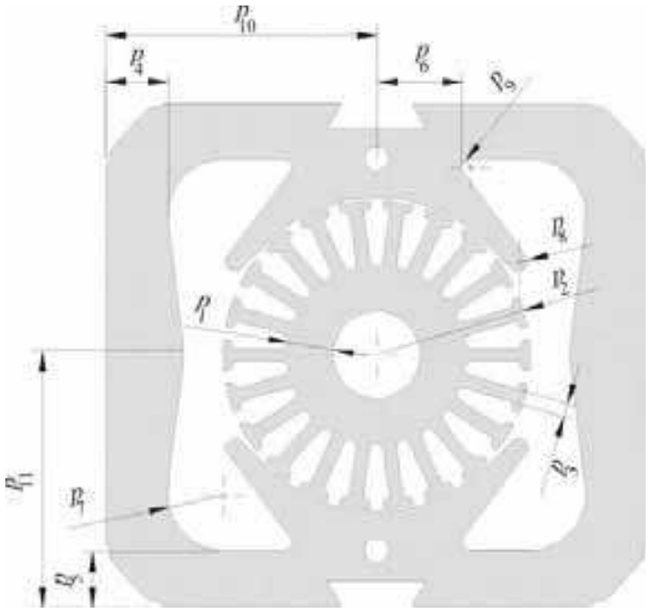

Fig. 6. Parameters that define the rotor and stator geometries

There are several invariable and variable parameters that define the rotor and stator geometries. The invariable parameters are fixed; they cannot be altered, either for technical reasons (e.g., the air gap) or because of the physical constraints on the motor (e.g., the radius of the rotor's shaft). The variable parameters do not have predefined optimum values. Some variable parameters are mutually independent and without any constraints. Others are dependent, either on some invariable parameters or on mutually independent ones.

In our case, 11 mutually independent variable parameters defining the rotor and stator geometries are subject to optimization (see Fig. 6): rotor yoke thickness, p_1, rotor external radius, p_2, rotor pole width, p_3, stator yoke horizontal thickness, p_4, stator yoke vertical thickness, p_5, stator middle-part length, p_6, stator internal edge radius, p_7, stator teeth radius, p_8, stator slot radius, p_9, stator width, p_{10}, and stator height, p_{11}. We optimized only 10 parameters (the ratio between p_{10} and p_{11} was set constant) of the UM rotor and stator geometries.

Power losses, P_{losses}, are the main factor affecting the efficiency of a UM. The efficiency of a UM, η, is defined as the ratio of the output power, P_{out}, to the input power, P_{inp}:

$$\eta = \frac{P_{\text{out}}}{P_{\text{inp}}} = \frac{P_{\text{out}}}{P_{\text{out}} + P_{\text{losses}}} \tag{18}$$

and it is very dependent on various power losses (Sen, 1997):

$$P_{\text{losses}} = P_{\text{Cu}} + P_{\text{Fe}} + P_{\text{brush}} + P_{\text{vent}} + P_{\text{frict}}. \tag{19}$$

The overall copper losses, P_{Cu}, occurring in the rotor and the stator slots are as follows:

$$P_{\text{Cu}} = \sum_i (J^2 A \rho l_{\text{turn}})_i, \tag{20}$$

where i stands for each slot, J is the current density, A is the slot area, ρ is the copper's specific resistance and l_{turn} is the length of the winding turn. The calculation of the iron losses, P_{Fe}, is less exact because iron has non-linear magnetic characteristics. The iron losses are therefore separated into two components: the hysteresis losses and the eddy-current losses. This means the motor's iron losses can be expressed with the formula

$$P_{\text{Fe}} = c_e B^2 f_r^2 m_r + c_e B^2 f_s^2 m_s + c_h B^2 f_s m_s, \tag{21}$$

where c_e is the eddy-current material constant at 50Hz, c_h is the hysteresis material constant at 50Hz, B is the maximum magnetic flux density, f_r is the frequency of the magnetic field density in the rotor, f_s is the frequency of the magnetic field density in the stator, m_r is the mass of the rotor, and m_s is the mass of the stator. Three additional types of losses occurring in the UM, i.e., the brush losses, P_{brush}; the ventilation losses, P_{vent}; and the friction losses, P_{fric}; depend mainly on the speed of the motor. When optimizing the geometries of the rotor and the stator, the motor's speed is assumed to be fixed; hence P_{brush}, P_{vent}, and P_{fric} have no impact on the motor's efficiency, and so these losses are not significantly affected by the geometries of the rotor and the stator. Therefore, in our case, the overall copper and iron losses can be used to estimate the cost function:

$$f_c(\boldsymbol{p}) = P_{\text{Cu}} + P_{\text{Fe}}. \tag{22}$$

Our goal is to find such parameter-value settings, where $f_c(\boldsymbol{p})$ is minimized.

The DASA starts its search with the existent solution. The stopping criterion was set to 1,400 calculations. The calculation of a single solution via the ANSYS Multiphysics simulation takes approximately two minutes, which means that the execution of 1,400 calculations needs about two days. The optimization method was run 20 times. The obtained results in terms of the UM power losses are presented statistically in Table 1.

Existing solution	Optimized solutions		
	Worst	Mean	Best
177.9	136.7	129.5	113.8

Table 1. Optimized UM's power losses in Watts after 1,400 calculations

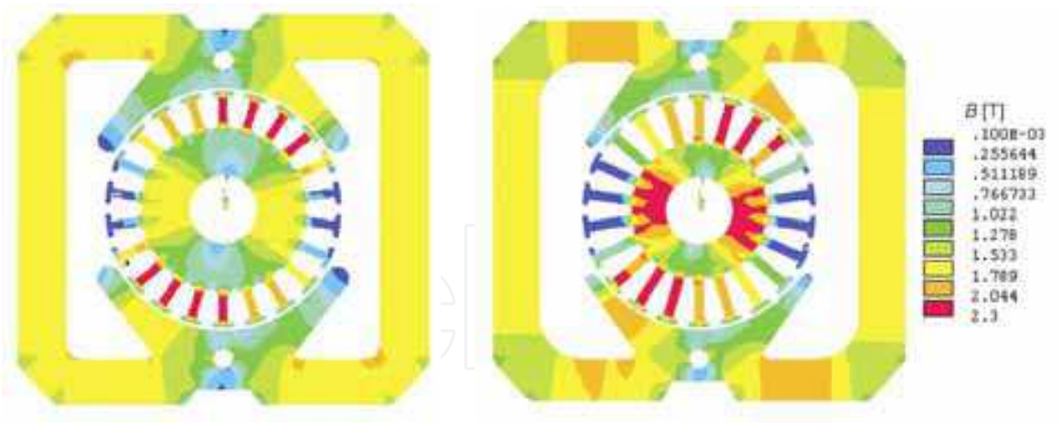

Fig. 7. Laminations of the original engineering design with power losses of 177.9 W (left) and laminations of an optimized design acceptable from the production point of view with power losses of 129.1 W (right)

The engineering rotor and stator design (the existent solution) results in power losses of 177.9 W, and can be seen in Fig. 7 (left). The figure shows the magnetic flux density in the laminations (higher magnetic flux density, B, causes higher power losses). Figure 7 (right) present a typical example of a feasible rotor and stator geometry, with power losses of 129.1 W. This solution has very low iron losses in the rotor due to its small size and in spite of its high magnetic saturation. The small rotor and its saturation are compensated by large stator poles that ensure large enough a magnetic flux. This design is completely feasible from the technical and production points of view.

3.2 An electric motor casing stiffness maximization

The casing is a part of the dry vacuum cleaner motor which is built into vacuum cleaners. The casing is basically an axisymmetric shell structure which is built of steel suitable for forming. For this procedure which consists of eleven different phases it is important that the radii are growing or do not change while the height is growing. That is an important rule which must not be broken during the geometry optimization. The goal of optimization is to preserve the stiffness while using a thinner shell structure and consequently save material and reduce costs.

The computational model of the casing comprises 26 different parameters which generate the geometry variations. The classical parameters are: radius on the top, p_1, radius on the side, p_2, radius at the groove for brushes, p_3, deviation of the hole for diffuser fixation, p_4, radius of roundness at the beginning of the bearing groove, p_5, radius on the top of the bearing groove, p_6, radius on the top of air culverts, p_7, radius at the bottom of air culverts, p_8, roundness of air culverts, p_9, angle of slope, p_{22}, angle of culverts span, p_{23}, slope of bearing groove, p_{24}, height of bearing groove, p_{25}, and slope of the brushes groove, p_{26}.

Besides the above listed 14 classical parameters there are 12 more (p_{10} to p_{21}), with which the ribs on areas A, B and C are defined (Fig. 8). These are essential to improve the stiffness so we will search for those which maximize the cost function. With these parameters we can generate a lot of combinations of different rib shapes, depending on the number of given paths and accuracy of intervals for points deviations, positions and number of ribs. In our

case there were several billion combinations for ribs. It is obvious that instead of a defined path at position *j*, we could define displacements for individual points but this would result in many more parameters and time consuming calculation procedure. The question still remains what would be the benefit of that. From previous designs there exist some reasonable shapes for ribs which have been defined intuitively and these then change their position, magnitude and number, which are probably sufficient for a good result, as there has been over ten different rib shapes defined earlier.

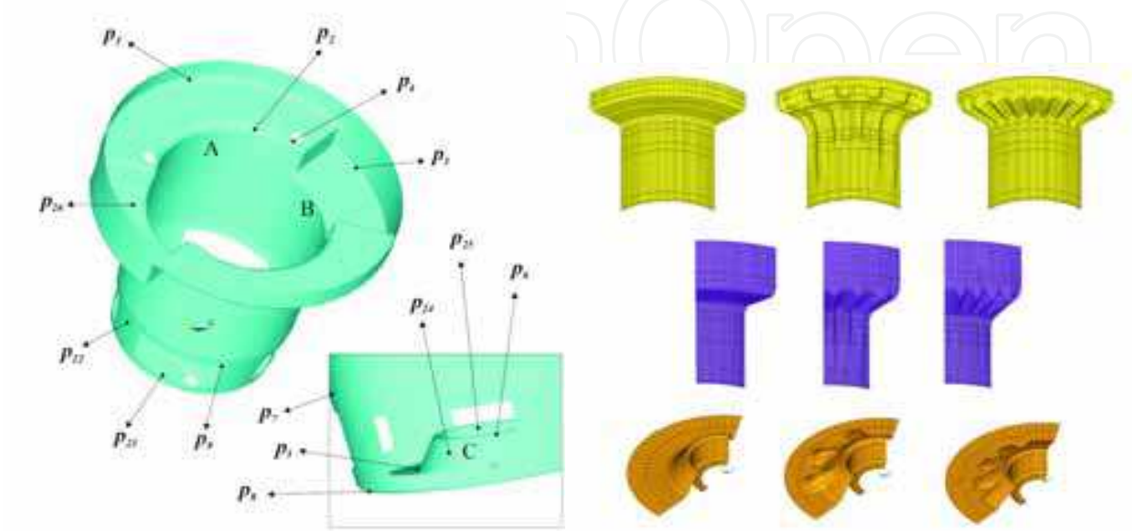

Fig. 8. Parameters of a casing and ribs. Ribs A (top), B (center), and C (bottom); basic (left) and modify forms (center and right).

Besides the parameters for the creation of ribs there are other classical parameters which also have an influence on their shape. Namely, if the radii p_1 or p_2 are varying then this is reflected in the shape of ribs of model A or if the height of bearing groove, p_{25}, and slope of groove, p_{24}, are modified then this will be reflected in the ribs of model C, etc.
The cost function is relatively simply defined when loadings are deterministic. Usually we optimize such cases with the minimization of displacements or stresses, maximization of elasticity, etc. (Dražumerič & Kosel, 2005). It is difficult to determine the cost function for a dynamically loaded assembly where the loads are stochastic. So the question is how to define the stiffness of a shell structure if we do not know the lateral loads. Let us consider a certain arbitrarily shaped plane shell. We can write its potential energy, W_p, as:

$$W_p = \frac{1}{2}\int \sigma_{ij}\varepsilon_{ij} dV. \tag{23}$$

If we consider that if the Poissons shear modulus is neglected, the stress tensor, σ_{ij}, and strain tensor, ε_{ij}, are the following:

$$\sigma_{ij} = -z\left\{\frac{\partial^2 w(x,y)}{\partial x^2}, \frac{\partial^2 w(x,y)}{\partial y^2}, \frac{\partial^2 w(x,y)}{\partial x \partial y}\right\}^T \tag{24}$$

and

$$\varepsilon_{ij} = -zE\left\{\frac{\partial^2 w(x,y)}{\partial x^2}, \frac{\partial^2 w(x,y)}{\partial y^2}, \frac{\partial^2 w(x,y)}{\partial x \partial y}\right\}^T. \tag{25}$$

Here $w(x,y)$ is displacement in z direction and E is elasticity modulus. The potential energy can then be written as:

$$W_p = \frac{1}{2E}\int_V (\sigma_{xx}^2 + \sigma_{yy}^2 + 2\tau_{xy}^2)dV. \tag{26}$$

We can see that in this case the potential energy is an integral of squares of stresses over the volume. We define also the kinetic energy, W_k, for plate vibrations:

$$W_k = \frac{\rho\omega^2}{2}\int_V zw^2(x,y)dV, \tag{27}$$

where ρ represents the density of material. Here the second derivative of displacement is approximated with respect to time with the product of squared displacement and eigen frequency, ω.

For conservative systems the maximal potential energy is equal to maximal kinetic energy and from this follows the expression for cost function which can be in our case the stiffness itself:

$$f_c(p) = \frac{\int_V (\sigma_{xx}^2 + \sigma_{yy}^2 + 2\tau_{xy}^2)dV}{\int_V zw^2(x,y)dV}. \tag{28}$$

The DASA was run 20 times and each run consisted of 2,000 calculations. The cost function was calculated by the ANSYS Multiphysics simulation tool. The obtained results are presented statistically in Table 2.

Existing solution	Optimized solutions		
	Worst	Mean	Best
$5.87 \cdot 10^{-3}$	$6.13 \cdot 10^{-3}$	$6.81 \cdot 10^{-3}$	$7.34 \cdot 10^{-3}$

Table 2. Optimized casing's stiffness after 2,000 CFD calculations

The results of optimization were quite surprising (Fig. 9 bottom row). It was expected that the ribs would form on all the given surface but they did not. The ribs were more distinct where the vertical part of the surface was turning into the horizontal one (ribs A). There were no ribs at the surface where the stator is in a tight fit, probably because of pre-stressing of the casing through the stator. Also in the groove for brushes (ribs B) there were no distinct ribs, probably because the groove itself is a kind of rib. Instead of this there is a given slope for a groove which was not there before. The radii of roundings were in most cases bigger than before as we can clearly see at the air culverts (Fig. 9 middle bottom).

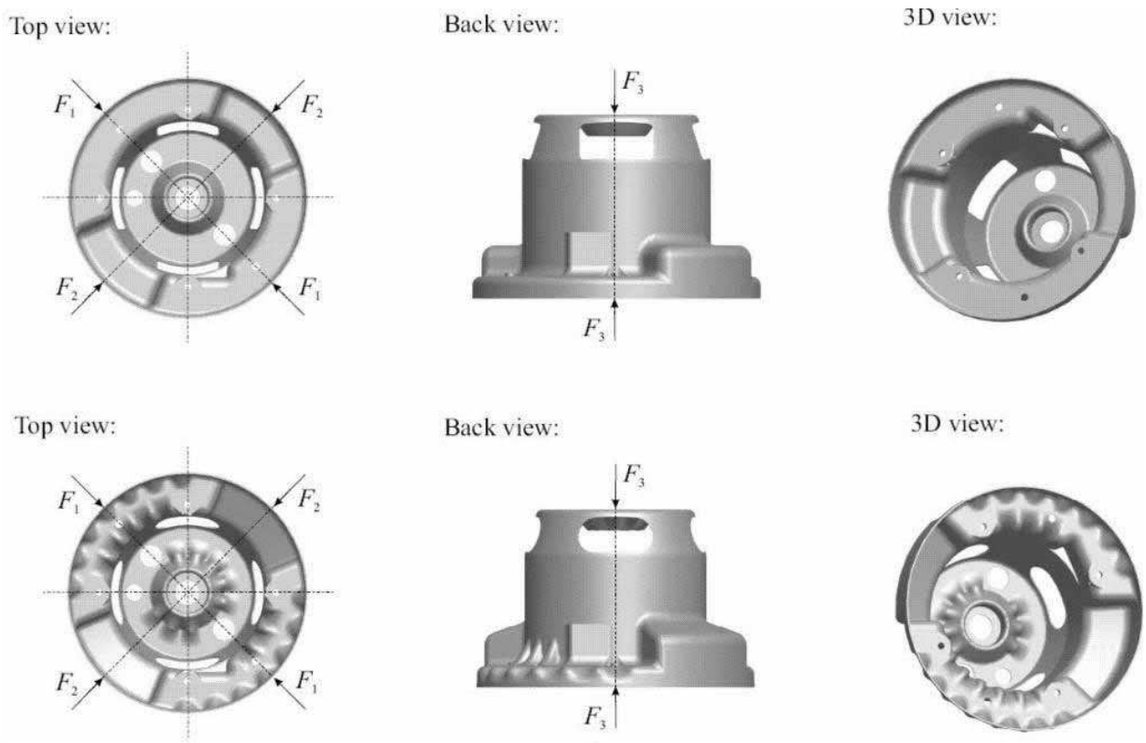

Fig. 9. Existent (up) and optimized (bottom) casing and different loading cases

3.3 A turbo-compressor aerodynamic power maximization

Radial air impellers are the basic components of many turbo-machines. In the following we will concentrate on relatively small impellers and subsonic speeds used in a dry vacuum cleaner. Our main aim was to find an impeller shape that has a higher efficiency, i.e., greater aerodynamic power, than the one currently used in production.

An impeller is constructed from blades, an upper and a lower side. The sides enclose the blades and keep them together. The blades, which are all the same, were the main part of the optimization. The geometry of a blade is shown in Fig. 10, where the gray color represents the blade. The method of modeling is as follows: we construct the points at specific locations, draw the splines through them and spread the area on the splines. Once a blade is made an air channel must be constructed in a similar way.

In Fig. 10a the point 1 has two parameters: the radius p_1 and the angle p_2. Similarly, the points 2, 5 and 6 have parameter pairs p_3 and p_4, p_5 and p_6, p_7 and p_8. The points 3 and 4 are fixed on the x axis. This is because the impeller must have a constant outer radius p_9 and the outer side of the blade must be parallel to the z axis. On the other hand, the outer angle of the blade p_{10} and the angle of the spline at points 3 and 4, can be varied. Analogously, the angles p_{11} and p_{12} are the inner-blade angles for the upper and lower edges of the blade at the input, respectively.

In Fig. 10c the points 1, 2, and 3 form the upper spline, and the points 4, 5, and 6, the lower spline. Between the points 1 and 6 is the point 7, which defines the spline of the input shape of the blade. In this figure, the points 1, 2, 5, and 6 have the parameters p_{13}, p_{14}, p_{15}, and p_{16}, respectively, describing their heights.

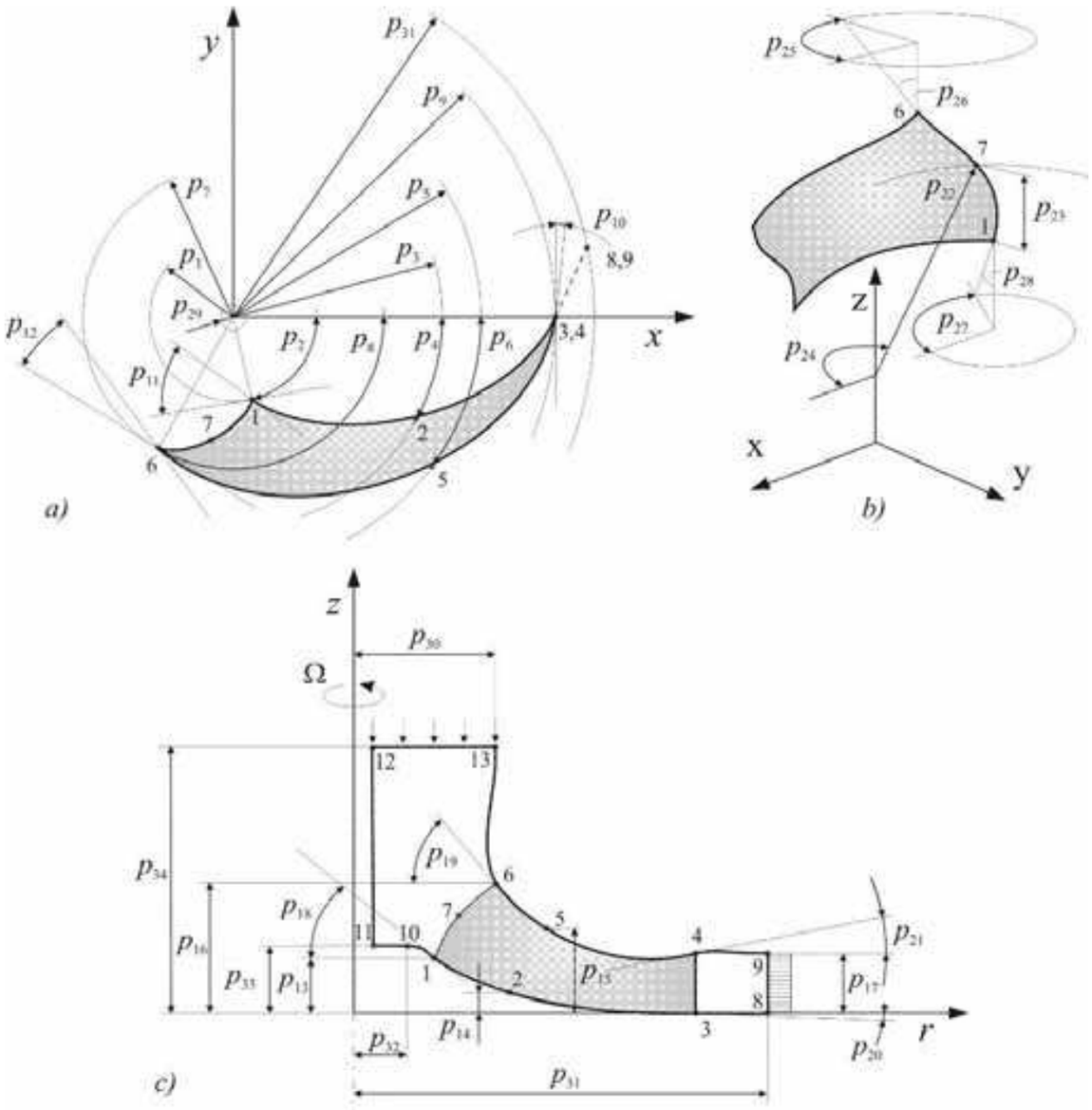

Fig. 10. Parametric modeling: a) top view; b) 3D view, c) side view

Point 3 stays on the x axis and point 4 has a constant height p_{17}. In other words, the designer of the impeller must know at least the outer diameter p_9 and the height p_{17}. The parameters p_{18} and p_{19} describe the input angles of the lower and upper parts of the blade with respect to the $r-z$ plane. Similarly, the parameters p_{20} and p_{21} describe the outer blade angle with respect to the same plane.

In Fig. 10b the meaning of point 7 is explained more precisely. The parameters p_{22}, p_{23}, and p_{24} define the radius, height, and angle, respectively. The radius and angle dictate where the point should appear with respect to the $x-y$ plane and the height with respect to the $r-z$ plane. Similarly, the angles p_{25}, p_{26}, p_{27}, and p_{28} are needed to define the starting and ending angles of the spline constructed between the points 1, 7, and 6.

If we look closely at Fig. 10c then we can see the contour surrounding the blade. This is the air channel with the following parameters: the inner radius p_{29} (see Fig. 10a), which is needed for the hexahedral mesh (explained later), the air intake radius p_{30}, the air outflow radius p_{31}, the bolt radius p_{32}, the bolt height p_{33}, and the impeller height p_{34}.
In this way we have successfully modeled the impeller geometry with 34 parameters. For each parameter we have a predefined search interval with a given discrete step. Therefore, the size of the search space can be obtained as the product of the number of possible settings over all the parameters. It turns out that there are approximately $3 \cdot 10^{34}$ possible solutions.
The mesh of the air channel between the two blades is constructed with more than 6,000 hexahedral elements. The boundary conditions are zero velocity at all the solid regions and symmetry boundary conditions at the fluid regions. At the influx and outflux the intake velocity, v_{in}, and reference pressure, p_{ref}, are defined, respectively. The intake velocity is parabolically distributed, because we expect that the intake flow is laminar and so:

$$v_{\text{in}} = v(\Phi(t))\frac{6r}{r_{\text{up}}}\left(\frac{r}{r_{\text{up}}} - 1\right). \tag{29}$$

Here, $v(\Phi(t))$ is a velocity dependent on the stream, which further depends on time, as we shall see later, r_{up} is the upper radius, defined before, and r is the radius within the limits from r_s to r_{up}. The reference pressure, p_{ref}, is set to zero.
With respect to the maximum time, $t_{\max}$, the flux is:

$$\Phi(t) = vA_{\text{in}}\frac{t}{t_{\max}}, \tag{30}$$

where A_{in} is the influx area and t is the current time.

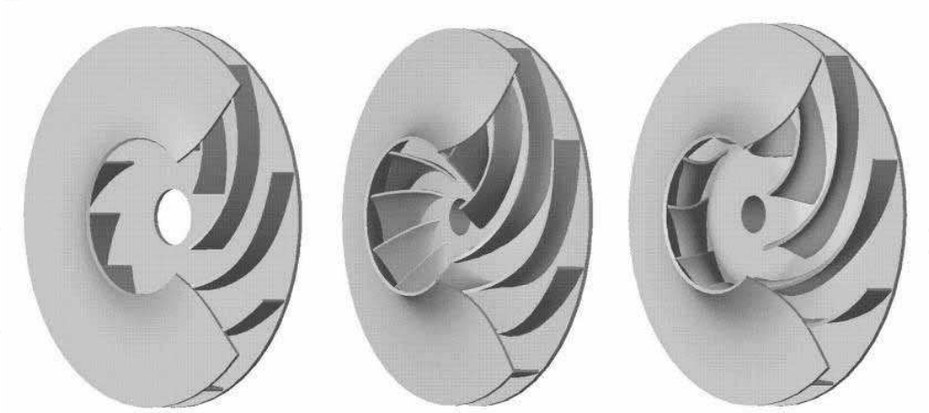

Fig. 11. The existent impeller (left), the worst optimized impeller (middle), and the best optimized impeller (right)

The distribution of the relative pressure can be used to estimate the cost function. The average pressure, p_{in}, is chosen from the air-intake area. Finally, the aerodynamic power, which represents the cost function, is as follows:

$$f_c(\boldsymbol{p}) = P_{\text{air}} = (p_{\text{out}} - p_{\text{in}})\Phi(t_{\text{opt}}), \tag{31}$$

where p_{out} is the output pressure at the radius r_{out} and $\Phi(t_{opt}) = 40\,l/s$ is the flux near the desired optimum performance of the impeller. Our goal is to find such parameter-value settings, where P_{air} is maximized.

The DASA was run 10 times and each run consisted of 2,000 CFD calculations. For the CFD calculations we used the ANSYS Multiphysics package. A single CFD calculation takes approximately seven minutes. The obtained results, in terms of aerodynamic power, are presented statistically in Table 3.

Existing solution	Optimized solutions		
	Worst	Mean	Best
411.00	432.00	472.00	483.00

Table 3 Optimized impeller's aerodynamic power in watts after 2,000 CFD calculations

Results show that we were able to increase aerodynamic power for approximately 20 %. Figure 11 shows a 3D view of the existent and two optimized impellers (best and worst of 10 runs).

4. Discussion

In this chapter the Differential Ant-Stigmergy Algorithm (DASA) was presented, where the main goal was an evaluation of the DASA on some real-world engineering applications. Case studies were selected from a R&D project where more efficient turbo-compressor for dry vacuum cleaner was developed. Here the DASA was used to improve the efficiency of an electric motor, increase casing stiffness, and increase impeller's aerodynamic power. In all these cases the improvement was evident.

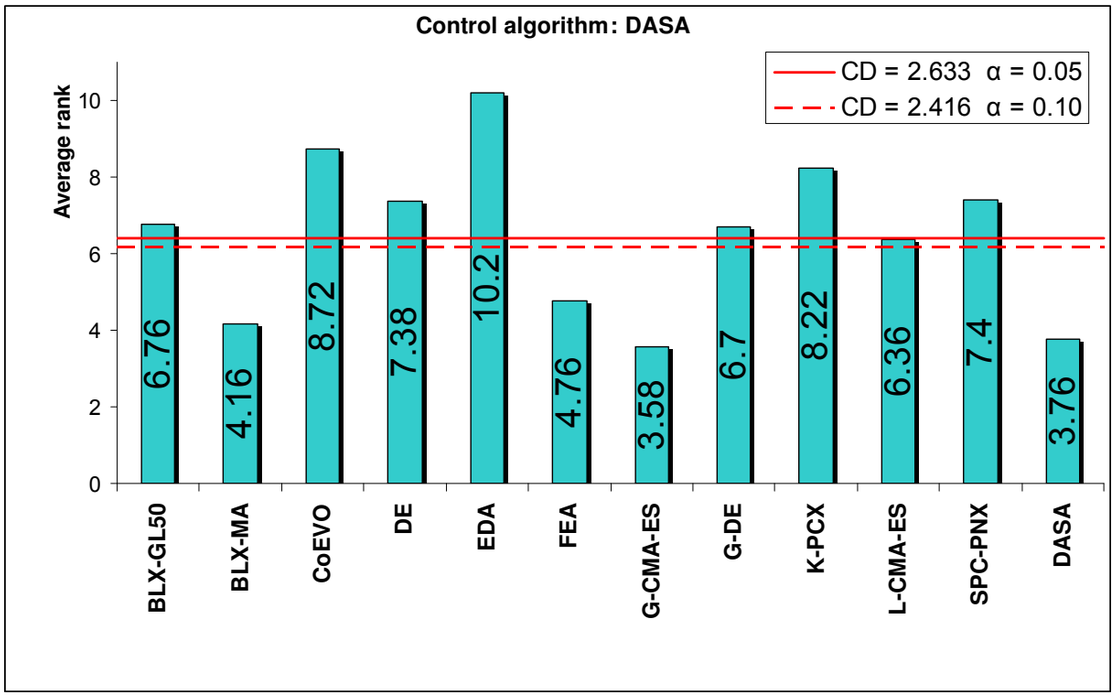

Fig. 12. The Bonferroni-Dunn post-hoc statistical test

In (Korošec & Šilc, 2009a) we also compared the DASA to the eleven state-of-the-art stochastic algorithms for continuous optimization which were presented in the CEC'2005 *Special Session on Real Parameter Optimization*. The algorithms are:

- BLX-GL50, a hybrid real-coded genetic algorithm with female and male differentiation (García-Martínez & Lozano, 2005),
- BLX-MA, a real-coded memetic algorithm with adaptive local search parameters (Molina et al., 2005),
- CoEVO, an evolutionary algorithm with mutation step co-evolution (Pošík, 2005),
- DE, a differential evolution (Rönkkönen et al., 2005),
- EDA, a continuous estimation of distribution algorithm (Yuan & Gallagher, 2005),
- FEA, a flexible evolutionary algorithm (Alonso et al., 2005),
- G-CMA-ES, a restart covariance matrix adaptation evolutionary strategy with increasing population size (Auger & Hansen, 2005a),
- G-DE, a guided differential evolution (Bui et al., 2005),
- K-PCX, a population-based steady-state procedure (Sinha et al., 2005),
- L-CMA-ES, an advanced local search evolutionary algorithm (Auger & Hansen, 2005b), and
- SPC-PNX, a steady-state real-parameter genetic algorithm (Ballester et al., 2005).

Obtained results confirmed that the DASA is comparable to above algorithms and therefore generally applicable to global optimization problems.

The experimental results have shown that the DASA ranks among the best algorithms for real-life-like applications. Its main advantage is in consistent problem solving which is indicated by 19 rankings in top third, 4 ranking in middle third and only 2 in bottom third.

Statistical analysis following the Bonferroni-Dunn post-hoc test with $\alpha = 0.10$ showed that the DASA is significantly better than 8 out of 11 compared algorithms.

The algorithm was also applied to dynamic optimization problems with continuous variables proposed for CEC'2009 *Special Session on Evolutionary Computation in Dynamic and Uncertain Environments* (Korošec & Šilc, 2009b). If we compare the DASA to:

- a self-adaptive differential evolution (Brest et al., 2009),
- a dynamic artificial immune algorithm (de França & Von Zuben, 2009),
- an evolutionary programming (Yu & Suganthan, 2009), and
- a clustering particle swarm optimizer (Li & Yang, 2009),

the results show that the DASA can find reasonable solutions for all of the problems. It can be seen that the DASA performs not many worse than a self-adaptive differential evolution and much better than the other three algorithms. 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.

5. References

Alonso, S.; Jimenez, J.; Carmona, H.; Galvan, B. & Winter, G. (2005). Performance of a flexible evolutionary algorithm. www.ntu.edu.sg/home/EPNSugan.

Auger, A. & Hansen, N. (2005a). A restart CMA evolution strategy with increasing population size, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1769–1776, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Auger, A. & Hansen, N. (2005a). Performance evaluation of an advanced local search evolutionary algorithm, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1777–1784, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Ballester, P. J.; Stephenson, J.; Carter, J. N. & Gallagher, K. (2005). Real-parameter optimization performance study on the CEC-2005 benchmark with SPC-PNX, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 498–505, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Bilchev, G. & Parmee, I. C. (1995). The ant colony metaphor for searching continuous design spaces, In: *Evolutionary Computing*, L. Fogarty (Ed.), 25–39, Springer, ISSN 0302-9743, Lecture Notes in Computer Science, Vol. 993.

Blum, C. (2005). Ant colony optimization: Introduction and recent trends, *Physics of Life Reviews*, Vol. 2, No. 4, (December 2005) 35325–39373, ISSN 1571-0645.

Brest, J.; Zamuda, A.; Bosković, B.; Sepesy Maučec, M. & Žumer, V. (2009) Dynamic optimization using self-adaptive differential evolution, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 415–422, ISBN 978-1-4244-2959-2, Trondheim, Norway, May 2009, IEEE.

Bui, L. T.; Shan, Y.; Qi, F. & Abbass, H. A. (2005). Comparing two versions of differential evolution in real parameter optimization, www.ntu.edu.sg/home/EPNSugan.

Cordón, O.; Herrera, F. & Stützle, T. (2002). A review on the ant colony optimization metaheuristic: Basis, models and new trends, *Mathware and Soft Computing*, Vol. 9, No. 3, (2002) 141–175, ISSN 1134-5632.

de França, F. O. & Von Zuben, F. J. (2009) A dynamic artificial immune algorithm applied to challenging benchmarking problems, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 423–430, ISBN 978-1-4244-2959-2, Trondheim, Norway, May 2009, IEEE.

Dorigo, M. (1992). Optimization, learning and natural algorithms, Ph.D.Thesis, Politecnico di Milano, Italy.

Dorigo, M.; Bonabeaub, E. & Theraulaz, G. (2000). Ant algorithms and stigmergy, *Future Generation Computer Systems*, Vol. 16, No. 8, (June 2000) 851–871, ISSN 0167-739X.

Dorigo, M. & Stützle, T. (2004). *Ant Colony Optimization*, The MIT Press, ISBN 978-0-262-04219-2, Cambridge, Massachusetts.

Dražumerič, R. & Kosel, F. (2005). Optimization of geometry for lateral buckling process of a cantilever beam in the elastic region, Thin-Walled Structures, Vol. 43, No. 3, (March 2005) 515–529, ISSN 0263-8231.

Dréo, J. & Siarry, P. (2002). A new ant colony algorithm using the heterarchical concept aimed at optimization of multiminima continuous functions, In: *Ant Algorithms*, M. Dorigo et al. (Eds.), 216–227, Springer, ISSN 0302-9743, Lecture Notes in Computer Science, Vol. 2463.

García-Martínez, C. & Lozano, M. (2005). Hybrid real-coded genetic algorithm with female and male differentiation, *Proceedings of the IEEE Congress on Evolutionary*

Computation, pp. 896–903, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Korošec, P. (2006). Stigmergy as an approach to metaheuristic optimization, Ph.D.Thesis, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia.

Korošec, P.; Oblak, K.; Kosel, F. & Šilc, J. (2007). Multi-parameter numerical optimization of selected thin-walled machine elements using a stigmergic optimization algorithm, *Thin-Walled Structures*, Vol. 45, No. 12, (December 2007) 991–1001, ISSN 0263-8231.

Korošec, P. & Šilc, J. (2008). Using stigmergy to solve numerical optimization problems, *Computing and Informatics*, Vol. 27, No. 3, (2008) 377–402, ISSN 1335-9150.

Korošec, P. & Šilc, J. (2009a). A stigmergy-based algorithm for continuous optimization tested on real-life-like environment, In: *EvoWorkshops 2009*, M. Giacobini et. al (Eds.), 675–684, Springer, ISSN 0302-9743, Lecture Notes in Computer Science, Vol. 5484.

Korošec, P. & Šilc, J. (2009b). The differential ant-stigmergy algorithm applied to dynamic optimization problems, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 407–414, ISBN 978-1-4244-2959-2, Trondheim, Norway, May 2009, IEEE.

Li, C & Yang, S. (2009), A clustering particle swarm optimizer for dynamic optimization, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 439–446, ISBN 978-1-4244-2959-2, Trondheim, Norway, May 2009, IEEE.

Molina, D.; Herrera, F. & Lozano, M. (2005). Adaptive local search parameters for real-coded memetic algorithms, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 888–895, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Pošík, P. (2005). Real-parameter optimization using the mutation step co-evolution, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 872–879, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Rönkkönen, J.; Kukkonen, S. & Price, K. V. (2005). Real-parameter optimization with differential evolution, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 506–513, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Sen, P. C. (1997). *Principles of Electric Machines and Power Electronics*, 2nd edition, John Wiley & Sons, ISBN 978-0-471-02295-4, New York.

Sinha, A.; Tiwari, S. & Deb, K. (2005). A population-based, steady-state procedure for real-parameter optimization, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 514–521, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Socha, K. (2004). ACO for continuous and mixed-variable optimization, In: Ant Colony Optimization and Swarm Intelligence, M. Dorigo et al. (Eds.), 25–36, Springer, ISSN 0302-9743, Lecture Notes in Computer Science, Vol. 3172.

Tsutsui, S. (2006). An enhanced aggregation pheromone system for real-parameter optimization in the ACO metaphor, In: *Ant Colony Optimization and Swarm Intelligence*, M. Dorigo et al. (Eds.), 60–71, Springer, ISSN 0302-9743, Lecture Notes in Computer Science, Vol. 4150.

Tušar, T.; P. Korošec, P.; Papa, G.; Filipič, B. & Šilc, J. (2007). A comparative study of stochastic optimization methods in electric motor design, *Applied Intelligence*, Vol. 27, No. 2, (2007) 101–111, ISSN 0924-669X.

Yu, E. L. & Suganthan, P. (2009) Evolutionary programming with ensemble of external memories for dynamic optimization, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 431–438, ISBN 978-1-4244-2959-2, Trondheim, Norway, May 2009, IEEE.

Yuan, B. & Gallagher, M. (2005). Experimental results for the special session on real-parameter optimization at CEC 2005: A simple, continuous EDA, *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1792–1799, ISBN 0-7803-9363-5, Edinburgh, UK, September 2005, IEEE.

Evolutionary Computation

Edited by Wellington Pinheiro dos Santos

ISBN 978-953-307-008-7
Hard cover, 572 pages
Publisher InTech
Published online 01, October, 2009
Published in print edition October, 2009

This book presents several recent advances on Evolutionary Computation, specially evolution-based optimization methods and hybrid algorithms for several applications, from optimization and learning to pattern recognition and bioinformatics. This book also presents new algorithms based on several analogies and metafores, where one of them is based on philosophy, specifically on the philosophy of praxis and dialectics. In this book it is also presented interesting applications on bioinformatics, specially the use of particle swarms to discover gene expression patterns in DNA microarrays. Therefore, this book features representative work on the field of evolutionary computation and applied sciences. The intended audience is graduate, undergraduate, researchers, and anyone who wishes to become familiar with the latest research work on this field.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Peter Korosec and Jurij Silc (2009). Applications of the Differential Ant-Stigmergy Algorithm on Real-World Continuous Optimization Problems, Evolutionary Computation, Wellington Pinheiro dos Santos (Ed.), ISBN: 978-953-307-008-7, InTech, Available from: http://www.intechopen.com/books/evolutionary-computation/applications-of-the-differential-ant-stigmergy-algorithm-on-real-world-continuous-optimization-probl

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen