COMBINATORIAL TOPOLOGY OPTIMIZATION OF TRUSS STRUCTURES USING DISTRIBUTED GRID COMPUTING

Aleksandr Igumenov, Julius Žilinskas
Institute of Mathematics and Informatics
Tel. +370 52109300, e-mail: Igumenov@gmail.com

Topology optimization of truss structures is considered in this paper. Trusses are widely used in various constructions: bridges, towers, roof supporting structures. Topology optimization of trusses requires large amount of computing resources. Therefore distributed computer networks are used to solve this kind of problems. In this paper a distributed branch-and-bound combinatorial algorithm for topology optimization of trusses is proposed, implemented and investigated in computational grid.

Introduction

Optimization is widely used in science and engineering. In this paper one of many engineering problems is considered – optimization of truss structure. Trusses are used in construction of bridges, towers, roof supports, etc. A truss includes joint elements (beams), which may have different parameters: position, length, cross-sectional area, material. Truss structure may be optimized in several aspects: sizing, shape, and topology [2,3,5,8,10]. In this paper topology optimization [6,7] is considered, where material and cross-section of beams as well as the position of immovable nodes are defined. The problem is to find the necessary beams so that the mass of the structure would be minimal and constraints would be satisfied.

The problem of topology optimization of truss structure can be formulated as combinatorial optimization problem. It can be approached using various methods: complete enumeration of all possible structures [1], branch-and-bound, local search, genetic algorithms [6,7], etc. In this paper we consider branch-and-bound algorithm, which can guarantee that the global optimum is found.

The number of different possible truss structures grows exponentially with the number of nodes. Therefore branch-and-bound algorithms require many computations. To make optimization faster parallel branch-and-bound [4] may be performed on high performance computers and computational grids [9]. In this paper distributed version of branch-and-bound has been developed and experimentally investigated.

Problem formulation

The aim of topology optimization of truss structure is to find the best structure (layout) of connections (beams) between the given set of immovable nodes. Mathematically the presence of beams in the structure can be coded by binary variables and the problem can be formulated as a combinatorial optimization problem. The objective function of the problem is the total mass of the structure and the constraints are set so that the stresses in beams do not exceed critical values, the structure is locally stable and it is in the static equilibrium state [7]. The problem may be mathematically formulated as

$$\min f(X) = \sum_{i=1}^{n} x_i L_i \rho_i A_i, \qquad \text{s.t. } g(X) = 0, \quad h(X) \le 0, \quad x_i \in \{0,1\},$$

where the objective function $f(X)$ corresponds to the total mass of the structure; binary variables x_i code the presence of i-th beam in the structure; L_i is a length of the i-th beam; ρ_i is density of beam material; A_i is cross-sectional area of the same i-th beam; n is the number of possible beams, $n=m(m-1)/2$, where m is the number of nodes; equality constraints defined by $g(X)$ ensure that the total sum of forces at each node would be zero – the system would be in the static equilibrium state; and the inequality constraints defined by $h(X)$ ensure that the stresses in beams do not exceed critical values and the system is locally stable. Constraint functions $g(X)$ and $h(X)$ are modelled by a finite element method [6] and provided as black-boxes for the considered optimization algorithms.

Algorithms for optimization of truss structure

Branch-and-bound is a general structure for implementation of algorithms for combinatorial optimization [12] and coverings algorithms for global optimization [13]. The main concept of branch-and-bound is to search for the optimum, constructing a search tree so that only some of the possible structures should be explicitly checked detecting sets of structures, represented by branches of the search tree, which cannot contain optimal structures. The bound for the objective function over a set of possible structures should be evaluated and compared with the best objective function value found so far. If the evaluated bound is worse than the best function value known so far, the set cannot contain optimal structures and the branch

describing the set can be pruned. The fundamental aspect of branch-and-bound is that the sooner the branch is pruned, the smaller the number of structures will need to be explicitly checked.

Evaluation of bounds for the objective function is the most important part of the branch-and-bound technique. Performance of branch-and-bound algorithms depends on the tightness of bounds [11]. If the bounds are not tight, the search may lead to complete enumeration of all possible structures. This is not acceptable practically for all but the smallest problems, because the number of possible structures grows exponentially with the size of problem. Construction of the bound depends on the objective function and the type of sets of possible structures over which the bound is evaluated.

The developed branch-and-bound algorithm for optimization of truss structure is presented in Algorithm 1. The set of possible structures is represented by a part of binary variables defined by p. Existence of additional beams can only increase the mass of structure, therefore the lower bound for the objective function can be found assuming that there are no other beams except those that already have been defined. If this bound is larger than the best function value known so far (f_{min}), which is the smallest mass of structure satisfying constraints, this set of structures must not be further investigated.

Algorithm 1. Sequential branch-and-bound algorithm for optimization of truss structure

$f_{min} \leftarrow \infty$; $X = (0,...,0,0)$; $p = 1$

while $p > 0$ **do**

if $f(X) < f_{min}$ **and** $g(X) = 0$ **and** $h(X) \leq 0$ **then** $f_{min} \leftarrow f(X)$; $X^* \leftarrow X$ end if

if $f(X) < f_{min}$ **then** $p \leftarrow n$ **while** $x_p = 1$ **and** $p > 1$ **do** $x_p \leftarrow 0$; $p \leftarrow p$-1 end while

if $p > 0$ **then** $x_p \leftarrow 1$ end while

The developed distributed branch-and-bound algorithm for optimization of truss structure is presented in Algorithm 2. The algorithm differs from the sequential algorithm only that some of the binary variables are defined by the rank of the processes.

Algorithm 2. Distributed branch-and-bound algorithm for optimization of truss structure

Input: *rank*; *size*

$f_{min} \leftarrow \infty$; $X \leftarrow$ (*rank*,0, . . . ,0,0); $p \leftarrow \log(size)/\log(2)+1$ **while** $p > \log(size)/\log(2)$ **do**

if $f(X) < f_{min}$ **and** $g(X) = 0$ **and** $h(X) \leq 0$ **then** $f_{min} \leftarrow f(X)$; $X^* \leftarrow X$ end if

if $f(X) < f_{min}$ **then** $p \leftarrow n$

while $x_p = 1$ **and** $p > \log(size)/\log(2)+1$ **do** $x_p \leftarrow 0$; $p \leftarrow p-1$ end while

if $p > \log(size)/\log(2)$ **then** $x_p \leftarrow 1$ end while

Experimental researches

Software and hardware

For execution of experiments the following hardware and the program equipment was used:

- LitGrid project resources (grid.mii.lt).
- The implementations of the algorithms have been written using C++.

Experiments

Experiments have been fulfilled 10 times and their results have been averaged. Execution times (in seconds) of sequential and distributed algorithms are presented in Table 1. As it could be expected, execution time grows exponentially depending on the number of nodes in the truss structure. Execution time decreases when the number of processes is increased. Table 2 shows time of sequential algorithm in clusters of LitGrid. *vdupdc.vdu.lt* and *grid.pri.kmu.lt* clusters are heterogeneous: min and max time differ up to twice. The speed of various clusters differs up to four times (*vdupds.vdu.lt* is the slowest and *ce.grid.lei.lt*, *grid.marko.lt* and *grid.mii.lt* are the fastest). Some of the clusters are not accessible during experimentation. The largest accessible cluster is *grid.mii.lt*.

Results of experimental investigation of distributed version of branch-and-bound algorithm are presented in Figs. 1, 2 and 3. Performance of distributed optimization is good and branch-and-bound algorithm scales well enough. The speedup is almost linear and close to the number of processes when the number of processes is up to 4. Efficiency of parallelization is 0.6 - 1. When the number of processes is larger the efficiency of parallelization drops to 0.4 – 0.6. The utilization (Fig. 3) shows the average load of processes. The utilisation is 0.6 – 1. That shows that the processes are not equally loaded.

Table 1. ***Times of execution of the branch-and-bound algorithm in grid (grid.mii.lt)***

Number of processes to solve task	**1 process**	**2 processes**	**4 processes**	**8 processes**	**16 processes**	**32 processes**
6 nod. str., sec	0.080	0.040	0.025	0.014	0.010	0.006
7 nod. str., sec	4.75	2.50	1.64	1.33	0.88	0.43
8 nod. str., sec	633.34	354.10	292.20	200.77	154.65	69.90
9 nod. str., sec	142067	98458	64466	52186	49376	24990

Table 2. **Times of execution of the branch-and-bound algorithm in various clusters**

The cluster of BalticGrid	**The number of processes**	**The 7 nodes task**		**The 8 nodes task**	
		min	**max**	**Min**	**Max**
ce.grid.lei.lt	10	4.59	4.73	628.47	632.42
ce.grid.vgtu.lt	8	Not accessible			
ce1.grid.vgtu.lt	6	Not accessible			
ce2.grid.vgtu.lt	44	Not accessible			
dane.ku.lt	18	5.68	5.78	765.82	785.07
grid.akolegija.lt	5	5.78	5.81	770.80	772.26
grid.fi.lt	13	Not accessible			
grid.marko.lt	5	4.5	5.18	600.16	603.32
grid.mii.lt	68	4.75	4.76	632.19	633.34
grid.panko.lt	6	Not accessible			
grid.pri.kmu.lt	12	9.20	12.48	1258.78	1709.95
grid.su.lt	32	Not accessible			
grid2.mif.vu.lt	8	Not accessible			
grid4.mif.vu.lt	18	Not accessible			
grid9.mif.vu.lt	64	8.68	8.75	1265.43	1266.90
pupa.elen.ktu.lt	12	9.59	11.95	1615.86	1658.5
spectras.itpa.lt	4	Not accessible			
vdupdc.vdu.lt	10	9.35	13.32	1274.35	2660.00

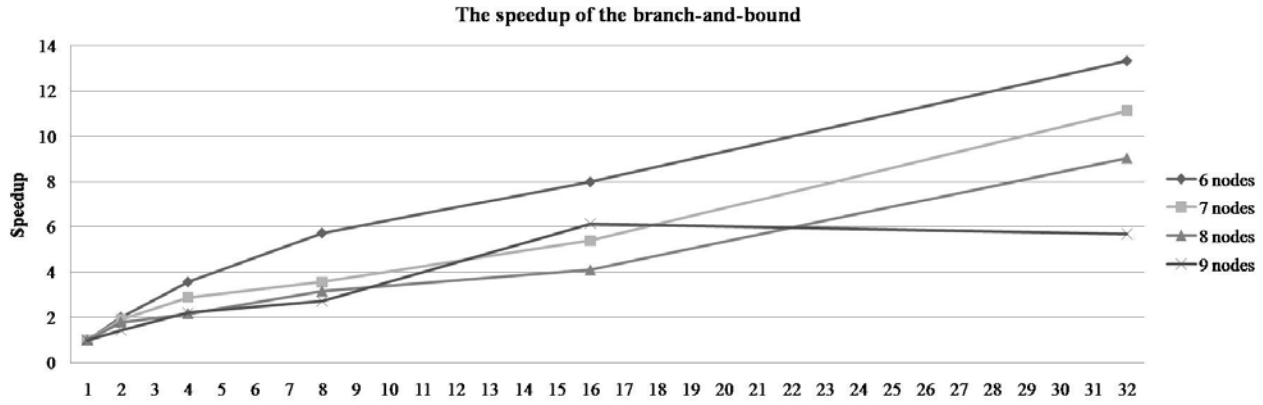

Figure 1. Speedup of the distributed version of branch-and-bound algorithm

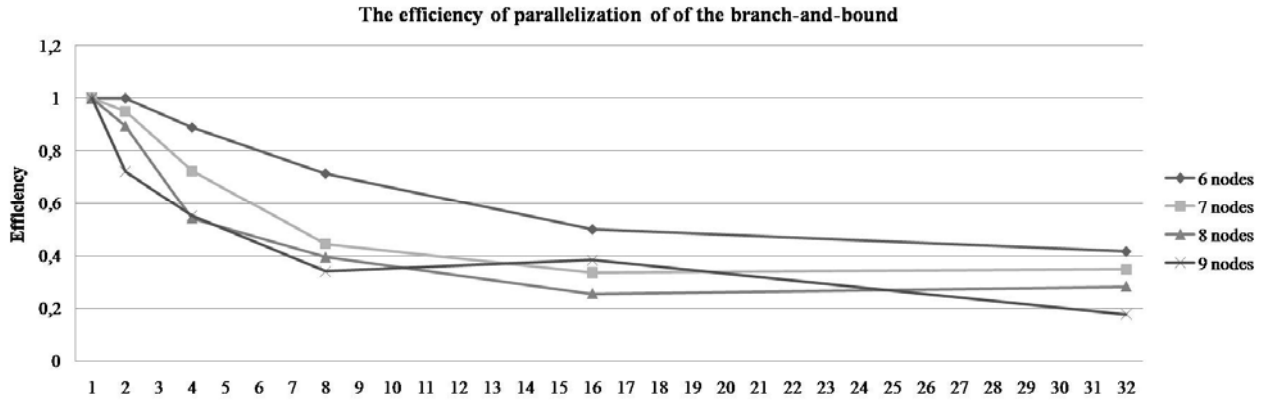

Figure 2. Efficiency of parallelization of the distributed version of branch-and-bound algorithm

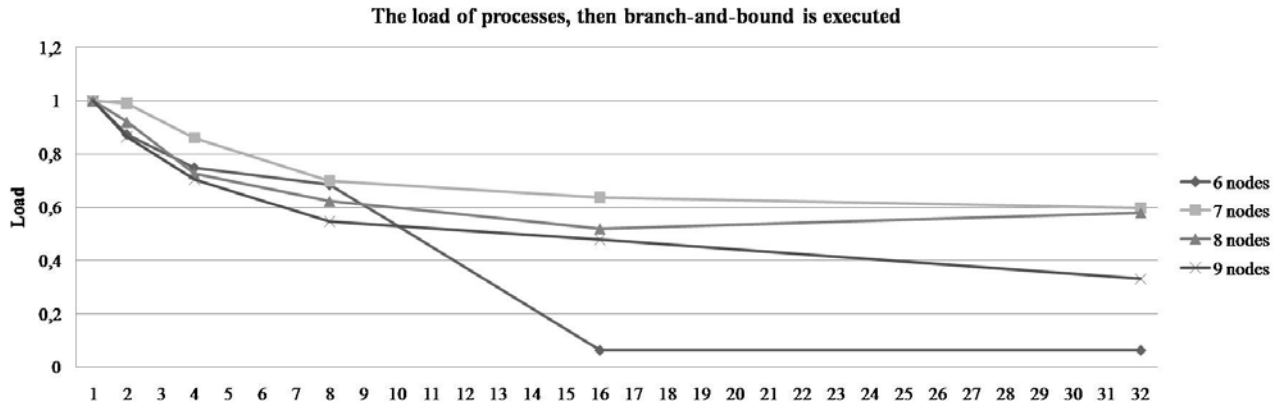

Figure 3. Utilization of the distributed version of branch-and-bound algorithm

Conclusions

The performance of distributed version of branch-and-bound algorithm for optimization of truss structure is good and the algorithm scales well.

For up to 4 parallel processes the speedup is close to the number of processes and efficiency of parallelization is close to 1. Efficiency is smaller for larger number of processes.

Acknowledgement

The research is partially supported by LitGRID programme and the Research Council of Lithuania.

References

1. Igumenov A., Žilinskas J. Combinatorial Algorithms for Topology Optimization of Truss Structure. *Proceedings of the 15th International Conference on Information and Software Technologies*, pp. 229 – 234, 2009.
2. Imai K., Schmit L.A. Configuration optimization of trusses. *Journal of the Structural Division*, 107(5), 745-756. 1981.
3. Janušaitis R., Keras V., Mockienė J. Development of methods for designing rational trusses. *Journal of Civil Engineering and Management,* 9(3), 192-197. 2003.
4. Paulavičius R., Žilinskas J. Parallel branch and bound algorithm with combination of Lipschitz bounds over multidimensional simplices for multicore computers. In: R. Čiegis, D. Henty, B. Kågström, J. Žilinskas (Eds.), Parallel Scientific Computing and Optimization. Vol. 27 of *Springer Optimization and Its Applications, Springer*, pp. 93-102, doi:10.1007/978-0-387-09707-7_8. 2009.
5. Smith J., Hodgins J., Oppenheim I., Witkin A. Creating models of truss structures with optimization. *ACM Transactions on Graphics,* 21(3), 295-301. 2002.
6. Šešok D., Belevičius R. Use of genetic algorithms in topology optimization of truss structures. *Mechanika,* 64(2), 34-39. 2007.
7. Šešok D., Belevičius R. Global optimization of trusses with modified genetic algorithm. *Journal of Civil Engineering and Management,* 14(3), 147-154. 2008.
8. Topping B.H.V. Shape optimization of skeletal structures: A Review. *Journal of Structural Engineering*, 109(8), 1933-1951, doi:10.1061/(ASCE)0733-9445(1983)109:8(1933). 1983.
9. Varoneckas A., Žilinskas A., Žilinskas J. Parallel multidimensional scaling using grid computing: assessment of performance. *Information Technology and Control*, 37(1), 52-56. 2008.
10. Yates D.F., Templeman A.B., Boffey T.B. The complexity of procedures for determining minimum weight trusses with discrete member sizes. *International Journal of Solids and St*ructures, 18(6), 487-495, doi:10.1016/0020-7683(82)90065-8. 1982.
11. Žilinskas A., Žilinskas J. On efficiency of tightening bounds in interval global optimization. *Lecture Notes in Computer Science*, 3732, 197-205, doi:10.1007/11558958_22. 2006.
12. Žilinskas A., Žilinskas J. Branch and bound algorithm for multidimensional scaling with city-block metric. *Journal of Global Optimization*, 43(2-3), 357-372, doi:10.1007/s10898-008-9306-x. 2009.
13. Žilinskas J. Branch and bound with simplicial partitions for global optimization. *Mathematical Modelling and Analysis*, 13(1), 145-159, doi:10.3846/1392-6292.2008.13.145-159. 2008.