Data Mining-Assisted Parameter Tuning of a Search Algorithm

Jurij Šilc
Computer Systems Department, Jožef Stefan Institute, Jamova cesta 39, SI-1000 Ljubljana, Slovenia
E-mail: jurij.silc@ijs.si

Katerina Taškova
Institute of Computer Science, Johannes Gutenberg University Mainz, Staudingerweg 9
55128 Mainz, Germany
E-mail: ktaskova@uni-mainz.de

Peter Korošec
Computer Systems Department, Jožef Stefan Institute, Jamova cesta 39, SI-1000 Ljubljana, Slovenia, and
Faculty of Mathematics, Science and Information Technologies, University of Primorska
Glagoljaška 8, SI-6000 Koper, Slovenia
E-mail: peter.korosec@ijs.si

Keywords: data mining, differential ant-stigmergy algorithm, low-discrepancy sequences, meta-heuristic optimization, parameter tuning

Received: December 1, 2014

The main purpose of this paper is to show a data mining-based approach to tackle the problem of tuning the performance of a meta-heuristic search algorithm with respect to its parameters. The operational behavior of typical meta-heuristic search algorithms is determined by a set of control parameters, which have to be fine-tuned in order to obtain a best performance for a given problem. The principle challenge here is how to provide meaningful settings for an algorithm, obtained as result of better insight in its behavior. In this context, we discuss the idea of learning a model of an algorithm behavior by data mining analysis of parameter tuning results. The study was conducted using the Differential Ant-Stigmergy Algorithm as an example meta-heuristic search algorithm.

Povzetek: Osnovni namen članka je pokazati, kako se lahko z uporabo tehnik podatkovnega rudarjenja lotevamo problema uglaševanja sposobnosti metahevristčnega iskalnega algoritma z vidika njegovih parametrov. Delovanje značilnega metahevrističnega iskalnega algoritma je določeno z naborom njegovih krmilnih parametrov, ki morajo biti za dosego najboljših sposobnosti pri danem problemu dobro uglašeni. Temeljni izziv je kako zagotoviti najboljšo nastavitev algoritma, ki bo rezultat vpogleda v njegovo vedenje. V zvezi s tem razpravljamo o ideji učenja modela za obnašanje algoritma na osnovi analize podatkovnega rudarjenja rezultatov uglaševanja njegovih parametrov. Študija je narejena z uporabo Diferencialnega algoritma s stigmergijo mravelj, kot primera metahevrističnega iskalnega algoritma.

1 Introduction

The research interest for meta-heuristic search algorithms has been significantly growing in the last 25 years as a result of their efficiency and effectiveness to solve large and complex problems across different domains [2]. The state-of-the-art nature-inspired meta-heuristic algorithms for high-dimensional continuous optimization include also algorithms inspired from the collective behavior of social organisms [14].

One such algorithm, which we will address in this paper, is the Differential Ant-Stigmergy Algorithm (DASA) initially proposed by Korošec [6], and further improved in [8]. DASA is inspired by the efficient self-organizing behavior of ant colonies emerging from a pheromone-mediated communication, known as stigmergy [3]. One of the first stigmergy-based algorithms designed for continuous function optimization was Multilevel Ant Stigmargy Algorithm [7].

Naturally, DASA can be classified within the Ant Colony Optimization (ACO) framework. However, the use of a continuous pheromone model in the form of Cauchy probability distribution with representation of the search space in the form of the so-called differential graph makes DASA dissimilar from the original ACO paradigm. The rationale behind DASA is in memorizing (via the pheromone model updates) the "move" in the search space that improves the current best solution and using it in further search. As it is the case with most of the meta-heuristic algorithms, the operational behavior of DASA is determined by a set of control parameters. In practice, these parameters have to be fine-tuned in order to obtain best performance of the algorithm. It can be quite inconvenient for the users as:

– they usually do not have insight into the behavior of the algorithm, and

– even if a default setting exists, it may not be adequate for a specific instance or type of a problem. Moreover, parameter tuning is computationally expensive task.

The principle challenge here is how to provide meaningful (default) settings for DASA, obtained as result of better insight into the algorithm's behavior. Furthermore, can we find optimal regions in DASA parameter space by analyzing the patterns in the algorithm's behavior with respect to the problem characteristics? Related to this, we discuss the preliminary findings based on data mining analysis of parameter tuning results. More precisely, the parameter tuning task is approached by two-step procedure that combines a kind of experimental design with data mining analysis.

We use Sobol' sequences [10] for even sampling of the algorithm parameter space to generate a large and diverse set of parameter settings. These are used as input to DASA to be tuned on a given function optimization problem. The performance of DASA on the given function optimization problem, in terms of function error, is captured at different time points for all sampled parameter settings. The data collected in the first step, DASA performance with corresponding parameter settings, is subject for intelligent data analysis, i.e., multi-target regression with Predictive Clustering Trees [1].

Parameter sampling combined with regression has been already used by Stoean et al. [11] for tuning metaheuristics: Latin hypercubes parameter sampling is combined with single-target regression with Support Vector Machines. Our approach modifies the former by replacing the Latin hypercube sampling by Sobol' sequences, as the former is best suited in the case when a single parameter dominates the algorithm's performance, while it should be used with care if there are interactions among the sampled parameters [9]. Moreover, we define the regression task as multi-target regression, taking into account more than one target (in this case the function error at few time points) with the goal to find parameter settings for the given algorithm that will not only solve the problem but will also solve the optimization problem fastest.

The reminder of this paper is structured as follows. Section 2 introduces the differential ant-stigmergy algorithm. Then, Section 3 addresses the parameter tuning task and Section 4 presents the experimental evaluation with the results. After that, Section 5 discusses the idea of post-hoc analysis of parameter tuning by data mining. Finally, Section 6 summarizes this study and outlines possible directions for further work.

2 The Differential Ant-Stigmergy Algorithm

The version of DASA used in our experimental evaluation is described in details by Korošec et al. [8] (see Figure 1).

DASA introduces the concept of variable offsets (referred as to parameter differences) for solving the continuous optimization problems. By utilizing discretized offsets of the real-valued problem parameters, the continuous optimization problem is transformed to a graph-search problem. More precisely, assuming a multidimensional parameter space with x_i being the current solution for the i-th parameter, we define new solutions x_i' as follow:

$$x_i' = x_i + \omega\delta_i, \tag{1}$$

where δ_i is called the parameter difference and is selected from the following set:

$$\Delta_i = \Delta_i^- \cup \{0\} \cup \Delta_i^+, \tag{2}$$

where

$$\Delta_i^- = \{\delta_{i,k}^-|\ \delta_{i,k}^- = -b^{k+L_i-1}, k = 1, 2, \ldots, d_i\} \tag{3}$$

and

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

Here,

$$d_i = U_i - L_i + 1, \tag{5}$$

$$L_i = \lfloor \log_b(\epsilon_i) \rfloor, \tag{6}$$

$$U_i = \lfloor \log_b(\max(x_i) - \min(x_i)) \rfloor, \tag{7}$$

$i = 1, 2, \ldots, D$, D is dimension of the problem, b is the discretization base, ϵ is the maximal computer arithmetic's precision, and the weight $\omega = \texttt{Random\_Integer}(1, b - 1)$ is added to enable a more flexible movement over the search space.

In principle, DASA relies on two distinctive characteristics, differential graph and continuous pheromone model. Here, we will briefly discuss these two characteristics and outline the main loop of the DASA search process.

First, DASA transforms the D-dimensional optimization problem into a graph-search problem. The corresponding differential graph is a directed acyclic graph obtained by fine discretization of the continuous parameters' offsets. The graph has D layers with vertices, where each layer corresponds to a single parameter. Each vertex corresponds to a parameter offset value that defines a change from the current parameter value to the parameter value in the next search iteration. Furthermore, each vertex in a given layer is connected with all vertices in the next layer. The set of possible vertices for each parameter depends on the parameter's range, the discretization base and the maximal computer arithmetic's precision, which defines the minimal possible offset value. Ants use these parameters' offsets to navigate through the search space. At each search iteration, a single ant positioned at a certain layer moves to a specific vertex in the next layer, according to the amount of pheromone deposited in the graph vertices belonging to this layer.

1: $\vec{x}^{\text{tbest}} = \texttt{Rnd\_Solution}()$
2: $y^{\text{best}} = f(\vec{x}^{\text{tbest}})$
3: $y^{\text{tbest}} = \text{inf}$
4: $\mathcal{G} = \texttt{Graph\_Initialization}(b, \vec{L}, \vec{U}, \vec{\epsilon})$
5: $\texttt{Pheromone\_Initialization}(\mathcal{G})$
6: **while** terminating condition is not met **do**
7: $k = 0$
8: **for all** m ants **do**
9: **repeat**
10: $\vec{p}_i = \texttt{Find\_Path}(\mathcal{G})$ {path of the i-th ant}
11: $k = k + 1$
12: **if** $k > m^2$ **then**
13: $\vec{x}^{\text{tbest}} = \texttt{Rnd\_Solution}()$
14: $y^{\text{best}} = f(\vec{x}^{\text{tbest}})$
15: $\texttt{Pheromone\_Initialization}(\mathcal{G})$
16: **goto** line 7 {a local optimum was found, so the search process is restarted}
17: **end if**
18: **until** ($\vec{p}_i = \mathbf{0}$) {means of all parameters' offsets are 0}
19: $\omega = \texttt{Random\_Integer}(1, b - 1)$
20: $\vec{x}_i = \vec{x}^{\text{tbest}} + \omega\delta(\vec{p})$
21: **end for**{ants created solutions}
22: $y^{\text{cbest}} = \text{inf}$
23: **for all** m ants **do**
24: $y = f(\vec{x}_i)$ {function evaluation}
25: **if** $y < y^{\text{cbest}}$ **then**
26: $y^{\text{cbest}} = y$
27: $\vec{p}^{\text{cbest}} = \vec{p}_i$
28: $\vec{x}^{\text{cbest}} = \vec{x}_i$
29: **end if**
30: **end for** {created solutions were evaluated}
31: **if** $y^{\text{cbest}} < y^{\text{tbest}}$ **then**
32: $y^{\text{tbest}} = y^{\text{cbest}}$
33: $\vec{x}^{\text{tbest}} = \vec{x}^{\text{cbest}}$
34: $s = \texttt{Update\_Scales}(s_{\text{global}}, s_{\text{local}})$
35: $\texttt{Pheromone\_Redistribution}(\vec{p}^{\text{cbest}}, s)$
36: **if** $y^{\text{tbest}} < y^{\text{best}}$ **then**
37: $y^{\text{best}} = y^{\text{tbest}}$
38: $\vec{x}^{\text{best}} = \vec{x}^{\text{tbest}}$
39: **end if**
40: **else**
41: $\texttt{Update\_Scale}(s_{\text{global}})$
42: $\texttt{Pheromone\_Evaporation}(\mathcal{G}, \rho)$
43: **end if**
44: **end while**

Figure 1: The Differential Ant-Stigmergy Algorithm

Second, DASA performs pheromone-mediated search that involves best-solution-dependent pheromone distribution. The amount of pheromone is distributed over the vertices according to the Cauchy Probability Density Function (CPDF) [9]. DASA maintains a separate CPDF for each parameter. Initially, all CPDFs are identically defined by a location offset set to zero and a scaling factor set to one. As the search process progresses, the shape of the CPDFs changes: CPDFs shrink and stretch as the scaling factor decreases and increases, respectively, while the location offsets move towards the offsets associated with the better solutions. The search strategy is guided by three user-defined real positive factors: the global scale increase factor, s_+, the global scale decrease factor, s_-, and the pheromone evaporation factor, ρ. In general, these three factors define the balance between exploration and exploitation in the search space. They are used to calculate the values of the scaling factor and consequently influence the dispersion of the pheromone and the moves of the ants.

Finally, the main loop of DASA consists of an iterative improvement of a temporary-best solution, performed by searching appropriates paths in the differential graph. The search is carried out by m ants, all of which move simultaneously from a starting vertex to the ending vertex at the last level, resulting in m constructed paths. Based on the found paths, DASA generates and evaluates m new candidate solutions. The best among the m evaluated solutions is preserved and compared to the temporary-best solution. If it is better than the temporary-best solution, the latter is replaced, while the pheromone amount is redistributed along the path corresponding to the path of the preserved solution and the scale factor is accordingly modified to improve the convergence. If there is no improvement over the temporary-best solution, then the pheromone distributions stay centered along the path corresponding to the temporary-best solution, while their shape shrinks in order to enhance the exploitation of the search space. If for some fixed number of tries all the ants only find paths composed of zero-valued offsets, the search process is restarted by randomly selecting a new temporary-best solution and re-initializing the pheromone distributions.

3 Parameter Tuning

To obtain the best possible performance on a given problem, one should consider a task specific tuning of the parameter setting for the optimization algorithm used. Determining the optimal parameters is an optimization task in itself, which is extremely computationally expensive. There are two common approaches for choosing parameters values: parameter tuning and parameter control. The first approach selects the parameter settings before running the optimization algorithm (and they remain fixed while performing the optimization). The second approach optimizes the algorithm's parameters along with the problem's parameters. Here, we will focus on the first approach, parameter tuning.

A detailed discussion and survey of parameter tuning methods is given by Eiben and Smit [4]. According to this survey, one way to approach parameter tuning is by sampling methods. Sampling methods reduce the search effort by decreasing the number of investigated parameter settings as compared to the full factorial design: the basic full factorial design investigates 2^k parameter settings, subject to k parameters, each of which have 2 possible values; in the more general case, parameters can have arbitrary number of values; moreover, an increase in the number of investigated parameters means an exponential increase in the number of parameter settings to be tested. Two widely used sampling methods are Latin-squares [9] and Taguchi orthogonal arrays [12]. However, these are not the most robust sampling techniques, e.g., Latin-squares or Latin hypercube sampling is good in the case where one of the parameters dominates the algorithm's performance, while it should be used with care if there are interactions among the parameters.

Ultimately, we would like to find a sampling schema that will be able to detect the interactions among the parameters, will be independent from user-specified information regarding the particular parameter values to be considered (typical for factorial design), and will deliver small but representative sample of the parameter search space. The first two requirements are satisfied by the pure random sampling, but the last is not, as random sampling does not guarantee that the sampled values are evenly spread across the entire domain. The so-called low-discrepancy sequences were specially designed to fulfill all three requirements. Therefore, Sobol' sequences, a representative variation of low-discrepancy sequences introduced by Sobol' [10], was considered for sampling the parameter space of DASA in this study.

Sobol' sequences, sampled from a D-dimensional unit search space, are quasi-random sequences of D-tuples that are more uniformly distributed than uncorrelated random sequences of D-tuples. These sequences are neither random nor pseudo-random: they are cleverly generated not to be serially uncorrelated by taking into account which tuples in the search space have already been sampled. For a detailed explanation and overview of the schemas for generating Sobol' sequences, we refer to [9]. The particular implementation of Sobol' sampling used in our analysis is based on the Gray code order [5].

4 Experimental Evaluation

Since data mining methods can only discover patterns actually present in the data, the dataset subject to analysis must be large enough and informative enough to contain these patterns, i.e., to describe different types of algorithm's behavior. For this reason, we decided to use a simple test function, which matched this requirement and was used for building an example case model.

Table 1: Parameter settings for DASA* and DASA°

Algorithm	DASA°		DASA*	
Parameter	$D = 20$	$D = 40$	$D = 20$	$D = 40$
m	10	10	5	7
ρ	0.2	0.2	0.324	0.388
s_+	0.02	0.02	0.201	0.136
s_-	0.01	0.01	0.289	0.344
b	10	10	6	8

Table 2: Median values of the function errors for the Sphere function

Algorithm	DASA°		DASA*	
FEs	$D = 20$	$D = 40$	$D = 20$	$D = 40$
$25 \times D$	16.7	18.3	2.53	9.31
$250 \times D$	0.0003	0.0021	0	0
$2500 \times D$	0	0	0	0
$25000 \times D$	0	0	0	0

Therefore, the performance of DASA was evaluated on the Sphere function:

$$f(x) = |z|^2 + f(x_{\text{opt}}), \tag{8}$$

where $z = x - x_{\text{opt}}$ and x_{opt} is optimal solution vector, such that $f(x_{\text{opt}})$ is minimal. Function $f(x)$ is defined over D-dimensional real-valued search space x and is scalable with the dimension D. It has no specific value of its optimal solution (it is randomly shifted in x-space) and has an artificially chosen optimal function value (it is randomly shifted in f-space). In this study, we considered the Sphere function with respect to two dimensions, $D = 20$ and $D = 40$.

The performance of DASA is dependent on the values of five parameters: three real-valued parameters that directly influence the search heuristic (s_+, $s-$, and ρ) and two integer-valued parameters (m and b). Therefore, we considered all of them for tuning DASA performance on the Sphere function for both search space dimensions, $D = 20$ and $D = 40$. Using the Gray-code-based Sobol' generator we generated 5000 parameter settings (5-tuples). Note that the Sobol' sampling generates numbers on the unit interval: in order to obtain the true parameter settings, we mapped these values on the predefined search range of parameter values. The latter for each of the five tuned parameters was defined as follows: $4 \leq m \leq 200$, $0 \leq \rho \leq 1$, $0 \leq s_+ \leq 1$, $0 \leq s_- \leq \rho$, and $2 \leq b \leq 100$. Moreover, the mapped values for the integer-valued parameters m and b were rounded to the closest integer value. Finally, due to implementation reasons, the upper bound of the global scale decrease factor s_- was actually limited by the value of the evaporation factor ρ.

In the next step, the performance of the Sobol' sampled parameter settings were tested on the Sphere benchmark function. Due to the stochastic nature of DASA, every parameter setting was used in a multiple-run experimental evaluation. Each run included $25000 \times D$ function evaluations (FEs). The number of runs was set to 15. The results gathered by the parameter tuning process are most often subjected to ordinal data analysis, which includes ranking of the different sampled parameter sets according to some calculated statistics, e.g., best or mean performance of the algorithm in some predefined number of runs [13]. In this case, performance of the algorithm is expressed in terms of the function error, i.e., the difference between the obtained and optimal function value. In order to find a setting that will be satisfactory in terms of convergence speed, we captured the error values at four different time points, corresponding to $25 \times D$, $250 \times D$, $2500 \times D$, and $25000 \times D$ FEs.

The optimal performing parameter setting was chosen based on the median best performance over all runs aggregated over all time points for a given dimension ($D = 20$ and $D = 40$). A common approach is to use the mean performance, but we took the median in order to avoid the problems that the mean has when observing large variance in the function values across the runs. More precisely, given a function, an individual rank is assigned to every setting (out of 5000) for the four time points. A single final rank is calculated by ranking the sum of the four individual rankings assigned to the parameter settings. The best-ranked parameter setting for a given dimension defines instance of DASA referred to as DASA*.

The results of DASA tuning subject to ordinal data analysis are presented in Tables 1 and 2. Table 1 reports the tuned parameter settings for both DASA* instances.

In addition, the default parameter setting for DASA from [8] is given as a reference for comparison. The corresponding instance is referred to as DASA°.

Results in Table 2 represent the median values of the function errors, at the four time points, obtained by DASA* and DASA° instances for both dimensions. Note, that error value below 10^{-8} was treated as zero. The table clearly shows that DASA* is better than DASA°.

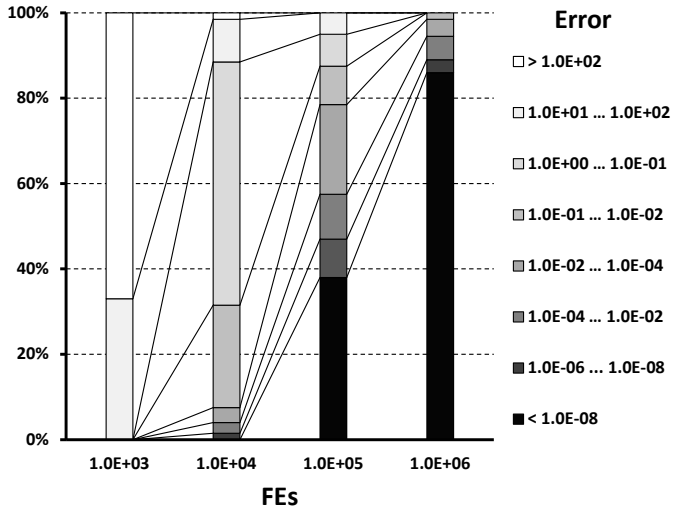

Figure 2: Median error distributions for the Sphere function in the case of $D = 40$.

5 Data Mining Analysis

Parameter tuning of an algorithm leads to a better performance, however it is a computationally expensive and problem-dependent task. Considering this, the idea is to extend the simple tuning that delivers a single parameter set and analyze the gathered data in an intelligent way. The intelligent analysis can extract patterns (regularities) in the explored parameter space that define a specific behavior of DASA. To this end, data mining methods for automated discovery of patterns in data can be used. As data mining methods can only discover patterns that are present in the data, the dataset subject to analysis must be large enough and informative enough to contain these patterns, i.e., to describe different types of algorithm's behavior. Related to this, we considered a data mining approach on a representative example, i.e., error model of the Sphere function.

To begin with, consider the graph in Figure 2 that visualizes the Sphere function error distribution. The graph depicts the distribution of the median error values obtained by 5000 parameter settings at four different points for $D = 40$. As we are more concerned with the practical significance between a large and a small error value than the statistical significant difference between two actual error values, the error values are discretized into nine intervals, each of which is represented with a color chosen according to the error magnitude between black (error below 10^{-8}) and white (error above 10^2). The graph clearly shows that the sampled settings determine different DASA performance. As evident, there is a big cluster of parameter settings that solve this function to the aimed accuracy (error below 10^{-8}) in the given time budget (10^6 FEs). Moreover, subset of this cluster solves the function for an order of magnitude less FEs. Our aim, therefore, is to find a (common) description of this cluster, in terms of DASA parameter relations, that represents a good behavior of DASA (as well as what parameter relations lead to a bad DASA performance).

For this purpose, we formulated the problem as a predictive modeling task using decision trees to model the function error values in terms of the values of DASA parameters. Since the function error variables are continuous, the task at hand is a regression task. Furthermore, as our goal is to model the behavior of DASA at all time points simultaneously, the problem at hand is then a multi-target regression. To this end, we used Predictive Clustering Trees (PCTs) which are implemented in Clus system [1]. In this case, the median error at the four time points define the four target attributes considered for modeling (with the PCT) in dependence from the descriptive attributes, i.e., the function dimension and the five DASA parameters. The resulting dataset is composed of 10002 rows described with the 5001 parameter settings (including the default setting) of DASA applied to the two different dimensions, and 10 columns corresponding to the 10 attributes.

Figure 3 presents a PCT model for the Sphere function error. Each internal node of the tree contains a test on a descriptive attribute, while the leaves store the model predictions for the function error, i.e., a tuple of four values. The predictions are calculated as the mean values of the corresponding error values for the data instances belonging to the particular leaf (represented by a box). In fact, each leaf identifies one cluster of data instances (the size of the cluster is the value in the small box). The predictive performance of the model was assessed with 10-cross-validation. Note that this particular model was learned on the complete dataset subject to constraints on the maximal tree size of 25 nodes. We did this because the original model contained 1643 nodes (of which 822 leaves) and despite its better predictive performance, both training and testing, it was not comprehensible; aiming for an explanatory model, small and comprehensible, we considered the smaller tree obtained with the limitation of the size. The predictive performance of both models in terms of Root Relative Mean Squared Error (RRMSE) and Pearson Correlation Coefficient (PCC) are given in Table 3. Note that RRMSE represents the relative error with respect to the mean predictor performance, while PCC represents the linear correlation between the data and the model predictions. Good models have RRMSE values closer to 0 and PCC closer to 1.

The model in Figure 3 outlines 13 clusters of data instances, of which two (depicted with light-gray boxes) represent a good DASA performance. According to this model, the number of ants, m, is the most important DASA parameter for its performance on the Sphere function. More precisely, if $m > 83$, independent of the values of the other parameters, DASA solves the 20-dimensional functional problem for the given time budget. Moreover, if $m \leq 83$ another DASA parameters become important as well. For example, if $43 < m \leq 83$ and $s_+ > 0.009$ and $D = 20$ then DASA solves the function with error 3×10^{-6}, while the pattern $m \leq 43$ and $s_{\leq}0.040$ and $s_- > 0.656$ describes a poor DASA performance regard-

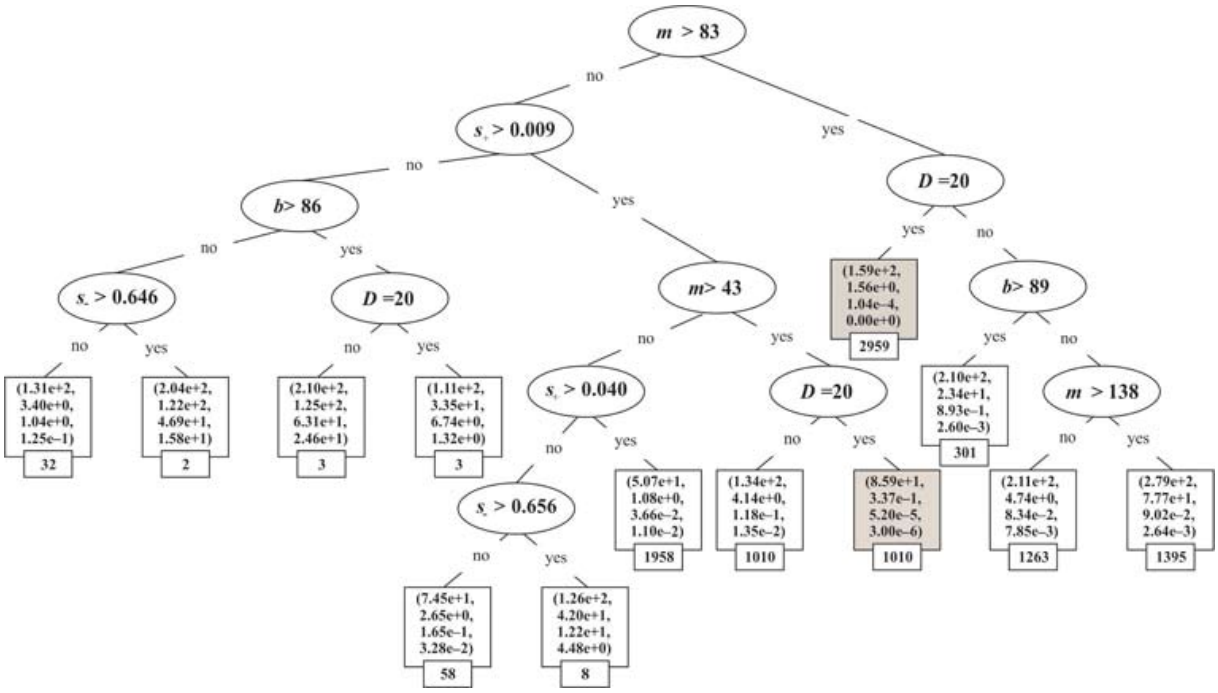

Figure 3: Predictive clustering tree representation of the error model for the Sphere function.

less of the function dimension. An interesting fact is that, the evaporation factor is not essential for DASA performance on the Sphere function. Moreover, the model also shows that is more difficult to describe the behavior of DASA for the 40-dimensional function problem than the 20-dimensional one.

Finally, note that the training performance (learned on the complete dataset) of the model in terms of the error and the correlation coefficient is best for the first target, while it gets worse with respect to the other three targets (see Table 3). This is especially significant if we take into account the testing performance of the model estimated with 10-cross-validation. However, the training performance is acceptable in our case, as we are interested in understanding the behavior of DASA and not aiming to obtain a model for prediction.

6 Conclusion

The principle challenge of meta-heuristic design is providing a default algorithm configuration, in terms of parameter setting, that will perform reasonably well in general (problem) case. However, while it is a good initial choice, the default algorithm configuration may result in low quality solutions on a specific optimization problem. In practice, the algorithms parameters have to be fine-tuned in order to obtain best algorithm's performance for the problem at hand, leading to the computational expensive task of parameter tuning. So, if the tuning task is unavoidable, the question is: can we use the results from the parameter tuning to extract some knowledge about the algorithm's behavior?

Related to this, the study focused on the problem of tuning the performance of the Differential Ant-Stigmergy Algorithm (DASA) with respect to its parameters. As it is the case with most of the meta-heuristic algorithms, the operational behavior of DASA is determined by a set (five) of control parameters. The existing default setting of DASA parameters [8] is obtained by experimentation with both real and benchmark optimization problems, but not as a result of some systematic evaluation. Furthermore, there is no deeper understanding of the impact of a particular parameter or parameters relations on the performance of DASA. In this context, we performed a systematic evaluation of DASA performance obtained by solving the Sphere function optimization problem with 5000 Sobol' sampled DASA parameter settings regarding two dimensions, 20 and 40.

Furthermore, we discussed the idea of learning a model of DASA behavior by data mining analysis of the parameter tuning results. In this context, we formulated the problem as multi-target regression and applied predictive clustering trees for learning a model of DASA behavior with respect to the function error performance. The obtained model revealed that the parameter denoting number of ants is the most important parameter for DASA performance on the 20-dimensional function problem. On the other hand, the evaporation factor is not essential for DASA performance on the Sphere function.

Further work will focus on additional experimental evaluation and data mining analysis of data with respect to more complex functions problems. This idea can be fur-

Table 3: Model performance with respect to RRMSE and PCC

Measure	RRMSE		PCC	
Training	1643 nodes	25 nodes	1643 nodes	25 nodes
$25 \times D$	0.166	0.408	0.986	0.913
$250 \times D$	0.373	0.679	0.928	0.734
$2500 \times D$	0.487	0.562	0.873	0.827
$25000 \times D$	0.472	0.546	0.882	0.837
Mean	0.396	0.557	0.843	0.689
Testing	1643 nodes	25 nodes	1643 nodes	25 nodes
$25 \times D$	0.219	0.450	0.976	0.893
$250 \times D$	0.638	0.817	0.777	0.584
$2500 \times D$	1.019	1.039	0.304	0.246
$25000 \times D$	1.053	1.055	0.234	0.218
Mean	0.806	0.875	0.426	0.312

ther extended to building models of DASA behavior that will include the optimization problem characteristics (such as multimodality, separability, and ill-conditioning) as descriptive attributes as well. The latter can provide insights on how to configure DASA performance with respect to the type of the optimization problem. Moreover, these insights can serve as a valuable information for improvement of DASA design.

References

[1] H. Blockeel, J. Struyf (2002) Efficient Algorithms for Decision Tree Cross-validation, *Journal of Machine Learning Research*, vol. 3, pp. 621–650.

[2] C. Blum, A. Roli (2003) Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison, *ACM Computing Surveys*, vol. 35, no. 3, pp. 268–308.

[3] E. Bonabeau, M. Dorigo, G. Theraulaz (1999) *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press.

[4] A. E. Eiben, S. K. Smit (2011) Parameter Tuning for Configuring and Analyzing Evolutionary Algorithms, *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 19–31.

[5] S. Joe, F. Y. Kuo (2008) Constructing Sobol Sequences with Better Two-dimensional Projections, *SIAM Journal on Scientific Computing*, vol. 30, no. 5, pp. 2635–2654.

[6] P. Korošec (2006) Stigmergy as an Approach to Metaheuristic Optimization, Ph.D. dissertation, Jožef Stefan International Postgraduate School, Ljubljana, Slovenia.

[7] P. Korošec, J. Šilc (2008) Using Stigmergy to Solve Numerical Optimization Problems, *Computing and Informatics*, vol. 27, no. 3, pp. 341–402.

[8] P. Korošec, J. Šilc, B. Filipič (2012) The Differential Ant-stigmergy Algorithm, *Information Sciences*, vol. 192, no. 1, pp. 82–97.

[9] W. H. Press, S. A. Teukolsky, W. T. Vetterling, B. P. Flannery (1992) *Numerical Recipes*, Cambridge University Press.

[10] I. M. Sobol' (1967) Distribution of Points in a Cube and Approximate Evaluation of Integrals, *USSR Compututational Mathematics and Mathematical Physics*, vol. 7, no. 4, pp. 86–112.

[11] R. Stoean, T. Bartz-Beielstein, M. Preuss, C. Stoean (2009) A Support Vector Machine-Inspired Evolutionary Approach for Parameter Setting in Metaheuristics, CIOP Technical report 01/09, Faculty of Computer Science and Engineering Science, Cologne University of Applied Science, Germany.

[12] G. Taguchi, T. Yokoyama (1993) *Taguchi Methods: Design of Experiments*, ASI Press.

[13] E.-G. Talbi (2009) *Metaheuristics: From Design to Implementation*, John Wiley & Sons.

[14] X.-S. Yang (2008) *Nature-Inspired Metaheuristic Algorithms*, Luniver Press.