Publishing House
AKAPIT

COMPUTER METHODS IN MATERIALS SCIENCE
Informatyka w Technologii Materiałów

Vol. 15, 2015, No. 1

MULTI-FRONTAL PARALLEL DIRECT SOLVER FOR ONE DIMENSIONAL ISOGEOMETRIC COLLOCATION METHOD

PAWEŁ LIPSKI*, MACIEJ PASZYŃSKI

AGH University of Science and Technology, al. Mickiewicza 30, Krakow, Poland
**Corresponding author: lipski@student.agh.edu.pl*

Abstract

In this paper we present a new multi-frontal solver for the isogeometric collocation method (ISO-C) on GPU. The ISO-C method constitutes an alternative for the isogeometric finite element method (ISO-FEM). The key advantage of ISO-C over ISO-FEM is that it does not include the computationally intensive operation of integrating the variational formulation. The ISO-C method requires using only a single collocation point per one basis function, whereas in ISO-FEM, Gaussian quadrature is applied on many points at each finite element. The presented multi-frontal solver for collocation method results in logarithmic execution time assuming that large enough number of GPU processors is available. In this article, the method is employed for an exemplary 1D nanolithography problem of Step-and-Flash Imprint Lithography (SFIL). The algorithm, however, may be applied to a wide class of 2D and 3D problems.

Key words: step-and-flash imprint litography, collocation method, isogeometric analysis, GPU, CUDA

1. INTRODUCTION

Efficient solvers and fast-processing techniques for the problems of material science have always been of great importance. Currently, with the emergence of multi-core and graphic processor architectures, a tremendous speedup can be achieved. As a consequence, there is a need to develop new state-of-the-art algorithms that could leverage this new power. Several attempts have already been made to rewrite existing solutions to benefit from the new architectures (e.g. PLASMA and MAGMA – see Agullo et al., 2009). The shared-memory graphic cards have already proven a good environment for running multi-frontal direct solvers for isogeometric finite element method (Wozniak et al., 2014). In this paper we propose a preliminary version of the multi-frontal direct solver for isogeometric collocation method, an efficient alternative to classical isogeometric finite element method (Auricchio et al., 2010). The structure of the solver is based on the one developed for one dimensional finite difference method (Obrok et al., 2010). The numerical results presented in this paper concern the Step-and-Flash Imprint Litography (SFIL), a modern patterning process (Bailey et al., 2001a, Bailey et al., 2001b). In the paper we also analyze the convergence of the isogeometric collocation method for a uniform ("naive") selection of the collocation points and for Demko algorithm (Demko et al., 1985) for selection of the collocation points. We present that the Demko point selection guarantees unconditional convergence.

2. THE MODEL FOR POLYMER NETWORK DAMAGE PROBLEM

We focus on modeling the shrinkage of the feature after the photopolymerization occurred, still before removal of the template. It is assumed that there is a damage in the polymer network and thus the interparticle forces are weaker in one part of the domain, as shown in figure 1(a). For simplicity, we consider one-dimensional horizontal cross-section of the domain.

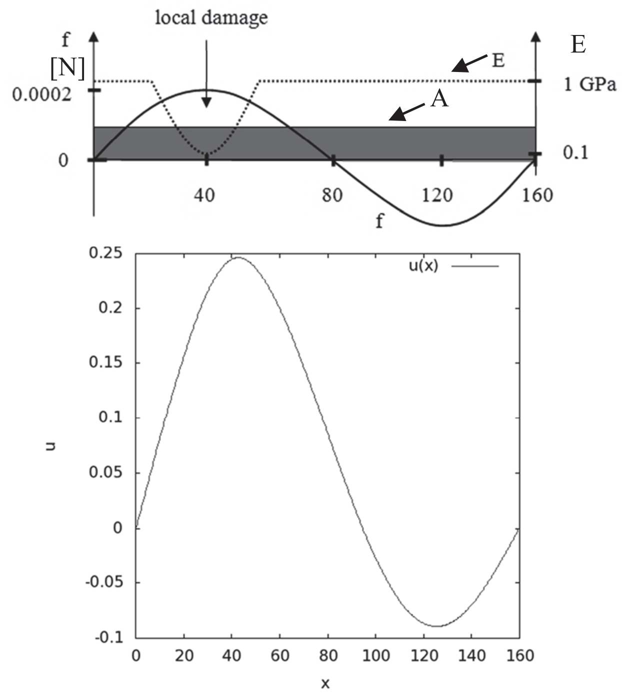


Fig. 1. Horizontal cross-section of the photopolymer (a), numerical result (b).

We need to solve the following linear elasticity equation:

$$-\frac{d}{dx}\left(EA\frac{du}{dx}\right) = f \tag{1}$$

where E stands for the Young modulus, A – for the cross-sectional area and f – for the body force. We seek u, which is vertical displacement field of the feature. The local damage is expressed by the E function. The boundary conditions are:

$$u(0) = u(160) = 0 \tag{2}$$

The material data are given by:

$$A(x) = 1, x \in [0,160] \tag{3}$$

$$E(x) = \begin{cases} 2.5\cdot 10^{-3}x^2 - 0.2x + 4, x \in (20,60) \\ 1, x \in [0,20] \cup [60,160] \end{cases} \tag{4}$$

$$f(x) = 0.0002\sin(\tfrac{2\pi x}{160}) \tag{5}$$

The spatial dimension is expressed in nanometers. The numerical result computed by using our GPU collocation method with fourth order B-splines is presented in figure 1(b).

3. ISOGEOMETRIC COLLOCATION METHOD

It is not only FEM that can be improved with the use of isogeometric analysis – the above concepts can as well be applied to the collocation method. In general, numerical solving of differential equations (either ordinary or partial) with collocation method (CM) is performed by choosing a space of candidate solutions (typically polynomials of relatively low degree) and then finding a linear combination of these candidates. This combination must satisfy given differential equations at certain points, called collocation points. In the isogeometric collocation method we provide a candidate space consisting of B-splines of an arbitrary order, from now on denoted as p. Each spline of order p spreads over p+1 basic intervals. The concept is illustrated in figure 2, where we have the following knot-vector for isogeometric computations: $\{0,0,0,0,1,2,3,4,4,4,4\}$. There are 7 uniform cubic B-spline basis functions in this example, denoted by $N_{i,p}$. In this case we need to utilize 7 collocation points (marked as c_i) to assure that the number of basis functions is equal to the number of equations. Then we approximate the solution with a linear combination of B-spline basis functions

$$u(x) \approx \sum_{i=1}^{7} B_i\, N_{i,p}(x) \tag{6}$$

We substitute the approximation to the equation:

$$\frac{d}{dx}\left(EA\,\frac{d}{dx}\left(\sum_{i=1}^{7} B_i\, N_{i,p}(x)\right)\right) = f(x) \tag{7}$$

and we get

$$\sum_{i=1}^{7} B_i \frac{d}{dx}\left(EA\,\frac{d}{dx}\left(N_{i,p}(x)\right)\right) = f(x) \tag{8}$$

Next, we select the collocation points $c_{i,i=1,\cdots,7}$ and we prescribe the equation (in the strong form) in the collocation points, to obtain

$$\sum_{i=1}^{7} B_i\,\frac{d}{dx}\left(EA\,\frac{d}{dx}\left(N_{i,p}(c_j)\right)\right) = f(c_j) \text{ for } j = 1,\dots,7 \tag{9}$$

Since at any given collocation point there are up to (p+1) non-zero B-spline basis functions (compare figure 2 where we have four B-spline basis functions at a point), we get a multi-diagonal matrix.

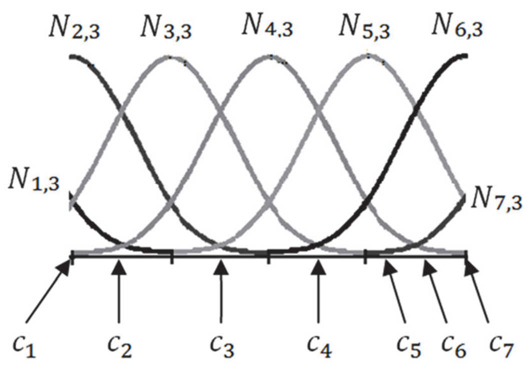


Fig. 2. *Basis B-spline functions.*

4. EMERGING ARCHITECTURES IN SCIENTIFIC COMPUTATIONS

Graphic Processing Units (GPUs) constitute a promising architecture for scientific computations since they allow to process data much faster when compared to the traditional CPU approach. GPUs feature large memory bandwidth and a substantial number of processing cores, allowing to perform the calculations in parallel on hundreds of threads. Compute Unified Device Architecture (CUDA) forms a platform for programming both CPUs and NVIDIA GPUs using a consistent programming model based off the C language with some extensions for expressing the parallelism.

CUDA treats the GPU as a coprocessor for the CPU. The CPU is called a *host*, whereas the GPU is called a *device*. The host plays the superior role, orchestrating computations and initiating memory transfers. Parallel fragments of the application are executed on the device as kernels, which, with respect to the source code, resemble serial programs. However, the GPU executes each kernel on many cores in parallel.

5. SOLVERALGORITHM

Let us focus on the isogeometric collocation method with quadratic B-splines over knot vector {0,0,0,1,2,3,4,5,5,5} and control points {0,1,2,3,4,5} which results in N+p=5+2=7 B-spline basis functions. In this case we select 5 collocation points located at the centers of elements {0.5,1.5,2.5,.3.5,4.5} and we get the following system of linear equations, completed with Dirichlet boundary conditions:

$$B_1 N_{1,2}(0) = 0$$
$$B_1 N''_{1,2}(c_1) + B_2 N''_{2,2}(c_1) + B_3 N''_{3,2}(c_1) = f(c_1)$$
$$B_2 N''_{2,2}(c_2) + B_3 N''_{3,2}(c_2) + B_4 N''_{4,2}(c_2) = f(c_2)$$
$$B_3 N''_{3,2}(c_3) + B_4 N''_{4,2}(c_3) + B_5 N''_{5,2}(c_3) = f(c_3)$$
$$B_4 N''_{4,2}(c_4) + B_5 N''_{5,2}(c_4) + B_6 N''_{6,2}(c_4) = f(c_4)$$
$$B_5 N''_{5,2}(c_5) + B_6 N''_{6,2}(c_5) + B_7 N''_{7,2}(c_5) = f(c_5)$$
$$B_7 N_{7,2}(160) = 0 \qquad (10)$$

By computing the derivatives of the B-spline basis functions at the collocation points, as well as the values at the boundary nodes, we get a tri-diagonal system of linear equations. The system is sparse and can be decomposed into the following set of subsystems:

$$\begin{bmatrix}1 & 0\\ 1 & -1\end{bmatrix}\begin{bmatrix}B_1\\ B_2\end{bmatrix} = \begin{bmatrix}0\\ 0.5f(c_1)\end{bmatrix} \quad \begin{bmatrix}-1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}B_2\\ B_3\end{bmatrix} = \begin{bmatrix}0.5f(c_1)\\ 0.5f(c_2)\end{bmatrix} \quad \begin{bmatrix}-1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}B_3\\ B_4\end{bmatrix} = \begin{bmatrix}0.5f(c_2)\\ 0.5f(c_3)\end{bmatrix}$$
$$\begin{bmatrix}-1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}B_4\\ B_5\end{bmatrix} = \begin{bmatrix}0.5f(c_4)\\ 0.5f(c_5)\end{bmatrix} \quad \begin{bmatrix}-1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}B_5\\ B_6\end{bmatrix} = \begin{bmatrix}0.5f(c_5)\\ 0.5f(c_6)\end{bmatrix} \quad \begin{bmatrix}-1 & 1\\ 0 & 1\end{bmatrix}\begin{bmatrix}B_6\\ B_7\end{bmatrix} = \begin{bmatrix}0.5f(c_6)\\ 0\end{bmatrix} \qquad (11)$$

These systems are equivalent to the original after summing up. We can then merge (which is denoted by $\oplus$ and $\xrightarrow[M]{}$) adjacent pairs of the systems, perform Gaussian elimination (denoted by $\xrightarrow[G]{}$) and eliminate one fully assembled node from each system (denoted by $\xrightarrow[E]{}$).

$$\left(\begin{bmatrix}1 & 0\\ 1 & -1\end{bmatrix}\begin{bmatrix}B_1\\ B_2\end{bmatrix} = \begin{bmatrix}0\\ 0.5f(c_1)\end{bmatrix}\right) \oplus \left(\begin{bmatrix}-1 & 1\\ 1 & -1\end{bmatrix}\begin{bmatrix}B_2\\ B_3\end{bmatrix} = \begin{bmatrix}0.5f(c_1)\\ 0.5f(c_2)\end{bmatrix}\right) \xrightarrow[M]{}$$

$$\begin{bmatrix}-2 & 1 & 1\\ 0 & 1 & 0\\ 1 & 0 & -1\end{bmatrix}\begin{bmatrix}B_2\\ B_1\\ B_3\end{bmatrix} = \begin{bmatrix}0.5f(c_2)\\ 0\\ 0.5f(c_3)\end{bmatrix} \xrightarrow[G]{}$$

$$\begin{bmatrix}1 & -\frac{1}{2} & -\frac{1}{2}\\ 0 & 1 & 0\\ 0 & \frac{1}{2} & -\frac{1}{2}\end{bmatrix}\begin{bmatrix}B_2\\ B_1\\ B_3\end{bmatrix} = \begin{bmatrix}-0.25f(c_2)\\ 0\\ 0.5f(c_3) + 0.25f(c_2)\end{bmatrix} \xrightarrow[E]{}$$

$$\begin{bmatrix}1 & 0\\ 1/2 & -1/2\end{bmatrix}\begin{bmatrix}B_1\\ B_3\end{bmatrix} = \begin{bmatrix}0\\ 0.5f(c_3) + 0.25f(c_2)\end{bmatrix} \qquad (12)$$

Similarly, for the other adjacent pairs, we perform Gaussian elimination and eliminate fully assembled equations (the latter operation not shown below):

$$\begin{bmatrix}-2 & 1 & 1\\ 1 & -1 & 0\\ 1 & 0 & -1\end{bmatrix}\begin{bmatrix}B_4\\ B_3\\ B_5\end{bmatrix} = \begin{bmatrix}0.5f(c_4)\\ 0.5f(c_3)\\ 0.5f(c_5)\end{bmatrix}$$

$$\underset{G}{\rightarrow}\begin{bmatrix}1 & -1/2 & -1/2\\ 0 & -1/2 & 1/2\\ 0 & 1/2 & -1/2\end{bmatrix}\begin{bmatrix}B_4\\ B_3\\ B_5\end{bmatrix} = \begin{bmatrix}-0.25f(c_4)\\ 0.5f(c_3)+0.25f(c_4)\\ 0.5f(c_5)+0.25f(c_4)\end{bmatrix}$$

$$\begin{bmatrix}-2 & 1 & 1\\ 1 & -1 & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}B_6\\ B_5\\ B_7\end{bmatrix} = \begin{bmatrix}0.5f(c_6)\\ 0.5f(c_5)\\ 0\end{bmatrix} \qquad (!3)$$

$$\underset{G}{\rightarrow}\begin{bmatrix}1 & -\frac{1}{2} & -\frac{1}{2}\\ 0 & -\frac{1}{2} & \frac{1}{2}\\ 0 & 0 & 0\end{bmatrix}\begin{bmatrix}B_6\\ B_5\\ B_7\end{bmatrix} = \begin{bmatrix}-0.25f(c_6)\\ 0.5f(c_5)+0.25f(c_6)\\ 0\end{bmatrix}$$

We end up again with three 2x2 sub-systems – one for B_1 and B_3, one for B_3 and B_5, one for B_5 and B_7. First two of them, which we consider as adjacent now, can be likewise merged on the next level:

$$\begin{bmatrix}-1 & 1/2 & 1/2\\ 0 & 1 & 0\\ 1/2 & 0 & -1/2\end{bmatrix}\begin{bmatrix}B_3\\ B_1\\ B_5\end{bmatrix} = \begin{bmatrix}f(c_3)+0.25f(c_2)+0.25f(c_4)\\ 0\\ f(c_5)+0.25f(c_4)+0.25f(6)\end{bmatrix}\ldots \qquad (14)$$

Again, we can eliminate the fully assembled first row, which yields a system for B_1 and B_5. The process of partial merging and eliminating continues up to the root of the tree. The process is followed by recursive backward substitutions. For higher order B-splines, the process requires merging more frontal matrices in order to get one fully assembled row, compare figure 3 for fourth order B-splines.

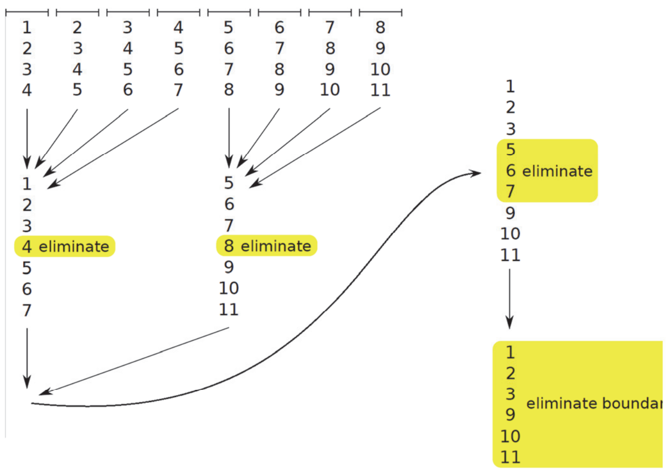


Fig. 3. Solver elimination pattern for fourth order B-splines.

6. IMPLEMENTATION CONCERNS

In the forward elimination phase, the number of nodes is gradually decreasing (about twice with each level) to reach one root node on the topmost level. This means the number of working cores falls down over the time. After each iteration, about half of the cores becomes idle (assuming that each core is assigned to a separate node).

On the contrary, in the backward substitution phase the algorithm starts from a single node (which is the root). This implies only one core is working and the rest remains idle. With each layer, twice as much cores as in the previous layer are assigned a node to process. This can be seen as a flow reverse to the flow of forward elimination.

The bottleneck of the algorithm is the topmost level of the forward elimination and the corresponding topmost level of the backward substitution. Regardless of the size of the input matrix, at this level only one thread can be active and any other cores must wait for the completion.

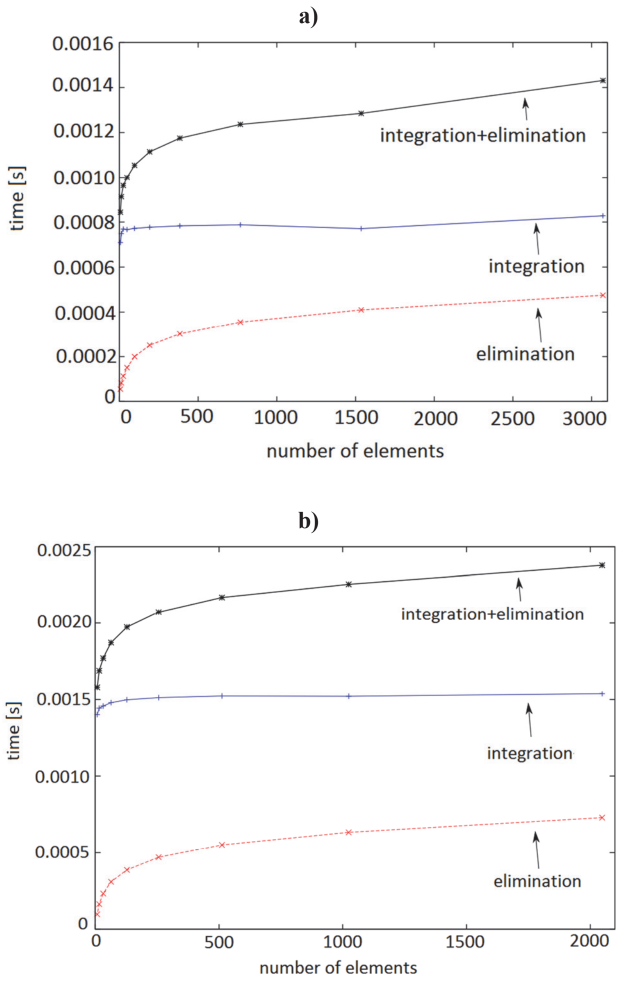


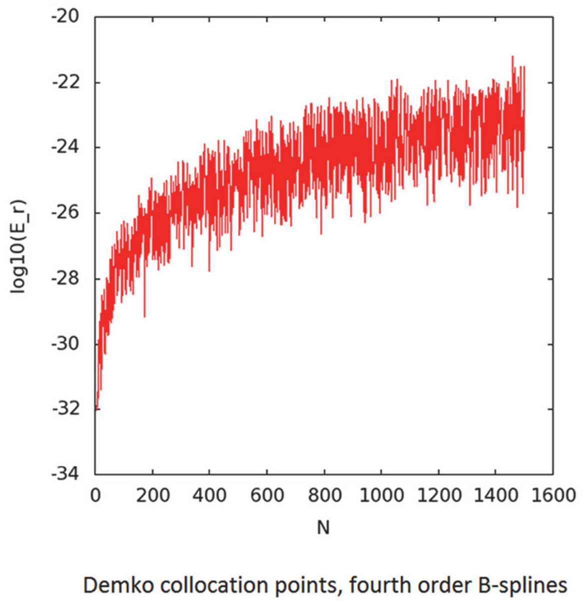
Fig. 4. Comparison of total execution time of ISO-FEM solver and ISO-C solver. Total ISO-FEM solver time = integration + elimination. Total ISO-C solver = elimination time only: quadratic B-splines (a), cubic B-splines (b).

7. NUMERICAL RESULTS AND CONCLUSIONS

The solution presented in figure 1b has been computed by a solver implemented using NVIDIA CUDA SDK 5.5 and run on Fedora 14.

Figures 4(a) and 4(b) point out the amount of time spent just on integration (generation of local systems) in the isogeometric finite element method in comparison to the elimination phase. In other words, switching from ISO-FEM to ISO-C method allows to save a substantial amount of time on the integration phase.

a)



b)

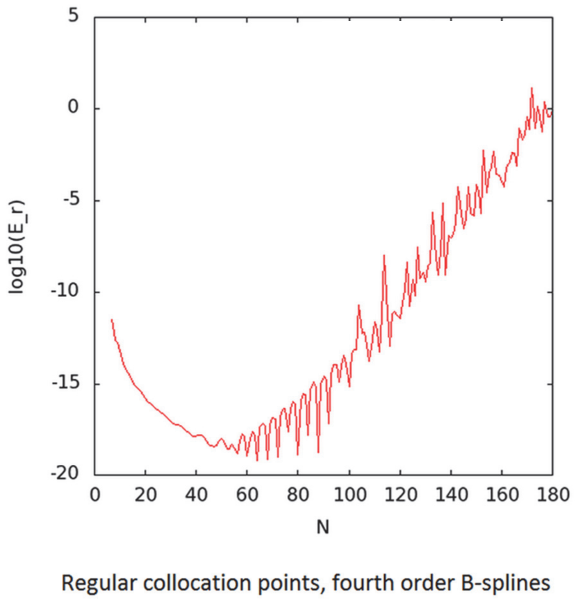


Fig. 5. Comparison of convergence for naive (a) and Demko (b) algorithms for selection of the collocation points.

Figures 4 (a) and (b) also show that the execution times of both solvers scale logarithmically when the number of cores is larger than the number of collocation points (ISO-C) or finite elements (ISO-FEM).

Figures 5 (a) and (b) show the common logarithm of relative error depending on the number of collocation points for fourth order B-splines.

In the figure 5(a), the points were chosen according to the Demko scheme. In the figure 5(b), the points on the right side were evenly distributed over the equation's interval. This is clearly complies with the statement that Demko collocation points guarantee convergence of the method.

ACKNOWLEDGEMENTS

The work presented in this paper has been supported by the National Science Center grant no. 2011/03/B/ST6/01393.

REFERENCES

Agullo, E., Demmel, J., Dongarra, J., Hadri, B., Kurzak, J., Langou, J., Ltaief, H., Luszczek, P., Tomov, S., 2009, Numerical Linear Algebra on Emerging Architectures: The PLASMA and MAGMA Projects, *SciDAC'09: Scientific Discovery through Advanced Computing*, San Diego, California, *J Phys Conf Ser*, 180, 012037.

Auricchio, F., Beirão da Veiga, L., Hughes, T. J. R., Reali, A., Sangalli G., 2010, Isogeometric collocation methods, *Math Mod Meth Appl S*, 20, 2075-2107.

Bailey, T., Smith, B., Choi, B. J., Colburn, M., Meissl, M., Sreenivasan, S. V., Ekerdt, J. G., Willson, C. G., 2001a, Step and flash imprint lithography: Defect analysis, *J Vac Sci Technol*, 19, 2806-2810.

Bailey, T., Smith, B., Choi, B. J., Colburn, M., Meissl, M., Sreenivasan, S. V., Ekerdt, J. G., Willson, C. G., 2001b, Characterization and modeling of volumetric and mechanical properties for step and flash imprint lithography photopolymers, *J Vac Sci Technol*, 19, 2685-2689.

Demko, S., 1985, On the existence of interpolation projectors onto spline spaces, *J Approx Theory*, 43, 151-156.

Obrok, P., Pierzchała, P., Szymczak, A., Paszyński, M., 2010, Graph grammar-based multi-thread multi-frontal parallel solver with trace theory-based scheduler, *Procedia Computer Science*, 1, 1993-2001.

Wozniak, M., Kuznik, K., Paszynski, M., Calo, V.M., Pardo, D., 2014, Computational cost estimates for parallel shared memory isogeometric multi-frontal solvers, *Comput Math Appl*, 67, 1864-1883.

WIELOFRONTALNY, RÓWNOLEGŁY SOLWER BEZPOŚREDNI DLA JEDNOWYMIAROWEJ METODY KOLOKACJI

Streszczenie

W artykule przedstawiamy nowy solwer wielofrontalny dla izogeometrycznej metody kolokacji (ISO-C) na GPU. Metoda ISO-C stanowi alternatywę dla izogeometrycznej metody elementów skończonych (ISO-FEM). Główną zaletą metody ISO-C jest redukcja znacznego kosztu obliczeniowego całkowania sformułowania wariacyjnego występującego w metodzie ISO-FEM. Metoda ISO-C wymaga bowiem użycia tylko jednego punktu kolokacji dla jednej funkcji bazowej, podczas gdy metoda ISO-FEM wiąże się z zastosowaniem kwadratury Gaussa w wielu punktach na każdym elemencie skończonym. Prezentowany solwer wielofrontalny dla metody kolokacji uzyskuje logarytmiczną złożoność obliczeniową przy założeniu odpowiednio dużej liczby procesorów graficznych GPU. Niniejsza publikacja przedstawia proste wykorzystanie metody dla jednowymiarowego przykładowego problemu nanolitografii Step-and-Flash Imprint Lithography (SFIL). Zaprezentowany algorytm znajduje jednak ogólnie zastosowanie dla szerokiej klasy problemów w dwóch i trzech wymiarach.

Received: September 21, 2014
Received in a revised form: November 21, 2014
Accepted: November 27, 2014