Universitat Politècnica de Catalunya
Facultat d'Informàtica de Barcelona

Bachelor Degree in Informatics Engineering
Computer Engineering Specialization
Degree Final Project
Thesis management course

Performance analysis and optimization of a combustion simulation

First assignment: Context and scope of the project

Author
Guillem Ramírez Miranda

Director
Marta Garcia Gasulla

Co-director
David Vicente Dorca

Tutor
Julian David Morillo Pozo

September 28, 2020

Contents

1 Context

1.1 Introduction

Computational science is a growing field that allows researchers to predict the behaviour of systems. For example, a physicist can write a program to predict the trajectory of a sphere instead of calculating it by hand.

What if we bring this further? We can try to optimize the aero-dynamism of a plane to save fuel, the behaviour of the human body cells to a recently designed drug or even given a genome figure out the risk of cancer.

Computing these are at a high cost. It is not conceivable to solve the problems on our laptops as the number of calculations and data required to process is overwhelming. To solve these tasks, we need supercomputers.

Supercomputers are machines with a huge compute power oriented to scientific and technique jobs. These machines are many small machines interconnected. Developers must write their code in a manner that the program is capable of run in parallel.

Writing efficient parallel programs is a difficult task and even more so if the developer's specialization is not computer science. In this work, we will study and improve Alya[1], a real use case application. In particular, we will focus on the module in charge of computing combustion processes.

This work is under an Educational Cooperation Agreement between the Facultat d'Informàtica de Barcelona and the Barcelona Supercomputing Center.

The study has been developed within a Curricular internship in the Barcelona Supercomputing Center precisely in the High-Level Support Team (HLST) inside the Operations department and developed under a Performance Optimization and Productivity[2] (POP) Center of Excellence Performance Analysis request.

1.2 Terms and concepts

In the following sections you will find basic concepts to better understand the project.

1.2.1 High-performance computing

High-performance computing (HPC) refers to the act of grouping compute power to do massive and complex computations and data processing. Nowadays HPC is associated to supercomputers.

1.2.2 Supercomputer

A supercomputer is a machine designed for HPC. The basic structure of a supercomputer consists of multiple computing nodes interconnected. Usually, a node is a shared memory system which can run programs by itself and may have add-ons like accelerators.

1.2.3 Parallel Programming model

Parallel programming models[3] add an abstraction to parallel computer architectures. It defines constructs that allow to operate with the parallel machine performing different actions. For example, sending and receiving messages between processes, reading and writing to the shared memory or spawning tasks in form of threads, all depending on the kind of programming model.

In this section, you will find an explanation of the parallel programming models used in this work.

1.2.4 Message Passing Interface (MPI)

The message passing interface[4] defines a standard interface for communications between processes. This allows applications to run in numerous nodes. The processes use the standard interface to send messages to each other using the underlying interconnection network of the machine. This interface allows, among other things, process synchronization, data-sharing via asynchronous and synchronous messages, gathering and scattering data, parallel input-output (I/0).

1.2.5 MareNostrum 4

MareNostrum 4[5] is a supercomputer managed by the Barcelona Supercomputing Center and managed by the Operations department. It is divided in two blocks, the general purpose block which represents the majority of the computing power and the emergent technologies block that aims to test new technologies. Figure 1 shows a picture of the supercomputer located in the "Torre Girona" chapel.

In this work we will be running the application on the general purpose block. The general purpose block consists of 3456 nodes each node containing 2 Intel Xeon 8160 with 24 cores each running at 2.1 GHz. The interconnection network consists of an Intel Omni-Path full-fat tree at 100 Gbps.

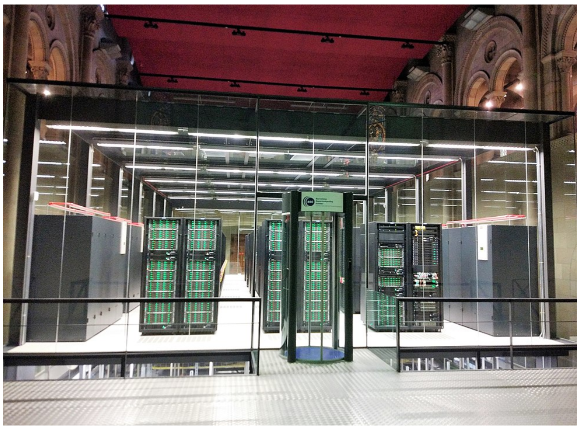

Figure 1: MareNostrum 4 machine. By Gemmaribasmaspoch - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=61308843

1.2.6 Alya

Alya[1] is the application we will be studying. It is a high-performance computational mechanics that aims to solve engineering coupled problems. In other words, Alya consists of multiple modules that can be joined to solve a specific problem. For this work, we will focalise the analysis into a new Alya module devoted to simulate combustion processes.

Alya supports a wide variety of programming models but the module we are given to analyse only supports MPI.

1.2.7 Performance Analysis

Running real use case scientific applications in supercomputer environments is crucial to identify flaws that deplete its performance when running on several machines.

Performance analysis refers to the process of gathering data from application executions, study the data and finally identify bottlenecks of the program.

In the state-of-the-art performance analysis, there is a wide variety of methodologies. However, this work is under the POP Center of Excellence resultantly we chose its methodology [6].

1.2.8 BSC tools

As said, for a performance analysis we need data about the execution of programs and tools to visualize the data to get an insight of the bottlenecks.

The main BSC tools we are going to use are:

- **Extrae**[7]: This is the central tool of all the collection of tools. It allows the user to hook up Extrae to the application and gather data during the execution. We can trace events such as the MPI calls, PAPI[8] counters, other programming models events and even manually instrument the code using Extrae API. All the data extracted is dumped into a file we call trace.
- **Basic Analysis** is a framework that given a set of traces automatically computes metrics such as POP metrics.
- **Paraver**[9][10] is a desktop application that allows to load up traces and visualize the executions. Figure 2 shows a trace visualization using Paraver. Y-axis displays processes, the x-axis represents time, and the colour represents what the process is doing at a certain time, in this case blue means running the application and black means not computing, either it can be idle waiting for a message or waiting for other processes or waiting for I/O...

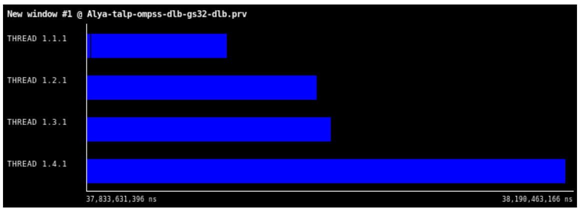

Figure 2: Sample Paraver trace

1.3 Problem to be resolved

This project is motivated by the Propulsion Technologies BSC group applying to a POP Performance Analysis assessment.

The problem to solve is directly defined by the objectives in Section 2.

1.4 Stakeholders

The following groups will benefit from the work.

1.4.1 Propulsion Technologies BSC group

This is the principal stakeholder as it is the developer of the module and the applicant for the analysis.

1.4.2 Scientific community

This stakeholder is also benefited from the job since the results and conclusions of the study are public and open to the scientific community.

2 Justification

For studying Alya as said we will use the POP methodology because:

- The study is a POP service.
- It is a well-known method and has been in part developed inside BSC.
- It is the methodology I have learnt during my formation period in BSC.

We will use the BSC toolset for extracting data to apply the POP methodology because:

- They are well-integrated with the POP methodology.
- The tools are well-known in BSC.
- It is the toolset I have learnt during my formation period in BSC.

3 Scope

In this section you will find a definition of the objectives of the projects and the identification of possible risks.

3.1 Objectives

The project is divided into two main objectives.

- Perform a rigorous Performance Analysis following the POP methodology and identify bottlenecks.
- Knowing the bottlenecks, propose, implement and test optimizations to the code.

Other sub-objectives are:

- Learn and gain experience on the performance analysis and optimization field.
- Write an easy to understand documentation about all the stages of the project making emphasis on the POP methodology.

3.2 Potential obstacles and risks

The following potential obstacles and risks have been identified:

- **Misinterpretation of a metric during the analysis:** This risk can lead the study to incorrect conclusions. However, by daily feedback with the director of the work and bi-monthly group meetings, we minimize this risk.
- **Incorrect runs:** This is a common risk that occurs when performing different tests of an application forgetting to change a parameter and leading to an incorrect run. These are usually easy to detect as numbers obtained does not match the expected. Still, if the risk is not detected the feedback received during the development should avoid this risk.
- **Machine noise:** Large supercomputer machines use complex environments that can lead to system noise. System noise refers to abnormal events that interrupt the application making counters and timings to have erratic values. Take an appropriate number of samples and doing a correct statistical analysis will avoid this risk.

4 Methodology and rigour

The methodology relays in constant feedback between Marta Garcia (the project director) and me. As tasks are sequential, once a task is done the results are shared with the director and conclusions and definition of the next task is discussed. The feedback and the discussion are done via email, personal meetings or group meetings.

This work has been developed during the global pandemic of COVID-19 so telematic meetings are contemplated.

Another important aspect to cover is "Version Control" of the project. A git repository has been created for keeping track of the POP deliverables and changes to the code.

For storing the data we will use a "google spreadsheet" data sheet with the corresponding information about the runs and the tests performed. However, trace files and run logs are kept inside the MareNostrum 4 shared storage.

References

[1] M. Vazquez, G. Houzeaux, S. Koric, A. Artigues, J. Aguado-Sierra, R. Aris, D. Mira, H. Calmet, F. Cucchietti, H. Owen, A. Taha, and J. M. Cela, "Alya: Towards exascale for engineering simulation codes," 2014.

[2] "Performance optimisation and productivity," POP-coe, accesed on 26/09/2020. [Online]. Available: https://pop-coe.eu/

[3] C. Kessler and J. Keller, "Models for parallel computing: Review and perspectives," *Mitteilungen - Gesellschaft für Informatik e.V., Parallel-Algorithmen und Rechnerstrukturen*, vol. 24, p. 13–29, 2007. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-40734

[4] D. W. Walker and J. J. Dongarra, "Mpi: a standard message passing interface," *Supercomputer*, vol. 12, pp. 56–68, 1996.

[5] "Marenostrum 4 techincal information," Barcelona Supercomputing Center, accessed on 26/09/2020. [Online]. Available: https://www.bsc.es/marenostrum/marenostrum/technical-information

[6] M. Wagner, S. Mohr, J. Giménez, and J. Labarta, "A structured approach to performance analysis," in *International Workshop on Parallel Tools for High Performance Computing*. Springer, 2017, pp. 1–15.

[7] "Bsc tools extrae website," Barcelona Supercomputing Center, accessed on 26/09/2020. [Online]. Available: https://tools.bsc.es/extrae

[8] D. Terpstra, H. Jagode, H. You, and J. Dongarra, "Collecting performance data with papi-c," in *Tools for High Performance Computing 2009*. Springer, 2010, pp. 157–173.

[9] V. Pillet, J. Labarta, T. Cortes, and S. Girona, "Paraver: A tool to visualize and analyze parallel code," in *Proceedings of WoTUG-18: transputer and occam developments*, vol. 44, no. 1. Citeseer, 1995, pp. 17–31.

[10] "Bsc tools paraver website," Barcelona Supercomputing Center, accessed on 26/09/2020. [Online]. Available: https://tools.bsc.es/paraver