SCIENTIFIC WORKFLOW MANAGEMENT SYSTEM APPLIED TO UNCERTAINTY QUANTIFICATION IN LARGE EDDY SIMULATION

Gabriel Guerra, *gguerra@ufrj.br*
Fernando Rochinha, *faro@mecanica.ufrj.br*
Programa de Engenharia Mecânica, COPPE , Universidade Federal do Rio de Janeiro, Caixa Postal 68503, CEP 21945 970, Rio de Janeiro - RJ, Brasil

Renato Elias, *renato@nacad.ufrj.br*
Alvaro Coutinho, *alvaro@nacad.ufrj.br*
Núcleo de Computação de Alto desempenho, NACAD, Programa de Engenharia Civil, COPPE, Universidade Federal de Rio de Janeiro, Caixa Postal 68506, CEP 21945 970, Rio de Janeiro - RJ, Brasil

Vanessa Braganholo, *braganholo@dcc.ufrj.br*
Departamento de Ciência da Computação, Instituto de Matemática, Universidade Federal do Rio de Janeiro, Caixa Postal 68530, CEP 21941 590, Rio de Janeiro - RJ, Brasil

Daniel de Oliveira, *danielc@cos.ufrj.br*
Eduardo Ogasawara, *ogasawara@cos.ufrj.br*
Fernando Seabra, *fernando_seabra@cos.ufrj.br*
Marta Mattoso, *marta@cos.ufrj.br*
Programa de Engenharia de Sistemas e Computação, COPPE, Universidade Federal de Rio de Janeiro, Caixa Postal 68511, CEP 21945 970, Rio de Janeiro - RJ, Brasil

Abstract. *Currently Large Eddy Simulation (LES) requires intensive computation and a lot of data management. Today this management is often carried out in a case by case basis and requires great effort to track it. This is due to the large amount of data involved, thus making this process prone to errors. Moreover, there is a need to explore parameter variability (e.g., eddy viscosities) for the same set of data. In this context, techniques and methodologies of scientific workflows can improve the management of simulations. This variability can be put in the general context of Uncertainty Quantification (UQ), which provides a rational perspective for analysts and decision makers. The objective of this work is to provide a systematic approach in: (i) modeling of LES numerical experiments, (ii) managing the UQ analysis (iii) running each variation in parallel under the control of the Scientific Workflow Management System (SWfMS). When using scientific workflows, one can collect data in a transparent manner, allowing the post-assessment of results and providing the information for the re-executing the experiment, thereby ensuring reproducibility, an essential characteristic in a scientific experiment.*

Keywords: *Uncertainty Quantification, Large Eddy Simulation, Workflow, Parallel Computation*

1 INTRODUCTION

Consistently replacing prototypes or physical experiments by computations has been pursued within the Engineering and Applied Science community in the last decades, to a large extent due to the impressive computer power available nowadays. But the use of computer simulations as an effective tool still faces conceptual and technical challenges. Chief among these is the reliability of the predictions regarding the real physical response of the systems. Computational analysis in general is intended to provide reliable predictions of particular events, which are used as the basis for crucial decisions. In that sense, Uncertainty Quantification (UQ), which is a critical element in experimental exploration, has been considered recently an important conceptual basis for improving the reliability and wide acceptance of computer simulation predictive capacity.

Among the possibilities, non deterministic approaches, grounded on a solid basis provided by the Probability theory, offer a unique opportunity of, systematically, accommodating lack of knowledge and variability regarding physical parameters and non modeled dynamics. It is still convenient due to its natural integration with risk analysis that often leads decision makers. Dealing with stochastic numerical models arising from a non deterministic view can be carried out through two different approaches: sampling or non sampling methods. Among the sampling options, the most adopted is the Monte Carlo method, which is non-intrusive, implying that well known and tested computer codes can be naturally incorporated as a kernel of the stochastic analysis.

These analyses require intensive computation and a lot of data management. Today this management is often carried out in a case by case basis and requires a great effort to track it. This is due to the large amount of data involved, thus making this process prone to errors. Moreover, there is a need to explore parameter variability (e.g., eddy viscosities) for the same set of data. In this context, techniques and methodologies of scientific workflows can improve the management of simulations.

Large scale scientific numerical experiments, as those found in UQ, are usually composed of a set of data flows that are processed by a chain of activities that may be represented as scientific workflows (Oinn et al 2004, Deelman et al, 2009). A scientific workflow can be seen as one of the scientific experiment trials to evaluate one controlled action, and, in that sense, it naturally accommodates UQ framework, particularly those methods associated to sampling, as Monte Carlo methods. The set of trials represented by each distinct workflow execution defines one scientific experiment (Ogasawara et al, 2009). Workflow Management Systems (SWfMS) (Deelman et al, 2009) are able to orchestrate the execution of scientific workflows. They play a key role on the scientific experiment management. However, controlling variations in several workflow parallel executions is currently an open problem to SWfMS.

Along the workflow design process, we can often identify activities that can be executed in parallel, such as the many realizations of Monte Carlo method. These realizations should be controlled, both on the specification of the workflow and on its corresponding execution. When the workflow is defined through scripts or low level programming interfaces, the researcher can be lost among the several realizations that are being tested for the same experiment. According to Zhao, Raicu and Foster (2008), there is “a need of common generic infrastructures and platforms in the science domain for workflow administration, scheduling, execution, monitoring, provenance tracking, etc.” Provenance (Freire et al, 2008) is a key feature of SWfMS, since the analysis of scientific results, using the whole data flow, can only be evaluated if provenance data has been collected in a structured manner. In other words, the scientific experiment life cycle must be coordinated as a whole.

The Hydra middleware (Ogasawara et al, 2009) has been developed to address the challenges of managing and controlling parallel executions of workflow variations as part of one scientific experiment. Hydra bridges the gap between SWfMS and high performance computing by supporting experiment management, parallelization of activities and provenance capturing. This paper aims at using and evaluating Hydra in UQ. We have designed a Computational Fluid Dynamics (CFD) workflow to perform an UQ analysis. This proof of concept deals with an evaluation of the well known Smagorinsky Turbulent dissipation model, often employed in Large Eddy Simulations.

The CFD workflow is executed by the VisTrails (Freire et al, 2008) SWfMS while Hydra controls parallelization of activities following the Many Tasks Computing (MTC) paradigm (Raicu and Foster, 2008). Hydra contributes by controlling and gathering provenance data during remote parallel workflow execution. Through provenance data, an MTC parallelization strategy can be reused. Hydra also aims at reducing the complexity involved in designing and managing activity/workflow parallel executions within scientific experiments. Hydra helps researchers to face issues such as: modeling workflow activities using MTC paradigm, submitting activities from the SWfMS to the distributed environment, identifying failures, detecting performance bottlenecks, monitoring processes status to let the SWfMS aware of the remote execution and gathering remote provenance data into the SWfMS. As we will show in the remaining of this paper, our performance evaluation has shown that using Hydra to control parallel executions of workflows activities presents speedup in addition to a controlled and reproducible experiment.

This paper is organized as follows. Section 2 presents fundamental knowledge of scientific workflows. Section 3 presents the specific CFD scientific workflow and how it was modeled using the Hydra approach. Section 4 presents a sensitivity analysis of a turbulent model by means of an Uncertainty Quantification approach. Section 5 presents the Hydra evaluation and experimental results. Finally, Section 6 presents a summary of our main conclusions.

2 SCIENTIFIC WORKFLOW BACKGROUND

In silico experiments are found in many different scientific domains such astronomy, physics and oil exploration. Those *in silico* experiments represent studies that encompass multiple combinations of program, services, huge amounts of data resources and the intense use of HPC environments. Thus, managing these experiments is a complex task to be accomplished manually.

A typical *in silico* experiment is conducted by a researcher, responsible for designing, executing, controlling and analyzing it. Currently, researchers execute many activities during their experimentation process, and most of them are related to either the execution of a flow of programs, services and data or tuning scientific tools, such as ParaView (2009).

Scientific workflows are being used as an abstraction to model flows of programs, services and data. A scientific workflow represents the structured composition of activities and data as a sequence of operations aiming a desired result. In this paper, we consider a scientific experiment as an *in silico* experiment that is directly associated to a set of duties that involves the construction of several variations of scientific workflows. However, to achieve reliable results, *i.e.*, results that can be reproduced in the future, *in silico* experiments must keep track of provenance data (Freire et al, 2008). There are two types of provenance: prospective and retrospective (Freire et al, 2008 and Davidson and Freire, 2008). The prospective provenance is related to the definition of the workflow, the used parameters, datasets and the chaining sequence. The retrospective provenance is related to the execution

of the workflow, like the results achieved following one specific execution path and error messages.

However, to achieve this objective, *in silico* experiments must use tools that automatically collect and document all variations (*i.e.* parameters, input data) related to the associated scientific workflows. SWfMS are software engines that manage the scientific workflow. They are defined as software packages that can specify, execute, and monitor scientific workflows. Thus, the main focus of SWfMS is on helping researchers to manage their scientific workflows. However, SWfMS do not support the concept of an experiment as a whole. On the contrary, they just support isolated workflow executions.

There are several SWfMS available (Callahan et al, 2006). VisTrails is the one that brings powerful provenance capabilities to the researcher. Unlike other SWfMS, VisTrails provides techniques and algorithms to support streamline and data exploration process through visualization. VisTrails focus on capturing and maintaining detailed provenance data about the workflow. It tracks workflow executions and results to facilitate visual result comparison and workflow redefinitions. VisTrails was our choice to model the uncertainty quantification scientific workflow due to its powerful and unique provenance support.

3 CFD SCIENTIFIC WORKFLOW USING HYDRA

As the complexity of *in silico* experiments increased, the demands for high performance in executing some activities in scientific workflows have also increased considerably. In the CFD realm, researchers often face problems like analyzing the role of physical parameters varying within intervals through a number of datasets for several times, studying the influence on flow features of several parameters, such as eddy viscosities and Reynolds number. Recently, there is a growing interest on complex CFD studies involving verification, validation and uncertainty quantification which usually require a massive amount of simulations and generate huge data. This scenario increases the complexity of managing such analyses. Usual scheduling, as provided by some schedulers such as Torque/PBS, does not give primitives to cope with such complexity. Scientific workflows are promising solutions to support such exploratory demands automatically, besides providing less error prone studies. Particularly, since the amount of generated data increases drastically, provenance plays a key role in this scenario, and it is supported by scientific workflows. For example, researchers may query the retrospective provenance data to find where the results and generated files are stored for each parameter. They may also request which combination of parameters generated a specific result. In addition, they may need both prospective and retrospective provenance data to identify what workflow generated a particular set of results.

CFD modeled as scientific workflows usually need HPC to execute long running activities or the entire scientific workflow. The Hydra middleware was designed and developed to achieve this objective. It aims at reducing the complexity involved in designing and managing parallel activity executions within scientific workflows. Hydra isolates the end user (researcher) from the HPC environment complexity, offering a transparent and explicit middleware to execute scientific workflows activities that demand HPC by using the MTC paradigm (Raicu and Foster, 2008).

Hydra aims at distributing, controlling and monitoring the execution of scientific workflow activities in a HPC environment. These activities are usually programs, but even completely independent scientific workflows may be executed in HPC environments. Hydra provides a suite of components that transparently manage the distributed execution of workflow activities in the HPC environment. Hydra components are modeled as workflow activities that may be inserted on demand into the scientific workflow definition, preserving

the original features of the workflow. Hydra is a two-layer middleware that is executed in the SWfMS (client layer) and in the HPC environment (server layer).

This section briefly presents the Hydra major features and a scientific workflow EdgeCFD case study. To summarize, Hydra is decoupled from the enactment workflow engine, and has some unique features: (i) Hydra components act in a transparent way; (ii) it allows researchers to monitor legacy applications on distributed resources, providing a high level of abstraction of the distributed environment; and (iii) it provides a loosely coupled interaction between the local SWfMS and the distributed HPC environment. These features guided us to an architecture in which Hydra is simple to deploy and able to be architecturally linked to any local SWfMS. The Hydra architecture is described in detail in (Ogasawara et al, 2009).

3.1 Types of Parallelism Provided by Hydra

When the researcher needs to specify parallel activities in scientific workflows to be executed by SWfMS, it becomes a barrier for the researcher to control and register provenance data, since managing a distributed scientific workflow execution in remote HPC environments requires a great effort and discipline to manage the huge data generated. Hydra provides two types of parallelism, data (Meyer et al, 2005 and Meyer et al., 2007) and parameter sweep parallelism (Samples et al, 2005 and Walker and Guiang, 2007). To detail these two types, we need to formally define a scientific workflow. Our formalism was adapted from (Meyer et al, 2005). A workflow *Wf* is represented by a quadruple (*P,Pt,I,O*) where: *P* is a set $\{p_1, p_2, ..., p_p\}$ of programs that composes the workflow *Wf*; *I* is a set $\{i_1, i_2, ..., i_m\}$ of input data; *Pt* is a set $\{pt_1, pt_2, ..., pt_r\}$ of parameters of the *Wf* and *O* is a set of sets $\{O_1, O_2, ..., O_p\}$, where O_j is a set of output data for program p_j.

Data parallelism can be characterized by simultaneous execution of one activity of the workflow *Wf* with different independent data. Each activity execution processes a subset of the entire input data. This parallelism can be reached by allocating a given p_p that represents an activity at each node of the HPC environment and the input data elements *I* is processed independently. The set of output data O_k generated in each *k* node must be merged in order to produce the final result. On the other hand, parameter sweep parallelism is characterized by the simultaneous execution of one activity of the workflow *Wf* (or the entire *Wf*), each execution processing different sets of parameters *Pt*. For example, suppose that a given *Wf* has parameters pt_1, pt_2 and pt_3. One instance of one activity of the *Wf* may consume values *x, y, z* for parameters pt_1, pt_2 and pt_3, and another instance may consume values *x', y'* and *z'*. This can be achieved by allocating a different configuration of *Pt* at each node of the HPC environment. The set of output data O_k generated by the execution *k* needs to be aggregated, since they refer to different executions of the same experiment. These two types of parallelism are also known as "bag-of-tasks". However, in the context of thousands of tasks and millions of processing units it is no longer an embarrassingly parallel application. It becomes an MTC challenge. Hydra takes care of the parallel specification and submission of jobs to the HPC environment. Furthermore, Hydra allows the researcher to analyze heterogeneous distributed provenance data in addition to automatically detecting these two types of parallelism.

3.2 The EdgeCFD Scientific Workflow

This section describes one possible CFD scientific workflow using EdgeCFD, a stabilized finite element incompressible flow solver (Elias and Coutinho, 2007). The EdgeCFD workflow is composed by four steps. The first one is the *modeling* stage, where computational models, describing a real geometry, are created and discretized in smaller

pieces (elements/cells). These pieces are used in the next steps of the workflow, where the equations describing the physical phenomena will be solved by a suitable method. The modeling stage is usually performed manually with the help of a modeling computational tool such as ANSYS. The second step is the *pre-processing* stage, where physical and solution parameters, such as boundary and initial conditions, solution method tolerances, simulation time and material properties are assigned to the computational model. The next step, the *solution* step is responsible for solving the equations that models the physical problem of interest. This is generally the most demanding step in a CAE (Computer Aided Engineering) simulation workflow. Finally, the last step of the workflow is the *post-processing* step. In this step, results produced in the solution stage are interpreted with the help of visualization tools which translate raw numbers (generally in binary format) in plots, pictures and movies. For the present test case, only the second, third and forth steps, the most time consuming and suitable for automation, were considered.

The lid-driven cavity flow problem (Bouffanais and Deuville, 2007) was chosen to evaluate the efficiency of the Hydra system in providing provenance information and parallelizing a CFD workflow scheduling. It is important to clarify that EdgeCFD solver is a classical example of an application that is aligned with the MTC paradigm and that can be run in serial or in parallel mode, using several processors that could be employed to speed up the solution process for each parameter variation.

3.3 EdgeCFD Scientific Workflow Using Hydra Components

To run our EdgeCFD workflow case study, we have chosen VisTrails SWfMS (Callahan et al, 2006) and Hydra. The high-level workflow described in the previous section as modeled using VisTrails and Hydra modules. This implementation on VisTrails is called a concrete workflow. Figure 1 shows the concrete workflow modules on VisTrails.

The experiment consisted of the pre-processing, solver and visualization activities. Recall that in this concrete representation, we do not have the modeling stage, since it is not an automatic activity. The pre-processing stage is represented, in this concrete workflow, by the Local Pre-Processor EdgeCFD activity. The next stage, the solver, is represented by the four Hydra components (Uploader, Dispatcher, Gatherer and Downloader), as we will explain later. Finally, the visualization stage is represented by the activities inside the blue rectangle in Figure 1. In this stage, we used the VisTrails spreadsheet. There are lots of activities in the concrete workflow of Figure 1 because this is a complex activity that requires a lot of data conversion.

The pre-processing and the visualization are run locally on the machine where the SWfMS is installed. The solver activity is the one that needs more computational resources to run. Thus, it was the activity chosen to run on the HPC environment. This is an exploratory activity in the EdgeCFD workflow that varies specific parameters, like the eddy viscosity in the Smagorinsky's turbulent dissipation model. Thus, the activity is executed repeatedly for each parameter, so the parallelization is important. The advantage of using Hydra in this context is that it is possible to register the prospective provenance for the parameter sweep of the exploratory experiment. Additionally, this would be a laborious error prone activity if done manually by researchers. In our Hydra infrastructure, this exploratory activity can be executed in a HPC environment considering a parameter sweep that generates as many configurations as required for the Solver to run.

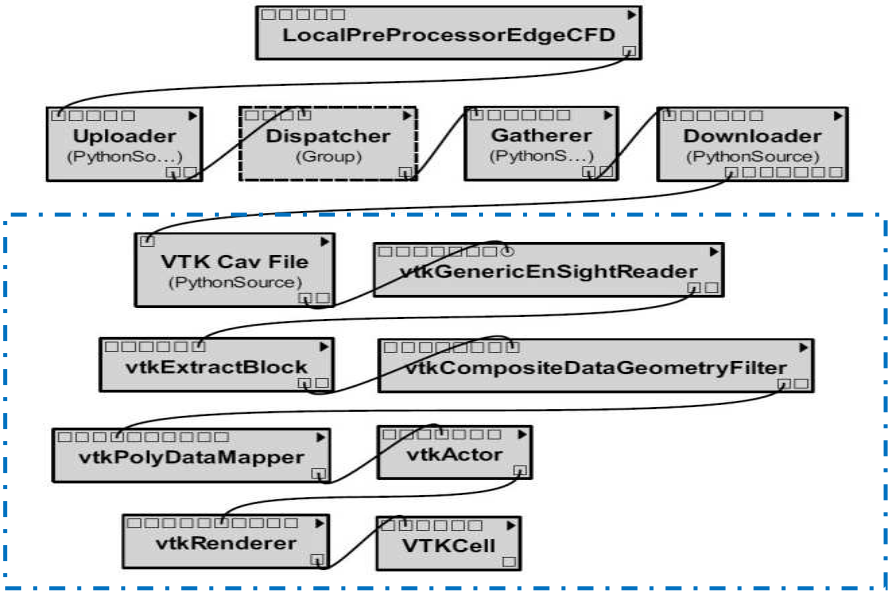

Figure 1 EdgeCFD workflow modeled using Hydra Client Components

We have setup the Hydra client components on top of a local SWfMS VisTrails framework, in the solver activity. There are four Hydra components (*Uploader, Dispatcher, Gatherer,* and *Downloader*) that control the distributed execution of the activity, in this case, the solver. The *Uploader* is the component responsible for sending the input data to the HPC environment, the *Downloader* is responsible for downloading the processed results to the local machine (the one where the SWfMS is running), the *Gatherer* is responsible for gathering the provenance data produced at the HPC environment and the *Dispatcher* is responsible for starting and controlling the HPC execution of an activity (in this case the Solver) or the entire workflow. The distributed components were setup directly into a cluster using the *Hydra Setup* component.

The implementation of the local components encompasses all the client components described in detail in (Ogasawara et al, 2009). The *Uploader* and the *Downloader* were implemented to connect to the cluster via SSH and to send/receive the data files directly to a repository on the cluster storage. The *Dispatcher* connects via SSH to the Hydra server layer to invoke the distributed activity at the HPC environment, which, in our example, is the Solver. The distributed components of Hydra were implemented on the cluster environment, and we have chosen Torque as our scheduler.

4 LES SCIENTIFIC WORKFLOW USING HYDRA

The LES Scientific workflow (EdgeCFD-LES for short) is an adaptation of the EdgeCFD workflow to solve a UQ problem for Large Eddy Simulation. This section explains in details in the UQ problem modeled by the LES scientific workflow.

4.1 Uncertainties in Large Eddy simulation

LES has been employed as a computational method to predict turbulent flows in various applications that try to combine accuracy with computer costs. Roughly speaking, LES results from applying a filter on the Navier-Stokes equations given rise to the need of modeling the resulting sub-grid scales influence over the resolved ones. That is usually carried out based on trying to mimic the real dissipation mechanisms of the flow. Therefore

the sub-grid influence is replaced by an extra eddy viscosity. The most popular model of eddy viscosity in LES is the so called Smagorinsky model. It considers that the deviatoric part of subgrid scale tensor is proportional to the velocity strain tensor $\overline{S}$ with a nonlinear eddy viscosity parameter defined as,

$$\nu_T = (C_S \overline{\Delta})^2 \overline{S} \text{ with } \left|\overline{S}\right| = \sqrt{2S_{ij}S_{ij}}$$

where Δ is the filter length and C_S is a parameter to be tuned. Therefore, the correct selection of the model parameter C_S is central for the quality of the results. Very different values are assumed, as 0.1 (Deardoff, 1970), 0.15 (Pope, 2000) and 0.23 (Lilly, 1966), some of them are either empirical results or theoretical estimations. Recently, it was demonstrated that the Smagorinsky coefficient is not constant. Instead, it depends on the local Reynolds number and on the resolution of the turbulent integral-length scales (Meyers and Sagaut, 2006, Voke, 1996). Clearly, in the absence of an experimental or a Direct Numerical Simulation (DNS) reference solution, the selection of the model coefficient C_S and the corresponding simulation quality are uncertain. In these cases, confidence in simulation results is typically founded on a mix of common sense and extensive sensitivity analysis of parameters. In this paper we will explore Hydra to perform a sensitivity analysis for selecting Smagorinsky's model parameter.

4.2 Scientific workflows applied to a Smagorinsky model sensitivity analysis

In this section, we explore the workflow computational tools introduced in the preceding sections. We present a sensitivity analysis of the Smagorinsky coefficient (Lucor et al, 2007). With this goal in mind, we have selected a classical benchmark problem already addressed by a number of researchers (for a review see Bouffanais and Deville, 2007)). The example consists in a lid driven cavity flow within the turbulent regime with the Reynolds number fixed as Re = 1000, as seen in Figure 2. The computational domain is discretized with the aid of a mesh consisting on 32×32×32 tetrahedral elements.

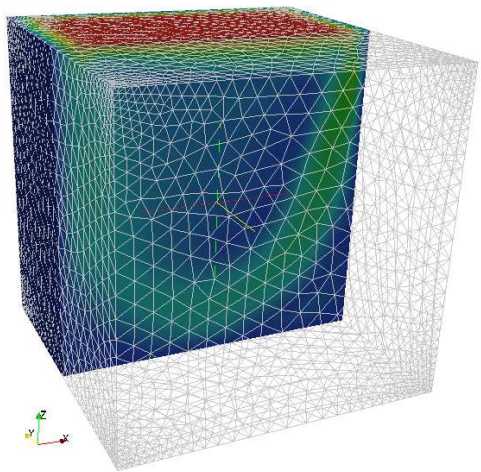

Figure 2 Lid driven cavity flow with 32×32×32 mesh

A sensitivity analysis starts from modeling the parameters. Here, as we adopt a non-deterministic approach, C_S is modeled as a random variable defined as $C_S = 0.1(1+\alpha\zeta)$ where α representing uncertainty level, here kept as 1, and ζ is a uniform random variable

assuming values in the interval [0,1]. It is important to notice that this definition encompasses the usual interval description for the variability of parameters, in which the same probability is attributed to any value within the prescribed interval. The advantage of modeling a parameter as a random variable relies in two main points. The first is related to the capacity of the probability model to accommodate better experimental evidences like the frequency of appearance of specific values. The second point consists on providing a rational basis to put the sensitivity analysis in a quantitative perspective.

We have generated, through sampling from ζ, 640 experiments with C_S taking values such that $0.1 \leq C_S \leq 0.2$. By using Hydra, we have explored the intrinsically parallel potential of the Monte Carlo method and ran 64 serial jobs simultaneously. Table 1 summarizes the simulation data.

Table 1 - Numerical and physical parameters of simulation

Reynolds Number	1000
Mesh	32^3
Time Step	0. 1
Sample Spam	120
Sampling rate	10
Size of sample	27.8Mb
Number of samples	640
Total data held in storage	17.2Gb

The obtained results illustrate variations often observed in LES results associated with changes in the Smagorinsky coefficient or in the spatial resolution.

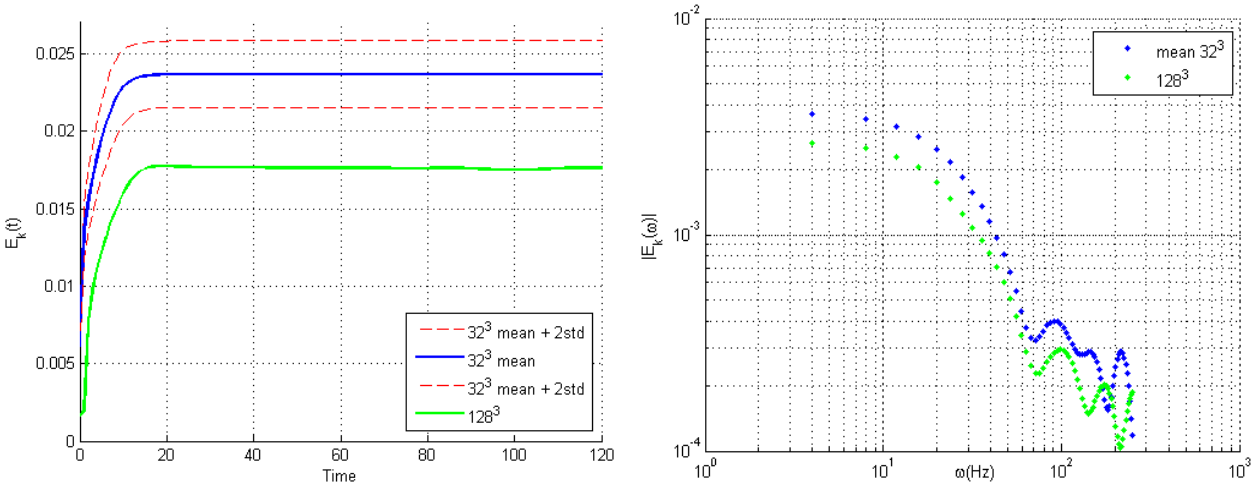

Figure 3 (a) Kinetic energy evolution (b) Energy spectra

The curves of **Figure 3** (a) present the average of kinetic energy evolution characterizing the general flow response and the dissipation regime as well. Moreover, a confidence interval computed with the aid of two standard deviations was also plotted. As a reference for the Smagorinky model performance, the kinetic energy calculated with $C_S = 0.1$ and finer mesh of 128×128×128 elements was included in the graph. In **Figure 3** (b) we show the energy spectra for the mean and for the fine mesh, where we can clearly see discrepancies. From **Figure 3** we can conclude that, for the flow conditions analyzed here, the static Smagorinky

model is not able to capture the actual flow characteristics, as the probability of achieving this behavior, assumed to be the one provided by the finer mesh computations, is almost zero. Within the context of the stochastic model built here allied with the workflow capabilities, a deeper and detailed analysis will be developed in the near future.

5 EDGECFD-LES WORKFLOW PARALLEL PERFORMANCE ANALYSIS

We investigate here the parallel performance of the EdgeCFD-LES workflow. EdgeCFD-LES is invoked by VisTrails from a desktop and distributes the Solver activity using the MTC paradigm into a cluster using Hydra to execute the parallel parameter sweep, that is, the sensitivity analysis. The execution of the EdgeCFD-LES workflow using Hydra was performed in the SGI Altix ICE 8200 with 64 nodes with Intel Xeon 8-core processors/node installed at the High Performance Computing Center, COPPE/ Federal University of Rio de Janeiro. To access performance, we run a simpler experiment than the one described in Section 4, using just 100 time steps in each simulation and varying the C_S with 100 random variables as described earlier. This leads to 100 small tasks. These tasks have little physical significance since the calculated solution was not yet stable, but the goal was to evaluate the performance of Hydra with respect to its overhead for an increasing number of cores. In this simple experiment we increase the number of cores from 8 to 128, as shown in Figure 4(a). The execution time of the whole EdgeCFD-LES workflow was significantly reduced as the number of nodes involved for the distributed activity (Solver) increased, as presented in Figure 4(a). We may observe that execution time drops from approximately three days to just one hour. The general overhead of transferring data from the local SWfMS to the HPC environment and vice-versa is acceptable when compared to the degree of parallelism obtained for the parameter sweep (sensitivity analysis) of the solver activities. We may note that for 128 cores, the execution time for Hydra alone is 2642 min and when the data transfer is taken into account the execution time increases to 3534 min.

To verify if the data transfer overhead is really acceptable, Table 2 presents the distributed execution in just 8 cores according to Hydra execution catalog. The Monte Carlo analysis was implemented within Hydra through the sweep functionality by varying the eddy viscosity through a set of previously generated random sampling set. During the distributed execution, a total of 100 parameters were explored. These parameters represented the number of distributed activities. The upload transfer time (1.3 min) and download transfer time (13.2 min) represents the overhead for data transfer between the SWfMS (client) and the HPC environment (server). The wait time for scheduler represents the time spent in the Torque/PBS queue (0.3 min). The execution time in HPC represents the total time needed to execute all distributed activities (556.1 min). This information combined with the execution catalog is important to validate and reproduce the simulation. Important provenance data could be lost if a systematic controlled approach such as Hydra was not used.

Table 2 - Hydra distributed execution in 8 cores for the simple experiment

Number of activities (parameters explored)	100
Upload Transfer Time	1.3 min
Wait time for scheduler	0.3 min
Execution Time in HPC (1node with 8 cores)	541.3 min
Download Transfer Time	13.2 min
Total Execution Time	**556.1 min**
Speedup	7.6
Number of errors	0

The retrospective provenance data obtained from the execution catalog allows building the execution time histogram of all activities, as presented in Figure 4(b). Based on the information obtained from the execution catalog, it is possible to calculate the average (42.3 min) and standard deviation (0.1 min) of each activity. It is also possible to estimate the equivalent sequential time for all activities, which was 4232.3 min. Additionally researchers can use the catalog to check which parameter generated a desired result.

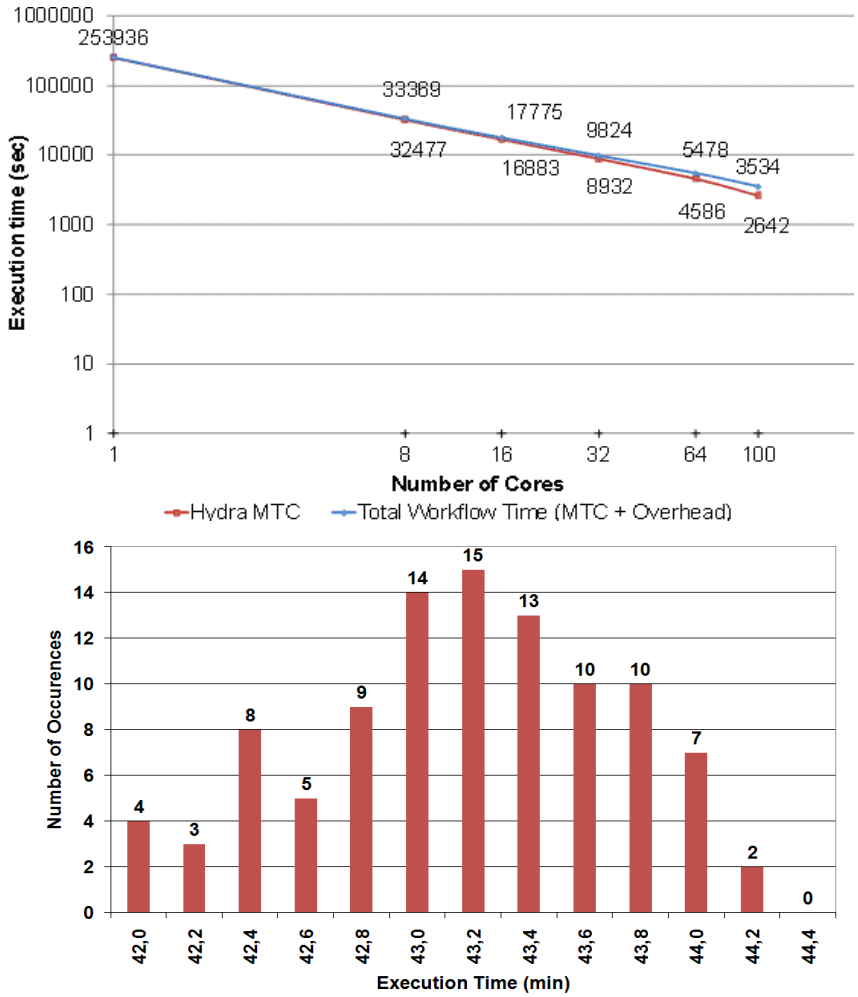

Figure 4 (a) Workflow execution time (b) Activity execution time distribution

6 CONCLUSIONS

Computer simulations and computational analyses aim to provide reliable predictions of particular events. Typically these simulations involve a chain of programs to be executed by exploring parameter variability and huge amounts of data flow. When the control of these computational executions is done manually, it is susceptible to errors, since one can easily be confused while trying to find the exact output dataset for one specific parameter exploration. Often, a simulation is re-executed to be sure of a conclusion. This becomes even more complex when using heterogeneous and distributed high performance computing environments. In this context, techniques and methodologies of scientific workflows can improve the management of simulations.

In this work we have modeled Large Eddy Simulation as a scientific workflow to perform an UQ analysis. We have used the VisTrails SWfMS to manage the workflow execution locally and we have used the Hydra middleware to manage the parallel executions of the workflow programs in a remote high performance computing environment. Hydra was able to provide parameter sweep using a systematic approach, thus decreasing the chances of error during the decomposition and execution of the parallel workflow. This sweep corresponds to a sensitivity analysis of eddy viscosities in LES. Provenance data related to the parallel execution was gathered and made available to the user for analysis. It was also shown that sensitivity of the LES solution to uncertain C_s can be successfully investigated in HPC environment using SWfMS tools.

Further, results show that we have successfully run the sensitivity analysis, collecting provenance data with an acceptable overhead. The speedup obtained with 128 cores shows a significant drop on the execution time from approximately three days to just one hour. Without the SWfMS and Hydra, the execution time would drop to 44 minutes instead of 60 min. However this difference shows a very good tradeoff, since when the parallel execution is controlled manually, the scientist has to control file locations, parameters used in each run, etc. On the other hand when using Hydra and VisTrails, one can collect data in a transparent manner, allowing the post-assessment of results and providing the information for the re-executing the sensitivity analysis, thereby ensuring reproducibility, an essential characteristic in a scientific experiment.

ACKNOWLEDGMENTS

The authors would like to thank the financial support of PETROBRAS Research Center, FAPERJ and CNPq.

REFERENCES

T. Oinn, M. Addis, J. Ferris, D. Marvin, M. Greenwood, C. Goble, A. Wipat, P. Li, and T. Carver, “Delivering web service coordination capability to users,” *13th international World Wide Web conference on Alternate track papers & posters*, 2004, pp. 438-439.

E. Deelman, D. Gannon, M. Shields, and I. Taylor, “Workflows and e-Science: An overview of workflow system features and capabilities,” *Future Generation Computer Systems*, vol. 25, 2009, pp. 528-540.

E. Ogasawara, C. Paulino, L. Murta, C. Werner, and M. Mattoso, “Experiment Line: Software Reuse in Scientific Workflows,” *21th SSDBM*, New Orleans, LA: Springer-Verlag, 2009, pp. 264–272.

Y. Zhao, I. Raicu, and I. Foster, “Scientific Workflow Systems for 21st Century, New Bottle or New Wine?,” *2008 IEEE Congress on Services*, IEEE Computer Society, 2008, pp. 467-471.

J. Freire, D. Koop, E. Santos, and C.T. Silva, “Provenance for Computational Tasks: A Survey,” *Computing in Science and Engineering*, vol. 10, 2008, pp. 11-21.

I. Raicu, I. Foster, and Yong Zhao, “Many-task computing for grids and supercomputers,” W*orkshop on Many-Task Computing on Grids and Supercomputers*, 2008, pp. 1-11.

Paraview, *Paraview*, http://www.paraview.org, 2009.

S.B. Davidson and J. Freire, “Provenance and scientific workflows: challenges and opportunities,” *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, Vancouver, Canada: ACM, 2008, pp. 1345-1350.

D.A. Bader, *Petascale computing: algorithms and applications*, Chapman & Hall/CRC, 2008.

S.P. Callahan, J. Freire, E. Santos, C.E. Scheidegger, C.T. Silva, and H.T. Vo, “VisTrails: visualization meets data management,” *Proceedings of the 2006 ACM SIGMOD*, Chicago, IL, USA: ACM, 2006, pp. 745-747.

E. Ogasawara, D. Oliveira, F. Chirigati, C. Barbosa, R.N. Elias, V. Braganholo, and M. Mattoso, “Exploring Many Task Computing in Scientific Workflows,” *Submitted to the 2nd Workshop on Many-Task Computing on Grids and Supercomputers*, 2009.

L.A. Meyer, S.C. Rössle, P.M. Bisch, and M. Mattoso, “Parallelism in Bioinformatics Workflows,” *High Performance Computing for Computational Science - VECPAR 2004*, 2005, pp. 583-597.

L. Meyer, D. Scheftner, J. Vöckler, M. Mattoso, M. Wilde, and I. Foster, “An Opportunistic Algorithm for Scheduling Workflows on Grids,” *High Performance Computing for Computational Science - VECPAR 2006*, 2007, pp. 1-12.

M.E. Samples, J.M. Daida, M. Byom, and M. Pizzimenti, “Parameter sweeps for exploring GP parameters,” *2005 workshops on Genetic and evolutionary computation*, Washington, D.C.: ACM, 2005, pp. 212-219.

E. Walker and C. Guiang, “Challenges in executing large parameter sweep studies across widely distributed computing environments,” 5th IEEE workshop on Challenges of large applications in distributed environments, Monterey, California, USA: ACM, 2007, pp. 11-18.

R.N. Elias and A.L.G.A. Coutinho, “Stabilized edge-based finite element simulation of free-surface flows,” *International Journal for Numerical Methods in Fluids*, vol. 54, 2007, pp. 965-993.

R.N. Elias, V. Braganholo, J. Clarke, M. Mattoso, and A.L. Coutinho, “Using XML with large parallel datasets: is ther any hope?,” *Parallel Computational Fluid Dynamics (ParCFD)*, 2009

M. Fowler, *UML Distilled: A Brief Guide to the Standard Object Modeling Language (3rd Edition)*, Addison-Wesley Professional, 2003.

Lucor D., Meyers J. and Sagaut P. “Sensitivity analysis of large eddy simulations to subgrid-scale-model parametric uncertainty using polynomial chaos” J.Fluid.Mech., vol 85 pp 255-279. 2007

Bouffanais R., Deville M. “*Large-eddy simulation of flow in a lid-driven cubical cavity*” Physics of Fluids 19,055108, 2007.

Leriche E., Gavrilakis S.”*Direct numerical of the flow in a lid driven cubical cavity* “ Physics of Fluids vol.12 n.6, 2000.

Deardoff J.W. “A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers” J.Fluid Mech., vol.41, pp 453-480, 1970.

Pope S.B. 2000. Turbulent Flows. Cambridge University Press.

Lilly D.K. 1967. The representation of small-scale turbulence in numerical simulations experiments. Proceedings of IBM Scientific Computing Symposium on Environmental Sciences. IBM Data Processing Division, White Plains, New York.

Meyers J. and Sagaut P. “On the model coefficients for the standard and the variational multi-scale Smagorinsky model”. J.Fluid Mech., vol. 569, pp. 287-319, 2006

Voke P.R. “Subgrid-scale modeling at low mesh Reynolds number”. Theor. Comput. Fluid Dyn., vol. 8, pp. 131-143, 1996.