Proceedings of the 2017 Winter Simulation Conference
W. K. V. Chan, A. D'Ambrogio, G. Zacharewicz, N. Mustafee, G. Wainer, and E. Page, eds.

SOME PROPERTIES OF COMMUNICATION BEHAVIORS IN DISCRETE-EVENT SIMULATION MODELS

Patrick Crawford

Dept of EECS, PO Box 210030
University of Cincinnati
Cincinnati, OH 45221, USA

Stephan J. Eidenbenz

Information Science & Technology Institute
Los Alamos National Laboratory
Los Alamos, NM 87545, USA

Peter D. Barnes Jr.

Lawrence Livermore National Laboratory
7000 East Avenue
Livermore, CA 94550, USA

Philip A. Wilsey

Dept of EECS, PO Box 210030
University of Cincinnati
Cincinnati, OH 45221, USA

ABSTRACT

This paper summarizes the profile data captured from 22 discrete-event simulation models captured from 4 different simulators. The profile data is captured by instrumenting the simulation engine and does not require any modification to the models. The profile data reported focuses on the communication properties of events exchanged between the various processes (LPs) in the model. The data suggests that some models share common behaviors and this work summaries their general characteristics. This permits a presentation of the principle characteristics using only six of the studied models. This resulting information can be used to: (i) provide configuration data for synthetic model generation, (ii) provide direction for configuring and optimizing parallel and distributed simulation engines, and (iii) provide insights into model correctness. The focus of this specific study is to determine regularities among LP event communications that can be exploited for model partitioning and event scheduling in parallel simulation.

1 INTRODUCTION

The quantitative analysis of program structures and execution profiles has been successfully exploited by the computer architecture community to develop and deploy highly effective optimizations to processor implementations for several decades (Hennessy and Patterson 2012). Most likely, the application of quantitative profiles of simulation models can likewise be exploited to enhance and optimize tools and techniques for the simulation community. In particular, within the field of simulation, quantitative measures from existing simulation models could potentially be applied in multiple ways. For example, quantitative data can be used to (i) optimize simulation kernel algorithms (Gupta and Wilsey 2017), (ii) provide model profiles to help configure synthetic workload generators (Balakrishnan et al. 2001, Ferscha and Johnson 1996, Fujimoto 1990, Park et al. 2015), and (iii) (potentially) aid in the verification and validation of simulation models; by providing, for example, observable profiles of event processing and communication behaviors that can help verify the model structure against expectations for the constructed model.

This paper explores that quantitative profiling and analysis of event communication behaviors in Discrete Event Simulation (DES) models. Early work has already emerged to develop quantitative profiling tools (Wilsey 2016) for DES models. Furthermore, some preliminary work to optimize the event scheduling algorithms for parallel simulation has also been performed (Gupta and Wilsey 2017). While the earlier

works were focused on event execution behaviors, this paper will examine the characteristics of event exchanges between the processing objects (LPs) of six representative DES models. These six models were selected from a total of 22 different simulation models that were analyzed as a part of this study. Fortunately, the profile characteristics of all 22 models reflect common structures that make presentation of results from six of the models meaningfully demonstrative of the general characteristics observed. The model data was captured from four different simulation kernels, namely: ROSS (Carothers et al. 2000), NS-3 (Henderson et al. 2008, Riley and Henderson 2010), Simian (Santhi et al. 2015), and WARPED2 (Weber 2016). These simulators have all been instrumented to capture profile data from any simulation model executed by these kernels. The specific models used are models taken from the repositories of those simulation environments and they were not developed specifically for this study.

The remainder of this paper is organized as follows. Section 2 contains some background information and discusses related work. Section 3 provides a high level overview of profiling DES models in the DESMetrics project. Section 4 presents the simulation kernels and simulation models studied and reported herein. Section 5 presents the quantitative data captured from these simulation models. Finally, Section 6 contains some concluding remarks.

2 BACKGROUND AND RELATED WORK

Studying the properties of DES models can aid researchers and model developers in a variety of ways. For example, developers of high-performance parallel simulation engines can use profile data to improve simulation kernel performance. Likewise, tool builders and modelers alike can use profile data to construct and configure synthetic model generators (Balakrishnan et al. 2001, Ferscha and Johnson 1996, Fujimoto 1990, Park et al. 2015). Such generators are often useful to derive larger models of problems for study that would otherwise be unavailable or, at least, very difficult to build. While such model generators have been widely used for many years, the capture of profile data from smaller hand built models can provide valuable information that can be used to configure more accurate model generation. Profile data can also be used to help setup and configure simulation and analysis tools for more effective execution. For example, profile data on events exchanged between LPs from a pre-simulation run can be used to guide profile guided partitioning tools (Alt and Wilsey 2014). While profile guided partitioning is already widely used, more extensive profiling results can also be used more extensively to tune model analysis tools. A demonstration of this can be seen in (Gupta and Wilsey 2017). Finally, the use of profile data can be used to aid in the validation and verification of simulation models. In fact, during this study of simulation model characteristics, the authors discovered a model that did not correctly implement the desired features. As a result, the model was flagged and removed from this study.

Early work analyzing properties of DES models was directed toward the analysis of the amount of parallelism available or the lookahead properties of the simulation. More specifically a technique called *critical path analysis* (Berry and Jefferson 1985, Jefferson and Reiher 1991, Livny 1985, Lin 1992) is a computation to locate the shortest path through the collection of events in a DES. Lookahead analysis is a method to relax the causal time chain between a source and destination LP in a conservatively synchronized parallel simulation (Fujimoto 1989). A generalized approach to capture, analysis, and visualize profile data from DES models was developed in a project called DESMetrics (Wilsey 2016). The the application of information gleaned from the DESMetrics project to optimize a parallel simulation kernel is reported in (Gupta and Wilsey 2017). This paper continues and extends the visualization tools within the DESMetrics project to expand the study of event communication properties of DES models. The next section presents a brief overview of the DESMetrics project. Readers interested in a more detailed presentation of the techniques and tools of the DESMetrics project will find them here (Wilsey 2016).

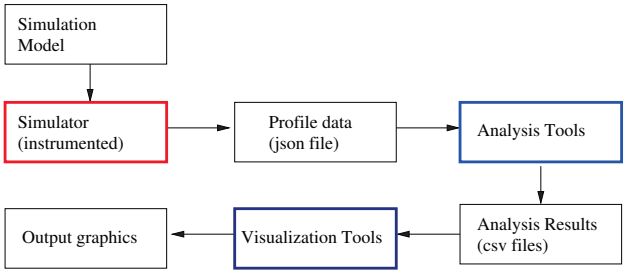


Figure 1: The DESMetrics tool flow.

3 PROFILING DES MODELS

The DESMetrics project contains analysis tools and visualization tools that process event trace data from DES models (Wilsey 2016). The trace (profile) data is captured separately by external methods. Although it is possible to add output commands to the simulation models to capture this data, the recommended method is to add instrumentation to the simulation engines that execute the simulation models. This is the approach used to capture profile data for all of the simulation models that are studied in this paper.

The general tool flow for the DESMetrics project tools is illustrated in Figure 1. A json file format is used for the event profile data. Once profile data is captured, this data is processed by a go program named `desAnalysis`. The `desAnalysis` program analyzes the profile data for a variety of different attributes and produces a collection of `csv` files that are processed by a Python script using the `matplotlib` library to produce various visual representations of the analysis findings. Some of the properties analyzed and visualized include: the number of events available for parallel execution, the number of source and destination LPs that each LP sends and receives event information, the number and type (self-generated or remotely generated) events that each LP processes, the number of events in each LP at each (simulated) timestep that could be processed as a block, the amount of lookahead (smallest, average, largest) that each LP has at each simulated execution cycle, and so on. The anlysis of this paper will focus on the results characterizing events exchanged by the LPs and the couplings among the LPs that results from these exchanges. Additional details on the DESMetrics tool flow can be found at (Wilsey 2016) and at the project webpages at github.com/wilseypa/desMetrics. All of the tools in the DESMetrics project are open sourced and freely available for use.

4 THE STUDIED DES MODELS

This study has examined the profile results of 22 simulation models from 4 different discrete event simulation engines. The discrete event simulation engines are: NS-3 (Henderson et al. 2008, Riley and Henderson 2010), ROSS (Carothers et al. 2000), WARPED2 (Weber 2016), and Simian (Santhi et al. 2015). NS-3 is a highly-popular, widely used general purpose network simulation engine. It supports either sequential simulation or conservatively synchronized parallel simulation capabilities. ROSS is a general purpose simulation engine supporting sequential and parallel execution using either conservative or optimistic synchronization. It supports process level execution and has been developed and used with extreme scale parallel processing hardware. WARPED2 is a general purpose simulator that supports either sequential or optimistically synchronized parallel simulation. It is tuned for parallel execution on clusters of multi-core processors and contains both threaded and process-based parallelism. Finally, Simian is minimalist, sequential and conservatively synchronized, parallel discrete simulation engine implemented entirely in interpreted languages. A Lua Simian version is the standard reference, however, a functionally equivalent Python version called SimianPie is also available. Used with Just-In-Time compilation mode (LuaJIT or Pypy), Simian's performance matches or exceeds C-based engines, while offering the fast prototyping that interpreted languages allow. All of these simulation engines are open source software and freely available for use. The ROSS simulator is available at https://github.com/carothersc/ROSS and the ROSS Models

Name	Engine	Num of LPs	Total Events
CSMA System	NS-3	107	110,184
Lena Dual stripe	NS-3	259	101,649
Multirate	NS-3	101	71,803
Market	ROSS	207	77,850,502
NeMo	ROSS	65,664	121,372,541
PCS	ROSS	1,048,576	159,088,718
9D Torus	ROSS	1024	11,442,461
Traffic	ROSS	1,048,576	392,808,018
Viral	ROSS	18	10,242,359
IMC V4 444	Simian	5	43,029
IMC V4 888	Simian	50	518,379
IMC V4 8816	Simian	84	1,344,039
IMC V4 81616	Simian	152	3,454,656
IMC Bypass 161616	Simian	256	573,347
IMC Crossbar 161616	Simian	257	8,532,306
IMC Edison 161616	Simian	320	5,380,791
IMC Hopper 161616	Simian	422	9,292,890
IMC Moonlight 161616	Simian	288	6,403,616
Epidemic	WARPED2	10,000	253,895,687
PCS	WARPED2	10,000	135,724,327
Traffic	WARPED2	10,000	61,523,815
Volcano	WARPED2	125,000	125,605,980

Figure 2: Names and characteristics of simulation models studied.

are available at https://github.com/carothersc/ROSS-Models. ROSS has builtin event capture capabilities that dump all of the data required for the DESMetrics analysis. The WARPED2 simulation engine is available at https://github.com/wilseypa/warped2 and the WARPED2 models are available at https://github.com/wilseypa/warped2-models. The Simian simulator is available at https://pujyam.github.io/simian/ and the NS-3 simulator and models are available at https://www.nsnam.org/. The ROSS, WARPED2, and NS-3 simulation engines all contain builtin event capture capabilities that dump all of the data required for the DESMetrics analysis tools. The Simian simulation engine had to be instrumented for this study to capture the event data needed for the DESMetrics analysis tools. A summary presentation of all of the studied simulation models is given in Figure 2.

Unfortunately space considerations prevent a detailed reporting of data from all of the simulation models. However, after examining the data from all of the simulation models, there are some key characteristics that are common to various subsets of the the simulation models. These shared characteristics permit the grouping of the models together to be represented by one of a set of a base case simulation models. Therefore, this paper presents the results from six different models to highlight the principle features observed by these 22 models. These results are presented in Section 5. The remainder of this section presents a brief description of the six simulation models that are examined in detail in Section 5.

A brief description of each of the models from which detailed data is presented in this paper are:

warped2 Traffic Model: This model represents a traffic simulation where cars move through and between four-way intersections with three lanes in each direction. On arriving at an intersection, a car attempts to go in the direction that would get it closer to its target. The flow of cars going through an intersection is regulated using thresholds to limit congestion and spread out the traffic.

ROSS 9D Torus Model: This model represents the simulation of a torus network of 9 dimensions. Traffic is synthetically generated into the model with a uniform random destination routing.

ROSS PCS Model: This simulation model simulates the circuit-switched network on a cellular grid. Each cell is a simulation object which has a fixed place on a cellular grid. Portables can move from one cell to the other; this is simulated via event diffusion from one simulation object to the other on the cellular grid (Carothers et al. 1994).

warped2 Epidemic Model: This simulation model simulates the epidemic outbreak phenomena following the model description in (Barrett et al. 2008). The epidemic is modeled using a combination of reaction and diffusion processes. The reaction is defined as a result of inter-entity interactions, (*e.g.,* influenza or physical proximity).

IMC Hopper 161616: This is a synthetic model that comes from runs of a computational performance application of the Performance Prediction Toolkit (PPT) (Ahmed et al. 2016). PPT applications mimic the loop structure and behavior of computational physics codes on modeled hardware architectures. The application model (IMCSim) used is an Implicit Monte Carlo (IMC) method for radiative transport simulation of a hot box physical system, which is a standard scenario for (IMC) codes. In the Hopper $(16,16,16)$ scenario, IMCSim predicted performance of a $16^3 = 4096$ simulated MPI rank run on a hardware model of a 6,384 compute nodes, 153,216 cores, 1.28 PF system named Hopper at NERSC. IMCSim generates events for all simulated MPI packets as well as computational kernel phases and interrupts.

NS-3 Lena Dual Stripe Model: This is a model of LTE wireless network traffic among apartments in two 10-unit apartment building on each side of a street. The appartments, 10m X 10m each, contain a total 9 femto-cells, and there is one outdoor macro-cell. The model configuration studied here is the default configuration that is contained in the stanadard NS-3 code base.

Except for IMC Hopper, the remaining models are all analyzed using their default confgurations that are set by the corresponding simulator development teams. These models cover a broad range of model sizes and number of total events processed. The size an total events processed for each are given in Figure 2.

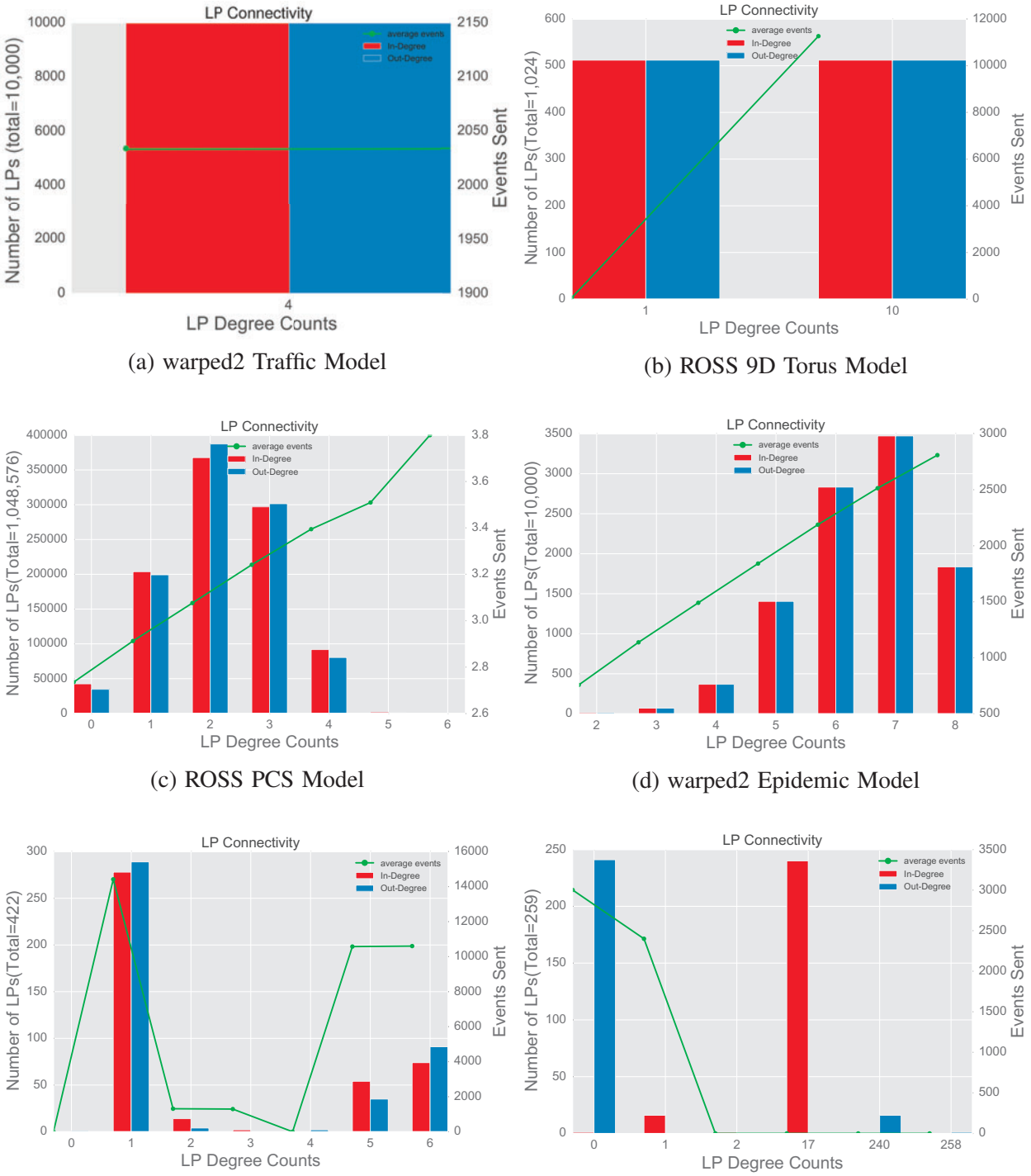
5 COMMUNICATION PROPERTIES OF THE STUDIED MODELS

The communication properties being observed for the studied models are *LP connectivity*, *Betweeness Centrality*, and *Modularity*. The idea behind this is to recognize patterns in LP by LP communication for models from different simulators. These properties and the results are described in more detail below.

5.1 LP Connectivity

LP connectivity is determined by the number of LPs any given LP is sending events to (known as *out-degree*), and the number of LPs any given LP is receiving events from (known as *in-degree*). Thus, if an LP sends events to 3 different other LPs, its out-degree is 3 and if it receives events from 5 different LPs, its in-degree is 5. Events that an LP sends to itself are not counted in the in-/out-degree computation. From this, it can be determined whether a simulation model has LPs that are processing many events from numerous LPs, or if the number of LPs that are communicating with one another are relatively distributed. Beyond showing the degree of connectivity between the LPs, this graph is also valuable in documenting to the model developer something about the structure of communications between the LPs. We have already successfully used data from these graphs to highlight a problem with one of the simulation models that have previously been developed and used erroneously. The reminder of this section describes the principle structures observed in these graphs and highlights the behaviors that are shared with some of the other 22 studied models.

Figure 3 shows this connectivity. The x-axis records the in-/out-degree value and the y-axis shows the number of LPs with that degree of connectivity. The in-degree is shown in red and the out-degree is shown in blue. While some of the graphs have equal in-/out-degree, this does not mean that each LP has the same in-/out-degree value. In fact, examining (especially) the NS-3 model, we can see a large degree



(a) warped2 Traffic Model

(b) ROSS 9D Torus Model

(c) ROSS PCS Model

(d) warped2 Epidemic Model

(e) Simian 161616, Hopper Network

(f) ns-3 Lena Dual Stripe

Figure 3: In/Out degree and average communicated events by LP.

of asymmetry in the values. Finally, these graphs show the average number of events sent by the LPs at the corresponding out-degree count.

The data shows three broad classes of in-/out-degree connectivity. The top two WARPED2 Traffic and ROSS 9D Torus show LPs that fall into one or two broad classes of connectivity. This type of connectivity is shared with the following models: ROSS Viral, WARPED2 traffic model, IMC V4 444, IMC V4 888, IMC V4 8161, IMC Moonlight, and IMC Crossbar models. This makes sense as all of these models have regular low-degree underlying network structures. The average events processed are also instructive as it tells us that the generated events tend to be dominated by one of the specific (generally the highest) out-degree connected LPs. The single exception to this is the (not shown) ROSS Viral model where the LPs with the lowest out-degree connectivity dominate.

The second broad class of LP connectivity is demonstrated by the ROSS PCS and WARPED2 Epidemic Model and the IMC Hopper 161616 Model. These models show a range of small and nearby connectivity values ranging from 0 to 10. Again the average out-events occur with the larger out-degree values. Other models that exhibit similar connectivity graphs are: IMC Bypass 161616, and IMC Edison 161616.

The third broad class of LP connectivity is illustrated by the NS-3 models. These models show much less regularity and a broad range of connectivity values up to a few hundred connections for some LPs with many large gaps in unfilled spatial connectivity values. Likewise, the average events generated are much more chaotic and unpredictable. Other models that exhibit this type of connectivity behavior are: ROSS Market, ROSS Neuromorphic (but only weakly).

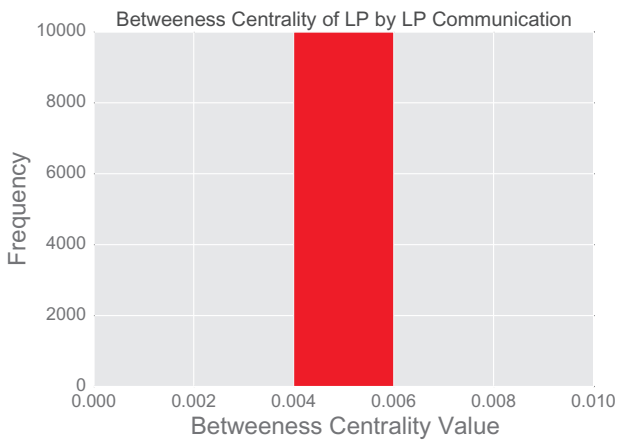
From a usage standpoint, this data is quite useful to evaluate model correctness and, as previously stated, we successfully used it to help identify a model that was not properly constructed. From the standpoint of synthetic model generation, it definitely indicates that there are connectivity properties that a generator should subscribe to replicate a specific model type. From the standpoint of optimizing the underlying algorithms of a simulation execution engine, the implications are far less clear. Perhaps the connectivity could implicate the difficulty of partitioning or event scheduling, especially if the more diverse range of connectivity structures imply a more unbalanced distribution of centrality of an LP. This question is most likely related to the Betweeness Centrality described in the next section.

5.2 Betweenness Centrality

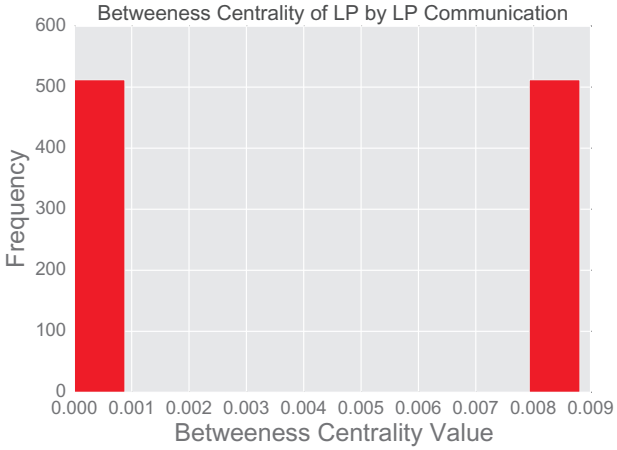
Betweenness Centrality is one measure of centrality in a network, where it values a node by how often it lies on the shortest path between every pair of nodes (Anthonisse 1971, Freeman 1977). The algorithm used is fairly efficient for relatively large networks, for up to tens of thousands of nodes (Brandes 2001). However, when a network contains millions of nodes, approximate calculations were needed using `n` nodes to estimate the betweeness values (Brandes and Pich 2007), in order to reduce the computation time. Fortunately, for the models studied, approximate calculations were not required.

Figure 4 shows the Betweeness Centrality of the studied models. The x-axis displays the betweeness value, normalized between zero and one. The y-axis shows the number of LPs with that value of betweeness. A low betweeness value indicates that the node is not central to the network. Coupling this with LP connectivity, a network with a high degrees and low centrality indicates that the node is well connected. Conversely, a high value indicates the network is highly central to the network, meaning that a network with a low number of degrees and high centrality is not well connected. The y-axis of the figures show the frequency, or number of LPs.

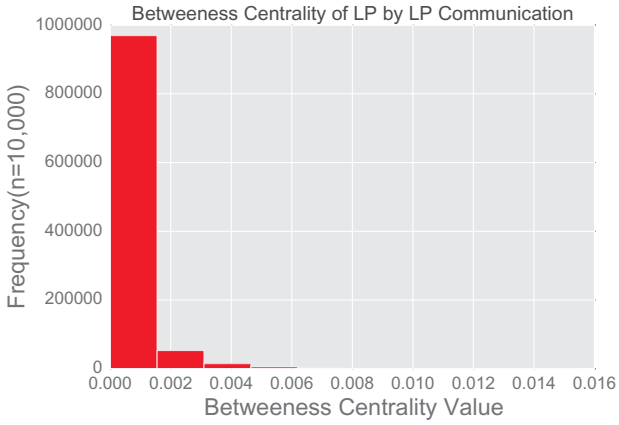
From the data, we see that nearly all of the models exhibit fairly low values for Betweeness Centrality. This is good as it documents that those models will not have many LPs that are highly connected across the entirety of the simulation and that effective partitioning should, in general, be successful in distributing a simulation workload across a collection of parallel processing nodes. While the worst case Betweeness Centrality of the highlighted models is .25 in the WARPED2 Epidemic Model, two of the overall models (IMC V4 444 and NS-3 CSMA) have some small number of LPs with betweenness values of 0.8-1.0. These values indicate that these models may have structure that would make them difficult to parallelize.



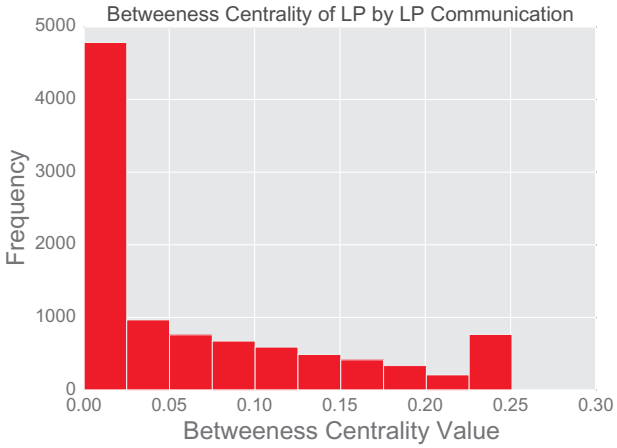
(a) warped2 Traffic Model



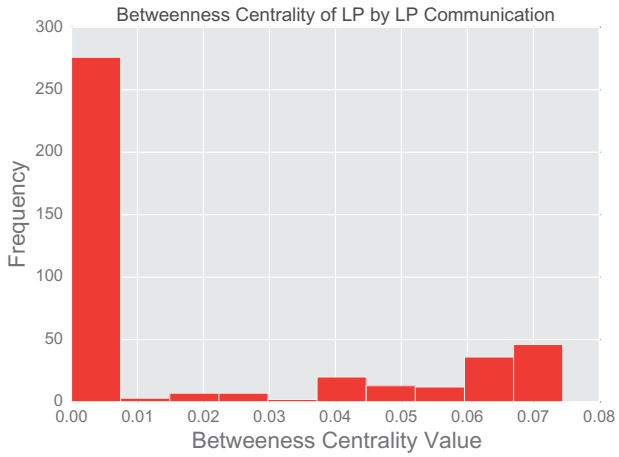
(b) ROSS 9D Torus Model



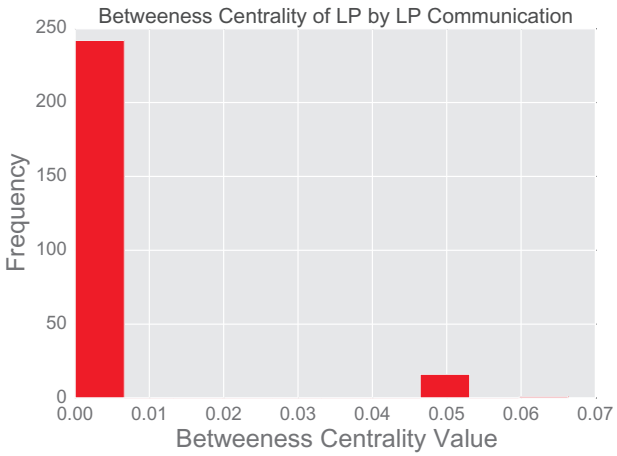
(c) ROSS PCS Model



(d) warped2 Epidemic Model



(e) Simian 161616, Hopper Network



(f) ns-3 Lena Dual Stripe

Figure 4: Betweeness Centrality.

5.3 Modularity

Modularity describes the structure networks, and measures how well sub-networks (*communities*) a large network is decomposed into. Communities are sets of highly inter-connected nodes (Blondel et al. 2008). A network with a high modularity have dense connections within its communities, but sparse connections among different communities. In other words, LPs that belong to the same community are sending and receiving a large portion of their events with LPs in the same community. Recognizing tightly coupled LPs and grouping them into communities may be beneficial for partitioning and group/block event scheduling policies (Gupta and Wilsey 2017). Models with few communities would be challenging to partition and group/block events for scheduling.

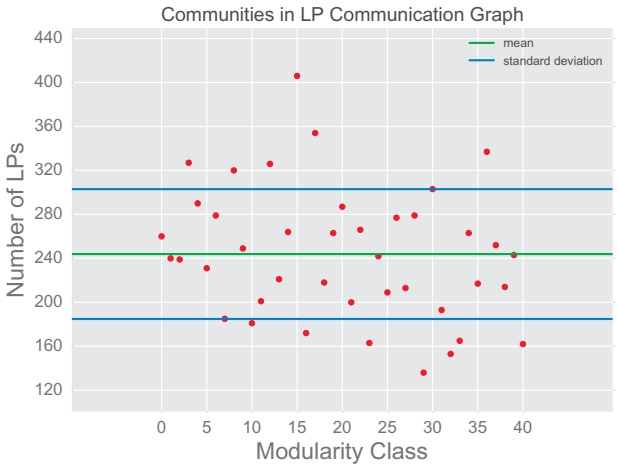
Figure 5 shows these communities. The x-axis are the different communities, Modularity class. The range of values on this axis are not significant other than they enumerate the different communities. The y-axis are the number of LPs belonging to each community. The green and blue lines represent, respectively the mean and standard deviation of the number of LPs about the mean.

The data from these plots are generally very encouraging for parallel execution. The WARPED2 Traffic, ROSS 9D Torus, IMC Hopper 161616, and WARPED2 Epidemic models contain rich sets of communities all with a fairly large number of LP populations in each. These number of communities and their relatively large populations of LPs (in the hundreds each) should partition well. Traffic and Epidemic should easily support parallelism to 10s of processing nodes. Other models with similarly available communities with notable populations of LPs include: ROSS Neuromorphic, WARPED2 PCS, IMC Edison, IMC V4 81616, IMC Bypass, IMC Moonlight, and (although it is a small model) NS-3 CSMA. The ROSS PCS Model is interesting in that it has a rich set of communities each with a large number of LPs. However, it also contains a number of communities (approximately 125) with only a few LPs. Fortunately, given the large number of sizable communities (approximately 250), these small communities should not negatively impact partitioning and lead to successful parallelization. However, from a group/block scheduling perspective (Gupta and Wilsey 2017), these small communities may present challenges.

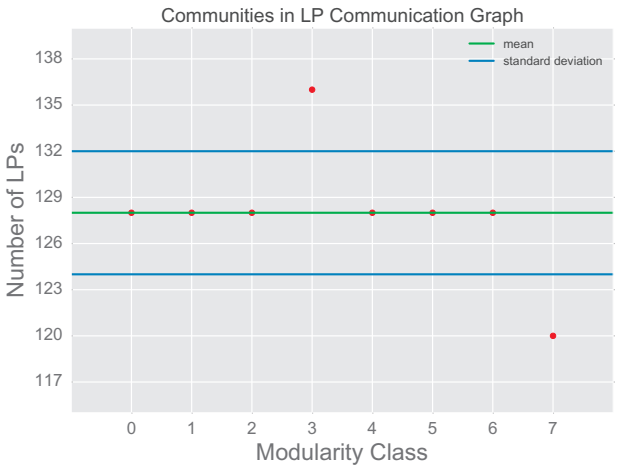
Finally, the NS-3 Lena Dual Stripe Model contain only two communities. While this is indicative of poorer partitioning possibilities, this model is actually quite small and they would not be good candidates for parallelism anyway. So the challenges are less concerning in these cases. Overall, the overall Modularity results are encouraging for parallelism. All of the sizable simulation models exhibit a good degree of Modularity that should provide ample opportunities for parallelism and parallel speedup.

6 CONCLUSIONS

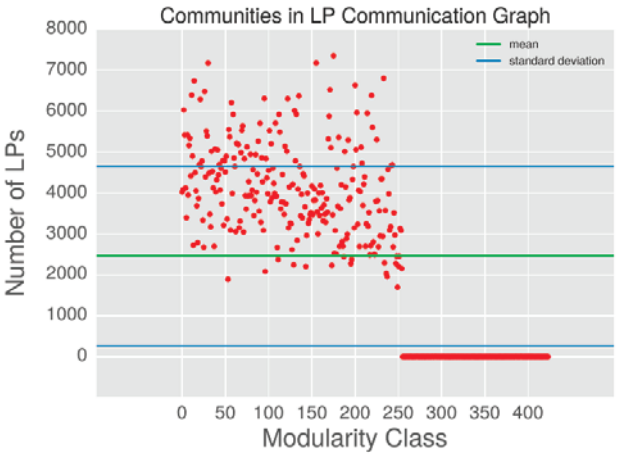
This paper explores the event communication properties of the LPs in 22 discrete-event simulation models captured from simulation runs on one of 4 different simulators. The independently developed models come from a variety of domains and were not developed for this study. The DES model properties are computed using profile data that characterizes the events generated and processed in these simulation models. The analysis results reported in this paper are: (i) the *in-/out-degree connectivity* of the LPs, (ii) the *Betweenness Centrality* of the LPs, and (iii) the *Modularity* of the LPs. Results from these analyses show that, in general, LPs tend to communicate with only a few other LPs, that the LPs are not tightly coupled and the models should therefore be good candidates for parallel execution. Finally, the Modularity studies show that most of the models also have numerous and sizable sub-communities of more frequently communicating LPs that should form good partitions for distribution to a parallel processing platform (of limited size). That said, the IMC V4 444 and NS-3 CSMA models both contained some LPs with high values to their Betweenness Centrality which indicates that they may present challenges for parallel execution. Overall, however, the data shows that the simulation models tend to fall into categories with shared properties that should provide indicators for model types and behaviors that tool builders can target for optimization and improved analysis support.



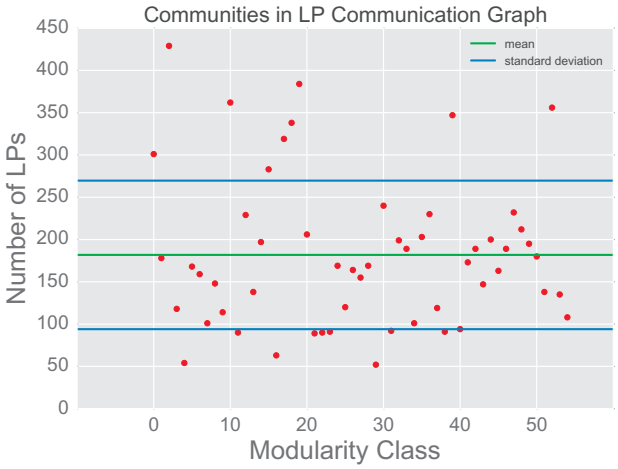
(a) warped2 Traffic Model



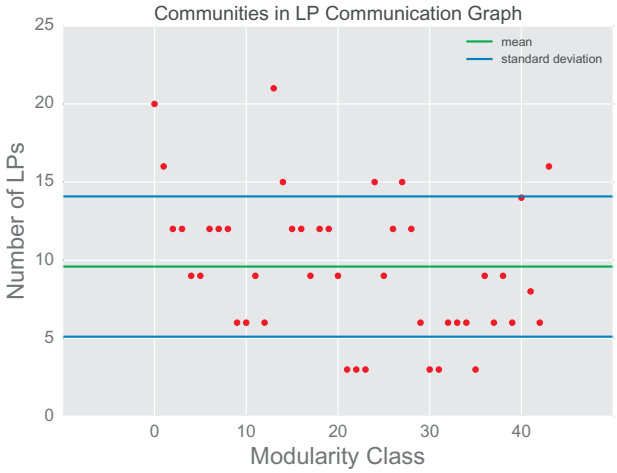
(b) ROSS 9D Torus Model



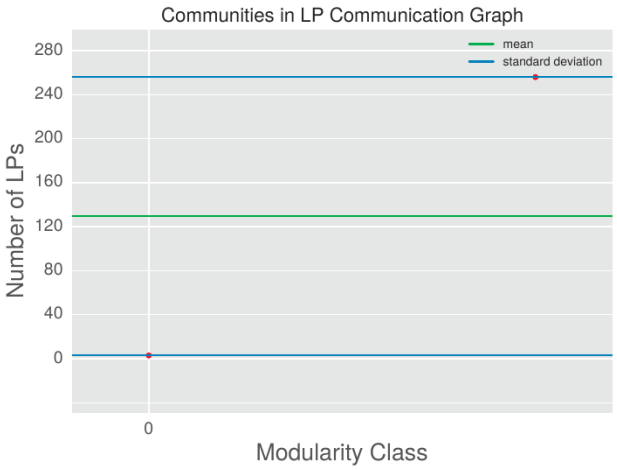
(c) ROSS PCS Model



(d) warped2 Epidemic Model



(e) Simian 161616, Hopper Network



(f) ns-3 Lena Dual Stripe

Figure 5: Subnetwork Communities of tightly coupled LPs.

ACKNOWLEDGMENTS

This material is based upon work supported by the AFOSR under award No FA9550–15–1–0384.

REFERENCES

Ahmed, K., M. Obaida, J. Liu, S. Eidenbenz, N. Santhi, and G. Chapuis. 2016. "An Integrated Interconnection Network Model for Large-Scale Performance Prediction". In *Proceedings of the 2016 Annual ACM Conference on SIGSIM Principles of Advanced Discrete Simulation, SIGSIM-PADS*, 177–187. New York, NY, USA: ACM.

Alt, A. J., and P. A. Wilsey. 2014, December. "Profile Driven Partitioning of Parallel Simulation Models". In *Proceedings of the 2014 Winter Simulation Conference*, edited by A. Tolk, S. Y. Diallo, I. O. Ryzhov, L. Yilmaz, S. Buckley, and J. A. Miller, 2750–2761. Piscataway, New Jersey, USA: Institute of Electrical and Electronics Engineers, Inc.

Anthonisse, J. M. 1971, October. "The Rush in a Directed Graph". Technical report, Stichting Mathematisch Centrum, Amsterdam.

Balakrishnan, V., R. Radhakrishnan, D. M. Rao, N. B. Abu-Ghazaleh, and P. A. Wilsey. 2001. "A Performance and Scalability Analysis Framework for Parallel Discrete Event Simulators". *Simulation Practice and Theory* 8:529–553.

Barrett, C. L., K. R. Bisset, S. G. Eubank, X. Feng, and M. V. Marathe. 2008. "EpiSimdemics: An Efficient Algorithm for Simulating the Spread of Infectious Disease over Large Realistic Social Networks". In *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, SC '08. Piscataway, New Jersey, USA: IEEE Press.

Berry, O., and D. Jefferson. 1985. "Critical Path Analysis of Distributed Simulation". In *Distributed Simulation*, 57–60. San Diego, CA: SCS: SCS.

Blondel, V. D., J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. 2008. "Fast Unfolding of Communities in Large Networks". *Journal of Statistical Mechanics: Theory and Experiment* 10:10008.

Brandes, U. 2001. "A Faster Algorithm for Betweenness Centrality". *Journal of Mathematical Sociology* 25:163–177.

Brandes, U., and C. Pich. 2007. "Centrality Estimation in Large Networks". *International Journal of Bifurcation and Chaos* 17 (07): 2303–2318.

Carothers, C. D., D. Bauer, and S. Pearce. 2000. "ROSS: A High-performance, Low Memory, Modular Time Warp System". In *Proceedings of the Fourteenth Workshop on Parallel and Distributed Simulation*, PADS '00, 53–60. Washington, DC, USA: IEEE Computer Society.

Carothers, C. D., R. M. Fujimoto, Y.-B. Lin, and P. England. 1994, January. "Distributed Simulation of Large-Scale PCS Networks". In *Proceedings of the Second International Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS '94)*, 2–6. New York, NY, USA: ACM.

Ferscha, A., and J. Johnson. 1996, December. "A Testbed for Parallel Simulation Performance Predictions". In *1996 Winter Simulation Conference Proceedings*, edited by J. M. C. andd D. J. Morrice, D. T. Brunner, and J. J. Swain, 637–644. Piscataway, New Jersey, USA: Institute of Electrical and Electronics Engineers, Inc.

Freeman, L. C. 1977. "A Set of Measures of Centrality Based on Betweenness". *Sociometry* 40 (1): 35–41.

Fujimoto, R. 1990, January. "Performance of Time Warp under Synthetic Workloads". In *Proceedings of the SCS Multiconference on Distributed Simulation*, edited by D. Nicol, Volume 22, 23–28. San Diego, CA: SCS.

Fujimoto, R. M. 1989, April. "Performance Measurements of Distributed Simulation Strategies". *Transactions of the Society for Computer Simulation* 6 (2): 89–132.

Gupta, S., and P. A. Wilsey. 2017. "Quantitative Driven Optimization of a Time Warp Kernel". In *Proceedings of the 2017 ACM SIGSIM Conference on Principles of Advanced Discrete Simulation*, PADS 17. New York, NY, USA: ACM.

Henderson, T. R., M. Lacage, G. F. Riley, C. Dowell, and J. Kopena. 2008. "Network Simulations with the ns-3 Simulator". *SIGCOMM demonstration* 14:527.

Hennessy, J. L., and D. A. Patterson. 2012. *Computer Architecture: A Quantitative Approach*. 5th ed. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Jefferson, D., and P. L. Reiher. 1991, April. "Supercritical Speedup". In *Proceedings of the 24th Annual Simulation Symposium*, edited by A. H. Rutan, 159–168. IEEE Computer Society Press.

Lin, Y.-B. 1992, July. "Parallelism Analyzer for Parallel Discrete Event Simulation". *ACM Transactions on Modeling and Computer Simulation* 2 (3): 239–264.

Livny, M. 1985, January. "A Study of Parallelism in Distributed Simulation". In *Proceedings 1985 SCS Multiconference on Distributed Simulation*, 94–98. San Diego, CA: SCS.

Park, E. J., S. Eidenbenz, N. Santhi, G. Chapuis, and B. Settlemyer. 2015. "Parameterized Benchmarking of Parallel Discrete Event Simulation Systems: Communication, Computation, and Memory". In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, 2836–2847. Piscataway, New Jersey, USA: Institute of Electrical and Electronics Engineers, Inc.

Riley, G. F., and T. R. Henderson. 2010. *The ns-3 Network Simulator*, 15–34. Springer.

Santhi, N., S. Eidenbenz, and J. Liu. 2015. "The Simian Concept: Parallel Discrete Event Simulation with Interpreted Languages and Just-In-Time Compilation". In *Proceedings of the 2015 Winter Simulation Conference*, edited by L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, 3013–3024. Piscataway, New Jersey, USA: Institute of Electrical and Electronics Engineers, Inc.

Weber, D. 2016. "Time Warp Simulation on Multi-core Processors and Clusters". Master's thesis, University of Cincinnati, Cincinnati, OH.

Wilsey, P. A. 2016. "Some Properties of Events Executed in Discrete-Event Simulation Models". In *Workshop on Parallel and Distributed Simulation*, PADS 16. New York, NY, USA: ACM.

AUTHOR BIOGRAPHIES

PATRICK CRAWFORD is a graduate student in the Department of Electrical Engineering and Computer Science at the University of Cincinnati. His email address is crawfopw@mail.uc.edu.

STEPHAN EIDENBENZ is the Director of the Information Science and Technology (ISTI) institute at Los Alamos National Laboratory. He has a PhD from the Swiss Federal Institute of Technology, Zurich (ETHZ) in Computer Science. His research interests include performance prediction, cyber security, computational co-design, communication networks, scalable modeling and simulation, and theoretical computer science. His email address is eidenben@lanl.gov.

PETER D. BARNES JR. leads the Network Simulation team at Lawrence Livermore National Laboratory. He has a PhD in Physics from the University of California, Berkeley. His current research interests include the scaling of simulation models to very large sizes to be executed in parallel on extreme scale parallel processing hardware. His email address is barnes26@llnl.gov.

PHILIP A. WILSEY is a professor in the Department of Electrical Engineering and Computer Science at the University of Cincinnati. His research interests are in high performance computing, parallel and distributed systems, parallel simulation, high-performance clustering of hith-dimensional data, embedded system, and point-of-care medical devices. His email address is wilseypa@gmail.com.