Jaroslav Mlýnek; Radek Srb; Roman Knobloch

The use of graphics card and nVidia CUDA architecture in the optimization of the heat radiation intensity

In: Jan Chleboun and Petr Přikryl and Karel Segeth and Jakub Šístek and Tomáš Vejchodský (eds.): Programs and Algorithms of Numerical Mathematics, Proceedings of Seminar. Dolní Maxov, June 8-13, 2014. Institute of Mathematics AS CR, Prague, 2015. pp. 150--155.

Persistent URL: http://dml.cz/dmlcz/702677

Programs and Algorithms of Numerical Matematics 17
J. Chleboun, P. Přikryl, K. Segeth, J. Šístek, T. Vejchodský (Eds.)
Institute of Mathematics AS CR, Prague 2015

THE USE OF GRAPHICS CARD AND NVIDIA CUDA ARCHITECTURE IN THE OPTIMIZATION OF THE HEAT RADIATION INTENSITY

Jaroslav Mlýnek[1], Radek Srb[2], Roman Knobloch[1]

[1] Department of Mathematics and Didactics of Mathematics, FP
jaroslav.mlynek@tul.cz, roman.knobloch@tul.cz
[2] Institute of Mechatronics and Computer Engineering
radek.srb@tul.cz
Technical University of Liberec
Studentská 2, 461 17 Liberec, Czech Republic

Abstract

The paper focuses on the acceleration of the computer optimization of heat radiation intensity on the mould surface. The mould is warmed up by infrared heaters positioned above the mould surface, and in this way artificial leathers in the automotive industry are produced (e.g. for car dashboards). The presented heating model allows us to specify the position of infrared heaters over the mould to obtain approximately even heat radiation intensity on the whole mould surface. In this way we can obtain the uniform material structure of artificial leather. The gradient methods are not suitable to optimize the position of heaters because the minimized function contains many local extremes. Therefore, we used an evolutionary algorithm, specifically the differential evolution algorithm. In this case the optimization procedure needs a lot of operations (especially when the mould volume is large and we use a large number of heaters). A substantial acceleration of the calculation can be achieved by parallel programming using a graphic card and nVidia CUDA architecture. The numerical calculations were performed by the Matlab code written by the authors and were run on a standard PC.

1. Introduction

The article describes the application of parallel programming for the utilization of a graphic card and nVidia CUDA architecture on a standard PC when calculating heat radiation intensity on a shell nickel mould surface and optimization of heat radiation intensity. In practice, a nickel mould is at first preheated by infrared heaters located above the outer mould surface. Then the inner mould surface is sprinkled with a special PVC powder and the outer mould surface is continually heated by infrared heaters.

The goal of the optimization is to determine the position of heaters over the mould so that their position ensures approximately the same heat radiation intensity on the whole mould surface. In this way we obtain uniform material structure

and colour tone of the artificial leather. During the optimization process we have to avoid possible collisions of two heaters as well as a heater and the mould surface. Therefore, the optimization process is more complicated. The minimized function has many local extremes and it is not suitable to use gradient methods in the optimization process. We used an evolutionary algorithm, specifically the differential evolution algorithm. Evolutionary algorithms generally require a lot of operations and long computation time (especially if the mould volume is larger and we use a higher number of heaters). This was the main reason for the use of parallel programming techniques.

In the following part of the article we focus on the implementation of parallel algorithms using the Matlab Parallel Computing Toolbox. The solved technical problem of the heat radiation intensity optimization, the used mathematical model and the calculation of the heat radiation intensity on a mould surface are described in more detail in [1] and [2].

2. Mathematical model and optimization of heat radiation intensity

The heater and the warmed mould are represented in 3-dimensional Euclidean space E_3 using the Cartesian coordinate system (O, x_1, x_2, x_3) with basis vectors $e_1 = (1, 0, 0), e_2 = (0, 1, 0), e_3 = (0, 0, 1)$.

The heater is represented by a straight line segment with a given length. The position of every heater Z can be defined by the following 6 parameters $Z : (s_1, s_2, s_3, u_1, u_2, \varphi)$, where the first three parameters are coordinates of the heater centre, the following two parameters are the first two coordinates of the unit vector u of the heat radiation direction (the third coordinate is negative, i.e. the heater radiates "downward") and the last parameter is the angle φ between the vertical projection of unit vector r of the heater axis onto the x_1x_2-plane and the positive part of axis x_1 (the vectors u and r are orthogonal).

The outer mould surface P is described by elementary surfaces p_j, where $1 \leq j \leq N$. It holds that $P = \cup p_j$, where $1 \leq j \leq N$ and $int\ p_i \cap int\ p_j = \emptyset$ for $i \neq j$, $1 \leq i, j \leq N$. Each elementary surface is described by the centre of gravity $T_j = [t_1^j, t_2^j, t_3^j]$, by the unit outer normal vector $v_j = (v_1^j, v_2^j, v_3^j)$ at the point T_j (we suppose v_j faces "upwards "and therefore is defined through the first two components v_1^j and v_2^j) and by the area of elementary surface w_j. Each elementary surface can thus be defined by the following 6 parameters $p_j : (t_1^j, t_2^j, t_3^j, v_1^j, v_2^j, w_j)$.

Now, we briefly describe the numerical computation procedure for the total heat radiation intensity on the mould surface. We denote L_j as the set of all heaters radiating on the jth elementary surface p_j $(1 \leq j \leq N)$ for the fixed position of heaters, and I_{jl} the heat radiation intensity of the lth heater on the p_j elementary surface (I_{jl} is a constant value on the whole p_j in our model). Then the total radiation intensity I_j on the elementary surface p_j is given by the following relation

$$I_j = \sum_{l \in L_j} I_{jl} \ . \tag{1}$$

The producer of artificial leathers recommends the constant value of heat radiation intensity I_{rec} on the whole outer mould surface. We can define F (respectively $\tilde{F}$), the deviation of the heat radiation intensity, by the relation

$$F = \frac{\sum_{j=1}^{N} |I_j - I_{\mathrm{rec}}| w_j}{\sum_{j=1}^{N} w_j}, \quad \tilde{F} = \sqrt{\sum_{j=1}^{N} (I_j - I_{\mathrm{rec}})^2 w_j} \,. \tag{2}$$

Function F defined by relation (2) (and analogously function $\tilde{F}$) has many local extremes. As we stated in this chapter, the position of every heater is defined by 6 parameters. Therefore, $6M$ parameters are necessary to define the position of all M heaters. We will successively construct a population of individuals y in the differential evolution optimization algorithm. Every population includes NP individuals, where every individual y represents one possible position of heaters above the mould. The generated individuals are saved in the matrix $\mathbf{B}_{NP\times(6M+1)}$. Every row of this matrix represents one individual, y, and its evaluation, $F(y)$. We seek the individual $y_{\min} \in C$ satisfying the condition

$$F(y_{\min}) = \min\{F(y); y \in C\}, \tag{3}$$

where $C \subset E_{6M}$ is the examined set. Every element of C is formed by a set of $6M$ allowable parameters and this set defines just one position of the heaters above the mould. The identification of the individual $y_{\min}$ defined by relation (3) is not realistic in practice. But we are able to determine an optimized solution y_{opt}.

3. Differential evolution algorithm and use of parallel programming

Now we describe schematically the particular steps of the differential evolution algorithm named *DE/rand/1/bin* (for more details see [3]) which is applied to our problem and was programmed in Matlab code by the authors.

Differential evolution algorithm

Input: the initial individual y_1, population size NP, the number of used heaters M (dimension of the problem is $6M$), crossover probability CR, mutation factor f, the number of calculated generations NG.

Internal computation:

1. create an initial generation ($G = 0$) of NP individuals $y_i^G, 1 \le i \le NP$,
2.a) evaluate all the individuals y_i^G of the generation G (calculate $F(y_i^G)$ for every individual y_i^G), b) store the individuals y_i^G and their evaluations $F(y_i^G)$ into the matrix $\mathbf{B}$,
3. *repeat until* $G \le NG$
a) *for* $i := 1$ *step* 1 *to* NP *do*
   (i) randomly select index $k_i \in \{1, 2, \ldots, 6M\}$,
   (ii) randomly select indexes $r_1, r_2, r_3 \in \{1, 2, \ldots, NP\}$,

where $r_t \neq i$ for $1 \leq t \leq 3$ and
$r_1 \neq r_2, r_1 \neq r_3, r_2 \neq r_3$;

(iii) *for* $j := 1$ *step* 1 *to* $6M$ *do*
if $(rand(0,1) \leq CR$ *or* $j = k_i)$ *then*
$y_{i,j}^{\text{trial}} := y_{r_3,j}^G + f\left(y_{r_1,j}^G - y_{r_2,j}^G\right)$
else
$y_{i,j}^{\text{trial}} := y_{i,j}^G$
end if
end for (j)

(iv) *if* $F\left(y_i^{\text{trial}}\right) \leq F\left(y_i^G\right)$ *then* $y_i^{G+1} := y_i^{\text{trial}}$
else
$y_i^{G+1} := y_i^G$

end for (i),

b) store individuals y_i^{G+1} and their evolutions $F\left(y_i^{G+1}\right)$ $(1 \leq i \leq NP)$ of the new generation $G+1$ into the matrix $\mathbf{B}$, $G := G+1$
end repeat.

Output:
the row of matrix $\mathbf{B}$ that contains the corresponding value $\min\{F(y_i^G); y_i^G \in \mathbf{B}\}$ represents the best found individual y_{opt}.
Note that function $rand(0,1)$ randomly chooses a number from the interval $\langle 0, 1\rangle$. The denomination $y_{i,j}^G$ means the jth component of an individual y_i^G in the Gth generation. The individual y_{opt} is the final optimized solution and includes information about the position of each heater.

The parallel programming tools (using the graphics card and nVidia CUDA architecture, see [4]) can be successfully applied in the optimization process. Randomly generated individuals y and their evaluation $F(y)$ (given by relation (2)) are completely independent. Therefore, it is appropriate to use *the central processing unit* (CPU) *for parallel computing* during the creation of a new generation G of individuals y. Calculation of the function value $F(y)$ given by relation (2) of a new individual y is numerically rather demanding. We gradually calculate the heat radiation intensity I_j at each elementary surface p_j given by relation (1). In doing so, we use the experimentally measured values of the heat radiation intensity in the neighbourhood of the heater when calculating the value I_{jl} in the relation (1). Calculations of heat radiation intensities I_j and I_k on different elementary surfaces p_j and p_k are completely independent. Thus we conveniently use the *graphics processing unit* (GPU) *for parallel computing* when evaluating the relation (1). It is tested whether two different heaters of individual y are in a collision or the heater and mould surface are in collision. If a collision occurs, the individual y is penalized and value $F(y)$ is significantly increased. The determination of heater collisions with two different elementary surfaces p_j and p_k is entirely independent and we also use GPU for parallel computing during testing possible collisions of all heaters (the position of heaters is given by individual y) with the mould surface.

4. Practical examples of the use of parallelization

We made calculations on a PC with CPU: IntelCore i7-3770 CPU @3.4 GHz, RAM: 32 GB and GPU: GeForce GTX 460. We choose the common input parameters of the algorithm in the following examples (Example 1, Example 2): CR (crossover factor) = 0.98, f (mutation factor) = 0.60, NP (population size) = 200 individuals. The initial individual y_1 represents even distribution of the heaters over the mould and in the plane parallel with $xy-$plane and in distance 10[cm] over the mould surface. Type of heater: capacity 1,600 [W], length 15 [cm], width 4 [cm].

Example 1

The heated surface is a part of a spherical surface, sphere centred at the origin of the coordinate system, radius of the sphere $r = 0.4$[m], the ground plan of the surface is 0.5×0.5[m^2], I_{rec} (recommended heat radiation intensity) = 68[kW/m^2], M(number of heaters) = 16, N (number of elementary surfaces) = 1,000, NG (number of calculated generations) = 10,000 . The value of the function F for the initial individual y_1 is $F(y_1) = 20.87$. We received y_{opt} with deviation $F(y_{\text{opt}}) = 1.72$ after creation of 10,000 generations. The position of heaters over the testing surface corresponding to the individual y_{opt} is shown on the left-hand side of Figure 1.

Real times of the calculations y_{opt} for different default parameters are shown in Table 1. The first column includes different numbers of elementary surfaces (N). The following columns contain the corresponding times of calculations when using ordinary calculation (column labelled CPU), GPU (labelled CPU+GPU), CPU with quad-core (labelled CPUPAR) and simultaneous use of a CPU with quad-core and GPU (labelled CPUPAR+GPU). The values given in parentheses from the third to the fifth column indicate the reduction of time calculation relative to the corresponding ordinary calculation. Time-consuming calculations in the table were estimated based on the average duration of one generation calculating.

Example 2

We will heat a shell nickel mould (see right-hand side part of Figure 1, this mould is used in production of artificial leathers for dashboards of passenger cars). The size of the mould is $1.5 \times 0.4 \times 0.4$[m^3], I_{rec} (recommended heat radiation intensity) = 68[kW/m^2], M(number of heaters) = 96, N (number of elementary surfaces) = 40,663, NG (number of calculated generations) = 20,000.

Number of elementary surfaces	CPU	CPU+GPU	CPUPAR	CPUPAR+ +GPU
10^3	9.98	7.03 (1.42x)	3.68 (2.71x)	2.87 (3.48x)
10^4	31.13	7.39 (4.21x)	10.39 (3.00x)	3.15 (9.90x)
10^5	261.37	9.80 (26.67x)	98.88(2.64x)	5.00 (52.26x)
10^6	2552.73	34.59 (73.81x)	1153.96 (2.21x)	27.24 (93.72x)

Table 1: Time ([h]) required for the optimization procedure for 10,000 generations

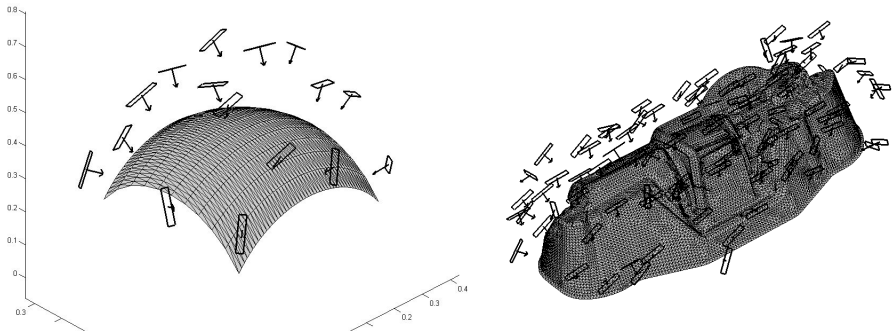

Figure 1: The position of the heaters corresponding to the individual y_{opt}.

The value of the function F for the initial individual y_1 is $F(y_1) = 40.14$. We received y_{opt} with deviation $F(y_{\text{opt}}) = 6.21$ after $20,000$ generations. The calculation lasted 41.06 hours using parallel computing (simultaneously GPU and CPU with quad-core). The calculation time with only "CPU" used would take 1,072 hours (44.66 days). The position of heaters over the mould surface corresponding to individual y_{opt} is shown on the right-hand side of Figure 1. The presented examples demonstrate that the computing time can be significantly reduced even on an ordinary PC with the use of parallel programming supported by a graphic card and nVidia CUDA architecture.

Acknowledgements

This work was supported by the grant SGS-FP-TUL 21049/2014 and by the grant SGS-FM-TUL, Technical University of Liberec.

References

[1] Mlýnek, J. and Srb, R.: The process of an optimized heat radiation intensity calculation on a mould surface. In: K. G. Troitzsch (Ed.), *Proc. of the 29th European Conference on Modelling and Simulation*, Digitaldruck Pirrot GmbH, Koblenz, Germany, pp. 461-467, May 2012.

[2] Mlýnek, J. and Srb, R.: The optimization of heat radiation intensity. In: J. Chleboun, K. Segeth (Eds.), *Proc. of the 16th Conference Programs and Algorithms of Numerical Mathematics*, Horní Maxov, pp. 142-148, June 2012.

[3] Price, K. V., Storn, R. M., and Lampien, J. A.: *Differential evolution.* Springer-Verlag Berlin, Heidelberg, 2005.

[4] Cook, S.: *CUDA programming: a developer's guide to parallel computing with GPUs.* Elsevier, Walthan, USA, 2013.