UCRL-SR-216012

LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

Final Report for #Queuing Network Models of Performance of High End Computing Systems#

Jeff Buckwalter

October 10, 2005

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

University of San Francisco
2130 Fulton Street
San Francisco, CA 94117-1080

Final Report for "Queuing Network Models of Performance of High End Computing Systems"

Submitted by:
Jeff Buckwalter
Principal Investigator

FINAL REPORT
For the period ending September 30, 2005

Prepared for:
University of California
Lawrence Livermore National Laboratory
Attn: Linda Becker
P.O. Box 808, L-561
Livermore, CA 94551

Under
B546340

September 30, 2005

Final Report on

LLNL Subcontract B546340
"Queuing Network Models of Performance of High End Computing Systems"

Jeff Buckwalter

1. Executive Summary

The primary objective of this project is to perform general research into queuing network models of performance of high end computing systems. A related objective is to investigate and predict how an increase in the number of nodes of a supercomputer will decrease the running time of a user's software package, which is often referred to as the strong scaling problem.

We investigate the large, MPI-based Linux cluster MCR at LLNL, running the well-known NAS Parallel Benchmark (NPB) applications. Data is collected directly from NPB and also from the low-overhead LLNL profiling tool mpiP.

For a run, we break the wall clock execution time of the benchmark into four components: switch delay, MPI contention time, MPI service time, and non-MPI computation time. Switch delay is estimated from message statistics. MPI service time and non-MPI computation time are calculated directly from measurement data. MPI contention is estimated by means of a queuing network model (QNM), based in part on MPI service time.

This model of execution time validates reasonably well against the measured execution time, usually within 10%. Since the number of nodes used to run the application is a major input to the model, we can use the model to predict application execution times for various numbers of nodes.

We also investigate how the four components of execution time scale individually as the number of nodes increases. Switch delay and MPI service time scale regularly. MPI contention is estimated by the QNM submodel and also has a fairly regular pattern. However, non-MPI compute time has a somewhat irregular pattern, possibly due to caching effects in the memory hierarchy.

In contrast to some other performance modeling methods, this method is relatively fast to set up, fast to calculate, simple for data collection, and yet accurate enough to be quite useful.

2. Project Motivation

An important question that occurs with high end computing systems is: "Given that a particular application program uses P processors, how long will it take to execute the program?" When the execution time is considered as a function of P, and the input data to the program remains the same, the question is often referred to as the strong scaling problem. We build models and methods that are an important step toward answering this performance question.

Thus a naturally related question, that is also a motivation for our research, is how to build models of application programs running on high end computing systems, where the models are useful for analyzing and predicting the behavior of the system. Further, can we extend these modeling methods to systems with heterogeneous processors, especially processors with different clock speeds.

Our research, performed under this subcontract, provides useful tools and methods for approaching these motivating questions.

3. Technical Approach

Since the overall questions we are trying to answer are quite broad, we began our approach to modeling high end computing systems in the following way.

1. We use the Linux clusters at LLNL for our high end computing systems. These clusters have large numbers of processors, which is good for studying the strong scaling problem. They are also representative of many high end computing systems in the world today. The clusters are fairly well-understood, well-documented, and some of their performance characteristics have been studied by other researchers.
2. We use the NAS Parallel Benchmarks (NPB) as representative application programs. This compute-intensive program suite exercises a parallel computer in a number of ways, and is often used as a benchmark for performance studies. Furthermore, the program is free and the source code is available, which makes it much easier to study and modify the behavior of the programs.
3. We use the performance measurement tool, mpiP, available from LLNL, for collecting and performing partial analysis of our runs. The mpiP tool has low overhead, collects useful information, is relatively easy to install, and is well-known at LLNL.
4. To collect measurement data for building models, we made two major rounds of runs. The runs spanned the four major classes (sizes) of the NAS Parallel Benchmarks, different numbers of processors, different methods of allocating processors to the benchmarks, and different repetitions in order to determine variance from one run to the next.
5. Having collected initial sets of data, we modeled the wall clock execution time of a particular application run by breaking the execution time into four components:
    - Switch delay: amount of time it takes the network switch to process (transfer) all the messages.
    - MPI contention: amount of time MPI waits for servicing of all messages.
    - MPI active time: amount of time MPI actively services messages.

- Compute time for the application: amount of time spent on non-overlapped computation, not related to MPI.

6. The switch delay models the network interconnect and is determined by dividing average message size (L) by bandwidth (BW) and adding network latency (Lat). It represents the average network transfer time for a message. The switch delay per message is then multiplied by the number of messages per processor to model the switch delay component of wall clock time. Note that the parameters for modeling this component of wall clock time are easily obtainable from the measurement tool mpiP and from known (or measurable) characteristics of the cluster.
7. The MPI contention time is calculated from a queuing network model (QNM, Figure 1 below) that estimates the average amount of contention among MPI messages that are attempting to obtain service from a processor. The primary input for the QNM solver is the MPI active time. The number of processors, switch delay, and compute time are also inputs to the queuing network model.
8. The MPI active time is the amount of time spent by a processor when servicing MPI calls It excludes time waiting for MPI transmissions, and also excludes processor computation that occurs between MPI calls. MPI active time is calculated as the total MPI time reported by mpiP minus the MPI contention described above.
9. The compute time for the application is the amount of non-overlapped computation that occurs between MPI calls. It is calculated as the difference between application execution time (as reported by mpiP), and total MPI time (also reported by mpiP).
10. We then validate the wall clock model against a substantial number of measurement runs. This gives us confidence that the model is a reasonable estimator of the actual application run time, over a range of processors (1 to 1024), applications (CG and FT from the NAS Parallel Benchmarks), processor allocations (block, cyclic, and linear), and machines (MCR and the Keck cluster at University of San Francisco).
11. In addition to building and validating the wall clock model, we also investigate scaling behavior of the model. In particular, we investigate how the four components of wall clock time vary as the number of processors increases, for a given application running on a given system.

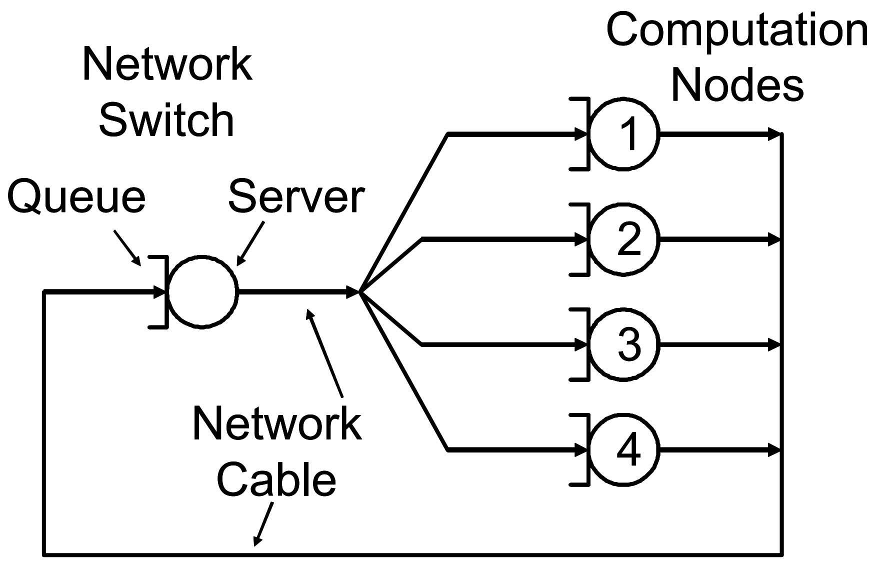


Figure 1. Diagram of Queuing Network Model

4. Research and Results

4.1 Activities

The major activities for the project are summarized below. The work was performed by the Principal Investigator, Jeff Buckwalter, and the Research Assistant, Michael Elliott.

1. Installed and used NAS PB (Numerical Aerodynamic Simulation Parallel Benchmarks) from NASA Ames as the initial application program we are modeling.
2. Installed and used the communications profiling tool mpiP, from LLNL, for collecting information about MPI communications performance.
3. Developed our own queuing network model solver, in addition to initially making use of the SHARPE model solving tool, from Kishor Trevedi of Duke University.
4. Established contact with the Performance Modeling and Characterization laboratory at the San Diego Supercomputer Center, about the use of their tools, such as MetaSim Tracer and MetaSim Convolver, for obtaining memory operation performance.
5. Collected data from about 450 runs of the NAS PB software on MCR, as instrumented with mpiP and the Linux time tool. These runs have included
    a) Full benchmark suites of Class A, B, C, and D size problems, on various numbers of processors, such as 1, 4, 16, 64, 256, and 1024.
    b) CG (Conjugate Gradient) benchmarks for Class A, B, C, and D size problems, on various numbers of processors, such as 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024.
    c) Various NAS PB scenarios, such as block processor allocation, cyclic processor allocation, and linear (one processor per node).
    d) Repetitions of runs to determine the "natural" variability between runs.
6. Wrote a Java program, mpiPfilter, that filters the output files from mpiP and NAS PB, and creates a text file usable as input for the NBP Spreadsheet.
7. Built the NPB spreadsheet, which receives, as input, select values from the NAS suite and mpiP files, as generated by mpiPfilter. The NPB spreadsheet then uses these values to calculate model inputs for the QNM Solver. The spreadsheet also performs error analysis between modeled and measured values, and graphically displays the results. The graphs also compare the components of the model's wall clock time to the measured components of the wall clock time for the application.
8. Wrote the QNM (Queuing Network Model) Solver, which is a Java program, ported from algorithms and FORTRAN code in [29]. Using input files generated by our utility inMaker, based on values from the NPB spreadsheet, as gathered from the mpiP and NAS PB suite files, the program models the system as a queuing network. The solver performs single and multiple class mean value analysis, single class load dependent service center modeling, and is capable of batch execution The output of the solver is entered into the NPB spreadsheet to complete the modeled vs. measured validations
9. Built performance models for most of the 450 runs described above, using the tools described above.
10. Investigated scaling behavior of the model as the number of processors increases.

4.2 Related Activities at Other Institutions

The Research Assistant, Michael Elliott, applied these same modeling tools and techniques to the University of San Francisco Department of Computer Science's Keck Cluster supercomputer, which is a 64 node Beowulf cluster of dual Pentium III 1GHz CPUs, connected by a 2Gbps Myrinet network. This work was part of a directed research course at USF, and was not directly funded by LLNL, although it used tools and techniques developed under the LLNL subcontract. The validation between modeled and measured results, which were similar to those of LLNL's MCR, helped to increase confidence in the ability of these techniques to predict wall clock execution times reasonably accurately.

4.3 Conferences Attended

Both the PI and the RA attended HPCA-11 (Eleventh International Symposium on High-Performance Computer Architecture) in San Francisco, February 12-16, 2005. This was valuable for gaining background and for learning the state of the art in computer architectures for the type of systems we are modeling.

The PI, Prof. Buckwalter, attended IPDPS (International Parallel and Distributed Processing Symposium) in Denver, April 4-8, 2005. However, this conference was funded by USF.

The PI also attended HPDC-14 (High Performance Distributed Computing) in Raleigh-Durham, July 24-27. Again, this was a valuable conference for gaining background and making contacts with other researchers.

4.4 Results

In general, there is good correlation between measured and modeled values for application execution time, or wall clock time. Almost all relative error values fall within the 30% tolerance typical of QNM models, and most fall within 10% or less. Relative errors tend to be less for smaller numbers of processors, and greater for larger numbers of processors, where there is more contention among MPI messages.

These validations hold up well across the different problem sizes (Class A through Class D). However, the smallest size, Class A, which has application execution times of only a few seconds, tends to show the greatest variability from run to run. We hypothesize that operating system noise and network contention from processors not in the run are having a larger impact on these short Class A runs than on the longer runs, such as for Classes B, C, and D.

We give two typical examples of graphs generated by the NPB spreadsheet. Both examples are from runs of NPB CG (Conjugate Gradient) Class B on MCR. Note that the modeled application execution time validates reasonably well with the measured times. In the first chart, the error bars indicate the 30% relative error range. In the second chart's legend, the upper three time components refer to the measured aggregate execution time and are represented by the outer column for each set of processors. The lower four time components refer to the modeled aggregate execution time and are represented by the inner column. (The electronic version of this document is in color, which is easier to read.)

Figure 2. Modeled vs. Measured Application Wall Clock Times

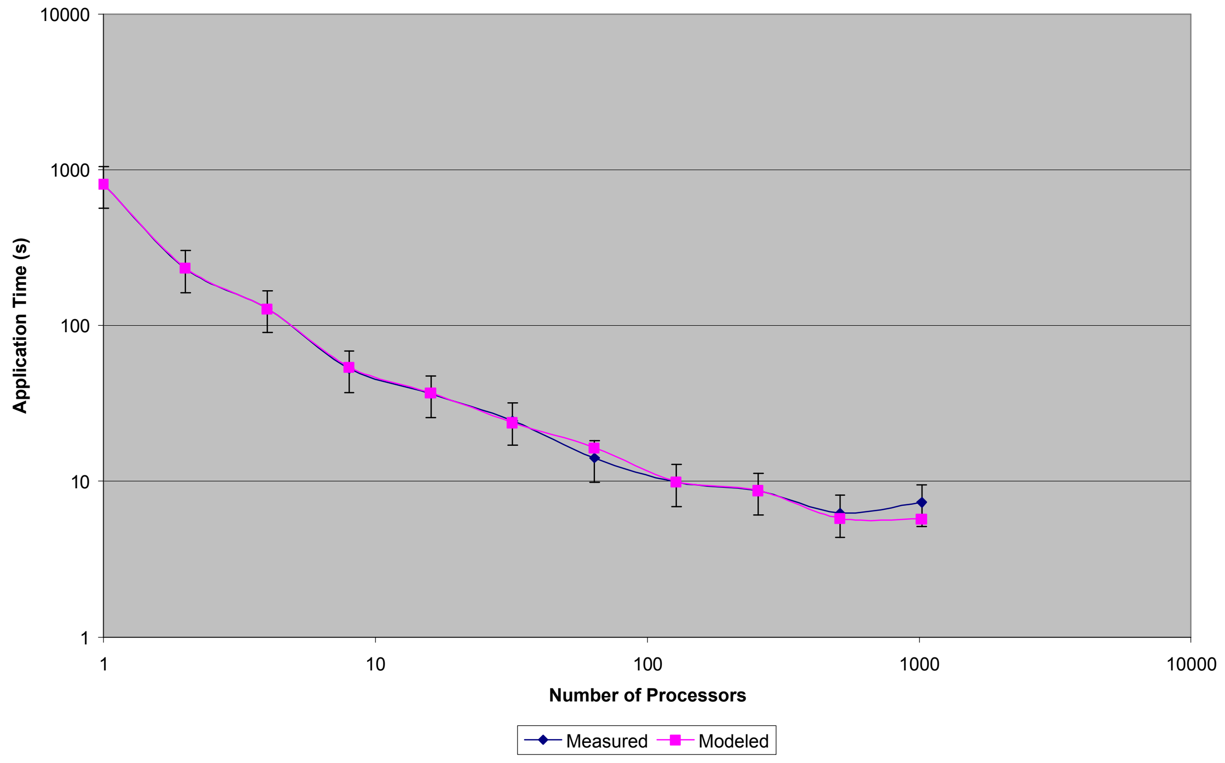
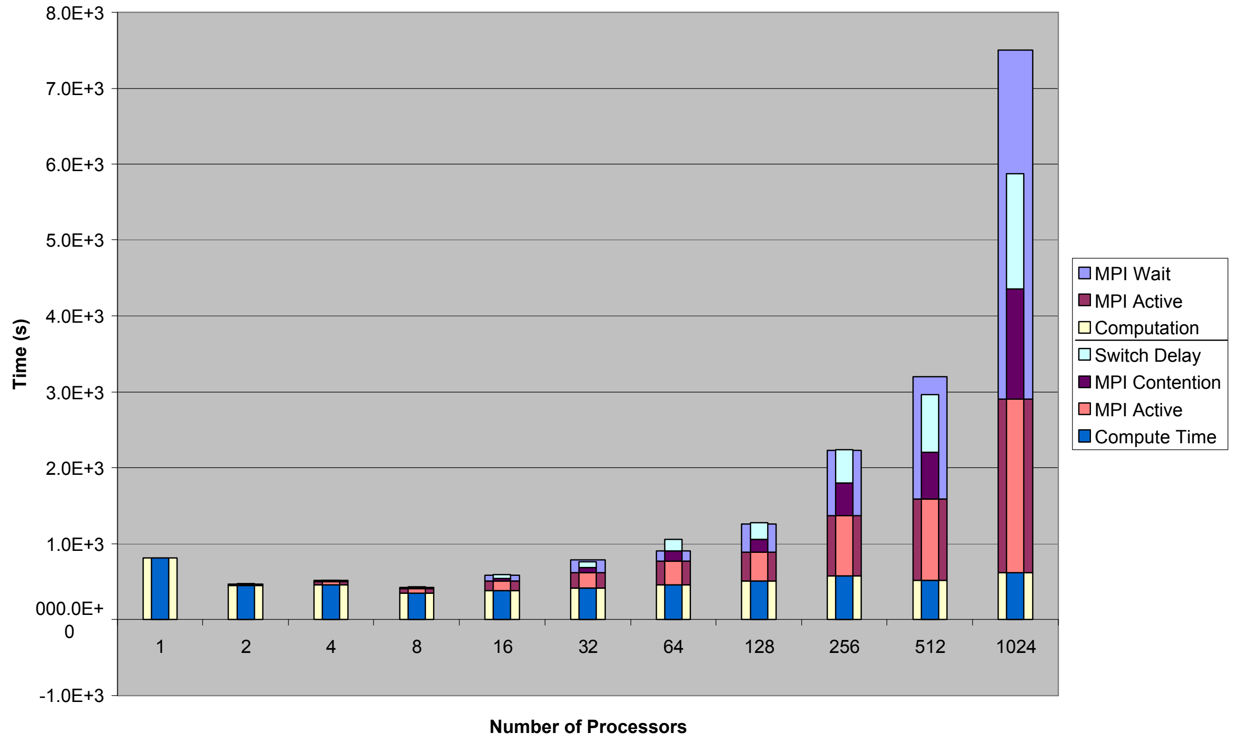


Figure 3. Modeled vs. Measured Aggregate Component Times



Our initial investigation of scaling behavior of the NPB CG benchmark on MCR indicates that most of the major inputs to the model are reasonably well behaved. In particular, as the number of processors increases,

- Average message size decreases approximately as the inverse square.
- The number of messages. increases near-linearly, with a small quadratic component.
- Aggregate MPI active time (MPI service time) also increases near-linearly, with a small quadratic component.
- However, aggregate compute time (non-MPI computation) behaves more erratically. It roughly follows an overall U-shaped curve, higher on the right, with small, irregular sawtooths. We hypothesize that caching at the different layers of memory hierarchy is having a strong effect as the working set decreases due to smaller message sizes.

Thus, the following prediction problem will likely have to wait until we have a better model of how compute time scales with the number of processors:

> *Predict how application A will behave on target machine T for large numbers of processors, given knowledge of how application A behaves on machine T for small numbers of processors and given extensive knowledge of how A behaves on baseline machine B.*

On the other hand, our results are quite promising for predicting how an application will behave on a baseline machine, given a small sample of runs at various numbers of processors.

We note that our technique will accommodate systems with heterogeneous processors (in terms of processor clock speeds), although we have not yet experimented with a heterogeneous system at LLNL.

In summary, this queuing network modeling technique shows good potential for performance prediction of high end computing systems. It gives much better accuracy than is often seen in other estimation methods, is easy to calculate, and requires relatively little data collection.

5. Papers and Book Chapters Supported in Part by the Subcontract

Software tools and research techniques developed under the subcontract were used in the technical report "Using Queueing Network Modeling to Analyze the University of San Francisco Keck Cluster Supercomputer," by Michael Elliott, University of San Francisco, August 19, 2005.

We are also working to publish these results as at least one conference proceeding, although we have not yet selected a target conference.

6. Bibliography

Portions of this report were adapted from a technical report written by the RA [25}.

[1] Alexandrov, Albert, et al. "LogGP: Incorporating Long Messages into the LogP Model – One step closer towards a realistic model for parallel computation." Proceedings of the Seventh Annual ACM Symposium on Parallel Algorithms and Architectures. 24 – 26 Jun. 1995. Santa Barbara: U. of California at Santa Barbara, 1995. 95 – 105.

[2] Bailey, D., et al. "The NAS Parallel Benchmarks." RNR Technical Report. NASA Ames Research Center, Mar. 1994.

[3] Bailey, David H., et al. "Performance Technologies for Peta-Scale Systems: A White Paper Prepared by the Performance Evaluation Research Center and Collaborators." White paper. Lawrence Berkeley National Laboratories, 2003.

[4] Balsa, André. Linux Benchmarking – Concepts. "3. FPU tests: Whetstone and Sons, Ltd." 21 Sept. 1997. The Linux Gazette. 26 Nov. 2004. <http://www.tux.org/~balsa/linux/benchmarking/articles/html/Article1d-3.html>.

[5] Barney, Blaise M. "Introduction to Parallel Computing." 21 Jun. 2004. Lawrence Livermore National Laboratory. 28 Feb. 2005. <http://www.llnl.gov/computing/tutorials/parallel_comp/>.

[6] Barney, Blaise M. "Message Passing Interface (MPI)." 21 Dec. 2004. Lawrence Livermore National Laboratory. 28 Feb. 2005. <http://www.llnl.gov/computing/tutorials/mpi/>.

[7] Barney, Blaise M. "MPI Performance Topics." 24 May 2004. Lawrence Livermore National Laboratory. 28 Feb. 2005. <http://www.llnl.gov/computing/tutorials/mpi_performance/>.

[8] Barney, Blaise M. "Performance Analysis Tools." 15 Jul 2004. Lawrence Livermore National Laboratory. 30 Mar. 2005. <http://www.llnl.gov/computing/tutorials/performance_tools/>.

[9] Bramer, Brian. System Benchmarks. DeMontfort University, UK. 26 Nov. 2004. <http://www.cse.dmu.ac.uk/~bb/Teaching/ComputerSystems/SystemBenchmarks/BenchMarks.html>.

[10] Calzarossa, Maria, Louisa Massari, and Daniele Tessera. "Workload Characterization Issues and Methodologies." Performance Evaluation, LNCS. Ed. G. Haring et al. Berlin: Springer-Verlag, 2000. 459 – 482. 9 Jul. 2004. <http://www.cs.huji.ac.il/course/2003/perf/calza1.pdf>.

[11] Cameron, Kirk and Rong Ge. "Predicting and Evaluating Distributed Communication Performance." SC 04 Proceedings. 6 – 16 Nov. 2004. Pittsburgh: David L. Lawrence

Convention Center, 2004. 1 – 15.

[12] Cameron, Kirk and Vimi S. Carol. "High-Performance, Power-Aware Computing." Lecture. Lawrence Livermore National Laboratory Institute for Scientific Computing Research. Livermore, CA, May 9, 2005.

[13] Campbell, Roy L., Jr., and Larry P. Davis. "Influence of Workload Characterization on DoD High Performance Computing Modernization Program Acquisitions." Proceedings of the Second workshop on Productivity and Performance in High-End Computing. 13 Feb. 2005. San Francisco: Palace Hotel, Eleventh International Symposium on High Performance Computer Architecture, 2005. 7 – 10.

[14] Carrington, Laura, et al. "Applying an Automated Framework to Produce Accurate Blind Performance Predictions of Full-Scale HPC Applications." Proceedings: UGC 2004. 7 – 11 Jun. 2004. Williamsburg: HPCMP Office, Arlington, VA, 2004.

[15] Compaq Extended Math Library. "LAPACK." Compaq / Hewlett-Packard Inc. 9 Dec. 2004. <http://h18000.www1.hp.com/math/documentation/cxml/lapack.3dxml.html>.

[16] Culler, David, et al. "LogP: Towards a Realistic Model of Parallel Computation." Proceedings of the Fourth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. 19 – 22 May, 1993. San Diego: SIGPLAN: ACM, 1993. 1 – 12.

[17] Deelman, Ewa, et al. "POEMS: End-to-end Performance design of Large Parallel Adaptive Computational Systems." University of Texas. 9 Jul. 2004. <http://www.cs.utexas.edu/users/poems/Papers/Wosp/wosp_dave.html>.

[18] Denning, Peter J., and Jeffrey P. Buzen. "The Operational Analysis of Queueing Network Models." ACM Computing Surveys. Vol. 10.3. New York: ACM Press, Sep. 1978. 225 – 261.

[19] Dhrystone. "What is Dhrystone? – A Word Definition from the Webopedia Computer Dictionary." Webopedia.com. 26 Nov. 2004. <http://www.webopedia.com/TERM/D/Dhrystone.html>.

[20] Dickens, Phillip M., Philip Heidelberger, and David M. Nicol. "Parallelized Direct Execution Simulation of Message-Passing Parallel Programs." 10 Jun. 1994. NASA. 9 Jun. 2004. <http://techreports.larc.nasa.gov/icase/1994/icase-1994-50.pdf>.

[21] Dongarra, Jack. Netlib Repository at UTK and ORNL. "Frequently Asked Questions on the Linpack Benchmark and Top500." 28 Oct. 2004. AT&T Bell Laboratories, et al. 26 Nov. 2004. <http://www.netlib.org/utk/people/JackDongarra/faq-linpack.html#_Toc27885709>.

[22] Dongarra, Jack. Netlib Repository at UTK and ORNL. "LAPACK -- Linear Algebra PACKage." 28 Oct. 2004. AT&T Bell Laboratories, et al. 26 Nov. 2004. <http://www.netlib.org/lapack/>.

[23] Dongarra, Jack. Netlib Repository at UTK and ORNL. "Linpack." 28 Oct. 2004. AT&T Bell Laboratories, et al. 26 Nov. 2004. <http://www.netlib.org/linpack/>.

[24] Dongarra, Jack. "An Overview of High Performance Computing and Self-Adapting Numerical Software." Lecture. Lawrence Livermore National Laboratory ASC Institute for Terascale Simulation. Livermore, CA, May 17, 2005.

[25] Elliott, Michael. "Using Queueing Network Modeling to Analyze the University of San Francisco Keck Cluster Supercomputer." Report for CS 698-95 Directed Research - Performance Modeling. University of San Francisco, August 19, 2005.

[26] Frank, Matthew I., Anant Agarwal, and Mary K. Vernon. "LoPC: Modeling Contention in Parallel Algorithms." Proceedings of the Sixth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming. 18 – 21 Jun. 1997. Las Vegas: SIGPLAN: ACM, Las Vegas. 276 – 287.

[27] Fowler, Robert, et al. "Practical Performance Measurement and Analysis with HPCToolkit." Seminar. Lawrence Livermore National Laboratory Institute for Scientific Computing Research. Livermore, CA, May, 2005.

[28] Karatza, Helen D. "Current Trends in Modelling and Simulation of Parallel and Distributed Systems. [sic]" Editorial. I. J. of Simulation, Vol. 3.1-2. 9 July 2004. <http://ducati.doc.ntu.ac.uk/uksim/journal/Vol-3/No%201&2%20Special%20issue%20Karatza/Karatza/Karatza.pdf>.

[29] Lazowska, Edward D., et al. Quantitative System Performance: Computer System Analysis Using Queueing Network Models. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[30] Marin, Gabriel, and John Mellor-Crummey. "Cross-Architecture Performance Predictions for Scientific Applications Using Parameterized Models." SIGMETRICS/Performance '04. 12 – 16 June 2004. New York: Columbia University, 2004. 2 – 13.

[31] Mauer, Hans, et al. Top 500 Supercomputer Sites. 2004. Top 500 Supercomputer Sites. 26 Nov. 2004. <http://www.top500.org/lists/linpack.php>.

[32] Moore, Shirley, et al. "Improving Time to Solution with Automated Performance Analysis." Proceedings of the Second workshop on Productivity and Performance in High-End Computing. 13 Feb. 2005. San Francisco: Palace Hotel, Eleventh International Symposium on High Performance Computer Architecture, 2005. 20 – 26.

[33] Numerical Aerodynamic Simulation Power Benchmarks. Vers. 2.4. Computer software. NASA Ames Research Center, 1990. MPI. Source code.

[34] Nystrom, Nicholas A., John Urbanic and Christina Savinell. "Understanding Productivity Through Non-intrusive Instrumentation and Statistical Learning." Proceedings of the Second workshop on Productivity and Performance in High-End Computing. 13 Feb. 2005. San Francisco: Palace Hotel, Eleventh International Symposium on High

Performance Computer Architecture, 2005. 53 – 61.

[35] Snavely, A, et al. “Performance Modeling of HPC Applications.” Proceedings of ParCo 2003. 2 – 5 Sept. 2003. Dresden: Center for High Performance Computing, Dresden University of Technology, 2003.

[36] Snavely, Allan, et al. “A Framework for Performance Modeling and Prediction.” Proceedings of SC 2002. 16 – 22 Nov. 2002. Baltimore: Baltimore Convention Center, 2002.

[37] Snavely, Allan, Nicole Wolter, and Laura Carrington. “Modeling Application Performance by Convolving Machine Signatures with Application Profiles.” IEEE Workshop on Workload Characterization. 2 Dec. 2001. Austin: ACES Building, University of Texas, 2001.

[38] Squires, Susan, Walter F. Tichy, and Lawrence Votta. “What Do Programmers of Parallel Machines Need? A Survey.” Proceedings of the Second Workshop on Productivity and Performance in High-End Computing (PPHEC-05). 12 – 16 Feb. 2005. San Francisco: HPCA05, Palace Hotel, 2005.

[39] Taylor, Valerie E. “Analysis and Modeling of Parallel and Distributed Applications.” Seminar. Lawrence Livermore National Laboratory Institute for Scientific Computing Research. Livermore, CA, June 14, 2005.

[40] Tvrdik, Pavel. CS838: Topics in Parallel Computing. “Section \#2: PRAM Models.” 23 Jan. 1999. University of Wisconsin – Madison. 10 Dec. 2004. <http://www.cs.wisc.edu/~tvrdik/2/html/Section2.html>.

[41] Varga, András, Y. Ahmet Şekercioğlu, and Gregory K. Egan. “A Practical Efficiency Criterion for the Null Message Algorithm.” Oct 2003. European Simulation Symposium 2003. 9 July 2004. <http://ctieware.eng.monash.edu.au/twiki/pub/Simulation/ParallelSimulation/nmaperf.pdf>.

[42] Vetter, Jeffrey, and Chris Chambreau. “mpiP: Lightweight, Scalable MPI Profiling.” 25 Jan. 2005. Lawrence Livermore National Laboratories. 19 Jul. 2005. <http://www.llnl.gov/CASC/mpip/>.

[43] Zheng, Gengbin. “Achieving High Performance on Extremely Large Parallel Machines.” Diss. U. of Illinois at Urbana-Champaign, 2005. <http://charm.cs.uiuc.edu/papers/GengbinThesis.shtml>.