Top500 Versus Sustained Performance – the Top Problems with the TOP500 List – And What to Do About Them

William Kramer
National Center for Supercomputing Applications
University of Illinois
1205 W Clark Street
Urbana, Illinois 61821
217 333-6260
wtkramer@illinois.edu

ABSTRACT

A popular U.S. talk show host uses "top 10" lists to critique events and culture every night. Our HPC industry is captivated by another list, the TOP500 list, as a way to track HPC systems' performance based on FLOPS/S assessed by a single, long-lived benchmark—Linpack. The TOP500 list has grown in influence because of its value as a marketing tool. It simplistically, but unrealistically, describes performance of HPC systems. The proponents have advocated for the TOP500 list for different reasons at different times. This paper critiques the Top 10 problems with the TOP500 list and provides suggestions on how to correct those shortcomings. It discusses why the TOP500 list is limiting the impact of HPC systems on real problems and other metrics that may be more meaningful and useful to represent the real effectiveness and value of HPC systems.

Categories and Subject Descriptors

C.4 [**Computer Systems Organization**]: Performance Of Systems - *Measurement techniques; Reliability, availability, and serviceability*

General Terms

Measurement, Performance, Economics.

Keywords

Performance, Performance, Linpack, Top500, PERCU, Benchmarks, System Evaluation, Supercomputing, HPC.

1. BACKGROUND

The TOP500 list was introduced almost 20 years ago as a way of classifying computer performance. The list is based on the floating point computational performance assessed by a single benchmark, Linpack. It has grown in influence because it has become a marketing tool and a simplistically described measurement of performance for high-performance computers. The proponents of the list advocate its importance. At first it was presented as an important distinguisher of the performance of different systems and an indicator of the amount of scientific and/or engineering work particular systems could accomplish. In later years, the advocates have admitted its shortcomings, but shifted to extolling the historic value of the list and its ability to project future trends in high-performance computing.

1.1 The Linpack benchmark

The widely discussed Linpack benchmark [1] that is used to determine the TOP500 List [2][3] is a single test that solves Ax=b where matrix A is a dense linear matrix. Linpack uses Gaussian elimination with partial pivoting. For matrix A, that is size M x M, Linpack requires $2/3\ M^3 + 2M^2$ operations. The latest Linpack benchmark implementation, HPL [4], can run on any number of processors, but uses weak scaling to achieve higher performance. In order to provide enough work to each processor, the size of the A matrix has to increase, not only taking more memory, but increasing the wall clock time of the run. Linpack is the only metric the TOP500 uses to assess system performance.

2. THE TOP 10 PROBLEMS WITH THE TOP500 APPROACH

While initially providing interesting information, for a long time, unfortunately, the Top500 list has done more harm than good to the high-performance computing community for the following reasons, which are presented in reverse order of importance as David Letterman does on his Late Show.

2.1 Number 10: The Linpack benchmark serves only one of the four purposes of a good benchmark.

Effective benchmark tests should serve four purposes in one implementation. Benchmark tests are representative approximations of the real work a computer system can accomplish. In other words, benchmark tests estimate the potential of computer systems to solve a set of real world problems. Benchmark tests are made up of computer programs (codes) and one or more input data sets that state a problem the program is to solve. One set of computer codes can exhibit different behavior based on the problem being solved and the parameters involved. Each purpose of the benchmark tests influences the selection and the characteristics of the benchmarks as well. The four purposes of benchmarks are:

1. Evaluation and/or selection of a system from among its competitors.
2. Validating that the selected system works as expected once it is built and/or arrives at a site.

3. Assuring the system performance stays as expected throughout the systems lifetime (e.g. after upgrades, changes, and regular use.)
4. Helping guide future system designs.

The TOP500 Linpack test, as the Top500 requires it to be run, does not serve any of the above purposes well. In fact the Top500 test is not able to serve purpose #3 at all because of the long run times required. For reasons discussed in the following sections, Linpack is seldom very effective in distinguishing between systems either. This is because most modern processors are designed to support cache-friendly, dense matrix algorithms. Indeed, the efficiency of all the systems on the most recent TOP500 list only varied by a modest amount that can be examined with other less intrusive tests.

In a limited way, Linpack can validate whether a system meets performance expectations at time of arrival (purpose #2), but is not efficient in doing so. The limitation of Linpack for this purpose is due to it correlating very well with peak performance. But there are many (even most) applications whose performance does not correlate with Linpack. Further, running Linpack at scale takes very long run times to achieve expected performance, as discussed below.

Linpack also has little to add to future architectural improvements, except possibly as a regression test to insure architectures continue to do well with dense, cache-friendly computations. But this regression test can be achieved by much simpler and less time-consuming tests, such as DGEMM running on a node. Since Linpack only partially addresses purposes #1 and #2, and does not address purposes #3 or #4, it is a not a useful indicator of how well as system is able to process work.

2.2 Number 9: The TOP500 list disenfranchises many important application areas.

Linpack solves a set of dense linear equations, yet many if not most applications involve algorithms that are not well represented by dense matrix solvers. Figure 1 shows a collection of scientific disciplines and the algorithms that were in use by informed applications in science domains at the National Energy Research Scientific Computing Center between 1996 and 2008 [5]. Table 2 is a similar summary of analysis of the applications used by 26 science teams with initial allocations on the NSF Blue Waters system [6]. The methods, or motifs, across the top of the charts directly relate to system architectural features. In assessing these and similar requirements analysis, well-balanced systems are required for most science applications.

These tables demonstrate that high-end computation involves much more than a single method, even within a single science domain. While dense linear algebra plays a role in problem solving, that role has significantly diminished as new computational methods evolve. Methods such as sparse linear algebra and adaptive mesh refinement have become increasingly important components in the computational scientist's toolkit but do not correlate to Linpack performance. Furthermore, computer architectures have changed over the past 15 years. When Linpack was introduced, vector processors and high-bandwidth memory systems dominated the HPC world. As shown in the table, dense methods are only one of many algorithmic methods in use today. All science disciplines use multiple methods, if not in one application, then in different applications to pursue science goals in each area.

Science areas	Multi-physics, Multi-scale	Dense linear algebra (DLA)	Sparse linear algebra (SLA)	Spectral Methods (FFT)s (SM-FFT)	N-Body Methods (N-Body)	Structured Grids (S-Grids)	Unstructured Grids (U-Grids)	Data Intensive
Nanoscience	X	X	X	X	X	X		
Chemistry	X	X	X	X	X			
Fusion	X	X	X			X	X	X
Climate	X		X	X		X	X	X
Combustion	X		X			X	X	X
Astrophysics	X	X	X	X	X	X	X	X
Biology	X	X					X	X
Nuclear		X	X		X			X
System Balance Implications	General Purpose balanced System	High Speed CPU, High Flop/s rate	High Performance Memory	High Interconnect Bisection bandwidth	High Performance Memory	High Speed CPU, High Flop/s rate	Irregular Data and Control Flow	High Storage and Network bandwidth

Table 1. This table indicated significant algorithmic methods in use for different science areas. System architectural features are also indicated according the algorithmic method. The TOP500 metric corresponds only to one architectural feature since it is dominated by dense linear algebra.

Science Area	Number of Teams	Codes	Structured Grids	Unstructured Grids	Dense Matrix	Sparse Matrix	N-Body	Monte Carlo	FFT	Significant I/O
Climate and Weather	3	CESM, GCRM, CM1, HOMME	X	X		X		X		
Plasmas/ Magnetosphere	2	H3D(M), OSIRIS, Magtail/UPIC	X				X		X	X
Stellar Atmospheres and Supernovae	2	PPM, MAESTRO, CASTRO, SEDONA	X			X		X		X
Cosmology	2	Enzo, pGADGET	X			X	X			
Combustion/ Turbulence	1	PSDNS	X						X	
General Relativity	2	Cactus, Harm3D, LazEV	X			X				
Molecular Dynamics	4	AMBER, Gromacs, NAMD, LAMMPS			X		X		X	
Quantum Chemistry	2	SIAL, GAMESS, NWChem			X	X	X	X		X
Material Science	3	NEMOS, OMEN, GW, QMCPACK			X	X	X	X		
Earthquakes/ Seismology	2	AWP-ODC, HERCULES, PLSQR, SPECFEM3D	X	X			X			X
Quantum Chromo Dynamics	1	Chroma, MILD, USQCD	X		X	X	X		X	
Social Networks	1	EPISIMDEMICS								
Evolution	1	Eve								
Computer Science	1			X	X	X			X	X

Table 2. Table showing the analysis of codes used by 26 science teams with initial allocations on the Blue Waters sustained petascale system. Note many teams use more than one application to achieve their science goals.

The TOP500 metric only deals with dense linear systems and gives no insight as to how well a system works for most of the algorithmic methods (and hence applications) in use today.

2.3 Number 8: There is no relationship between the TOP500 ranking and user productivity or system usability for real application workloads.

The TOP500 list does not indicate whether the systems listed are usable or productive. In a number of cases, systems have been listed while being assembled at factories rather than when installed at their host facilities, leaving a gap of months between when a system is listed and when it is usable by scientists and engineers. Even if the system is installed before being listed, it does not mean the system is usable or productive. To demonstrate the disconnect between the ranking and how effective a system is for a real workload, Figure 1 shows a comparison of the Linpack rating for systems procured and installed at NERSC and the NERSC Sustained System Performance [7] metrics for a 14-year period. The SSP has been shown to be a reasonable representation of the actual NERSC workload that covers almost all science disciplines and hundreds of applications [8]. As shown in Figure 1, there is an increasing discrepancy between the actual sustained performance

and both the peak and Linpack performance across a number of systems and architectures.

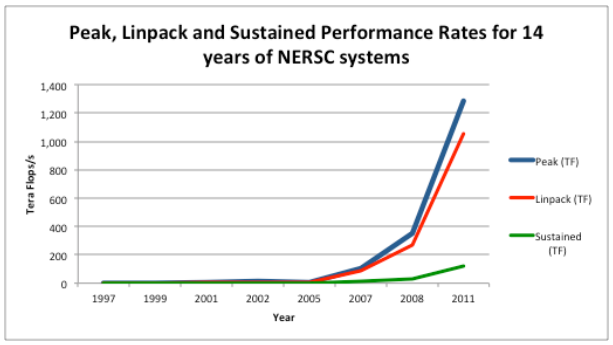

Figure 1. Comparison of NERSC SSP to Linpack performance over a 14-year period shows Linpack greatly misrepresents the real performance one can expect from a computer system.

In another view of the same data, Figure 2 compares the reported Linpack performance with the reported average sustained application performance using between six and eight applications and multiple input sets. The observer can see that sustained performance bears little correlation to Linpack performance.

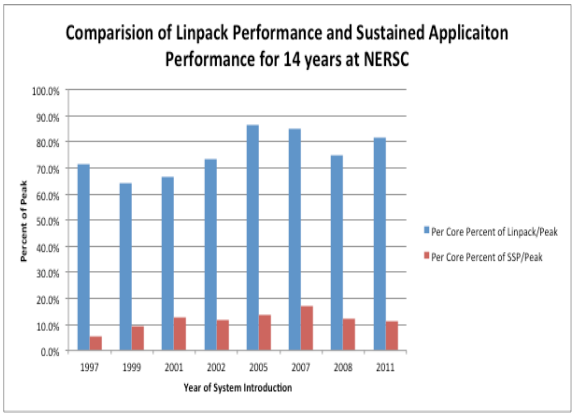

Figure 2. Comparison of Linpack performance and average sustained application performance for 8 systems over 14 years at NERSC

The results seen for NERSC-5 are aligned with the results seen using the similar versions of the tools for the DOD Modernization Program TI-05 evaluation and acquisition [9] which reports:

- simple tests are inadequate for predicting or assessing application performance on systems, with Linpack being the poorest simple test studied;
- combining simple tests with optimized weights also is inadequate for meaningful application performance prediction;
- convolving application traces with metrics derived from a specific set of simple tests (the PMaC methodology) can predict performance of applications to about 80% accuracy for the same system (no comment is made for similar but not identical systems); and
- acquiring the application-specific traces is "painful." It is noted that PMAC has since abandoned this version of the framework and has moved to a new framework for performance prediction [10] that recently reported predictions with 90% accuracy on a limited set of applications for the same system as was instrumented.

2.4 Number 7: The Linpack performance test is dominated by single-core, dense linear algebra peak performance.

Over the 16 years of the lists being reported, Linpack results have tracked the peak performance of the systems with only a few exceptions. Figure 3 shows the ratio of the reported Linpack performance (R_{max}) to peak performance (R_{peak}) for the #1 system for each list. It also shows the average ratio across all years.

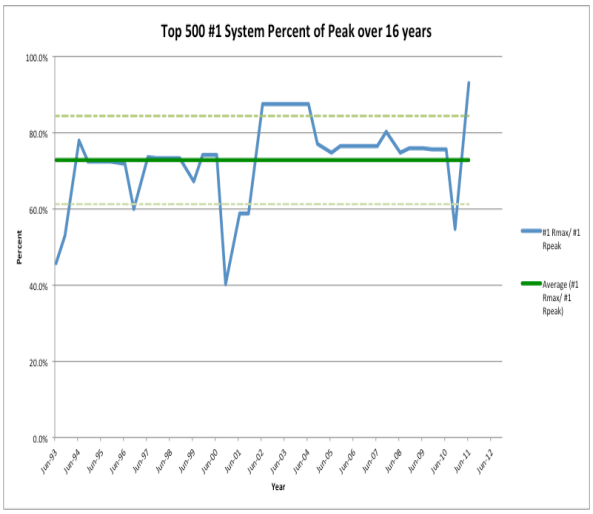

Figure 3. The 16-year comparison of system peak speed (R_{peak}) to measured Linpack rates (R_{max}) for the number 1 system on the list. The figure shows only a few times when there are any significant deviations, which are mostly explained by other causes.

There are only a few places where the ratios show a move away from the long-term average. In the first two lists, this difference is probably sites starting to pay attention to the implementation of their submitted times once they realized others would notice. The next difference in June 1996 is due to a new system that may have had limited tuning time. The next major difference in June 1999 is informative. It is the ASCI Red Storm system, which was also number 1 on the November 1998 list with improved R_{max}, probably due to additional tuning. The changes in November 2000 and June 2001 are both the ASCI White system at Lawrence Livermore National Laboratory, but with a percent of peak improvement of almost 50% in between the two lists. In this case, the low initial performance was good enough to claim the top spot on the list in November 2000 when the system was not complete, but the system's performance got much better before it went into service. Obviously the improvement in 2002 was the Earth Simulator, which was not only a case of vector vs. RISC system but also had very significantly more funding than any other system ever on the list. The improvement in June 2006 was the Los Alamos National Laboratory's Roadrunner system with a specialized, but limited use, Cell architecture, and November 2010 #1 on the list was again a specialized system with accelerators.

2.5 Number 6: The TOP500 metric has not kept up with changing algorithmic methods.

Today, computational scientists have to deal with massive parallelism using computational nodes that are often under-provisioned with memory and memory bandwidth. Therefore,

computational methods have evolved significantly since Linpack represented the majority of methods used to solve problems.

The effectiveness of a metric for assessing delivered performance is founded on its accurate mapping to the target workload. A static benchmark suite will eventually fail to provide an accurate means for assessing systems. Several examples, including Linpack, show that over time, fixed benchmarks become less of a discriminating factor in predicting application workload performance. This is because once a simple benchmark gains traction in the community; system designers customize their designs to do well on that benchmark. The Livermore Loops, SPEC [11], Linpack, NAS Parallel Benchmarks (NPB) [12], etc. all have had this issue. It is clear Linpack now tracks peak performance in the large majority of cases.

Simon and Strohmaier [13] showed, through statistical correlation analysis, that within two Moore's Law generations of technology and despite the expansion of problem sizes, only three of the eight NPBs remained statistically significant distinguishers of system performance. This was due to system designers making systems that responded more favorably to the widely used benchmark tests with hardware and software improvements.

Thus, long-lived benchmarks should not be a goal except possibly as regression tests to make sure improvements they generate stay in the design scope. There must be a constant introduction/validation of the "primary" tests that will drive the features for the future, and a constant "retirement" of the benchmarks that are no longer strong discriminators. On the other hand, consistency of methodology and overlapping of benchmark generations are useful so there can be comparison across generations of systems. Consequently, the metrics that continue to evolve to stay current representative with of current workloads and future trends by changing both the application mix and the problem sets in a coordinated manner have shown to be useful over longer periods. The NAS Parallel Benchmarks partially addressed this by providing multiple size versions (Class A thru E) so as systems scaled to higher core counts there was some relevancy. Other tests, such as the NERSC Sustained System Performance test, SPEC, and the DOD Technology Insertion [14] benchmarks do a reasonable job of introducing new applications and methods, while at the same time correlating new versions to past versions. It is possible to compare the different measures so long-running trends can be tracked.

This lack of relevance to current methods will get worse for Linpack as we move from petascale to exascale. Linpack, and therefore the Top500, is highly parallelizable, particularly in the weak scaling mode, due to the static properties of the problem. So using this relatively simplistic measure may overlook or minimize issues with the upcoming "massive parallelism" systems to be developed in the next decades.

2.6 Number 5: The TOP500 measure takes too long to run and does not represent strong scaling.

The latest Linpack benchmark implementation, High Performance Linpack (HPL), can run on any number of processors, but in order to provide enough work to each processor, the size of the A matrix has to increase, not only taking more memory, but increasing the wall clock time of the run more than linearly. This is *memory-constrained scaling* "which is attractive to vendors because such speed ups are high" [15]. In order to keep scaling high, as much work per processor as possible has to be loaded into the system's memory. The amount of memory used grows at $O(N^2)$; the run time to do the work grows at $O(N^3)$. So for a system such as the NERSC Cray XT-4 with ~39,000 cores and ~80 TB of aggregate memory, a single run of Linpack may take 17-20 hours on the entire system. On large systems today, Linpack takes multiple days, and multiple tuning runs are typically done to achieve high list rankings. While few sites admit the time used for Top500 results, informal statements indicate it often takes weeks of dedicated system time for a submitted ranking run.

To address this specific issue, Dongarra recently suggested modifications to the TOP500 run rules to allow entries that do not complete the entire calculation [16]. This will make the run time needed for an entry shorter, but also have impact on the expressed performance. Furthermore, it lowers the probability the Linpack run will detect any abnormality in the system (e.g. incorrect answers) but is likely to allow people to post results for systems above tens of Petaflops.

2.7 Number 4: The TOP500 is dominated by who has the most money to spend–not what system is the best.

The TOP500 list's proponents represent it as an important indicator that is useful for historical purposes and its ability to predict the future of HPC technology. However, the dominant factor for list performance is who has the most funding to spend, followed by Moore's Law. Figure 4 is the typical historical graphic showing the Top500 number 1 and number 500 systems, with a new line showing known U.S. government spending in HPC over the same time period [17]. The funding is adjusted for inflation as reported by the Consumer Price Index [18]. While the speeds documented in the list are indeed increasing, it is clear that a major contributor to the increased speeds reported is the increased funding available for HPC systems.

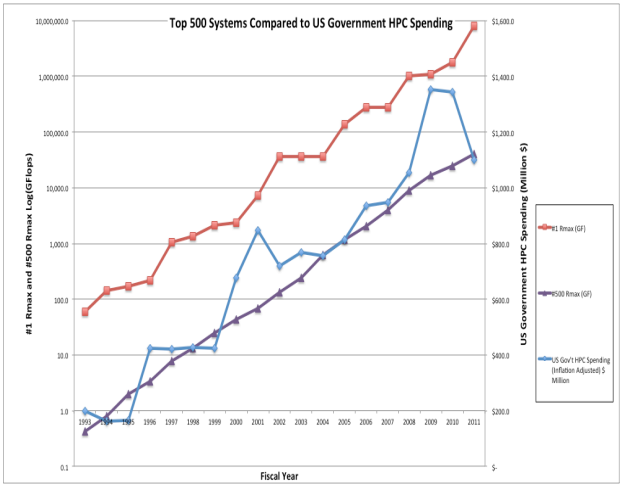

Figure 4. Linpack reported results compared with U.S. government funding for the 19-year period

This chart shows that a major contributing factor to the increases in performance on the Top500 is increases increased U.S. government funding. The U.S. government funding is representative of funding increases for other governments. For example, it is well documented that the Japanese government devoted significant resources to HPC over this time period as well, at times surpassing the U.S. investments. Indeed, list sponsors sometimes tout the fact that the list increases faster than Moore's Law [19] would predict, but this figure clearly shows that funding

increases contribute to that, and in fact, the list is increasing more slowly than Moore's Law acceleration once the funding is factored in.

2.8 Number 3: The TOP500 provides little historical value.

Recently, while people acknowledge the lack of correlation with real workloads, the Top500 list is defended as having historical value to show changes over time for HPC systems. While there may be some historical value for such information, such as the number of systems by vendor or geographic region, most of the historical insights can come from other information that is more meaningful. Since the list convolves multiple factors, predominately how much money is available for HPC purchases and Moore's law, the list does nothing to resolve the influences of other respective components. Both of the two main factors are clearly available from other sources.

Another attribute of the list that can lead to historic misinformation is that there is no correlation between being able to list a system on the Top500 and having it available to perform its mission of running science and engineering applications. Systems have been known to be listed even though they are still in their factory bring up and not close to being shipped and installed. Other systems listed have been engineering prototypes that are never intended for production use. This gives the appearance of an acceleration of performance that is not reality.

2.9 Number 2: The TOP500 encourages organizations to make poor choices.

There have been notable examples of systems being ill-configured in order to increase the ranking on the Top500 list, leaving organizations with systems that are imbalanced and less efficient for their application workloads. Repeatedly, storage capacity and bandwidth and memory capacity are sacrificed in order to increase the number of peak (and therefore Linpack) flops in a system. In these cases–which include some very large systems–it is often the case that the types of applications that can be run are limited.

Also, the goal of listing a system can be so important to organizations that they may actually defer any use of the system. The modest example is that runs of the Linpack code take days to complete and often require multiple trials to tune–costing sites weeks of non-productive time. But occasionally more significant issues arise, such as occurred with the "System X" listed as number 3 on the November 2003 list[1]. After publication, it was disclosed that the system was unable to run any other application because it had memory without error-correcting hardware and could not be trusted to produce correct answers. Furthermore, the system had numerous Linpack runs, not for tuning, but to get a single correct result. System X was completely disconnected and almost all parts replaced with entirely different parts that took a year to put into place. Hence, System X did no useful work for approximately one year but still holds its place as the number 3 highest-performing system at the time to this day. The desire to gain a notable list rank meant the organization had no recourse to pursue their science and engineering work for this time.

[1] Private communication with Dr. Wu Feng during presentation at LBNL, January 2004.

2.10 Number 1: The TOP500 gives no indication of the cost of value of a system.

The Number 1 issue of the Top500 is that is gives no indication of the cost or value of the systems it lists. Hence, there is no way to compare different system architectures or implementations since the dominant factor in performance and list ranking is how much money was spent. While the exact cost of the Earth Simulator was never fully documented in public, indications are it cost substantially more than other contemporary systems. Indeed, using NEC list pricing for equivalent parts at the time, the Earth Simulator added up to well over $1 billion. The next-ranked systems cost five to 10 times less. Similar situations exist throughout the history of the list where high positions on the list can be "bought" if an organization has sufficient funding.

Value is the ratio of the amount of work a system can do to the cost of the system. Even if Linpack were an accurate measure of the amount of real work a system is capable of, without an expression of cost listed alongside the performance metric it is impossible to understand the relative value of the systems. Without the ability of assessing value, it is not meaningful to discuss comparisons of systems either within a particular list or across lists.

One defense used for not requiring any cost information is that different discounting is used and/or the actual cost is not possible to know for all systems at high accuracy. While it is unrealistic to have an exact cost figure for all systems, it is misleading to say that gathering information is impossible and hence should not be attempted. For example, many press releases about new systems, a cost figure is given, and many procurement documents provide the estimated available funds. Furthermore, an existence proof that gathering cost data is possible is in place because the original NAS parallel benchmark rules [20] were specific that a system would not be listed unless a cost estimate was provided for that system–and the NPBs were highly successful in capturing meaningful performance information for many years.

3. RECOMMENDATIONS FOR IMPROVEMENTS

Recommendations to improve the state of HPC system comparisons over time separate into two classes. *Evolutionary* improvements are meaningful improvements that can be made within the context of the current list structure that do not require major effort. *Revolutionary* improvements are major restructuring that will result in new methods to assess systems and community progress. These improvements address one or more of the issues discussed above.

3.1 Evolutionary Improvements

The most important, immediate improvements are listed below.

3.1.1 Require cost data for every system listed

Require all list submissions provide a system cost along with the Linpack run submission. The cost estimate can be flexible, as it was in the NAS Parallel Benchmark. It could be the actual cost paid or a cost estimated from pricing tables (e.g. U.S. government GSA contracts) or other methods. For the latter, it may be that large system units are not listed, but smaller ones are so there would be some proration. At the worst, a component-wise estimate can be done as discussed above for the Earth Simulator.

It is not necessary to have the cost data as precise as the performance run, but having even roughly comparable cost data

would begin to allow comparisons of value. Furthermore, it would provide additional historical relevance of the list since then the community has readily available information about changing investments in HPC.

This improvement is similar to the change in the list managers made recently to request (and possibly eventually require) energy consumption information associated with the performance information. The measurement method for energy consumption is not proscribed, but it is useful data to have. The positive experience with this (and the growth of new comparative metrics such as the Green 500 list) show some variation in costing calculations can be tolerated while still being very meaningful.

3.1.2 Do not allow a system to be listed until it is fully accepted and performing its mission

Often, one of the very first things a site does is run the Linpack test for submission to the list. A system may spend six to 12 months before it is in real production service. So the list gives an unrealistic perspective of how much computing capability is being delivered to the community.

It is more realistic to require that any ranked system actually be doing its intended work before listing the system. This could mean requiring the submitter to verify it is in production use or requiring some verification of meaningful scientific results being accomplished along with the data of the Linpack run.

3.1.3 Require a complete description for every system listed

Require sites to fully specify their system capacities and feeds. For example, the amount and speed of memory and the amount and speeds of the I/O subsystems should be recorded for all entries. This improvement would allow assessment of how well balanced a system is and would reflect the investment strategy for a system. It would also document how different types of components influence the performance results.

3.1.4 Move from weak scaling to strong scaling Linpack

The weak scaling approach of Linpack, essentially requiring more and more work to be performed to achieve a certain level of performance, makes the metric less meaningful. While in the past many applications did increase the amount of work they performed as system size and performance increased, that option is less practical as we move to the future. Systems are moving into a time when parallelism will dominate, but system bottlenecks such as memory capacity will not increase as quickly, making strong scaling dramatically more important than in the past. Hence a simple improvement is for the list to set levels of problem size that are independent of system scale. The NAS Parallel Benchmarks has introduced five classes of problems over time so that there is enough work in the problem set, and the Top500 criteria should follow a similar approach.

3.2 Revolutionary Improvements

Revolutionary approaches are to make major modifications and/or entirely replace the Top500 with a new list(s) that is much more realistically aligned with application performance.

3.2.1 Align the metric to best practices in benchmarking

There are attributes of any benchmark and metric that are generally agreed upon. Combining the criteria from [21] and [22] provides the following list of good attributes performance measurement should have.

- Proportionality—a linear relationship between the metric used to estimate performance and the actual performance observed by the workload. In other words, if the metric increases by 20%, then the real performance of the system should be expected to increase by a similar proportion.
  - A scalar performance measure for a set of benchmarks expressed in units of time should be directly proportional to the total time consumed by the benchmarks.
  - A scalar performance measure for a set of benchmarks expressed as a rate should be inversely proportional to the total time consumed by the benchmarks.
- Reliability—if the metric shows System A is faster than System B, it would be expected that System A outperforms System B in a real workload represented by the metric.
- Consistency—the definition of the metric is the same across all systems and configurations.
- Independence—the metric is not influenced by outside factors such as a vendor putting in special instructions that just impact the metric and not the workload.
- Ease of use—so the metric can be used by more people and/or more frequently.
- Repeatability—running the test for the metric multiple times should produce close to the same result.

While the Top500 metric does have some of these attributes, it misses several others. Whatever new approach is adapted should improve the ability for the metric to have all these attributes and also be more closely aligned to the spirit of these attributes. Furthermore, it would be beneficial if the metrics could meet the goals of any benchmark in representing an evolving workload and covering all the purposes of benchmarks discussed in section 2.1 above.

3.2.2 Aggregate the HPCC metrics into a single value

Dongarra's HPCC tests [23] are an attempt to provide a more general view of a system and its ability to perform on a wider range of applications. While each year individual measures are recorded and "winners" announced, this approach has never replaced the single metric Top500 list because it is harder to issue press releases or make simplified claims by using four values rather than one. However, there is no reason most of the HPCC measures cannot be generalized into a single composite measure. While each test assesses different characteristics, with the exception of *randomring* latency, the tests all have one thing in common: They do a certain amount of work in a given time period and the work unit is encapsulated in some activity, be the activity a floating-point operation or moving data from one location to another. Therefore, each test has a given amount of work to be done represented by the total number of reference actions that have to be carried out to complete the test. If the number of actions is deterministic, then the difference between two rates is only the wall clock time it takes to carry out all those actions. Hence, an appropriate composite function (see PERCU reference for a discussion of appropriate compositing functions) can result so that time-to-solution for each test is the only variable. As everyone agrees, time-to-solution is the only real measure of performance. Making this improvement will provide the community with a

single "actions/second" measure more representative of realistic system use.

3.2.3 Individualized rankings

Given enough data—such as the HPCC values and perhaps a few others, such as I/O and perhaps a deterministic random access, represented by the Graph500 test—it would be possible to create meaningful customized rankings. For example, one could set weights and create a custom listing—the top 500 machines for the workload in which a community is interested. This improvement would help our community get away from the mass-market rankings and promote personalized rankings that are really useful.

3.2.4 Weighted composite of actions

Combining improvements 2 and 3 may be a straightforward way to greatly improve the realism of a single metric without tremendous effort. As a thought experiment, assume there are three measures (from the HPCC or others tests): one for an arithmetic operation for a quantum of values, one for moving a quantum of data between memory locations, and one for moving a quantum of data across the interconnect. Call these actions A_1, A_2, and A_3. Each test would have a deterministic number of actions it carries out, represented as Na_1, Na_2, and Na_3, and the measurement of the time it takes to complete the amount of actions, shown as t_1, t_2 and t_3. Then the rate of actions is $\Phi(Na_1/t_1, Na_2/t_2, Na_3/t_3)$ where Φ is a composite function such as the geometric mean. Since $Na_{1\text{-}n}$ are deterministic, the only variables are the times to completing the work, giving this approach the proportional and other properties discussed above.

Good rates for some actions may be necessary, but not sufficient for achieving good overall performance. For example, for large-scale applications, flops/s are no longer the bottleneck ("flops are free"), but memory bandwidth and interconnect bandwidth often are. In the future, the shift of the bottleneck will be more true since moving data will dominate the use of energy within systems.

So, one can use a weighted composite function, with weights w_1, w_2 and w_3. Since systems today may have 100 to 1,000 more flops than interconnect bandwidth and 10 to 100 times more flops than memory bandwidth, it is possible to use 1, 10 and 100 for the weighting factors. Using weights, particularly with some study matching the best weights to workloads, could provide a more realistic single indicator for real (sustained) performance than just a Linpack value.

3.2.5 Create a new, meaningful suite of benchmarks

Many of the benchmark suites that are held in high regard (Livermore Loops [24], NPBs, SPEC) over time are suites of pseudo and/or full applications. While the best case for any benchmark is to be a statistically representative sample of real workload, in realty, this is not possible for community tests. Mashey [25] labels the Workload Characterization Analysis (WCA), a statistical study of all the applications in a workload, including their frequency of invocation and their performance. WCA is equivalent to the methodology outlined in [26]. This type of analysis provides a statistically valid random sampling of the workload. Of course, WCA takes a lot of effort and is rarely done for complex workloads. WCA also cannot be done with standard benchmark suites such as NPB or SPEC. While such suites may be related to a particular workload, by definition they cannot be random samples of a workload. The Workload Analysis with Weights (WAW) is possible after extensive WCA because it requires knowledge of the workload population. It can predict workload behavior under varying circumstances but still is not possible for broad community measures. For the goals of cataloguing a wide range of workloads, WAC and WAW are impracticable. The other type of analysis is the SERPOP (Sample Estimation of Relative Performance of Programs) method. In this category, a sample of a workload is selected to represent a workload. However, the sample is not random and cannot be considered a statistical sample. SERPOP methods occur frequently in performance analysis and reflect very meaning measures that span individual communities.

Many common benchmark suites—including SPEC, TCP and NPB, as well as many acquisition test suites—are SERPOP. In SERPOP analysis, the workload should be related to SERPOP tests, but SERPOP does not indicate the frequency of usage or other characteristics of any individual workload.

4. A COMMUNITY CALL TO ARMS

The community desperately needs a new, comprehensive SERPOP metric for HPC systems, particularly when extreme architectures will be evolving in the face of many technology constraints. Several SERPOP metrics exist and are in use today, such as the NERSC SSP test series, the DOD Technology Insertion benchmark series, and the NSF/Blue Waters SPP test [27]. While none represent every possible workload component, all are much more representative of real workloads than Linpack. So, any such SERPOP test would be more useful than Linpack.

It is time the HPC community stop apologizing for a poor and misleading metric that we claim we have to run and post. It is clear much better measures can do a much better job in representing the important advances of HPC technology and investment while better representing the end purpose of our systems and programs. So, this paper ends by asking all community segments system vendors, facility managers, funding stakeholders, and the press—to resist using measures that are grossly flawed.

Rather, we should insist in immediate evolutionary improvements while, in a very timely manner, moving to implement revolutionary improvements. It should be our shared goal that in November 2013, the 20th anniversary of the first release of the Top500 list, an entirely re-engineered metric is in place to guide our discussions, investment strategies and to record the amazing accomplishments of a vibrant segment of computing.

5. ACKNOWLEDGEMENTS

First, the author wishes to thank the four originators of the Top500 list, Jack Dongarra, Hans Meuer, Horst Simon, and Erich Strohmaier, for providing many years of data and insight. While this paper points out issues that have developed over time with the Top500 list it does not in any way indicate ill will or lack of respect for the accomplishment of creating the Top500 list. The author also wants to anonymously thank the community members for the thoughts and ideas on this topic. Finally, all comments are those of the author alone and do not represent any formal view of the organizations with which he has been or is associated.

6. REFERENCES

[1] Dongarra, Jack. *Performance of Various Computers Using Standard Linear Equations Software.* Computer Science , University of Tennessee, Knoxville TN, 37996: University of Tennessee, 1985

[2] Top 500 List. 1993-2012. http://www.top500.org (accessed 2012).

[3] http://www.netlib.org/Linpack

[4] Linpack Download. 2008. http://www.netlib.org/LINPACK (accessed 2008).

[5] Kramer, William T.C., *PERCU Results in a Reawakened Relationship for NERSC and Cray*, Cray User Group (CUG 2007) Conference, Seattle, WA, May 2007.

[6] Kramer, William. Blue Waters – A Super System for Super Challenges. *Proceedings of the Cray User Group, Stuttgart*, Germany, April 29-May3, 2012.

[7] NERSC SSP Project. *NERSC Projects*. http://www.nersc.gov/projects/ssp.php (accessed 2011).

[8] Kramer, William. *PERCU: A Holistic Method for Evaluating High Performance Computing Systems*, A Dissertation, University of California Berkeley, October 2008. http://www.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-143.pdf

[9] Carrington, Laura, M. Laurenzano, Allan Snavely, Roy Campbell, and Larry Davis. How Well Can Simple Metrics Represent the Performance of HPC Applications? *SC 05 - The High Performance Computing, Storage, Networking and Analysis Conference 2005*. Seattle, WA: Association of Computing Machinary (ACM), 2005.

[10] Tikir, M., L. Carrington, E. Strohmaier, and A. Snavely. A Genetic Algorithms Approach to Modeling the Performance of Memory-bound Computations. *Proceedings of SC07*. Reno, NV: Association of Computing Machinery (ACM), 2007.

[11] SPEC Benchmarks. 2000. http://www.spec.org (accessed 2008).

[12] Bailey, David H., and el.al. The NAS Parallel Benchmarks. *Intl. Journal of Supercomputer Applications* vol 5, no. 3 (Fall 1991): 66-73.

[13] Simon, Horst, and Erich Strohmaier. *Statistical Analysis of NAS Parallel Benchmarks and LINPACK Results.* Vol. 919, in *Lecture Notes In Computer Science*, edited by Bob Hertzberger and Guiseppe Serazzi, 626 - 633. London: Springer-Verlag, 1995.

[14] High Performance Technology Insertion 2006 (TI-06). *DOD Modernization Program.* 2005. http://www.fbodaily.com/archive/2005/05-May/08-May-2005/FBO-00802613.htm (accessed 2005).

[15] Culler, David E., and Jaswinder Pal Singh. *Parallel Computer Architecture: A Hardware/Software Approach.* First. Edited by Denise P.M. Penrose. San Francisco, CA: Morgan Kaufmann Publishers, Inc., 1999.

[16] Dongarra, Jack, *Reduced Linpack to Keep the Run Time Manageable for Future TOP500 Lists*, invited talk, International Supercomputing Conference, ISC 12, Homburg, Germany, June 17-21, 2012

[17] The Networking and Information Technologies Research and Development Program, Supplement to the President's Budget FY1993 to FY 2013, http://www.nitrd.gov/pubs/2013supplement/FY13NITRDSupplement.pdf

[18] Consumer Price Index – Bureau of Labor Statistics. www.bls.gov/cpi/, (Accessed April 2012)

[19] Moore, Gordon E., Cramming more components onto integrated circuits. *Electronics Magazine*, 1965.

[20] Saini, Subach and David Bailey, *NAS Parallel Benchmark Results (Version 1.0) 11-96*. NAS Technical Report 96-18, Nasa Ames, tables 13-17. http://www.nas.nasa.gov/assets/pdf/techreports/1996/nas-96-018.pdf - other data is available in similar reports.

[21] Smith, J. E. Characterizing Computer Performance with a Single Number. *Communications of the ACM* (Association of Computing Machinery) 31, no. 10 (October 1988): 1202-1206.

[22] Lilja, David. *Measuring Computer Performance: A Practitioner's Guide.* Cambridge University Press, 2000.

[23] Dongarra, Jack. *Performance of Various Computers Using Standard Linear Algebra Software in a Fortran Environment*. HPCC. http://www.netlib.org/benchmarks/performance.ps (accessed 2012).

[24] McMahon, F. *The Livermore Fortran Kernels: A computer test of numerical performance range*. Technical Report , Lawrence Livermore National Laboratory, Livermore, CA: University of California, 1986.

[25] Mashey, John R. War of the Benchmark Means: Time for a Truce. *ACM SIGARCH Computer Architecture News* (Association for Computing Machinery) 32, no. 4 (September 2004).

[26] Bucher, Ingrid, and Joanne Martin. *Methodology for Characterizing a Scientific Workload.* Technical Report, Los Alamos, NM 87545: Los Alamos National Laboratory, 1982

[27] Gregory H. Bauer, Torsten Hoefler, William T. Kramer and Robert A. Fiedler. Analyses and Modeling of Applications Used to Demonstrate Sustained Petascale Performance on Blue Waters, *Proceedings of the Cray User Group*, Stuttgart, Germany, April 29-May3, 2012