Parallel Computing 41 (2015) 1–13

Contents lists available at ScienceDirect

Parallel Computing

journal homepage: www.elsevier.com/locate/parco

# A prediction-based dynamic file assignment strategy for parallel file systems

Saiqin Long [a], Yuelong Zhao [a,*], Wei Chen [a], Yuanbin Tang [b]

[a] School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China
[b] Modern Educational Technology Center, Sichuan University of Arts and Science, Dazhou 635000, China

ARTICLE INFO

*Article history:*
Received 26 July 2013
Received in revised form 27 February 2014
Accepted 17 October 2014
Available online 25 October 2014

*Keywords:*
File assignment
Parallel file systems
Load prediction
Genetic algorithm
Mean response time

ABSTRACT

Nowadays, the rapid development of the internet calls for a high performance file system, and a lot of efforts have already been devoted to the issue of assigning nonpartitioned files in a parallel file system with the aim of pursuing a prompt response to requests. Yet most of the existing strategies still fail to bring about an optimal performance on system mean response time metrics, and new strategies which can achieve better performance in terms of mean response time become indispensable for parallel file systems. This paper, while addressing the issue of assigning nonpartitioned files in parallel file systems where the file accesses exhibit Poisson arrival rates and fixed service times, presents an on-line file assignment strategy, named prediction-based dynamic file assignment (*PDFA*), to minimize the mean response time among disks under different workload conditions, and a comparison of the *PDFA* with the well-known file assignment algorithms, such as *HP* and *SOR*. Comprehensive experimental results show that *PDFA* is able to improve the performance consistently in terms of mean response time among all algorithms for comparison.

© 2015 Published by Elsevier B.V.

## 1. Introduction

In recent years, parallel file systems, such as *Lustre* [19], *GPFS* [23], *HDFS* [2] and *PVFS2* [22] have received considerable attention for their ability to provide fast and reliable access. Data nodes are widely deployed to store data in numerous scenes for efficient data reading or writing. In today's web service scenario with performance requirements, various applications not only require throughput for applications, but also a speedy response time of their requests so that data can be read quickly from the data nodes. In order to meet the user's requirements, files should be assigned onto data nodes appropriately and efficiently before being accessed. Such a placement problem is referred to as file assignment problem (*FAP*) [34] which can be summarized as follows: Given a set of *N* files and *M* data nodes, we try to find the optimal solution of assigning the *N* files onto *M* data nodes so that the requests for these files can be responded rapidly. Since *FAP* formulation is known to be an NP-complete problem, we need to develop heuristic algorithms to solve this problem.

Generally, there are two kinds of file assignment algorithms: static ones and dynamic ones. While static algorithms require prior knowledge about the workload statistics such as service times and access rates of all the files, dynamic file assignment does not have such requirement and files are allocated when they arrive at the system to adapt to varying workload patterns. In this paper, we address the problem of dynamically assigning non-partitioned files in parallel file systems where file accesses exhibit Poisson arrival rates and fixed service times.

* Corresponding author. Tel.: +86 18670218554.
*E-mail addresses:* longsaiqin@163.com (S. Long), zhaolab@126.com (Y. Zhao), chw521@163.com (W. Chen), inr.ltn9100@163.com (Y. Tang).

http://dx.doi.org/10.1016/j.parco.2014.10.004
0167-8191/© 2015 Published by Elsevier B.V.

It is now well known that the processing requirements of modern computer workloads are highly variable. These workloads generally follow a *Zipfian* distribution [13]. The probability of a request for the $i^{'}th$ most popular file is proportional to 1/i. Moreover, the request frequency is believed to inversely correlate with the file size, which means that the most popular files are usually small in size, while the large files are typically unpopular. Based on these presumptions of the characteristic of workloads, a lot of file assignment algorithms including static file assignment algorithms such as *Greedy* [6], *SP* [13], and *SOR* [34], and dynamic algorithms such as *HP* [13] were developed to improve the performance in terms of mean response time of the system.

To better optimize the data layout design of parallel file systems, we propose a dynamic file assignment strategy, called prediction-based dynamic file assignment (*PDFA*), which aims at minimizing the mean response time with the consideration of load balancing between data nodes under different workload conditions. To sum up, we make the following contributions in the paper. (1) An analysis model of file assignment and file access to parallel file systems is proposed to estimate the performance metric in terms of mean response time of data accesses for different file assignment policies. (2) A load prediction model for parallel file systems is constructed to estimate the future loads of data nodes. (3) A dynamic file assignment strategy based on the overall load estimation is proposed for the purpose of choosing an appropriate file layout scheme under different workload conditions. (4) The performance of the newly proposed dynamic file assignment strategy together with the famous *HP* and *SOR* is evaluated in our execution-driven simulator. Our analytical and experimental results show that the proposed *PDFA* performs best in terms of mean response time, while *SOR* takes the second place.

The remainder of this paper is organized as follows: In the following section, background and related work are discussed. Section 3 presents the system architecture and analysis model. Section 4 describes the load prediction model. Section 5 presents the prediction-based dynamic file assignment algorithm and Section 6 shows the simulation results to validate our strategies. Finally, Section 7 concludes the paper with a summary and future work.

## 2. Background and related work

As data access performance is widely recognized as one of the major bottlenecks in data-intensive parallel applications, a lot of study efforts have been devoted to improving $I/O$ performance. Typically, $I/O$ performance optimization falls into two camps: interface optimizations and storage optimizations [24]. Interface optimizations mainly focus on request re-arrangement by combining several small requests into a larger contiguous request or requests re-scheduling. Storage optimizations which are usually implemented in the file server layer, mainly focus on data re-organization such as data partition [9,7,26,33,16], replication [8,1,12,18,4] or file layout [25,36,32,35,37,28] to increase the degree of $I/O$ parallelism. File layout of a parallel file system is often restricted by data access patterns occurring at the file system level which needs sufficient application $I/O$ workload information. Therefore, data layout manners and user request patterns seriously affect the $I/O$ performance in parallel file systems, but current strategies are not designed to catch the data access pattern, and this limitation will be well addressed in this study.

In parallel file systems, file is distributed among multiple data nodes. The efficient allocation of data nodes to files is the task of file dispatcher. The quality of the file dispatcher has a high impact on the overall performance of the parallel file systems. To this end, there have been numerous research efforts dedicated to file assignment strategy in parallel file systems.

In the early 1960s, researchers investigated Greedy, one of the most well-known static heuristic file assignment algorithms, which aims to minimize the disk utilization by balancing the system load across all disks. It is originated from Longest Processing Time (*LPT*) algorithm that aims at multiprocessor load balancing [6]. From 1980s, needs in high performance file systems have turned the research focus to improving the data retrieval performance [5,29,21,17,27]. With the advances of distributed systems, new algorithms have been developed to solve lots of problems such as data object replica placement [11,15], data management for large-scale storage systems [31] and work-conserving migration [3]. These problems are closely related to the *FAP* problem. The *FAP* exists in a wide range of distributed system such as distributed databases, distributed file systems, content distribution networks, video servers and Grid. The solution for *FAP* commonly consists of static strategies and dynamic strategies. Both strategies target at optimizing mean response time and throughput by minimizing the queuing delays on data nodes.

According to Lee et al. [13], the utilization of each disk and the variance of service times mainly determine the queuing delay at each disk. Many heuristic algorithms for file assignment try to improve the performance by minimizing the variance of disk loads and service times. They distribute load among all disks for balancing. *Greedy* is one of the most famous algorithms which aim at multi-processor load balancing. The basic idea of *Greedy* is to calculate the mean load of all files first and then assign a consecutive set of files whose total load is equal to the mean load onto each disk. Another well-known static file assignment algorithm Sort Partition (*SP*) [13] tries to improve the mean response time by minimizing the variance of service time among all disks.It first sorts all files in descending order of their service times into a list and calculates the average disk load with the load of all files. Then *SP* assigns a contiguous set of files in the list to each disk until the disk load reaches the average disk load. After one round of allocation, all remaining files in the list are assigned to the last disk. *SP* improves the performance of the *Greedy* algorithm by minimizing the variances of service times at each disk. *SOR* was proposed to get over the workload characteristic assumed by *SP*. The basic idea of *SOR* is to assign all files sorted in their size onto an array of disks in a round-robin fashion.

The above mentioned *Greedy*, *SP* and *SOR* are off-line file assignment algorithms. Lee also proposed an online algorithm Hybrid Partition (*HP*) to address the dynamic file assignment with batch arrival. Files in each batch are sorted in descending order of service times upon the batch's arrival. *HP* assigns files to disks in turns of batches at distinct allocation intervals. For

Download English Version:

https://daneshyari.com/en/article/524298

Download Persian Version:

https://daneshyari.com/article/524298

Daneshyari.com