Scalable Computational Science

Carmine Spagnuolo

March 6, 2017

Università degli Studi di Salerno

Dipartimento di Informatica

Dottorato di Ricerca in Informatica e Ingegneria dell'Informazione

DOCTOR OF PHILOSOPHY

Computer Science

Parallel and Distributed Computing

Scalable Computational Science

Carmine Spagnuolo

Supervisor Prof. Vittorio Scarano

Supervisor Dott. Gennaro Cordasco

2017

Carmine Spagnuolo
Scalable Computational Science
Parallel and Distributed Computing, Supervisors: Prof. Vittorio Scarano and Dott. Gennaro Cordasco

Università degli Studi di Salerno

Dottorato di Ricerca in Informatica e Ingegneria dell'Informazione
Dipartimento di Informatica
Via Giovanni Paolo II, 132
84084 and Salerno

Abstract

Computational science also know as scientific computing is a rapidly growing novel field that uses advanced computing in order to solve complex problems. This new discipline combines technologies, modern computational methods and simulations to address problems too complex to be reliably predicted only by theory and too dangerous or expensive to be reproduced in laboratories.

Successes in computational science over the past twenty years have caused demand of supercomputing, to improve the performance of the solutions and to allow the growth of the models, in terms of sizes and quality. From a computer scientist's perspective, it is natural to think to distribute the computation required to study a complex systems among multiple machines: it is well known that the speed of single-processor computers is reaching some physical limits. For these reasons, parallel and distributed computing has become the dominant paradigm for computational scientists who need the latest development on computing resources in order to solve their problems and the "Scalability" has been recognized as the central challenge in this science.

In this dissertation the design and implementation of *Frameworks, Parallel Languages* and *Architectures*, which enable to improve the state of the art on *Scalable Computational Science*, are discussed.

Frameworks. The proposal of D-MASON, a distributed version of MASON, a well-known and popular Java toolkit for writing and running Agent-Based Simulations (ABSs). D-MASON introduces a framework level parallelization so that scientists that use the framework (e.g., a domain expert with limited knowledge of distributed programming) could be only minimally aware of such distribution. D-MASON, was began to be developed since 2011, the main purpose of the project was overcoming the limits of the sequentially computation of MASON, using distributed computing. D-MASON enables to do more than MASONin terms of size of simulations (number of agents and complexity of agents behaviors), but allows also to reduce the simulation time of simulations written in MASON. For this reason, one of the most important feature of D-MASON is that it requires a limited number of changing on the MASON's code in order to execute simulations on distributed systems.

D-MASON, based on Master-Worker paradigm, was initially designed for heterogeneous computing in order to exploit the unused computational resources in labs, but it also provides functionality to be executed in homogeneous systems (as HPC systems) as well as cloud infrastructures.

The architecture of D-MASON is presented in the following three papers, which describes all D-MASON layers:

- Cordasco G., Spagnuolo C. and Scarano V. *Toward the new version of D-MASON: Efficiency, Effectiveness and Correctness in Parallel and Distributed Agent-based Simulations.* 1st IEEE Workshop on Parallel and Distributed Processing for Computational Social Systems. IEEE International Parallel & Distributed Processing Symposium 2016.
- Cordasco G., De Chiara R., Mancuso A., Mazzeo D., Scarano V. and Spagnuolo C. *Bringing together efficiency and effectiveness in distributed simulations: the experience with D-MASON.* SIMULATION: Transactions of The Society for Modeling and Simulation International, June 11, 2013.
- Cordasco G., De Chiara R., Mancuso A., Mazzeo D., Scarano V. and Spagnuolo C. *A Framework for distributing Agent-based simulations.* Ninth International Workshop Algorithms, Models and Tools for Parallel Computing on Heterogeneous Platforms of Euro-Par 2011 conference.

Much effort has been made, on the Communication Layer, to improve the communication efficiency in the case of homogeneous systems. D-MASON is based on Publish/Subscribe (PS) communication paradigm and uses a centralized message broker (based on the Java Message Service standard) to deal with heterogeneous systems. The communication for homogeneous system uses the Message Passing Interface (MPI) standard and is also based on PS. In order to use MPI within Java, D-MASON uses a Java binding of MPI. Unfortunately, this binding is relatively new and does not provides all MPI functionalities. Several communication strategies were designed, implemented and evaluated. These strategies were presented in two papers:

- Cordasco G., Milone F., Spagnuolo C. and Vicidomini L. *Exploiting D-MASON on Parallel Platforms: A Novel Communication Strategy* 2st Workshop on Parallel and Distributed Agent-Based Simulations of Euro-Par 2014 conference.
- Cordasco G., Mancuso A., Milone F. and Spagnuolo C. *Communication strategies in Distributed Agent-Based Simulations: the experience with D-MASON* 1st Workshop on Parallel and Distributed Agent-Based Simulations of Euro-Par 2013 conference.

D-MASON provides also mechanisms for the visualization and gathering of the data in distributed simulation (available on the Visualization Layer). These solutions are presented in the paper:

- Cordasco G., De Chiara R., Raia F., Scarano V., Spagnuolo C. and Vicidomini L. *Designing Computational Steering Facilities for Distributed Agent Based Simulations.* Proceedings of the ACM SIGSIM Conference on Principles of Advanced Discrete Simulation 2013.

In DABS one of the most complex problem is the partitioning and balancing of the computation. D-MASON provides, in the Distributed Simulation layer, mechanisms for partitioning and dynamically balancing the computation. D-MASON uses field partitioning mechanism to divide the computation among the distributed system.

The field partitioning mechanism provides a nice trade-off between balancing and communication effort. Nevertheless a lot of ABS are not based on 2D- or 3D-fields and are based on a communication graph that models the relationship among the agents. In this case the field partitioning mechanism does not ensure good simulation performance.

Therefore D-MASON provides also a specific mechanisms to manage simulation that uses a graph to describe agent interactions. These solutions were presented in the following publication:

- Antelmi A., Cordasco G., Spagnuolo C. and Vicidomini L.. *On Evaluating Graph Partitioning Algorithms for Distributed Agent Based Models on Networks*. 3rd Workshop on Parallel and Distributed Agent-Based Simulations of Euro-Par 2015 conference.

The field partitioning mechanism, intuitively, enables the mono and bi-dimensional partitioning of an Euclidean space. This approach is also know as uniform partitioning. But in some cases, e.g. simulations that simulate urban areas using a Geographical Information System (GIS), the uniform partitioning degrades the simulation performance, due to the unbalanced distribution of the agents on the field and consequently on the computational resources. In such a case, D-MASON provides a non-uniform partitioning mechanism (inspired by Quad-Tree data structure), presented in the following paper:

- Lettieri N., Spagnuolo C. and Vicidomini L.. *Distributed Agent-based Simulation and GIS: An Experiment With the dynamics of Social Norms*. 3rd Workshop on Parallel and Distributed Agent-Based Simulations of Euro-Par 2015 conference.
- G. Cordasco and C. Spagnuolo and V. Scarano. *Work Partitioning on Parallel and Distributed Agent-Based Simulation*. IEEE Workshop on

Parallel and Distributed Processing for Computational Social Systems of International Parallel & Distributed Processing Symposium, 2017.

The latest version of D-MASON provides a web-based System Management, to better use D-MASON in Cloud infrastructures. D-MASON on the Amazon EC2 Cloud infrastructure and its performance in terms of speed and cost were compared against D-MASON on an HPC environment. The obtained results, and the new System Management Layer are presented in the following paper:

- M Carillo, G Cordasco, F Serrapica, C Spagnuolo, P. Szufel, and L. Vicidomini. *D-Mason on the Cloud: an Experience with Amazon Web Services.* 4rd Workshop on Parallel and Distributed Agent-Based Simulations of Euro-Par 2016 conference.

Parallel Languages. The proposal of an architecture, which enable to invoke code supported by a Java Virtual Machine (JVM) from code written in C language. Swft/T, is a parallel scripting language for programming highly concurrent applications in parallel and distributed environments. Swift/T is the reimplemented version of Swift language, with a new compiler and runtime. Swift/T improve Swift, allowing scalability over 500 tasks per second, load balancing feature, distributed data structures, and dataflow-driven concurrent task execution.

Swif/T provides an interesting feature the one of calling easily and natively other languages (as Python, R, Julia, C) by using special language functions named leaf functions. Considering the actual trend of some supercomputing vendors (such as Cray Inc.) that support in its processors Java Virtual Machines (JVM), it is desirable to provide methods to call also Java code from Swift/T. In particular is really attractive to be able to call scripting languages for JVM as Clojure, Scala, Groovy, JavaScript etc.

For this purpose a C binding to instanziate and call JVM was designed. This binding is used in Swif/T (since the version 1.0) to develop leaf functions that call Java code. The code are public available at GitHub project page.

Frameworks. The proposal of two tools, which exploit the computing power of parallel systems to improve the effectiveness and the efficiency of Simulation Optimization strategies. Simulations Optimization (SO) is used to refer to the techniques studied for ascertaining the parameters of a complex model that minimize (or maximize) given criteria (one or many), which can only be computed by performing a simulation run. Due to the the high dimensionality of the search space, the heterogeneity of parameters, the irregular shape and the stochastic nature of the objective evaluation function, the tuning of such systems is extremely demanding from the computational point of view. The first frameworks is *SOF: Zero Configuration Simulation Optimization Framework on the Cloud*, it was designed to run SO process in

the cloud. SOF is based on the Apache Hadoop infrastructure and is presented in the following paper:

- Carillo M., Cordasco G., Scarano V., Serrapica F., Spagnuolo C. and Szufel P. *SOF: Zero Configuration Simulation Optimization Framework on the Cloud.* Parallel, Distributed, and Network-Based Processing 2016.

The second framework is *EMEWS: Extreme-scale Model Exploration with Swift/T*, it has been designed at Argonne National Laboratory (USA). EMEWS as SOF allows to perform SO processes in distributed system. Both the frameworks are mainly designed for ABS. In particular EMEWS was tested using the ABS simulation toolkit Repast. Initially, EMEWS was not able to easily execute out of the box simulations written in MASON and NetLogo. This thesis presents new functionalities of EMEWS and solutions to easily execute MASON and NetLogo simulations on it.

The EMEWS use cases are presented in the following paper:

- J. Ozik, N. T. Collier, J. M. Wozniak and C. Spagnuolo *From Desktop To Large-scale Model Exploration with Swift/T.* Winter Simulation Conference 2016.

Architectures. The proposal of an open-source, extensible, architecture for the visualization of data in HTML pages, exploiting a distributed web computing. Following the Edge-centric Computing paradigm, the data visualization is performed edge side ensuring data trustiness, privacy, scalability and dynamic data loading. The architecture has been exploited in the Social Platform for Open Data (SPOD). The proposed architecture, that has also appeared in the following papers:

- G. Cordasco, D. Malandrino, P. Palmieri, A. Petta, D. Pirozzi, V. Scarano, L. Serra, C. Spagnuolo, L. Vicidomini *A Scalable Data Web Visualization Architecture.* Parallel, Distributed, and Network-Based Processing 2017.
- G. Cordasco, D. Malandrino, P. Palmieri, A. Petta, D. Pirozzi, V. Scarano, L. Serra, C. Spagnuolo, L. Vicidomini *An Architecture for Social Sharing and Collaboration around Open Data Visualisation.* In Poster Proc. of the 19th ACM conference on "Computer-Supported Cooperative Work and Social Computing 2016.
- G. Cordasco, D. Malandrino, P. Palmieri, A. Petta, D. Pirozzi, V. Scarano, L. Serra, C. Spagnuolo, L. Vicidomini *An extensible architecture for an ecosystem of visualization web-components for Open Data* Maximising interoperability Workshop— core vocabularies, location-aware data and more 2015.