# Monitoring Large Systems via Statistical Sampling *

Celso L. Mendes and Daniel A. Reed
Department of Computer Science
University of Illinois
Urbana, Illinois 61801
{cmendes,reed}@cs.uiuc.edu

**Abstract**

As the trend in parallel systems scales toward petaflop performance tapped by advances in circuit density and by an increasingly available computational Grid, the development of efficient mechanisms for monitoring large systems becomes imperative. When computational components are coupled via dynamically shifting connections with various remote resources, the number of potential factors affecting system behavior is enormous. Yet the overhead of monitoring can be prohibitive. This paper presents a new technique for monitoring large systems based on statistical sampling. Rather than monitoring each component, we select a statistically valid sample and measure the behavior of sample members. We describe the formal requirements of sample selection and verify the feasibility of our approach with experiments on large parallel systems and wide-area networks. Our results show that this technique can be a powerful tool to enable effective monitoring without incurring the large costs typically associated to exhaustive checking.

## 1 Introduction

Recent hardware advances have enabled the creation of parallel systems with thousands of processors. Current systems typically employ multiple processors per board and, given the present trends in microelectronics and packaging, systems with multiple processors per chip will soon become widely available. Furthermore, with the ongoing adoption of the Grid [6] computing platform, systems with increasingly large numbers of processors will play a central role in the future of parallel computing.

Despite their excellent potential in terms of computing capability, systems with large numbers of processors present two major challenges. The first regards resource management: algorithms that achieve efficient usage of available resources typically require information from all system components. Comprehensive data collection can be costly on a large system, becoming completely prohibitive in some instances. Consider, as an example, a parallel machine comprising 20,000 processors where one needs to determine the average processor load. This could be achieved by posting queries to each individual processor. However, even assuming an optimistic delay of 10 ms per query, which corresponds to the timer period in many systems, it would take more than three minutes to complete the measurements. This duration would probably make the obtained result useless.

*This work was supported in part by the National Science Foundation under grant NSF EIA-9975020 and by the Department of Energy under contract DOE W-7405-ENG-36.

The second challenge faced by large systems is reliability. While the probability of failure increases with the number of basic components, the cost of inspecting each component for failure may preclude undertaking regular inspections. In many situations, the system can still produce useful work even under the presence of a limited amount of failing components. The critical problem thus becomes to determine if the current number of failures has crossed a certain threshold. Again, the procedures available to count the number of existing failures may take too long and become impractical.

In this paper, we propose a new technique to monitor large systems, based on statistical sampling. Instead of checking every system component individually, we select a statistically valid subset of components, inspect this subset in detail, and derive estimates for the whole system based on the properties found in the subset. The quality of these estimates depends on the components in the chosen subset. Statistical sampling provides a formal basis for quantifying the resulting accuracy of the estimation, and guides the selection of a subset that meets accuracy specifications.

In contrast to traditional monitoring schemes, where the monitoring cost grows linearly with the number of system components, statistical sampling has the interesting property that sampling costs may depend only on the characteristics of the components, not on the number of components. Thus, when such characteristics remain within given limits, the sampling costs may remain bounded, regardless of system size. This property is essential to enabling effective monitoring of increasingly large systems that are expected to be available in the future.

The remainder of this paper is organized as follows. §2 presents the theoretical fundaments of statistical sampling. We apply the sampling technique, in §3, to estimate the status of clusters and large parallel machines, and in §4, to assess network status as viewed from a given site. Those two sections clearly demonstrate the sampling effectiveness and the gains that are obtained in terms of measurement cost. §5 discusses related work, and we conclude in §6 presenting directions for our future work.

# 2 Statistical Sampling Theory

The principles and methods of collecting and analyzing data from finite populations form a branch of Statistics known as *Sample Survey Methods.* Their formal basis is termed *Sampling Theory*. The goal of sampling is to estimate some property of the entire population based on data collected from only a a subset of the elements — what is called a *sample.* In this section, we present the major requirements of an appropriate sampling scheme.

## 2.1 Principles of Random Sampling

Assuming that a population has $N$ elements and that there is a variable, $Y$, that may take different values across these elements, we can represent the population by the set of values $Y_1, Y_2, \ldots, Y_N$, where each $Y_i$ is the value taken by element $i, 1 \leq i \leq N$. A subset of that population with $n$ elements $y_1, y_2, \ldots, y_n$ is said to be a *simple random sample* if that subset is chosen at random from all $\binom{N}{n}$ distinct possible subsets in which no element is included more than once. In other words, each of the $\binom{N}{n}$ subsets has the same probability $\binom{N}{n}^{-1}$ of being chosen. A simple random sample can be obtained sequentially by randomly drawing elements from the population one at a time, without replacement, i.e., once chosen, an element is removed from subsequent drawings [1, 4].

An important factor related to sampling is the appropriate sample size, which corresponds to the sampling cost. The sample size is intrinsically associated with two features of the sample as an estimator: the *confidence* provided by the sample and its *accuracy*. The accuracy or precision of an estimation scheme defines an interval, centered on the actual value of the population's property, where the estimate is assumed to be. The confidence determines how often the estimate is expected to be

| $\alpha$ | 0.10 | 0.05 | 0.02 | 0.01 | 0.002 | 0.001 |
|---|---|---|---|---|---|---|
| **Confidence** | 90% | 95% | 98% | 99% | 99.8% | 99.9% |
| $z_\alpha$ | 1.645 | 1.960 | 2.326 | 2.576 | 3.090 | 3.291 |

Table 1: Values for the confidence parameter.

inside that interval. There is always a balance of confidence and accuracy versus sampling cost: the better the desired confidence or accuracy, the more expensive is the sampling.

## 2.2 Estimation of Mean Values

Given the $Y$ values for the various population elements and their mean $M_Y$, we can estimate the value of $M_Y$ using the sample mean $m_y$, computed from the sample values $y_1, y_2, \ldots, y_n$. A condition for the minimum sample size $n$ can be determined by restricting to an acceptable level the probability that the absolute difference between $M_Y$ and $m_y$ be greater than some specified value. This condition can be represented by

$$Pr(|M_Y - m_y| > d) \leq \alpha \tag{1}$$

for some specified accuracy $d$ and risk $\alpha$ of not obtaining such accuracy. According to [1], this can be achieved with a sample size $n$ such that

$$n \geq N \left[ 1 + N \left( \frac{d}{z_\alpha S} \right)^2 \right]^{-1} \tag{2}$$

where $S$ is the standard deviation of the population values $Y_i$ and $z_\alpha$ is the double-tailed $\alpha$-point of a normal distribution with zero mean and unity variance. Some values of $z_\alpha$, extracted from the Standard Normal probability table, are shown in Table 1.

Thus, given the population size $N$ and specifications for the desired confidence $z_\alpha$ and for the accuracy $d$ as a certain fraction of $S$, one can use Formula 2 directly to determine the minimum sample size. A potential problem arises when the accuracy $d$ is specified in absolute terms (instead of relatively to $S$) and the value of $S$ is unknown. There are a few different possibilities to estimate $S$: we can use results from previous experiments with the same population, or we can split the estimation in two phases: in the first phase, a small sample is selected, and the standard deviation from this first sample is used to compute the required sample size; in the second phase, that first sample is completed with more elements from the population, forming the required full sample.

Analysis of Formula 2 shows that, for large values of $N$, the minimum sample size approaches $(z_\alpha S/d)^2$. Thus, the sampling cost depends on $S^2$, and does not depend on the population size. If the standard deviation $S$ remains bounded as the population grows, the required sample size does not increase. In that case, the relative sampling cost $n/N$ becomes progressively smaller.

## 2.3 Estimation of Proportions

Rather than studying a quantitative measure, $Y$, we may sometimes be concerned with some qualitative attribute or characteristic that can exist for the elements in a population. We might then need to assess the proportion, $P$, of elements who possess that attribute. We can define a variable $X_i$ describing the $i^{th}$ element of the population such that $X_i = 1$ if the element possesses the attribute, and $X_i = 0$ otherwise. The proportion $P$ thus becomes

$$P = \frac{1}{N} \sum_{i=1}^{N} X_i \tag{3}$$

To estimate $P$, we take a sample of $n$ elements, and define the variables $x_1, x_2, \ldots, x_n$ for those elements such that $x_i$ is 1 if the $i^{th}$ element in the sample possesses the underlying attribute, and 0 otherwise. An estimation of $P$ is provided by the mean $p$, computed from the sample variables $x_1, x_2, \ldots, x_n$. Similarly to the case of mean value estimation, we can define a condition for the minimum sample size $n$ by imposing that

$$Pr(|P - p| > d) \leq \alpha \tag{4}$$

Under such condition, the minimum sample size is given[1] by

$$n \geq N \left[ 1 + \frac{N-1}{P(1-P)} \left( \frac{d}{z_\alpha} \right)^2 \right]^{-1} \tag{5}$$

where $z_\alpha$ is the same as in Table 1 and $d$ is the specified accuracy in the proportion estimation. Notice that we are faced with a situation where the sample size depends on the factor which we are trying to estimate — the proportion $P$. We could obtain a priori estimates of $P$ by using the same methods suggested in the previous subsection for estimating the standard deviation, namely, prior experiments or a two-phase sampling approach. However, in this case there is an additional feature that makes the estimation different. Because the proportion $P$ can only vary between $0.0$ and $1.0$, the product $P(1-P)$ is always between $0.0$ and $0.25$, reaching a maximum when $P = 0.5$.

For large values of $N$, the right-hand side of Formula 5 approaches $P(1-P)(z_\alpha/d)^2$. Given the specifications for accuracy ($d$) and confidence ($z_\alpha$), the required sample size has an upper bound corresponding to $0.25(z_\alpha/d)^2$, derived from the case of $P = 0.5$. When the actual proportion $P$ is close to $0.0$ or to $1.0$, that upper bound will result in an oversampling, meaning that we will be using a larger sample than would be really needed.

# 3 Application to System Monitoring

To verify the feasibility of monitoring large systems using statistical sampling techniques, we conducted an extensive series of experiments covering a variety of the largest parallel systems existing presently. We also simulated larger systems that have not been built yet, but are expected to exist in the future.

## 3.1 Monitoring of Large Clusters

In our first sampling experiment, we analyzed the utilization of NCSA's IA-32 cluster. In this cluster, application programs can run on up to 480 compute nodes, under management of a PBS batch system. Each node is a dual-processor Pentium-III box running at 1 GHz, and is exclusively allocated to a job from one of the submission queues. NCSA has a cluster monitor with a Web interface[2] that displays cluster information updated at every minute. We collected periodic snapshots of such node status information during the first ten days of February/2002, and for each snapshot we measured the fraction of nodes with a status of "Available". We used a period of 3 minutes (20 measurements per hour, or 4800 for the full period[3]), obtaining the free-node information shown in Figure 1.

After collecting this full dataset, we took each snapshot and applied sampling to assess our ability to estimate the fraction of free nodes at that moment. As a first approach, we used a fixed sample size.

[1] For details, see [1].

[2] http://padmin2.ncsa.uiuc.edu

[3] During some of those 4800 measurements, either the cluster was completely down or we did not have access to the monitor information. The number of effective measurements was 4645.

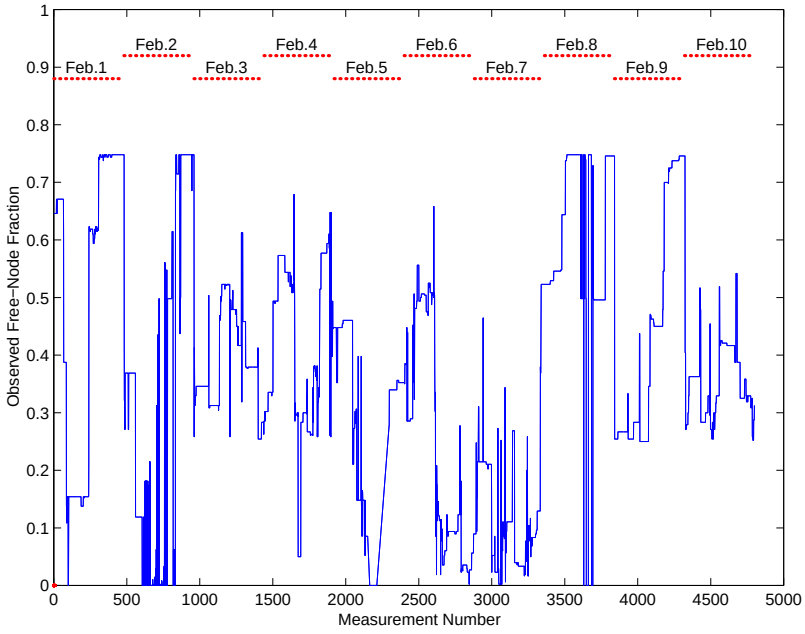
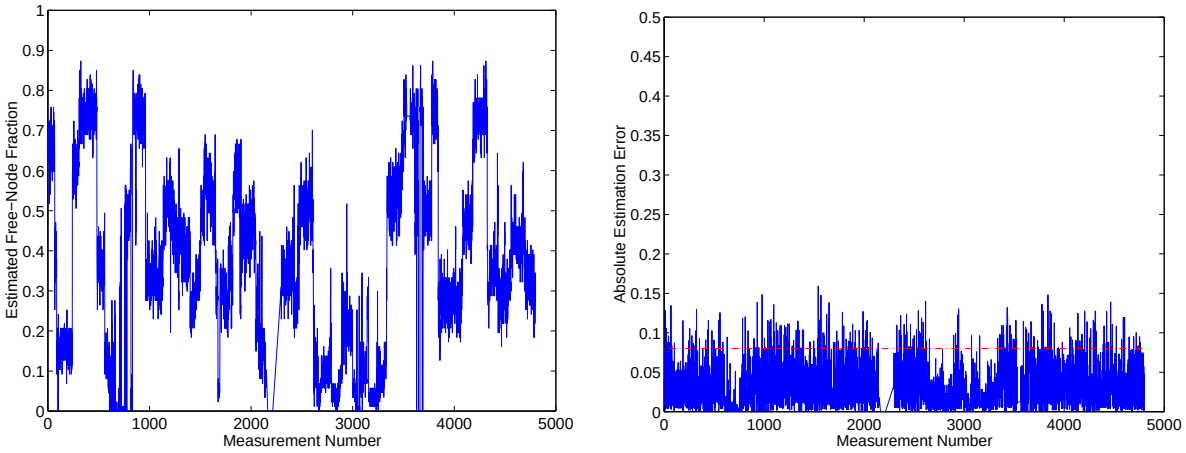



Figure 1: Observed free-node fraction on NCSA's IA-32 cluster.

For a population size of 480 nodes, an expected confidence of 90% and an accuracy of 8%, Formula 5 from §2.3 prescribes a minimum sample size of 87. Thus, for each snapshot, we randomly picked 87 nodes and computed the free-node fraction on that subset. Figure 2(a) shows the estimation results across all snapshots, and Figure 2(b) presents the absolute value of the error in the free-node fraction estimation.

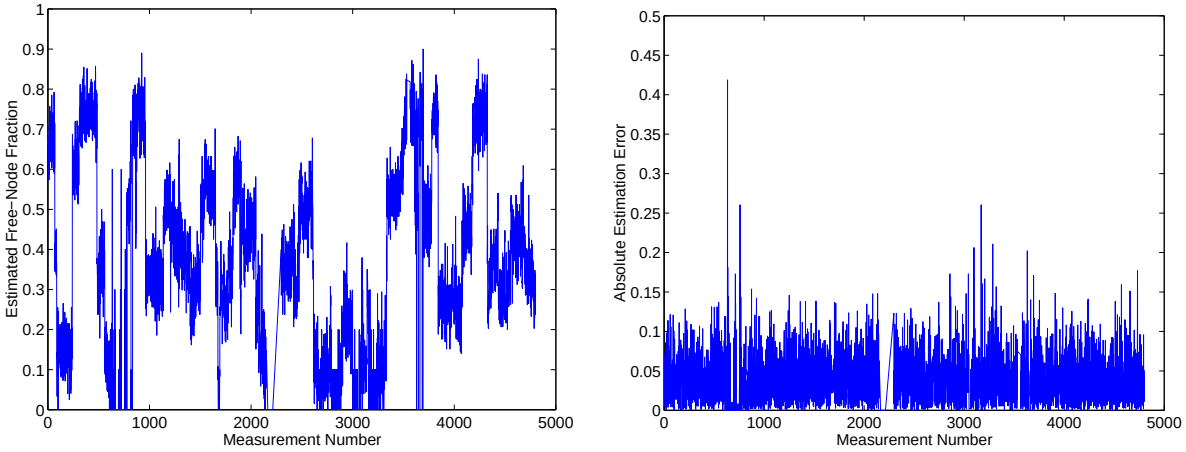
Inspection of data in Figures 1 and 2 reveals that our estimates tracked very well the real cluster status. Among the 4645 valid measurements, our estimates were within the valid 8% accuracy range on 4366 of them, corresponding to a success rate of 94%. This success rate was better than our designed confidence rate of 90% because of the fixed sample size used: our sample size of 87 nodes was computed assuming a free-node fraction of 0.5. When the actual free-node fraction is closer to 0.0 or to 1.0, smaller samples would be sufficient. Based on such observation, we repeated the sampling experiment, this time using a variable sample size. We implemented the new sampling scheme by using the free-node fraction observed in the previous sampling moment to compute the new sample size for a snapshot, and allowed this sample size to be as low as 10 nodes. This scheme is based on the assumption that the real free-node fraction does not vary significantly from one snapshot to the next.

Using the variable sample size scheme, we obtained the new estimates shown in Figure 3(a), with the corresponding estimation error of Figure 3(b). Estimates for 4190 measurements were within the valid accuracy range, representing a 90.2% success rate. The mean value for the sample size across all measurements was 65.8 nodes. Thus, our new sampling scheme had a cost that was 24% lower than in the case of fixed sample size (i.e. samples with 65.8 nodes on average, instead of the fixed sample size of 87 nodes), while still achieving the expected 90% confidence level. The lower cost comes from adjusting the sample size at each moment. Our results show that by sampling, on average, only 66 nodes (14% of the cluster size), we still managed to obtain a good assessment of the cluster status.




(a) Estimation results (b) Estimation error and accuracy threshold

Figure 2: Estimation on NCSA's IA-32 cluster with fixed sample size of 87 nodes.



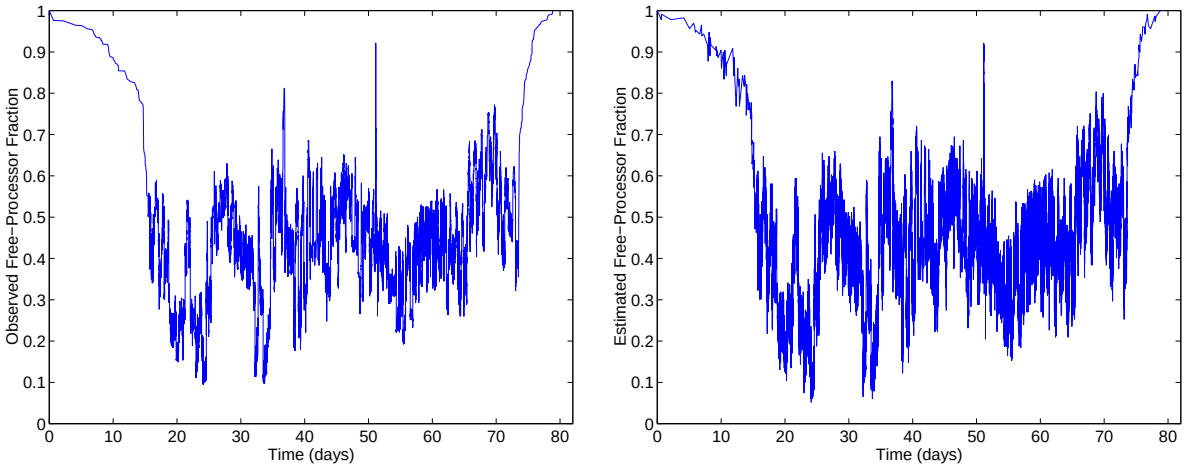

(a) Estimation results (b) Estimation error

Figure 3: Estimation on NCSA's IA-32 cluster with variable sample size.

## 3.2 Analysis of Shared Memory Arrays

In our next experiment, we analyzed processor utilization in the SGI Origin2000 array at NCSA. That array consists of twelve machines, with each machine containing between 64 and 256 MIPS-R10K processors. One of the machines accepts interactive access, while the remaining eleven machines, comprising a total of 1464 processors, service batch jobs. We focused our analysis on this part of the array, since batch jobs constitute the major load on the array utilization.

The Origin2000 array does not have a real-time monitor of processor utilization like the other system that we analyzed before. However, NCSA maintains a log with information about each executed batch job. By analyzing such log records from a ten-week period in the past, we reproduced the behavior of the array upon executing jobs during that period. For every moment of a job dispatch or a job termination, we applied our sampling scheme to estimate processor availability across the entire array. Considering the array population of 1464 processors, we specified a sampling confidence of 90% and a sampling accuracy of 5%, obtaining a minimum sample size of 229 processors. Thus, on every sampling moment we randomly picked 229 processors and computed the free-processor fraction




(a) Observed free-processor fraction (b) Estimated free-processor fraction

Figure 4: Estimation results from sampling NCSA's SGI Origin2000 array.

for that subset. This sample size corresponds to 15.6% of the total array size.

Figure 4 shows the observed and estimated fractions of free processors during the period of our analysis. The extreme right and left parts in the plots correspond to periods where the logs did not contain information about all jobs in the array. Our estimates achieved the selected accuracy in 91.5% of the sampling trials. Hence, our sampling goals were fully met.

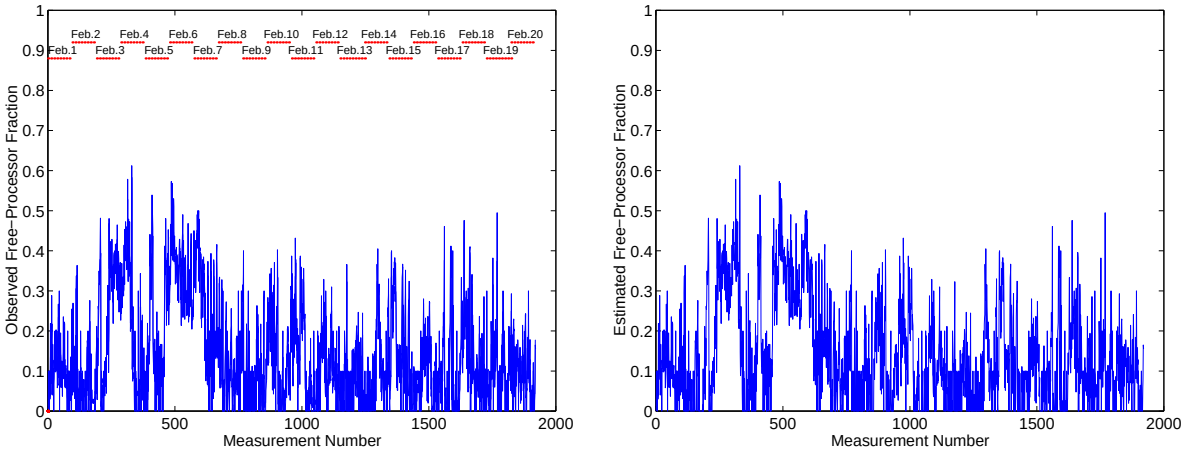
## 3.3 Monitoring of Distributed Memory Machines

We analyzed the usage of NERSC's system known as *Seaborg*, an IBM-SP with 188 compute nodes where each node contains 16 shared-memory Power3 processors running at 375 MHz. The job scheduling policy at NERSC is such that jobs are exclusively allocated to an integer number of nodes for execution, and a job can start between one and sixteen tasks per node.

There is a Web interface[4] that shows processor usage for each node in the system; this information is updated every 15 minutes. Using that interface, we monitored this cluster during the first twenty days of February/2002. On each measurement, we sampled the set of 3008 processors to estimate the fraction of processors not used by the current jobs. We used the same sampling parameters as in the NCSA's IA-32 cluster (confidence of 90% and accuracy of 8%). However, due to the larger interval between measurements, we adopted a different method to determine the sample size. First, we took a sample consisting of ten random processors, and then used the fraction of free processors in that pre-sample to find the final sample size prescribed by Formula 5.

Figure 5 shows the results from our estimation. Among the total 1920 measurements, our estimation was within the required 8% accuracy 1747 times, corresponding to an estimation success rate of 91%. The selected sample size varied from a minimum of 10 processors to a maximum of 103 processors; the average sample size obtained was 46 processors, or 1.5% of the machine size. Hence, we still track the machine's utilization quite well with a cost of only 1.5% of an exhaustive monitoring scheme. This example shows that administrators of this system could take a similar approach to obtain other kinds of collective information about the processors (e.g. processor load, faults detected, etc.) with a cost that is more than 60 times lower than the cost of regular monitoring.

[4]The URL is http://hpcf.nersc.gov/cgi-bin/qstat/llq_seaborg but that page is protected by username/password.




(a) Observed free-processor fraction (b) Estimated free-processor fraction

Figure 5: Estimation on NERSC's IBM-SP with variable sample size.

### 3.4 Simulation of Future Systems

To assess the effectiveness of sampling on future systems, we simulated a system containing 50,000 processors. In our simulation scheme, jobs are randomly created and belong to one of two classes, *small* and *large*, according to their expected duration and number of processors. We simulated this machine running for 1000 cycles, and a total of 122,127 jobs were executed. The average processor load was 75.5%, which is compatible with the load observed in real production systems [3].

Using the simulation data, we conducted several sampling experiments where we varied the sampling specifications. For a given confidence/accuracy combination, we sampled the processors on each cycle using a fixed sample size, and estimated the fraction of free processors. Table 2 shows the sample size used for each experiment, as prescribed by Formula 5 when $P = 0.5$, and the resulting success rates from the estimations.

We can note several facts in Table 2. First, the success rate in the estimation was always better than the designed confidence; this was due to the use of a fixed sample size for each experiment. Second, for a given accuracy, increasing confidence levels demanded larger sample sizes, but provided, in return, better success rates, as expected. Finally, one can see that our sampling was very effective: a sample with 106 processors (0.2% of the machine size) achieved the estimation accuracy of 8% in 0.950 of the cycles; our largest sample, with 1778 processors (3.6% of the machine size), reached an estimation accuracy of 3% in 0.998 of the cycles.

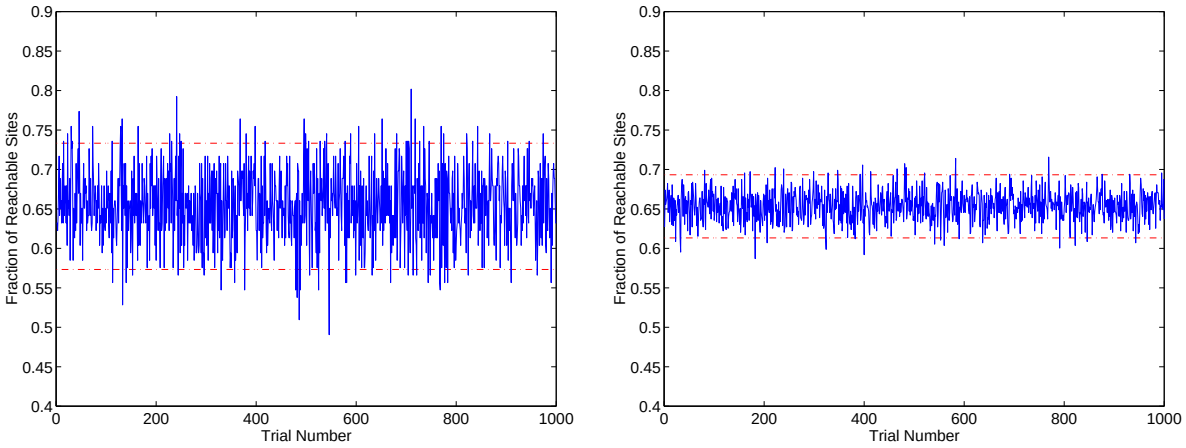
The numbers in Table 2 confirm both the expected tradeoff between sampling cost and accuracy, and the effectiveness of the sampling technique. We can monitor an extremely small subset of the processors and still achieve specified levels of accuracy and confidence in the estimation of global system state. This monitoring cost economy, with the corresponding reduction of a few orders of magnitude in the time required to derive global system figures, may enable the utilization of system control algorithms that would be impractical under traditional monitoring schemes.

## 4 Application to Network Connectivity Assessment

Another field where sampling can be extremely useful is the analysis of wide-area networks, comprising thousands of geographically distributed sites. We can apply sampling to estimate both the fraction of sites currently reachable from a network point, and the mean latency observed from that point.

| Accuracy | Sample Size (Processors) | | | | Success Rate | | | |
|---|---|---|---|---|---|---|---|---|
| | Confidence | | | | Confidence | | | |
| | 90% | 95% | 98% | 99% | 90% | 95% | 98% | 99% |
| 8% | 106 | 150 | 211 | 258 | 0.950 | 0.972 | 0.993 | 0.995 |
| 5% | 270 | 382 | 536 | 655 | 0.962 | 0.977 | 0.987 | 0.998 |
| 3% | 741 | 1045 | 1460 | 1778 | 0.948 | 0.981 | 0.987 | 0.998 |

Table 2: Estimation results from sampling simulated machine.




(a) Confidence=90%, Accuracy=8%
106 sampled sites

(b) Confidence=95%, Accuracy=4%
598 sampled sites

Figure 6: Results of network reachability estimation.

## 4.1 Network Reachability Estimation

Given a collection of interconnected sites, with one of them as a source site, one may be interested in determining how many of the remaining sites are reachable at a certain moment. Some sites may be down or unreachable due to routing problems. To provide an experimental basis for our tests, we considered the list of Internet sites assembled by the Internet Mapping Project [2]. This list comprises 126,930 IP addresses of what the project developers claim to be "registered Internet entities". Each address may correspond to a machine, a router, or any device that represents a certain sub-network.

Taking that list of sites as our universe, we conducted experiments to determine the fraction of sites that were reachable from a specific machine at Urbana/Illinois. By "reachable" we consider a site that responds to a *ping* request. Initially, we made one request to every site in the list, and stored the obtained results. On such exhaustive sequence of accesses, 82,920 sites responded to the request, corresponding to 65.3% of the total sites. Next, we conducted two experiments to estimate the fraction of reachable sites under distinct sampling specifications, repeating the sampling process 1000 times in each case. Figure 6 shows the estimation results, with dotted horizontal lines indicating the underlying accuracy specifications. The numbers of successful estimations were 918 and 962, respectively; thus, the corresponding confidence levels from our design (90% and 95%) were achieved in both cases.

## 4.2 Network Latency Estimation

Another useful metric in network environments is the latency between sites. Many of the current efforts to develop Grid applications envision using latency and bandwidth numbers to select execution

resources. Some services [13] provide this information in real-time, but only across limited sets of sites. Hence, an efficient mechanism to derive these figures for a large environment would be useful. In our tests, latency is defined as the mean response time for packets returned from the *ping* requests.

For the exhaustive sequence of requests that we had conducted before, and considering only those sites that responded to a request, we observed a mean latency of 143.3 ms. Next, we conducted a sampling experiment to estimate the mean access latency to those responding sites. We imposed an accuracy $d = S/8$, so that it was not necessary to know $S$ to determine the sample size in Formula 2. Selecting a confidence of 95%, with our population of 82,920 sites, results in a minimum sample size of 246 sites. Using this sample size on repeated sampling trials, we obtained the latency estimates of Figure 7(a). These estimates were within the selected accuracy in 95.9% of the trials.

It is instructive to analyze the gains provided by the sampling technique is this case. Assume a 3 second duration per *ping* request, such that we obtain at least a few return packets even from the most network-distant sites. Measurements with the subset of 246 sites will complete in nearly 12 minutes, whereas an exhaustive measurement with all 82,920 sites would require 69 hours. It is clearly impractical to conduct this exhaustive procedure on a regular basis.

### 4.3 Cost-Constrained Sampling

There may be situations where the sampling process has constraints other than just confidence and accuracy. Assuming the wait of 3 seconds per *ping* request and a sample size of 246 sites as in the previous subsection, a latency estimation measurement would require nearly 12 minutes. This duration might be excessively large in some cases. We must then view the sampling design from a different angle: given the maximum sampling cost that can be tolerated, find the corresponding confidence and accuracy that one can expect from sampling.

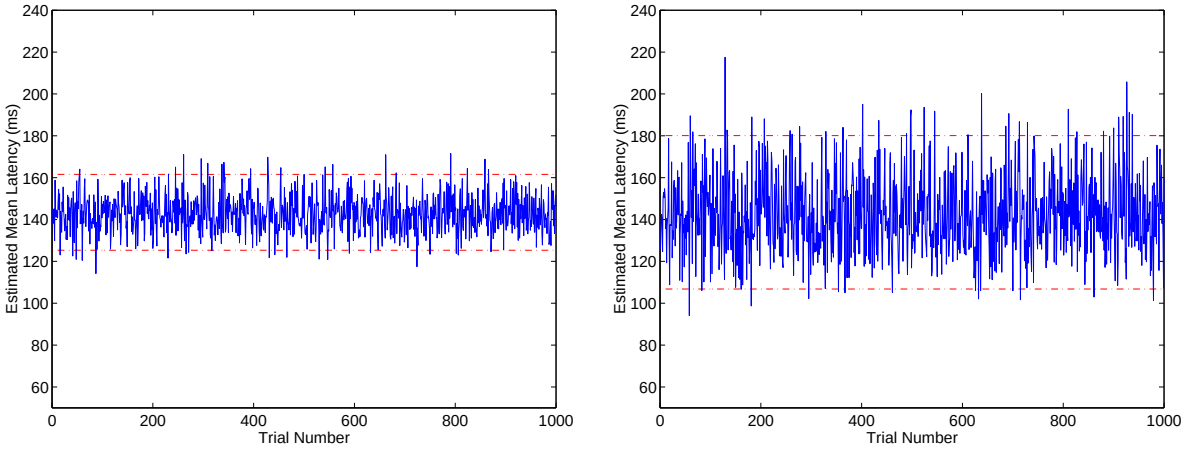
Supposing that we had at most 6 minutes to conduct a complete experiment, we would be limited to accessing 120 sites. If at least half of such sites responded to the requests, we might design our experiment with an effective sample set of 60 sites. According to Formula 2, this reduced sample size would correspond to lower confidence or worse accuracy. We took the responses from our exhaustive access to all sites and conducted an experiment where we repeatedly sampled 60 of the responding sites. Under the same confidence of 95% as before, this new sample size represents an accuracy of $S/3.95$. Figure 7(b) presents the results of this experiment, where our estimates achieved the required accuracy in 952 out of 1000 trials.

The effectiveness and tradeoffs involved in a sampling scheme should now be clear. While a decrease in the sample size results in more estimation variability, as one can verify by comparing Figures 7(a) and 7(b), the expected confidence in the estimation was always achieved. Even with a small sample size, one can still obtain a good approximation on the status of accessibility to a large network. Furthermore, by using sampling techniques, this approximation is achieved in minutes, instead of days like required by a full precision measurement.

## 5 Related Work

Most applications of statistical sampling in the past [1, 10, 14] have been in the context of human or animal population surveys. Statistical sampling was also applied in [8] to estimate software reliability. That study combined operational testing and sampling techniques in order to reduce the number of program executions that must be checked manually for conformance to requirements.

In computer systems, applications of sampling techniques usually focus on fault detection and testing issues related to hardware design. In [7], such techniques were used to generate a subset of the possible design errors or test patterns that one must consider when verifying a new hardware design.




(a) Accuracy $d = S/8$, 246 sampled sites (b) Accuracy $d = S/3.95$, 60 sampled sites

Figure 7: Estimation of mean network latency from machine at Urbana/Illinois.

[5] applied statistical sampling to estimate power demands in circuits, and compared the results from stratified sampling to those from simple random sampling experiments.

In the area of performance analysis, statistical clustering [9] and projection pursuit [12] have proved to be efficient techniques to handle large amounts of performance data. Clustering can reduce the number of processors to be monitored, and projection pursuit can reduce the number of captured metrics. These techniques, however, require periodic analysis of data from all processors and all available metrics. Statistical sampling could complement those capabilities, by allowing the analysis to focus on subsets of processors or metrics, thus becoming more cost effective in large systems.

# 6 Conclusion and Future Work

We have presented a new methodology to monitor large systems, based on statistical sampling techniques. Our experiments demonstrated the effectiveness of these techniques to estimate the fraction of available processors in parallel machines, the fraction of network sites reachable from a certain point, and the mean latency expected from that point to the rest of the network. In all those experiments, the confidence and accuracy specifications were fully achieved by observed data.

Our results show that we can reliably estimate the state of a large system with a cost that is a small fraction of the cost required by traditional monitoring schemes. Our various experiments also confirm that the relative reduction in measurement cost improves as system sizes increase. This cost reduction, in turn, can enable measurements that would be completely impractical by regular means. It can also enable the use of more powerful algorithms for system management and for optimized resource utilization.

We are working to expand the applicability of this technique in several directions. On parallel systems, we are studying the feasibility of other sampling schemes, such as stratified sampling, which could improve accuracy or decrease sampling cost for some heterogeneous environments, like the Origin2000 array. We are also investigating a coupling to time-series analysis [11], as an attempt to improve estimation based on past observed behavior. Another current effort is to apply this technique to improve instrumentation efficiency; one could activate instrumentation in a subset of the processors running an application and still capture sufficient data to understand application performance.

In network environments, we are trying to combine and correlate sampled data collected at more than one site, as a way to improve our assessment of global network state. In that direction, another

goal is to find the convenient number and location of collection sites. We are confident that these efforts will be essential for the efficient use of the Grid as a computing platform. We believe that a statistical approach is one of the most effective ways to manage, select and monitor Grid resources.

# Acknowledgement

We would like to thank John Towns, from NCSA, for providing the logs of the SGI Origin2000 jobs.

# References

[1] BARNETT, V. *Sample Survey: Principles and Methods.* Oxford University Press, New York, 1991.

[2] CHESWICK, W., AND BURCH, H. The Internet mapping project. *Wired Magazine* (December 1998). See http://research.lumeta.com/ches/map/index.html.

[3] CHIANG, S., AND VERNON, M. K. Production job scheduling for parallel shared memory systems. *Submitted to publication* (2002).

[4] COCHRAN, W. G. *Sampling Techniques*, third ed. John Wiley and Sons, 1977.

[5] DING, C., HSIEH, C., WU, Q., AND PEDRAM, M. Stratified random sampling for power estimation. In *Proceedings of the 1996 IEEE/ACM International Conference on Computer-Aided Design* (San Jose, California, 1996).

[6] FOSTER, I., AND KESSELMAN, C., Eds. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, July 1998.

[7] HUR, Y., AND SZYGENDA, S. A. Design error simulation based on error modeling and sampling techniques. *Mathematics and Computers in Simulation*, 46 (1998), 35–46.

[8] PODGURSKY, A., ET AL. Estimation of software reliability by stratified sampling. *ACM Transactions on Software Engineering and Methodology 8*, 3 (July 1999), 263–283.

[9] REED, D. A., NICKOLAYEV, O. Y., AND ROTH, P. C. Real-time statistical clustering for event trace reduction. *Journal of Supercomputing Applications and High-Performance Computing 11*, 2 (1997), 144–159.

[10] SUKHATME, P. V., AND SUKHATME, B. V. *Sampling Theory of Surveys with Applications*, second ed. Iowa State University Press, 1970.

[11] TRAN, N. *Automatic Arima Time Series Modeling and Forecasting for Adaptive Input/Output Prefetching.* PhD thesis, University of Illinois, 2001.

[12] VETTER, J. S., AND REED, D. A. Managing performance analysis with dynamic statistical projection pursuit. In *Proceedings of Supercomputing-99* (Portland, November 1999).

[13] WOLSKI, R., SPRING, N., AND HAYES, J. The Network Weather Service: A distributed resource performance forecasting service for metacomputing. *Journal of Future Generation Computing Systems 15*, 5 (October 1999), 757–768.

[14] YOCCOZ, N. G., NICHOLS, J. D., AND BOULINIER, T. Monitoring of biological diversity in space and time. *TRENDS in Ecology and Evolution 16*, 8 (August 2001), 446–453.