

Workshop on state-of-the-art in scientific  
computing PARA04

**R. ČIEGIS,**

**M. Baravykaitė, R. Belevičius**

Vilnius Gediminas Technical University, Lithuania

# **Parallel Global Optimization of Foundation Schemes in Civil Engineering**

PARA04 Lyngby, Denmark

June 20–23, 2004

# A plan of the talk

1. Problem formulation
2. Global optimization algorithm
3. Parallelization of the computational algorithm
4. Applications
5. Conclusions

## Problem formulation

Optimization is an inherent part of all engineering practice.

Grillage consists of separate beams, which may be supported by piles, or may reside on other beams. The **optimal placement** of grillage should possess, depending on given carrying capacities of piles, the minimum possible number of piles.

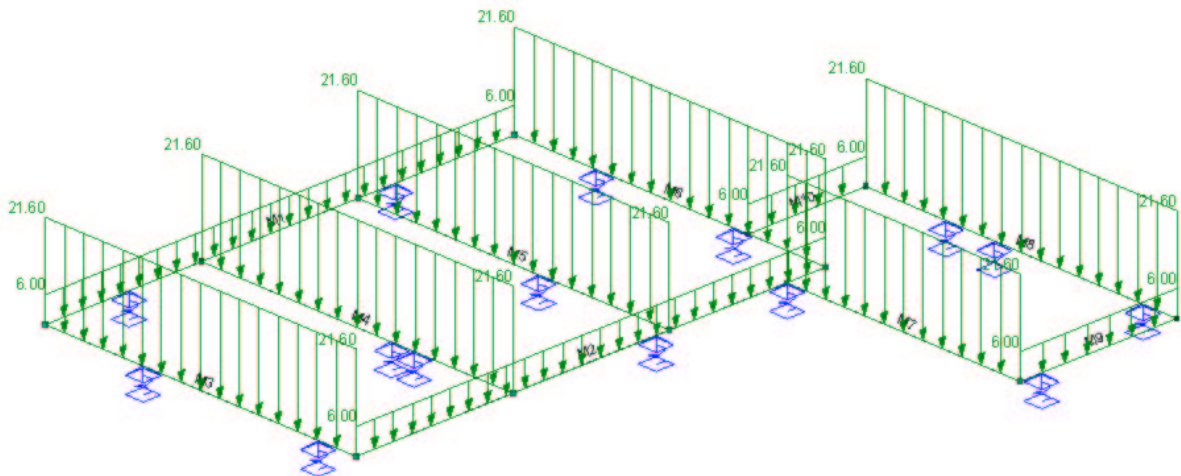


Fig. 1 A scheme of the grillage.

The optimization problem is stated as follows

$$\begin{aligned} \min \quad & P(\mathbf{x}), \\ \text{s.t. } \quad & \mathbf{x} \in D \end{aligned}$$

$P$  is the maximum difference between vertical reactive force at a support and allowable reaction for this support.

$$P(\mathbf{x}) = \max_{1 \leq i \leq N_s} |R_i - f_i R_{allowable}|,$$

$N_s$  denotes the number of supports,  $R_{allowable}$  is allowable reaction,  $f_i$  are factors to this reaction and  $R_i$  are reactive forces in each support.

- $P(x)$  is not given analytically ( a **black-box** case), it is computed by FEM discretization,
- the properties of this function (e.g. the Lipschitz constant) can not be obtained apriori.

$D$  is the feasible shape of structure.

## Local Optimization Algorithm

The solution requires three steps:

- finite element analysis,  
(simple two-node beam element with 4 d.o.f.'s has been implemented for FEM discretization),
- sensitivity analysis with regard to design parameters,  
( or the nonlinear Simplex method),
- optimal re-design with linear programming.

**Remark.** In order to find a proper solution, the algorithm should be launched from a near-optimum initial scheme, which is delivered by special expert system.

## Global Optimization Algorithm 1.

The most simple global optimization algorithm is obtained by starting a local search algorithm from many different trial points.

The quality of the obtained solution depends on the initial trial points, and some heuristic for the selection of such initial approximations should be given.

Such strategy can be easily implemented in **parallel** by using **Master – Slaves** paradigm.

We have developed a tool for automatic parallelization of Master – Slaves algorithms. It is written in C and freely distributed at

*[http : //perkunas.vtu.lt/Milda/MST3.html](http://perkunas.vtu.lt/Milda/MST3.html)*

## A tool for automatic parallelization of Master – Slaves algorithms

```
int main()
{
    struct t_init_data id;
    struct t_data      t;
    struct t_result    res;

    M_prepare_job_pool(&id);
    if (! S_initialize(id)) return 0;
    while ( M_take_piece_from_pool(&t))
    {
        S_compute(t, &res);
        M_add_to_result(res);
    }
    M_print_result();
    return 1;
}
```

## Global Optimization Algorithm 2.

We have applied a more sophisticated global optimization method, which is based on combination of

- The Branch and Bound method,
- Interval algorithms.

### Main Steps

- The monotonicity test (no local minimum exists in the tested subdomain);
- Once a local minimizer has been found, a domain around it is taken out from father searches;
- A subdivision step ( and the estimation of the lower bound, if needed).

## Parallel BB Algorithms

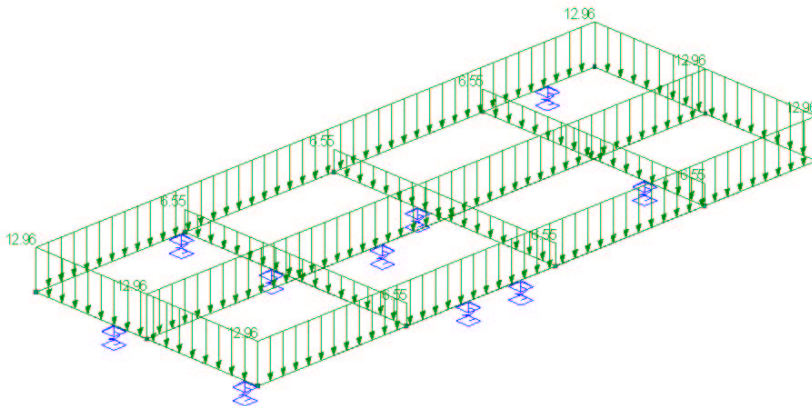
Parallelization of Branch and Bound has many common points with parallelization of adaptive multidimensional integration algorithms.

- The Domain distribution method;  
(The domain is divided into sub-domains, which are distributed among processors. Only the currently known best value of the objective function is exchanged between processors).
- Distributed Dynamic load balancing algorithms (to ensure that the work load is distributed uniformly among processors).

# Computational Experiments

Computational experiments were performed on a cluster of VGTU. It consist of 10 dual processor PC, running LINUX.

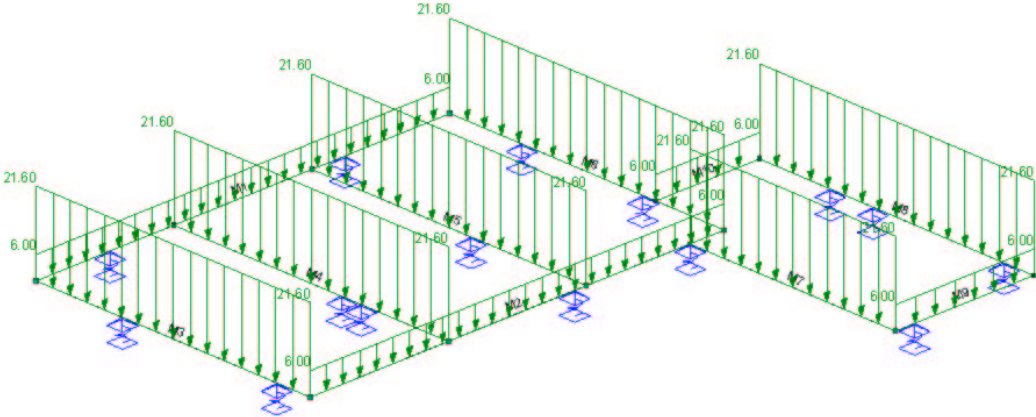
**Example 1.** Grillage of rectangular shape is loaded with two sets of distributed vertical loadings. The stopping condition of computations is specified by a limit of CPU time. After 60 minutes of computations the proposed algorithm yields the following supports:



Reactive forces for supports:

–206.3, –170.6, –204.5, –189.0, –94.96,  
–196.8, –168.6, –207.5, –192.1, –207.4.

**Example 2.** Grillage consists of 2 rectangular frames under distributed loadings. Theoretical number of supports for initial data on a limiting factor 150 is 15. After 5 hours of computations on 20 processors cluster the following placement scheme was achieved:



Reactive forces for supports are the following:

- 156.4, -158.1, -147.4, -161.2, -153.6,
- 118.8, -130.6, -161.3, -161.5, -139.7,
- 161.2, -144.0, -94.37, -121.1, -136.4.