Verification of Simulation via Reproducibility

Abstract of Contributed Talk to REPPAR 2016 Workshop

Franziska Hoffeins[1], Florina M. Ciorba[2], Ioana Banicescu[3], and Wolfgang E. Nagel[1]

[1] Technische Universität Dresden {franziska.hoffeins,wolfgang.nagel}@tu-dresden.de
[2] University of Basel florina.ciorba@unibas.ch
[3] Mississippi State University ioana@cse.msstate.edu

Scientific applications are often large in time and/or space, computationally intensive, data parallel, and irregular. A dominant performance degradation factor is load imbalance. Load imbalance occurs due to application, systemic, and/or algorithmic variability. Dynamic load balancing can be used to achieve load balanced execution of applications with unpredictable changing of workload, or if processing elements differ in performance, or if perturbations in the system or in the network occur. Scientific applications often consist of iterative computations in the form of computationally intensive loops. Such loops represent the underlying numerical model, which may require a large amount of time steps or comprise a large amount of data points of the computational domain. These loops are a rich source of parallelism.

The research area dynamic loop scheduling (DLS) addresses algorithmic and systemic sources of load imbalance by dynamically assigning chunks of iterations to processing elements. Over the years, different loop scheduling techniques were developed and it is proven that these techniques are very successful in balancing applications' work load. The use of DLS techniques is not restricted to loops, when iterations are viewed as independent tasks.

Real testbeds are time intensive to create and have limited control over the dynamic behavior of the computing system. Therefore, it is challenging to perform repeated and controlled experiments of scheduling scientific applications on real computer systems. SimGrid [4] is a simulation framework providing the ability to rapidly prototype and evaluate dynamic loop scheduling techniques applied in different types of applications. In addition, it provides an adequate level of abstraction and simulation scalability.

In this talk I will present results from ongoing work in the reproducibility of scheduling experiments using DLS techniques published in earlier literature. Among others, I will describe the reproducibility process of simulations published by T. Hagerup[5]. In this paper the author proposed a new DLS technique - the BOLD strategy - and compared the results when applying this technique to different types of applications and systems to the earlier ones, static chunking, self scheduling, fixed size chunking, guided self scheduling, trapezoid self scheduling, and factoring (a review of these techniques and later ones can be found here[6]).

The DLS techniques have been implemented in SimGrid (MSG module) and the reproducibility is carried out by comparing the SimGrid-based experiments with those reported in the literature. Additionally, I will also discuss the challenges I have encountered in reproducing the experiments.

Once the SimGrid implementation is verified, the impact of the overhead of dynamic loop scheduling techniques on the performance of scientific applications in heterogeneous computing systems can be assessed. The overhead can also be quantified by analytical and experimental analysis. The purpose of this study is obtaining an accurate modelling of the overhead of the DLS techniques, such that the most suitable one can be selected for improving the application and/or system performance.

---

[4] H. Casanova, A. Giersch, A. Legrand, M. Quinson, and F. Suter. *Versatile, Scalable, and Accurate Simulation of Distributed Applications and Platforms.* Journal of Parallel and Distributed Computing, 74(10):2899-2917, 2014.

[5] T. Hagerup. *Allocating Independent Tasks to Parallel Processors: An Experimental Study.* Journal of Parallel and Distributed Computing, 1997.

[6] I. Banicescu and R. L. Cariño. *Addressing the Stochastic Nature of Scientific Computations via Dynamic Loop Scheduling.* Electronic Transactions on Numerical Analysis - Special Issue on Combinatorial Scientific Computing, 21:66-80, 2005.