Proceedings of Supercomputing 94
Washington, D.C.
November 14-18, 1994

Dynamic I/O Characterization of I/O Intensive Scientific Applications

Barbara K. Pasquale and George C. Polyzos

Computer Systems Laboratory
Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093-0114

{bittel, polyzos}@cs.ucsd.edu

Abstract

Understanding the characteristic I/O behavior of scientific applications is an integral part of the research and development efforts for the improvement of high performance I/O systems. This study focuses on application level I/O behavior with respect to both static and dynamic characteristics. We observed the San Diego Supercomputer Center's Cray C90 workload and isolated the most I/O intensive applications. The combination of a low-level description of physical resource usage and the high-level functional composition of applications and scientific disciplines for this set reveals the major sources of I/O demand in the workload. We selected two applications from the I/O intensive set and performed a detailed analysis of their dynamic I/O behavior. These applications exhibited a high degree of regularity in their I/O activity over time and their characteristic I/O behaviors can be precisely described by one and two, respectively, recurring sequences of data accesses and computation periods. Key Words: empirical I/O behavior, supercomputer applications

1.0 Introduction

Research investigations in many scientific disciplines require enormous computational power and, consequently, have fueled the need for larger and faster computer systems. However, combined with the ability to perform GFLOPS computations, scientific research also requires the ability to store, access and generate massive data sets. The growing I/O needs of scientific applications and the existing performance gap between processing power and the I/O system has intensified I/O demand within the local system and communication requirements across networks [17]. As a result, attention is now focused on research and development in the area of high performance I/O systems [5, 8, 10, 14]. Lower level I/O system improvements have been achieved by partitioning data across large disk arrays, enabling parallel data access [14], yet many issues remain regarding interconnection networks, operating system support, and file system structure for high performance I/O systems. Finding effective solutions to potential I/O bottlenecks in high performance systems will require an integrated approach [6]. I/O issues must be addressed at all system levels, including applications, system software and architectures.

This work is funded in part by grants from DEC, the U.C. Micro program, and the Sequoia 2000 Project.

We focus on high performance I/O at the application level. Specifically, we are interested in isolating an I/O intensive set of scientific applications from which a small subset of representative applications can be selected and their dynamic I/O behavior characterized. Recent studies indicate that scientific applications have a high degree of regularity in I/O resource usage (e.g., stable I/O rates across different execution times, consistent main memory usage, cyclical patterns of I/O activity, sequential file accesses, and fixed-sized data transfers) [9, 10, 11]. Such regularities can be exploited to improve I/O performance at all levels, but first must be characterized and quantified across a wide spectrum of applications.

The research presented in this paper describes our efforts in studying the dynamic I/O behavior of two scientific applications representative of the most I/O intensive applications in a typical high performance workload. We observed a Cray C90 workload and isolated the I/O intensive population. We analyzed this population at two different levels [3]: the low-level physical resource usage and the higher-level functional composition of applications and scientific disciplines. The results of this analysis allowed us to describe the I/O intensive population in terms of individual applications, scientific disciplines, and I/O demands, i.e., volume and rate. From this description, we selected two interesting applications and studied their dynamic I/O behavior. In the sections that follow, we present our research methodology, workload characterization study, selection of an I/O intensive set, and the dynamic I/O analysis of the two selected scientific applications.

2.0 Methodology

To isolate a representative set of I/O intensive applications, we first conduct a general workload characterization study. This allows us to observe the characteristic features of the workload and to focus our attention on one important component, the *private user applications*. We regard *private user applications* as programs whose availability is limited to a single user or a small group of users. Many of these private user applications truly represent the cutting edge of high performance computing. Specifically, these

ISSN 1063-9535.

applications require the fastest and largest resources available due to (1) the length and complexity of the problem being investigated, and (2) the data access and storage requirements in main memory and on local disk.

Using only the private user component of the workload, we then analyze all distinctly named applications. In this analysis we consider the respective scientific discipline, execution frequency, CPU time, and two I/O metrics: (1) the number of total bytes transferred by the application and (2) the *virtual I/O rate*, i.e., the number of bytes transferred per CPU second. The *virtual I/O rate* is an important metric because it allows us to assess the relative I/O intensity of an application while providing some basic isolation from the details of the underlying architecture and surrounding workload. Based on *virtual I/O rate*, we produce a rank ordering of the applications and successively calculate each application s contribution to total workload bytes transferred. Each successive application in the I/O rate ordering is then added to the I/O intensive set until there is practically no change in the set s cumulative contribution to total workload bytes transferred (See Figure 1). At this point, we have isolated the most I/O intensive applications and have a description of their individual and cumulative resource usage.

Next we perform a functional level characterization of the I/O intensive set with respect to scientific discipline. Each application is mapped to its respective scientific discipline and resource usage totals are calculated for each discipline. This not only provides a system-independent description of the major I/O disciplines [3], but also serves to highlight the demands particular scientific disciplines place on system I/O. By combining the low-level description of physical resource usage and the higher level functional composition of applications and scientific disciplines [1, 2, 4, 16], we define the most I/O intensive applications and characterize the relationship between the individual applications, the resources they consume, and the scientific disciplines they represent.

To investigate the underlying sources and characteristic I/O activity producing such high I/O demand, it is necessary to observe the run-time behavior of individual applications. We have made significant progress in characterizing the dynamic I/O behavior of scientific applications and present two applications which are representative of our I/O intensive set. Through software instrumentation of the operating system and run-time I/O library, trace records detailing I/O events are collected as the application executes. These I/O trace records contain the time and type of I/O event, I/O transfer size, and file descriptor number. Although both wallclock event time and CPU event time are available for I/O activity, we restrict our analysis to CPU event time or the *virtual time* of I/O events. Considering I/O with respect to virtual time reveals the I/O activity as embodied in the source code, eliminating the effects of the system workload. Aggregating I/O events over one second intervals of virtual time, we construct time series of the total number of I/O transfers performed and the total number of bytes transferred per unit time. At this resolution we can observe the general patterns of recurring I/O. We also produce microsecond resolution times series of I/O. At this resolution, we can observe detailed I/O activity in terms of individual file accesses, transfer sizes, and times between the I/O events. Together, these time series provide a detailed characterization of the application s dynamic I/O behavior. Extracting out the recurring cycle of I/O activity, we obtain the dynamic I/O profile of the application.

3.0 Workload Characterization Study

The workload observed for this study was that of the San Diego Supercomputer Center s (SDSC) Cray C90. SDSC is one of the five U.S. National Science Foundation sponsored Supercomputer Centers. The SDSC workload spans all major scientific disciplines, including materials science, physics, biochemistry, atmospheric science, chemistry, astronomy, oceanography, and electrical engineering. The user community consists of over 2500 academic researchers and industrial partners. The Cray C90 can be considered the workhorse of the Center and supports the most diverse workload. Each of the eight C90 processors has a cycle time of 4.2 nanoseconds, producing a peak computational speed of 7.6 GFLOPS. There is 1 GB of main memory (128 million 64-bit words) and a 4 GB SSD used principally for caching frequently accessed disk files and for very fast swapping of jobs in and out of main memory. The broad base of this system s workload [7], its volume of I/O, and the scientific significance of its applications, make it an important and appealing environment to study.

Measurements of resource usage were obtained from the Cray System Accounting (CSA) utility [19] and collected over a one-month period during January 1994. On a per process-basis, the CSA utility records resource usage through the use of kernel probes and creates a process account record that summarizes the total resource usage. The record includes: process name, identification numbers (job, process, user, account), start and end times, system and user CPU time, total number of bytes transferred, number of logical I/O requests, number of physical I/O requests, the memory high water mark, per CPU connect times for multitasked jobs, and I/O wait times. The necessary data for our static I/O analysis of the workload was obtained directly from the individual fields of the process account record (i.e., process name, identification numbers, total number of bytes transferred) with the exception of total process CPU time. To obtain total process CPU time we added together the recorded clock tick values of system CPU and user CPU time. Although this measure of total process CPU time is not completely isolated from the effects of the system workload, the degree of approximation achieved is adequate for the purpose of this general analysis.

The data collected for January 1994 was separated into two categories: *system* and *user*. *System* represents processes that are available to all users whereas *user* represents processes that are limited to a single user or a small group of users. Table 1 shows how each component contributes to the overall workload resource usage. As can be noted from the numbers of Table 1, the *user* component of the workload has a tremendous impact on total resource usage. Even though it represents an extremely small portion of all executed jobs (4%), it accounts for 93% of total CPU time and 96% of total bytes transferred.

We compared these results to our previous study of SDSC s Cray YMP workload of February 1992 [11]. The characterizations are extremely similiar with the exception of cumulative bytes transferred. In February 1992, cumulative bytes transferred for the entire workload was in excess of 47TB and user processes accounted for 88% of total workload bytes transferred. For January 1994, user processes account for 96% of total workload bytes transferred, an 8% increase. The more important difference, though, is the change in the cumulative bytes transferred total for the entire workload. In January 1994 cumulative bytes transferred for the entire workload is in excess of 81TB, an increase of 37TB from the February 1992 workload. This increase is explained, at least in part, by the increase in CPU performance of the C90, estimated to be 1.5 - 2 times faster than the YMP. However, these same results indicate the increasing demand imposed on the I/O system.

Table 1: January 1994 Resource Usage

	Number of Processes Executed	Number of Distinct Processes	Cumulative CPU Time (s)	Cumulative Bytes Transferred (MB)
System	5,952,082 (96%)	327 (14%)	1,503,473 (7%)	3,064,431 (4%)
User	253,338 (4%)	2,080 (86%)	18,739,660 (93%)	78,640,155 (96%)
Total Workload	6,205,420	2,407	20,243,134	81,704,587

3.1 Selection of I/O Intensive Applications

To extract the I/O intensive applications from the private user component of the workload, we rank ordered all processes by their virtual I/O rate. Using the cumulative CPU time and bytes transferred across all executions of an individual process, we successively calculated each process s contribution to total workload resources. Figure 1 shows the top I/O ranked processes cumulative contribution to total workload resources. (It should be noted that all processes considered in the I/O rate ordering have average CPU execution times of 100 seconds or more. This threshold was determined in [11] as a result of the observation that short jobs do not have significant impact when considering their contribution to total workload bytes transferred.)

The Cumulative Contribution to Total Workload Bytes Transferred graph in Figure 1 clearly delineates the I/O intensive processes. In fact at the knee of the curve, we observe that the top 80 ranked user processes cumulatively contribute to 88% of the total number of workload bytes transferred while only utilizing 33% of total workload CPU time. This small set of I/O intensive processes represents less than 4% of all distinctly named user processes present in the January 1994 workload. (Recall from Table 1, there are 2,080 distinctly named user processes in the January 1994 workload which are responsible for 96% of total system bytes transferred and 86% of CPU time.)

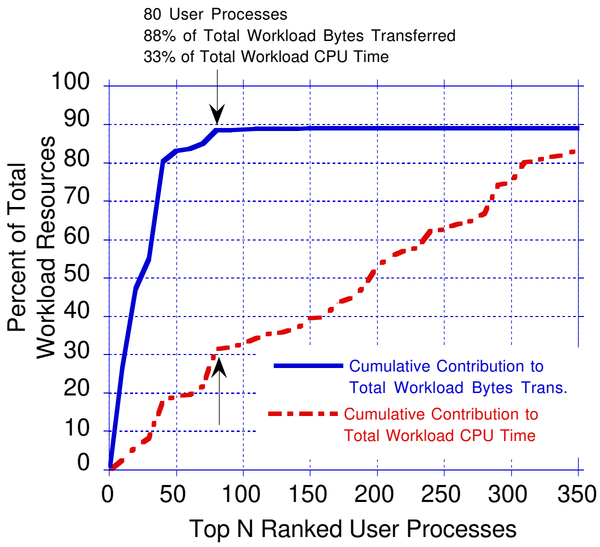

Figure 1: User Processes' Contribution to Workload Resource Usage

Isolating this I/O intensive set is a significant result in our analysis. However, the process-level accounting data does not provide any direct information indicating whether a process is a single-process application or one process of a multiple-process application. In the case where an application is a single process, application resource usage is given by the process resource usage. If an application is comprised of many processes, application resource usage is the cumulative resource usage of all processes. However, in many instances, there is a predominant process whose resource usage represents the whole application. Since we are interested in characterizing I/O behavior at the application level, it is necessary to make this distinction and investigate whether these I/O intensive processes represent single-process applications or combine to form multiple-process applications.

Utilizing the *process id* and *parent process id* fields of the process accounting record, we were able to relate each process in the I/O intensive set to its respective ancestors and decendants and construct the family hierarchy containing the I/O intensive process. We observed the following: (1) Forty-eight of the I/O intensive processes are single-process applications. These processes are invoked at the login-shell level and do not spawn any child processes. Thus, application resource usage is the single I/O intensive process resource usage. (2) Twenty-three I/O intensive processes are the parent processes of multiple-process applications. Child processes spawned by the I/O intensive parent (e.g., `sh`, `date`, `cat`, `expr`) have insignificant individual as well as cumulative resource usage and, therefore, the resource usage of the I/O intensive parent process is representative of total application resource usage. (3) Nine I/O intensive processes are child processes of a multiple-process application. The application or parent process, e.g., `xg92`, spawns an I/O intensive child process like `19999.ex` or `1801.ex`. In these cases parent process resource usage is insignificant compared to the resource usage of the child process and therefore, the resource usage of the I/O intensive child process is representative of total application resource usage. Based on this analysis, we conclude that the named processes in the I/O intensive set and their resource usage are representative of I/O intensive applications and will refer to them as applications.

A complete listing of these I/O intensive applications ordered by average virtual I/O rate is given in [13]. The listing also provides the execution frequency, scientific discipline, average CPU time, and average bytes transferred for each application. Figures 2 and 3 detail the actual distributions of average virtual I/O rate and average CPU time for this I/O intensive set.

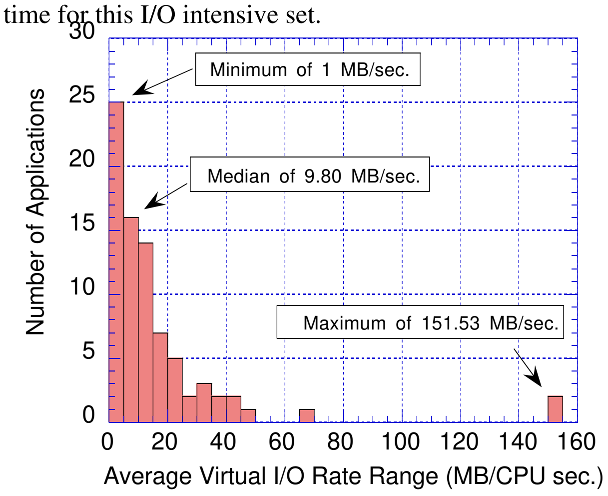

Figure 2: Average Virtual I/O Rate Distribution

We now provide a description of the applications having the maximum, median, and minimum I/O rates and CPU times. The chemistry application, `eiglz1.x`, has the maximum I/O rate of 151.53 MB/s and has an average CPU time of 12 minutes. Another chemistry application, `19999.ex` (or `xg92`), has the median I/O rate of 9.80 MB/s and has an average CPU time of 38 minutes. The application, `rmat.x`, from the Block Grant category, has the minimum I/O rate of 1 MB/s and has an average CPU time of 5 minutes. Analogously, the chemistry application, `perci2.x`, has the maximum average CPU time of 5.2 hours and has an I/O rate of 1.93 MB/s. The atmospheric sciences application, `camel12`, has the median average CPU time of 11 minutes and has an I/O rate of 1.84 MB/s. Finally, another atmospheric sciences application, `jul.x`, has the minimum average CPU time of 110.16 seconds and has an I/O rate of 3.80 MB/s.

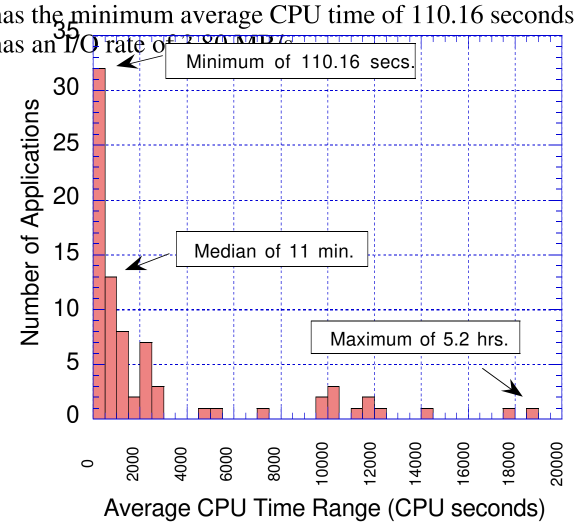

Figure 3: Average CPU Time Distribution

We compared the top 80 I/O ranked applications with those from the February 1992 analysis. Twelve distinctly named applications appeared in both rank orderings, as well as an atmospheric sciences application having the name of the general form, `cgcm.`*suffix*. Five `cgcm.`*suffix* applications appeared for February 1992 and 13 appeared for January 1994.

We also compared the cumulative resource usage totals for the two top 80 ranked orderings. (The actual table of values can be found in [13].) With the exception of the top ranked application, there is a general increase in both cumulative CPU and cumulative bytes transferred from the rank ordering of February 1992 to that of January 1994. Cumulative CPU time increased by a factor of 1.8 to 2.6 and cumulative bytes transferred increased by a factor of 1.3 to 2.0. Considering the minimum I/O rate of the top 1 through the top 60 groups, we observe that January 1994 is consistently higher. Even though there is an overlap in applications between the 92 and 94 rank orderings, as noted above, this overlap is limited and therefore we cannot draw any definitive conclusions at this time. However, it is obvious that the top I/O applications, even though not necessarily different or more I/O intensive (i.e., increase in data set size commensurate with increased processing speed), are placing increased demands on the I/O subsystem.

3.2 Functional Characterization

A workload can be characterized at different levels. The most common and important are: the physical, the virtual and the functional levels [3]. At the physical level, the description of the workload is architecture-dependent and represents the consumption of hardware and software resources (e.g., CPU time, memory). The virtual level describes the logical resources (e.g., number and type of code statements) and is less architecture dependent and closer to the user s point of view. At the highest level, a functional level description characterizes the workload in terms of the functions it performs and the functions performed are system independent. For example, the type of application (e.g., electrical engineering, chemistry, biology) and the type of computation or algorithm implemented (e.g., animation, simulation or FFT s, linear system solvers).

Using the scientific discipline information associated with each application, we performed a functional level characterization of the I/O intensive set, namely the top 80 ranked I/O applications. Characterizing the I/O intensive set with respect to scientific discipline provides us with a system-independent description of the major I/O consumers. Due to the nature of their research investigations, different scientific disciplines may perform specific computations. In turn, these discipline specific computations may result in particular patterns of I/O activity: the type of I/O (e.g., read, write), the purpose for I/O (e.g., data staging, checkpointing, scratch file), and the volume of I/O. Conversely, different scientific disciplines may employ the same or similar computational methods. Therefore, focusing first on major I/O disciplines and then on their computational methods and resulting I/O activity will provide a broad understanding of scientific application I/O behavior in general.

The results of our functional level characterization combined with each discipline s resource usage statistics are given in Table 2. The disciplines shown in this table are ordered by their I/O contribution, cumulative bytes transferred and percentage of bytes transferred. At the highest level of inspection, we note that 12 distinct disciplines are represented in the I/O intensive set as well as two multidiscipline categories. Industrial Partner and Block Grant are two general categories used in SDSC s internal accounting system and represent application executions for many different disciplines. (Unfortunately, no further information by scientific discipline is available for applications in these categories.) Within the I/O intensive set, atmospheric sciences ranks number 1, representing 27.39% of the I/O intensive set bytes transferred. With respect to the entire workload, the total number of bytes transferred by atmospheric sciences applications in the I/O intensive set represent 24.21%. In contrast, the last four listed disciplines, physics, biology, electrical engineering, and oceanography, make no significant contribution to bytes transferred either within the I/O intensive or with respect to the entire workload.

Considering the other attributes of the scientific disciplines (e.g., total number of executions, maximum I/O rate for discipline, CPU time), we observe some interesting and more complicated relationships. Although atmospheric sciences ranks number one in terms of I/O and CPU time (i.e., 15.96% of total CPU time within the I/O intensive set and 5.01% with respect to the workload) and has a maximum I/O rated application that is ranked fourth in the I/O intensive set, atmospheric sciences is ranked sixth in total number of executions. On the other hand, physics is ranked number three for total number of executions, yet has no influence on bytes transferred and a small influence on CPU time (i.e., 2.29% of total CPU time within the I/O intensive and 0.72% with respect to the entire workload).

This type of analysis is tedious, but in many regards must be approached at some level of detail. As described

Table 2: I/O Ranking of Scientific Disciplines

Scientific Discipline	Total Number of Execs.	Cumulative CPU Time (secs.)	% of CPU Time I/O Set & Wkld		Cumulative Bytes Transferred (MB)	% of Bytes Transferred I/O Set & Wkld		Max. I/O Rate for Discipline (MB/sec.)
Atmospheric Sci.	861	1,014,089.53	15.96	5.01	19,777,397.53	27.39	24.21	49.62
Industrial Partner	1915	837,514.81	13.18	4.14	17,490,761.10	24.23	21.41	151.53
Materials Science	742	486,117.24	7.65	2.40	12,127,171.20	16.80	14.84	32.61
Chemistry	1042	741,873.16	11.68	3.66	8,528,873.69	11.81	10.44	20.94
Block Grant	969	788,584.85	12.41	3.90	4,994,781.11	6.92	6.11	34.11
Chemical Eng.	293	183,918.44	2.90	0.91	789,434.62	1.09	0.97	11.54
Mechanics	1650	345,474.51	5.44	1.71	202,013.47	0.28	0.25	9.21
Education	2	8,034.37	0.13	0.04	85,171.46	0.11	0.10	10.57
Biochemistry	308	108,048.14	1.70	0.53	33,624.68	0.05	0.04	37.55
Astronomy	64	464,258.44	7.31	2.29	8,226.63	0.01	0.01	unknown
Physics	1547	145,771.64	2.29	0.72	2,471.26	0.00	0.00	unknown
Biology	1	94.98	0.00	0.00	67.65	0.00	0.00	.71
Electrical Eng.	308	12,301.85	0.19	0.06	15.34	0.00	0.00	unknown
Oceanography	6	60.08	0.00	0.00	5.35	0.00	0.00	unknown

in [1], the construction of a representative workload model relies on the number of elements of each particular class being either (1) proportional to the number of elements in each class, (2) a function of the number and weight of elements in each class, or (3) proportional to the influence of each class on performance. The elements, once selected, can be taken from the workload directly without manipulations (i.e., natural workload elements) or they may be designed and implemented independent of the real workload (i.e., artificial workload elements) [2]. We believe it is important to use the real or natural applications, but we cannot hope to monitor the run-time behavior and analyze the resulting trace of every single application in the I/O intensive set. Therefore, we must meaningfully select a small subset of those applications which exemplify the major I/O disciplines with respect to the type of computations performed, the significance of the underlying research, and how often the application is executed.

4.0 Dynamic I/O Analysis

Based on our characterization of the I/O intensive set, we selected two representative applications from Atmospheric Sciences to begin our investigation of dynamic I/O behavior of scientific applications. The first application, *R*, is an analysis application used in the Climate Research Division at Scripps Institution of Oceanography [15]. *R* is used for the verification of a fine-grained regional atmospheric climate model over the Western United States with measured large-scale, global climate and weather station site data. The second application, *OGCM*, is a simulation application used in the Department of Atmospheric Sciences at the University of California, Los Angeles [18]. *OGCM*, the ocean global circulation model, simulates patterns in water temperature, velocity, etc. in the Pacific Ocean basins.

R and *OGCM* serve to highlight different characteristics of scientific application I/O. *R* uses 6 input files, reading a total of 750 MB, and two output files, writing a total of 18 MB. Aggregate run-time statistics for *R* reveal that bytes transferred for read I/O represents 97.6% of all application I/O activity and the average virtual I/O rate is approximately 0.9 MB/CPU sec. *OGCM* uses 5 input files, reading a total of 0.6 MB and 7 output files, writing a total of 281 MB. The aggregate run-time statistics for *OGCM* reveal that bytes transferred for write I/O represents 99.79% of all application I/O activity and the average virtual I/O rate is approximately 9.25 MB/CPU sec. Aggregate run-time statistics such as these can be used to assess the relative intensity of I/O, however, we are more interested in observing how I/O activity occurs over time, namely, the specific times at which I/Os occur, the type of I/O performed and files accessed, and the number of bytes transferred for each I/O event. Constructing a time series of this I/O activity reveals any recurring patterns of I/O and allows us to define the characteristic I/O behavior for the application.

4.1 R's Dynamic I/O Behavior

We monitored the run-time I/O behavior of *R* with a typical data set of atmospheric model climate data. Aggregating I/O events over one second intervals of virtual time, we constructed *R* s virtual time profile with respect to I/O transfers. The entire profile is shown in Figure 4.

At the one-second resolution, this profile clearly depicts the characteristic I/O behavior. *R* has two distinct phases of processing: (1) a short, initial phase with frequently occurring, repeating bursts of I/O and (2) a longer, main phase, during which I/O activity repeats at larger intervals and is composed of two individual periods of I/O. Although both phases show a regular and repeating pattern of I/O activity, there is some variation between I/O bursts in each phase. Between theses two phases, there is a unique spike of I/O.

As the virtual time profile shows (Figure 4), the repeating pattern of I/O activity in the initial phase is a single, large burst of I/O. There are 30 such cycles, repeating approximately every 6 seconds. The detailed view of the initial phase I/O cycle is shown in Figure 5. After four seconds

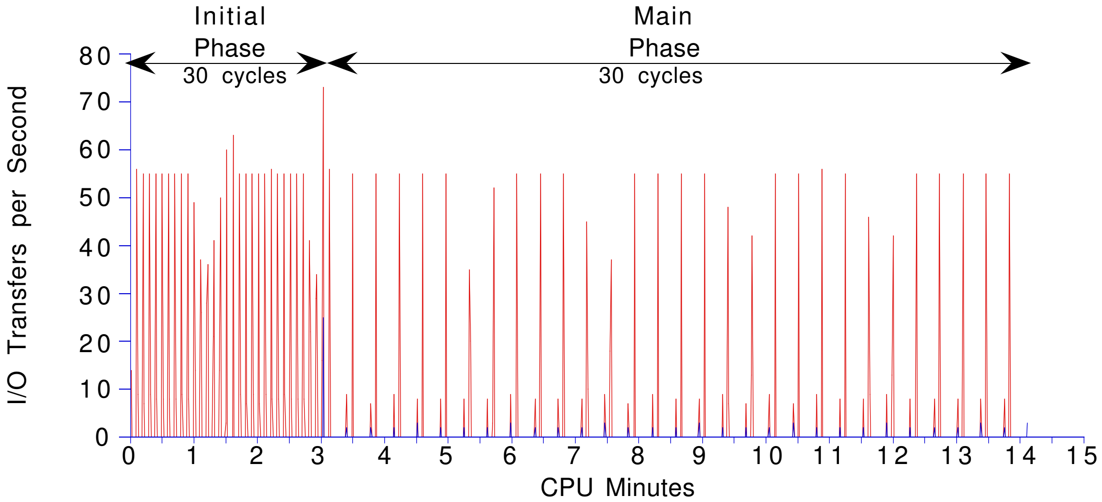

Figure 4: R's Virtual Time I/O Profile (1 second resolution)

of computation, there is a two-second burst of I/O activity. In the first second, 55 read I/Os are performed, transferring 10.8 MB, and in the subsequent second, 9 read I/Os are performed, transferring 1.7 MB. The pattern then repeats.

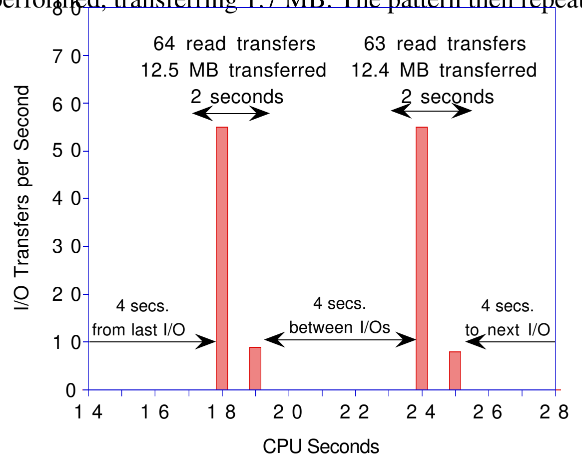

Figure 5: Initial Phase VIrtual Time I/O Cycle (1 second resolution)

The main phase has a similar repeating pattern of I/O but consists of two separate periods: a short burst of low I/O activity and a short burst of high I/O activity. There are 30 distinct cycles which repeat approximately every 22 seconds. Figure 6 details the main phase I/O cycle. After 16 seconds of computation, 8 read I/Os, transferring 1.5 MB, and 3 write I/Os, transferring 0.5 MB, are performed in the same one-second period of time. Then, following 4 seconds of computation, there is one second of read I/O, with 55 I/Os transferring 10.08 MB. The pattern then repeats.

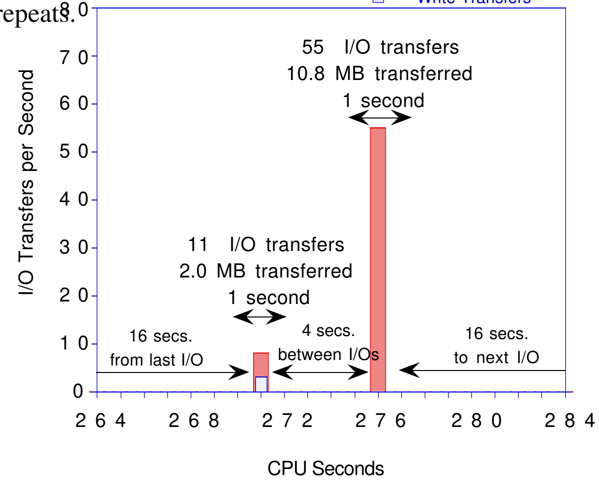

Figure 6: Main Phase Virtual Time I/O Cycle (1 second resolution)

The diagram in Figure 7a is a representation of the initial phase I/O cycle at the microsecond resolution. After 0.65 seconds, the first period of I/O begins. During the next 2,097 microseconds, 3 read I/Os to file 5 are performed followed by 6 read I/Os to file 6. The number of bytes transferred in each read I/O is exactly 196,608. (196,608 B is equivalent to 48 blocks of 512 64-bit words.) Computation resumes for 5.13 seconds and then the second period of I/O begins. During this 279,389 microsecond period, all 55 read I/Os are to file 4, each transferring 196,608 bytes of data. The entire cycle then repeats. Considering all 30 initial phase cycles at this level, we observed the following characteristics. In the period of read I/O to files 5 and 6, transfers complete within one microsecond and the time between successive transfers averages 277.19 microseconds. The first computation takes an average of 0.65 seconds. During the second period of read I/O to file 4, transfers also complete within one microsecond and the time between successive transfers averages 5082.01 microseconds. The second computation takes an average of 5.13 seconds. Statistics summarizing interevent times and computation periods can be found in [13].

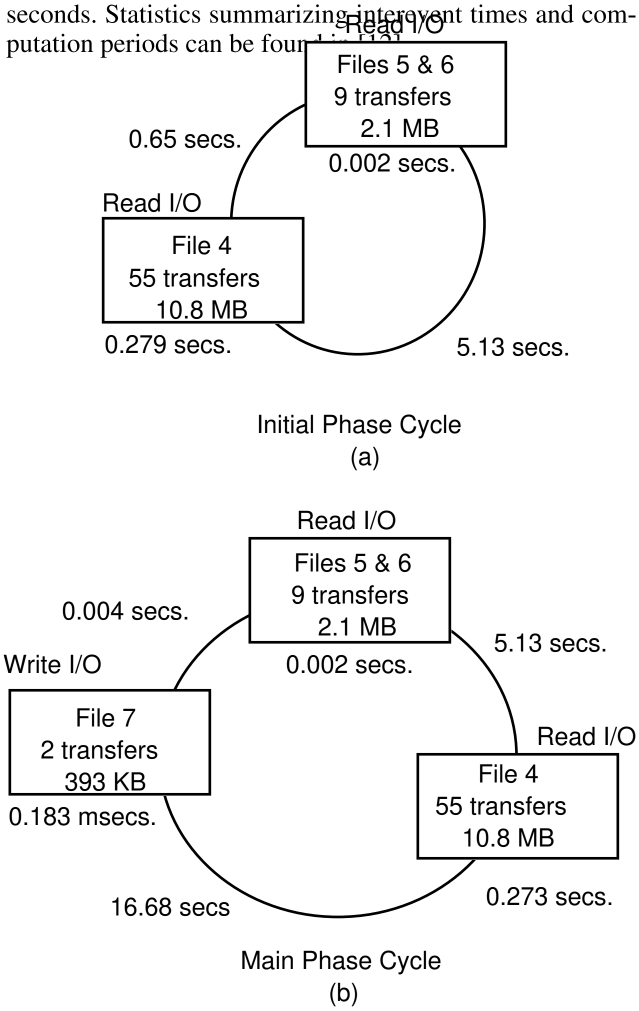

Figure 7: R's Dynamic I/O Profile

The diagram in Figure 7b is a representation of the main phase I/O cycle at the microsecond resolution. After 16.68 seconds of computation, two write I/Os, each transferring 196,608 bytes of data, are performed to file 7 in 183 microseconds. Following a period of 4,217 microseconds, 3 read I/Os to file 5 are performed followed by 6 read I/Os to file 6 in 2095 microseconds. The number of bytes transferred in each read I/O is also 196,608. Computation resumes for 5.13 seconds and then the second period of I/O begins. During this 273,200 microsecond period, all 55 read I/Os are to file 4, each transferring 196,608 bytes of data. The cycle then repeats. Considering all 30 main phase cycles, we observed many characteristics similar to those in the initial phase. In the periods of write I/O to file 7 and read I/O to files 5 and 6, only one transfer completes within one microsecond. The average interevent time for the write I/Os is 186.20 microseconds and for the file 5 and 6 read I/Os it is 279.15 microseconds. The brief period of computation after the write I/O to file 7 averages 4277 microseconds. The next computation phase follows the read I/O to files 5 and 6 and lasts an average of 5.16 seconds. The second period of read I/O to file 4 is consistent with the corresponding period in the initial phase, with an average interevent time of 5107.55 microseconds. The second computation phase follows the read I/O to file 4 and lasts for an average of 16.70 seconds. Additional statistics on interevent times and computation periods can be found in [12].

4.2 OGCMÕs Dynamic I/O Behavior

OGCM is executed with a set of run-time parameters which regulate the degree of resolution in the simulation process. As a result, total CPU time and I/O volume are commensurate with simulation granularity. We monitored the run-time I/O behavior of a basic version of *OGCM*, one that uses a 3-dimensional grid to simulate ocean basins at 9 depth levels and 1 x 1 degree spacing. In our view, using this basic version does not diminish the importance of the isolated characteristic I/O behavior.

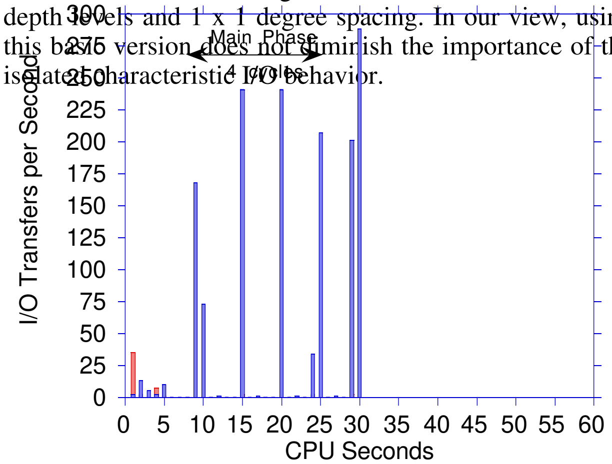

Figure 8: OGCM's Virtual Time I/O Profile

(1 second resolution)

To obtain *OGCM* s virtual time profile with respect to I/O transfers, we again aggregated I/O events over one second intervals of virtual time. This profile is shown in Figure 8. At the one-second resolution, we observe that *OGCM* has three phases of processing: (1) a short, initial phase with frequent, but low levels of I/O activity, (2) a longer, main phase with large bursts of I/O activity repeating at larger intervals, and (3) a short, final phase with a single burst of high I/O activity.

Our analysis focused on the main phase, where there is a clear repetitive pattern of the I/O activity. There are 4 such cycles of I/O activity, repeating approximately every 6 seconds. The detailed view of the main phase cycle is shown in Figure 9. After two seconds of computation, there are two seconds during which I/O activity is present. In the first second, 1 read I/O is performed, transferring 220 B, and 168 write I/Os are performed, transferring 32.83 MB. In the next second, 73 write I/Os are performed, transferring 14.27 MB. After one second of computation, there is 1 read I/O, transferring 13B, followed by two seconds of computation. The basic pattern then repeats. As Figure 9 shows, in the second cycle in the series the single read I/O of 220B and all 241 write I/Os occur during the same 1-second interval of time.

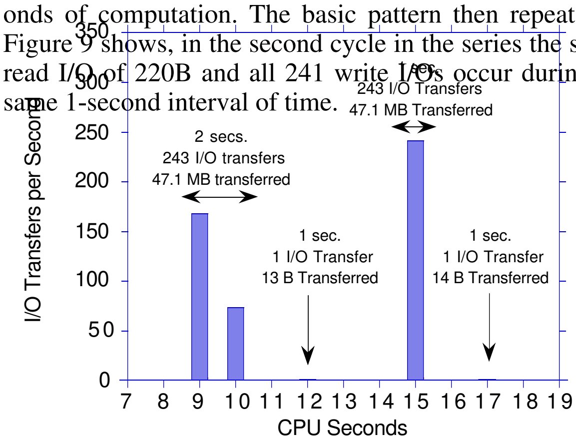

Figure 9: Main Phase Virtual Time I/O Cycle (1 second resolution)

The diagram in Figure 10 is a representation of the main phase I/O cycle at the microsecond resolution. The main phase cycle begins with 1 read I/O to file 9, transferring 220 B. After 1,748 microseconds of computation, 231 write I/Os to file 10 occur during a period of 104,532 microseconds. The number of bytes transferred in each write I/O is exactly 196,608. Computation resumes for 508,779 microseconds and then a second period of I/O begins. During this 12,781 microsecond period, all 9 write I/Os are to file 8, each transferring 196,608 bytes. Computation resumes for the next 2.13 seconds, followed by 1 write I/O of 13 B to file 6. Again computation resumes for 2.24 seconds, followed by 1 write I/O of 14 B to file 6. After 597 microseconds, the entire cycle repeats. Considering all four main phase cycles at this level, we observed the following characteristics. In the period of write I/O to file 10, all 231 transfers complete within one microsecond and the time between successive transfers averages 454 microseconds. During the second major period of I/O, the write I/O transfers to file 8 also complete within one microsecond and the time between successive transfers averages 1,597 microseconds.

5.0 Conclusions

To broaden our understanding of high performance I/O at the application level, we observed a typical supercomputer workload and isolated a representative set of I/O intensive applications from the private user component of the workload. We focused exclusively on private user applications, believing that many of them require the fastest and largest resources available due to the complexity of their research investigations and the immense size of their data sets. The resulting I/O intensive set contained 80 distinctly named applications. Although this set consists of less than 4% of all user applications, it is responsible for 88% of total workload bytes transferred and only 33% of total workload CPU time.

The virtual I/O rate of each application examined and the contribution of its aggregate resource usage to workload resource usage were the fundamental criteria for including it in the I/O intensive set. However, these applications have additional attributes which make them noteworthy. They are long-running (i.e., average CPU execution times greater than 100 secs.), frequently executed applications from a variety of different scientific disciplines. The combined effects of long-running, I/O intensive applications that are frequently present in the workload place sustained demands on system resources and can exert peaks of I/O load that stress the I/O system to the point of affecting their own response time or impeding the progress of other applications. Understanding the dynamic behavior of applications such as these can provide insight into the many issues surrounding interconnection networks, operating system support, and file system structure for high performance I/O systems.

Although the I/O intensive set is small, it still would be an arduous task to analyze the dynamic behavior of all the individual applications. Therefore, we conducted a functional level characterization of the set with respect to scientific discipline to determine the relationships between the disciplines, their applications, and their I/O demands. Clearly, the numerous atmospheric sciences and industrial partner applications have a major influence on system bytes transferred, each representing 20% to 30% of all bytes transferred within the I/O intensive set and with respect to the entire workload. This type of functional level characterization provides us with important information to guide the selection of a small subset of representative applications which can then be analyzed individually.

To investigate the underlying sources and characteristic I/O activity producing such high I/O demand in the applications we have observed, we must monitor the run-time behavior of the application and collect detailed information about each I/O event. The dynamic analyses of *R* and *OGCM* demonstrate our ability to characterize the I/O behavior of scientific applications, while showing two different I/O behaviors of atmospheric sciences applications. *R* is a read I/O intensive analysis application with two similar patterns of recurring I/O activity. The only difference between the initial and main phase cycles is that in the main phase cycle there is write I/O activity after a longer, second computation

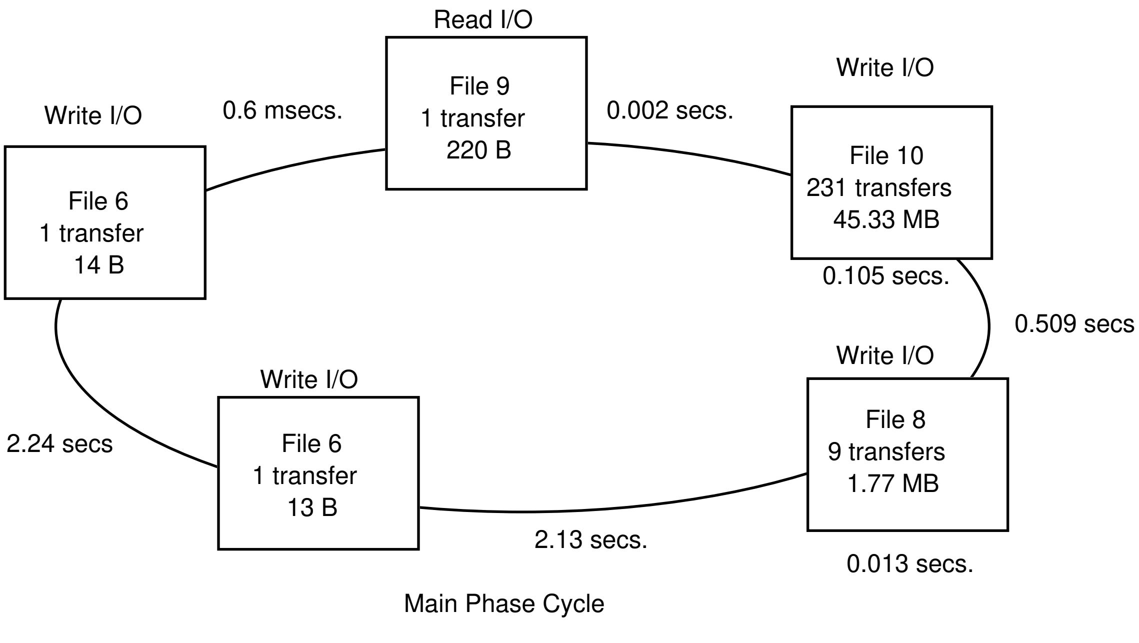

Figure 10: OGCM's Dynamic I/O Profile

period. *OGCM* is a write I/O intensive simulation application which has only one pattern of recurring I/O activity.

Although the I/O behaviors of these codes are different, we can draw the following conclusions with respect to both applications. The seemingly complex behavior of many large data files and high I/O rates can be reduced to a precise pattern of recurring I/O activity. The individual I/O events within the cycle are regular and predictable as well as the duration of computation periods. We can attribute this regularity to two different sources: the logical design of application source code and the physical configuration of the system. The source code contains a loop or repetitive sequence of data accesses and computations which controls the highest operational level of application. Every iteration causes the same files to be accessed, in the same order, transferring the same amount of data. The amount of data requested can depend on main memory size or the underlying format of the data itself. To satisfy user requests for data, the system transfers data in block-sized units. If the request size of the data is greater than the system s unit of transfer, multiple fixed-sized transfers will occur. For the Cray C90 the standard unit of data transfer is 196,608 bytes which is equivalent to 48 blocks of 512 64-bit words.

We believe that the regularity and consistency exhibited in both *R* and *OGCM* is not uncharacteristic of scientific applications in general. Scientific applications for supercomputers are highly structured, regular codes which utilize carefully formatted data sets. Based on the strict design of these applications and the physical constraints of the system, it is likely that the resource usage of scientific applications, especially I/O, follows a regular and predictable pattern. Our goal is to observe additional applications from the I/O intensive set and study the characteristic patterns of I/O behavior.

6.0 Acknowledgments

The authors are grateful to Dr. C. Roberto Mechoso of UCLA and Dr. John O. Roads of Scripps Institution of Oceanography for giving us access to the *OCGM* and the *R* applications, respectively. They would also like to thank Dr. Reagan Moore, Larry Diegel, and Michael Vildibill of the San Diego Supercomputer Center for their assistance in providing data.

7.0 References

[1] Calzarossa, M., and Serazzi, G., Workload Characterization for Supercomputers, Performance Evaluation of Supercomputers, Ed. J.. L. Martin, pp. 283-315, North-Holland, 1988.

[2] Ferrari, D., Workload Characterization and Selection in Computer Performance Measurement, Computer, pp. 18-24, July/August 1972.

[3] Ferrari, D., Serazzi, G., and Zeigner, A., Measurement and Tuning of Computer Systems, Prentice-Hall, 1983.

[4] Heidelberger, P., and Lavenberg, S. S., Computer Performance Evaluation Methodology, IEEE Transactions on Computers, Vol. 33, No. 12, pp. 1195-1220, December 1984

[5] Hennessy, J. L., and Patterson, D. A., Computer Architecture: A Quantitative Approach, Morgan Kaufmann Publishers, Inc., 1990.

[6] Jain, R., Requirements and Heuristics for Scheduling Parallel I/O Operations, Proceedings of IPPS 93, Workshop on Input/Output in Parallel Computer Systems, Newport Beach, CA, April 1993.

[7] Karin, S., The Evolving Supercomputer Environment at the San Diego Supercomputer Center, Supercomputing, NATO ASI Series, Vol. F62, pp. 97-108. Edited by J. S. Kowalik, Springer-Verlag Berlin Heidelberg, 1990.

[8] Katz, R. H., Gibson, G.A., and Patterson, D. A., Disk System Architectures for High Performance Computing, Proceedings of the IEEE, Vol. 77, No. 12, pp. 1842-1858, December 1989.

[9] Lim, S. B. and Condry, M. W., Supercomputing Application Access Characteristics , Technical Report No. UIUCDCS-R-91-1708,University of Illinois at Urbana-Champaign, October 1991.

[10] Miller, E. L. and Katz, R. H., Input/Output Behavior of Supercomputing Applications, Proceedings of Supercomputing 91, November 1991.

[11] Pasquale, B. K. and Polyzos, G. C., A Static Analysis of I/O Characteristics of Scientific Applications in a Production Workload, Proceedings of Supercomputing 93, Portland, OR, pp. 388-397, November 1993.

[12] Pasquale, B. K. and Polyzos, G. C., A Case Study of Scientific Application I/O Behavior, Proceedings of MASCOTS 94, International Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, Durham, NC, pp. 101-106, Jan. 31 - Feb 2, 1994. (Also appears as Sequoia 2000 Technical Report #93/36, University of California Berkeley, December 1993.)

[13] Pasquale, B. K. and Polyzos, G. C., Dynamic I/O Characterization of I/O Intensive Scientific Applications, Technical Report No. CS94-364, University of California, San Diego, April 1994.

[14] Patterson, D. A., Gibson, G. A., and Katz, R. H., A Case for Redundant Arrays of Inexpensive Disks, Proceedings of ACM SIGMOD Conference, Chicago, IL, June 1988.

[15] Roads, J. O., et al., A Preliminary Description of the Western U.S. Climatology , Proceedings of the Ninth Annual Pacific Climate (PAClim) Workshop, September 8, 1992.

[16] Serazzi, G., A Functional and Resource-Oriented Procedure for Workload Modeling, Performance '81, pp. 345-361, North-Holland, 1981.

[17] Stonebraker, M., and Dozier, J., Overview of the Sequoia 2000 Project, Proceedings of COMPCON 93, San Francisco, CA, February 1992.

[18] Corporation for National Research Initiatives, CASA Gigabit Testbed: 1991 Annual Report, June 1991.

[19] [UNICOS System Administration Manual, Vol. 1 (SG-2113 8.0), Cray Research Inc., 1994.