A Search Space Exploration Framework for e-Science Applications

Eric B. Gauch, Bruno E. C. Milanesi, Bruno Silva, Renato L. F. Cunha, Marco A. S. Netto

[1]IBM Research
Rua Tutóia, 1157 - São Paulo - SP - Brasil

{ebgauch,milanesi,sbruno,renatoc,mstelmar}@br.ibm.com

Abstract. *High Performance Computing (HPC) has always been a fundamental component to conduct scientific experiments. Model calibrations/simulations often require several executions of scientific applications by changing their input parameters. This process is a common practice in research even though it represents a tedious and error-prone task. In this paper we propose Copper framework which employs a black-box strategy and contains a set of plugins to accelerate user experiments for exploring search spaces in HPC parametric applications. Copper has been used to conduct scientific experiments in different areas including, agriculture, oil & gas, flood simulation, and bioinformatics.*

1. Introduction

High Performance Computing (HPC) is crucial for the execution of e-Science applications in various fields including agronomy, health, aerospace, circuit design, astronomy, and oil & gas exploration. Usually, researchers have to execute the same applications several times with different parameter values to calibrate models or evaluate what-if scenarios. Without a proper tool, this process is often cumbersome, error-prone, and a time consuming activity. Additionally, e-Science users are highly specialized in their respective research areas but they may not have the necessary expertise to execute their applications in HPC dedicated infrastructure (e.g., cluster).

To fill this gap, we propose Copper, a search space exploration tool for e-Science applications. The tool can execute any application that has a command line interface and makes transparent the access and monitoring of HPC infrastructures such as clusters and clouds. Copper presents a set of plugins to improve the search space exploration by providing hints to the user during the experiment set. In this paper, we present a brief overview of Copper and describe how to connect Copper to user applications. The proposed tool has been used to explore search spaces in e-Science applications from different areas such as agriculture, oil & gas, flood simulations, and bioinformatics [Silva et al. 2018, Silva et al. 2016].

2. Related Work

The execution of large number of independent jobs may become a complex task to users, mainly when distributed resources are used to execute these jobs. Resources may have different computational power and availability and may have different mechanisms to be accessed. Therefore, over the years, several tools have been created such as Condor

[Litzkow et al. 1988], XtremWeb [Fedak et al. 2001], BOINC [Anderson 2004], Nimrod [Abramson et al. 2000], and OurGrid [Andrade et al. 2003] to facilitate user access to these computational platforms. Gil *et al.* [Gil et al. 2011] introduced Wings, a system to assist scientists in designing experiments by tracking constraints and ruling out invalid designs. These tools mainly focus on managing users' jobs looking into the computational infrastructure. The main difference between our tool and the previous ones is that ours provides useful plugins to accelerate search space explorations [Silva et al. 2018]. For instance, Copper presents JobPruner which is a tool that looks into the user's workload and finds patterns from past experiments, which allow users to drastically reduce their search spaces when performing experiments of similar nature.

3. Copper

This section presents an overview of Copper's modules including the backend and frontend interfaces. The main idea is to demonstrate how general applications can be plugged into Copper and accessed via a high level user interface.

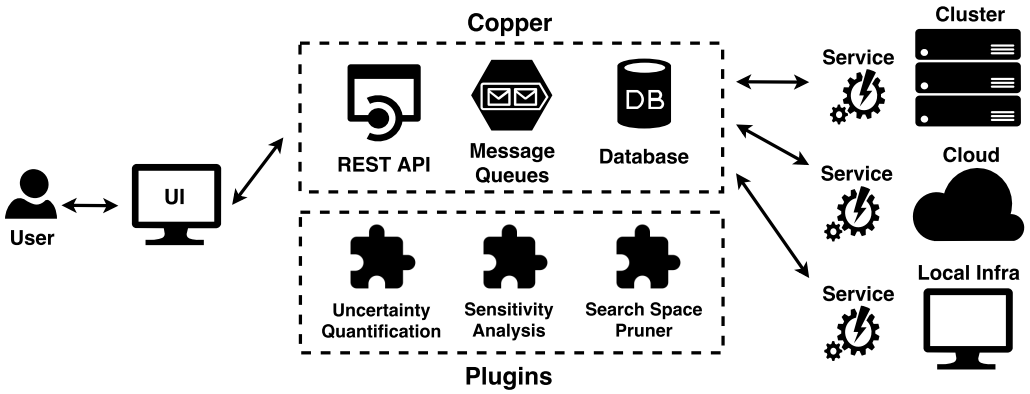

Figure 1. Overview of Copper's workflow

3.1. Backend

As illustrated in Figure 1, the user interacts with the user interface selecting one or more parameters with a single value or a range of values for each of them. These parameters and their respective values are then validated in Copper's backend where a corresponding application with its parameters definition must have already been registered. If a range of values is chosen for a parameter, Copper generates a set of jobs for each combination of parameters and values, and store them in a database. These jobs are sent through a message queue for local or remote processing. Copper can utilize techniques like uncertainty quantification, sensitivity analysis, and search space pruning available as plugins to achieve the user's goal [Silva et al. 2018]. The subsystem that consumes jobs from the message queue, executes user's software, and returns a result to Copper consists of two main components:

- A *service* that pulls messages from the message queue, parses it and calls the User Application, passing as argument the parameters and values from the message.
- The *User Application* must have a command line interface where we can specify the arguments for that application.

In order to execute user's jobs, both the User Application and the service itself have to be installed in the user's computational infrastructure (e.g., cluster, cloud or local machine). Having that satisfied, a configuration file is used to point where the User Application's executable is and some other options like the message queue's address. With all that set, the service can start pulling messages from the message queue. These messages sent from Copper to the message queue and acquired by the service are represented as a JSON (*JavaScript Object Notation*) file and contains all the parameters for the job execution. The service calls the executable specified in the configuration file passing the parameters from the message as command line arguments and will then, retrieve the result and send a message back to Copper. For the User Application, any software that takes a group of parameters as input and outputs a result can take advantage of Copper. The only requirement is that this application has to offer a command line interface. In some cases, this application takes a file with all the parameters as input, so a parser may be needed in order to create this file with the information that came from the message queue.

3.2. Frontend

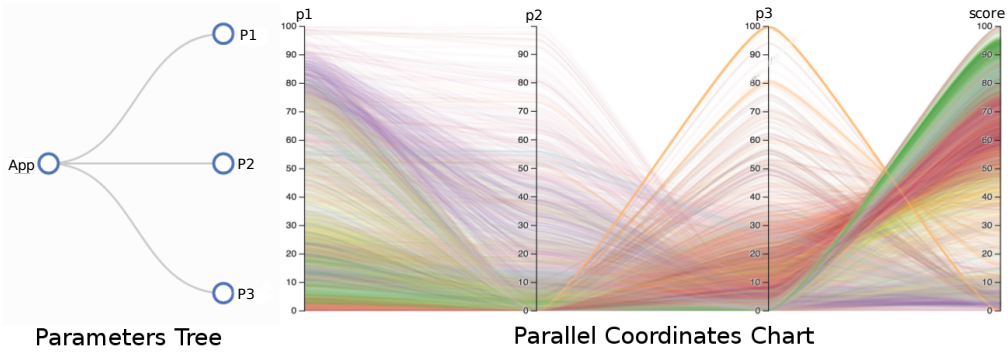

Figure 2. Copper's parameters tree and parallel coordinates chart

Copper's frontend was designed to assist the user with the application's parameters selection, tracking/analysing infrastructure usage, and display of experiments results. To achieve these goals, the interface was subdivided into three main parts: parameter tree (model), performance and infrastructure usage dashboard, and results visualizations. In Figure 2, we present the results visualization tab, in which each job corresponds to a line in the parallel coordinates graph. In this particular experiment, the application has three parameters (p1, p2, and p3) and a single value is provided as a result.

Copper accepts any application parameter to be used by the user, thus allowing a black box strategy. By using a JSON parameter description file, the user interface is automatically created with either the input parameters three (Figure 2) or a group of forms. Therefore, users can easily specify the experiment's input parameters and intuitively visualize the respective results. Users can monitor the execution of ongoing experiments via a infrastructure dashboard, which presents the computational resources (VMs, containers, or dedicated servers) allocated for each set of jobs. Real-time performance metrics can also be tracked in the infrastructure dashboard. These metrics include: job execution throughput and number of waiting/running/completed jobs. The results are then displayed on a parallel coordinates chart, also in real-time, making it easy to visualize all the variables (dimensions) and understand their impact on the model and associated scores.

4. Conclusion

Several areas in science and engineering require many executions of a software system with different values for each supported parameter. These executions are responsible for evaluating complex models using diverse scenarios. Executing all possible values for all supported parameters are usually not feasible, especially under cost and deadline constraints.

To fill this gap, this work introduced a tool to help users in executing their software systems by exploring parameter search spaces. Our main lesson, while developing the tool and evaluating User Applications was related to the particular characteristics of each User Application. Instead of focusing on each possible feature of any application, we concentrated our efforts on creating a general framework that addresses the main characteristics of parametric applications (execution with a variety of input parameters). Advanced users can also take benefit from our framework by creating user interfaces customized to their needs and employ Copper REST API to execute their parametric applications.

References

Abramson, D., Giddy, J., and Kotler, L. (2000). High Performance Parametric Modeling with Nimrod/G: Killer Application for the Global Grid. In *Proceedings of the 14th International Parallel & Distributed Processing Symposium*. IEEE.

Anderson, D. P. (2004). BOINC: A system for public-resource computing and storage. In *Proceedings of the 5th International Workshop on Grid Computing*. IEEE.

Andrade, N., Cirne, W., Brasileiro, F., and Roisenberg, P. (2003). OurGrid: An approach to easily assemble grids with equitable resource sharing. In *Proceeding of the Workshop on Job Scheduling Strategies for Parallel Processing*. Springer.

Fedak, G., Germain, C., Néri, V., and Cappello, F. (2001). Xtremweb: A generic global computing system. In *Proceedings of the 1st International Symposium on Cluster Computing and the Grid*. IEEE.

Gil, Y., Ratnakar, V., Kim, J., Gonzalez-Calero, P., Groth, P., Moody, J., and Deelman, E. (2011). Wings: Intelligent workflow-based design of computational experiments. *IEEE Intelligent Systems*, 26(1):62–72.

Litzkow, M. J., Livny, M., and Mutka, M. W. (1988). Condor - A hunter of idle workstations. In *Proceedings of the 8th International Conference on Distributed Computing Systems*. IEEE.

Silva, B., Netto, M. A., and Cunha, R. L. (2018). JobPruner: A machine learning assistant for exploring parameter spaces in HPC applications. *Future Generation Computer Systems*, 83:144 – 157.

Silva, B., Netto, M. A. S., and Cunha, R. L. F. (2016). SLA-aware Interactive Workflow Assistant for HPC Parameter Sweeping Experiments. In *Proceedings of the 11th Workshop on Workflows in Support of Large-Scale Science with The International Conference for High Performance Computing, Networking, Storage and Analysis*.