I/O Workload Overview of the Applications on Intrepid Supercomputer

Pablo J. Pavan[1], Valéria S. Girelli[1], Jean L. Bez[1], Francieli Z. Boito[2], Philippe O. A. Navaux[1],

[1]Federal University of Rio Grande do Sul (UFRGS) – Porto Alegre, RS – Brazil
{pablo.pavan, vsgirelli, jean.bez, navaux}@inf.ufrgs.br
[2]Univ. Grenoble Alpes, Inria, CNRS, Grenoble INP, LIG, 38000 Grenoble, France
francieli.zanon-boito@inria.fr

Abstract

In this work, we analyzed the I/O workload of HPC applications on Argonne's Intrepid Blue Gene/P supercomputer, using Darshan logs. For our analysis, it was necessary to perform a data treatment to filter the information regarding the execution time, I/O time, among others. Initial results show that we had 26,034 applications executed in the year 2012, of which ten applications represent 35.36% of the total number of executions observed and spend 20% of their execution time performing I/O operations.

1. Introduction

In High Performance Computing (HPC) systems, supercomputers are the most significant example, several applications require considerable computational power. These applications often use large volumes of data and I/O operations are made in Parallel File Systems (PFS), where dedicated machines act as the data servers. The servers aim to receive requests from the computational nodes and to process them, accessing the storage devices. These I/O operations are a bottleneck for a growing number of applications due to the difference between the processing speed and the data access speed [1].

We can characterize the application's behavior to look for new ways to improve I/O operations. This characterization can be done by generating traces or I/O operations profiling, using some tool such as Darshan[1]. Darshan was developed by the Argonne Leadership Computing Facility (USA) and aims to characterize the application's I/O operations. It shows a representation of the application behavior, reporting information such as access pattern, time and number of operations, among other information.

In this work presents an analysis of the I/O workload of the applications executed in a supercomputer, by data collected through Darshan on the Intrepid Blue Gene/P supercomputer located in Argonne.

The remaining sections of this paper are organized as follows. Section 2 discusses related work. In Section 3 we present the details of the evaluation methodology. In Section 4 we address the results obtained from our analysis. Finally, the Section 5 we show the conclusion and the future work.

2. Related Work

Zoll et al. [5] studied a set of application-side I/O traces in an HPC environment (the ASCI cluster from the LLNL) using the Lustre parallel file system for storage. Their traces were obtained in 2003 with the *strace* tool, ranged from tens of seconds to half an hour, and included two scientific applications from the physics domain and three benchmarks generated with IOR (file-per-process, shared-contiguous and shared-strided). They concluded that a Markov model could not represent the request arrival rate of applications' I/O streams present self-similarity. They presented a stochastic model to predict I/O arrival rate.

Wang et al. [4] used the same traces aiming at characterizing the HPC I/O workload. They also analyzed files size and lifetime, showing 80% of the files have a size between 512 KB and 16 MB, and these files account for 80% of data stored in the file system. 60% of the files (50% of the bytes) stay in the system from 2 to 8 weeks. Their results for request size and inter-arrival time are somewhat specific to the traced applications and show they issue large numbers of small requests (from a few bytes to 1 MB) in small time intervals.

To motivate their work on cross-application coordination, Dorier et al. [2] used data from the Parallel Workload Archive, from the period between January and September 2009. They showed half the jobs on this platform used less than 2048 cores (1.25% of the machine). Through a simple optimistic model, they used the distribution of some concur-

1 http://www.mcs.anl.gov/research/projects/darshan/

rent jobs to show there is a high probability of having multiple applications concurrently performing I/O operations, even when applications spend as little as 5% of their execution time on I/O.

Luu et al. [3] examine the I/O behavior of thousands of supercomputing applications. They analyze the Darshan logs of over a million jobs over Intrepid and Mira supercomputers, at the Argonne Leadership Computing Facility (ALCF), and Edison, at the National Energy Research Scientific Computing Center (NERSC).

The authors draw several conclusions from these observations. First, they point out that every widely adopted I/O paradigm (file per process, shared file, subsetting I/O) is represented among the best-performing and worst-performing applications. Hence, the usage of a paradigm does not provide any guarantees in terms of performance. Regarding throughput, they demonstrate that almost a third of the jobs have an aggregated throughput of no more than 256MB/s. They also point out that over a third of the jobs spend more time in metadata operations than actually transferring data. Additionally, despite the existence of high-level parallel libraries, three-quarters of the jobs use only POSIX to perform I/O.

In the same way as the works related in this section, we seek to better understand the HPC application's behavior from their I/O operations, analyzing information collected about them. Differently from Zoll et al. and Wang et al., we used as a basis for our analysis the traces obtained transparently by the Darshan profiler, an approach also used by Luu et al. Unlike the other works, we have sought to obtain an overview of the I/O workload of the applications. In this way, such information can be used to make smarter decisions in various I/O optimization techniques such as I/O scheduling, I/O forwarding and reconfiguration of the stack.

3. Methodology

In the Intrepid Blue Gene/P supercomputer data was collected from different applications using different Darshan versions in the years 2010, 2012 and 2013. For this work we only use the observations of 2012, which were generated by versions 1.23, 1.24 and 2.0, totaling 36, 359 and 91,603 observations with each, respectively. We used only data from version 2.0 since previous ones did not capture a counter needed for our analysis.

Darshan saves data collected from applications in a binary format. In order to extract and transform the necessary data for our analysis, we used the *darshan-parser* tool, available with Darshan. It transforms the binary file into a text file, which contains all the counters related to the I/O operations collected by the profiler. The text-format file generated by the *darshan-parser* is organized into two sections. At the file's beginning, stored in the key-value pattern, data is described as the executed application, such as name, job identifier, runtime and number of processes. The rest of the text file contains the observations of each counter captured by Darshan, relative to each file opened by the application (*filehandle*), where each line follows a tabular format.

We divided the collection of relevant data for our analysis and the characterization of the application's I/O workload into stages.. In order to extract the counters, it was first necessary to extract relevant data (Step 1) from the text file generated by *darshan-parser*. For this, we chose to use the Python programming language, which presents libraries that facilitate data manipulation and allow us to easily create dictionaries from the data that were later saved in compressed JSON format in order to reduce the storage space. Subsequently, to perform a specific analysis, we use an application in Python that reads the JSON files generated by Step 1, extracting the relevant data for analysis, and writes a file in CSV format (Step 2), containing a summary of the execution or the information access intervals. We chose the CSV format because it is easily manipulated by the R language, with which we perform the final analysis of the data (Step 3).

4. Results

From the data of 2012, we first look at the behavior of the applications during the whole year and find the number of different applications that were submitted to the execution and applications with the highest number of executions. For this, we used the value of the *exec* collected by Darshan.

In order to understand the annual cluster behavior, in Figure 1 we show the relation between the I/O time (y-axis) and the execution time (x-axis) of all submitted jobs, where each point represents one execution. We color these points according to the I/O percentage that each job performed, where the yellow and the blue represent smaller and higher percentage, respectively. We note that the most executions does not exceed 45,000 seconds executing and 2,500 seconds performing I/O. Within this space, we have 85,710 jobs submitted, and of these, a total of 10,050 jobs have an I/O percentage above 50% and execution time below 4,609 seconds. With this, we can conclude that executions with a high I/O percentage do not run for long periods of time.

In order to exploit these applications that have significant use of I/O, we focused the analysis on the executions whose I/O percentage was above 50% and from these we selected those with highest numbers of submitted jobs. From this filter, we chose three applications. Table 2 presents the anonymized ID of the executable (exec) and the number of times it was observed.

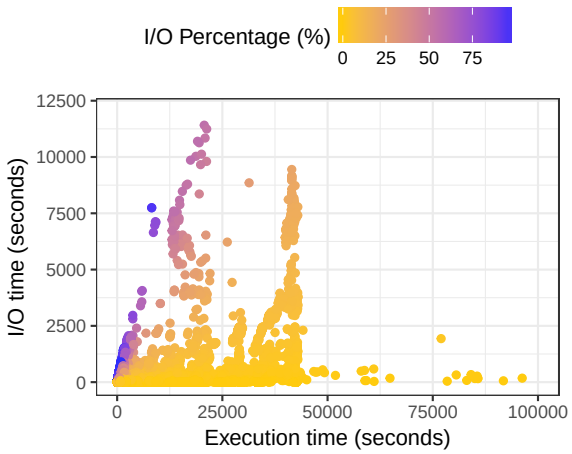

Figure 1. Annual behavior overview of the Intrepid supercomputer, relative to the percentage of time spent on I/O operations per job.

Exec (Anonymised)	Jobs with $> 50\%$ of the time in I/O operations
1176110786	980
1338247359	898
902685977	761
Total	2,639

Table 1. Applications with jobs that spend more than 50% of their execution time performing I/O operations.

Expanding the analysis for the exec *1176110786*, we noticed that it executed with 1,024 or 2048 MPI processes. This application was executed 980 times during the period from June 06, 2012 to December 18, 2012. We can see that the first executions used only 2,048 MPI processes, and it can be either the application's tests or the performance on other dates. After this period, from observation 104, dated July 9, 2012, we can observe a change in the application behavior, where the executions began to have a higher I/O operations and the use of 1,024 MPI processes. We can suppose these runs with 1,024 processes can indicate application testing with a new input set.

With 1,024 processess, we can describe one more conclusion about this application. Although transferring less data, the I/O percentage remains high. This scenario may indicate that the entry for 1,024 processes was smaller than the one used with 2,048 processes. However, with the using fewer processes, the execution time also increased, causing the same I/O percentage to be maintained.

For the application with exec *1338247359*, we had 1,051 observations during the period from March 21, 2012 to July 31, 2012. The I/O operations account for more than 50% of execution time in 898 observations. The application used different numbers of MPI processes over different executions. The highest occurrence of executions occurred between 1 and 100 processes and the processes number most used was 64, accounting for 441 observations. In cases about 1 to 100 processes, the maximum data transferred was 27.94GB and the maximum for the application was 135.88GB, using 485 MPI processes. We note that in this application the processes number reflects the data size used.

The application with exec *902685977* was submitted for execution in the period from October 23, 2012 to October 30, 2012, during which period there are 763 jobs. When analyzing this application, at the beginning of the period we had 60 executions with 64 processes, transferring a maximum of 18.05GB of data, and after that, the 701 executions with 32 processes that transferred a maximum of 9.02 GB, for the most part, behave similarly throughout the execution period.

There were also two executions using 128 processes, which transferred a maximum of 36.10GB. These executions can represent application tests since they were in smaller number, they did not repeat themselves, and they presented different behavior of the other executions.

Another analysis with the data was to find the ten most executed applications during the year. The total executions count of these ten applications was 32,535, which represents 35.36% of the total collected in the supercomputer this same year, showing that these applications are indeed significant in our analysis. With the objective to analyze these applications, which had the longest time spent in I/O operations, Table 2 presents the data on the jobs of these applications. The largest I/O time was observed in job *1538301448* relative to the application *1633035531*, with 2.27 hours of its execution time spent in I/O operations, representing 19.46% of the total execution time.

Besides, there are only three jobs out of the ten that have a low time in I/O operations, not exceeding 10% of the total execution time of each one. The remainder of the runs presents vast amounts of time spent with I/O, as can be seen from the job *2939212379* of the application *2425255765*, whose I/O time is the second largest of the realized executions. The time that this execution was performing I/O operations corresponds to almost 80% of the total execution time, totaling 49.45 minutes. The similar behavior we can be observed with job *556397787* of application *1338247359*, which has the third largest time spent on I/O operations and again corresponds to almost 80% of the ex-

Exec (Anonymised)	JOB ID	Execution time (seconds)	I/O Time (seconds)	I/O %	Data transferred (GB)
685531913	3182318857	1,370	3.12	0.22	0.003
1330277471	1330277471	1,314	536.63	40.84	5.69
931947437	2946033326	2,906	985.65	33.91	0.013
3069475893	1473997298	2,808	3.92	0.13	0.012
1074553177	1507238004	1,489	533.55	35.83	1.12
1648769576	1591638629	1,993	191.82	9.62	7.59
1633035531	1538301448	42,003	8,176.28	19.46	6,223.78
2425255765	2939212379	3,720	2,967.61	79.77	2.85
3475271559	567534836	1,512	310.14	20.51	2.25
1338247359	556397787	2,415	1,924.92	79.70	17.80

Table 2. The jobs that have spent the most time on I/O operations over of the ten applications identified as the most executed.

ecution time. In this way, it is clear to see how the ten most executed applications represent significantly the I/O pattern observed in the supercomputer.

5. Conclusion

In this work, we have analyzed the I/O workload of HPC applications executed in the Intrepid Blue Gene / P supercomputer of Argonne (USA). For this we use logs collected with the profiler Darshan, extracting those data relevant to our analysis.

When we analyzed the results of I/O workload found during the year 2012 on the Argonne supercomputer, we obtained 91,973 submitted jobs, which grouped by exec, resulted in 26,034 different applications. From this, we seek to analyze the applications following two approaches. In the first one, we looked for applications that had more than 50% of their time in I/O operations, and when we analyzed three of them, we could notice different behaviors, such as the use of different MPI process numbers, transferred data, execution period during the year and the form of access to your data. The other approach chosen was to analyze the ten most executed applications, representing 35.36% of the total number of executions observed. When we were observed these ten applications, for the most part, spend much of their execution time doing I/O operations, representing more than 20% of their execution time.

As future work we intend to extend this analysis of the characterization of the intervals for all the jobs of Argonne, seeking to group applications that have a similar behavior regarding their phases. Besides, we also intend to aggregate data from multiple applications to have a global view of the use of I/O in a supercomputer.

Acknowledgment

This research received funding from the Petrobras project, grant n. 2016/00133-9. It was also supported by PIBIC CNPq-UFRGS and PROBIC FAPERGS-UFRGS. This research used resources of the Argonne Leadership Computing Facility at Argonne National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357.

References

[1] J. L. Bez, F. Z. Boito, L. M. Schnorr, P. O. Navaux, and J.-F. Méhaut. Twins: server access coordination in the i/o forwarding layer. In *Parallel, Distributed and Network-based Processing (PDP), 2017 25th Euromicro International Conference on*, pages 116–123. IEEE, 2017.

[2] M. Dorier, G. Antoniu, R. Ross, D. Kimpe, and S. Ibrahim. Calciom: Mitigating i/o interference in hpc systems through cross-application coordination. In *Parallel and Distributed Processing Symposium, 2014 IEEE 28th International*, pages 155–164. IEEE, 2014.

[3] H. Luu, M. Winslett, W. Gropp, R. Ross, P. Carns, K. Harms, M. Prabhat, S. Byna, and Y. Yao. A multiplatform study of i/o behavior on petascale supercomputers. In *Proceedings of the 24th International Symposium on High-Performance Parallel and Distributed Computing*, pages 33–44. ACM, 2015.

[4] F. Wang, Q. Xin, B. Hong, S. A. Brandt, E. Miller, D. Long, and T. McLarty. File system workload analysis for large scale scientific computing applications. Technical report, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States), 2004.

[5] Q. Zoll, Y. Zhu, and D. Feng. A study of self-similarity in parallel i/o workloads. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–6. IEEE, 2010.