

Benefits of High Performance Computing applied to the numerical simulation of forged parts

Michel Pérémé, Stéphane Marie, Mickaël Barbelet, Etienne Perchat, Richard Ducloux, Lionel Fourment, Jean-Loup Chenot

▶ **To cite this version:**

Michel Pérémé, Stéphane Marie, Mickaël Barbelet, Etienne Perchat, Richard Ducloux, et al.. Benefits of High Performance Computing applied to the numerical simulation of forged parts. 20th International Forging Congress, Nov 2011, Hyderabad, India. 11 p. hal-00733962

HAL Id: hal-00733962
https://hal.science/hal-00733962

Submitted on 20 Sep 2012

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Benefits of High Performance Computing applied to the numerical simulation of forged parts

M. PEREME[1], S. MARIE[1], M. BARBELET[1], E. PERCHAT, R. DUCLOUX[1]

L. FOURMENT[2], J.-L. CHENOT[2]

[1]*Transvalor, Parc de Haute Technologie, 694 av du Dr Maurice Donat, 06255 Mougins, France*
[2]*Mines ParisTech, CEMEF, CNRS UMR 7635, BP 207, 06904 Sophia Antipolis Cedex, France*

Abstract: The use of numerical simulation in the forging and more generally in the forming industry has been continuously growing and spreading over the past fifteen years. From pure R&D tools in the 1980s and 1990s, these software packages are now used at the designer's level on a daily basis in an industrial and highly competitive environment. This has been achieved through ongoing software development in order to fully benefit from continuously faster computer hardware. Since the end of the 1990s the software package FORGE from Mines ParisTech CEMEF has been designed to support parallel processing. The benefits of this technology using today's multi-core systems are numerous, including drastically reducing computation times, simulating new challenging forming processes, using automatic optimization methods and investigating micro-structural aspects.

Keywords: Parallel solving, Automatic optimization, Bulk forming, FORGE® software, Forging, Open die forging, Closed-die forging.

Introduction

From pure R&D tools in the 1980s and 1990s, numerical simulation software packages are now commonly used on a daily basis in most bulk metal forming companies. This is partly due to a number of technical and financial factors in a globalized and highly competitive environment: in some industries, there is a constant need for reducing costs such as development time, raw material input and number of trial and errors, together with a requirement for quality improvement and innovation. At the same time, hardware platforms used to run numerical simulations have undergone a total revolution, from the original UNIX based RISC systems to most modern multi-core Windows and Linux technology. In order to successfully address the forming industry's new challenges, it is essential for numerical simulation software to fully benefit from these modern computer hardware systems by being able to use the computation power of their multi-core processors efficiently. Since the end of the 1990's the software package FORGE® from Mines ParisTech / CEMEF has been designed to support parallel processing. The benefits are numerous, including drastically reducing computation times, simulating new challenging forming processes, using automatic optimization methods and investigating micro-structural and final part properties aspects.

Parallel 3D forging code

In order to understand the implications of parallel processing on computation efficiency, we need to briefly describe the architecture of computer codes.

The numerical simulation of bulk metal forming processes requires to compute large deformations of metal. The deformation of material is mainly due to contact interactions with moving dies. A Finite Element formulation is used with a moving grid technique deformed as the material.

Space domain is discretized with robust linear elements triangle in 2D and tetrahedral in 3D and a mixed velocity-pressure Finite Element formulation is developed. Meshes are automatically adapted by topology remeshing techniques. The behavior of material, the contact conditions with dies, the thermo-mechanical coupling, lead to solve highly non-linear equations. At each time step, non linear systems are linearized using the classical Newton-Raphson method. Resulting well conditioned linear systems are then solved with iterative methods such as preconditioned conjugate minimal residual.

All these improvements have allowed the computer code FORGE® to be fully parallelized using a SPMD (Single Program on Multiple Data) strategy where each processor runs a full version of the code on a partitioned mesh; updating and synchronizing between sub-domains are done within matrix-vector operations in the iterative solver. The fully parallel mesh generator uses a combination of local remeshing and parallel repartitioning, where remeshing is done independently for each sub domain with fixed interfaces and parallel repartitioning is performed to move these interfaces in an optimal way.

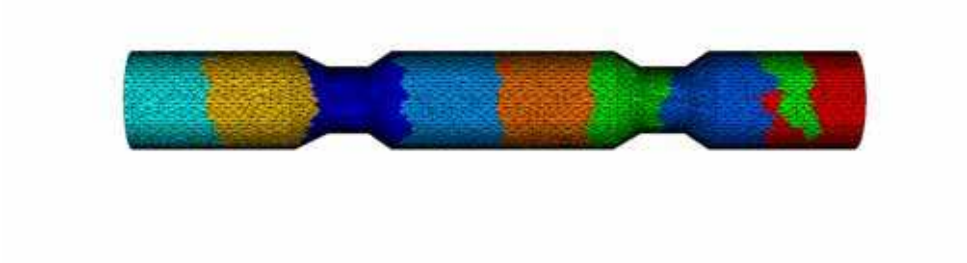


Figure 1: initial mesh of 47546 elements in 8 sub-domains.

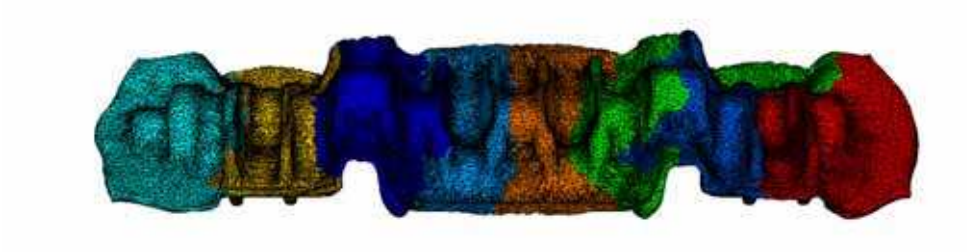


Figure 2: final mesh of 163888 elements in 8 sub-domains.

The SPMD strategy and the maximal parallel code FORGE® components is very efficient for simulations of very complex parts on a large number of cores as well on entry level parallel computers.

Parallel automatic optimization

Automatic optimization is the perfect complement to simulation.

An automatic optimization method based on evolution strategies assisted by meta-model technique has been developed in Forge®. This optimization strategy is quite robust with respect to local extrema and makes it possible to solve the most complex optimization problems in the metal forming area. A meta-modeling based on Kriging technique continuously evolves during generations and reduces the number of exact cost function evaluations.

Selection, recombination and mutation operators allow the evolution of following generations, each generation containing several different individuals. Exact cost function evaluation of an individual requires FORGE® calculations on specific parameters values.

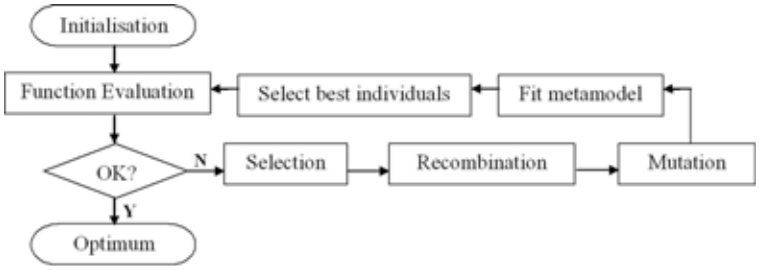


Figure 3: overview of the MAES algorithm.

This optimization strategy makes it easy to handle two levels of parallelization: the parallelization of the Finite Element software itself which has been discussed before and the parallelization of the optimization algorithm. This latest parallelism level is very efficient because of the genetic algorithm where several individuals are running simultaneously and independently on several computers. Generally the optimization evolution requires less than ten generations where each generation is formed by an exact number of individuals evaluated at the same time independently on parallel computer machines. The optimization CPU time is then divided by the number of individual per generation and optimization time amounts to ten FORGE® computations.

Examples of CPU-time reduction

The fully parallel strategy in the Finite Element code FORGE® opens the way to a drastic reduction of CPU time. Three different parts were simulated on different hardware, each time using from 1 to 16 cores. The efficiency of the parallel computer code was then measured.

The first example shows the final mesh of a brass valve core (Figure 4). The same simulation was run 5 times using 1, 2, 4, 8 and 16 cores of a recent workstation. The chart below (Figure 5) shows a sharp drop in computation time, from about 11 hours using 1 core to close to 1 hour using 16 cores and a speed-up reaching 12,07.

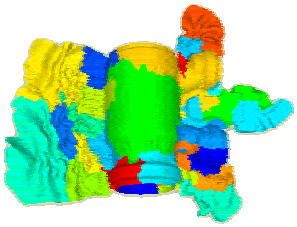


Figure 4: brass valve core.

Brass valve core					
# Core(s)	1	2	4	8	16
Time	11h 16m	5h 17m	3h 02m	1h 42m	0h 56m
Speed-up	1	2.13	3.71	6.63	12.07

Figure 5: CPU-time vs. number of core(s) used.

In the second example, the same exercise was conducted with the simulation of the hot forging of a steel spider (Figure 6). The initial mesh has 19646 elements. This time, the comparison was done using an 8-core workstation. The corresponding chart below (Figure 7) shows the impact of the number of core(s) used on the computation time.

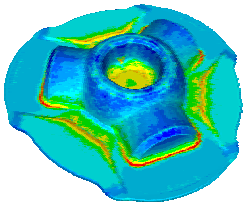


Figure 6: Spider.

Spider (Tripod)				
# Core(s)	1	2	4	8
Time	1h 56m	1h 11m	0h 42m	0h 21m
Speed-up	1	1.6	2.8	5.5

Figure 7: CPU-time vs. number of core(s) used.

The last example is about the hot forging of a 6-cylinder commercial vehicle crankshaft (Figure 8). The initial number of elements is 107108 and 246641 in the final mesh. The simulation was run on a 16 node cluster on 4, 8 and 16 cores. The computation times are shown in Figure 9.



Figure 8: Crankshaft.

Crankshaft			
# Core(s)	4	8	16
Time	10h 05m	5h 42m	3h 54m
Speed-up	1	1.77	2.58

Figure 9: CPU-time vs. number of core(s) used.

The previous examples show a good overall efficiency of the code. However there are some slight efficiency differences among these simulations. For instance, if we compare the efficiency of the code on the first two cases (Figure 10), we notice that FORGE® is more efficient in the brass valve case than in the spider case. The first reason to these differences is that the computations were all run on different hardware platforms: some on workstations, others on cluster systems. The second and main reason is that the level of efficiency of parallel computing is strongly dependent on the problem size vs. number of sub-domains. The bigger the problem, the more efficient is the parallel technology. In case of small meshes, the system spends a lot of time communicating between the different sub-domains, rather than in solving. In case of larger meshes, this communication time is small compared to solving time.

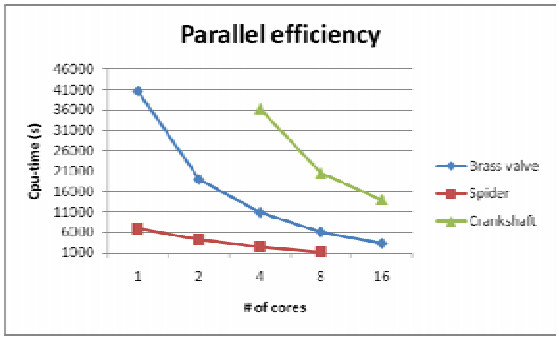


Figure 10: parallel efficiency on the three test cases.

Simulation of challenging forming processes

An additional benefit of high performance computing applied to forming simulation is the capability of the computer code FORGE® to model complex processes. This complexity can be of different natures such as long processes leading to large number of increments, e.g. ring rolling, requirement for a very fine mesh with millions of elements or processes involving sophisticated die kinematics. We can also add to these processes, challenging simulations such as microstructure modeling and coupled elasto-plastic die stress analysis.

This example involves a mix of fine mesh together with long process time and complex die kinematics. It was run as part of feasibility study at Transvalor. The process is an incremental

cold forming of a hollow shaft by rotary swagging. Rotary swagging is a free forming method for reducing cross sections of metal rod and tube material using two or more tool segments which partially or wholly surround the cross section to be reduced, acting simultaneously in the radial direction and in rotation relative to the workpiece (Figure 11). Rotary swaging is particularly advantageous for forming tubes, with or without internal profile.

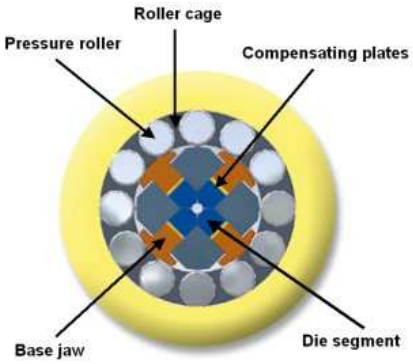


Figure 11: Design of a rotary swagging machine.

The goal of this study was to investigate whether FORGE® was able to simulate this very demanding process within acceptable computation time.

The simulation input parameters were the following: Elasto-Visco-Plastic material behavior for the workpiece, complex rigid die kinematics with a mix of translations and rotation of both the workpiece and the mandrel, a total number of 250 blows and an initial fine mesh of 495542 elements. The total process time was set to 8,8 seconds. The complete setup of the simulation is shown in Figure 12 below.

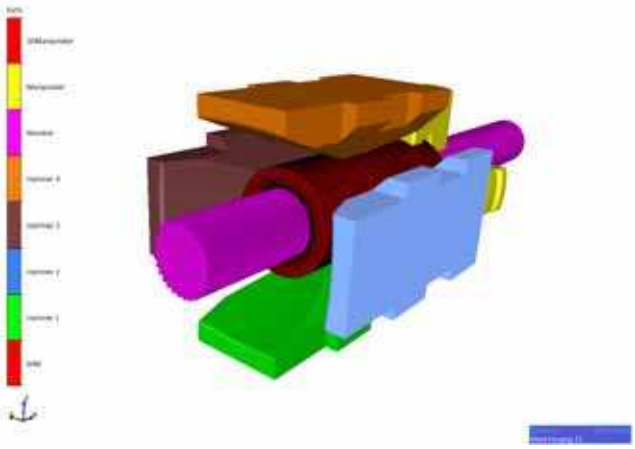


Figure 12: Setup of the rotary swagging process simulation.

The simulation ran on an 8-core server equiped with one Intel Xeon X5570 processor clocked at 3 Ghz with 24 Gb RAM. 4228 increments were necessary to complete the computation.

The final mesh of the workpiece consisted of 1214614 elements. Figure 13 below shows the simulation results in cross section.

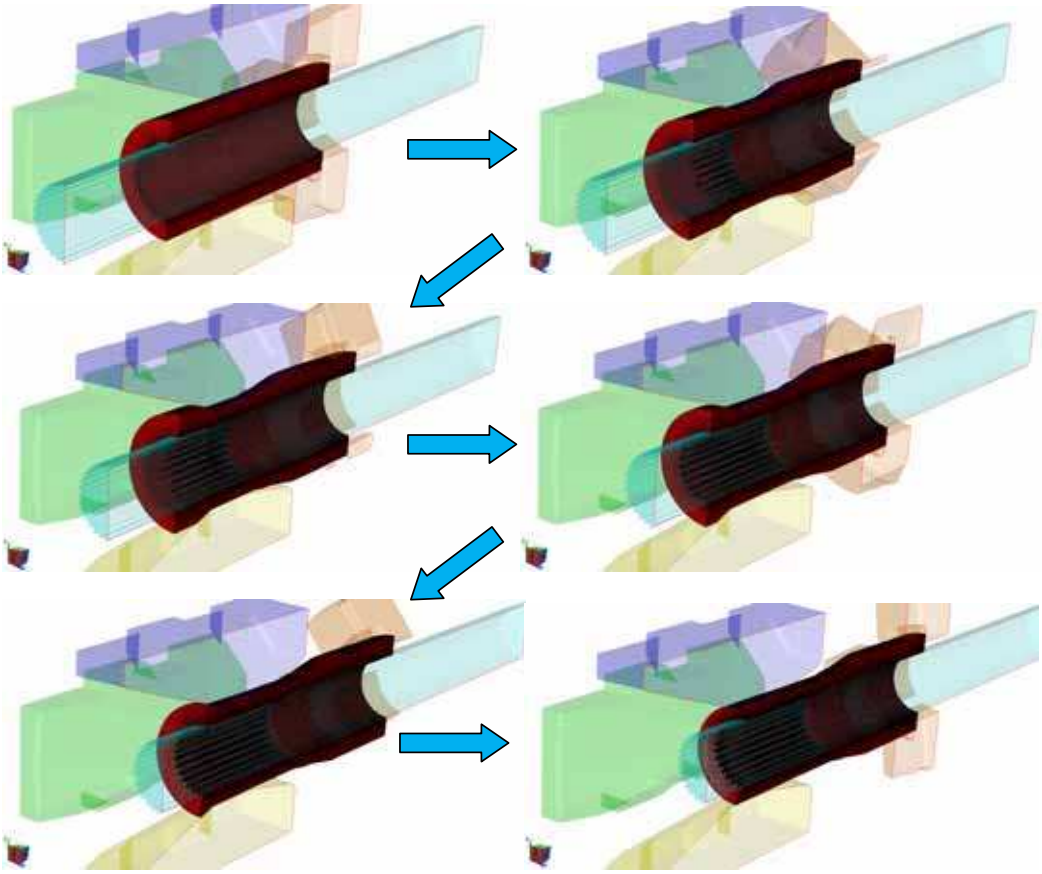


Figure 13: Simulation results - cross sections of the shaft.

Using such a refined mesh enables a detailed result analysis such as the prediction of cracks or an accurate estimation of filling vs. length (Figure 14).

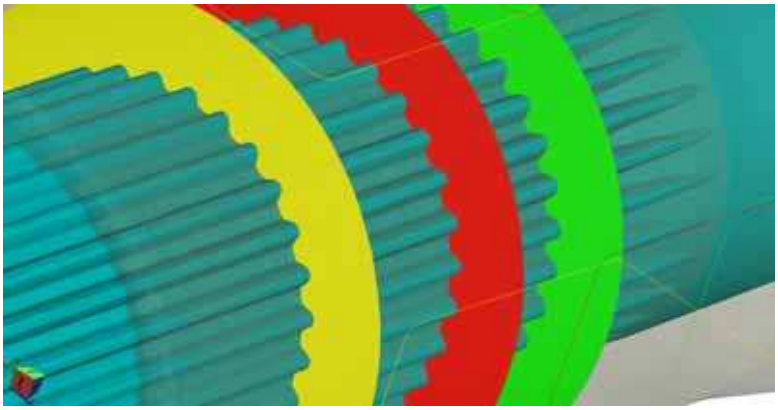


Figure 14: cross section along the shaft s length.

On 8 cores this simulation required 182 hours to run through. Considering the complexity of this process in terms of mesh size, die kinematics, material model and process time, this kind of computation time is acceptable.

HPC applied to automatic optimization

Back in 2009, a fully automatic optimization engine was introduced in the applied simulation program FORGE®. This evolutionary algorithm with evolution strategy and meta-model coupled with the numerical simulation computer code made it possible to optimize a situation under given constraints by finding ideal parameters automatically. These parameters or individuals belong to so-called generations, which as part of an iterative process, are combined and mutated until the best individuals are found. Obviously, this process requires a large number of iterations until optimized individuals are found. As standard, FORGE® suggests twice the number of parameters to be optimized over 10 generations, in order to be sure that the optimum is found. For example, to find an optimum of a process having two parameters to be optimized, the total number of chained simulations will be 40 (2x2x10). If the chained simulations contain three stages (upsetting, blocker and finisher), the total number of simulations for this particular optimization will be 120 computations.

Supprimé : genetic

Being also parallel, this optimization algorithm allows to run several individuals simultaneously, thus saving a lot of optimization time. This is particularly true for small to medium-size meshes, where as we explained earlier, the efficiency of using a large number of cores is low. In an ideal case, it is possible to run one full generation at a time. For example, using an 8-core machine, it is possible to run one full generation of a process having two parameters to optimize, each running on 2 cores of the workstation.

In the next example, the objective of the automatic optimization is to minimize the cut weight in order to save material. The component (a steel steering sector shaft) is forged on a hammer in 5 stages: 3 preforming stages, 1 blocker and 1 finisher. Additionally there is a cooling simulation in order to take into account any loss of temperature from the furnace to the first preforming stage (Figure 15). The input material is a rectangular billet (RCS). Transitions between each stage are managed automatically by the code.

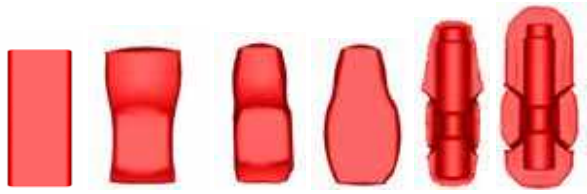


Figure 15: Forging sequence of the steering sector shaft

The parameters to be optimized are the billet length and width. The constraint is obviously to get a full filling of the finisher impression, using the lightest possible billet as input material. Before optimization, the billet dimensions were H=156,3mm and W=75mm (Figure 16), for a weight of 6,82 Kg.

Supprimé : ,

Supprimé : ,

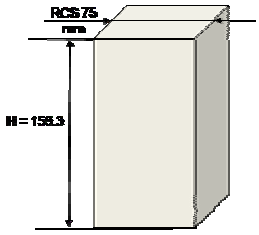


Figure 16: initial billet before optimization

The total number of simulations to be run is 240 (2x2x10x6). The hardware used is a 8-core workstation equiped with an Intel Xeon E5540 clocked at 2.53 Ghz and 24 Gb RAM.

Supprimé : ,

After 48 hours, the results of this optimization were available. Figure 17 below shows the flash pattern of the finisher stage before optimization in blue and after optimization in red.

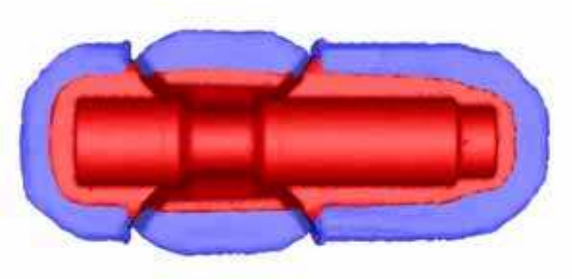


Figure 17: flash patterns of the finisher stage

The optimum dimensions of the billet found by the optimizer were H = 142mm and W=75mm as shown in Figure 17. The resulting weight of this new billet is 6.19 Kg which equals to a 9.18% material input saving, i.e. 0.63 Kg per piece. Two simulations were giving slightly better results, but the user has the opportunity to choose which of the suggested optima is compatible with an industrial process, especially when it comes to flash trimming. In this case, the best result had even less flash but was impossible to trimm in reality.

Supprimé : .

Supprimé : ,

Supprimé : ,

Supprimé : ,

Conclusion

As computer systems are becoming more and more powerful and the number of embedded cores larger and larger, it is essential for material forming simulation codes to be able to make use of this new power, in order for the forging industry to fully benefit from it. We have seen that FORGE® has been specifically designed for many years to be able to run efficiently on these systems so as to:

- Cut computation times drastically,
- Model demanding processes from the F.E.M. point of view,
- Optimize forming processes in reasonable times.

REFERENCES

[] T. Coupez. Stable Stabilized finite element for 3D forming calculation. Communication interne CEMEF, École nationale supérieure des mines de Paris, 1995

[] J.-L. Chenot and L. Fourment. Numerical formulations and algorithms for solving contact problems in metal forming simulation, Computational Mech., E. Oñate and S.R. Idelsohn (Eds), Barcelona, Spain, 1998.

[] T. Coupez and J-L. Chenot. Large deformation and automatic remeshing. In D.J.R. Owen and E. O ˜nate, editors, Computational Plasticity. Pineridge Press, 1992.

[] T. Coupez and J-L. Chenot. Mesh topology for mesh generation problems. In Chenot, Wood, and Zienkiewicz, editors, Numerical Methods in Industrial Forming Processes. Balkema, Rotterdam, 1992.

[] T. Coupez. Grandes Déformations et remaillages automatiques,Thèse de doctorat, École Nationale Supérieure des Mines de Paris. Novembre 1991.

[] T. Coupez and S. Marie, From a Direct Solver to a Parallel Iterative Solver in 3-D Forming Simulation, The International Journal of Supercomputer Applications and High Performance Computing,volume 11, number 4, pages 277-285, Winter 1997.

[] Marie S. Un modèle de parallélisation S.P.M.D. pour la simulation de procédés de mise en forme de matériaux. Thèse de Doctorat, ENSMP,CEMEF, 1997.

[] H. Digonnet. Repartionnement dynamique, mailleur parallèle et leurs applications à la simulation numérique en mise en forme des matériaux. Thèse de doctorat, École Nationale

[] Coupez T, Digonnet H., Parallel meshing and remeshing by repartioning, Applied Mathematical Modelling, 2000 ; 25:153-175.

[] M. Emmerich, A. Giotis, M Ozdemir, T. Bäck, K. Giannakoglou, Metamodel-assisted evolution strategies. In: Proceedings of the International Conference on Parallel Problem Solving from Nature, 2002

[] L. Fourment, T. Massé, S. Marie, R. Ducloux, M. Ejday, C. Bobadilla, P. Montmitonnet: Optimization of a range of 2D and 3D bulk forming processes by a meta-model assisted evolution strategy. 12th ESAFORM conference on material forming. Enschede, 27-29 April 2009

[] R.Ducloux, L. Fourment, S. Marie, D. Monnereau: Automatic optimisation techniques applied to large range of industrial test cases", 13th international ESAFORM Conference, Brescia, Italy, April 7-9, 2010

[] R.DUCLOUX, S. MARIE, D. MONNEREAU, N. BEHR, L. FOURMENT, M. EJDAY "Automatic optimization techniques applied to a large range of industrial test cases using metamodel assisted Evolutionary Algorithm" NUMIFORM 2010, June 13~17, 2010, Pohang, Korea

[] L. Fourment, R.Ducloux, S. Marie, M. Ejday, D. Monnereau, T. Masse, P. Montmitonnet: Mono and Multi Objective optimisation techniques applied to a large range of industrial test cases using

metamodel assisted evolutionary algorithms", 10th international NUMIFORM Conference, Pohang, Korea, June 13-17, 2010