U.S. Department of Energy Best Practices Workshop on
File Systems & Archives
San Francisco, CA
September 26-27, 2011

Andrew Uselton
NERSC/LBL
acuselton@lbl.gov

Jason Hick
NERSC/LBL
jhick@lbl.gov

ABSTRACT / SUMMARY

This position paper addresses the business of storage systems and practices related to planning for future systems (I-1A) and establishing bandwidth requirements (I-1B), with some discussion also relating to the administration of storage systems and the monitoring of specific metrics (II-2A). The best practice is to balance I/O with compute capability.

We present a quantitative characterization of "HPC and I/O system balance" by examining the relative costs of compute resources and I/O resources on the one hand and the relative impact of compute and I/O activities on the other.

INTRODUCTION

An HPC system with too little I/O infrastructure to support its workload could leave much of the compute resource idle as it waits for I/O operations to complete. The idle compute resource represents an opportunity cost in that it may have no other useful work to do during the wait.

BACKGROUND

One study [2] suggests that memory capacity is the key determinant of necessary I/O bandwidth and capacity. Figure 1 presents a traditional guideline for balancing I/O.

The relationship between performance and memory comes from the need to flush the contents of memory to persistent local storage in a combination of reasonable time and cost.

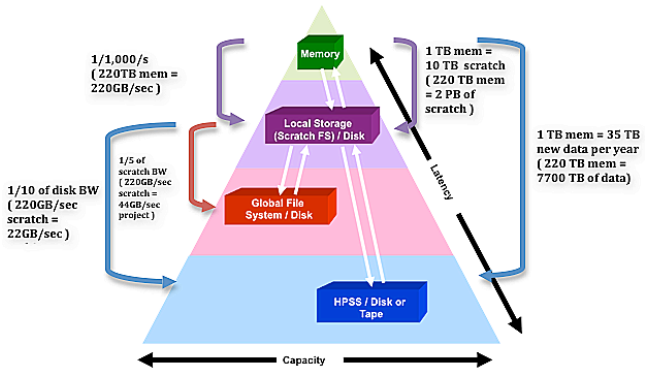

Figure 1. Conventional HPC I/O Planning Guidelines

Additional I/O resources provide diminishing returns, so there is a point of balance at which bandwidth is "just enough", and in this case the heuristic is to move all of memory in about 1000 seconds.

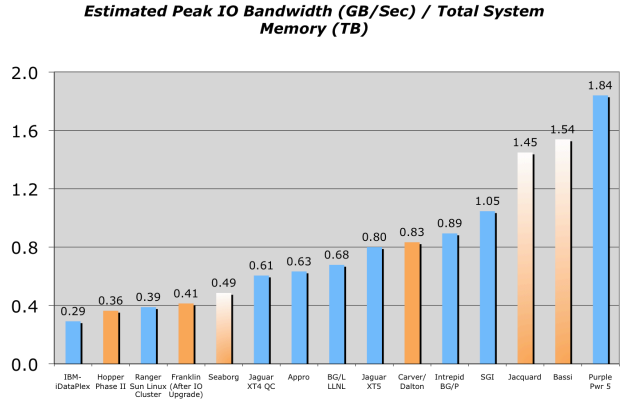

Figure 2. Peak bandwidth to system memory

Figure 2 presents this heuristic as applied to several HPC systems. By that metric, systems with a value over 1.0 have over-provisioned I/O

subsystems relative to system memory capacity. A subjective review of such systems reveals that users are happy with the I/O bandwidth they deliver.

The purpose of this paper is to propose an alternative characterization of balance using a cost-based model in conjunction with the compute and I/O workloads of the HPC system. As a starting point, this discussion abstracts away much of the complexity to arrive at some core ideas.

SYSTEM BALANCE

As a first simplifying assumption, suppose that the cost of an HPC system is composed entirely of the budget for the compute capability and the budget for the I/O capability. Next, suppose that the work produced by an HPC system is measured as the number of jobs completed weighted by the size of each job in two dimensions: the number of *node-seconds* used in the computation and the number of *node-seconds* used in I/O.

Further, assume that the aggregate compute capability is near linear in the cost of the of compute nodes:

$$C(n) = M_n \times n$$

where M_n is the marginal cost of nodes. Similarly, assume the aggregate I/O capability (measured as its peak rate) is near linear in the cost of the I/O infrastructure:

$$I(r) = M_r \times r$$

where M_r is the marginal cost of adding a unit of bandwidth r.

Now let the utilization U of the HPC system be given by the fraction of *node-seconds* spent on compute activity, given a particular workload. Our characterization of "system balance", given n and r, is given by:

$$B = \left(\frac{U}{(1-U)}\right)\left(\frac{I(r)}{C(n)}\right)$$

Our claim is that at $B = 1$, the system is in balance in that it achieves the maximum amount of workload per dollar spent. As an example, if you spend 10% of your HPC system budget on I/O infrastructure ($\frac{I(r)}{C(n)} \cong 0.1$), then the nodes should be spending 10% of their time on I/O, and the rest on computation ($\frac{1-U}{U} \cong 0.1$).

This is a relatively intuitive idea given the simplifying assumptions, but it begs the question, "What is B on my system, given its workload?" Those who design and purchase HPC systems are very familiar with the total cost and the fraction spent on I/O infrastructure. On the other hand, it is not at all clear what the value of U is. It will certainly be different at different times and for different workloads. We propose that monitoring the jobs and I/O on the system for any given day's activity and for longer intervals will yield the value of U.

CHALLENGES

Some system designs and I/O strategies attempt to improve I/O performance by departing from this simple model. For example, a strategy that overlaps computation and I/O will yield a higher utilization. In that case it becomes important to estimate both the expected impact and the extent to which the strategy is implemented in practice. If a strategy can entirely "hide" I/O activity but only affects 10% of the workload, then the simplified model is still close to correct.

The model has plenty of room for improvement. For example, the cost model does not need to as simple as presented. There may be fixed costs and nonlinearities, and the model could incorporate them without difficulty. The model can also include other aspects of HPC system architecture, for example, adding *node-seconds* spent in (node to node) communication. In some cases that communication will compete for bandwidth with the I/O requirements, leading to additional complexities.

CASE STUDY

The *Carver* IBM Dataplex cluster at NERSC was provisioned with approximately 15% of its budget dedicated to I/O infrastructure. The

system has 30 TB of memory suggesting a target bandwidth to storage of 30 GB/s using the heuristic from the background discussion. *Carver's* measured bandwidth is about 25 GB/s, so it is designed to be at about 83% of that target. *Carver* runs the Integrated Performance Monitoring (IPM) library [3] with every scheduled job. Each job produces a report at the end of its execution giving the time spent in computation, the time spent in I/O, and the amount of data moved (among other quantities). IPM provides a comprehensive profile of compute and I/O activity for a given interval. From that profile it is possible to directly calculate the utilization. For example, in June 2011 $U \cong 0.94$. The balance factor for the actual workload is around 2.5. By this measure, the system's balance favors I/O and could handle a heavier load.

CORRELATING I/O ACTIVITY WITH JOBS

Most HPC systems do not have IPM or other direct measures of the utilization. Without that information we do not know what balance has been achieved in practice after having applied the heuristics from Figure 1. An alternative strategy is under development at NERSC that infers the utilization U from server-side I/O monitoring with the Lustre Monitoring Tool (LMT) [4]. Server-side data is anonymous with respect to the nodes that generate the I/O. Nevertheless, it is often possible to infer the job from the I/O pattern. When that can be done comprehensively it will yield the utilization as before, and therefore give a quantitative gauge of the balance.

On NERSC's *Franklin* Cray XT4 there are commonly more than one hundred jobs running at a given time, and the I/O workload resulting from that compute workload is potentially composed of I/O from many jobs simultaneously. Often, an application runs many times repeating the same I/O pattern each time. From that collection of jobs (call it a *job class*) we calculate the average I/O behavior for the application, which is an approximation of its expected behavior in isolation from other jobs. The individual calculated behavior of each of the whole suite of applications provides the initial estimate for the behavior of the system as a whole, and the estimates can be iteratively refined via a generalized linear regression. This is a computationally expensive task but straight forward, in principle.

As an example, we examine the IOR [5] file system benchmark, which runs as a regularly scheduled test of the *Franklin* scratch file systems. 175 such tests were run in July 2011, and the system job log records the start and stop time of each job along with the number of nodes used. The IOR application runs using the same parameters in order to provide a repeatable health-check of the file system. Each job writes *4GB* to the file system from each of 64 tasks on 16 nodes. It then reads that data back in. Jobs generally run for 150 to 200 seconds, but can run much longer when the file system is occupied with other I/O. The jobs are submitted to an I/O-oriented scheduling queue, which (voluntarily) serializes I/O intensive applications.

LMT records the bytes written and bytes read for each server every five seconds. That data shows the I/O resulting from the IOR jobs and anything else running at the same time. In order to calculate the average behavior we "warp" (artificially lengthen or shorten) the sequence of LMT observations for each job so that they fit the same length-scale – chosen as the median job run length. A standard linear regression on that data set provides the calculated average behavior.

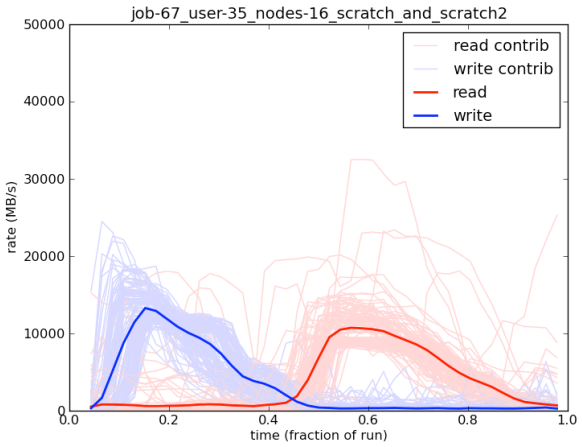

Figure 3.

Figure 3 displays the result of carrying out this analysis. The *x-axis* is the artificial time scale – arbitrarily set to 0 to 1 – to which each series of observations is warped. The *y-axis* gives the aggregate data rate (blues for writes and red for reads) of the application over the course of the idealized run. The single dark line of each color is the calculated average behavior of the application. Shown in a lighter shade is the collection of 175 separate contributing runs as they appear after being warped. Most of the contributing runs follow the average behavior closely, and demonstrate that the IOR test was running without much interference. A few traces depart wildly from the average and it is those runs that were in contention for I/O resources. Once we calculate the idealized average behavior all of the applications with significant amount of I/O, those idealizations become initial estimates for the coefficients in a big matrix implementing the generalized linear regression. For a file system that does not have a lot of I/O contention the initial estimates will be close to their final values and the computation will converge quickly. In other cases the computation may take significant resources. The end result is a quantified, job-by-job measure for the impact of the application on the I/O system from which we recover the utilization U and therefore the balance B.

CONCLUSIONS

The HPC community has developed a set of heuristics to guide the design of HPC systems so that the I/O capability is matched to the users' needs. In one case where the heuristic was applied in an effort to make a system *I/O-friendly*, a comprehensive characterization of balance using our metric showed that the system deployment was successful. Our measure for balance, $B = 2.5$, says that the system is cost effective for an even heavier I/O load than was observed.

The proposed job-log-and-LMT analysis extends the applicability of our metric to cases where direct observation of the utilization U is impossible or impractical. That characterization can be combined with system cost details to establish a rigorous evaluation of the balance of the system.

It is always difficult to argue that past performance is guide to future behavior. When planning for a new HPC system the application behavior produced in this analysis must be combined with theoretical considerations for how the new system might behave differently. Nevertheless, the application characterizations and the underlying model that produced them are a valuable starting point to be used during the procurement of new systems.

Once deployed, a new system needs data acquisition systems like IPM, Darchan [6], and LMT in order to evaluate the system balance actually achieved.

ACKNOWLEDGEMENTS

We would like to thank Dr. Anthony Gamst of UC San Diego for discussion, explanations, and guidance of the generalized linear regression techniques alluded to in this paper.

REFERENCES

1. NERSC Storage Policies, produced by the Storage Policies Working Group, 2010.
2. J. Shoopman, LLNL internal study on memory capacity to archive data generated 2008-2009.
3. D. Skinner, *Integrated Performance Monitoring: A Portable Profiling Infrastructure for Parallel Applications.* Proc. ISC 2005, International Supercomputing Conference, Heidelberg, Germany, 2005
4. A. Uselton, *Deploying Server-side File System Monitoring at NERSC,* Cray User Group Conference, Atlanta, GA, 2009
5. H. Shan, K. Antypas, J. Shalf, *Characterizing and Predicting the I/O Performance of HPC Applications Using a Parameterized Synthetic Benchmark.* Proc. SuperComputing 2008, Austin, TX, 2008
6. P. Carns, R. Latham, R. Ross, K. Iskara, S. Lang, K. Riley, *24/7 characterization of petascale I/O workloads*. Proc. Of 2009 Workshop on Interfaces and Architectures for Scientific Data Storage, Sep. 2009.