Complexity and Accuracy Trade-off Analysis of Parallel Application Simulation Using SST/Macro

Zhou Tong, Xin Yuan
Department of Computer Science
Florida State University
Tallahassee, Florida, USA
{tong, xyuan}@cs.fsu.edu

Scott Pakin, Michael Lang
Computer, Computational, and Stat. Sci. Div.
Los Alamos National Laboratory
Los Alamos, NM, USA
{pakin,mlang}@lanl.gov

I. Introduction

Modeling and simulating of HPC applications on contemporary supercomputing platforms is challenging due to the size of the increasingly growing system and application size. To understand the performance characteristics, parallel applications are often simulated at different levels including flow-level, packet- level, and flit-level. It is commonly believed that more detailed simulation will result in more useful information about the applications. However, it remains unclear how much more accurate simulation results finer-grained simulations can deliver at the cost of higher simulation complexity. In this study, we investigate the trade-off between modeling and simulation complexity and accuracy using the macroscale component of the Structured Simulation Toolkit (SST) [1] and a modeling tool [4]. We measure the performance of a set of parallel applications on four current generation supercomputers, analyze the modeling and simulation results produced by the tools, and draw conclusions from the analysis of the statistics about the trade-off between complexity and accuracy of modeling and simulation. Our results indicates that coarse-grained simulations with higher simulation complexity does not guarantee more accurate application performance prediction. The results may be used to guide the selection of proper simulation and modeling study of parallel applications and systems, which is often a time-consuming task. Moreover, the results also help better understand the modeling and simulation results of parallel applications and systems.

II. Modeling and Simulation of Parallel Applications

This trade-off study is performed by collecting execution traces in the DUMPI format [2] by a set of representative applications on four current generation supercomputers. From the traces, the application statistics such as total execution time and communication time, for the physical execution of the application on the supercomputers are obtained. We then use a modeling tool and SST/Macro to replay the traces using the physical machine setting so that the modeled and simulated performance statistics for the applications on the supercomputers. By analyzing the statistics of the measured, modeled, and simulated results, we draw conclusions on the trade-off between complexity and accuracy, answer key questions regarding modeling and simulation such as "does more detailed simulation yield more accurate results?"

The trace-driven fast classification tool predicts application performance for many latency and bandwidth parameters in one simulation using Hockney's model ($\alpha + \beta m$), where α is the latency, β is the inverse of bandwidth and m is the message size. It uncovers the relative benefits of various networking options and identifies application performance bottlenecks [4].

SST/Macro supports three network congestion models: packet model, flow model and a hybrid packet-flow model. Packet-level modeling segments message into small transmission units as packets and routed individually through the network switches. Figure 1(A) shows when two messages compete for the same channel, arbitration occurs and decides which packet has the right-of-way. Packets that lose arbitration are delayed, leading to network congestion. In the flow model, messages traverse the network as a fluid, sharing link bandwidth between competing flows as in Figure 1(B). Additionally, arbitration can occur on flits, the smallest flow control units that are divided into few-byte chunks by hardware. However, flit-level arbitration is usually computationally prohibitive for fine-grained large-scale system simulation. Like the packet model, the packet-flow model routes fixed-sized packets, but channel arbitration is not exclusive. When multiple packets compete for a channel, each packet "samples" the congestion, then estimates its expected bandwidth and latency using a greedy algorithm. Figure 1(C) illustrates how the experienced delay is represented by the length of each packet.

In summary, the modeling technique is fast and scalable. It models application communication performance with little overhead, but it does not consider the network contention. Flow model simulates the network contention at the flow level and it requires two events, start and finish, to be modeled under zero network congestion, much faster than the packet level. However, the flow model is subject to the

"ripple effect" as flow updates propagate across the system under heavy network congestion or large systems. Packet model provides details about packet arbitration and flow control, but its complexity is proportional to the number of packets events updates. It also systematically overestimates (de)serialization latency and exclusively reserves network links for the entire length of the packet. The hybrid packet-flow model uses a sampling procedure to allow multiplexing as in flow model and models messages as discrete chunks as in packet mode, avoiding latency overestimation errors as well as the ripple effect [3].

III. Data Collection and Trade-off Analysis

In this study, we measure the performance of a set of parallel applications and analyze the modeling and simulation results, then draw conclusions from the analysis of the statistics about the trade-off between complexity and accuracy of modeling and simulation.

A. Trace Collection

We collect DUMPI traces from various application runs on Edison, Hopper, Mustang and Cielo/Cielito. Table I shows a list of available applications traces on DOE's Design Forward Web Site that will be used in this study.

B. Trade-off Analysis

We measure the physical execution time of these applications on the host systems. The modeling tool predicts application's sensitivity to the speed-up and slow-down of the network, and provides estimation of the relative benefits of various networking options in one run. With SST/macro off-line simulation mode, we simulates the performance of application on the host systems by using the system parameters measured via benchmarking techniques and task-mapping recorded in the DUMPI traces.

By analyzing the measurement results, the modeled results, and the simulated (at different levels), we can quantify how accurate are the modeling technique and SST/Macro's simulations. Then, we can correlate performance results based on these three measurements, 1) physical application runtime, 2) the fast classification tool and 3) SST/Macro's estimated application runtime to obtain the trade-off results.

C. Empirical Results

In this section, we show some preliminary results using the Crystal Router (CR) traces collected on Hopper and Mustang and 7 NAS benchmarks to demonstrate the scope of trade-off study between the simulation complexity and simulation accuracy. Table I shows that the percentage of communication time increases from 14% to 69% as more ranks are deployed. In Figure 6, the modeling tool indicates that CR is sensitive to the slow-down and speedup of network.

We simulate CR with 10 MPI ranks using traces collected on Hopper to demonstrate the trade-off between the simulation complexity and simulation accuracy. Figure 2 depicts that packet model, flow model and packet-flow model all make estimations reasonably close to the observed physical execution time. However, Figure 3 shows that packet model and flow model's simulation accuracy with error rate of 16.7% and 21.1%, respectively, yet the packet-flow model offers 12.7% error rate and significantly lower simulation overhead. This observation is consistent with the design of packet-flow model that addresses the drawbacks of both packet and flow model in section II.

Figure 4 and Figure 5 illustrate the trade-off analysis of 7 NAS benchmarks. In Figure 4, we show the simulation error of packet-level, flow-level, packet-flow and the fast classification tool. Among four simulation models, the hybrid packet-flow model provides the best simulation accuracy for most cases while packet-level simulation trails at the second with a small margin. In Figure 5, on the other hand, shows that packet-flow and flow-level simulation are roughly one magnitude faster than packet-level simulation in terms of simulation overhead.

IV. Future Work & Conclusions

We have presented some preliminary results for the trade-off analysis of simulation complexity and simulation accuracy using physical measurement, the fast classification tool and SST/Macro. Packet-level simulation does not guarantee more accurate application performance prediction than the hybrid packet-flow model, but at a much higher simulation overhead. The packet-flow model could be served as a more efficient replacement for simulators with packet-level or flow-level models, while our fast classification provides fast performance predictions with at least an order of magnitude less simulation overhead and competitive simulation accuracy. The next step of the trade-off analysis requires the collection of DUMPI traces for a more comprehensive trade-off analysis.

References

[1] H. Adalsteinsson, S. Cranford, D. A. Evensky, J. P. Kenny, J. Mayo, A. Pinar, and C. L. Janssen. A simulator for large-scale parallel computer architectures. *Int. J. Distrib. Syst. Technol.*, 1(2):57–73, Apr. 2010.

[2] S. Corporation. SST: The structural simulation toolkit. 2014. [Online; accessed 14-DEC-2012].

[3] S. N. Labs. SST/macro 6.0: User's manual. May 2016. [Online; accessed 14-JUN-2016].

[4] Z. Tong, S. Pakin, M. Lang, and X. Yuan. Fast classification of mpi applications using lamport's logical clocks. In *Parallel and Distributed Processing, 2016. IPDPS 2016. IEEE International Symposium on*, pages 1–8. IEEE, 2016.