GPU based parallel optimization of members of a truck floor

Sz. Nagy\textsuperscript{1} and K. Jármai\textsuperscript{2}

\textsuperscript{1} PhD student, University of Miskolc, HU
\textsuperscript{2} Professor, University of Miskolc, HU

E-mail: szilard.nagy@emerson.com

Abstract. Grillage - formally lattice - structures are made of longitudinal beams and cross members. Grillages can model vehicle frames, bus floor frames or an entire vehicle. The optimization of the cross members of this structure is shown in this article. The optimization method is a nature-inspired metaheuristic evolutionary method, the flower pollination algorithm. The target function is the total weight and cost of the optimized cross-member. Unknowns are typical cross-sectional dimensions of the cross-member. The design constraints considered are local buckling of web plate and flanges, and fatigue. The calculation was performed assuming the usage of aluminium alloys. In this paper, we propose a possible parallelization method, for computation of base algorithm and one group of fitness functions. The results show, that parallel computations can lead to significant reduction in computational time, if the population size is large and the number of variables are also large.

1. Introduction
Evolutionary algorithms belong to the heuristic branch of artificial intelligence; they have received considerable attention in recent years. This is not surprising, as they are well-suited for solving non-linear, multi-variable, sophisticated, search and optimization problems. They can also be successful in situations, where traditional gradient methods are challenging to use, or not applicable at all. Another great advantage is, that they can treat the target function as a black box. You do not need to know the specific internal structure of the function, only the inputs and their responses are important. A disadvantage though, is that in the operation it is not possible to determine whether the result obtained is a local or global minimum. From the many evolutionary algorithms available, in this paper we will introduce and apply, the Flower Pollination (FPA) algorithm.

Support grilles and support grid-like structures are used in many places in the vehicle industry. Such a structure can be used, for example, to model the chassis of trucks, buses, or even the cellular plate design of ships.

Parallelization can decrease the required computation time. Evolutionary algorithms may be parallelized in three different ways [1]:

- **global model**: This is the simplest way. Only the elementary operations are parallelized.
- **regional model**: The whole population is divided into equal-sized sub-populations and they run independently. The connection between sub-populations is the migration.
- **local model**: Each individual runs on a separate microprocessor. They communicate only with neighbors.
In this paper, a method for the parallel computation of the flower pollination algorithm (FPA) is shown and calculating in parallel, the fitness or the objective functions are proposed.

2. CUDA environment

GPUs no longer only support graphical applications and gaming. These are becoming cheap and powerful tools for scientific and general-purpose computations, for topology optimization [2], [3], or structural optimization [4] [5], and also in manufacturing technologies [6][7]. They provide a massively parallel environment, with the support of a single instruction multiple data (SIMD) programming model. Nowadays, two frameworks are typically used. The first one is an OpenCL, which is an open-source standard, parallel programming of heterogeneous systems. The second one is CUDA, which was developed by NVIDIA, supporting parallel programming of their video cards.

CUDA is an extension of standard C/C++ language. It is formally named CUDA-C. Three function types are introduced [8][9]:

- **host functions**: They are called by the host (CPU) and run on the host device; they are exactly the same original functions of C/C++ language.
- **kernel functions**: They are called by the host and executed by the GPU on many threads. They are declared with the prefix \_global\_.
- **device functions**: They are called only from the inside kernel and will run on the same thread where they are called. They are declared with the prefix \_device\_.

Kernel functions run on many threads. Threads are organized into blocks. Upper-level blocks are in grids, as it is visible in Figure 1. Grids and blocks can have up to three dimensions. Their maximum size depends on the type of graphics card used. The actual size of the grids and blocks depends on the problem to be executed. Each dimension is identified by a system variable: gridDim, blockIdx, blockDim, threadIdx. More information about the meaning of these variables can be found in [8][9].

3. Parallel flower pollination algorithm (FPA)

Flower pollination is a major reproduction process of plants. It can take two forms: abiotic (local) and biotic (global). Pollens are transferred long distances during global pollination by pollinators such as insects, birds, wind, etc. This is common to about 90% of flowering plants. The abiotic form does not require any pollinator. This inspired a method to develop the FPA [11].

In the FPA algorithm, global pollination is modelled by the following equation:

\[
\tilde{x}_i^{(G+1)} = \tilde{x}_i^{(G)} + L \left( \tilde{x}_i^{(G)} - \bar{g}_i^{(G)} \right)
\]  

Figure 1 CUDA architecture [8]
where \( \bar{x}_i \) is \( i^{th} \) individual (pollen), \( L \) is the Levy distribution random number and \( g^* \) represents the fittest individual. Local pollination can be described:

\[
\bar{x}_i^{(G+1)} = \bar{x}_i^{(G)} + \epsilon (\bar{x}_j^{(G)} - \bar{x}_k^{(G)}) \quad i \neq j \neq k \in [1, D]
\]

where \( \epsilon \) is a uniform distribution random number in \([0, 1]\) and \( i, j, k \) are independent indexes, \( D \) is the number of unknowns. A \( p \) probability variable, decides between the two methods. A more detailed description of the sequential algorithm can be found in [11].

For running in parallel, let \( \bar{L}, \bar{\epsilon}, \bar{j}, \bar{i} \) be vectors. Their elements can be generated independently on many threads by the cudaRAND library. The size of these vectors is equal to the \( n_p \) population size.

Let all individuals at \( G^{th} \) iteration step be stored in the \( \bar{P} \) matrix in the following structure:

\[
\bar{P} = \begin{bmatrix}
    x_1^{(G)} \\
    x_2^{(G)} \\
    : \\
    x_n_p^{(G)} \\
\end{bmatrix} = \begin{bmatrix}
    x_{1,1}^{(G)} & x_{1,2}^{(G)} & \cdots & x_{1,s}^{(G)} & \cdots & x_{1,D}^{(G)} \\
    x_{2,1}^{(G)} & x_{2,2}^{(G)} & \cdots & x_{2,s}^{(G)} & \cdots & x_{2,D}^{(G)} \\
    : & : & \cdots & : & \cdots & : \\
    x_{r,1}^{(G)} & x_{r,2}^{(G)} & \cdots & x_{r,s}^{(G)} & \cdots & x_{r,D}^{(G)} \\
    : & : & \cdots & : & \cdots & : \\
    x_{n_p,1}^{(G)} & x_{n_p,2}^{(G)} & \cdots & x_{n_p,s}^{(G)} & \cdots & x_{n_p,D}^{(G)}
\end{bmatrix}
\]

(3)

Rows contain pollens and each cell in the row stores the coordinates of individuals. In a mutation step, every element of \( \bar{P} \) can calculate independently, using previously defined vectors according to SIMD programming model.

\[
P_{r,s}^{(t)} = \begin{cases}
    P_{r,s}^{(G)} + L_r (P_{r,s}^{(G)} - g^*_s) & P_r \leq \text{rand}[0,1] \\
    P_{r,s}^{(G)} + \epsilon_r (P_{i,r,s}^{(G)} - P_{k,r,s}^{(G)}) & \text{otherwise}
\end{cases}
\]

(4)

After mutation parallel selection is:

\[
P_{r,s}^{(G)} = \begin{cases}
    P_{r,s}^{(t)} & \mathcal{F}(\bar{P}_{r}^{(t)}) \leq \mathcal{F}(\bar{P}_{r}^{(G)}) \\
    P_{r,s}^{(G)} & \text{otherwise}
\end{cases}
\]

(5)

A flow chart of the proposed method is illustrated in Figure 2. The main steps follow each other sequentially, but the operations inside steps are executed in parallel. The generation of random vectors is represented by one symbol; but if graphics card computation capability allows, they can run separately as asynchronous sequences.
4. Modified parallel reduction

The fitness function is a special function, which is different for each evolutionary algorithm. These rank individuals in a population in every iteration step. Despite their diversity, they have a common feature: They make reductions. This means, they order a real number in the $D$ dimensional searching space, and they depend on target function and constraints.

$$\mathcal{F}(\bar{x}) = \mathcal{F}(f(\bar{x}), g_1(\bar{x}), \ldots, g_q(\bar{x}), h_1(\bar{x}), \ldots, h_p(\bar{x}))$$

(6)

where $\mathcal{F}: \mathbb{R}^D \rightarrow \mathbb{R}$

Parallel reduction - also called (parallel) prefix scan in the literature - is a well-known element of a group of data parallel algorithms.

**Definition** [12]: Let $B$ a base set and let $\otimes$ binary, associative operator defined over elements of $B$. Assume that the operation can be calculated in one step and set $B$ is closed for this operation. If $X = \langle x_1, x_2, \ldots, x_p \rangle \in B$ array is input data, the parallel reduction will be

$$\langle x_1 \otimes x_2, \ldots, x_1 \otimes x_2 \otimes \ldots \otimes x_p \rangle$$

(7)

Implementation of this algorithm has been documented and optimized for various cases in [10], but these are not suitable for direct use in calculating the fitness function. They use the same binary, associative operation between every data; but fitness values are rarely built-up from such operations.

In many cases, the fitness function can be divided into the variation of simple sub-functions, as originally described.

$$\mathcal{F}(x) = \mathcal{F}_1(x_{v1}) \otimes_1 \mathcal{F}_2(x_{v2}) \otimes_2 \ldots \otimes_{n-1} \mathcal{F}_n(x_{vn})$$

(8)

where elements of $x_{v1}, x_{v2}, \ldots, x_{vn}$ vectors are variations of original $x$ vector's elements. The given sub-functions could again be divided into a variety of more straightforward functions. This should be repeated until the final hierarchy is obtained; a variety of simple functions depending on a maximum of two input variables and a constant vector $\alpha$.

$$\mathcal{F} = \mathcal{F}(x_i, x_j, \alpha)$$

(9)

where $i, j$ index. These functions should be calculated in one step.

Binary operators $\otimes_i$ also let us handle simple functions as described in (9). The given hierarchy of sub-functions can be described in a tree structure in Figure 3. Leaves are variations of elements of the $x$
vector. Nodes represent simple sub-functions. Inputs of nodes in the lower levels are the outputs of the previous level. Every function or binary operator(s) - represented as functions - depends only on the upper level. That means in practice, that each node on the same level can calculate in parallel independently of the others.

![Figure 3 Example hierarchy of sub-functions of original fitness in a tree structure](image)

C/C++ programming language and CUDA from compute capability version 2.0 support function pointers. This feature allows to store the address of functions in a variable - a pointer - and they are called as a value of a variable. Addresses of sub-functions (for example in Figure 3) are stored in a matrix.

\[
OP = \begin{bmatrix}
\mathcal{F}_{1,1} & \mathcal{F}_{1,2} & \mathcal{F}_{1,3} & \mathcal{F}_{1,4} \\
\mathcal{F}_{2,1} & \mathcal{F}_{2,2} & \text{null} & \text{null} \\
\mathcal{F}_{3,1} & \text{null} & \text{null} & \text{null}
\end{bmatrix}
\]  

Each row of the matrix contains the node's address, which can be calculated in parallel within the same step. Columns in a given row contain sub-functions working with two pieces of data from the results of the previous level. Using this approach, the original parallel reduction can be modified and implemented for solving problems with non-associative, non-homogeneous binary operations and functions.

The novelty of this new approach is that one can use parallel reduction, for non-associative operation and non-homogeneous sequence of operations.

5. Truck flatbed

In the present case, the chassis of the truck under investigation consists of two longitudinal steel beams. A three-layer platform is connected to this, through an intermediate support (Figure 4). Namely, the three layers are cross members, longitudinal members and floor slabs. The cross-braces are made of AlMgSi0.7 [13] and the floorboard is made of AlMg2.5 [14]. The structure is surrounded by a side frame, which carries the loads from other superstructures (roof, sidewalls, doors).
The purpose of the optimization is to reduce the cost of the truck platform material, by changing the cross-sectional dimensions of the cross members. The cross-sectional dimensions are marked and explained in Figure 5.

Previous calculations in [15] have shown, that the most favorable results in terms of mass for the tested original RHS cross-section, the I-section and C-section were, is the I-section. Therefore, in the following, the calculations were carried out only with the this.

The effective width of the deck plate is 50t, where t is the thickness of the deck plate. The geometric characteristics of the I-section, such as cross-sectional area, center of gravity distances and second-order torque, are:

\[ A = A_1 + A_2; \quad A_1 = h t_w + 2 b t_f; \quad A_2 = 50 t^2 \]  

(11)
\[
\begin{align*}
    y_G &= \frac{A_1}{A} \left( \frac{h + t + c}{2} \right); \quad y_c = h + c + \frac{t}{2} - y_G \\
    l_x &= \frac{h^3 t_w}{12} + \frac{b t_f h^2}{2} + A_1 \left( \frac{y_c - h}{2} \right)^2 + A_2 y_G^2
\end{align*}
\]

According to the symbols (7) to (9) and the previously stated goals, the fitness function of optimization is:

\[ F(\bar{x}) = \rho A_1 L_c n_c; \quad \bar{x} = [b, t_f] \] (14)

where \( \rho = 2.7 \times 10^{-6} \, \text{kg/mm}^3 \) is the density of aluminum, \( L_c = 2440 \text{mm} \) is the length of a cross member, and \( n_c \) is the number of cross members. It can be seen that only the typical dimensions of the belt plate change. The height of the backboard \( h = 100 \text{mm} \) is the same as the original RHS profile. Thickness \( t_w = 3.4 \text{mm} \) is the minimum required for manufacturing.

![Mechanical model of cross-member semi-console [15]](image)

The load on the crossbars, the superposition of two power systems, can be interpreted as a bending moment and a shear force (Figure 6). First force system distributed load due to payload weight.

\[ p = \frac{F_p n_p}{B L} \] (15)

where \( F_p = 8500 \text{N} \) is the assumed weight of the pallets, \( n_p = 5 \) is the number of pallets placed, \( B = 720 \text{mm} \) and \( L \) is the typical dimension of the semi-cantilevered platform. Load distributed along a line per cross member:

\[ p_c = \frac{p L}{n_c - 1} \] (16)

The second force system is the concentrated force \( F_1 = 1946 \text{N} \) due to the weight of the superstructure. In the horizontal position the bending moment

\[ M_h = \frac{p_c B^2}{2} + F_1 B = \frac{F_p n_p B}{2(n_c - 1)} + F_1 B \] (17)

the shear force

\[ Q = \frac{F_p n_p}{n_c - 1} + F_1 \] (18)

and bending stress

\[ \sigma = \frac{M_h}{l_x} \max(y_G, y_c) \] (19)

\[ \tau = \frac{Q}{ht_w} \] (20)

The limits of the optimization can be derived from the failure states. The first such limitation is the fatigue limit of the welds. According to [16] and [17], the allowed voltages are \( \sigma_c = 28 \text{MPa} \) and \( \tau_c = 28 \text{MPa} \) shear stress at \( 2 \times 10^6 \) cycles. This gives the value for the actual number of cycles \( N = 2 \times 10^5 \).

\[
\begin{align*}
    \log \Delta \sigma_N &= \frac{1}{3} \log \left( 2 \times 10^6 \right) + \log \sigma_c \\
    \log \Delta \tau_N &= \frac{1}{3} \log \left( 2 \times 10^6 \right) + \log \tau_c
\end{align*}
\]

and the limits are expressed in equations (19) to (22):
\[ g_1(\bar{x}) = \frac{Y_{MF}}{\Delta \sigma_N} - 1 \leq 0 \]  
\[ g_2(\bar{x}) = \frac{Y_{MF}}{\Delta \tau_N} - 1 \leq 0 \]  
where \( Y_{MF} = 1.25 \).

Further limitations arise from the stability conditions. Due to the condition of the deck plate:

\[ g_3(\bar{x}) = \frac{\beta h}{22t_w \varepsilon_f} - 1 \leq 0 \]  
where \( \beta \) and \( \varepsilon_f \) can be calculated as follows

\[ \beta = \begin{cases} 
0.65 + 0.35 \frac{y_0}{y_c} & \text{if } 1 > \frac{y_0}{y_c} \geq 0 \\
0.65 + 0.30 \frac{y_0}{y_c} & \text{if } 0 > \frac{y_0}{y_c} \geq -1 
\end{cases} \]  
\[ \varepsilon_f = \sqrt{\frac{250}{\sigma/\gamma_{ML}}} \]  
\[ y_0 = y_G - \frac{t}{2} - c \]

local buckling constraint for the flange plate

\[ g_4(\bar{x}) = \frac{b}{14t_f \varepsilon_f} - 1 \leq 0 \]  

6. The result of optimization

| \( n_c \) | Original RHS section [15] | I-section optima |
|-----|-----------------|-----------------|
| \( n_c \) | 14 | 12 | 10 | 16 | 14 | 12 | 10 | 8 |
| \( b \) [mm] | 55,0 | 115,0 | 120,0 | 73,9 | 66,1 | 80,8 | 78,1 | 74,3 |
| \( t_f \) [mm] | 5,4 | 3,0 | 3,4 | 4,6 | 5,9 | 5,6 | 7,0 | 9,4 |
| \( A_1 \) [mm²] | 1274 | 1370 | 1496 | 1019,88 | 1116,36 | 1246,57 | 1437,52 | 1738,32 |
| Mass [kg] | 117,50 | 108,31 | 98,56 | 107,50 | 102,96 | 98,55 | 94,70 | 91,62 |
| Material cost [$] | 202,10 | 186,28 | 169,51 | 184,90 | 177,09 | 169,50 | 162,89 | 157,58 |

The optimization was performed with \( n_c = 8,10,12,14,16 \) cross members. The results are summarized in Table 1. I-section with optimized dimensions results in weight reduction and cost savings over the original RHS section.

The indicated material cost represents the cost of all necessary cross-braces.

\[ K_m = k_m m_c = k_m \rho A_1 n_c L_c \]  
where \( k_m = 1.72 $/kg \) [8] is the specific material cost. The tooling cost has been neglected because, based on earlier calculations of [15] and [18], is negligible in relation to the cost of material per support.

As the number of cross members increases, the cross-sectional area decreases and their total weight increases. Further savings can be achieved by reducing the number of brackets even more, compared to the original \( n_c = 10,12,14 \). During optimization, the active constraint was the fatigue condition of the welded joints.
We have made a comparison with sequential and previously described parallel runs. You can see the result on Figure 7. The parallel version is faster than the sequential. Reachable time increase rapidly grows in function of population size.

![Figure 7 Comparison of sequential and parallel running](image)

7. Conclusion
In the optimization presented, the cross-sectional dimensions of a drawn aluminum, welded truck platform were determined. Limitations were made for the fatigue conditions of the local buckling of the section plates. Compared to the original RHS sectional structure, it resulted in both weight and cost savings for each cross member. This not only reduces production costs, but also decreases the operating costs.

In this article, nature-inspired evolutionary optimization algorithms are used, and the calculations are made using a GPU. The graphical processor unit needs a special programming language. The CUDA language is applied, to make parallel calculations for solving non-linear optimization problems. A special engineering optimization problem has been solved: A truck floor was optimized. The engineering problem showed that using massively parallel calculations, one can improve the calculation speed significantly. The time can be ~100 times less, compared to the original CPU calculations.

Acknowledgments

References
[1] Borgulya, I.: Optimization with Evolutionary Computation, Scholar’s Press, Saarbrücken (2015)
[2] Duarte, L.S., Waldemar, C., Anderson, P., Ivan, I.F., Glauco, P.: Polytop++: an efficient alternative for serial and parallel topology optimization on cpus & gpus, Structural and Multidisciplinary Optimization 52(5), 845–859 (2015). DOI 10.1007/s00158-015-1252-x
[3] Xia, Z., Wang, Y., Wang, Q., Mei, C.: Gpu parallel strategy for parameterized lsm-based topology optimization using isogeometric analysis. Struct. Multidiscip. Optim. 56(2), 413–434 (2017). DOI 10.1007/s00158-017-1672-x. URL https://doi.org/10.1007/s00158-017-1672-x
[4] Kalivarapu, V., Winer, E.: A study of graphics hardware accelerated particle swarm optimization with digital pheromones. Struct. Multidiscip. Optim. 51(6), 1281–1304 (2015). DOI 10.1007/s00158-014-1215-7. URL http://dx.doi.org/10.1007/s00158-014-1215-7
[5] Kan, G., He, X., Ding, L., Li, J., Hong, Y., Liang, K.: Heterogeneous parallel computing accelerated generalized likelihood uncertainty estimation (glue) method for fast hydrological
model uncertainty analysis purpose. *Engineering with Computers* **22** (2019). DOI 10.1007/s00366-018-0685-4

[6] Rothlin, M., Klippel, H., Afrasiabi, M., Wegener, K.: Metal cutting simulations using smoothed particle hydrodynamics on the gpu. *The International Journal of Advanced Manufacturing Technology* **102**(9), 3445–3457 (2019). DOI 10.1007/s00170-019-03410-0

[7] Wang, J., Zhang, D., Luo, M., Zhang, Y.: A gpu-based tool parameters optimization and tool orientation control method for four-axis milling with ball-end cutter. *The International Journal of Advanced Manufacturing Technology* **102**(5), 1107–1125 (2019). DOI 10.1007/s00170-018-2954-1

[8] Cheng, J., Grossman, M., McKercher, T.: Professional CUDA C programming. *John Wiley and Sons*, Inc., Indianapolis (2014)

[9] Sanders, J., Kandrot, E.: CUDA by example - An introduction to general purpose GPU programming. *Addison-Wesley*, Boston (2011)

[10] Martin, P.J., Ayuso, L.F., Torres, R., Gavilanes, A.: Algorithmic strategies for optimizing the parallel reduction primitive in cuda. *2012 International Conference on High Performance Computing Simulation (HPCS)*, pp. 511–519 (2012)

[11] Yang, X.S.: Flower pollination algorithm for global optimization. In: J. Durand-Lose, J. Nataša (eds.) *Unconventional Computation and Natural Computation*, pp. 240–249. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)

[12] Iványi, A., Lucz, L., Gombos, G., Matuszka, T.: Parallel enumeration of degree sequences of simple graphs. *II. Acta Universitatis Sapientiae*, Informatica 2, 254–270 (2013)

[13] DIN 1725-1983: Aluminiumlegierungen. Knetlegierungen, Knetlegierungen 1983

[14] DIN 1748-1983: Strangpressprofile aus Aluminium und Aluminium-Knetlegierungen. Eigenschaften, Zulässige Abweichungen, 1938

[15] Farkas J., Jármai K., Dúl R.: Minimum cost design of a truck floor welded from aluminium-alloy profiles, *Welding in the World*, Pergamon Press, Vol. 45, No. 9-10, (2001), pp. 19-22, ISSN 0043-2288

[16] Eurocode 3 Part 1.1.: Design of steel structures. General rules and rules for buildings, European Committee for Standardization, Brussels, 2005

[17] Hobbacher A.: iiw Recommendations for fatigue design of welded joints and components, *IIW-doc*, IIW-1823-07, ex XIII-2151r3-07/XV-1254r3-07

[18] Farkas J., Jármai K.: Truck floor design for minimum mass and cost using different materials, *Vehicle and Automotive Engineering*, Springer 2017, ISBN 978-3-319-51188-7