LA-UR- 02-4849

Approved for public release;
distribution is unlimited.

Title: USE OF PREDICTIVE PERFORMANCE MODELING DURING LARGE-SCALE SYSTEM INSTALLATION

Author(s): Darren J. Kerbyson Z# 176262, CCS-3
Adolfy Hoisie Z# 120864, CCS-3
Harvey J. Wasserman Z# 094600, CCS-3

Submitted to: LACSI 2002
Santa Fe, NM

Use of Predictive Performance Modeling during Large-Scale System Installation

Darren J. Kerbyson, Adolfy Hoisie, Harvey J. Wasserman

Los Alamos National Laboratory,
CCS-3 Modeling, Algorithms and Informatics Group,
Parallel Architectures and Performance Team
Los Alamos, NM 87545
Email: {djk,hoisie}@lanl.gov

Abstract. We concern ourselves in this paper with one important application of predictive performance modeling – to validate the measured performance during system installation. In general, models can provide performance and scalability expectations of a system for a given workload. The application characteristics of the ASCI workload utilized in this paper is SAGE, a multidimensional, 3D, multi-material hydrodynamics code with adaptive mesh refinement. We review the salient features of an analytical model of this code that can be applied to predict its performance on a large class of large-scale parallel systems. We then utilize the model to validate system performance on a Compaq Alpha-server ES45 Supercomputing system being built at Los Alamos, and expected to grow to 30T peak performance in the next few years. We describe the methodology applied during system installation and upgrades to establish a baseline for the achievable "real" performance of the system. We show that utilization of predictive performance models is also a powerful debugging tool.

1 Introduction

The peak performance of a system results from the underlying hardware architecture including processor design, memory hierarchy, inter-processor communication system, and their interaction. Moreover, the achievable performance is dependent upon the workload that the system is to be used for, and specifically how this workload utilizes the resources within the system. Thus increasing the peak performance of a system component is only valuable if it has an impact on the achievable performance of the workload. Performance analysis is required in order to ascertain the impact on performance resulting from architectural evolution and innovation.

Performance modeling is a key approach that can provide information on the expected performance of a workload given a certain architecture configuration. It is useful throughout a system's lifecycle: starting at design when no hardware is available for measurement, in procurement for the comparison of systems, through to implementation / installation, and to examine the effects of updating a system over time. At each point the performance model should provide an expectation of the achievable performance with a reasonable fidelity.

In this work we consider the installation of a Tera-scale Compaq Alpha system at Los Alamos. By using detailed knowledge of a major application, a performance model has been constructed and validated on a number of large-scale platforms [1]. This is one of the few large scale applications models, others include [2,3]. The model is used here to give an

expectation of the performance that should be obtained on the system being installed. It is shown that performance observations made on a newly installed system do not necessarily represent just the cost of processing the workload but often include idiosyncrasies of the hardware and system software such as bugs in the system software, faulty or poorly configured hardware components, and so on. Without an expectation of performance it is difficult to know when a reasonable level of performance is being obtained.

The experience gained from this installation has shown that performance modeling is a valuable tool. Several sets of measurements of the application performance were made on the system while it was being installed and debugged over a period of months. Only after a number of iterations of hardware refinements and software fixes did the performance of the system achieve the expected performance that was predicted.

An overview of the system and the application is given in Sections 2 and 3 respectively. In Section 4 the main characteristics of the application and of the system are combined into a parameterized model. Through a validation process on a number of large-scale systems, it is shown that this model is highly accurate. In Section 5 the model is used to provide an expectation of the performance of the system being installed, and compared to measured performance during installation.

2 The Compaq Alpha-Server ES45 Supercomputing System

The system considered here consists of 512 Compaq Alpha-Server ES45 nodes. This is the first part of the ASCI Q system – a 30Tflop machine being installed at Los Alamos. Each node contains 4 Alpha Ev68 processors running at 1GHz which are internally connected using two 2GB/s memory buses to 16GB of main memory. Each processor has an 8MB unified level 2 cache, and 64KB L1 data cache. The Alpha processor has a peak performance of 2 floating point operations per cycle. Thus this first subset of the Q-machine has a peak performance of 4Tflops/s.

Nodes are interconnected using the Quadrics QSnet high-performance network. This network boasts high-performance communication with a typical MPI latency of 5μs and a throughput of up to 340MB/s in one direction (detailed performance figures are discussed in Section 5). The Quadrics network contains two components – the Elan network interface card, and the Elite switch. The Elan/Elite components are used to construct a quaternary fat-tree topology (Figure 1). A quaternary fat-tree of dimension n is composed of 4^n processing nodes and $n.4^n$-1 switches interconnected as a delta network. Each Elite switch contains an internal 16x8 full crossbar. A detailed description of the Quadrics network can be found in [4].

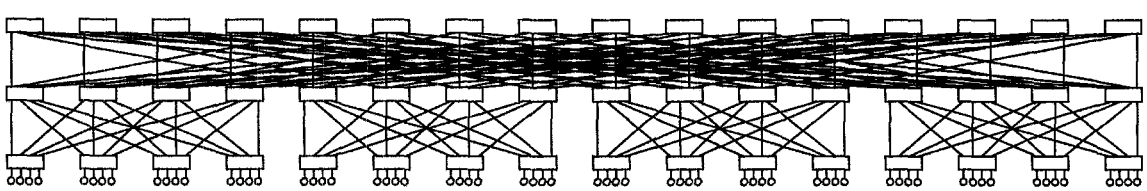

Figure 1. Network topology for a dimension 3 quaternary fat-tree network with 64 nodes

In order to implement a single rail (a single fat-tree network), a single Elan PCI interface card is used per node, in addition to a number of Elite switch boxes. The Elite switches are

packaged in 128-way boxes. The first level of boxes implements the first 3 levels of the fat-tree and consists of 64 down and 64 up ports. The second level of boxes implements the upper two levels of the fat-tree and consists of 128 down ports. Thus in the 512 node system, 12 switch boxes are utilized to provide a fat-tree of dimension 5 as illustrated in in Figure 2.

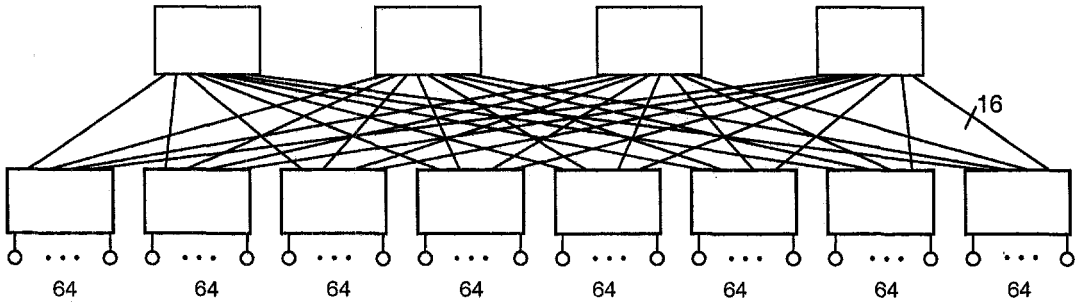

Figure 2. Interconnection of a federated Quadric network for a dimension 5 fat-tree

The system being installed at Los Alamos contains two rails of the Quadrics network, i.e. two parallel independent networks using two Elan cards per node, and two complete sets of Elite switches.

3 The Application

The application used to analyze the performance of the Compaq system is SAGE (SAIC's Adaptive Grid Eulerian hydrocode). It is a multidimensional (1D, 2D, and 3D), multimaterial, Eulerian hydrodynamics code with adaptive mesh refinement (AMR). It comes from the Los Alamos National Laboratory Crestone project, whose goal is the investigation of continuous adaptive Eulerian techniques to stockpile stewardship problems. SAGE has also been applied to a variety of problems in many areas of science and engineering including: water shock, energy coupling, cratering and ground shock, stemming and containment, early time front end design, explosively generated air blast, and hydrodynamic instability problems [5]. SAGE represents a large class of production ASCI applications at Los Alamos that routinely run on 1,000's of processors for months at a time.

SAGE is a large application consisting of 200,000+ lines of Fortran 90 code using MPI for inter-processor communications. Versions of SAGE exist for vector systems, cluster systems, and has been ported to all tera-scale ASCI architectures.

Adaptive mesh refinement operations are performed on cells as necessary at the end of each cycle in the processing. Each cell at the top most level (level 0) can be considered as root node of an oct-tree of cells in lower levels. For example, the shock-wave indicated in the 3-D spatial domain in Figure 3 by the solid line may cause cells associated with it (and close to it) to be split into smaller cells. In this example, a cell at level 0 is not refined, while a cell at level n is a domain 8^n times smaller.

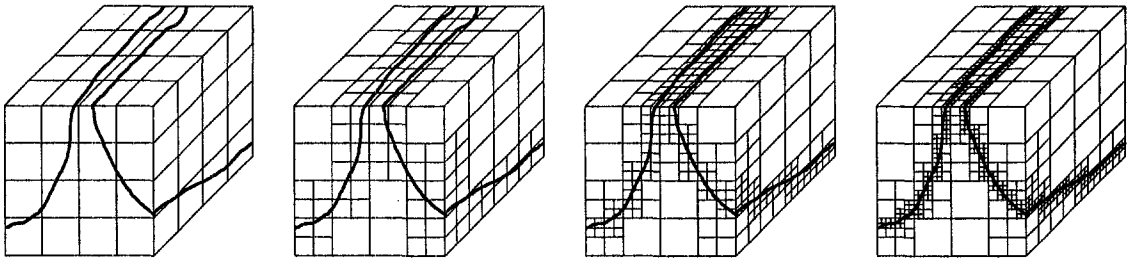

Figure 3. Example of Adaptive Mesh Refinement at multiple levels

In order to provide an expectation of the performance of an application, such as SAGE, the key processing and scaling characteristics need to be fully analyzed [6]. For SAGE, these are:

Data decomposition – SAGE uses a spatial discretization of the physical domain utilizing Cartesian grids. This spatial domain is partitioned across processors in sub-grids such that the first processor is assigned the first E cells in the grid (indexed in dimension order – X,Y,Z), and so on. This results in a 1-D *slab* partitioning of the spatial domain across processors. The problem size grows proportionally with the number of processors in the normal operational mode of SAGE, i.e. it has the characteristic of weak-scaling.

Processing flow – the processing proceeds in cycles. In each cycle there are a number of stages that involve the three operations of: one (or more) data gathers to obtain a copy of remote neighbor data, computation on each of the gathered cells, and one (or more) scatter operations to update data on remote processors.

AMR and load-balancing – at the end of each cycle, each cell can either be split into smaller 2x2x2 cells, combined with its neighbors to form a larger single cell, or remain unchanged. A load-balancing operation takes place if any processor contains 10% more cells than the average cells across all processors.

The 1-D slab decomposition affects the key gather/scatter communications as the number of processors used increases. This leads to two important factors influencing the achievable performance. Firstly, the amount of communication increases as the number of processors increases. This is due to the boundary surface of a sub-domain in a 1-D decomposition scaling at the 2/3 power of the number of processors. Secondly, since the number of cells per processor is constant, there is a point at which the sub-domain will be only a single cell wide, and also when a single slab is mapped to more than one processor. At this point the gather/scatter communications will not be just between adjacent processors but rather between processors a certain distance apart. This distance is actually equal to the number of processors sharing a single slab. This distance will be reflected in the number of simultaneous out-of-node communications that will contend for the communication channel. The maximum number of processes contending will be equal to the number of processors within a node, i.e. 4 for the Compaq ES45. Full details on the characteristic scaling behavior of SAGE are given in [1].

4 An Analytical Model for the Performance of the application

The performance model of SAGE consists of three main components: computation, memory contention within a node, and inter-node communication. The model is expressed to separate out the software details, the mapping details, and the actual resource costs. This is a similar approach to that taken by the PACE analytical performance prediction tool [7]. The run-time for one cycle of the code can be modeled as:

$$\begin{aligned} T_{cycle_i}(P,E,\mathbf{D},\mathbf{A},\mathbf{M_{cm}}) = \; & T_{comp}(E.D_i) + T_{memcon}(P,E.D_i) + \\ & T_{allreduce}(P) + T_{GScomm}(P,E,D_i) + \\ & T_{divide}(A_i) + T_{combine}(E.D_i) + T_{load}(M_{cm_i},P) \end{aligned} \quad (1)$$

where

$T_{comp}(E.D_i)$ is the computation time
$T_{GScomm}(P,E,D_i)$ is the gather and scatter communication time
$T_{allreduce}(P)$ is the allreduce communication time
$T_{memcon}(P,E.D_i)$ is memory contention that occurs between PEs within a node
$T_{divide}(A_i)$ is the time to divide cells in the current cycle
$T_{combine}(E.D_i)$ is the time to combine cells in the current cycle
$T_{load}(M_{cmi}, P)$ is the time to perform the load-balancing

The model consists of an additive sum of communication and computation components since the local gather/scatter communications effectively synchronize the processing across processors.

The gather and scatter communication time is the time taken to provide boundary information by processors owning boundary data. This is related to the 1D slab decomposition and the number of processors that share a single slab. The frequency of this operation was measured to be 160 floating point and 17 integer gathers and scatters in total per cycle. A linear model for the communication time is assumed which uses the latency (L_c) and Bandwidth (B_c) of the communication network. In addition, contention on the communication channel per node is also modeled – this is taken to be a multiplicative factor on the communication time representing the number of processors performing simultaneous out of node communications. This factor is a function of the number of cells per processor, and the number of processors.

The memory contention represents the extra time required per cycle when multiple processors within a node contend for memory. This can be measured by considering different configurations of processors for the same problem – for instance using all processors with a node, or using 1 processor in each of P_{SMP} nodes (P_{SMP} is the number of processors per node). The difference in execution time is approximately the additional time due to memory contention (assuming the communication time is small).

The AMR operation is modeled from a number of time histories of values defined on a cycle by cycle basis which can be measured for a particular calculation. These time histories represent the level 0 cell division factor (**D**), the maximum number of cells added over all processor by the division process (**A**), and the maximum number of cells moved from a single node ($\mathbf{M_{cm}}$) in the load balancing operation.

Table 1. Input parameters to the SAGE performance model (M – measured, S – specified)

Category	Type	Parameter	Description
Application	S	E	Cells per processor
	M	$\boldsymbol{D, A, M_{cm}}$	Time histories of the AMR operation for a particular calculation
Mapping	S	$Surface_x$, $Surface_y$, $Surface_Z$	Surface size (in cells) of the sub-grid on each processor (in 3 dimensions)
System	S	P	Number of processors
	S	P_{SMP}	Processors per SMP box
	S	C_L	Communication Links per SMP box
	M	$L_c(S)$, $B_c(S)$	Latencies and Bandwidths achieved in one direction on bi-directional communication (dependent on message size S in bytes)
	S	MPI_{Real8}, MPI_{INT}	Size of MPI data types
	M	$T_{comp}(E)$	Sequential cycle time of SAGE on E cells
	M	$T_{mem}(P)$	Memory contention

The parameters used in this model are listed in Table 1 according to whether they are application, system, or mapping parameters. A detailed discussion of the formation of this model can be found in [1].

The SAGE performance model has already been validated on large scale systems including several ASCI machines and the CRAY T3E. A summary of the validation results are listed in Table 2. The maximum number of processors used and the number of different processor configurations are listed. The average and maximum prediction error across the number of processor configurations are also listed. It can be seen that the model is highly accurate with an average prediction error of 5% and maximum of 11% being typical across all machines.

Table 2. SAGE performance model validation results

System	Configurations tested	Processors tested (Max)	Maximum error (%)	Average error (%)
ASCI Blue (SGI O2K)	13	5040	12.6	4.4
ASCI Red (Intel Tflops)	13	3072	10.5	5.4
ASCI White (IBM SP3)	19	4096	11.1	5.1
Compaq AlphaServer ES40	10	464	11.6	4.7
Cray T3E	17	1450	11.9	4.1

5 Use of the SAGE model to validate system performance

The SAGE model as described in Section 4, is used here to provide an expectation of the performance of the Compaq Alpha-Sever ES45 system (as described in Section 2) prior to installation. The model requires a number of measurements obtained from a small scale system. Included are the sequential processing time and the communication network performance as listed in Table 3 below. The same calculation test case was used as used for the validation (as described in Section 4).

Two performance scenarios were considered prior to the installation of the system. These differed only in the speed of the internal PCI bus of the Compaq ES45 nodes, which were initially set at 33MHz, and later upgraded to 66MHz via a software update. The speed of the PCI bus determines the available bandwidth between the Quadrics NIC and the processor memory and thus it can have a significant impact on the performance of any parallel application. In addition, when two NICs are present within the node (in a 2-rail Compaq system), the achievable communication performance increases by approximately 180% if simultaneous messages can take advantage of the two channels. Individual messages are not stripped across channels.

Table 3. Measured Compaq ES45 performance parameters for the SAGE model

Parameter	33 MHz PCI bus	66 MHz PCI bus
$T_{comp}(E)$ (s)	0.38	0.38
$L_c(S)$ (µs)	$\begin{cases} 9.00 & S < 64 \\ 9.70 & 64 \le S \le 512 \\ 17.4 & S > 512 \end{cases}$	$\begin{cases} 6.10 & S < 64 \\ 6.44 & 64 \le S \le 512 \\ 13.8 & S > 512 \end{cases}$
$1/B_c(S)$ (ns)	$\begin{cases} 0.0 & S < 64 \\ 17.8 & 64 \le S \le 512 \\ 12.8 & S > 512 \end{cases}$	$\begin{cases} 0.0 & S < 64 \\ 12.2 & 64 \le S \le 512 \\ 8.30 & S > 512 \end{cases}$
$T_{mem}(P)$ (µs)	$\begin{cases} 1.8 & P = 2 \\ 4.8 & P > 2 \end{cases}$	$\begin{cases} 1.8 & P = 2 \\ 4.8 & P > 2 \end{cases}$

5.1 Expected Performance

The performance model was used to provide the expected performance of SAGE on the Compaq ES45 system with either a 33MHz or 66MHz PCI bus using either one or two Quadrics Rails. These predictions are shown in Figure 4.

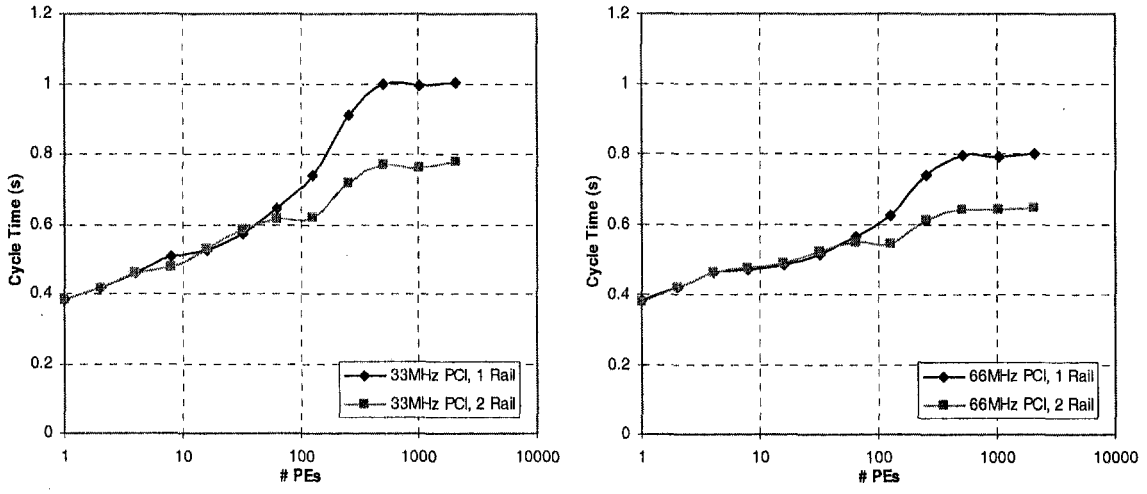

Figure 4. Performance predictions of SAGE on a Compaq ES45 system with QsNet for an ES45 with a) 33MHz PCI bus, and b) 66MHz PCI bus.

There are a number of observations that should be noted from these predictions: 1) since the runs of SAGE were performed for a weak scaling characteristic, the time to perform a single cycle should ideally be constant across all processor configurations; 2) the predicted performance is better when using 2 rails than that when using 1 rail. This occurs after a certain point (48 processors) – the point at which a single slab is mapped to more than one processor in this test case resulting in more than one simultaneous out of node communications on a gather/scatter operation; 3) The 66MHz PCI bus results in a performance of approximately 20% better than that of the 33MHz PCI bus; 4) The cycle time is predicted to plateau at above 512 processors – this is the point at which all gather/scatter communications are out-of-node.

5.2 Measured Performance

The performance of SAGE was measured at several points after the installation of the machine had taken place: as soon as the machine was up and running, after a software patch and faulty hardware replaced, and after an O/S upgrade. These three sets of measurements are compared with the model in Figure 5. The system initially had a 33MHz PCI bus. A similar process was followed after the system was upgraded to a 66MHz PCI bus. The comparison with the 66MHz PCI bus model are shown in Figure 6. Note that not all 512 nodes were available on each test date.

It can be seen from Figure 5a) that it was only after all the upgrades and system debugging had taken place that the measurements matched the expected performance. Without the model, it would have been difficult to ascertain if the application was achieving a reasonable performance or not. When differences occurred between the model and measurements, further low-level kernel tests were made on the computational nodes, and the communication network to help identify the source of the problem.

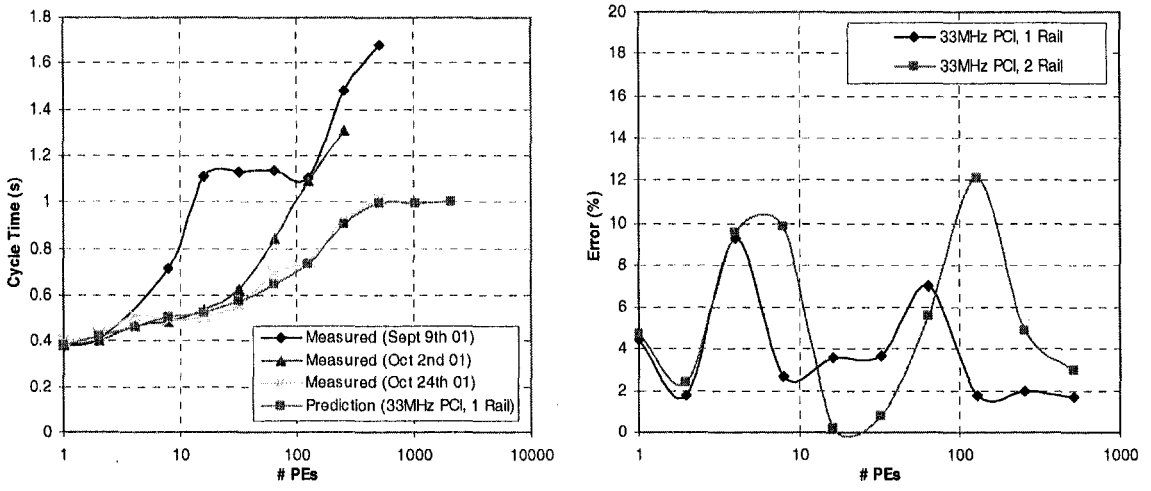

Figure 5. Measured performance of SAGE (33MHz PCI bus) compared with model predictions. a) Measurement history and model predictions using a single rail, and b) error between final measurements and model when using either 1 or 2 rails.

The errors between the model prediction and the final measurements taken on the system when in the 33MHz mode is shown in Figure 5b) when using either 1 or 2 rails. The model again proved to be highly accurate across the range of processors available for measurement.

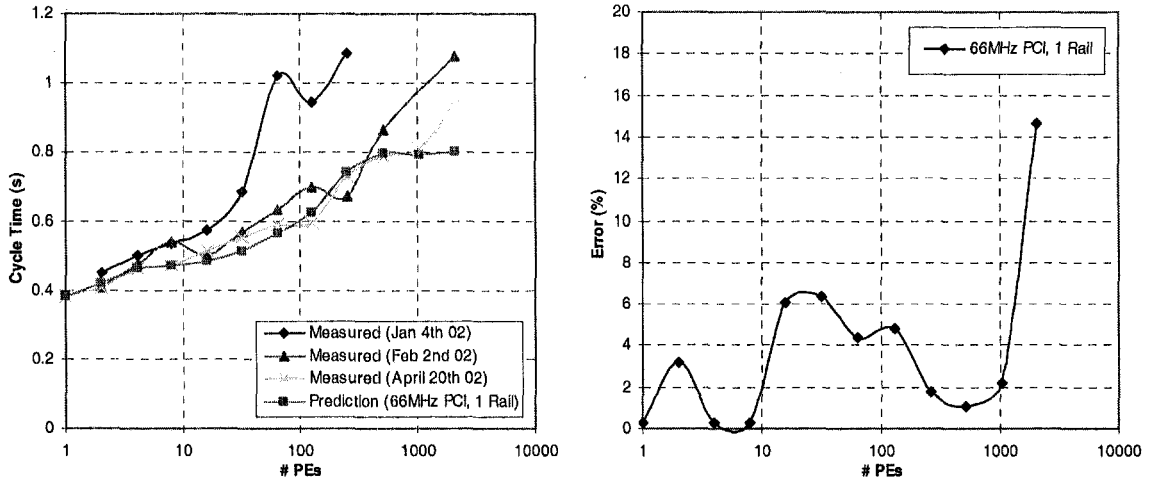

Figure 6. Measured performance of SAGE (66MHz PCI bus) compared with model predictions. a) Measurement history and model predictions using a single rail, and b) error between final measurements and model on 1.

6 Summary

In this work we have shown that an accurate performance model can be used to validate the performance of a system during its installation. The performance model of one of the ASCI codes, SAGE, has been utilized here to provide an expectation of the runtime on the system prior to its availability. Through a validation process, this model has been shown to be accurate on many large-scale systems in different configurations.

When installing a new system there are often a number of refinements that need to be done in both the software system, and hardware components, before the machine operates at the expected level of performance. The performance model for SAGE has been shown to be of great use in this process. The model has effectively provided the performance and scalability baseline for the system performance on a realistic workload. Initial system testing showed that its performance was almost 50% less than expected. After several system refinements and upgrades over a number of months, the achieved performance matched exactly the expectation provided by the model. Thus performance models can be used to validate system performance.

The model will be used further as the installation of the ASCI Q 30Tflop system proceeds. The performance analysis will be coupled with a further model of an ASCI code – Sweep3D [3]. This work is part of an on-going effort to accurately model the ASCI applications on existing and future large-scale systems.

References

1. Kerbyson, D.J., Alme, H.J., Hoisie, A., Petrini, F., Wasserman, H.J., Gittings, M.L.: Predictive Performance and Scalability Modeling of a Large-scale Application, in Proc SC2001, Denver (2001)

2. Worley. P.H.: Performance Tuning and Evaluation of a Parallel Community Climate Model, in Proc. SC99, Portland (1999)

3. Hoisie. A., Lubeck, O., Wasserman. H.: Performance and scalability analysis of teraflop-scale parallel architectures using multidimensional wavefront applications, Int. J. of High Performance Computing Applications, 14 (2000) 330-346

4. Petrini, F., Feng, W.C., Hoisie, A., Coll, S., Frachtenberg, E.: The Quadrics Network: High-Performance Clustering Technology, IEEE Micro, 22(1) (2002) 46-57

5. Weaver, R.: Major 3-D Parallel Simulations, BITS -Computing and communication news, Los Alamos National Laboratory, June/July, 1999, 9-11, http://www.lanl.gov/orgs/cic/cic6/bits/99june_julybits/opener.html

6. Goedecker, S., Hoisie, A.: Performance Optimization of Numerically Intensive Codes, Society for Industrial & Applied Methematics; ISBN: 0898714842 (2001)

7. Nudd, G.R., Kerbyson, D.J., et.al. PACE: A Toolset for the Performance Prediction of Parallel and Distributed Systems, Int. J. of High Performance Computing Applications, 14 (2000) 228-251