

Storage Speedup: An Effective Metric for I/O-intensive Parallel Application

Wei Hu^a, Guangming Liu^{ab}, Qiong Li^a, Xiaofeng Wang^a, Wenrui Dong^a

^a College of Computer, National University of Defense Technology, Changsha, China

^b National Supercomputer Center in Tianjin, Tianjin, China

{huwei, liugm}@nsc-tj.gov.cn, qiong_li@yahoo.com.cn, xf_wang@nudt.edu.cn, dongwr@nsc-tj.gov.cn

Abstract— With supercomputer system scaling up, the performance gap between compute and storage system increases dramatically. The traditional speedup only measures the performance of compute system. In this paper, we firstly propose the speedup metric taking into account the I/O constraint. The new metric unifies the computing and I/O performance, and evaluates practical speedup of parallel application under the limitation of I/O system. Furthermore, this paper classifies and analyzes existing parallel systems according to the proposed speedup metric, and makes suggestions on system design and application optimization. Finally, we provide the analysis of this new speedup metric by case studies. The storage speedup reflects the degree of parallel application scalability affected by performance of storage system. The results indicate that the proposed speedup for parallel applications is an effective metric.

Keywords—high performance computing, storage speedup, scalability, system classification, storage architecture

I. INTRODUCTION

With the scaling up of supercomputers, the system performance advances considerably. In the latest TOP500 supercomputing list in 2015, Tianhe-2 consisting of 3,120,000 cores reaches the performance of 33.86 petaflop/s [1]. Planned exascale supercomputers (10^{18} floating point operations per second or 10^3 petaflop/s) are promising to come in this decade [2].

Since the 1980s, the average growth of processor performance could reach 60%, while the read and write bandwidth of disk which is the main storage device only increased by 10% to 20% per year. In addition, the computing system scales much faster than I/O system. Both of above reasons exacerbated the mismatch between application requirements and I/O performance. For I/O-intensive applications, the performance is not only determined by compute system, but also greatly affected by I/O system.

In this research, we analyze and quantify the effects of storage bottleneck of supercomputers. Our theory provides the new metric on how to quantify the impact of I/O system on application performance. The main contributions of our work lie in the following aspects.

- This paper introduces a new speedup metric called storage speedup taking into account the I/O performance constraint. The new metric unifies the computing and I/O performance, and evaluates practical speedup of parallel

application under the limitation of I/O system.

- The existing parallel systems are classified and analyzed according to the storage speedup, and the suggestions are acquired on system design and application optimization.
- Through the case studies, the effectiveness of storage speedup is validated.

The rest of paper is organized as follows. Section 2 reviews the related work. Section 3 defines the storage speedup and classifies the existing systems according to it. Section 4 uses some case studies to analyse the I/O architecture in supercomputing system to validate the effectiveness of the new metric. Finally, in section 5 we conclude this paper and discuss some future works.

II. RELATED WORK

The essentiality of supercomputer is to reduce the execution time of applications by parallel computing, while speedup has been almost exclusively used for measuring scalability in parallel computing [3].

In 1967, Amdahl advocated a speedup model for a fixed-size problem [4]:

$$S_{Amdahl} = \frac{1}{f + (1-f)/P}$$

where P is the number of processors and f represents the serial ratio of the program. Obviously, the equation reveals a pessimistic view on the usefulness of large scale parallel computers since the maximum speedup cannot exceed $1/f$.

In 1988, Gustafson introduced a scaled speedup for a fix-time problem [5], which scales up the workload with the increasing number of processors to preserve the execution time:

$$S_{Gustafson} = f + (1-f)P$$

This speedup proves the scalability of the parallel computers and overcomes the shortcomings of Amdahl's speedup model.

Sun and Ni [6] presented a memory-bounded speedup model which scales up the workload according to the memory capacity of the system, demonstrates the relationship among memory capacity, parallel workload and speedup.

Culler [3] proposed a speedup model taking the communication and synchronization overhead into account. Researchers can improve the system performance depending on this

model by reducing the overhead of communication and synchronization such as overlapping communication and computing, load balancing and so on.

Yang [7], [8] studied the reliability and power issues of supercomputers, introduced the reliability and power related speedup. They provided a new perspective to solve reliability and power related issues.

In this work, we develop our storage speedup metric to analyze and quantify the effects of I/O performance for parallel applications in supercomputing.

III. THE STORAGE SPEEDUP

Suppose a parallel system consisting of P nodes of single core processor, denoted by $N_1 \dots N_P$. Let a parallel program Q be executed on the system with one process per node, resulting in a total of P processes. If a processor has more than one core or one node has more than one processor, P denotes the number of cores in the system with cores being treated as nodes. Hereby, there will be one process running on each core. In the following paper, it is assumed that there will be one process running on one node to simply the model.

A. Definition for the Storage Speedup

As shown in Figure 1, for a given program Q which has I/O workloads can be divided into two kinds of phases: I/O phases and compute phases. The compute phases are denoted as shaded parts compared with the I/O phases denoted as the white parts. The single process execution case is shown in Figure 1(a), where 0 represents the process and T_l denotes the execution time. The execution case of parallel system is shown in Figure 1(b), where P represents the number of processes and T_p^S denotes the practical execution time of parallel program. The I/O workloads mainly consist of reading the raw data, storing intermediate data, writing the final results and so on. Every I/O phases we denoted as an I/O operation point.

For the serial program Q running on one compute node in Figure 1(a), let N^S be the amount of I/O operation points. I^{sint} is the average time interval among I/O operation points, and on each I/O operation point the average I/O service time is I^{sser} , then all the execution time of I/O-free phases is T^{snon} , so the time of the program Q runs on a node serially is:

$$T_{Slo}^S = T^{snon} + N^S \cdot I^{sser} = T^{snon} + (T^{snon} / I^{sint}) \cdot I^{sser} \quad (1)$$

For the parallel program Q running on P compute nodes in

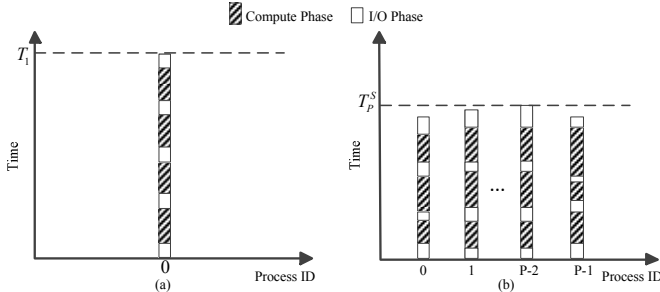


Figure 1. Execution state of a program on a P node system. (a) Single process. (b) P processes in parallel.

Figure 1(b), in regard to the i -th ($0 \leq i < P$) process in the P processes, we denote that N_i^P is the amount of I/O operation points, I_i^{pint} is the average time interval among I/O operation points, I_i^{pser} is the average I/O service time on each I/O operation point, and T_i^{pnon} is the execution time of I/O-free phases, so the average values of the parameters related to all processes can be given as follows.

The average value of I/O operation points:

$$N^P = \frac{1}{P} \sum_{i=0}^{P-1} N_i^P$$

The average value of average time intervals among I/O operation points:

$$I^{pint} = \frac{1}{P} \sum_{i=0}^{P-1} I_i^{pint}$$

The average value of average I/O service time on every I/O operation point:

$$I^{pser} = \frac{1}{P} \sum_{i=0}^{P-1} I_i^{pser}$$

The average value of average I/O-free execution time:

$$T^{pnon} = \frac{1}{P} \sum_{i=0}^{P-1} T_i^{pnon}$$

Then the real execution time of program Q on P processors is approximated as follows:

$$T_{Slo}^P \approx T^{pnon} + N^P \cdot I^{pser} = T^{pnon} + (T^{pnon} / I^{pint}) \cdot I^{pser} \quad (2)$$

So we give the definition of storage speedup model.

Definition 1 (Storage Speedup). When program Q runs on a parallel system, the storage speedup achieved is defined as follows:

$$S_{Slo}^P = \frac{T_{Slo}^S}{T_{Slo}^P} \quad (3)$$

According to (1) & (2), suppose that S_p is the speedup of the I/O-free part in the program Q , while $N = I^{pser} / I^{pint}$ and $M = I^{sser} / I^{sint}$, the storage speedup can be converted below:

$$\begin{aligned} S_{Slo}^P &= \frac{T_{Slo}^S}{T_{Slo}^P} \approx \frac{T^{snon} + (T^{snon} / I^{sint}) \cdot I^{sser}}{T^{pnon} + (T^{pnon} / I^{pint}) \cdot I^{pser}} \\ &= \frac{T^{snon} (1 + I^{sser} / I^{sint})}{T^{pnon} (1 + I^{pser} / I^{pint})} = S_p \cdot \frac{1 + M}{1 + N} = S_p \cdot \frac{1}{1 + \frac{N - M}{1 + M}} \end{aligned} \quad (4)$$

Let us define

$$O(P) = \frac{N - M}{1 + M} \quad (5)$$

which reflects the variation of I/O execution with the parallel application scaling and is called the storage workload factor.

As a result, our storage speedup formula is refined to

$$S_{Slo}^P = S_p \cdot \frac{1}{1 + O(P)} \quad (6)$$

which indicates the speedup affected by I/O performance variation with the processors scales.

In formula (6), if S_p is instantiated by Amdahl's speedup and Gustafson's speedup, we obtain different storage speedup formula as follows.

Amdahl Storage Speedup

$$S_{Amdahl-Sto}^P = S_{Amdahl} \cdot \frac{1}{1+O(P)} = \frac{P}{1+f(P-1)} \cdot \frac{1}{1+O(P)} \quad (7)$$

Gustafson Storage Speedup

$$S_{Gustafson-Sto}^P = S_{Gustafson} \cdot \frac{1}{1+O(P)} = (f + P(1-f)) \cdot \frac{1}{1+O(P)} \quad (8)$$

Due to the competition of I/O resources, I/O service time is prolonged in parallel computing, resulting in $O(P) \neq 0$. Ideally, if $O(P)=0$, our two storage speedup models are simplified into the traditional Amdahl's and Gustafson's speedup formulas.

$$S_{Sto}^P = S_P \quad (9)$$

B. System Classification

First we make some symbols conventions. We denote $f(x) > g(x)$ if $\lim_{x \rightarrow \infty} f(x)/g(x)$ is ∞ and $f(x) \geq g(x)$ if $\lim_{x \rightarrow \infty} f(x)/g(x)$ is a positive constant or ∞ . Adversely, $f(x) < g(x)$ indicates $\lim_{x \rightarrow \infty} f(x)/g(x)$ is 0 and $f(x) \leq g(x)$ indicates $\lim_{x \rightarrow \infty} f(x)/g(x)$ is a non-negative constant. Suppose $\Theta(x)$ is a set consisting of all functions of x , $f(x) \in \Theta(x)$ and $g(x) \in \Theta(x)$. If $\lim_{x \rightarrow \infty} f(x)/g(x)$ is a positive constant, it denotes $f(x) = \Theta(g(x))$ or $g(x) = \Theta(f(x))$.

From the definition of storage speedup formula (6), it is easy to observe that $O(P)$ is the key factor in storage speedup variations. According to the characteristic of $O(P)$ related to different parallel systems, considering these systems to be classified as follows.

Definition 2 (Constant and Incremental Systems). Suppose that a program Q satisfying $\lim_{P \rightarrow \infty} S_P = \infty$ runs on a W system. If $O(P) \leq \Theta(1)$, the W system is considered to be a constant system. And if $O(P) > \Theta(1)$, the W system is considered to be an incremental system.

As shown in Figure 2, there are three examples about the classification based on $O(P)$. If $O(P) = K \leq \Theta(1)$, where K is a positive constant, the W system is a constant system. We find the storage speedup increases linearly with the number of nodes which indicates that the scalability of the application is not bound by the I/O performance obviously.

If $O(P) = KPlgP > \Theta(1)$, the W system is an incremental system. This is different from above, the scalability of the application is bound by the I/O performance.

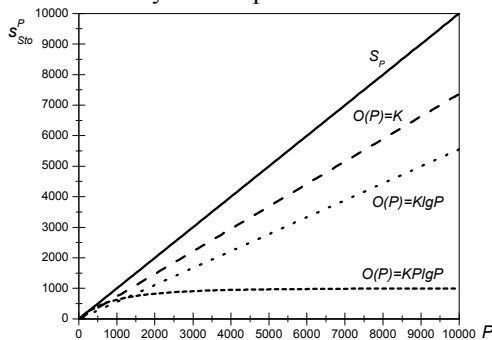


Figure 2. Three examples about storage speedup of different kinds of systems based on $O(P)$

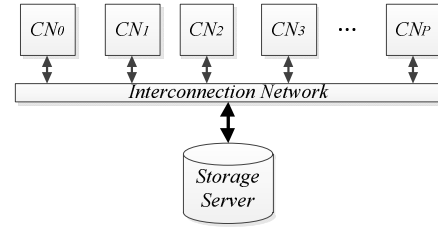


Figure 3. Centralized I/O Architecture

IV. I/O ARCHITECTURE ANALYSIS BASED ON STORAGE SPEEDUP

According to the interconnection relationship between compute and storage in supercomputing systems, the architecture of supercomputing storage systems can be divided into three categories: centralized architecture, distributed and parallel architecture, centralized, distributed and parallel architecture (The following in the paper is abbreviated as CDP I/O architecture). The analyses for each I/O architecture related to storage speedup are as follows.

A. Centralized I/O Architecture

Figure 3 shows a centralized I/O architecture which is often used in small-scale supercomputers, such as NAS. The supercomputer storage system with centralized I/O architecture is easy to be configured and managed, but has a poor scalability. It can only scale up instead of scaling out, since the storage system cannot scale horizontally to multiple servers but only to enhance the performance of a single server vertically. Therefore the performance of storage system is severely limited by the I/O architecture. As the storage system with centralized I/O architecture cannot scale out to more storage nodes, we obtain $O(P) \geq \Theta(P)$. According to definition 2, the supercomputer with centralized I/O architecture is an incremental system whose parallel computing scalability is constrained by the performance of I/O system. Figure 4 shows us a case about supercomputer with centralized I/O architecture whose storage speedup is bounded by the I/O performance.

B. Distributed and Parallel I/O Architecture

Figure 5 shows a distributed and parallel I/O architecture. A supercomputer storage system with distributed and parallel I/O architecture usually refers to the architecture in which

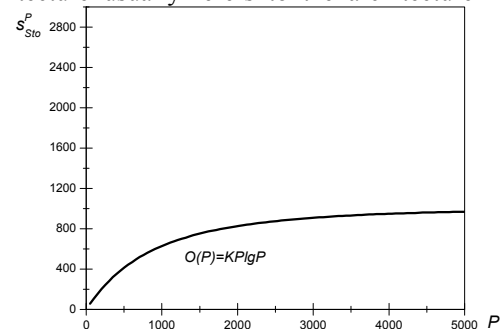


Figure 4. A case about storage speedup bounded by the I/O performance related to centralized I/O architecture

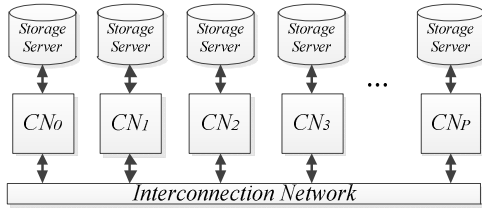


Figure 5. Distributed and Parallel I/O Architecture

each compute node with built-in storage or directed attached storage server. Each compute node has file system which can provide I/O service to itself and other compute nodes. Due to this special nature, the implementations of the parallel file system are complicated due to I/O scheduling and data global consistency. And the storage servers have different positions to the compute nodes themselves and other compute nodes, therefore the system load balance also becomes a problem. So, distributed and parallel I/O architecture is usually adopted by supercomputers with small or medium scale.

For distributed and parallel I/O architecture, the storage nodes and compute nodes scaling equally. So we obtain $O(P) = \Theta(1)$. The supercomputer with distributed and parallel I/O architecture is a constant system and the parallel computing scalability is not constrained by the performance of I/O system, as the Figure 6 shows. With the node number increasing, the performance of storage system expands proportionally.

C. Centralized, Distributed and Parallel Architecture

Figure 7 shows a CDP I/O architecture which is dominant in the supercomputers of TOP500 [1]. For the Medium-sized supercomputers (the amount of compute nodes reaches 10^3), compute nodes and storage servers are connected directly through interconnection network as shown in the Figure 7(a). Parallel storage system consists of storage servers which are usually installed with parallel file system, such as Lustre, PVFS and so on, while compute nodes access the storage system by the client of parallel file system. Typical systems are Titan, Tianhe-1A and so on. With the supercomputer's development, I/O nodes (ION) are inserted between compute nodes and parallel storage systems to provide the function of I/O forwarding and management, as the Figure 7(b) shows. On the one hand, the amount of compute nodes can increase continually without the limitation of the client amount of

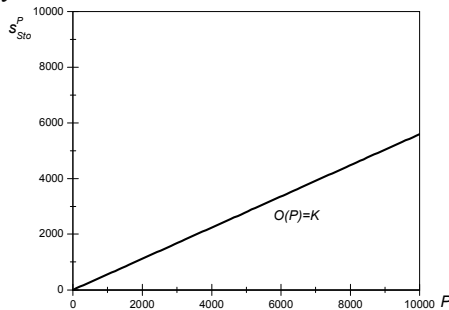


Figure 6. A case about storage speedup which is not affected by the I/O performance related to distributed and parallel I/O architecture

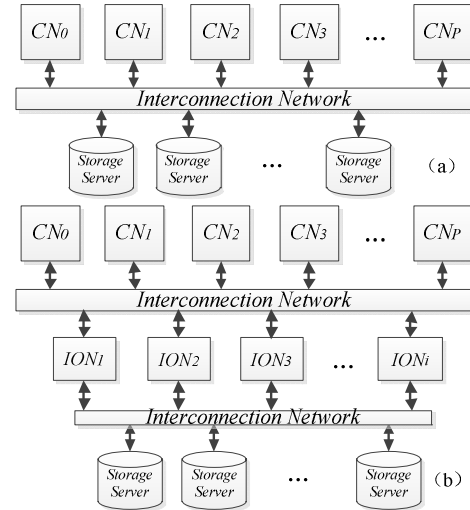


Figure 7. Centralized, Distributed and Parallel Architecture

parallel file systems. On the other hand, I/O performance can be improved by I/O scheduling and caching. All of these are attributed to the introduction of I/O nodes in the systems. The typical systems are IBM Bluegene series and Tianhe-2. Although the system has been improved by the I/O nodes, the essence of I/O architecture doesn't change. Thus, to simplify the model, I/O node layer is omitted in the subsequent analysis.

In the CDP I/O architecture, the storage system can not only scale up, but also scale out. But they still have limitations, first, for a specific supercomputer the scale of storage system is fixed within a certain time period; second, parallel file system usually has up limits on the number of storage nodes, the aggregate performance and the storage capacity. So we can also obtain $O(P) \geq \Theta(P)$. According to definition 2, the supercomputer with CDP I/O architecture is an incremental system whose parallel computing scalability is constrained by the performance of I/O system as shown by Figure 8.

The performance of storage system with CDP I/O architecture is much higher than the system with centralized I/O architecture, so the speedup affected by the I/O performance is not obvious until the parallel application increases to a huge scale.

V. CONCLUSIONS

This paper introduces a new speedup metric called storage

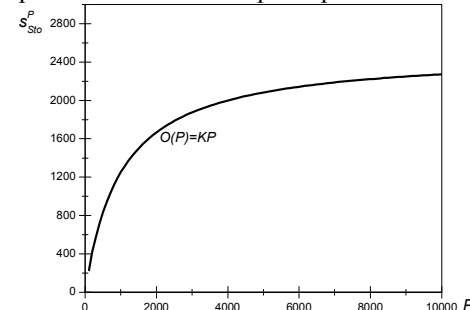


Figure 8. A case about storage speedup which is constrained by the I/O performance related to centralized, distributed and parallel architecture

speedup taking into account the I/O performance constraint. The new metric unifies the compute and I/O performance, and evaluates practical speedup of parallel application under the limitation of I/O system. The existing parallel systems are classified and analyzed according to the storage speedup, and the suggestions are acquired on system design and application optimization. Through the case studies, the effectiveness of storage speedup is validated.

In the future, our efforts will mainly focus on the following aspects. For the storage speedup, we will refine the theory and explore more factors. At the same time, we will make further research on characteristics of massively parallel applications and supercomputer storage systems to improve the scalability of system.

REFERENCES

- [1] Top500 website. [Online]. Available: <http://www.top500.org/>.
- [2] *Darpa sets ubiquitous hpc program in motion*, http://www.hpcwire.com/2010/08/10/darpa_sets_ubiquitous_hpc_program_in_motion/.
- [3] D. E. Culler, A. Gupta and J. P. Singh, *Parallel Computer Architecture: A Hardware/Software Approach*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1997.
- [4] G. M. Amdahl, "Validity of the single processor approach to achieving large scale computing capabilities," in *Proc. AFIPS '67*, 1967, pp.483-485.
- [5] J. L. Gustafson, "Reevaluating amdahl's law", *Communications of the ACM*, vol. 31, no. 5, pp. 532-533, 1988.
- [6] X. Sun and L. Ni, "Scalable problems and memory-bounded speedup," *Journal of Parallel and Distributed Computing*, vol. 19, no. 1, pp. 27-37, 1993.
- [7] Z. Wang, "Reliability speedup: an effective metric for parallel application with checkpointing", in *Proc. 2009 International Conference on Parallel and Distributed Computing, Applications and Technologies*. 2009, pp. 247-254.
- [8] X. Yang, J. Du, and Z. Wang, "An effective speedup metric for measuring productivity in large-scale parallel computer systems", *The Journal of Supercomputing*, vol. 56, no. 2, pp. 164-181, 2011.



Xiaofeng Wang has been working as an assistant professor in the College of Computer at National University of Defense Technology in China. He received the B.S., M.S. and Ph.D. degrees in computer science from National University of Defense Technology in 2004, 2006 and 2009 respectively. His research interests include trustworthy networks and systems, applied cryptography, network security.



Wenrui Dong received the B.S. and M.S. degrees from National University of Defense Technology, China, in 2009 and 2011, respectively. He currently pursues the Ph.D. degree in the College of Computer, National University of Defense Technology. His research interests include high performance computing and massive storage.



Wei Hu received the B.S. degree from PLA University of Science and Technology, China, in 2004, and the M.S. degree from National University of Defense Technology, China, in 2010. He currently pursues the Ph.D. degree in the College of Computer, National University of Defense Technology, Changsha, China. His research interests include high performance computing and machine learning.



Guangming Liu received the B.S. and M.S. degrees from National University of Defense Technology, China, in 1980 and 1986 respectively. He is now a professor in the College of Computer, National University of Defense Technology. His research interests include high performance computing, massive storage and cloud computing.



Qiong Li received the B.S., M.S. and Ph.D. degrees from National University of Defense Technology, China, in 1993, 1996 and 2010, respectively. She is now a professor in the College of Computer, National University of Defense Technology. Her research interests include high performance computing and massive storage.