COMPARISON OF OPTIMIZATION METHODS EFFICIENCY IN DIFFERENT DIMENSIONS OF THE SEARCH SPACE OF SIMULATION MODELS

Pavel Raska[(a)], Zdenek Ulrych[(b)]

[(a)] Department of Industrial Engineering - Faculty of Mechanical Engineering, University of West Bohemia, Univerzitni 22, 306 14 Pilsen
[(b)] Department of Industrial Engineering - Faculty of Mechanical Engineering, University of West Bohemia, Univerzitni 22, 306 14 Pilsen

[(a)]praska@kpv.zcu.cz, [(b)] ulrychz@kpv.zcu.cz,

ABSTRACT
The paper deals with the comparison of selected optimization methods - Random Search, Hill Climbing, Tabu Search, Local Search, Downhill Simplex, Simulated Annealing, Differential Evolution, SOMA and Evolution Strategy – efficiency used for optimization of four selected testing functions. We tested these optimization methods on different dimensions of the search space to compare their efficiency of finding the global optimum in the search space.

Keywords: simulation optimization, testing function, dimension of the search space

1. INTRODUCTION

Simulation optimization is one possible solution for solving an NP-hard problem. The simulation model is the modelled problem. The problem is to find suitable settings for the simulation model input parameters (input variables):

$$\mathbf{X} = \lfloor x_j \rfloor \forall j : j = \{0,1,2,\ldots,n-1\} \tag{1}$$

Where $\mathbf{X}$ denotes the concrete setting of the input parameters; x_j denotes the value of the j-th simulation model input parameter (first simulation model input parameter is indexed by 0 because the first item in the list is commonly labelled by 0 in programming language); n denotes the number of the simulation model input parameters – dimension of the search space. Each possible solution – candidate solution (the concrete setting of the input parameters $\mathbf{X}$) is a representation of the element of the search space $\tilde{X}$. The dimension of the search n space equals the number of simulation model input parameters. The search space in the case of Box Constraint can be formulated as follows:

$$\tilde{X} = \prod_{j=1}^{n-1} [a_j, b_j] \forall j : j = \{0,1,2,\ldots,n-1\} \tag{2}$$

Where a_j denotes the lower bound of the j-th simulation model input parameter; b_j denotes the lower boundary of the j-th simulation model input parameter. We should also say that we obtain one or more simulation model outputs after the simulation run (the result of simulation experiment with a concrete setting) which are inputs of the objective function. The candidate solution in the search space can be evaluated by the objective function value $F(\mathbf{X})$ which represents the quality of candidate solution regarding the specified objective. It is clear that if the number of simulation model input increases, the search space contains a large number of possible solutions.
The basic problem of finding an optimal/suboptimal feasible solution (respecting the defined model constraints) can be formulated as follows:

$$\breve{\mathbf{X}} = \arg\min_{\mathbf{X}\in\tilde{X}} F(\mathbf{X}) = \{\breve{\mathbf{X}} \in \tilde{X} : F(\breve{\mathbf{X}}) \le F(\mathbf{X}) \forall \mathbf{X} \in \tilde{X}\} \tag{3}$$

where $\breve{\mathbf{X}}$ denotes the global minimum of the objective function; $F(\mathbf{X})$ denotes the objective function value of the candidate solution – the range commonly includes real numbers; $\tilde{X}$ denotes the Search space. (Raska & Ulrych, 2013)
The next figure (Figure 1) shows the possible optimization process if the objective function is minimized. We can see that the objective function has a local and global optima if the objective function is maximized – problem of premature convergence. Premature convergence – the global optimization process can converge prematurely to a local optimum because there is no opportunity to examine other areas of a space of possible solutions (currently, only a particular area is examined). Another area of search space exists (which is not currently known) that contains a better solution than the currently known solution. (Weise, 2009). Other basic problems of the optimization methods used in global optimization are (Rockwell Automation, 2014): The whole search space cannot be examined (testing all possible solutions) because of large demands on computer memory, or time

spent on examining the space – nondeterministic polynomial problems; the landscape of objective function – multimodal objective function – premature convergence, objective function smoothness landscape etc.; multi-objective optimization; identification of a suitable method for handling the constraints; specification of appropriate termination criteria; setting the parameters of the optimization method, etc.

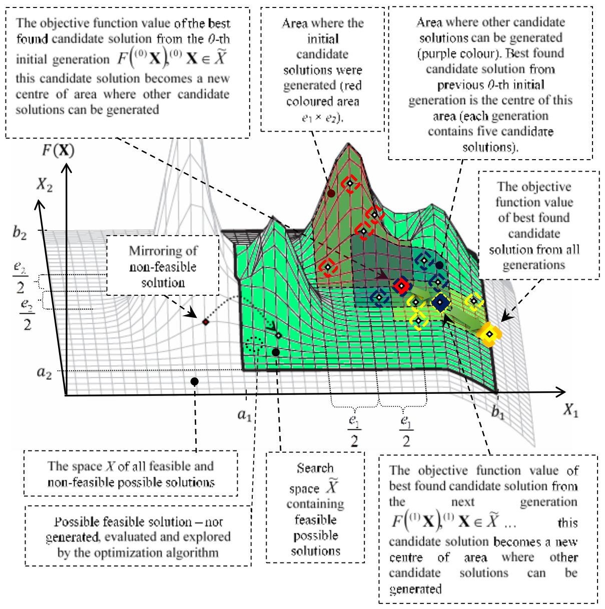
The objective function value of the best found candidate solution from the *0*-th initial generation $F\left({}^{(0)}\mathbf{X}\right), {}^{(0)}\mathbf{X} \in \widetilde{X}$ this candidate solution becomes a new centre of area where other candidate solutions can be generated
Area where the initial candidate solutions were generated (red coloured area $e_1 \times e_2$).
Area where other candidate solutions can be generated (purple colour). Best found candidate solution from previous *0*-th initial generation is the centre of this area (each generation contains five candidate solutions).
$F(\mathbf{X})$
X_2
b_2
$\frac{e_2}{2}$
$\frac{e_2}{2}$
Mirroring of non-feasible solution
The objective function value of best found candidate solution from all generations
a_2
a_1
$\frac{e_1}{2}$
$\frac{e_1}{2}$
b_1
X_1
The space X of all feasible and non-feasible possible solutions
Possible feasible solution – not generated, evaluated and explored by the optimization algorithm
Search space $\widetilde{X}$ containing feasible possible solutions
The objective function value of best found candidate solution from the next generation $F\left({}^{(1)}\mathbf{X}\right), {}^{(1)}\mathbf{X} \in \widetilde{X}$ ... this candidate solution becomes a new centre of area where other candidate solutions can be generated

Figure 1: Possible Optimization Process - Objective Function Minimization

The whole space X contains feasible and non-feasible possible solutions when some constraints are specified e.g. Box Constraints. The green area denotes the feasible possible solutions (axes in the search space are indexed from index 1, because it is not common to mark first axes by the index 0. Indexing with index 0 will be used in mathematical notation or in the algorithm). If this candidate solution is generated by the optimization method then it is accepted and evaluated by the objective function value regarding the simulation model output. There are different variants for generating candidate solutions: neighbourhood relation which allows us to define the set of feasible candidate solutions neighbouring a selected candidate solution through a series of transformations of selected candidate solution. Condition of availability is important for a definition of neighbourhood relations , which requires that every feasible solution has to be reached from any other feasible solution by sequential application of neighbourhood relation (Stefka, 2005), generating a random candidate solution using different distribution, genetic operators – e.g. mutation, crossover (Mitchell, 1996; Miranda, 2008; Hynek, 2008), etc.
The process of generating new initial candidate solutions in the specified (red) area is shown in Figure 1.
Optimization methods usually generate more than one candidate solution using the iterations in cycle. These candidate solutions are sequentially placed in the list in the order which they were generated (in the context of evolutionary algorithms this list is represented by the population - generation). This candidate solution (individual) in the population can be formulated as follows:

$$\mathbf{X}_i = L[i] \forall i : i = \{0,1,2,\ldots,m-1\} \qquad (4)$$

Where $\mathbf{X}_i$ denotes i-th candidate solution – individual; i denotes the index of candidate solution; L denotes the list of candidate solutions - population; m denotes the number of generated candidate solutions - the population size.
The optimization method selects the best candidate solution from this generation (population) regarding the objective function values of candidate solutions. This best candidate solution from the *0*-th initial generation becomes the centre of area where other candidate solutions can be generated. These processes are repeated until some of the termination criteria are satisfied. If the termination criterion is met the optimization method returns one or more best found solutions.
Optimization method can generate the non-feasible solution. Optimization method can use the repair algorithms and special operators (Hynek, 2008). Mirroring of non-feasible solution into a feasible region is shown in the next figure. The non-feasible solution is flipped into the feasible region around the respective edge of the search space at a distance of this non-feasible solution to the edge of the search space (Tvrdik, 2010).

2. SELECTED OPTIMIZATION METHODS

We have transformed some of the selected optimization methods to use the principle of evolutionary algorithms. Different variants of selected optimization methods obtained from a literature review were united into the algorithm.

2.1. Random Search

A new candidate solution is generated in the search space with uniform distribution (Monte Carlo method). This method is suitable for cases where the user has no information about the objective function type. The user is able to perform a number of simulation experiments.

2.2. Downhill Simplex

This method uses a set of n + 1 linearly independent candidate solutions (n denotes search space dimension) - Simplex. The method uses four basic phases – Reflection, Expansion, Contraction and Reduction. (Tvrdík 2004; Weise 2009)

2.3. Stochastic Hill Climbing

Candidate solutions are generated (populated) in the neighbourhood of the best candidate solution from the previous population. Generating new possible solutions is performed by mutation. This method belongs to the family of local search methods.

2.4. Stochastic Tabu Search

The newly generated candidate solution is an element of the Tabu List during the optimization process. This candidate solution cannot be visited again if the aspiration criterion is not satisfied (this feature prevents the method from becoming stuck at a local optimum). The method uses the FIFO method of removing the candidate solution from the Tabu List. The user can set whether the new candidate solution is generated using mutation of the best candidate solution from the previous population or the new solution is generated using mutation of the best found candidate solution. (Monticelli, Romero and Asada 2008; Weise 2009)

2.5. Stochastic Simulated Annealing

A candidate solution is generated in the neighbourhood of the candidate solution from the previous iteration. This generating could be performed through the mutation of a randomly selected gene or through the mutation of all genes. Acceptance of the worse candidate solution depends on the temperature. Temperature is reduced if the random number is smaller than the acceptance probability or the temperature is reduced if and only if a worse candidate solution is generated. If the temperature falls below the specified minimum temperature, temperature is set to the initial temperature. (Monticelli, Romero and Asada 2008; Weise 2009)

2.6. Stochastic Local Search

A candidate solution is generated in the neighbourhood of the best candidate solution.

2.7. Evolution Strategy

This optimization method uses Steady State Evolution – population consists of children and parents with good fitness. A candidate solution (child) is generated in the neighbourhood of the candidate solution (parent) and it is based on the Rechenberg 1/5th-rule. The population is sorted according to the objective values (Rank-Based Fitness Assignment). The optimization method uses Tournament selection. (Koblasa, Manlig and Vavruska 2013; Miranda 2008; Tvrdik 2004)

2.8. Differential Evolution

Selection is carried out between the parent and its offspring. The offspring is created through a crossover between the parent and the new candidate solution (individual) which was created through the mutation of four selected individuals and the best one selected from the population – BEST method. The optimization method uses General Evolution and the Ali and Törn adaptive rule. The user can define the probability of a crossover between the new candidate solution and the parent. (Tvrdík 2004; Wong, Dong, 2008)

2.9. SOMA

SOMA is based on the self-organizing behaviour of groups of individuals in a “social environment”. It can also be classified as an evolutionary algorithm, despite the fact that no new generations of individuals are created during the search. Only the positions of the individuals in the search space are changed during a generation, called a “migration loop”. Individuals are generated at random according to what is called the “specimen of the individual” principle. The specimen is in a vector, which comprises an exact definition of all these parameters that together led to the creation of such individuals, including the appropriate constraints of the given parameters. SOMA is not based on the philosophy of evolution (two parents create one new individual – the offspring), but on behaviour of a social group of individuals. (Zelinka, 2004)

3. TESTING FUNCTIONS

Considering the time requirements of testing the behaviour of optimization methods (according to different settings) (Raska & Ulrych, 2015) we substitute the testing on the simulation models (and its objective function) by a different testing function. Implemented optimization methods were tested on four standard testing functions - domain of the function is a defined step for each axis – substitution of the simulation model input parameter (discrete) values of the discrete event simulation model. All testing functions were minimized.

3.1. De Jong´s Function

A convex and unimodal testing function. The function definition: (Pohlheim, 2006)

$$\begin{aligned}
&F(\mathbf{X}) = \sum_{j=1}^{n} x_j^2, \\
&\forall x_j : \left(x_j - \left\lfloor \frac{x_j}{0.01} \right\rfloor \cdot 0.01 = 0 \right) \wedge \left(-30 \le x_j \le 30 \right), \\
&j = 1 : n, n \in [2,10,20,30,44]
\end{aligned} \tag{5}$$

where $F(\mathbf{X})$ denotes the objective function; j denotes index of control; n denotes the dimension of the search space - the dimension of the search space is 2; 10; 20; 30; 44; x_j denotes the value of control - testing functions (except Michalewicz function) input parameters values are in the range from -30 (lower boundary) to 30 (higher boundary). We substitute the testing on the simulation models by testing on the testing function hence we defined the smallest step that can be performed by the optimization methods is 0.01 for each axis in the search space (x_j mod 0.01=0). The input parameters are not continuous. This resolution represents 36,012,001 possible solutions (combinations of testing function input parameters) in a two dimensional search space and the same resolution represents 1.7452×10^{166} possible solutions in a forty-four dimensional search space. We chose a forty-four dimensional search space regarding the tested discrete event simulation model of automated guided vehicle transportation with forty-four simulation model input parameters to see the optimization methods' efficiency. To achieve a better idea of the testing functions landscapes the continuous testing functions are shown in the following four figures. De Jong´s continuous function is shown in Figure 2.

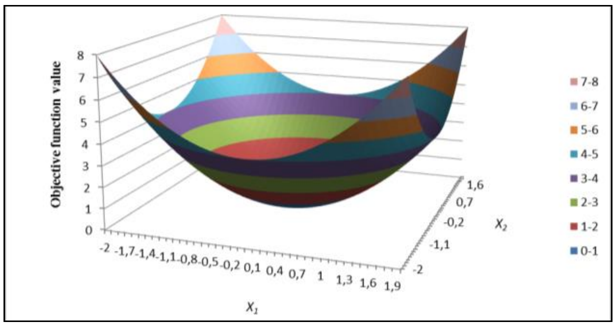

Figure 2: De Jong´s Function

3.2. Rosenbrock´s Function

Rosenbrock´s (Rosenbrock's valley, Rosenbrock's banana) function is unimodal and non-convex testing function. The function definition: (Pohlheim, 2006)

$$F(\mathbf{X}) = \sum_{j=1}^{n-1} 100 \cdot (x_j^2 - x_{j+1})^2 + (1 - x_j)^2,$$
$$\forall x_j : \left(x_j - \left\lfloor \frac{x_j}{0.01} \right\rfloor \cdot 0.01 = 0 \right) \wedge (-30 \le x_j \le 30), \quad (6)$$
$$j = 1:n, n \in [2,10,20,30,44]$$

The Rosenbrock´s continuous function is shown in Figure 3.

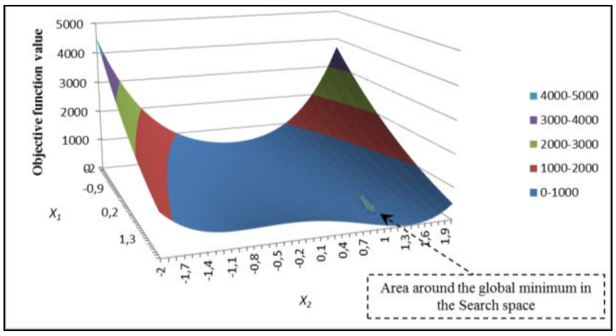

Figure 3: Rosenbrock´s Function

3.3. Michalewicz Function

Michalewicz function is a multimodal test function (n! local optima). The parameter *m* defines the "steepness" of the valleys or edges. Larger *m* leads to a more difficult search. For very large *m* the function behaves like a needle in a haystack (the function values for points in the space outside the narrow peaks give very little information on the location of the global optimum). (Pohlheim 2006)

$$F(\mathbf{X}) = -\sum_{j=1}^{n} \sin(x_j) \cdot \left(\sin\left(\frac{j \cdot x_j^2}{\pi} \right) \right)^{2 \cdot m},$$
$$\forall x_j : \left(x_j - \left\lfloor \frac{x_j}{0.01} \right\rfloor \cdot 0.01 = 0 \right) \wedge (0 \le x_j \le \pi), \quad (7)$$
$$j = 1:n, n \in [2,10,20,30,44]$$

We selected $m = 5$ in our simulation model. The Michalewicz continuous function is shown in Figure 4.

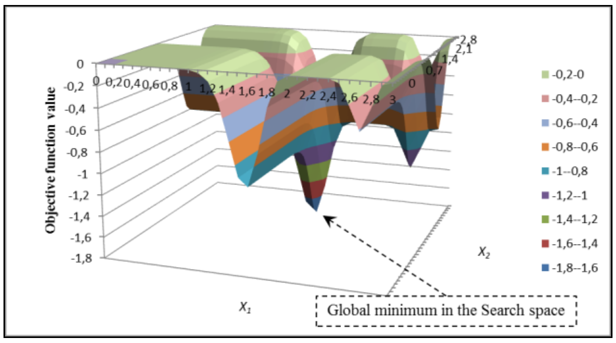

Figure 4: Michalewicz Function

3.4. Ackley´s Functions

Ackley´s function is a multimodal test function. This function is a widely used testing function for premature convergence. (Tvrdik, 2010)

$$F(\mathbf{X}) = -20 \cdot \exp\left(-0.02 \cdot \sqrt{\frac{1}{n} \cdot \sum_{j=1}^{n} x_j^2} \right) - \exp\left(\frac{1}{n} \cdot \sum_{j=1}^{n} \cos 2 \cdot \pi \cdot x_j \right) + 20 + \exp(1),$$
$$\forall x_j : \left(x_j - \left\lfloor \frac{x_j}{0.01} \right\rfloor \cdot 0.01 = 0 \right) \wedge (-30 \le x_j \le 30), \quad (8)$$
$$j = 1:n, n \in [2,10,20,30,44]$$

The Ackley´s continuous function is shown in Figure 5.

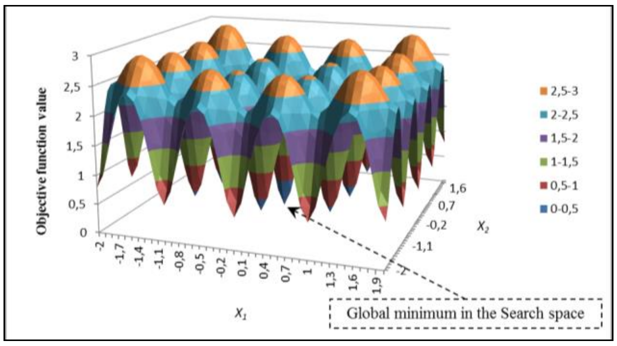

Figure 5: Ackley´s Function

4. OPTIMIZATION EXPERIMENTS

We tested optimization methods on the testing functions where the dimension of the search space was 2; 10; 20; 30; 44. We specified the same conditions which had to be satisfied for each optimization method, e.g. the same termination criteria: The optimization method could perform a maximum of 20,000×*n* (parameter *n* denotes the dimension of the search space) simulation experiments to find the global optimum in the search space (Tvrdik, 2010) or the termination criterion was met if the optimum was found - VTR (value to reach); the same search space where the optimization method can search for the global optimum, etc. If the optimization method has the same parameters as another optimization method, we set up both parameters with the same boundaries (same step, lower and upper boundaries).

We performed optimization experiments with different optimization method settings to find a suitable setting, but the analysis of these results is not the purpose of this paper.

We evaluated these optimization experiments with different settings - series. These series we replicated several times to reduce the random behaviour of the tested optimization methods. The following charts show the average optimization method success of finding optimum (suboptimum if the optimum was not found).

The first criterion f_1 is the value of not finding the known VTR (value to reach). This value is expressed by pseudopascal code:

$\mathfrak{f}_1 \longleftarrow$ AssignFitnessFindingOptimum(X^*, $\mathbf{X}^*$, ε)	
Function whose output is normalized scalar value in the range $\mathfrak{f}_1 \in [0,1]$. This value represents the failure of finding global optimum by the optimization method in a particular series – value minimization	
Input:	X^*: The list of found optima in each optimization experiment in the series
Input:	$\mathbf{X}^*$: Global optimum $\mathbf{X}^*$ in the search space
Input:	ε: Tolerated deviation from the value of the objective function value of global optimum
Data:	$F(\mathbf{X})$: Objective function value
Data:	n_{Succ}: Counter of successful finding optimum
Output:	$\mathfrak{f}_1$: Standardized scalar value

```
1   begin
2     $n_{Succ} \longleftarrow 0$;
3     for $i \longleftarrow$ 0 to Length($X^*$) − 1 do
4         if $|F(X^*[i]) - F(\mathbf{X}^*)| \le \varepsilon$ then
5             $n_{Succ} \longleftarrow n_{Succ} + 1$;
    (*Optimum or acceptable candidate solution was found *)
6     result $\longleftarrow \frac{\mathrm{Length}(X^*) - n_{Succ}}{\mathrm{Length}(X^*)}$;
    (*standardization - % share of unsuccessful series*)
7   end;
```

Figure 6: Pseudopascal Algorithm of First criterion – Finding the Global Optimum or Suboptimum

If the failure is 100[%] the first criterion equals 1 therefore we try to minimize this criterion. Average Method Success of Finding Optimum can be formulated as follows:

$$f_{avg} = \left(1 - \frac{\sum_{i=1}^{s} f_{1_i}}{s}\right) \cdot 100\,[\%] \qquad (8)$$

where i denotes the index of one series, f_{1_i} denotes the value of the first criterion (Optimization method success – the best value is zero), s denotes the number of performed series.

The series were also evaluated regarding specified tolerance between the best optimum (suboptimum) found in the series and the specified parameter ε. We initially specified $\varepsilon = 0.001$. The optimization method had to find the candidate solution whose objective function value is nearly the same as the objective function value of global optimum in the search space (the tolerance equals 0.001). This aspect is unfounded in practice. Hence we specified the tolerance to one percent of the difference between the objective function value of the global maximum and the objective function value of the global minimum of the search space:

$$\varepsilon = \frac{\left|F(\hat{\mathbf{X}}) - F(\check{\mathbf{X}})\right|}{100} \qquad (9)$$

Where ε denotes the difference between the objective function value of the global maximum and the objective function value of the global minimum of the search space; $F(\hat{\mathbf{X}})$ denotes the objective function value of the global maximum of the search space; $F(\check{\mathbf{X}})$ denotes the objective function value of the global minimum of the search space.

The following figures show the average optimization methods success of finding optimum (suboptimum – the best found candidate solution of the series). Chart values of optimization method success are maximized.

We set up the termination criterion in such a manner that the optimization method can perform a maximum of 20,000×n (parameter n denotes the dimension of the search space) simulation experiments for each series to find the global optimum in the search space. The other part of termination criterion is the value to reach condition¨- VTR. We tested the optimization method in two; ten; twenty; thirty; forty-four-dimensional search space. We specified two different settings of acceptable tolerance (ε) of best found candidate solution testing (objective) function value from the objective function value of global optimum ($\varepsilon = 0.001$). If the global optimum of testing function is not known (e. g. Michalewicz testing function) the optimum is represented by the best found candidate solution of all performed simulation experiments performed on testing function.

4.1. Two-Dimensional Search Space

We started our simulation optimization experimentation in a two-dimensional search space where tolerated deviation from the value of the objective function value of global optimum is $\varepsilon = 0.001$. Figure 7 shows the average success of finding optimum. All tested optimization methods are successful (except Random Search) when the testing (objective) function landscape is not complicated – De Jong. Gradient optimization methods (Local Search, Hill Climbing and Tabu Search) have a problem to find the optimum if the testing function is multimodal – Ackley´s testing function.

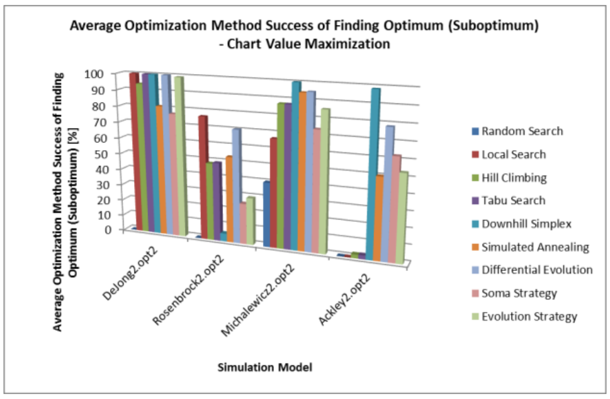

Figure 7: Average Optimization Method Success of Finding Optimum (Suboptimum) - Two-Dimensional Search Space – each series contains 40,000 simulation experiments; $\varepsilon = 0.001$

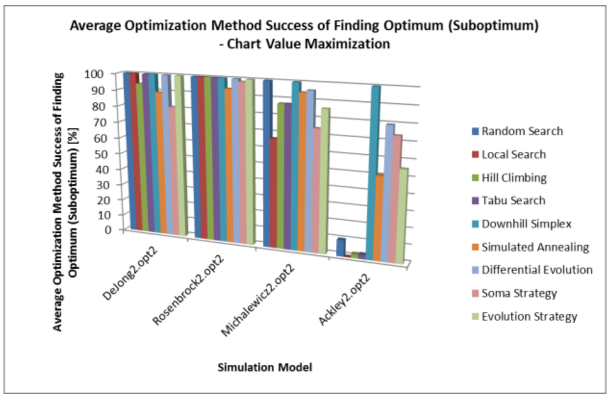

Figure 8: Average Optimization Method Success of Finding Optimum (Suboptimum) - Two-Dimensional Search Space – each series contains 40,000 simulation experiments; $\varepsilon = 1$ % of objective function value range

If the tolerance is set to one percent of the difference between the objective function value of the global maximum and the objective function value of the global minimum of the search space the optimization methods have no problem to find the global optimum of the testing function in two-dimensional search space - Figure 8. If the function landscape is complicated (especially Ackley´s function and also Michalewicz) the gradient methods are not effective.

4.2. Ten-Dimensional Search Space

Other optimization experiments are performed in a ten-dimensional search space. The finding of global optimum of testing functions is much more difficult - Figure 9. When the testing function landscape is not complicated – De Jong´s convex and unimodal function – some methods - Downhill Simplex, Differential Evolution, Evolution Strategy and eventually SOMA Strategy - are more effective than gradient based methods - Local Search, Hill Climbing and Tabu Search.

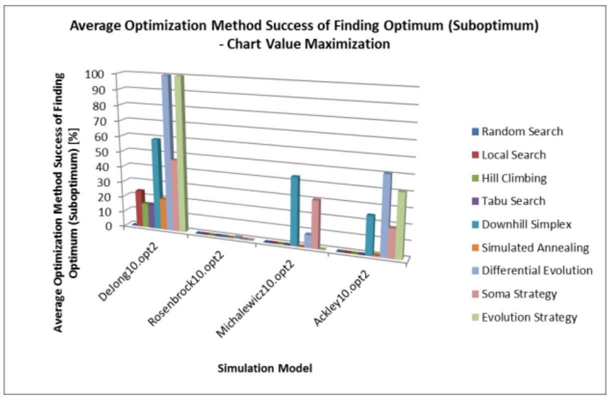

Figure 9: Average Optimization Method Success of Finding Optimum (Suboptimum) - Ten-Dimensional Search Space – each series contains 200,000 simulation experiments; $\varepsilon = 0.001$

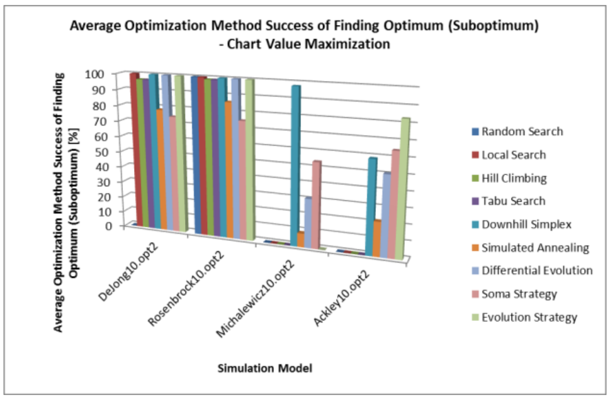

Figure 10: Average Optimization Method Success of Finding Optimum (Suboptimum) - Ten-Dimensional Search Space – each series contains 200,000 simulation experiments; $\varepsilon = 1$ % of objective function value range

If the user accepts one percent of the difference between the objective function value of the global maximum and the objective function value of the global minimum of the search space, the optimization methods have no problem to find the global optimum of simple testing function – De Jong´s and Rosenbrock´s testing function - Figure 10. If the testing function landscape is complicated – Michalewicz and Ackley´s testing function – gradient methods have a problem with finding the optimum.

4.3. Twenty-Dimensional Search Space

The problem of finding optimum in the search space increases with the higher number of the dimension of the search space. We could see that the optimization method has a problem with a ten-dimensional search space - Figure 9. Gradient based optimization methods are not effective. Local Search reaches 6.6 % success of finding optimum in the search space of De Jong´s testing function - Figure 11. Other gradient methods - Hill Climbing and Tabu Search - reach almost the same 2.3 % value of success of finding optimum in the search space of De Jong´s function.

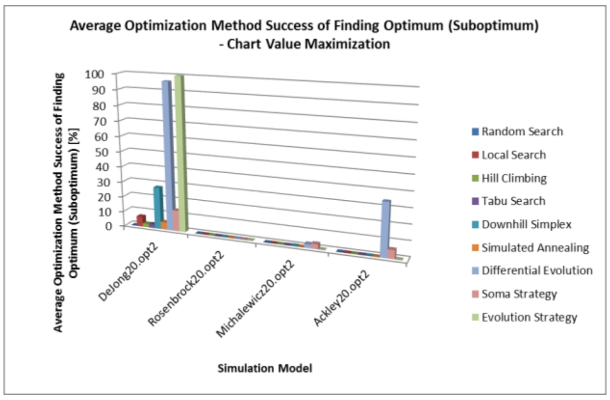

Figure 11: Average Optimization Method Success of Finding Optimum (Suboptimum) - Twenty-Dimensional Search Space – each series contains 400,000 simulation experiments; $\varepsilon = 0.001$

If we compare the efficiency of finding optimum in all searched search spaces of all testing functions when the epsilon is $\varepsilon = 0.001$ we can say that we should favour SOMA and Differential Evolution optimization methods. SOMA is another variation of the Differential Evolution methods.

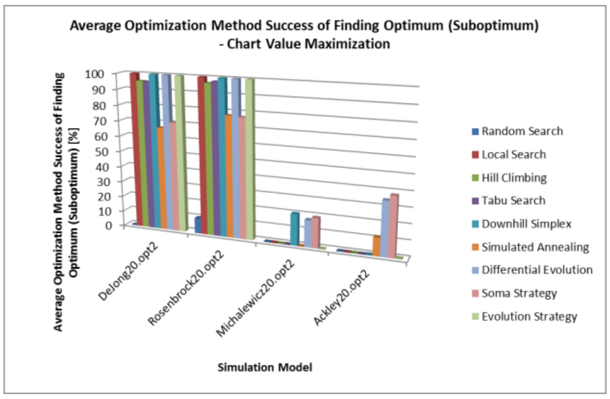

Figure 12: Average Optimization Method Success of Finding Optimum (Suboptimum) - Twenty-Dimensional Search Space – each series contains 400,000 simulation experiments; $\varepsilon = 1$ % of objective function value range

If we increase the tolerated deviation ε all tested optimization methods (except Random Search) can find the optimum in the search space of simple testing function – De Jong´s and Rosenbrock´s testing function – Figure 12. Only two methods - Simulated Annealing and SOMA - are less successful. Their average optimization method success of finding optimum varies between 66 % and 77%.

4.4. Thirty-Dimensional Search Space

Differential Evolution is very successful when we optimize De Jong´s thirty-dimensional search space - Figure 13.

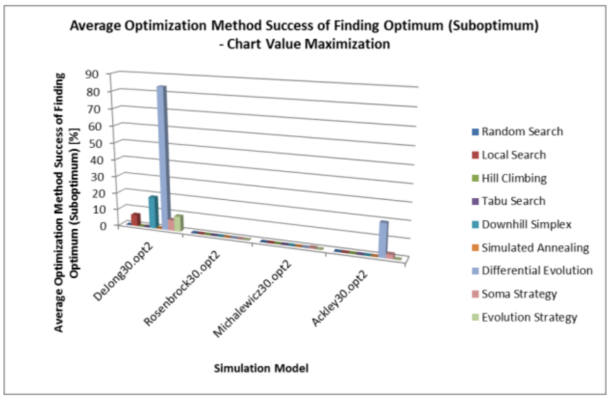

Figure 13: Average Optimization Method Success of Finding Optimum (Suboptimum) - Thirty-Dimensional Search Space – each series contains 600,000 simulation experiments; $\varepsilon = 0.001$

After comparing all charts representing the average optimization methods success we can say that the Differential Evolution is very effective in the case of our tested optimization methods. When ε equals 1 % of objective function value range all optimization methods are able to find the optimum in the simple testing function – De Jong´s and Rosenbrock´s function - with more than 60 % certainty - Figure 14.

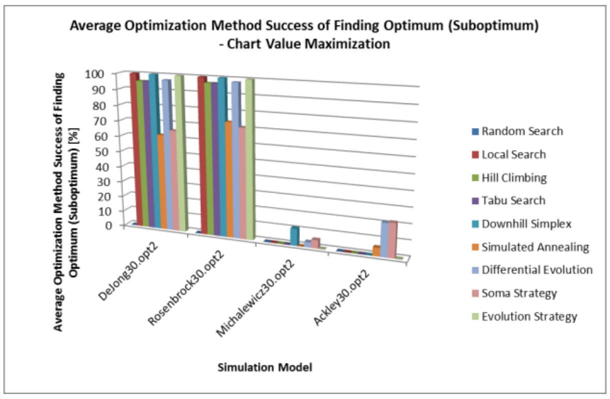

Figure 14: Average Optimization Method Success of Finding Optimum (Suboptimum) - Thirty-Dimensional Search Space – each series contains 600,000 simulation experiments; $\varepsilon = 1$ % of objective function value range

SOMA, Differential Evolution, eventually Downhill Simplex and Simulated Annealing are able to find the optimum if the testing function landscape is complicated – multimodal test function.

4.5. Forty-Four-Dimensional Search Space

Differential Evolution is also very successful when we want to find the optimum of De Jong´s thirty-dimensional search space. Compared to thirty-dimensional search space Evolution Strategy is not able to find the optimum in forty-four dimensional search space of De Jong´s testing function - Figure 15.

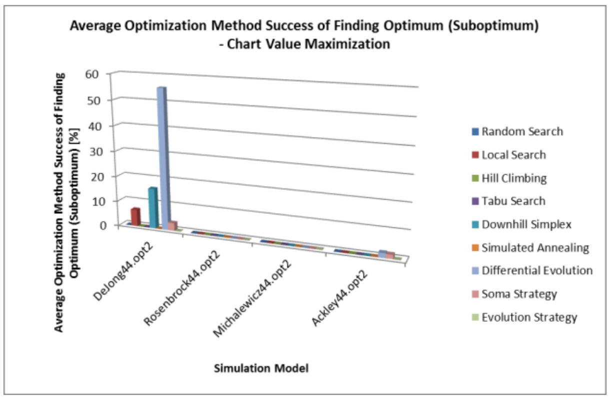

Figure 15: Average Optimization Method Success of Finding Optimum (Suboptimum) - Forty-four-Dimensional Search Space – each series contains 880,000 simulation experiments; $\varepsilon = 0.001$

Gradient based methods are not effective in the case of complicated testing functions with higher dimensions. These methods are useful for finding the optimum of a simple testing function landscape. Differential Evolution is successful optimization for optimization of different dimensional search spaces. This method is the best from our tested optimization methods.

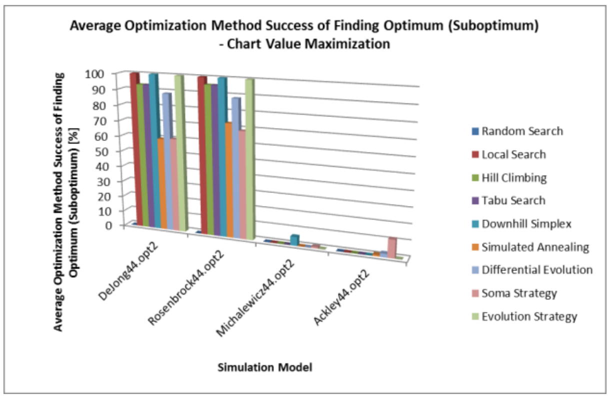

Figure 16: Average Optimization Method Success of Finding Optimum (Suboptimum) - Forty-four-Dimensional Search Space – each series contains 880,000 simulation experiments; $\varepsilon = 1$ % of objective function value range

SOMA is derived from Differential Evolution. This optimization method is useful when we can accept deviation equals one percent of the difference between the objective function value of the global maximum and the objective function value of the global minimum of the search space. If we do not want to use the methods based on evolution processes we can use Downhill Simplex method or Simulated Annealing.

4.6. Forty-Dimensional Search Space and Higher Number of Performed Simulation experiments

Regarding the large number of possible candidate solutions in the search space we set up the termination criterion in a way that the optimization method can perform a maximum of 100,000×*n* (parameter *n* denotes the dimension of the search space) simulation experiments to find the global optimum in the search space. The next condition of termination criterion is to stop the optimization experiment if the optimum is found - VTR (value to reach). We tested the optimization method on a forty-dimensional search space (i. e. optimization method could perform a maximum of 4,000,000 simulation experiments in each series).

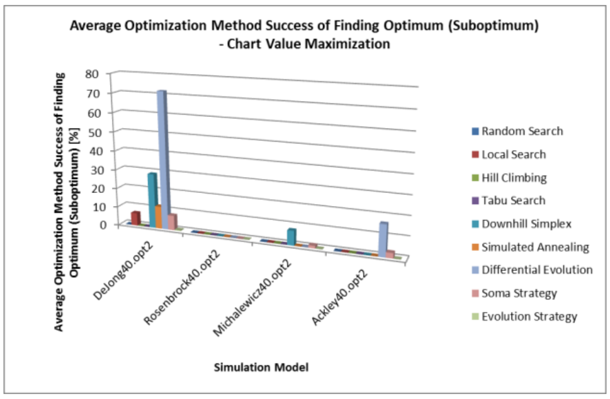

Figure 17: Average Optimization Method Success of Finding Optimum (Suboptimum) - Forty-Dimensional Search Space – each series contains 4,000,000 simulation experiments; $\varepsilon = 0.001$

If we compare the previous chart (forty-four dimension search space – see Figure 15) of success of finding optimum to the following chart (Figure 17) where the same condition of $\varepsilon = 0.001$ is met, it is obvious that the success of finding optimum by SOMA strategy increased the most in the case of Michalewicz, De Jong´s and Ackley´s testing functions. Differential Evolution and Downhill Simplex also improve their efficiency of finding the optimum if the number of performed simulation experiments increases.

If we compare the previous chart (Figure 16) of success of finding optimum to the following chart (Figure 18) where the same condition of $\varepsilon = 1$ % of objective function value range is met, we can say that the success of finding optimum of Simulated Annealing, Differential Evolution and SOMA strategy particularly increased in the case of Ackley´s function. Downhill Simplex and SOMA method also increase their efficiency of finding optimum of Michalewicz function if the number of performed simulation experiments in one series is five times higher. Gradient based optimization methods like Hill Climbing and Local Search also increase their average success of finding optimum with the higher number of simulation experiments but not as much as previously mentioned methods.

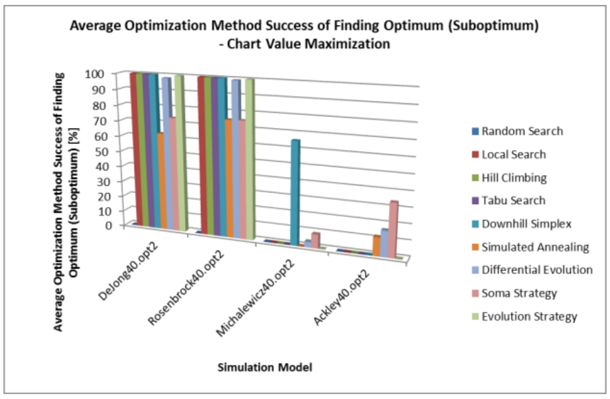

Figure 18: Average Optimization Method Success of Finding Optimum (Suboptimum) - Forty-Dimensional Search Space – each series contains 4,000,000 simulation experiments; $\varepsilon = 1$ % of objective function value range

5. SUMMARY

The goal of the research is to compare selected optimization methods - Random Search, Hill Climbing, Tabu Search, Local Search, Downhill Simplex, Simulated Annealing, Differential Evolution, SOMA and Evolution Strategy – used for optimization of four selected testing functions. We tested these optimization methods on different dimensions of the search space to compare their efficiency of finding the global optimum in the search space.

We substituted the testing on the discrete event simulation models by a different testing function – De Jong´s, Rosenbrock´s, Michalewicz and Ackley´s function. The dimension of the search space was 2; 10; 20; 30; 44. We specified the same conditions which had to be satisfied for each optimization method. Regarding the large number of possible candidate solutions in the search space we set up the termination criterion in a way that the optimization method can perform a maximum of 20,000 × the dimension of the search space simulation experiments to find the global optimum in the search space. We also tested the optimization method if the number of performed simulation experiments in one series is five times higher.

We initially specified tolerated deviation from the value of the objective function value of global optimum to a specified value ($\varepsilon = 0.001$). The optimization method had to find the candidate solution whose objective function value is nearly the same as the objective function value of global optimum in the search space. This aspect is unfounded in practice. Hence we specified the tolerance to one percent of the difference between the objective function value of the global maximum and the objective function value of the global minimum of the search space. The success of finding the optimum of the optimization method especially increases when we specified this new value of tolerated deviation (De Jong´s and Rosenbrock´s testing function).

The success of heuristic optimization methods depends on the objective function landscape. Gradient based methods are not effective in the case of complicated testing functions with higher dimensions. These methods are useful for finding the optimum of simple testing function landscapes. Differential Evolution is successful optimization for optimization of different dimensional search spaces. This method is the best from our tested optimization methods. SOMA is also a useful method for optimization. This method is derived from Differential Evolution.

Regarding the large time demands we would like to test the Client-Server architecture for parallel management of simulation experiments with different optimization methods settings. We also would like to test the use of a knowledge database to increase the speed of finding the optimum. The server sends the information about the setting of the optimization method parameters. The optimization experiment (simulation runs of a discrete event simulation model) is performed on the client PC. The client searches for the result of the simulation experiment with the same simulation model input parameters setting in the knowledge database before the simulation run. If this information is found, the client loads the objective function value from the database. If this information is not found, the client performs the simulation model run and the information about the simulation experiment (the setting of the simulation model input parameters and the objective function value) is sent to the knowledge database where this information is stored.

We would like to test optimization method behaviour and optimization parameters setting on different simulation models (e. g. simulation model of internal company logistics; this discrete event simulation model contains multiple simulation model input parameters).

ACKNOWLEDGEMENTS

This paper was created with the subsidy of the project SGS-2015-065 “The development of parameters of the sustainable manufacturing system” carried out with the support of the Internal Grant Agency of University of West Bohemia.

REFERENCES

Koblasa, F., Manlig, F., Vavruska, J., 2013. Evolution Algorithm for Job Shop Scheduling Problem

Constrained by the Optimization Timespan. *Applied Mechanics and Materials*, 309, 350-357.
Hynek, J., 2008. Genetic Algorithms and Genetic Programming (in Czech language: Genetické algoritmy a genetické programování). Prague: Grada.
Miranda, V., 2008. Fundamentals of Evolution Strategies and Evolutionary Programming. In: El-Hawary, M.E., ed. *Modern heuristic optimization techniques*. New Jersey: John Wiley & Sons, 43–60.
Mitchell, M., 1996. An Introduction to Genetic Algortihms. Cambridge: MA: MIT Press.
Monticelli, A.J., Romero, R., Asada, E., 2008. Fundamentals of Tabu Search. In: El-Hawary, M.E., ed. *Modern heuristic optimization techniques*. New Jersey: John Wiley & Sons, 101–120.
Monticelli, A.J., Romero, R., Asada, E., 2008. Fundamentals of Simulated Annealing. In: El-Hawary, M.E., ed. *Modern heuristic optimization techniques*. New Jersey: John Wiley & Sons, 123–144.
Pohlheim, H., 2006. GEATbx: Example Functions. Available from: http://www.geatbx.com/docu/fcnindex-01.html#P204_10395 [accessed 20 November 2011]
Raska, P. & Ulrych, Z., 2013. Simulation Optimizer and Optimization Methods Testing On Discrete Event Simulations Models and Testing Functions. ATHENS, GREECE, DIME Universita, pp. 50-59.
Raska, P. & Ulrych, Z., 2015. (in press) Comparison of Optimization Methods Tested On Testing Functions and Discrete Event Simulation Models. International Journal of Simulation and Process Modelling.
Rockwell Automation, 2014. OptQuest for Arena. Available from: http://www.arenasimulation.com/Products_OptQuest.aspx [accessed 3 January 2014]
Stefka, D., 2005. Alternatives to the evolutionary optimization algorithms (in Czech language: Alternativy k evolučním optimalizačním algoritmům) - Diploma Thesis. Prague: CVUT - Czech Technical University In Prague.
Tvrdik, J., 2010. Stochastic Algorithms for Global Optimization (in Czech language: Stochastické algoritmy pro globální optimalizaci). Available from: http://www1.osu.cz/~tvrdik/wp-content/uploads/STAGO_10.pdf [accessed 5 January 2014]
Weise, T., 2009. E-Book "Global Optimization Algorithms - Theory and Application" 2nd Edition. Available from: http://www.it-weise.de/projects/book.pdf [accessed 2 February 2011]
Wong, K.P., Dong, Z.Y., 2008. Differential Evolution. In: El-Hawary, M.E., ed. *Modern heuristic optimization techniques*. New Jersey: John Wiley & Sons, 171–186.
Zelinka, I., 2004. SOMA — Self-Organizing Migrating Algorithm. New Optimization Techniques in Engineering Studies in Fuzziness and Soft Computing. Berlin: Springer Berlin Heidelberg, 167-217.

AUTHORS' BIOGRAPHIES

Pavel Raska is Doctor at the Department of Industrial Engineering and Management at the University of West Bohemia in Pilsen (Czech Republic). He holds M.Sc., Ph.D. in Mechanical Engineering at the same university. His research interests are oriented towards discrete event simulation, simulation optimization, modelling and simulation tools (ARENA, Plant Simulation) and working on practical simulation projects for companies.

Zdenek Ulrych is Associate Professor at the Department of Industrial Engineering and Management at the University of West Bohemia in Plzen and he is also a research worker in the Regional Technological Institute at the University of West Bohemia in Pilsen (Czech Republic). He holds M.Sc., Ph.D. and doc. in Mechanical Engineering at the same university. His research interests are oriented towards discrete event simulation, optimization in the simulation, modelling and simulation tools (ARENA, Plant Simulation), design and development of software and working on practical simulation projects for companies.