Computational Mechanics Software as a Service Project

Carlos García Garino[1,2], Elina Pacini[1], David A. Monge[1], Claudio Careglio[1,2], and Anibal Mirasso[1,2]

[1]ITIC-UNCuyo, Mendoza, Argentina
[2]Facultad de Ingeniería-UNCuyo, Mendoza, Argentina

ABSTRACT

Cloud computing promises great opportunities for the execution of scientific and engineering applications. However, the execution of such kind of applications over Cloud infrastructures requires the accomplishment of many complex processes. In this paper we present a Computational Mechanics Software as a Service (SaaS) project which will allow scientists to easily configure and submit their experiments to be transparently executed on the Cloud. For this purpose, a finite element software called SOGDE is used to perform parametric studies of computational mechanics on the basis of underlying computing resources. Moreover, a web service provides an interface for the abovementioned functionalities allowing the remote execution of scientific applications in a simple way.
Keywords: SaaS, Computational Mechanics, Cloud Computing, Parametric Studies

1 INTRODUCTION

Cloud Computing [5] is a computing paradigm that has been recently incepted in the academic community [2]. Within a Cloud, services that represent computing resources, platforms or applications are provided across (sometimes geographically) dispersed organizations. Moreover, a Cloud provides resources in a highly dynamic and scalable way and offers to end-users a variety of services covering the entire computing stack. Besides, the spectrum of configuration options available to scientific users through Cloud services is wide enough to cover any specific need from their research. An example of scientific application from computational mechanics is parameter sweep experiments (PSE). PSEs consist of the repeated execution of the same application code with different input parameters resulting in different outputs. PSEs have the major advantage of generating lists of independent jobs. This makes these kinds of studies embarrassingly parallel from a computational perspective.

From the seminal paper on Computational Mechanics from Bathe and Oden [21], many advances can be cited. For the case of non-linear solid mechanics in general, and Finite Strain Plasticity in particular, a issue addressed in this paper, the literature has been benefited from the works of Simo and Ortiz [23, 24, 25]. In different practical applications problems it is important to study the sensitivity of results in terms of changes of variable data. For instance, García Garino et al. [11] have discussed the sensitivity of results of the necking problem of circular cylindrical bars in terms of applied imperfections. The non-linear finite element SOGDE [13, 14, 15] solver, developed by the authors is taken as basis for the proposed SaaS ongoing Project. SOGDE has been extended in a semiautomatic way in order to be processed in distributed environments in previous works of the authors [7, 6]. Section 2 provides the main concepts related to Cloud Computing and parametric studies. The discussion of PSEs is addressed in section 3 and application examples are presented in section 4. Finally, conclusions are provided in 5.

2 BACKGROUND

In this section a brief review of Cloud Computing and Computational Mechanics concepts are provided.

Cloud Computing

The growing popularity of Cloud Computing has led to several definitions, as previously indicated by Vaquero et al. [27]. Some of the definitions given by scientists in the area include:

- Buyya et al. [5] define Cloud Computing in terms of its utility to end users: "A Cloud is a market-oriented distributed computing system consisting of a collection of interconnected and virtualized computers that are dynamically provisioned and presented as one or more unified computing resource(s) based on service-level agreements established through negotiation between the service provider and consumers".

- On the other hand, Mell and Grance [20] define Cloud Computing as "a model for enabling ubiquitous, convenient, on demand network access to a shared pool of configurable computing resources (i.e. networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This Cloud model is composed of five essential characteristics, three services models (Software / Platform / Infrastructure as a Service), and four deployment models, whereas the five characteristics are: on-demand self-service, broad network access, resource pooling, rapid elasticity, and measured services. The deployment models include private, community, public and hybrid Clouds".

As suggested, central to Cloud Computing is the concept of *virtualization*, i.e., the capability of a software system of emulating various operating systems. In a Cloud, virtualization is an essential mechanism for providing resources flexibly to each user and isolating

security and stability issues from other users. Clouds allow the dynamic scaling of users applications by the provisioning of computing resources via *machine images*, or VMs. In addition, users can customize the execution environments or installed software in the VMs according to the needs of their experiments.

A Cloud offer to users its services according to three fundamental models [28]:

- Infrastructure as a Service (IaaS) is the most basic but at the same time ubiquitous model in which an IT infrastructure is deployed in a datacenter as VMs. An IaaS Cloud enables on-demand provisioning of computational resources in the form of VMs deployed in a datacenter, minimizing or even eliminating associated capital costs for Cloud consumers, and letting those consumers add or remove capacity from their IT infrastructure to meet peak or fluctuating service demands.

- Platform as a Service (PaaS) provides a computing platform and a solution stack as a service. In this model, the consumer creates the software using tools and/or libraries from the provider. The consumer also controls software deployment and configuration settings.

- Software as a Service (SaaS) provides ready-to-run services that are deployed and configured by the user. In general, the user has no control over the underlying Cloud infrastructure with the exception of limited configuration settings. Regarding scientific applications, SaaS represent an access point for the end user to reach a service, like a portal or a visualization tool. A strong characteristic of SaaS is that there is no client side software requirement. All data manipulated in such systems are held in remote infrastructures where all the processing takes place. One of the most prominent advantages of SaaS is that the applications have universal accessibility regardless of the client system's software availability. This scheme provides flexibility to the end user and transparency of any complex mechanism involved. Some widely used examples of services that belong to this category are Google Apps and Salesforce.

Computational Mechanics PSE

A concrete example of a PSE is the one presented by Careglio et al. [6], which consists in analyzing the influence of size and type of geometric imperfections in the response of a simple tensile test on steel bars subject to large deformations. To conduct the study, the authors numerically simulate the test by varying some parameters of interest, namely using different sizes and types of geometric imperfections. By varying these parameters, several case studies were obtained, which was necessary to analyze and run on different machines in parallel. More recently, García Garino et al. [12] have discussed a large strain viscoplastic constitutive model. A plane strain plate with a central circular hole under imposed displacements stretching the plate has been studied. Different values were considered for viscosity and other constitutive model parameters in order to adjust the model response. As can be seen in Figure 1 rather different deformation patterns have been found for different values of viscosity η.

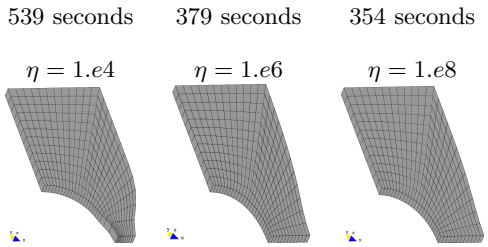

Figure 1: Deformed shapes for 2 m stretching: sensitivity of results in terms of viscosity parameter value

Consequently, different results can be expected for the different values of constitutive parameters considered, which in practice can led to significantly different CPU times in order to complete the execution of the associated numerical simulations. Even in the case of static assignation of computing resources [10, 19], a rather complex scheduling problem has to be solved. Particularly, the simulations in this work are based on a large strain elastoplastic/elastoviscoplastic constitutive model written in terms of internal variables theory and a hyperelastic free energy function [9, 12], following the ideas of Simo, Ortiz and co-authors [23, 24, 25]. It is important to mention that only few works devoted to the Finite Element Method (FEM) on Cloud Computing infrastructures can be found in the literature [1, 30].

3 COMPUTATIONAL MECHANICS PSE IN THE CLOUD

Scientists involved in this type of experiments need a computing environment that delivers large amounts of computational power over a long period of time. In general terms, such an environment is called a High Throughput Computing (HTC) environment. In HTC, jobs are dispatched to run independently on multiple computers at the same time. Interestingly, PSEs find their application in diverse scientific areas such as Bioinformatics [26], Earth Sciences [18], High-Energy Physics [4], Molecular Science [29] and even Social Sciences [3]. However, to deal with these problems, it is necessary large amounts of computational power. Cloud Computing [5] is a paradigm which suits well in solving the above cited computing problems, because of its promise of provisioning infinite resources.

In this paper we propose a computational mechanics based service to be implemented in a Cloud. Figure 2 presents a diagram of the software service for managing Parameter Sweep Experiments.

Parameter Sweep Experiments

Parameter Sweep Experiments (PSEs) is an experimental simulation-based methodology involving running the same application code several times with different input parameters to derive different outputs [31]. Running PSEs requires managing many independent jobs [22], since the experiments are executed under multiple initial configurations (input parameter values) a large number of times, to locate a particular point in the parameter space that satisfies certain user criteria. In addition, different PSEs have different number of parameters. This is a time-consuming task, which should be automated. However, it is not straightforward to provide a general solution, since each problem has a different number

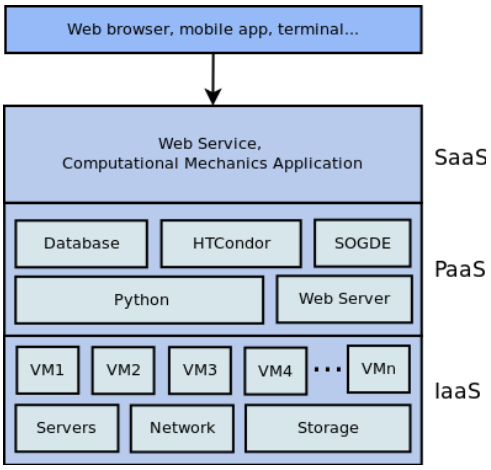

Figure 2: Schema of the proposed PSE SaaS in relation to the typical Cloud service models.

of parameters and each of them has its own variation interval.

In recent years, technologies such as Cloud Computing have been used for running such experiments. Typically, users exploit Clouds by requesting from them one or more machine images, which are virtual machines running a desired operating system on top of several physical machines (e.g. a datacenter). Interaction with a Cloud is performed by using Cloud services, which define the functional capabilities of a Cloud, i.e. machine image management, access to software/data, security, and so on.

Software Service for PSE

For delivering the functionalities of definition and analysis of PSEs in the context of Computational Mechanics we have developed a Web Service (WS). An approach based on WS was selected because they provide a standard way for the interoperability of different software systems. The main advantage of this approach lies on the fact that it permits the incorporation of the functionalities provided by our service into larger-scale experiments. The developed WS provides the following operations:

- load a finite element mesh file,
- define the parameters and ranges of values to analyze,
- select the finite element solver,
- define the number of computing nodes to use,
- retrieve the output results, and
- generate reports and graphics of interest.

The implementation of the WS has been performed by wrapping the parameter sweeping application presented on our previous works [7, 6]. Such application has been developed using the Python language.

Automation of PSEs

Conducting a study of this type involves processing a certain number of cases linked to the problem of interest. Figure 3 shows the classical steps of a finite element problem. Three different and typical stages

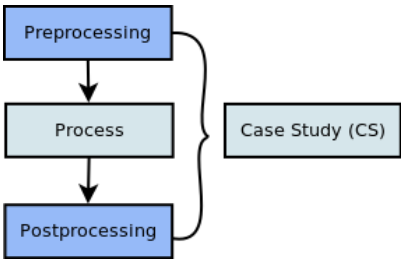

Figure 3: Typical FEM Case Study.

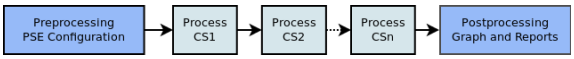

Figure 4: Sequential approach for solving case studies on PSEs.

are recognised: (*i*) *preprocessing* requires the preparation of the data to be used during the problem solving stage, (*ii*) *process* is the solution of the particular problem by using a code designed for that purpose, (*iii*) *postprocessing* is the manipulation of the output files from the task process in order to generate proper reports and graphs.

Regarding the study stages defined in Figure 3, when performing parametric studies significantly increases the amount of data files to prepare, as well as to simulate processes and post-processing work. Figure 4 shows how to perform a parametric study in a manual and sequential way providing *pre* and *postprocessing* tasks be unified.

On the downside, for users not proficient in distributed technologies, manually configuring PSEs is tedious, time-consuming and error-prone. As a consequence, users typically waste precious time that could be instead invested into analyzing results. The availability of elaborated GUIs that help in automating an experimentation process has in part mitigated this problem. However, the highly complex nature of today's experiments and thus their associated computational cost greatly surpasses the time savings that can be delivered by this automation.

To simplify the above mentioned difficulties, an application was developed in order to automate the steps of pre / post processing and to solve concurrently the various FEM simulation cases [7, 6]. This application uses the finite element code SOGDE [13, 14, 15], however any other non-linear general purpose FEM solver coud be used. Figure 5 shows a diagram where the different FEM simulations are concurrently solved and preprocessing and postprocessing are performed in an automatic a unified way, as it is discussed in next paragraphs.

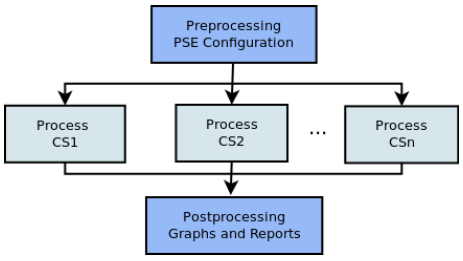

Figure 5: Parallel approach for solving PSEs.

To prevent the user should know discipline specific aspects of the application, we have developed a user-friendly interface that abstracts the complexities of using computer and tool that allows you to define, solve and visualize interactively and easily corresponding parametric study. However, the system also provides a web service which can be easily used for the interoperability with other systems.

Preprocessing: For each one of the different case studies processed different input parameters and data are required, denoted here *configurations*. The preprocessing stage addresses the problem of automatically generating the proper configurations to explore as part of the whole experiment. A simple approach would be to generate each file by separate using a suitable program such as GID [8], Gmsh [16], or similar. Nonetheless it is much more efficient to generate these files automatically varying the parameters of interest by an application designed for this purpose.

Distributed Execution: The different configurations generated for each case study are used in order to simulate the problems to analyze. As the number of case studies might be large and the amount of time required by each one may be big, usually the parallel execution of different FEM simulations could be advisable. This concurrent processing of the cases begins with the assembly of several packages comprising the files required for solving a subset of the cases to study. In other words, a package comprises the files for carrying the corresponding case studies:

- a subset of input data files,
- a copy of the finite element solver (SOGDE), and
- a copy of a script (called runtasks.py) in charge of running all the case studies for the given subset of case studies.

Generated packages are submitted to different computing nodes for the parallel processing of the case studies. Please note that the study cases on a particular node (all those from a particular package) are executed sequentially.

The distributed execution of the case studies is managed by the HTCondor[1] middleware. HTCondor is in charge of submitting each on of the packages to the computing nodes, start the executions of the runtask.py script, and returning the generated output files to the central computing node. The results obtained from every case o study are stored in a centralized database for the further postprocessing stage.

Postprocessing: Output data files corresponding to the analyzed cases are indexed in a database. From the output files, different interest results are analyzed to generate different curves. Furthermore, the generated files contain the commands necessary to generate these graphs. Curves generated are discussed in detail in the next section according to the applications studied.

4 PSE FOR THE NECKING PHENOMENA

The simple tension test is very useful problem in order to calibrate constitutive equations in presence of large strains. Then material parameters have to be adjusted in terms of experimental data. The simple tension test in presence of large strains show the necking phenomenon. A small imperfection is usually imposed to the specimen in the central zone in order to locate the necking there.

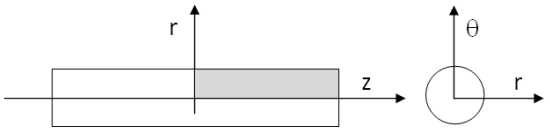

Figure 6: Longitudinal section of the circular cylindrical specimen.

In practice it is important to study the sensitivity of results in terms of the type and size of imposed imperfections. Finite element simulations are a valuable tool in order to process numerical studies. A HE30 aluminum circular cylindrical specimen is considered because experimental data due to Goicolea [17] are available. A longitudinal section of the specimen can be seen in the Figure 6. Due to symmetry conditions only a quarter of the specimen is modeled, shown as the shaded zone in the figure. The specimen has 75 mm length and perfect radius $R_0 = 8.1mm$. The finite element mesh used has 412 nodes and 360 $Q1/P0$ quadrilateral elements as can be seen in the Figure 7. The mesh has been properly refined in the central zone in order to capture the necking effect. The tension in the test is simulated imposing longitudinal displacements in the free end of the bar until a 10 mm value is reached.

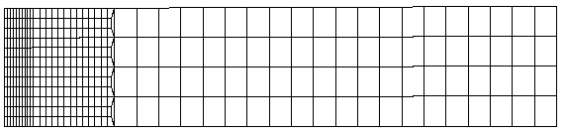

Figure 7: Finite element mesh for the cylindric bar.

Two different kind of imperfections, shown in Figure 8, are considered in order to study sensitivity of results. The first one is a linear variation of the radius along the length of the specimen that leds to a conical geometry. The second kind of imperfection considered is a sudden reduction of central diameter D.

For both kind of imperfections the parametric studies are based on different values of central diameter D in such way the dimensional parameter $\zeta = 1 - \frac{D}{D_0}$ varies from $1,852\%$ until $0,012\%$ taking into account 24 intermediate values. The smallest applied imperfection is very close to perfect specimen. García Garino et al. [9, 11] considered only 7 imperfect values, that are the largest ones in this study.

Axial Load Evolution

The first variable chosen to study the sensitivity of results is the evolution of applied load in terms of engineering deformation $\epsilon = \frac{\Delta L}{L_0}$ where ΔL denotes the imposed displacement of the bar and L_0 its initial length.

The Figure 9 show the evolution of applied load P in terms of engineering deformation ϵ for imperfection type 1. The upper graph corresponds to the

[1] research.cs.wisc.edu/htcondor

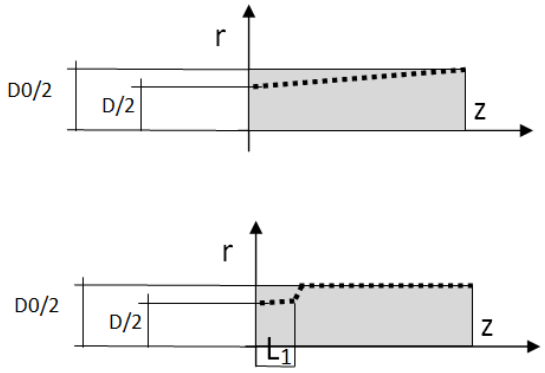

Figure 8: Different kind of imperfections.

lower initial radius of 8,099 mm; the lower one denotes the load for a initial radius of 7,95 mm and the other curves correspond to the intermediate values of the radius. As can be seen in the graph the applied imperfections practically do not affect results for precritical path. However for postcritical path the applied imperfections cause different results that can be clearly seen for the larger values of $\frac{\Delta L}{L_0}$.

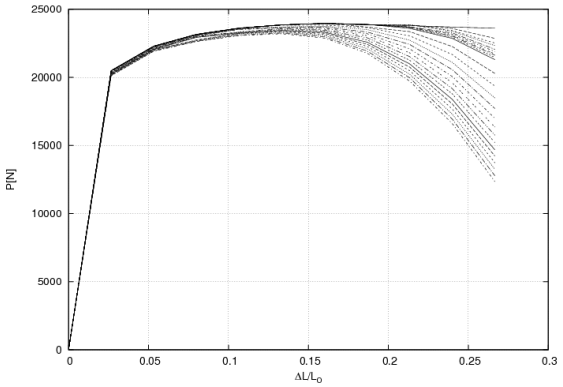

Figure 9: Evolution of Applied load P in terms of ε for imperfection type 1.

In the Figure 10, similar graphs to Figure 9 can be observed, but in this case for imperfection type 2. In this case the response of the bar is practically the same for values of of $\epsilon \leq 0,17$ approximately, but results are different for larger values of ϵ. For the maximum engineering deformation $\frac{\Delta L}{L_0} = 0,267$ that corresponds to an imposed displacement of 10 mm, the final applied load is higher than the obtained for imperfection type 1.

Necking Evolution

Another parameter of interest, typical for simple tension test simulations, is the evolution of necking at central zone in terms of the different imperfections considered, both for type and size. In the Figure 11 results computed with imperfection type 1 are shown. The y axis denote the necking ratio D/D_0 and the x axis the engineering deformation ε. D and D_0 account for the deformed and original diameter at central zone respectively.

Like in the previous figures the upper and lower curves correspond to smallest and largest value of imperfection respectively. The other graphs denote the response for intermediate values of imperfection. As can be seen in the picture for $\varepsilon \leq 1$ and the range of values for the considered imperfection, the deformed diameter practically does not change. However values of $\varepsilon \geq 0.15$ the size of the imperfection affects the ratio D/D_0, and it is even more remarkable for largest engineering deformation. It is important to point out that for the smallest values of the radius, 8,099 mm and 8,098 mm, that correspond to imperfections of 0,012% and 0,025% respectively, the response is very sensitive in terms of the size of considered imperfections, mainly for engineering deformations larger than 0.24, effect that can be seen in the Figure 11.

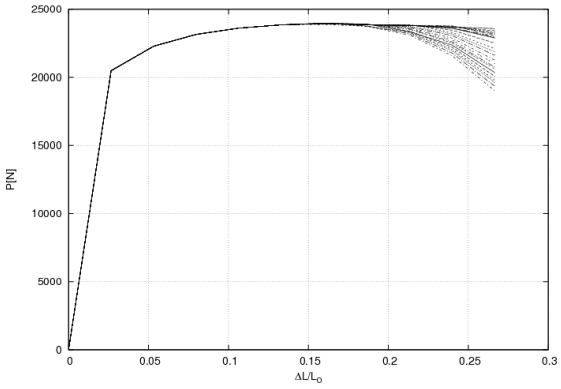

Figure 10: Evolution of Applied load P in terms of ε for imperfection type 2.

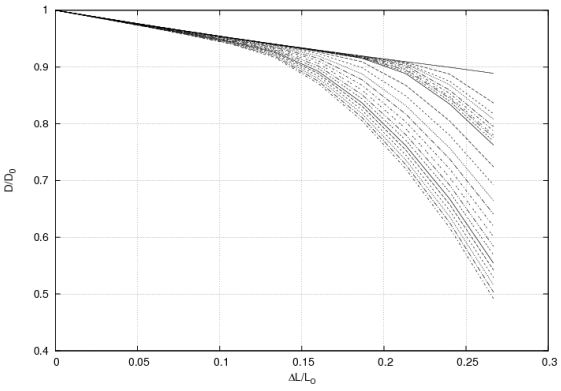

Figure 11: Necking evolution for imperfection type 1.

The necking evolution graph for the type of imperfection 2 is shown in Figure 12. For this case upper and lower graphs correspond to the maximum and minimal values of the radius 8,099 mm and 7,95 mm, respectively and the intermediate ones account for the other radius values.

In this case the applied imperfection practically don't affect the results for engineering deformations ε smaller than 0.16. For larger deformations the imperfections affect the computed results.

Final deformed shapes are shown in the Figure 13 for a imposed displacement of 10 mm and a initial radius of 7.95 mm. The upper graph corresponds to imperfection type 1 and the lower one to imperfection type 2.

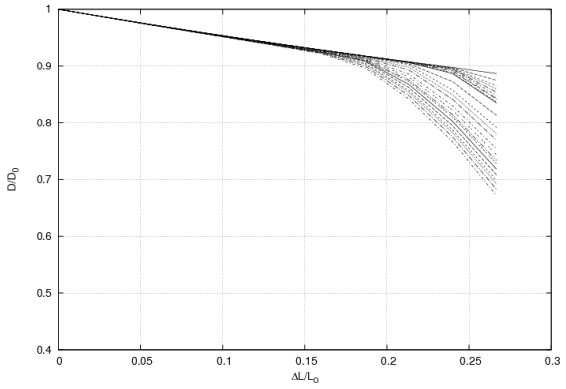

Figure 12: Necking evolution for imperfection type 2.

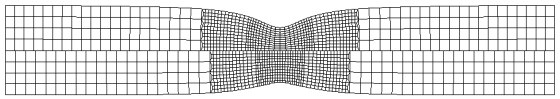

Figure 13: Deformed shapes for $\Delta l = 10$ mm and minimal initial radius 7.95. Lower and upper graphs correspond to type 1 and 2 of imperfection, respectively.

5 CONCLUSIONS

In this paper an ongoing Computational Mechanics Software as a Service project was presented. The web service allows the design and automatic execution of parametric studies using distributed resources from a Cloud. For such purpose, a non-linear finite element solver has been enhanced in order to include parameter sweeping capabilities. The manual preparation –and therefore time-consuming and error-prone– of input data as well as the execution of the jobs part of the parametric study is managed in an automatic fashion. In this way, errors proper of manual tasks can be circumvented and consequently, valuable time can be saved for the analysis of results. Finally, we have presented application examples which show the viability and advantages of our approach in a variety of problems. For extending this work, we are currently working on the performance evaluation of the presented software components.

Acknowledgments
We acknowledge the financial support provided by the National University of Cuyo project 06/B253 and ANPCyT, project PAE-PICT 2312. The second author acknowledges her Ph.D. fellowships granted by the PRH-UNCuyo Project and the National Scientific and Technological Research Council (CONICET).

REFERENCES

[1] I. Ari and N. Muhtaroglu. Design and implementation of a Cloud Computing service for finite element analysis. *Advances in Engineering Software*, 60(0):122–135, 2013.

[2] M. Armbrust, A. Fox, R. Griffithn, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. Above the Clouds: A Berkeley view of Cloud Computing. Technical Report UCB/EECS-2009-28, EECS Department, University of California, Feb 2009.

[3] R. Axelrod. The dissemination of culture: A model with local convergence and global polarization. *Journal of Conflict Resolution*, 41(2):203–226, 1997.

[4] J. Basney, M. Livny, and P. Mazzanti. Harnessing the capacity of Computational Grids for High Energy Physics. In *Conference on Computing in High Energy and Nuclear Physics*, pages 610–613, Padova, Italy, 7-11, February 2000.

[5] R. Buyya, C. Yeo, S. Venugopal, J. Broberg, and I. Brandic. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6):599–616, 2009.

[6] C. Careglio, D. Monge, E. Pacini, C. Mateos, A. Mirasso, and C. García Garino. Sensibilidad de resultados del ensayo de tracción simple frente a diferentes tamaños y tipos de imperfecciones. *Mecánica Computacional*, XXIX(41):4181–4197, 2010.

[7] C. Catania, C. Careglio, D. Monge, P. Martinez, A. Mirasso, and C. García Garino. Estudios paramétricos de mecánica de sólidos en entornos de computación distribuida. In A. Cardona, M. Storti, and C. Zuppa, editors, *Mecánica Computacional*, volume XXVII, pages 1063–1084, San Luis, 2008. AMCA.

[8] CIMNE. GiD 11 Reference Manual, Pre and post processing system for Numerical Simulations. International Center For Numerical Methods In Engineering, Available online: http://gid.cimne.upc.es/, 2013.

[9] C. García Garino, F. Gabaldón, and J. M. Goicolea. Finite element simulation of the simple tension test in metals. *Finite Elements in Analysis and Design*, 42(13):1187–1197, 2006.

[10] C. García Garino, C. Mateos, and E. Pacini. Job scheduling of parametric computational mechanics studies on cloud computing infrastructures. International Advanced Research Workshop on High Performance Computing, Grid and Clouds. Cetraro (Italy). Available online: http://www.hpcc.unical.it/hpc2012/pdfs/garciagarino.pdf, June 2012.

[11] C. García Garino, A. Mirasso, S. Raichman, and J.M. Goicolea. Imperfection sensitivity analysis of necking instability of circular cilyndrical bars. In D. R. J. Owen et al., editor, *Computational Plasticity: Fundamentals and Applications*, volume 1, pages 759–764. International Center for Numerical Methods in Engineering (CIMNE), 1997.

[12] C. García Garino, M.S. Ribero Vairo, S. Andía Fagés, A.E. Mirasso, and J.-P. Ponthot. Numerical simulation of finite strain viscoplastic problems. *Journal of Computational and Applied Mathematics*, 246:174–184, July 2013.

[13] C. García Garino. *Un modelo numérico para el análsis de sólidos elastoplásticos sometidos a*

grandes deformaciones. PhD thesis, Universidad Politécnica de Catalunya, Catalunya, Barcelona, 1993.

[14] C. García Garino and J. Oliver. Un modelo constitutivo para el análisis de sólidos elastoplásticos sometidos a grandes deformaciones: Parte i formulación teórica y aplicacón a metales. 11(1):105–122, 1995.

[15] C. García Garino and J. Oliver. Un modelo constitutivo para el análisis de sólidos elastoplásticos sometidos a grandes deformaciones: Parte ii implementación numérica y ejemplos de aplicación. 12(2):147–169, 1996.

[16] C. Geuzaine and J.-F. Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[17] J. Goicolea. *Numerical modelling in large strain plasticity with application to tube collapse analysis*. PhD thesis, PhD. Thesis, University of London, 1985.

[18] M. Gulamali, A. Mcgough, S. Newhouse, and J. Darlington. Using ICENI to run parameter sweep applications across multiple Grid resources. In *Global Grid Forum 10, Case Studies on Grid Applications Workshop*, Berlin, Germany, 2004.

[19] C. Mateos, E. Pacini, and C. García Garino. An ACO-inspired Algorithm for Minimizing Weighted Flowtime in Cloud-based Parameter Sweep Experiments. *Advances in Engineering Software*, 56:38–50, 2013.

[20] P. Mell and T. Grance. The NIST Definition of Cloud Computing. *National Institute of Standards and Technology*, 53(6):50, 2009. ISSN: 10476210.

[21] J. T. Oden and K. J. Bathe. A Commentary on Computational Mechanics. *Applied Mechanics Reviews*, 31(8):1053–1058, September 1978.

[22] M. Samples, J. Daida, M. Byom, and M. Pizzimenti. Parameter sweeps for exploring GP parameters. In *Conference on Genetic and Evolutionary Computation*, GECCO '05, pages 212–219, New York, USA, 2005. ACM Press.

[23] J.C. Simo. A framework for finite strain elastoplasticity based on maximum plastic dissipation and the multiplicative decomposition: Part I. Continuum formulation. *Computer Methods in Applied Mechanics and Engineering*, 66:199–219, 2 1988.

[24] J.C. Simo. A framework for finite strain elastoplasticity based on maximum plastic dissipation and the multiplicative decomposition. Part II: Computational aspects. *Computer Methods in Applied Mechanics and Engineering*, 68(1):1–31, 1988.

[25] J.C. Simo and M. Ortiz. A unified approach to finite deformation elastoplastic analysis based on the use hiperelastic constitutive equation. *Computer Methods in Applied Mechanics and Engineering*, 49:221–245, 1985.

[26] C. Sun, B. Kim, G. Yi, and H. Park. A Model of Problem Solving Environment for Integrated Bioinformatics Solution on Grid by Using Condor. In *Grid and Cooperative Computing*, pages 935–938, 2004.

[27] L. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner. A break in the clouds: towards a cloud definition. *ACM SIGCOMM Computer Communication Review*, 39(1):50–55, December 2009.

[28] C. Vecchiola, S. Pandey, and R. Buyya. High-performance cloud computing: A view of scientific applications. In *Proceedings of the 2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, ISPAN 09, pages 4–16, Washington, DC, USA, 2009. IEEE Computer Society.

[29] J. Wozniak, A. Striegel, D. Salyers, and J. Izaguirre. GIPSE: Streamlining the management of simulation on the Grid. In *38th Annual Simulation Symposium*, pages 130–137, 2005.

[30] B. Xiaoyong. High Performance Computing for Finite Element in Cloud. In *International Conference on Future Computer Sciences and Application (ICFCSA)*, pages 51–53. IEEE Computer Socienty, June 2011.

[31] C. Youn and T. Kaiser. Management of a parameter sweep for scientific applications on cluster environments. *Concurrency & Computation: Practice & Experience*, 22:2381–2400, 2010.