An Effective Speedup Metric Considering I/O Constraint in Large-scale Parallel Computer Systems

Guilin Cai*, Wei Hu*, Guangming Liu*,**, Qiong Li*, Xiaofeng Wang*, Wenrui Dong*

*College of Computer, National University of Defense Technology, Changsha, China

**National Supercomputer Center in Tianjin, Tianjin, China

cc_cai@163.com, { huwei, liugm}@nscc-tj.gov.cn, qiong_joan_li@yahoo.com.cn, xf_wang@nudt.edu.cn, dongwr@nscc-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. Based on the storage speedup, we also generalize these results into a general storage speedup by considering not only speedup but also costup. Finally, we provide the analysis of these new speedup metrics 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 speedups for parallel applications are effective metrics.**

***Keyword*—storage speedup, general storage speedup, scalability, system classification**

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]. Planed 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.

Due to the increasing scale of the supercomputer system and parallel application, the I/O performance and storage capacity requirements of the supercomputer increase rapidly. The concerns about the storage scalability are not only the performance can be obtained but also the cost-effectiveness related to investment. The larger the system is, the more investments improving system performance needs. How to evaluate and promote the effectiveness of the investments of the storage system is another important problem need to be addressed.

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, and we also present the new method to evaluate the cost-effectiveness of the storage system of supercomputer. 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.
- Based on the storage speedup, a general storage speedup is generalized to evaluate the cost-effectiveness of the storage system by considering not only speedup but also costup.
- Through the case studies, the effectiveness of these new

Manuscript received February 21, 2016. This work is a follow-up of the invited journal to the accepted conference paper of the 18th International Conference on Advanced Communication Technology.

G. Cai is with the College of Computer, National University of Defense Technology, Changsha, 410073, China (e-mail: cc_cai@163.com).

W. Hu is with the College of Computer, National University of Defense Technology, Changsha, 410073, China (corresponding author, phone: +86-22-65375500; fax: +86-22-65375501; e-mail: huwei@nscc-tj.gov.cn).

G. Liu is with the College of Computer, National University of Defense Technology, Changsha, 410073, China (e-mail: liugm@nscc-tj.gov.cn).

Q. Li is with the College of Computer, National University of Defense Technology, Changsha, 410073, China (e-mail: qiong_joan_li@yahoo.com.cn).

X. Wang is with the College of Computer, National University of Defense Technology, Changsha, 410073, China (e-mail: xf_wang@nudt. edu.cn).

W. Dong is with the College of Computer, National University of Defense Technology, Changsha, 410073, China (e-mail: dongwr@nscc-tj.gov.cn).

speedup metrics is validated.

The rest of paper is organized as follows. Section 2 reviews the related work. Section 3 defines the storage speedup, classifies the existing systems according to it and presents the general storage speedup. Section 4 uses some case studies to analyze the I/O architecture in supercomputing system to validate the effectiveness of the new metrics. 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.

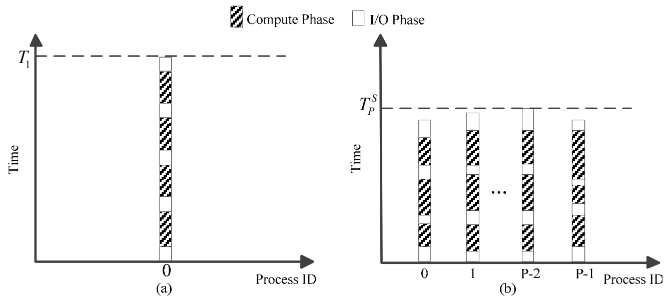


Fig. 1. Execution state of a program on a P node system. (a)Single process. (b)P processes in parallel.

III. Storage Speedup

Suppose a parallel system consisting of P nodes of single core processor, denoted by $N_1 ... N_P$. 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 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_1 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:

$$T_{Sto}^{S} = T^{snon} + N^{s} \bullet I^{sser} = T^{snon} + \left(T^{snon}/I^{sint}\right) \bullet I^{sser} \tag{1}$$

For the parallel program Q running on P compute nodes in Figure 1(b), in regard to the *i-th* $(0 \le i < P)$ 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:

$$T_{Sto}^{P} \approx T^{pnon} + N^{p} \bullet I^{pser} = T^{pnon} + \left(T^{pnon}/I^{pint}\right) \bullet I^{pser} \tag{2}$$

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:

$$S_{Sto}^{P}=\frac{T_{Sto}^{S}}{T_{Sto}^{P}} \tag{3}$$

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:

$$S_{Sto}^{P}=\frac{T_{Sto}^{S}}{T_{Sto}^{P}}\approx\frac{T^{snon}+\left(T^{snon}/I^{sint}\right)\bullet I^{sser}}{T^{pnon}+\left(T^{pnon}/I^{pint}\right)\bullet I^{pser}}$$
$$=\frac{T^{snon}\left(1+I^{sser}/I^{sint}\right)}{T^{pnon}\left(1+I^{pser}/I^{pint}\right)}=S_P\bullet\frac{1+M}{1+N}=S_P\bullet\frac{1}{1+\frac{N-M}{1+M}} \tag{4}$$

Let us define

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

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

$$S_{Sto}^{P}=S_P\bullet\frac{1}{1+O(P)} \tag{6}$$

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

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

Gustafson Storage Speedup

$$S_{Gustafson-Sto}^{P}=S_{Gustafson}\bullet\frac{1}{1+O(P)}=\left(f+P(1-f)\right)\bullet\frac{1}{1+O(P)} \tag{8}$$

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.

$$S_{Sto}^{P}=S_P \tag{9}$$

B. System Classification

First we make some symbols conventions. We denote $f(x)\succ g(x)$ if $\lim_{x\to\infty}f(x)/g(x)$ is ∞ and $f(x)\succeq g(x)$ if $\lim_{x\to\infty}f(x)/g(x)$ is a positive constant or ∞. Adversely, $f(x)\prec g(x)$ indicates $\lim_{x\to\infty}f(x)/g(x)$ is 0 and $f(x)\preceq g(x)$ indicates $\lim_{x\to\infty}f(x)/g(x)$ 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 $\lim_{x\to\infty}f(x)/g(x)$ is a positive constant, it denotes $f(x)=\Theta(g(x))$ or $g(x)=\Theta(f(x))$.

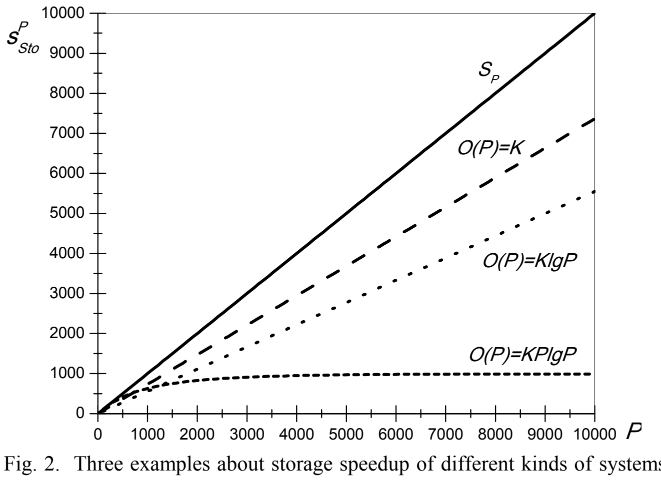


Fig. 2. Three examples about storage speedup of different kinds of systems based on $O(P)$.

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 $\lim_{P\to\infty}S_P=\infty$ runs on a W system. If $O(P)\preceq\Theta(1)$, the W system is considered to be a constant system. And if $O(P)\succ\Theta(1)$, 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 $O(P)=K\preceq\Theta(1)$, 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 $O(P)=KP\lg P\succ\Theta(1)$, the W system is an incremental system. This is different from above; the scalability of the application is bound by the I/O performance.

C. General Storage Speedup

In this subsection we generalize the storage speedup by considering not only the time cost introduced in the last subsection but also costup related to investment. Due to a powerful support provided by supercomputer, scientific exploration develops rapidly. Since more and more data need to be analyzed or calculated, I/O-intensive applications are becoming an important part of the scientific applications on the supercomputer. In order to meet the requirements of I/O-intensive applications, storage system of supercomputer need more large capacity and high performance equipment, and also more flexible and efficient software stack. All of these lead to more cost. The expansion of supercomputer system not only meets the requirements of applications, but also considers the cost-effectiveness of all the investment of manpower and money. So we want to find a metric to measure the cost-effectiveness of investment for supercomputer under the storage constraint.

We generalize a new speedup called general storage speedup to try to solve the problem. This generalization is increasingly important for modern petascale especially future exascale supercomputing systems as the storage systems employed can be more costly.

Based on the costup presented by D. A. Wood and Mark Hill [9], we introduced $costup_s$ to analyze the storage scalability which is more cost-effective related to various I/O architecture.

Definition 3 (Storage Costup). When program Q runs on a parallel system from one processor to P processors, the storage costup is defined as follows:

$$\cos tup_S=\frac{cost_P^S}{cost_1^S} \tag{10}$$

The cost could include not only the compute system cost but also the storage system cost. And the cost also include the

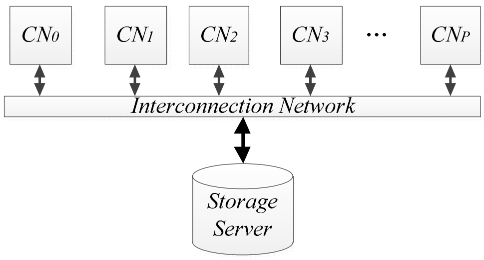


Fig. 3. Centralized I/O Architecture.

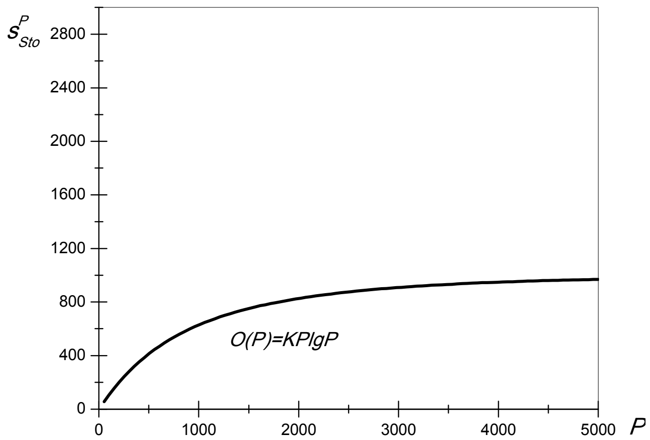


Fig. 4. Centralized I/O Architecture.

hardware and software cost. We assume that the compute system cost grows proportionally with the application scale *P*.

$$\mathrm{costup}_S = \frac{cost_P^{comp} + cost_P^{sto}}{cost_1^{comp} + cost_1^{sto}} = \frac{cost_1^{comp} \bullet P + cost_P^{sto}}{cost_1^{comp} + cost_1^{sto}} \quad (11)$$

Let us write C as $cost_1^{sto} / cost_1^{comp}$, M as $cost_P^{sto} / cost_1^{sto}$. So we have

$$\mathrm{costup}_S = \frac{P + MC}{1 + C} = P\frac{1 + (M/P)C}{1 + C} \quad (12)$$

Obviously, *M* represents storage investment trends which are closely related to system architecture, market variation, application requirements and so on.

Definition 4 (General Storage Speedup). When program *Q* runs on a parallel system, the general storage speedup achieved is defined as follows:

$$S_{Sto}^{GP} = \frac{S_{Sto}^P}{\mathrm{costup}_S} \quad (13)$$

According to (12), we have

$$S_{Sto}^{GP} = \frac{1 + C}{P(1 + (M/P)C)} S_{Sto}^P \quad (14)$$

Since *C* is a constant, the general storage speedup is determined by *P, M* and storage speedup.

M denotes the growth rate of the storage system cost with the compute system scaling. The general storage speedup allows us to study both the performance effect and cost effect of storage system to the whole parallel system.

IV. I/O Architecture Analysis

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

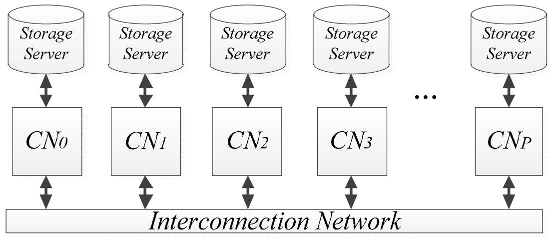


Fig. 5. Distributed and Parallel I/O Architecture.

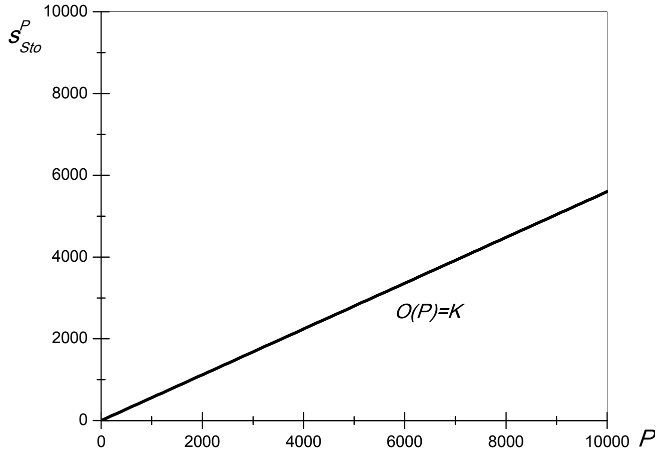


Fig. 6. A case about storage speedup which is not affected by the I/O performance related to distributed and parallel I/O architecture.

abbreviated as CDP I/O architecture). The analyses for each I/O architecture related to storage speedup and general storage speedup are as follows.

A. The Analysis of Storage Speedup

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

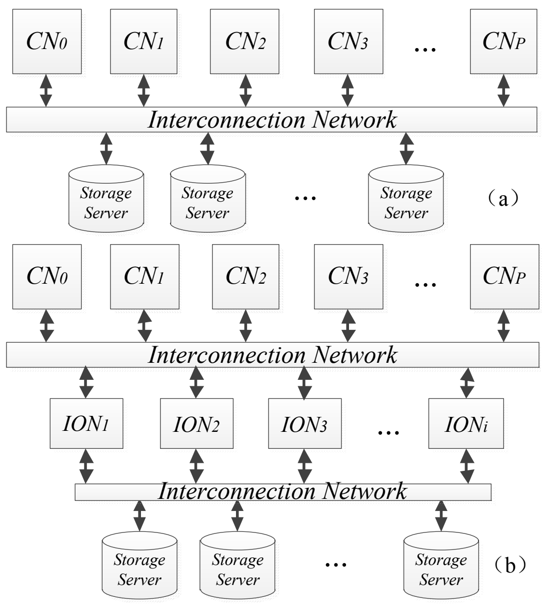


Fig. 7. Centralized, Distributed and Parallel Architecture.

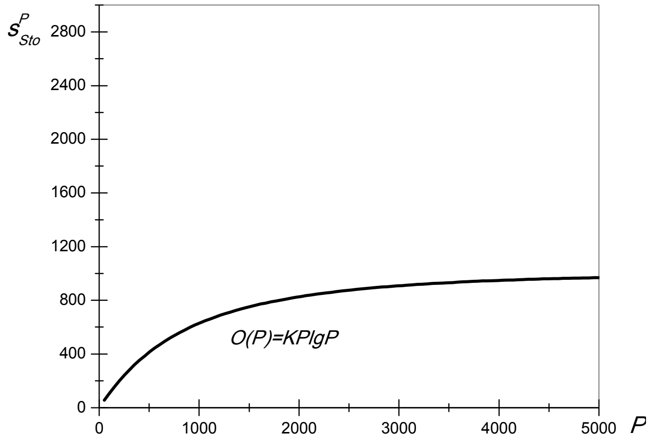


Fig. 8. A case about storage speedup which is constrained by the I/O performance related to centralized, distributed and parallel architecture.

storage system with centralized I/O architecture cannot scale out to more storage nodes, we obtain $O(P) \succeq \Theta(P)$. 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.

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 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.

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 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 $O(P) \succeq \Theta(P)$. 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

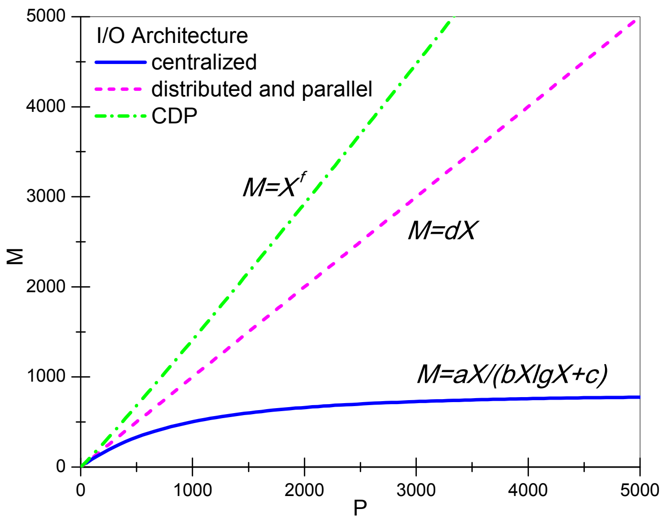


Fig. 9. Examples of changing law of parameter M under different I/O architectures when the storage system scaling with compute system. a, b, c, d and f are different constants.

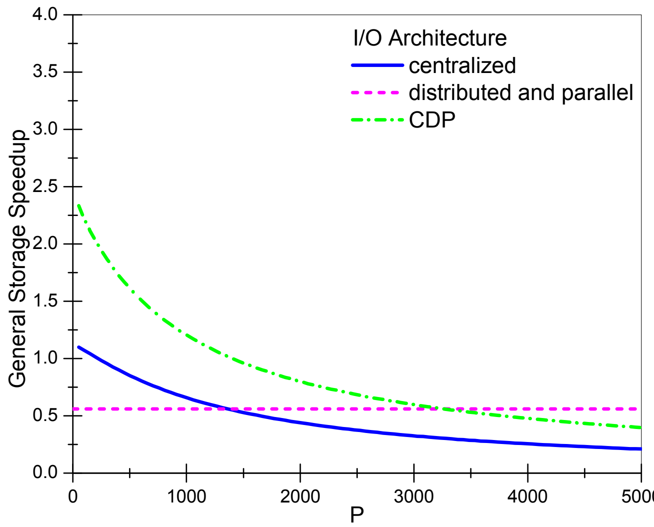


Fig. 10. Examples of changing law of general storage speedup under different I/O architectures when the storage system scaling with compute system.

performance is not obvious until the parallel application increases to a huge scale.

B. The Analysis of General Storage Speedup

Storage speedup is used to evaluate practical speedup of parallel application under the constraint of I/O system, while general storage speedup can be used to show how cost-effective when the storage system of supercomputer scaling. Since the market variation and the private cost details of system construction, there is no specific system to be described. Instead in this subsection we illustrate some examples of general storage speedup related to typical I/O architectures.

Figure 9 shows the examples of *M* function which is a key factor to reveal the variation ratio of storage cost with the scale of parallel application increasing. Due to a wide range of storage devices and complex price variability, these cases are only used to reveal the possible laws of *M* function under different I/O architectures. Different scaling methods need distinct types and quantities of hardware and software equipment, and this lead to different variation trends of *M*. *M* can be measured according to specific system at a specific time.

Figure 10 shows the examples of general storage speedup of different I/O architectures with the scale of parallel application increasing. Since the different extension methods of supercomputer storage system under different I/O architecture, general storage speedup reflects all kinds of cost-effective characteristics of storage systems. According to the definition, the larger value of general storage speedup represents a higher cost-effectiveness.

The examples in the figure 10 shows that the general storage speedup decreases slowly with the compute system and storage system scaling for the centralized I/O architecture and CDP I/O architecture, and the storage system with CDP architecture is more cost-effective than centralized I/O architecture. Since storage system scales with compute system proportionally under distributed and parallel I/O architecture, the general storage speedup is approximately to be a stable value. So, when the storage system expands to a larger scale, the system with distributed and parallel I/O architecture has better and more stable cost-effectiveness.

V. CONCLUSIONS

This paper introduces a new speedup metric called storage 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. Based on the storage speedup, a general storage speedup is generalized to evaluate the cost-effectiveness of the storage system by considering not only speedup but also costup. Through the case studies, the effectiveness of the two new speedup metrics 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 *Porc. 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.

[9] D. Wood, M. Hill, “Cost-Effective Parallel Computing”. *Computer*. vol.28, no.2, pp. 69-72, 1995.

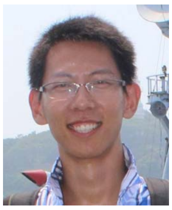
Guilin Cai is a doctor student in the College of Computer at National University of Defense Technology in China. She received the B.S. and M.S. degrees in computer science from National University of Defense Technology in 2005 and 2010, respectively. Her main research interests include high performance computing, cyber security and proactive defense.

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.

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.