Available online at www.sciencedirect.com
SCIENCE DIRECT®

Future Generation Computer Systems 22 (2006) 620–632

FGCS
FUTURE GENERATION COMPUTER SYSTEMS

www.elsevier.com/locate/fgcs

MAPFS: A flexible multiagent parallel file system for clusters

María S. Pérez[a,*], Jesús Carretero[b], Félix García[b], José M. Peña[a], Víctor Robles[a]

[a] *Department of Computer Architecture and Technology, Technical University of Madrid, Madrid, Spain*
[b] *Department of Computer Science, University Carlos III of Madrid, Spain*

Available online 2 November 2005

Abstract

The emergence of applications with greater processing and speedup requirements, such as *Grand Challenge Applications* (GCA), involves new computing and I/O needs. Many of these applications require access to huge data repositories and other I/O sources, making the I/O phase a bottleneck in the computing systems, due to its poor performance. In this sense, parallel I/O is becoming one of the major topics in the area of high-performance systems. Existing data-intensive GCA have been used in several domains, such as high energy physics, climate modeling, biology or visualization. Since the I/O problem has not been solved in this kind of applications, new approaches are required in this case. This paper presents MAPFS, a multiagent architecture, whose goal is to allow applications to access data in a cluster of workstations in an efficient and flexible fashion, providing formalisms for modifying the topology of the storage system, specifying different data access patterns and selecting additional functionalities.

Keywords: Parallel I/O; Cluster computing; Multiagent architectures; Access patterns

1. Introduction

Nowadays, there is a growing interest in the development of high-performance I/O systems, because the I/O phase has become a bottleneck in the computing systems due to its poor performance. In fact, one of the major goals of high-performance computing is to provide an efficient access to data, making parallel I/O one of the most relevant issues in this field.

Currently, there are different parallel file systems, such as Galley [25], Parallel File System (PFS) [9] and Portable Parallel File System (PPFS) [16], which offer high-performance services to access resources. Many of these systems have been widely developed for parallel machines and are not suitable for clusters of workstations. Nevertheless, there is an increasing trend in parallel computing towards the usage of clusters, mainly because of their prices and their ease of integration. In this sense, the Parallel Virtual File System (PVFS) [3] is used in dedicated clusters of

* Corresponding author.
E-mail addresses: mperez@fi.upm.es (M.S. Pérez), jmpena@fi.upm.es (J.M. Peña), vrobles@fi.upm.es (V. Robles), jcarrete@arcos.inf.uc3m.es (J. Carretero), fgarcia@arcos.inf.uc3m.es (F. García).

0167-739X/$ – see front matter © 2005 Elsevier B.V. All rights reserved.
doi:10.1016/j.future.2005.09.006

workstations. On the other hand, parallel file system optimizations provide improved I/O operations. The usage of hints related to different aspects of data distribution and access patterns allows parallel file systems to increase the performance of these operations. Processes such as caching or prefetching are useful approaches used in addition with these last ones [28,2]. The *agent technology* [10,15] is a suitable framework for integrating these functions in the storage system, because of its adaptability to different domains and its capability to achieve process autonomy.

This paper presents MAPFS, a multiagent architecture, whose goal is to allow applications to access data in a cluster of workstations in an efficient and flexible fashion, providing formalisms for modifying the topology of the storage system, specifying different data access patterns and selecting additional functionalities. The outline of this paper is as follows. Section 2 describes the problems of data-intensive applications, which we need to address by means of a flexible I/O architecture. Section 3 presents MAPFS as a suitable infrastructure for this kind of applications in a cluster environment. This section describes MAPFS architecture, showing the different modules in which the system is divided into. Additionally, it shows the formalism of storage groups, which is used for the dynamic management of servers. Finally, in this section the way in which MAPFS allows applications to define different access patterns is also described. Section 4 shows the results obtained for the evaluation of applications using MAPFS. With this aim, we have evaluated the different features of MAPFS. Furthermore, a comparison between MAPFS and PVFS, another parallel file system for clusters, is analyzed. Finally, Section 5 summarizes our conclusions and suggests further future work.

2. Problem statement and related work

2.1. Data-intensive applications and their I/O needs

The emergence of applications with greater I/O access requirements, also known as data-intensive applications or I/O-intensive applications, demands new I/O solutions. Examples of data-intensive and Grand Challenge applications [12] include data mining systems, data warehousing [17], high energy physics applications [34], and satellite data processing [1]. These applications may require access to data sources distributed among different nodes. Moreover, typical data-intensive applications require access to terabyte size datasets, which must be processed in an efficient way, in order to increase the performance of the applications executed on the cluster. Furthermore, data-intensive applications are very different depending on the kind of functional requirements and access patterns. It is critical for I/O system to be flexible enough to match these demands. The usage of hints, caching and prefetching policies or different data distribution configurations are optional features, which can reduce latency and increase I/O operations performance.

Respect to the underlying architecture, clusters are characterized to be modified dynamically. Operations such as addition of new nodes or elimination of existing nodes are typical in a cluster environment or in distributed systems, in general. In fact, it is desirable that the services in a cluster allow complex software systems to be built in a “plug-and-play” fashion. Therefore, we need a tool or formalism for the dynamic reconfiguration of the storage nodes. This paper describes a parallel I/O architecture for increasing the performance of data-intensive applications in a flexible way, providing formalisms for modifying the topology of the storage system and selecting additional functionalities.

2.2. Related work

Although I/O systems have traditionally tackled the I/O phase in the same way, independent of the applications domain and their access patterns, some studies [22,26,7] have demonstrated that a higher control of the user applications over the I/O system can increase their performance. Some of the features for increasing the control of the applications are the usage of access patterns, I/O caching and prefetching or the usage of hints.

The performance of I/O accesses depends on two related aspects: data layout in the files, which is named *storage pattern*, and the distribution of data through different nodes, that is, *access pattern*. In those cases in which these two different patterns are not equal, the efficiency of I/O accesses can be decreased, since every node must access in an independent way to data, resulting in a great number of small I/O accesses. Therefore,