Comparison of the ESP Benchmark with Observed System Utilization

Adrian T. Wong, William T. C. Kramer, Leonid Oliker, and David H. Bailey
National Energy Research Scientific Computing Center
Lawrence Berkeley National Laboratory
One Cyclotron Road, Berkeley, CA 94720, USA
Correspondence: William Kramer – kramer@nersc.gov

Abstract

This paper describes a new benchmark, called the Effective System Performance (ESP) test, which is designed to measure system-level performance, including such factors as job scheduling efficiency, handling of large jobs and shutdown-reboot times. In particular, this test can be used to study the effects of various scheduling policies and parameters. The ESP is validated by comparing dedicated time test results to actual utilization of the original workload running under similar scheduling parameters.

Introduction

The overall performance value of a high performance computing system depends not only on its raw computational speed but also on system management effectiveness, including job scheduling efficiency, reboot and recovery times and the level of process management. Common performance metrics such as the LINPACK and NAS Parallel Benchmarks [3, 1] are useful for measuring computational performance for individual jobs, but give little or no insight into system-level efficiency issues. This paper, continues the investigation of a new benchmark, the Effective System Performance (ESP) benchmark, first discussed in [12], which measures system utilization and effectiveness. Our primary motivation in developing this benchmark is to aid the evaluation of high performance systems. We use it to monitor the impact of configuration changes and software upgrades in existing systems, but are evolving this benchmark to provide a focal point for future research and development activities in the high performance computing community. We believe it will lead to significantly improved system-level efficiency in future production systems.

The ESP test extends the idea of a throughput benchmark with additional features that mimic day-to-day supercomputer center operation. It yields an efficiency measurement based on the ratio of the actual elapsed time relative to the theoretical minimum time assuming perfect efficiency. This ratio is independent of the computational rate and is also relatively independent of the number of processors used, thus permitting comparisons between platforms.

5. ESP Benchmark Design

As discussed indetail in [11 and 12], the throughput workload that is currently used in the ESP test consists of a set of jobs of varying partition sizes and times with the objective of obtaining the shortest elapsed run time. By reporting the utilization efficiency E instead of the absolute time, the ESP test is independent of the computational rate. The ESP test runs roughly four hours on 512 CPUs of the NERSC T3E and approximately 2 hours on the IBM SP. This time length was a compromise between a longer simulation that is more representative of actual usage and a shorter time that is more suitable to routine benchmarking.

The throughput of large-partition jobs is an ongoing concern at a large supercomputer center such as NERSC, since without this focus, the rationale for acquiring and operating a large tightly-coupled computer system is weakened. Thus the ESP test includes two "full configuration jobs", with partition sizes equal to the number of available processors. The run rules for the ESP test specify that upon submission, the full configuration jobs must be run before any further jobs are launched. The first full configuration job can only be submitted after 10% of the theoretical minimum time has elapsed such that it is non-trivial to schedule. Similarly, the second full configuration job must complete within 90% of the test and is not simply the last job to be launched. The requirement to run these two full configuration jobs is a difficult test for a scheduler, but it is nonetheless a common scenario in capability environments.

Large systems typically require a great deal of system administration to maintain and improve their operation. These activities often require a system outage, either scheduled or unscheduled, and the time required for shutting down and restarting the system can significantly impact the overall system utilization. For this reason, the ESP test includes a shutdown-reboot cycle, which is required to start immediately after the completion of the first full configuration job. the utilization efficiency can be computed as

$$\mathbf{E = \Sigma_i\ (p_i * t_i)/[P^*(T + S)]}$$

The jobs in the ESP suite are grouped into three blocks, and the order of submission is determined from a reproducible pseudo-random sequence. The total number of CPUs requested in the first block is at least twice the available processors and the number of CPUs in the second block at least equal to the available processors. The remaining jobs constitute the third block. The first block is submitted at the start with the second and third blocks submitted 10 and 20 minutes thereafter, respectively. This structure was designed to forestall artificially configured queues specific to this test and, at the same time, provide sufficient queued work to allow flexibility in scheduling. No manual intervention is permitted once the test has been initiated.

We consider it important that the ESP test be representative of the user workload, so we designed the distribution of job sizes and run times in the ESP suite to roughly match the distribution of jobs running on NERSC production systems (except that the ESP run times were scaled down so that the test could be run in a more reasonable elapsed time). Tables 2 and 3 show some of this data, each set taken from a recent month's accounting records.

The applications in the ESP job mix originate from our user community and are used in production computing. Furthermore, the job mix profile was designed to span the diverse scientific areas of research amongst our users. Attention was also paid to diversify computational characteristics such as the amount of disk I/O and memory usage. For each class, an application and problem set was selected to satisfy the time and partition size constraints. The number of instances (Count) of each application/problem was adjusted such that aggregate CPU-hours reflected the workload profile. Table 1 lists the final job mix for the ESP benchmark with the elapsed times for each job on the T3E and SP.

Application	**Discipline**	**Size**	**Count**	**T3E**	**SP**
gfft	Large-FFT	512	2	30.5	255.6
md	Biology	8	4	1208.0	1144.9
md		24	3	602.7	583.3
nqclarge	Chemistry	8	2	8788.0	5274.9
nqclarge		16	5	5879.6	2870.8
paratec	Material-Science	256	1	746.9	1371.0
qcdsmall	Nuclear-Physics	128	1	1155.0	503.3
qcdsmall		256	1	591.0	342.4
scf	Chemistry	32	7	3461.1	1136.2
scf		64	10	1751.9	646.4
scfdirect	Chemistry	64	7	5768.9	1811.7
scfdirect		81	2	4578.0	1589.1
superlu	Linear-Algebra	8	15	288.3	361.2
tlbebig	Fusion	16	2	2684.5	2058.8
tlbebig		32	6	1358.3	1027.0
tlbebig		49	5	912.9	729.4
tlbebig		64	8	685.8	568.7
tlbebig		128	1	350.0	350.7

Table 1: ESP Application Job Mix

6. Data from the ESP Test Runs

Two test runs were completed on the T3E[1] and one run on the IBM SP. In both T3E cases, a separate queue was created for full configuration jobs. The full configuration jobs can thus be launched immediately on submission independent of the queue of general jobs. Process migration/compaction was also enabled for both runs. In the first run, labeled *Swa*p, the system was oversubscribed by two and gang-scheduled with a time-slice of 20 minutes using standard system software. A single NQS queue was used for the general job mix. In the second run, labeled *NoSwa*p, the system was not oversubscribed. Each job ran uninterrupted until completion. Six queues for different maximum partition sizes; 256, 128, 64, 32, 16, with decreasing priority were used.

[1] The runs were done the NERSC T3E-900, with 696 900 Mhz Alphas EV56 CPUs. Each CPU has 256 MB of memory and the system has a total of 2.5 Terabytes of local disk.

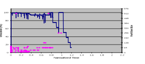

Figure 1: T3E Chronology with *Swap*

Figures 1, 2, 3 and 4 show the details of the three runs where the instantaneous utilization is plotted against time and the time axis has been rescaled by the theoretical minimum time. Additionally, the start time for each job is indicated by an impulse where the height equals the partition size.

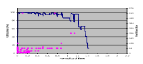

Figure 2: T3E Chronology with *NoSwap*

On the SP[2], two classes (queues) were created in *Loadleveller*; a general class for all jobs and a special high priority class for the full configuration jobs. It is not possible to *selectively* backfill with *Loadleveller*. Our preliminary runs shown in Figure 4, indicated that backfill would defer launching of the full configuration job until the end of the test. This would clearly violate the intent of the test. Backfill was implicitly disabled by

[2] The SP system is 604 nodes of 2 CPU SMPs. The CPUs are "Winterhawk 1" CPUs – which is a Power3 PCPU running at 200 MHz. Each node has 1 GB of memory and is connect with IBM's TBMX-3 switch.

assigning large wallclock times (several times greater than the complete test) to all jobs. Thus *Loadleveller* was reduced to a strictly FCFS strategy. The resulting run is shown in Figure 3.

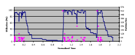

Figure 3: SP Chronology

On submission of the full configuration jobs, a considerable amount of time was spent waiting for running jobs to complete. This is evident in Figure 3, which shows two large regions where the instantaneous utilization drops to a very low value. The time lag to run preferential jobs is indicative of the difficulty in changing modes of operation on the SP. This is important for sites that routinely change system characteristics, for example between interactive and batch or between small and large partitions. The best remedy would be to either checkpoint or dynamically swap out running jobs.

As seen in Figure 1, the BFF mechanism on the T3E deferred large partition jobs (≤ 128) until the end. Consequently, at the end of the test there were large gaps that could not be filled by small jobs. On the SP, a FCFS strategy was indirectly enforced which can be seen illustrated in Figure 3 where the distribution of job start times is unrelated to partition size. It is evident from Figures 1 and 2 that a significant loss of efficiency on the T3E is incurred at the tail end of the test. In an operational setting, however, there are usually more jobs to launch. That is, the fact the ESP test is finite poses a problem since we are interested in a continual utilization given a hypothetical infinite number of queued jobs. Suggested solutions to this dilemma have proven to be awkward and require manual intervention.

The distribution of start times is qualitatively similar between the *Swap* and *NoSwap* runs on the T3E although the queue set up was differently. In the second run, increasingly higher priorities were deliberately assigned to larger partition queues in an attempt to mitigate starvation. However, shortly after the start, it is unlikely that a large pool of idle processors would become coincidently available. In this scenario, the pattern of job submission reverts back to BFF and the queue set up has little impact. On the other hand, there is considerable difference in efficiency between the two T3E runs. This is attributed to the overhead of swapping which is significant when the oversubscribed processes cannot simultaneously fit in memory and process images must be written to disk

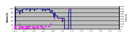

Figure 4: SP Chronology with backfill

Nonetheless, it is noteworthy that the SP system, in spite of its lower system-level efficiency, completed the test in less time due to its higher computational rate.

[Note to Reviewers – by the time the final paper is submitted, data for two new sections will be available. First, a revised test, designed to be more scalable and more portable will run. Results from at least 3 systems will be available, possibly more. Second, the new test results will be run on the NERSC 2,532 processor SP, with new operating system functionality.]

7. Summary of Results

The results of the ESP test for the T3E and the SP are summarized in Table 2. Two efficiency measurements, with and without the shutdown/reboot time factored in, are reported as well.

	T3E Swap	T3E NoSwap	SP
Available processors	512	512	512
Job mix work (CPU-sec.)	7437860	7437860	3715861
Elapsed Time (sec.)	20736	17327	14999
Shutdown/reboot (sec.)	2100	2100	5400
E - Efficiency	**64%**	**70%**	**36%**
$E^{'}$ - Efficiency (w/o reboot)	70%	84%	48%

Table 2: ESP Results

These results show that the T3E has significantly higher utilization efficiency than the SP within. This is mainly due to the lack of an effective mechanism to change operational modes in a reasonably short time period, such as is necessary to immediately launch full configuration jobs.

9. Validation

Is it important to validate the results of any benchmark test with real data. Is it now possible to do it by reviewing the utilization data on the T3E and the SP for the actual workload the ESP test was first designed to mimic. Figure 5 shows the T3E utilization for a period of 3 years. Figure 6 shows the SP utilization for 9 months.

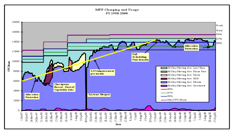

Figure 5: Actual Utilization of the NERSC T3E over a 3 year period. The date blue is the 30 day moving average of the CPU used by user applications

The T3E usage increased with the introduction and improvement of system function. The utilization in the first 30 days, was 57%. Utilization was limited by the fact the T3E needs to assign jobs CPUs in a contiguous block based on logical node numbers which are assigned to physical nodes at boot time. Contiguous logical nodes improves communications within the job but means there is the potential for fragmentation of unused processors that are left idle since there are no jobs of the size that can run. Indeed, is it possible that two long running small jobs (4 or 8 processors), poorly positioned, could prevent any jobs more than 64 CPUs from starting. It is important to note that during this period, in order to develop and test of the more advanced scheduler functions documented in [2], the T3E was taken out of service and rebooted at least two times a week – for 6 hours each time. The introduction of the job migration facility, followed by the deployment of checkpoint restart, allowed more efficient system operation while running full configuration jobs every night. This period, shown in the red box on Figure 5, which is the same level of function used for the ESP test runs, shows roughly a 70% utilization, very similar to the ESP test ratings in Table 2.

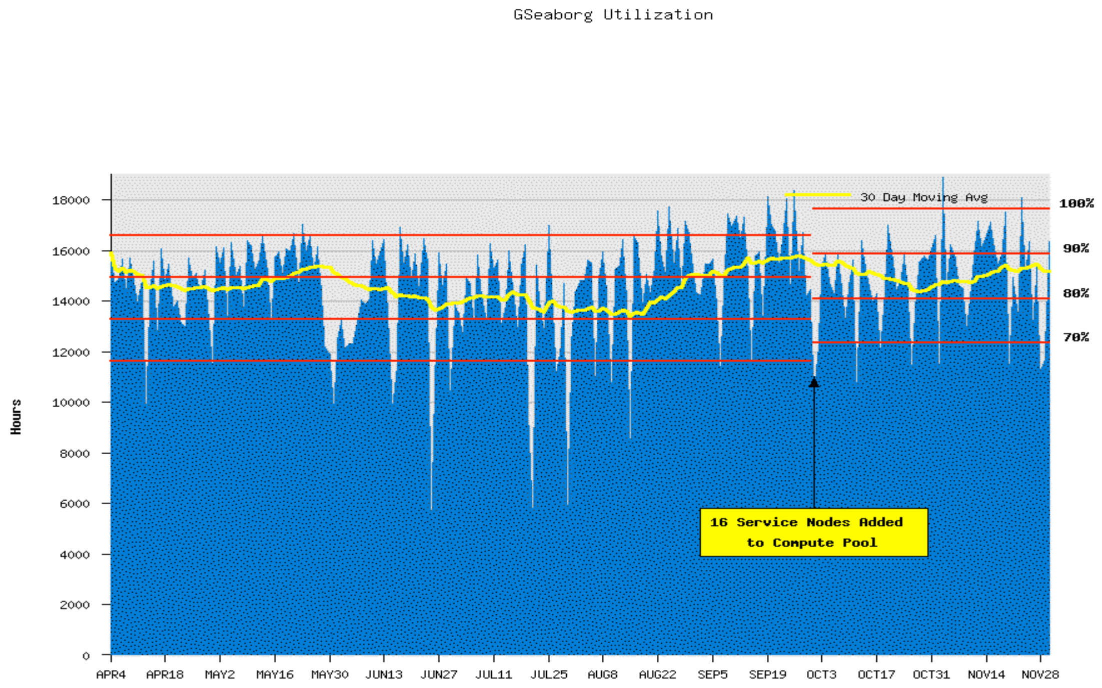

Figure 6: SP Utilization over a 7 month period, on a daily basis. The yellow line is the 30 day moving average.

The utilization on the SP started out higher than on the T3E since the backfill function was immediately available jobs can be assigned to any set of nodes to a job (although the scheduler taking locality of nodes into account would actually be a useful feature to reduce communication latency). With backfill on, the usage is well over the expected utilization indicated by the Efficiency Rating predicted by the ESP without backfill.

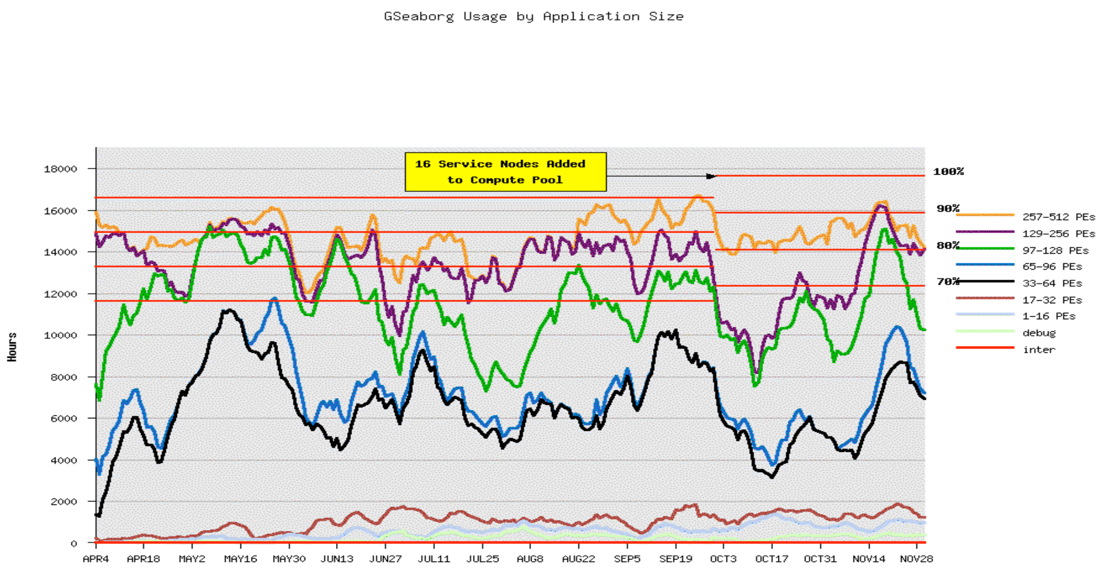

Figure 7: SP Workload – Cumulative CPU Time by Job Size

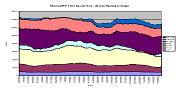

Figure 8: T3E Workload – Cumulative CPU Time by Job Size

Figures 7 and 8 show that the actual workloads on the T3E and the SP have several common characteristics, which is not surprising given they are similar in size and capability, and support the same user community. The first observation is that each system runs a number of jobs that are full configuration. In Figure 7, for the last two months of the period, 10% of the CPU time was used by full configuration (512 CPU) jobs. A similar result is shown in Figure 8 for the T3E. The other observation is that the vast majority of the CPU time on both systems goes to jobs of substantial size. In the T3E case, more than 50% of the time is used by jobs 128 CPUs or more – which is 1/4 of the system.

Even with backfill on in the production system, Figure 7 shows that eventually large jobs do run, and make up a significant part of the SP workload. What is not shown in a graph is that the length of time large wait for service is much longer on the SP, because the system has to age large jobs a very long time to get processors assigned, or alternatively manual intervention is used.

Table 3 compares the SP runs with and without backfill. As stated above, backfill violates the test rules because the full configuration jobs are not processed in a timely manner. Nonetheless, as a validation data point that the ESP Efficiency Rating is indicator of the utilization a system will be able to support, we look at what the ratings would be with backfill and without reboot, and see in E = 84%. This is very close to the observed utilization while running under the exact same operational parameters on the SP.

In both cases, the Efficiency Rating is dramatically reduced due to the extensive time it takes to do a reboot on the SP. While not a proven relationship, it may not be just coincidence that the difference between the 6% decrease on the T3E with *Swap* and the 32% decrease on the SP with backfill correlates with system managers' perceptions of the effectiveness of the system administrative functions on the two systems.

	SP without backfill	SP with backfill But violates test parameters
Available processors	512	512
Job mix work (CPU-sec.)	3715861	3715861
Elapsed Time (sec.)	14999	8633
Shutdown/reboot (sec.)	5400	5400
E - Efficiency	**36%**	**52%**
E' - Efficiency (w/o reboot)	48%	84%

Table 3: ESP Results

10. Conclusions and Future Plans

We described a new system utilization benchmark, which we have successfully run on two highly parallel production supercomputers. This test has provided quantitative data on the utilization and scheduling efficiency of these systems, as well as useful insights on how to better manage them. The most important conclusion is that certain system functionalities, such as checkpoint/restart, swapping and migration, are critical for the highly efficient operation of large systems. This test is has also been adopted by at least two other major HPC sites, including the ASCI project at LLNL, and NCAR. IT is under consideration for use at more sites as well.

We have summarized here the results that we have obtained so far. We are improving the portability of the test, but replacing the application benchmarks that are not freely distributed with other codes that are complete sharable. We are also simplifying the test to use 3 or 4 codes rather then the 8. Another effort is to be able to scale the test to more or less CPUs, and still have a comparable set of data. The most immediate implication is that we are working to run this test on a 2,532 CPU SP system are NERSC, where there are 2,132 CPUs in the parallel computation pool.

This test will also be used to assess IBM's new checkpoint restart implementation this summer. We plan to introduce some variations on the ESP test, for example a variation with a job mix that provides *a priori* runtimes to schedulers, so that they may exploit this information in achieving higher utilization.. Other areas of possible exploration include adding a formal I/O test to the workload and a way to model interactive and debugging jobs

Finally, we are working to make package the test in a freely available software archive, together with facilities for simple installation and execution. In this way we hope that this test will be of use to other sites and will help spur both industry and research to improve system utilization on future systems.

11. Acknowledgments

This work was supported by the Office of Computational and Technology Research, Division of Mathematical, Information and Computational Sciences of the U.S. Department of Energy, under contract DE-AC03-76SF00098.

References

[1] David H. Bailey, et. al, "The NAS Parallel Benchmarks", Intl. Journal of Supercomputer Applications, vol. 5 (1991), pg. 66.
[2] Jay Blakeborough and Mike Welcome, "T3E Scheduling Update", 41st Cray User Group Conference Proc., 1999.
[3] Jack Dongarra, "Performance of Various Computers Using Standard Linear Algebra Software in a Fortran Environment". Available from `http://www.netlib.org/benchmarks/performance.ps`.
[4] D. G. Feitelson, "A Critique of ESP", Workshop in Job Scheduling Strategies for Parallel Processing, D. G. Feitelson and L. Rudolph, ed., Springer-Verlag/IEEE, 2000, pg. 69-73.
[5] D. G. Feitelson and M. A. Jette, "Improved Utilization and Responsiveness with Gang Scheduling", Workshop in Job Scheduling Strategies for Parallel Processing, D. G. Feitelson and L. Rudolph, ed., Springer-Verlag/IEEE, 1997, pg. 238.
[6] Morris A. Jette, "Performance Characteristics of Gang Scheduling in Muliprogrammed Environments", Proc. of Supercomputing '97, IEEE, 1997.
[7] W. Leinberg, G. Karypis and V. Kumar, "Job Scheduling in the presence of Mulitple Resource Requirements", CS Dept. TR 99-925, University of Minnesota, 1999.
[8] Horst Simon and William Kramer and Robert Lucas, "Building the Teraflops/Petabytes Production Supercomputing Center", Fifth International Euro-Par Conference Proc., 1999, pg. 61.

[9] D. Talby and D. G. Feitelson, "Supporting Priorities and Improving Utilization of the IBM SP2 Scheduler using Slack-Based Backfilling", 13th International Parallel Processing Symposium, IEEE, 1999, pg. 513.
[10] B. Ujfalussy, Xindong Wang, Xioguang Zhang, D. M. C. Nicholson, W. A. Shelton, G. M. Stocks, A. Canning, Yang Wang, and B. L. Gyorffy, "High Performance First Principles Method for Comlex Magnetic Properties", SC'98 CD-ROM Proceedings, Nov. 1998.
[11] Adrian Wong, Leonid Oliker, William Kramer, Teresa Kaltz and David Bailey, "System Utilization Benchmark on the Cray T3E and IBM SP", Workshop in Job Scheduling Strategies for Parallel Processing, D. G. Feitelson and L. Rudolph, ed., Springer-Verlag/IEEE, pg. 56-67, IPDPS 2000 Workshop, Cancun Mexico, May 2000. Also available from `http://www.nersc.gov/~dhbailey`.
[12] Adrian Wong, Leonid Oliker, William Kramer, Teresa Kaltz and David Bailey, "Evaluating System Effectiveness in High Performance Computing Systems", 1999. Available from `http://www.nersc.gov/~dhbailey`.
[13] Dmitry Zotkin and Peter J. Keleher, "Job-Length Estimation and Performance in Backfilling Schedulers", 8th High Performance Distributed Computing Conference, IEEE, 1999.