Acta Polytechnica Vol. 52 No. 6/2012

Global Optima for Size Optimization Benchmarks by Branch and Bound Principles

Adéla Pospíšilová, Matěj Lepš

Faculty of Civil Engineering, CTU in Prague, Thákurova 7, 166 29 Prague, Czech Republic

Corresponding author: adela.pospisilova@fsv.cvut.cz

Abstract

This paper searches for global optima for size optimization benchmarks utilizing a method based on branch and bound principles. The goal is to demonstrate the process for finding these global optima on the basis of two examples. A suitable parallelization strategy is used in order to minimize the computational demands. Optima found in the literature are compared with the optima used in this work.

Keywords: benchmarks, discrete sizing optimization, branch and bound method, global optima, parallel programming.

1 Introduction

Optimization and search methodologies have become very popular for making products more desirable. The shape of a structure, the amount of reinforcement, the cross-sections, sheet thicknesses, design of the concrete mixture, and many other properties can be optimized. Recently, many heuristic algorithms have been developed and tested on benchmarks in order to assess their performance. In order to compare different optimization methods, it is necessary to have reliable information on the global optima of the benchmarks. In the past, it was not possible to obtain these optima by exhaustive search approaches, due to the large computational demands. As computational power is growing year by year, it now seems to be the right time to deal with this issue.

This paper outlines a process for searching global optima for sizing discrete optimization benchmarks. Various optimization methods can be used for obtaining optima, e.g. gradient methods [1], heuristic methods, and evolutionary algorithms [2]. These methods do not guarantee that a global optimum is obtained because only a portion of the space is explored. It is not always necessary to obtain the global optimum. A local optimum with good qualities found within a short time can be considered as a mark of a high quality algorithm. However, without knowledge of the global optima for selected benchmarks, it is not possible to make a reliable assessment of the performance of the methodology that is used.

In our work, we look for global optima for some fundamental benchmarks, using a method based on branch and bound principles. This approach requires two values called bounds for determining the searched space. A good estimate of these bounds reduces the searched space but still ensures that global optima can be found. The optimization problem for the benchmarks presented in this paper is defined by an objective function that is easy to solve, constraints with high computational demands and with a searched space that is discrete and huge. The algorithm presented here can be used for obtaining the global optimum for problems similar to those discussed in this paper.

2 Sizing optimization

Sizing optimization [3] is a type of structural optimization that deals with truss-like structures. These structures are defined by a fixed topology, material, loading, supports, and a set of cross-sections or, alternatively, minimum and maximum cross-sectional areas of individual truss bars. The objective function is the weight of the structure or its volume. The objective function is linear and easy to solve. Constraints are maximum stresses and maximum displacements, respectively. These functions are non-convex in the most cases, and it is more time-consuming to solve them that to solve the objective function.

The goal is to find sections for a given structure that satisfy the prescribed constraints and have the lowest possible weight. The selection of cross-sections from the given database then defines a discrete optimization problem, and variables chosen from given limits lead to a continuous case. The continuous optimization problem can be efficiently solved by mathematical programming methods, e.g. gradient-based methods, as will be shown below. When using discrete variables, no such option is available. Thus we first give our attention to the discrete case.

3 Discrete problem

The goal is to find a combination of cross-sections from the given list of profiles that leads to the lowest possible weight, while still fulfilling the given constraints. We present two methods that are able to find global optima for this discrete optimization problem.

3.1 Enumeration

Enumeration (also called a Brutal Force or Exhaustive search) is the simplest method for obtaining a global optimum of the discrete optimization problem. Here, it is necessary to compute values of an objective function and constraints for every combination of cross-sections from a given set. The enumeration therefore poses very large computational demands. If there are n sections and k variables (i.e., truss bars or groups of bars), then n^k possible solutions exist, i.e., the problem grows exponentially with the growing number of variables. Enumeration can therefore be applied only for small structures or for analysing the neighborhood of some local optima.

3.2 A method based on branch and bound principles

A *branch and bound method* is another method for obtaining global optima. Land and Doig [4] invented this method for linear programming problems. Later, it was modified for discrete problems and for mixed-discrete problems [5].

Branch and bound methods are based on a dividing the main problem into several subproblems, known as branches. To estimate which branches are to be evaluated, the existence of the lower and upper bounds is assumed to restrict the searched space. The lower bound can be obtained by any continuous optimization method, because the global optimum with discrete design variables will never provide a lower value of the objective function than the global optimum with continuous design variables. The upper bound can be obtained by any heuristic method, because a local optimum always has a greater or equal value of the objective function than the global optimum. Since the constraints for the sizing optimization problem are more computationally demanding than the value of the objective function, they are calculated only for solutions that lie between the lower and upper bounds. If we obtain a subproblem with a value of the objective function outside the given bounds, the rest of the branch is not calculated because a global optimum cannot be located there. The more accurate the estimates of the lower and upper bounds are, the narrower the searched space can be. It is thus advantageous to decrease the upper bound on the basis of already obtained objective function values.

Although this methodology is more efficient than enumeration, it is still time consuming. However, it is possible to parallelize the algorithm to reduce the computation time. The main idea of the parallel version of the algorithm is presented in Section 6.

4 Continuous problem

A continuous optimization problem is more complex than a discrete problem, because an infinite number of potential solutions exist in the space with real numbers. Therefore, for non-convex problems, it cannot be guaranteed that the optimum that is found is the global optimum. However, powerful and well-established continuous optimization algorithms such as mathematical programming methods, can be used for obtaining a potential global optimum. Obtaining a potential global optimum with continuous variables is therefore less demanding than solving the optimization problem with discrete variables.

The main disadvantage of this methodology is the uncertainty of the quality of the solution. This can be overcome in one of two ways:

- The branch and bound method expects that the lower bound has the same (or a lower) value of the objective function as the global optimum with continuous variables. Since the global optimum of the continuous problem cannot be generally known, the true lower bound cannot be ensured. As a solution, the lower bound is set to its lowest potential minimum, i.e., without using any continuous optimization method. This process provides a real global optimum with discrete variables. In most cases, however, the searched space will be enormous for the computation of all possible solutions in real time.

- Other approaches do not fully guarantee the acquisition of the global optimum. Nevertheless, the probability of obtaining the global optimum is acceptable. These approaches are based on estimating the global optimum with continuous variables and the value of its objective function. We use nonlinear programming that is implemented in the MATLAB environment (e.g. the `fmincon` function). This routine is executed several times from random initial points. If the obtained optima do not differ from each other and the results are comparable to optima published in the available literature, the estimate is considered as credible. If the obtained optima differ from each other, then it is not possible to use them as the lower bound. The first approach (without using the continuous optimization method) is then used, or the lower bound is estimated to be e.g. 20 % lower than the solution what is obtained.

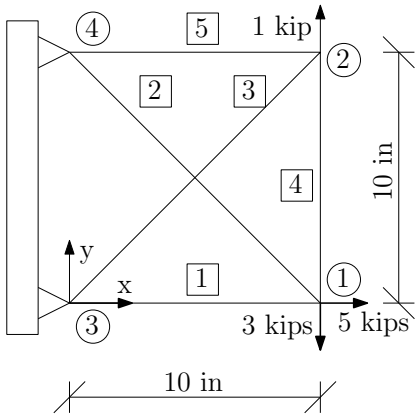

Figure 1: The 5-bar truss.

All continuous optima for problems mentioned below were consistent with published results. For the sake of certainty, the nonlinear programming method was launched hundred times with different starting vectors, and the best solution was considered as the lower bound.

5 Benchmarks

5.1 5-bar truss

It was necessary to have a representative example of a structure that was small enough for computational demands and yet big enough for branching purposes in order to test the branch and bound algorithm. Note that imperial units are used throughout the text, because our solutions will be compared with published optima in the available literature with the same units.

The structure in Fig. 1 has four nodes and five truss bars, and is made from aluminium. The density of the material is $0.1\,\mathrm{lb/in^3}$ and the Young's modulus is equal to $10^4\,\mathrm{ksi}$. The allowable stress is limited to $\pm 60\,\mathrm{ksi}$ in each element, and the displacements are limited to $\pm 0.06\,\mathrm{in}$ along the horizontal and vertical directions. The continuous variables can vary between the lower bound $0.01\,\mathrm{in^2}$ and the upper bound $0.1\,\mathrm{in^2}$.

A function for nonlinear programming `fmincon` [6] offers four variants of the optimization algorithms. In this paper, the Active Set Method [7] is suitable for our continuous sizing optimization problem. It is based on changing inequalities into equalities, followed by the line-search algorithm leading to a quadratic subproblem. This procedure is repeated in a sequence which converges in the limit to a critical point [8].

A starting point, i.e., a design variable vector composed of cross-sectional areas, an objective function, constraints and lower and upper bounds of the variables are necessary as an input at the beginning of the algorithm. The objective function is the weight

Variable	Units	Discrete optimization		Continuous optimization
		E	B&B	`fmincon()`
A_1	$\mathrm{in^2}$	0.05	0.05	0.0500
A_2	$\mathrm{in^2}$	0.01	0.01	0.01
A_3	$\mathrm{in^2}$	0.06	0.06	0.0471
A_4	$\mathrm{in^2}$	0.02	0.02	0.0167
A_5	$\mathrm{in^2}$	0.01	0.01	0.01
m	lb	**0.179**	**0.179**	0.157
$\max\lvert w_j\rvert$	in	0.059	0.059	0.06
$\max\lvert\sigma_i\rvert$	ksi	59.371	59.371	60.006
w_{lim}	in	0.06	0.06	0.06
σ_{lim}	ksi	60	60	60

Table 1: 5-bar truss optima. B&B — Method based on Branch and Bound principles, E — Enumeration.

of the structure

$$m = f(A) = \rho \sum_{i=1}^{N} A_i L_i, \tag{1}$$

where $N = 5$ is the number of truss bars, A_i is the cross-sectional area and L_i is the length of element i, and ρ is the density of the material. Constraints are defined as inequalities:

$$\max|\sigma_i| - 60 \leq 0, \tag{2}$$

$$\max|w_j| - 0.06 \leq 0, \tag{3}$$

where $\max|\sigma_i|$ is the maximum absolute value of stresses, j is an ordinal number of independent displacements and $\max|w_j|$ is the maximum absolute value of displacements. These values can be obtained by several methods. In this paper, geometrically and physically linear behaviour was assumed and the finite element method was used, see e.g. [9].

The results for the continuous optimization problem of the 5-bar truss appear in Tab. 1. The objective function value of the optima obtained with the Active Set Method is later used as the lower bound for the branch and bound method. Dealing with inequalities in the form of equalities is solved by a penalty type approach, and therefore the exact fulfillment of the constraints therefore cannot be ensured, see Table 1. This discrepancy is not crucial since only a lower bound is needed, not the global continuous optimum.

An identical topology of the 5-bar truss is used for the discrete optimization version. The material properties and constraints are also identical.

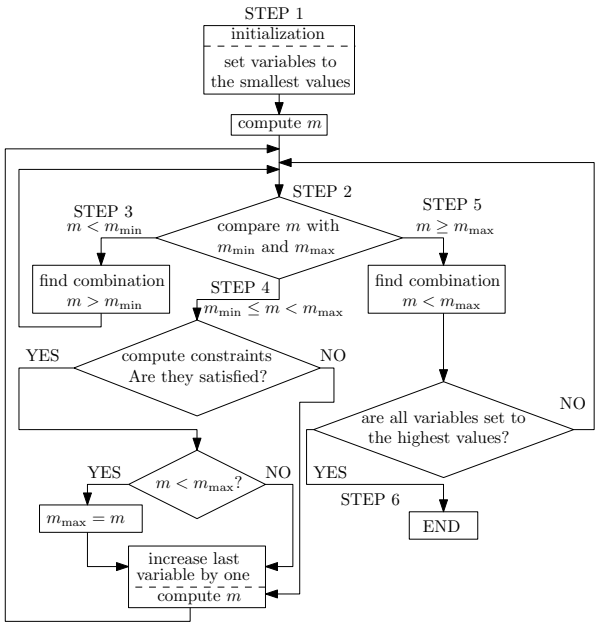

Figure 2: A flow chart of the algorithm.

The cross-sectional areas are chosen from the set $\{0.01, 0.02, \ldots, 0.1\}$ in^2. Since the structure has five elements ($k = 5$) and 10 cross-sectional areas ($n = 10$), the number of all possible solutions is $n^k = 10^5$. Therefore, the structure is small enough and the discrete global optimum can be obtained by the enumeration method. The results appear in Tab. 1.

The lower bound for the branch and bound method is set to the optimum objective function value obtained with continuous variables. The upper bound is set to the estimated weight of 0.23 lb, which is 25 % higher than the global optimum value of the objective function obtained by the enumeration. The space is searched systematically between these two bounds until the global optimum is found.

A flow chart of the algorithm is depicted in Fig. 2. The steps can be described as follows:

1. First, we have to decide which values will be used as the initial point. It is appropriate to begin with the smallest profiles and then increase them, since the objective function is linear with respect to the cross-section areas. From the programming point of view, it is easier to use integer variables that are the ordinal numbers of the given set of cross-sectional areas — set M. For example, the initial design variable vector is `1 1 1 1 1`, which means that the first area (0.01 in^2) from the given set is attached to each truss bar, according to the numbering of the trusses shown in Fig. 1.

2. The value of the objective function is then calculated and compared with the lower $m_{\min}$ and upper $m_{\max}$ bounds. If the weight of the structure is less than $m_{\min}$, the algorithm will go to Step 3. If the weight of the structure is between $m_{\min}$ and $m_{\max}$, the algorithm proceeds to Step 4. If the weight of the structure is greater than $m_{\max}$, Step 5 is executed.

3. The value of the objective function is less than $m_{\min}$. It is therefore necessary to find a combination of variables that corresponds to weight larger than $m_{\min}$. The last variable is raised to its maximum value, e.g. as `1 1 1 1 10`, and the value of the objective function is calculated and compared to $m_{\min}$.

   (a) If the value of the objective function is still less than $m_{\min}$, the algorithm searches for a combination with greater weight than the lower bound. This can be done as follows. The next-to-last variable is repeatedly raised by one, e.g. to (`1 1 1 2 10`). If the value of the next-to-last variable reaches its maximum, it is decreased to its minimum and the third from the end variable value is raised by one. The algorithm will go to Step 2 at the moment when all variables are set such that $m > m_{\min}$.

   (b) If the value of the objective function is greater than the lower bound, the last variable value is decreased to its minimum (`1 1 1 1 1`) and is then increased one by one (`1 1 1 1 2`, `1 1 1 1 3`, ..., etc.) until the weight is greater than $m_{\min}$. If $m_{\min} > m$ the algorithm goes to Step 2.

4. The value of the objective function is greater than $m_{\min}$ and less than $m_{\max}$. The global optimum is located somewhere in this subspace. Therefore, the constraints are evaluated, i.e the stresses and displacements are calculated.

   (a) If the constraints are fulfilled, i.e., $\max|\sigma_i| \leq 60$ ksi and $\max|w_j| \leq 0.06$ in, the upper bound is updated to the actual objective function value $m_{\max} = m$. Thus the upper bound is pushed down towards the global optimum and the searched space is reduced. The last variable value is then increased by one. If this variable value exceeds its maximum possible cross-sectional area value from a given set, e.g. `1 1 5 11 1`, its value is set to the minimum possible value, and the next-to-this variable value is increased by one, i.e., `1 1 6 1 1`. The algorithm goes to Step 2.

   (b) If the constraints are not fulfilled, the value of the last variable is increased by one. The algorithm continues with Step 2.

5. The value of the objective function is greater than $m_{\max}$. The value of the last variable is decreased

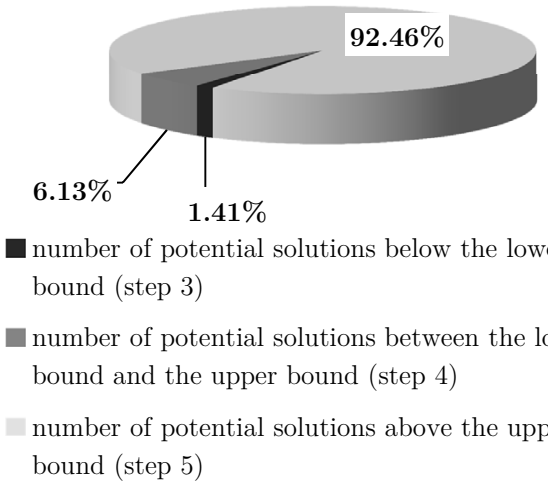

Figure 3: A pie chart of a distribution of the 5-bar truss problem solutions solved by the branch and bound method.

to its minimum, the next-to-last variable value is increased by one and the objective function value is calculated. If the variable exceeds its maximum possible value from the given set, the algorithm acts as in Step 4a.

(a) If the value of objective function m is lower than $m_{\max}$, the algorithm goes to Step 2.

(b) If objective function value m is greater than $m_{\max}$, the value of the third variable from the end increases by one and the value of the next-to-last variable is set to its minimum. The algorithm continues in this way until the objective function value is less than $m_{\max}$. If there is no such combination of cross-sectional areas, the algorithm is terminated.

6. If all variable values are set to their maxima, the algorithm ends.

Fig. 3 shows the distribution of 5-bar truss potential solutions. The dark grey part shows the number of potential solutions below the lower bound, where only the objective function values are calculated, i.e., Step 3 of the algorithm. The grey part shows the number of potential solutions between the lower and the upper bound, where the values of the objective function and also the constraints are calculated (Step 4). The global optimum is included in this subspace. The light grey part represents the number of potential solutions above the upper bound, where only the objective function values are calculated (Step 5). It is obvious that there is no need to compute constraints for more than 90% of potential solutions. Since the evaluation of constraints is very computationally demanding, this results in significant time savings. The reason for the bigger number of potential solutions above the upper bound is that the optimum is placed relatively close to the lower bound in the searched subspace.

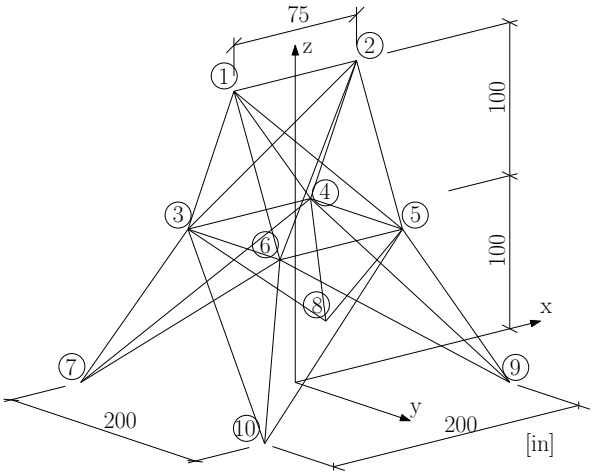

Figure 4: A 25-bar truss.

Tab. 1 presents results for the continuous optimization problem along with results for the discrete problem solved by the enumeration and the method based on branch and bound principles. Since the enumeration calculates the values of the objective function as well as constraints for all potential solutions, it is not possible to omit the global optimum. The results obtained by both presented methods are identical and this comparison serves as the verification of the branch and bound method.

5.2 25-bar truss

The 25-bar truss is one of the most widely-used benchmarks for size optimization. It was introduced by Fox and Schmit [10] in 1966. The structure has ten nodes and four supports (at nodes 7–10), see Fig. 4; therefore there are 18 free displacements. The structure is symmetric so some elements were gathered to eight groups, listed in Tab. 2. The truss is made of aluminium material, with density equal to $0.1\,\mathrm{lb/in^3}$ and with the Young's modulus equal to $10^4\,\mathrm{ksi}$. The loading is defined in Tab. 3. All cross-sectional areas are chosen from the given set: $0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0, 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.8, 3.0, 3.2$ and $3.4\,\mathrm{in}^2$, see [11]. The continuous variables range from $0.1\,\mathrm{in}^2$ to $3.4\,\mathrm{in}^2$. The allowable stress is set to $\pm 40\,\mathrm{ksi}$ in all truss bars and the maximum allowable displacement is $\pm 0.35\,\mathrm{in}$ at all nodes along the x, y and z directions.

The results for the continuous case are shown in Tab. 4 and they are compared with the results published in the available literature. The discrete case cannot be enumerated within a reasonable time because the number of potential solutions is $n^k = 30^8 = 6.561 \cdot 10^{11}$, where k is the number of element groups. For the discrete case, the lower bound was set to the

Group of bars	Conectivities
A_1	1-2
A_2	1-4, 2-3, 1-5, 2-6
A_3	2-5, 2-4, 1-3, 1-6
A_4	3-6, 4-5
A_5	2-4, 5-6
A_6	3-10, 6-7, 4-9, 5-8
A_7	3-8, 4-7, 6-9, 5-10
A_8	3-7, 4-8, 5-9, 6-10

Table 2: Member grouping for the 25-bar truss.

Node	F_x	F_y	F_z
1	1.0	−10.0	−10.0
2	0	−10.0	−10.0
3	0.5	0	0
6	0.6	0	0

Table 3: Loadings for the 25-bar truss (kips).

Variable	Unit	Perez & Behdinan	This paper
A_1	in^2	0.1	0.1
A_2	in^2	0.457	0.421
A_3	in^2	3.4	3.4
A_4	in^2	0.1	0.1
A_5	in^2	1.937	1.917
A_6	in^2	0.965	0.966
A_7	in^2	0.442	0.471
A_8	in^2	3.4	3.4
m	lb	483.84	**483.82**
$\max \lvert\sigma_i\rvert$	ksi	6.15	6.13
$\max \lvert w_j\rvert$	in	0.35	0.35
σ_{lim}	ksi	40	40
w_{lim}	in	0.35	0.35

Table 4: Comparison of results for the 25-bar truss continuous case.

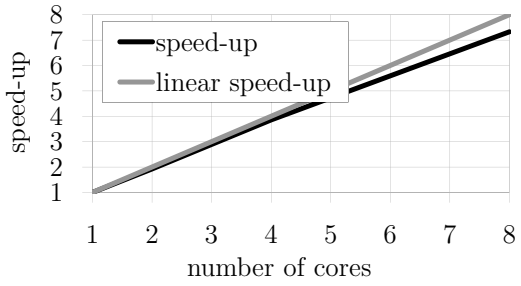

Figure 5: A speed-up of the 25-bar truss problem solved by the parallel branch and bound method.

value found with continuous optimization, and the upper bound was set to the worst available solution in the literature [12].

6 Parallelization

The 25-bar truss is relatively computationally demanding. Since the evaluations of the solutions are independent from each other (except updating the upper bound $m_{\max}$, as described in Step 4a of the algorithm), the method can be run in a parallel way. Nowadays, modern computers are equipped with several core processors, and we can to make use of this computational power. The MATLAB environment offers several parallelization tools, but not all of them provide shared memory. This consideration is essential for our algorithm for updating the upper bound.

The appropriate method is the `spmd` method, i.e., the Single Programm Multiple Data method, see e.g. [13] for more details. The `spmd` statement separates the block of the code to be run simultaneously on multiple labs. As in the `parfor` loop method, the `matlabpool open N` command opens the required number of labs. Data can be sent to another lab by the `labSend(data, X)` command, where `X` is the index of the receiving lab to which the data is sent. The data is received by the `labReceive(Y)` command, where `Y` is the index of the lab from which the data will come. It is appropriate to split the data only at one so-called *master* lab and to receive data with the other so-called *slaves*. The master can also process its own data as well.

The main problem here is to estimate a proper amount of data for each lab. If the data is sent too often communication between master and slaves is too costly. In the 25-bar truss task, permutations with repetition are generated in advance for several groups of elements (e.g. four groups), and the remaining groups of elements (four other groups) are generated in the branch and bound method independently at each lab. The generated combinations are assigned to individual labs in advance, and then the algorithm continues in the same way as for the 5-bar truss task. The maximum $m_{\max}$ values are collected at the end

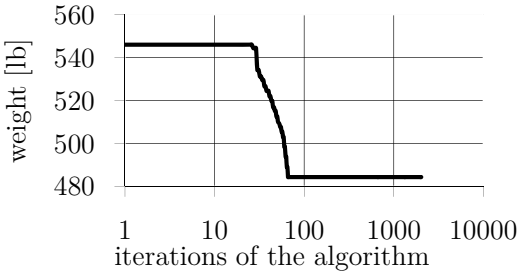

Figure 6: A graph of the decreasing upper bound for the 25-bar truss problem.

of each iteration. The smallest one is chosen as a new $m_{\max}$ and is resent to every lab as an initial value of $m_{\max}$ for another iteration. If all data has been used, the smallest value of $m_{\max}$ is taken as the global optimum.

For the parallel version of the algorithm, scaling the algorithm is very important. If the algorithm is scaled badly, parallelization is not useful at all. Ideally, we would like to achieve linear scaling, i.e., speed-up of n on n cores. However, it is very hard to obtain linear scaling, e.g. because of the time spent on communications. Fig. 5 shows a graph where the speed-up of the parallel algorithm is compared at 1 to 8 labs[1]. The HP Xeon Z600 Workstation with two Intel Xeon E5520 4-cores processors, frequency 2.27 GHz was used for computations within the Matlab R2009a 64-bit Debian GNU/Linux environment.

7 Conclusions

Fig. 6 shows a graph with the decreasing upper bound $m_{\max}$ for the 25-bar truss problem. The value of $m_{\max}$ determines the best-so-far solution found during the whole algorithm run. It can be interpreted as the convergence of the objective function to the global optimum. All combinations generated in advance were divided into smaller blocks of data containing fifty combinations and these "packs" were sequentially sent to eight labs. The number of all iterations was therefore $30^4/(8 \cdot 50) = 2025$. The global optimum was gained in the 66th iteration. Nevertheless, it should be pointed out that the number of iterations depends strictly on the data ordering or the starting point (minimum vs maximum cross-sectional areas). Since the task is to find the global optima, the whole subspace of potential solutions must be searched for, and it is not possible to shorten the computation.

In Tab. 5, we compare the optima obtained with the branch and bound method with the heuristic algorithms found in the literature. The result of the branch and bound method (B&B) is identical to the solution presented by Kripka [15] using the Simulated Annealing method (SA). However, he did not search the whole subspace of possible solutions, so he could not be sure that the obtained optimum is the global one.

To the best of the authors' knowledge, global optima for computationally demanding tasks such as the 25-bar truss problem have not been published yet. We hope that by publishing the algorithm as well as the value of the global optimum we will introduce a quality standard that will help to improve new optimization methods.

[1] It was not necessary to compute the whole task. Some variables were fixed to prescribed values, here 2 out of 8 variables, and the algorithm was run with this restriction.

Acknowledgements

This work was supported by the Grant Agency of the Czech Technical University in Prague, grants number SGS11/021/OHK1/1T/11 and number SGS12/027/OHK1/1T/11, the Czech Science Foundation GACR, grant number P105/12/1146 and by Ministry of Education, Youth and Sports, grant number MSM 6840770003.

References

[1] Shewchuk, J. R.: *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*, School of Computer Science, Carnegie Mellon Univ., Pittsburgh, [online], 1994.

[2] Eiben, A. E., Smith, J. E.: *Introduction to Evolutionary Computing*, Berlin: Springer, 2003.

[3] Bendsoe, M. P., Sigmund O.: *Topology Optimization: Theory, Methods and Applications*, Berlin: Springer, 2003.

[4] Land, A. H., Doig A. G.: *"An Automatic Method of Solving Discrete Programming Problems." Econometrica*, **28** (3), 1960, pp. 497–520.

[5] Arora, J. S.: *Methods for Discrete Variable Structural Optimization.* In: "Recent Advantages in Optimal Structural Design". American Society of Civil Engineers, 2002, p. 1–40.

[6] The MathWorks, *Find minimum of constrained nonlinear multivariable function - MATLAB.* [02/10/2011]. `http://www.mathworks.com/`

[7] Bhatti, M. A.: *Practical Optimization Methods*, New York: Springer-Verlag, 2000.

[8] The MathWorks, *Constrained Nonlinear Optimization Algorithms: Optimization Algorithms and Examples.* [02/10/2011]. `http://www.mathworks.com/`

Variable	Units	This paper	Kripka [15]	Lemonge & Barbosa [16]	Li & Liu [17]	Wu & Chow [11]	Coello [18]	Rajeev & Krishnamoorthy [12]
		B&B	SA	GA	PSO	GA	GA	GA
Date		2012	2004	2004	2009	1995	1994	1992
A_1	in^2	0.1	0.1	0.1	0.1	0.1	1.5	0.1
A_2	in^2	0.4	0.4	0.3	0.3	0.5	0.7	1.8
A_3	in^2	3.4	3.4	3.4	3.4	3.4	3.4	2.3
A_4	in^2	0.1	0.1	0.1	0.1	0.1	0.7	0.2
A_5	in^2	2.2	2.2	2.1	2.1	1.5	0.4	0.1
A_6	in^2	1	1	1	1	0.9	0.7	0.8
A_7	in^2	0.4	0.4	0.5	0.5	0.6	1.5	1.8
A_8	in^2	3.4	3.4	3.4	3.4	3.4	3.2	3
m	lb	**484.33**	**484.33**	484.85	484.85	486.29	539.78	546.01
$\max \lvert\sigma_i\rvert$	ksi	6.20	6.20	6.11	6.11	6.01	6.66	6.77
$\max \lvert w_j\rvert$	in	0.35	0.35	0.35	0.35	0.35	0.34	0.35

Table 5: A comparison of results for the 25-bar truss, discrete case, from the literature and from our work. B&B — Method based on Branch and Bound principles, SA — Simulated Annealing, GA — Genetic Algorithm, PSO — Particle Swarm Optimization. $\sigma_{\text{lim}} = 40\,\text{ksi}$, $w_{\text{lim}} = 0.35\,in$.

[9] Pospíšilová A.: *Analysis of Sizing Optimization Benchmarks*, Bachelor Thesis, CTU in Prague, 2010. `http://klobouk.fsv.cvut.cz/~pospisilova/publications/AP_BP_2010.pdf`

[10] Fox, R. L., Schmit, L. A. Jr.: *"Advances in the Integrated Approach to Structural Synthesis."* *J Spacecraft Rockets*, **3** (6), 1966, p. 858–866.

[11] Wu, S.-J., Chow, P.-T.: *"Steady-State Genetic Algorithms for Discrete Optimization of Trusses."* *Comput Struct*, **56** (6), 1995, p. 979–991.

[12] Rajeev, S., Krishnamoorthy, C. S.: *"Discrete Optimization of Structures Using Genetic Algorithms."* *J Struct Eng*, **118** (5), 1992, p. 1233–1250.

[13] The MathWorks, *Executing Simultaneously on Multiple Data Sets: Single Program Multiple Data (spmd).* [02/10/2011]. `http://www.mathworks.com/`

[14] Perez, R. E., Behdinan, K.: *Particle Swarm Optimization in Structural Design.* In: "Swarm Intelligence, Focus on Ant and Particle Swarm Optimization". Itech Education and Publishing, 2007, p. 373–394.

[15] Kripka, M.: *"Discrete Optimization of Trusses by Simulated Annealing."* *J Braz Soc Mech Sci Eng*, **26** (2), 2004, p. 170–173.

[16] Lemonge, A. C. C., Barbosa, H. J. C.: *"An adaptive penalty scheme for genetic algorithms in structural optimization."* *Int J Numer Meth Eng*, **59** (5), 2004, p. 703–736.

[17] Li, L. J., Huang, Z. B., Liu, F.: *"A heuristic particle swarm optimization method for truss structures with discrete variables."* *Comput Struct*, **87** (7-8), 2009, p. 435–443.

[18] Coello, C. A. C.: *Discrete Optimization of Trusses Using Genetic Algorithms.* In: "EXPERSYS-94: Expert Systems Applications and Artificial Intelligence". I.I.T.T. International, 1994, p. 331–336.