Abstract

Computational science is an ever-expanding research field. It combines technologies, modern computational methods, and simulations to address problems too complex to be effectively predicted by theory alone or too expensive or dangerous to be reproduced in the laboratory. This scientific domain is a multidisciplinary field and impacts several sciences, engineering, and humanities problems. The success of the computational science approach has resulted in an increasing demand for computing resources to improve the performance of solutions and enable the growth of models, both in size and quality. For these reasons, parallel and distributed computing paradigms and exploiting Cloud Computing have become essential in computational scientists' everyday lives. Cloud provides a huge amount of computational power, easily accessible to everyone in a price-aware manner. As a result, the notion of "scalability" of an application, running over parallel or distributed systems, has become central in computational science. In a nutshell, an application is scalable if it can efficiently exploit an increasing amount of computational power (e.g., number of nodes or processors). In this domain, a relevant research challenge is to provide scalability at different levels, from software libraries to frameworks and tools for helping the solution of scientific problems. Providing scalability permits scientists to face massive and complex problems in a transparent and easy as possible way. This dissertation discusses frameworks and parallel languages that allow scientists to approach computational science problems under the lens of the extreme scalability requirement. Contributions of this work can be summarized in three principal categories: languages and tools, Agent-based Model (ABM) and simulation, and Optimization via Simulation (OvS).

Many real-world scientific applications consist of orchestrating several different independent tasks or methods for accomplishing a particular workload. These workflows are usually computationally and time demanding, thus exploiting parallel and distributed techniques became essential. This dissertation presents FLY, a domain-specific language for scientific applications, which aims at reconciling Cloud and High-Performance Computing

paradigms. FLY adopts a multi-cloud approach by providing a powerful, effective, and pricing-efficient tool for developing scalable workflow-based scientific applications. FLY exploits different and at the same time Function-as-a-Service cloud providers as computational backends in a transparent manner. This dissertation describes the programming model of FLY, its language definition, and the FLY source-to-source compiler. Furthermore, it is discussed a performance evaluation of FLY on a popular benchmark for distributed computing frameworks, along with a collection of case studies with an analysis of their performance results and costs. Finally, a real use case scenario is shown that implements an Optimization via Simulation process using FLY for carrying out a distributed evaluation of simulations on an AWS backend.

ABM is a bottom-up modeling approach, where independent decision-making agents model a complex system. Large-scale emergent behavior in ABMs is affected by the dimension of the population and the complexity of each agent's behavior. However, the computational cost of the simulation grows together with the increasing of model details. This dissertation presents the architecture and the implementation of an open-source library for developing Agent-Based Models using the Rust language. Rust-AB is able to exploit both sequential and parallel computing platforms. An investigation on the ability of Rust to develop ABM simulations are discussed, as well as several models developed with Rust-AB are described. Finally, a performance comparison against the well-known Java ABM toolkit MASON is also presented.

OvS refers to techniques for discovering the parameters of a complex model by optimizing one or more objective functions, which can only be computed by running a simulation. Due to the high dimensionality of the search space, the heterogeneity of the parameters, and the stochastic nature of the objective evaluation function, optimizing such a simulation is extremely computational demanding. This dissertation discusses methods for exploiting parallel/distributed systems' computational power to improve the efficiency and effectiveness of OvS strategies. Specifically, three frameworks are presented that differ for their underlying computing system architecture adopted: i) heterogeneous – where CPU and GPU are used to execute simulations in a distributed system composed of a heterogeneous node in terms of hardware and software, ii) homogeneous – the computing system is composed of homogeneous nodes where are used MASON (simulation library) and ECJ (optimization library) software for elaborate the OvS process, and iii) cloud computing – the computing system is a MapReduce cluster running over the cloud.