An Adaptive Parallel File System for Clusters (C-PFS)

A Parallel File System supporting Parallel I/O under Task Migration for Parallel Computing on Network of non-dedicated Workstations

The 2nd Stage Proposal for Doctoral Research

By

Rajkumar Buyya
(Student No: N02277051)
Research Scholar, Programming Languages and Systems, School of Computing Science
Email: rajkumar@fit.qut.edu.au
URL: http://www.fit.qut.edu.au/~rajkumar

Principal Supervisor

A/Prof. Clemens Szyperski
Director, Programming Languages and Systems, School of Computing Science

Associate Supervisor

Dr. Wayne Kelly
Lecturer, School of Computing Science

School of Computing Science
Queensland University of Technology
Brisbane, Australia

Table of Contents

1. Research Centre and the Proposed Research Alignment

1.1 Research Centre

The proposed research will be conducted at the following research centre:

Programming Languages and Systems (PLAS) Laboratory
School of Computing Science
Faculty of Information Technology
Queensland University of Technology
Brisbane, Australia

1.2 Centre Background

The PLAS, Programming Languages and Systems research concentration of QUT, research scope spans programming language design and implementation, and programming environments. PLAS is actively involved or involving in several research projects and one of them is the ***Gardens Project***. The Gardens Project is aims to create a virtual parallel machine out of a network of computers (workstations/PCs). These computers are interconnected through low latency and high bandwidth communication links such as ***Myrinet***. The focus of Gardens is on tightly integrating the programming and system facilities by the use of a single sufficiently expressive language. The project is composed of a family of subprojects spanning most areas of interest covered by PLAS.

1.3 Proposed Research Alignment with Gardens Project

In order to create virtual parallel machine, the Gardens Project (Gardens, for short), currently supports the following:

- Abstraction for expressing logical parallelism through the Mianjin language
- Gardens Tasking Systems for Process Migration and Control

Gardens support migration of tasks from one workstation to another workstation for adaptive parallel computing on networks of non-dedicated workstations. Migrated tasks can continue to perform valid I/O operations on files, which are associated with the network file system (NFS), where *the underlying NFS is slow*. However, if a Gardens task opens a file on a disk storage space physically associated with a particular workstation, it cannot access such files after migration to another workstation. This is the first issue the proposed research addresses by providing ***transparent access mechanism for accessing files under task migration***. The second issue addressed by the proposed work is to develop ***adaptive parallel file system*** supporting parallel I/O under migration. The third issue addressed by the proposed research is a set of ***parallel I/O libraries*** and building mechanisms for efficient I/O in Gardens system.

The above discussion shows that the proposed research is well aligned with the Gardens Project and contributes towards achieving its goals.

2. Coursework to be undertaken

Literature search and review is a continuous part of the doctoral (Ph.D.) research program. To aid in these activities, the interfaculty unit IFN001, Advanced Information Retrieval Skills, will be studied in the second semester of 1998.

3. Proposed Title

The proposed title for the Ph.D. thesis is,

C-PFS: An Adaptive Parallel File System for Clusters

4. Objectives of the Program of Research, Motivations, and Investigation

4.1 Objectives of the Research

The major objectives of the proposed research are the following:

- to build and contribute to the knowledge of adaptive parallel computing across network of workstations and in particular to the field of high performance parallel file system and parallel I/O.
- to provide important additions to the knowledge of operating systems for distributed/cluster computing exhibiting a single system image.
- to build a generic framework for a parallel file system interface which can be easily exported to programming languages such as Mianjin, C/C++, Java, etc.
- to investigate and develop various techniques for improving I/O performance
- to investigate heterogeneous parallel file system that can co-exist with planned heterogeneous task migration support in the Gardens system.

4.2 Motivations for Cluster Computing/Gardens Project

We have seen a dramatic increase in workstation performance in the last few years, with workstations doubling in performance every 18-24 months. This is likely to continue for several years, with faster processors and multiprocessor machines. *Studies have shown that over half of the workstations CPU cycles are unused even during peak hours. One possible use of these idle cycles is to run parallel applications on a network of computers.*

In recent years, high-speed networking and improved microprocessor performance are making the networks of workstations an appealing vehicle for parallel computing. Clusters or networks of computers (workstations/PCs) built using commodity hardware or software are playing a major role in redefining the concept of supercomputing. As a whole, clusters are becoming an alternative to MPPs and supercomputers in many areas of application.

The Gardens Project started with the motivation of taking advantage of the changes in the technological trends as discussed above and creating a virtual parallel machine out of a network of workstations with low latency inter-machine communication links such as Myrinet.

4.3 Motivation for Parallel File System

Improvements in disk access time have not kept pace with microprocessor performance, which has been improving by 50% or more per year. Although magnetic-media densities have increased, reducing disk transfer times by approximately 60-80% per year [11], overall improvement in disk access times, which rely upon advances in mechanical systems, has been less than 10% per year.

Parallel/Grand challenging applications need to process large amounts of data and data sets. *Amdahl's law implies that the speedup obtained from faster processors is limited by the slower system components*; therefore, it is necessary to improve I/O performance such that it balances with CPU performance. One way of improving I/O performance is to carry out I/O operations in parallel, which can be supported by building a high performance—parallel—file system.

4.4 Investigation

The proposed work investigates and provides solution to the growing disparity between CPU and I/O performance. Traditional parallel supercomputers have overcome this by offering a mechanism to access the data in parallel through parallel file system. The proposed work investigates on the use of similar concept on a network of workstations that coexists with task migration for *adaptive* parallel computing. In addition, it also investigates on supporting parallel I/O libraries and language mechanisms required for supporting collective I/O, out-of-core data structures, etc to overcome I/O bottleneck.

5. Relation to Previous Work in the Same Field

5.1 By the Candidate

After completion of the Master of Engineering degree in Computer Science in 1995, I joined the Centre for Development of Advanced Computing, India and worked on building operating environment for their PARAM family of supercomputers. As a member of the Operating System group, I have designed and developed the POSIX Threads Interface as a subsystem on top of the PARAS Microkernel operating system [1]. I have led a team working on developing software systems exhibiting a single system image on Cluster of Workstations. The team has developed a comprehensive cluster monitoring system, PARMON, for monitoring system resource utilization. PARMON was developed by using the state-of-the-art Java and Client-Server Computing technologies. It allows to monitor the usage of system resources at component level, node level, and entire system level [6]. It is being successfully used in monitoring PARAM OpenFrame Supercomputer, which is a cluster of 48 Ultra-4 workstations. Working on PARAM supercomputer project has resulted in thorough understanding of concepts of client-server computing, implementation of multithreaded servers, design and development of software that needs to provide multiple services simultaneously, and kernel processing.

In addition to the above, I have also worked and supervised projects in the following areas:

- Microkernel based operating systems [3].
- Parallel and Distributed/Cluster Computing: Opportunities and Challenges [2].
- Suitability of Java for Parallel/Distributed Computing [5].
- Investigated various approaches for building single system image operating environment for high performance computing on network of workstations, cluster of workstations, vector and parallel supercomputers [4].

5.2 By the Supervisor

The principal supervisor is an active researcher in the areas of Component Software, Component-Oriented Programming, and Extensible Systems, Distributed Computing and High Performance Computing across Distributed Systems, Programming Languages and Systems. He has designed and developed an operating system called Ethos [12] using object-oriented techniques. The supervisor, as the Director of the PLAS centre activities, has involved extending the Oberon language [13] to support parallel programming and an extended new language is called Mianjin[14].

The associated supervisor is also an active researcher in the areas of High Performance Computing, Optimizing and Parallelizing Compilers, Application of Mathematical Frameworks, Performance estimation, and Code Generation. He has devised well known mechanism for Minimizing Communication while Preserving Parallelism [15] and Code Generation for Multiple Mappings [16].

5.3 By PLAS Group

The Programming Languages and Systems (PLAS) Group has initiated the Gardens Project in 1995 at the Queensland University of Technology. The project is led by Associate Professor Clemens Szyperski and Dr. Paul Roe, with the participation of fellow researchers, Ph.D. students, honours, and final year undergraduate students.

Gardens creates a virtual parallel machine out of a network of workstations with low latency inter-machine communication links. The focus is on tightly integrating the programming and system facilities by the use of a single sufficiently expressive language. The project is composed of a family of subprojects spanning most areas of interest covered by PLAS. The first prototype system, GATE, was completed in late 1996[9] and it successfully exhibited the characteristics required to support the initial project outline [8].

The Gardens Project currently support, abstractions for expressing parallelism through Mianjin language and Tasking Systems for Process Migration and Control. It does not support I/O and Parallel I/O under migration [10].

The aim of the proposed research, Adaptive Parallel File System, is to build a parallel file system for networks of workstations by using secondary storage unit, hard-disk, which exists with each workstation/node in a network/Gardens system. Unlike the existing Gardens system, the new Gardens system with proposed parallel file system will support parallel I/O under task migration and high performance I/O libraries for parallel processing.

5.4 By Others

Most commercial multiprocessor file systems are based on the Unix linear file model. Some simply provide a Unix-like interface, and some provide the full semantics required by Unix standards. Intel Concurrent File System [17] is frequently cited as the canonical first-generation parallel file system. It was written for the IPSC family of parallel machines. Its successor, PFS, is similar and was written for the Paragon [18]. Thinking Machines developed a file system called *sfs*, which supports data-parallel I/O on CM-5 through data parallel languages such as CMF, C*, or Lisp [19]. PIOFS, a parallel file system for IBM's SP2, allows users and applications to interact with it exactly as they would interact with AIX file system [20]. Indeed, the PIOFS is mounted on each of the nodes of the SP-2 using AIX's standard Virtual File System interface. Bridge was one of the earliest file systems [21], and is unusual in not separating I/O nodes from compute nodes. A file-system interface proposed for the nCUBE is based on a two-step mapping of a file into the compute-node memory [22]. Galley is another parallel file system for IBM-SP2 and it is based on a new three-dimensional structuring of files [23].

Parallel file systems discussed above do not support parallel I/O under task migration and they have been developed for proprietary parallel machines. Unlike these existing file systems, the proposed Adaptive Parallel File System for Clusters (C-PFS) supports *Parallel I/O under migration* for adaptive parallel computing on a network of non-dedicated workstations. Initially, C-PFS will use disks associated with workstations in a network. It is expected that later it can be easily extended to also support dedicated I/O servers.

6. Research Methods and Research Plan

The proposed work will address the following issues:

- ***I/O under Process Migration***
- ***Adaptive Parallel File System supporting Parallel I/O under Process Migrations***
- ***Improving Performance of Adaptive Parallel File System***

The following topics will be investigated as a part of doctoral research:

1. Study of parallel programming with Mianjin
2. I/O under Process Migration
3. Investigation of I/O Requirements
4. Parallel I/O: Solutions
5. Investigation of spaces of parallelism in parallel file system
6. Parallel File System supporting Parallel I/O under Process Migrations
7. Investigation techniques for improving parallel file system
8. Design of a prototypical Parallel File System
9. Implementation of the designed Parallel File System
10. Exporting C-PFS to various programming languages

6.1 Research Plan for the Next Six Month (July-Dec,1998)

Work in this period will concentrate on building I/O under migration support for Gardens as discussed in the following sections. It is planned to investigate the I/O requirements through literature review and survey and by devising I/O intensive parallel applications.

I have also planned to build a Gardens Monitoring System (GMS) along with a student of Master of Information Technology of QUT. GMS interacts with GATE (Gardens Application Tasking Environment) and presents Gardens system activities using Java GUI. Monitoring of Gardens using GMS helps in understanding effectiveness of load balancing and process migration policies used in Gardens and their

effects on parallel I/O under migration. It also helps in administrating of Gardens, which is difficult, otherwise.

6.1.1 I/O under Process Migration

A parallel application developed using Mianjin is made of up of multiple Gardens Application Processes (GAPs) (and each GAP manages a set of tasks) running on a set of idle workstations. When the owner of a workstation starts working on it, tasks associated with the local GAP need to be migrated to other idle workstation(s) or the home workstation—the machine on which execution of the Gardens Parallel Applications is initiated. Currently Gardens allow its tasks to perform file operations on those files stored in networked file system (NFS), where the underlying NFS is slow. That is, if task opens a file on a disk storage space physically associated with a node (home workstation), it cannot access such files after process migration. This is the first issue the proposed research addresses by providing ***transparent access mechanism for accessing files under process migration***.

Proposed Architecture

When a file is opened in a task, it will be associated with a pointer to the file data-structure which stores information necessary to support file operations. In order to support file I/O under process migration, the file date structure need to be extended so as to make provision for storing additional information related to the owner of a file and its physical location—home workstation. The components involved in supporting I/O under migration are shown in Figure 1.

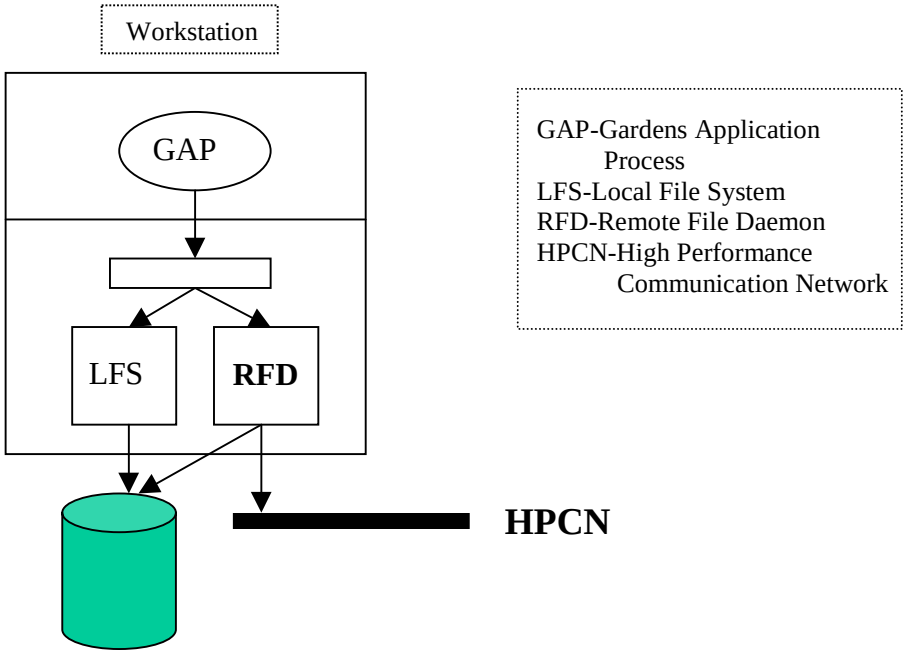

Figure 1: Architecture of I/O under Migration in Gardens

When a file is opened by a task, it will be associated with a home workstation from which execution of the Gardens parallel program is initiated. If a task tries to access a file, the access can be local or remote based on which workstation the task is running on. If the task is running on the home workstation, then the file access type is local, otherwise file access type is remote. Remote File Daemon (RFD) is responsible for resolving whether file access is local or remote. When file access is remote, the request will be passed to RFD running on the home workstation, which does the actual file operation and responds to the requesting RFD with necessary data. The communication between RFDs will takes place through High Performance Communication Networks such as ***Myrinet*** using fast communication interfaces like ***Active Messages***.

The Same API for Accessing File I/O under Migration

In order to support access to files even after process migration, APIs (Application Programming Interfaces) associated with file processing in the Mianjin/Oberon language, have to be modified appropriately. That is, the syntax of the file management API remains the same, but mechanism for supporting file access has to be modified. The file access mechanisms incorporating the necessary changes will be devised.

6.1.2 Reporting of Work

A seminar will be presented on the above work, and a report will be written.

The report will be developed into a paper for publication in a related journal or presentation and publication in a relevant conference.

6.2 Research Plan for the Following Three Months (Jan-March, 1999)

During this period a major focus will be on investigating I/O characteristics and various dimensions of potential parallelism in parallel file system discussed in the following section.

6.2.1 Investigation of I/O Characteristics and Parallelism in Data Access

Applications in the area of scientific computing, databases, multimedia will be investigated to study their I/O behavior. Detailed investigation will be carried out on the following three spaces of parallelism in parallel file system:

1. Program
2. File
3. Disk

The investigations will help selecting good or optimal data distribution protocols and file access policies when designing the proposed parallel file system supporting Parallel I/O under migration.

6.2.2 Reporting of Work

A seminar will be presented on the above work, and a Ph.D. confirmation report will be written.

A seminar on Gardens Monitoring System (GMS) project developed in Java, which I initiated along with a student of Master of Information Technology of QUT will be presented. It is planned to write a report/user manual and a paper for publication either in a relevant journal or conference.

6.3 Research Plan after Ph.D. Candidature Confirmation (April '99-Dec '2000)

6.3.1 Adaptive Parallel File System

The second issue addressed by the proposed work is to develop ***parallel file system coexisting with process migration*** providing abstractions through the Gardens language for accessing files efficiently.

6.3.2 Issues Addressed by Adaptive Parallel File System

The C-PFS will address the following essential issues:

- Uniform directory access: C-PFS needs to support a unified file system for both sequential and parallel file accesses.
- Ability to execute multiple file accesses: C-PFS needs to support multiple file pointers to a file (one per process and shared pointers--all processes can share a common file pointer).
- Availability of efficient Read/Write (R/W) procedures.
- Provision of transparent file/data management.
- Coexist with standard file system: C-PFS must coexist and work in cooperation with a workstation's own file system.

6.3.3 Proposed Architecture

Unlike traditional parallel processing systems, one of the characteristics of a cluster of workstations is that each node will have one or more disks attached to them. This property can be used to increase the bandwidth of the I/O system. The idea is to de-cluster data and store each slice of data on different disks so that requested information can be fetched from disks in parallel. It increases the data transfer bandwidth and is proportional to the number of disks used in parallel.

A simplified architecture of proposed file system is shown in Figure 2. It consists of two threads:

1. Building Software RAID (Redundant Arrays of Inexpensive Disks) using disks associated with each node in a cluster of workstations.
2. Providing Parallel File I/O facility under Task Migration.

As planned Adaptive Parallel File System has to address the above two threads. Every workstation participating in Gardens will have a disk partition called Gardens Partition. The proposed parallel file system builds software RAID using Gardens Partitions. Whenever a task performs write operation, the data will be distributed across all disks so that when read operation is requested, data stored on multiple disks will be read parallel and made available to the task transparently.

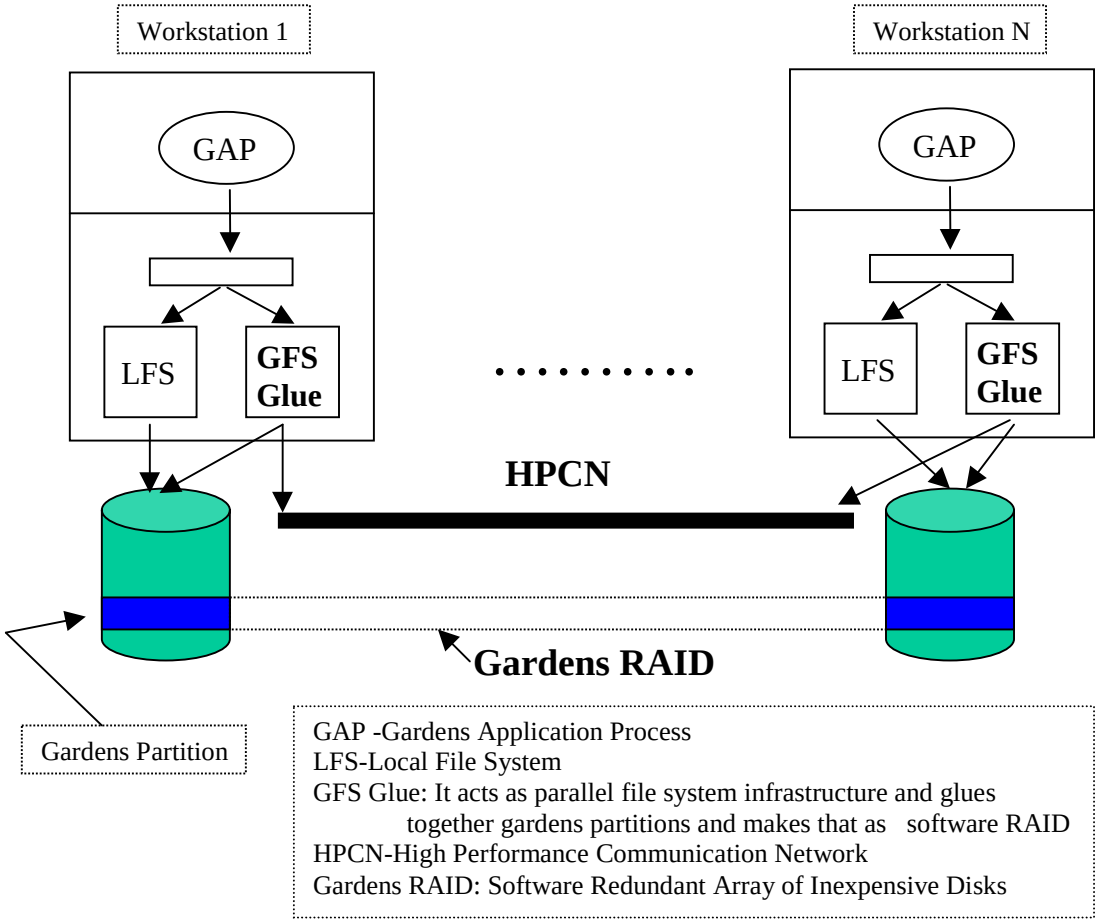

Figure 2: Architecture of Adaptive Parallel File System for Clusters

The C-PFS will also support Parallel I/O under migration. The techniques for implementing this is similar to the one discussed in the previous section, I/O under Migration. Some of the components of C-PFS will be implemented on top of local file system so the C-PFS appears like a layer which glues together file

system available on all workstations and at the same time supports high performance I/O under task migration.

6.3.4 API for Accessing File I/O under Migration

In additions to existing interfaces for file manipulation, C-PFS system will support additional API's required for fast parallel access to data, collective I/O, out-of-core data structures. Parallel scientific applications frequently make small requests to a file, with strided access patterns. C-PFS offers the following higher-level interfaces:

- Simple–strided Requests
- Nested-strided Requests
- Nested-batched Requests

6.3.5 Exporting C-PFS interface to Programming Languages

Exporting C-PFS interface to some of the popular programming languages/models enhances the usability of C-PFS. Hence, it is planned to export C-PFS facilities to the following:

- Mianjin/Oberon
- Java: C-PFS interface to Java will improve performance of Java programs and this also provides an opportunity to study behavior of various kinds of applications in the areas of databases, multimedia, WWW applications. This contributes to effort of Gardens Project future plan to build Java Interface to high performance communication interface, Active Messages.
- MPI: Message Passing Interface, MPI, has been accepted world-wide as a standard message passing interface for parallel programming and many parallel scientific-applications have be built. This infrastructure can be utilized for further investigations on the use of C-PFS for scientific computing.

6.3.6 Parallel Data Access Libraries and C-PFS Performance

The third issue addressed by the proposed research is providing efficient access to files under migration by implementing ***efficient Parallel I/O*** libraries functions in Gardens system.

The following techniques will be investigated and incorporated to improve performance of proposed Adaptive Parallel File System:

- Caching and Pre-fetching
- Two Phase I/O
- Disk Directed I/O
- Computer Node Caching
- Striping/de-clustering/slicing
- Compression
- Filtering

6.3.7 Reporting of Work

The following are a part of reporting activities after Ph.D. confirmation research:

- Seminars will be presented and progress report will be submitted every six month until submission of the final thesis.
- Planning to publish papers in journals or conferences after making substantial progress on each of the following:
  - Analysis and Design of Parallel I/O and File System Requirements
  - Implementation of Adaptive Parallel File System
  - Techniques of Improving Adaptive Parallel File System
  - Exporting C-PFS to various programming languages (such as Mianjin, Java, MPI, etc.) for the benefit of both sequential and parallel computing.
  - Experience in using C-PFS and recommendations of next generation file system

7. Schedule for Completion

The schedule of various activities carried out or to be carried out are outlined in Figure 3. The X-axis indicates time duration in months and Y-axis indicates activities to be performed in order to complete the proposed work successfully.

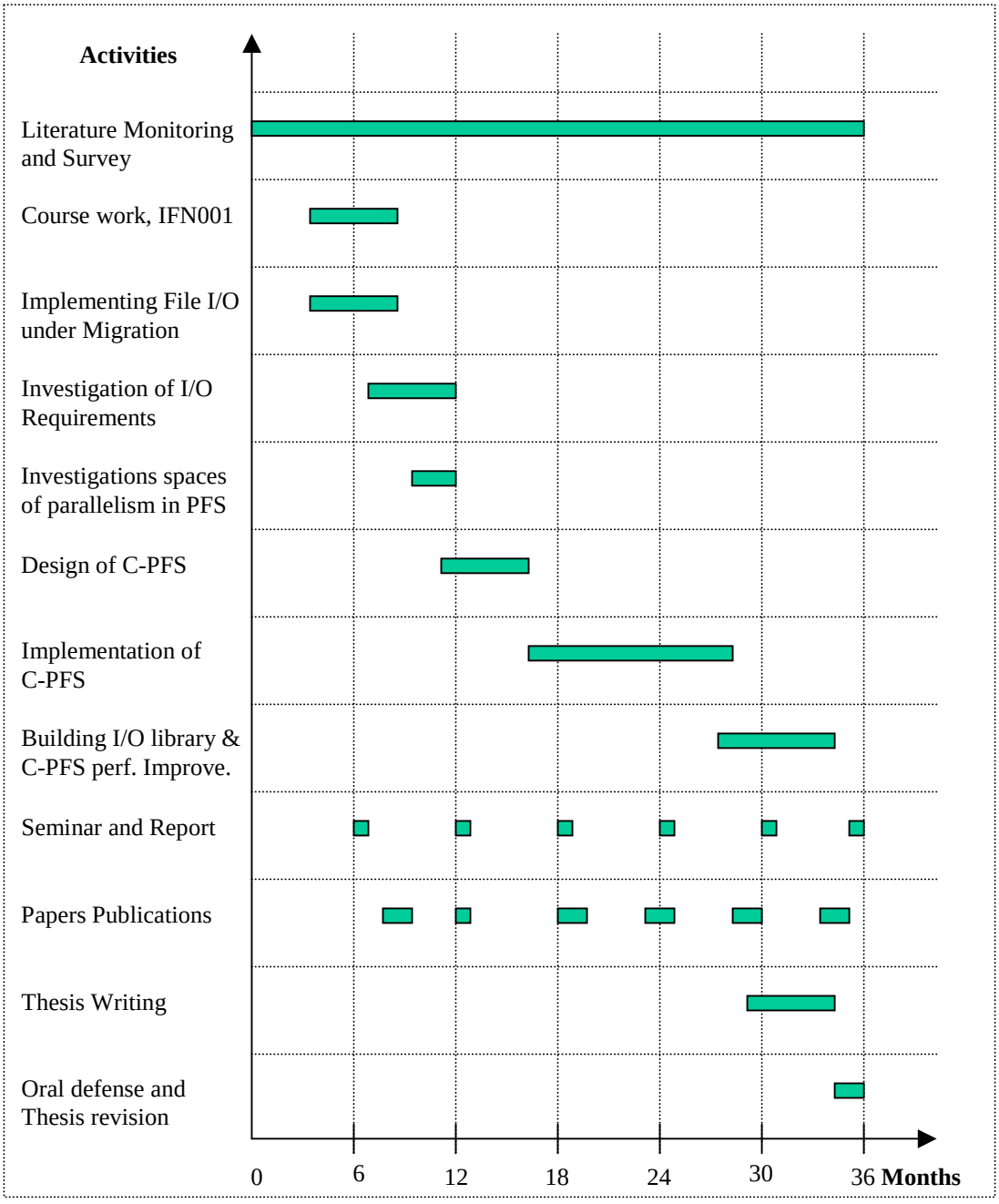

Figure 3: Schedule for Completion of Ph.D. Course Activities

8. Contribution to a Group Effort

As stated earlier, the Gardens Project aims at creating a virtual parallel machine out of a network of workstations with low latency inter-machine communication links. It focuses on tightly integrating the programming and system facilities by the use of a single sufficiently expressive language. Essentially Gardens research consists of two threads: programming languages and systems. Overall emphasis is on performance (of course), adaptation to dynamic processor sets, and safe and simple programming model with explicit cost model.

The proposed work will contribute towards the efforts of PLAS Group by supporting facility of Parallel I/O under Task Migration by building following components:

- Adaptive Parallel File System
- Parallel I/O Library
- Policies developed as a part of this project can be integrated with process migration policies
- Abstractions of Parallel I/O through Mianjin
- Mechanisms for exporting Parallel I/O to languages/models such as Java, MPI.

The above-proposed contribution shows that the proposed research is well aligned with Gardens Project and contributes towards achieving its goals.

9. Ethical Clearance

The proposed work does not require ethical clearance.

10. References

1. Rajkumar et al. ***POSIX Threads Interface on Microkernel***. Proceedings of the Fourth International Conference on Advanced Computing (ADCOMP'95), Bangalore, India, Tata McGraw Hill Publications, 1995.
2. Achu, Rajkumar, Bala, and Hari. ***PARDISC: A Cost Effective Model for Parallel and Distributed Computing***. Proceedings of the Third International Conference on High Performance Computing (HiPC'96), Trivendrum, India, IEEE Computer Society Press, 1996. Also appeared as PARDISC: making the 'most of both worlds
3. Mohan Ram, Rajkumar et al. ***A Microkernel Based Operating System for PARAM 9000***. Proceedings of the Fourth International Conference on Advanced Computing (ADCOMP'96), Bangalore, India, Tata McGraw Hill Publications, 1996.
4. Rajkumar. ***Single System Image: Need, Approaches, and Supporting HPC Systems***. Proceedings of the Fourth International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'97), Las Vegas, Nevada, USA, CSREA Press, 19
5. Rajkumar and Vijaya. ***JMPF: A Message Passing Framework for Cluster Computing in Java***. Proceedings of the Fifth International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'98), Las Vegas, Nevada, USA, CSREA Press, 1998.
6. Rajkumar et al. ***PARMON: A Comprehensive Cluster Monitoring System***. Proceedings of the Fifth International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'98), Las Vegas, Nevada, USA, CSREA Press, 1998.
7. G A Geist. ***Cluster Computing: The Wave of the Future?.*** Oak Ridge National Laboratory, Oak Ridge, Technical Report, TN 37831-6367, 1994.
8. Jaochim Diederich, John Gough, George Mohay, and Clemens Szyperski. ***The Gardens Project - An Introduction***. Presented at the Second Australian Computer Architecture Workshop, Adelaide, Southern Australia, January 1995.
9. Ashley Beitz, Siu Yuen Chan, and Nik Kwiatkowski. ***A Migration-friendly Tasking Environment for Gardens***. Fourth Australasian Conference on Parallel and Real-Time Systems (PART'97), Newcastle, September 1997,
10. Paul Roe and Clemens Szyperski. ***Gardens: An integrated programming language and system for parallel programming across networks of workstations.*** 21st Australian Computer Science Conference (ACSC'98), Perth, Western Australia, Australia, 4-6 February 1998, Springer Verlag, 981-3083-90-5.

11. Ruemmler, C., and Wilkes, ***J. Modelling Disks***. Tech. Rep. HPL-93-68, Hewlett Packard Laboratories, July 1993.
12. Clemens Szyperski. ***Insight Ethos: On Object Orientation in Operating Systems***. (PhD thesis; Swiss Federal Institute of Technology (ETH Zurich), Diss. No. 9884). Verlag der Fachvereine, Zurich, Switzerland, 1992, ISBN 3-7281-1948-2.
13. Cuno Pfister and Clemens Szyperski. ***Oberon/F Framework - Tutorial and Reference.*** Oberon microsystems, Inc., Zurich, Switzerland, 1994.
14. Paul Roe and Clemens Szyperski. ***Mianjin is Gardens Point: A Parallel Language Taming Asynchronous Communication***. Fourth Australasian Conference on Parallel and Real-Time Systems (PART'97), Newcastle, September 1997, Springer Verlag, ISBN 981-3083-62-X.
15. Wayne Kelly and William Pugh, ***Minimizing Communication while Preserving Parallelism***, Technical Report CS-TR-3571, Dept. of Computer Science, University of Maryland, College Park, Dec 1995 and Proceedings of International Conference on Supercomputing.
16. Wayne Kelly, William Pugh and Evan Rosser, ***Code Generation for Multiple Mappings***, The Fifth Symposium on the Frontiers of Massively Parallel Computation, pages 332-341, McLean, Virginia, February 1995.
17. Bill Nitzberg. ***Performance of the iPSC/860 Concurrent File System***. Technical Report RND-92-020, NAS Systems Division, NASA Ames, December 1992.
18. Brad Rullman and David Payne. ***An efficient file I/O interface for parallel applications***. Workshop on Scalable I/O, Frontiers'95, February 1995.
19. Susan J et al. ***sfs: A Parallel File System for the CM-5***. In Proceedings of the 1993 Summer USENIX Conference, 1993.
20. Peter F Corbett and Dror G. Feitelson. ***The Vesta Parallel File System***. ACM Transactions on Computer Systems, August 1996.
21. Peter C Dibble. ***A Parallel Interleaved File System***. Ph.D. Thesis, University of Rochester, March 1990.
22. Erik P DeDenedictis and Stephen C. Johnson. ***Extending Unix of Scalable Computing***. IEEE Computer, November 1993.
23. Nils Nieuwejaar. ***Galley: A New Parallel File System for Scientific Workloads***. Ph.D. Thesis, Dartmouth College, Hanover, 1996.