PARALLEL COMPUTING OF METAL FORMING SIMULATION IN QFORM SOFTWARE

Dmitry Gerasimov*, Arthur Gartvig

QuantorForm Ltd., Moscow, Russia
**Corresponding author: gerasimov@qform3d.com*

Abstract

In this study, the effectiveness of parallelization of data and tasks implemented in QForm software is investigated. The dependence of the number of simultaneously working logical processors or cores of a multi-core CPU on the solving time of metal forming simulation is shown. The simulation processes are parallelized in QForm software by means of Intel® Math Kernel Library, so the principle of parallelization is not described in this article. The aim was to show how the existing solution could be effectively used for simulation of metal forming processes and which solving processes can be parallelized.

Key words: numerical modeling, metal forming, QForm, parallelization of data and tasks

1. INTRODUCTION

Each year the demands on numerical simulation speed and accuracy increase. When it comes to finite element method (FE) the finer FE-mesh is the more accurate simulation results are and the longer numerical simulation takes.

The central processor unit (CPU) parameters primarily determine the speed of numerical simulation. Since multi-core CPUs are used in personal computers, the developers of simulation software face a problem of how to effectively parallelize a calculation process between cores (Schauer, 2008) or between logical processors (in the case of hyper-threading) (Marr et al., 2002) to fully utilize the capabilities of the CPU.

There are a number of researches related to investigation of parallelization algorithms at FEM simulation (Cârstea & Cârstea, 2008; Hoole, 1990; Wu et al., 2011). The results of experiments how the number of used cores or processors influences on solving time are shown in some researches (Butrylo et al., 2004; Meyer et al., 2012). For example, it is shown in (Choporov, 2013) that speedup ratio by simulation with 4 logical processors (2 physical cores + 2 virtual cores) in comparison with one logical processor equals about 1,6. Computer with following parameters was used in this experiment: Intel Core i3-380 (2.53 GHz) CPU, 3 GB RAM, openSUSE 12.2 operating system (compiler gcc 4.7), openMP library for parallelization management.

This paper describes how parallel processing by means of Intel® Math Kernel Library is used at simulation in QForm software (Biba & Stebunov, 2002) for numerical simulation of metal forming processes.

2. PARALLEL COMPUTING IN QFORM SOFTWARE

Two processes are launched after the start of a simulation in QForm software: *QSolver.exe* and *QFsolvhost.exe* (figure 1).

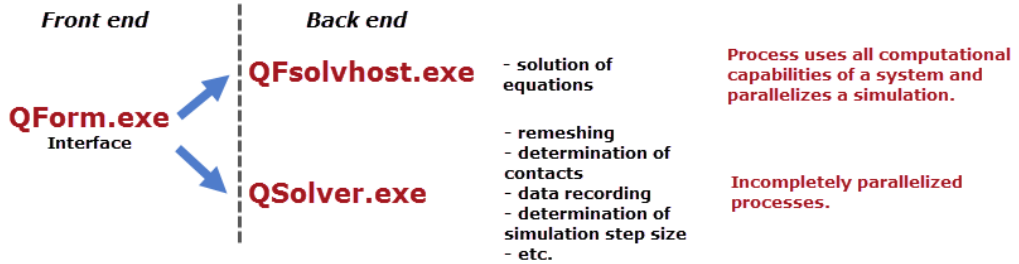

Fig. 1. *Parallelized and incompletely parallelized processes in QForm software*

The process *QFsolvhost.exe* solves systems of equations and it uses all computational capabilities of a system and parallelizes a simulation by means of Intel® Math Kernel Library.

The process *QSolver.exe* performs many necessary tasks such as: FE-mesh generation, determination of contacts between deformable shape and tools, recording of simulation results to the hard disk, determination of necessary simulation step size and other tasks. Some tasks performed with *QSolver.exe* are also parallelized, but the effect of this parallelization is not significant in comparison to the processes of *QFsolvhost.exe*.

3. EXPERIMENT

The computer used in the experiment had following parameters: processor Intel ® Core ™ i7-5960X CPU @ 3.00GHz (8 cores) with cache L1 = 512 KB, cache L2 = 2 MB, cache L3 = 20 MB, RAM 64 GB, operation system Windows 10 Pro x64.

To estimate how effectively the calculations are parallelized in QForm software, several simple three dimensional simulations of cylinder upsetting in flat dies were performed. To minimize the effect of some tasks using only one logical processor (*QSolver.exe*), the simulation was performed without influence of remeshing, friction and thermal problem.

Four cases with different mesh densities of the workpiece were investigated in this experiment: with finite element sizes of 20 mm, 15 mm, 10 mm and 7 mm. The solving time of only one simulation record was measured in the experiment: the height change from 305 mm to 300 mm (figure 2).

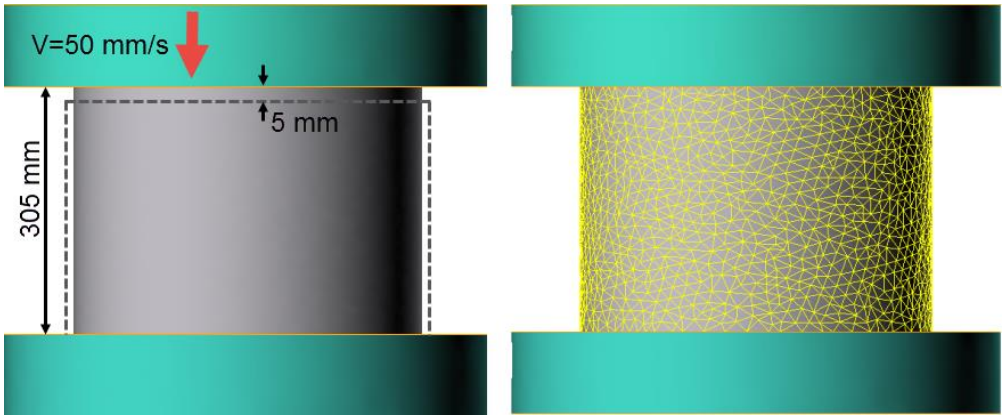

Fig. 2. *Scheme of upsetting (left) and finite element mesh with element size of 20 mm (right)*

Hyper-threading was activated in the first part of the experiment so the simulation process was parallelized in 16 logical processors with two logical processors for each core. By means of the command «set affinity» in task manager of Windows 10 operating system, a different number of logical processors can be activated for the process *QForm.exe*. It was investigated how the number of simultaneously activated logical processors (1, 4 and 16) influenced the simulation time. Time measured for all cases is shown in table 1.

Table 1. *Solving time by using of different number of logical processors. Hyper-threading: on.*

Size of element, mm	Number of FE-nodes	Number of elements	Solving time by using of different number of logical processors (LP), sec			Speedup ratio for 4 LP $t_{1\,LP}/t_{4\,LP}$	Speedup ratio for 16 LP $t_{1\,LP}/t_{16\,LP}$
			$t_{1\,LP}$	$t_{4\,LP}$	$t_{16\,LP}$		
20	9.430	50.500	12,4	6,1	2,6	**2,0**	**4,8**
15	21.800	120.300	57	27,1	9,6	**2,1**	**5,9**
10	68.700	396.100	499	221	73	**2,3**	**6,8**
7	194.500	1.146.000	4082	1722	509	**2,4**	**8,0**

Speedup ratio is an important index in measuring the performance of the parallel computing. In this paper speedup ratio for *n* logical processors is defined as the ratio of solving time with one logical

processor to solving time with *n* logical processors. The finer FE-mesh the more effectively parallel computing is performed in QForm.

Table 2. *Solving time by using of different number of cores. Hyper-threading: off.*

Size of element, mm	Number of FE-nodes	Number of elements	Solving time by using of different number of cores, sec			Speedup ratio for 4 cores	Speedup ratio for 8 cores
			$t_{1\ core}$	$t_{4\ cores}$	$t_{8\ cores}$	$t_{1\ core}/t_{4\ cores}$	$t_{1\ core}/t_{8\ cores}$
20	9.430	50.500	12,8	3,8	2,6	**3,4**	**4,9**
15	21.800	120.300	56,7	16	10,1	**3,5**	**5,6**
10	68.700	396.100	479	133	74	**3,6**	**6,5**
7	194.500	1.146.000	3658	1002	539	**3,7**	**6,8**

Additional numerical simulations with switched off hyper-threading were performed to estimate the benefit of hyper-threading. The results are shown in table 2. When simultaneously working of all cores the simulations with activated hyper-threading were 1-6% faster depending on FE-mesh density in the context of this experiment.

The larger the number of remeshings and simulation steps the more processes are not parallelized during modelling (figure 1). At the same time the effect of parallelization depends on FE-mesh density, how it was shown in the examples above. To estimate and show the speedup ratio for whole simulation process the solving time of fork forging simulation with different simulation parameters (figure 3) was investigated.

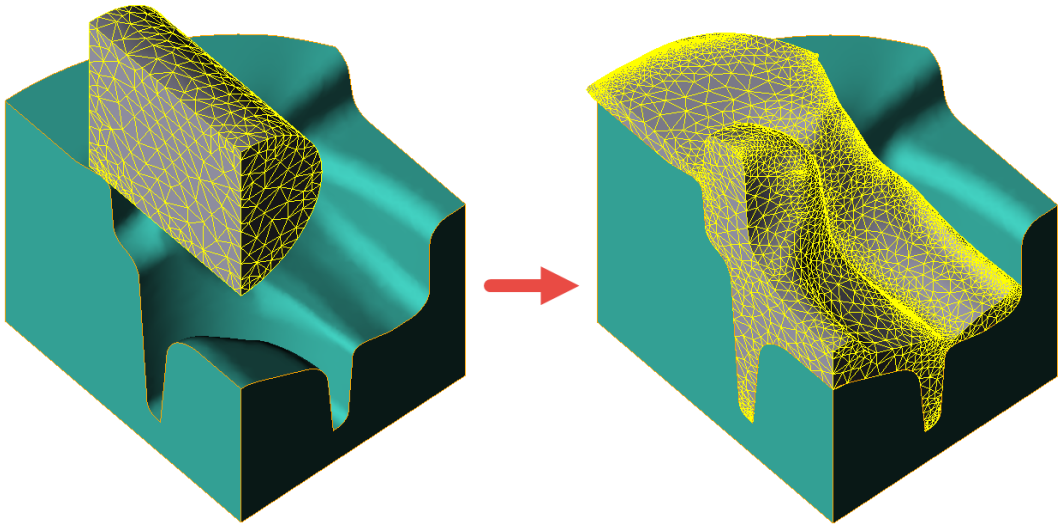

Fig. 3. *Results of bulk forging simulation. Mesh adaptation factor = 1*

Solving time of whole hot forging simulation process with different number of FE-nodes and simulation steps is shown in table 3.

Table 3. *Solving time of hot forging simulation by using of different number of cores. Hyper-threading: off.*

Adaptation factor, mm	Number of FE-nodes	Number of elements	Number of simulation steps	Solving time by using of different number of cores, sec		Speedup ratio for 8 cores
				$t_{1\ core}$	$t_{8\ cores}$	$t_{1\ core}/t_{8\ cores}$
1	12.800	58.000	168	2734	1113	**2,5**
2	56.000	267.000	250	33.012	11.546	**2,9**
3	126.000	615.000	371	179.625	51.325	**3,5**

In view of the fact that not all processes can be parallelized during simulation and that often it is necessary to investigate and to simulate several cases of one metal forming process (different die or

workpiece geometries, different initial parameters of deformed material, different friction conditions etc.) it makes sense to use the multitask possibility (simulation of several technological processes or cases simultaneously). Multitask feature can significantly increase the simulation efficiency. How the solving time depends on the number of simultaneously started identical simulations is shown in table 4. The simulation process showed in figure 3 with mesh adaptation factor equaled 1 (12.800 FE-nodes) was investigated in this experiment.

Table 4. *Solving time when using of multitask feature. 8-core processor was used for simulations.*

Number of simultaneously started simulations on one computer	Simulation time, min	Specific simulation time of one simulation process, min
1	19	**19**
2	25	**12,5**
4	34	**8,5**
8	65	**8,1**
16	118	**7,4**

4. CONCLUSION

The effectiveness of parallelization during FE simulation in QForm software depends generally on the FE-mesh density. Parallelization is more effective in simulations with a higher number of FE-nodes. Total number of non-parallelized or incompletely parallelized processes depends on the number of remeshings and simulation steps. Maximum measured speedup ratio equals 6,8 for 8-Core processor at full load with switched off hyper-threading. Calculation speed can be additionally increased by means of hyper-threading: maximum measured speedup ratio equals 8 for 8-Core processor at full load.

REFERENCES

Biba, N., Stebunov, S., 2002, 3D Finite Element Simulation of Material Flow, *Metallurgia*, Volume 69, No. 2, pp. FT8-FT10

Schauer, B., 2008, Multicore Processors – A Necessity, *Proquest Discovery Guides*, September, pp. 1-14.

Marr, D.T., Binns, F., Hill, D.L., Hinton, G., Koufaty, D.A., Miller, J.A., Upton, M., 2002, Hyper-Threading Technology Architecture and Microarchitecture, *Intel Technology Journal*, Volume 6, Issue 1.

Cârstea, T., Cârstea, D.P., 2008, Parallel computing in finite element applications, Proceedings of the 10th WSEAS International Conference on *MACMESE'08*, pp. 180-185

Hoole, S.R.H., Parallelism in Interactive Operations in Finite-Element Simulation, 1990, IEEE *Transactions on magnetics*, Volume 26, No. 4

Meyer, M., Sallwey, J., Blankenburg, R., Graeber, P.-W., 2012, Implementing Parallelism into an Unsaturated Soil Zone Simulation Model, *Boundary Field Problems and Computer Simulation*, pp. 25-29

Choporov, S., 2013, Parallel Computing Technologies in the Finite Element Method, Third International Conference "High Performance Computing" HPC-UA, pp. 85-91

Butrylo, B., Musy, F., Nicolas, L., Perrussel, R., Scorretti, R., Vollaire, C., 2004, A survey of parallel solvers for the finite element method in computational electromagnetics, *COMPEL: The International Journal for Computation and Mathematics in Electrical and Electronic Engineering*, Emerald, 23 (2), pp. 531-546.

Wu, X., Duan, B., Taylor, V., 2011, Parallel Finite Element Earthquake Rupture Simulations on Quad- and Hex-core Cray XT Systems, *the 53rd Cray User Group Conference (CUG2011)*