DOI:10.13718/j.cnki.xsxb.2018.05.017

一种分布式文件系统的推送式数据预取机制①

夏　苑[1]，　何映思[2]

1. 西南大学 经济管理学院，重庆 400715；
2. 西南大学 计算机与信息科学学院，重庆 400715

摘要：为了克服传统的数据预取机制的不足，提出了一种在分布式文件系统中存储服务器端的推送式数据预取机制. 在达到减少网络通信的数据量和通信时延目的的同时，可以让运行客户端文件系统的计算节点从跟踪 I/O 操作和预测 I/O 操作的工作中脱离出来，减轻了客户端的工作负担，提高了存储系统的性能.

关　键　词：分布式系统；数据预取；云计算；大数据

中图分类号：TP333　　**文献标志码**：A　　**文章编号**：1000－5471(2018)05－0101－05

随着云计算技术的推进和大数据(Big Data)时代的来临，CPU 与磁盘间的性能鸿沟问题变得日益严峻，并引起了国内外学者以及研究机构的广泛关注. 为了向云计算环境中的大数据应用程序提供更好的 I/O 系统性能，国内外学者近年来提出了包括写操作合并和数据预取在内的一系列 I/O 优化技术[1－10]，其中数据预取机制可以很大程度提高分布式文件系统的读吞吐量. 但是传统的数据预取机制要求客户端文件系统追踪 I/O 访问历史记录，通过分析历史记录去准确地预测未来的 I/O 访问信息，从而支持数据预取. 然而，云计算环境中用户终端设备往往需要租用云计算平台所提供的存储资源和计算资源以支持应用程序的运行，而其自身的计算能力和电源续航能力往往非常有限. 因此，如何让用户终端设备从数据预取操作中最大程度脱离出来，关注其自身的应用程序的调度和执行，从而最大限度延长设备的活跃时间的研究显得尤为重要. 目前还没有国内外学者针对这一问题进行研究，而且传统的数据预取机制的每次预取操作都涉及到来回程的网络通信，不利于提高网络通信的带宽. 如何减少数据预取操作带来的网络通信数据量和网络通信时延，进一步提高分布式文件系统性能还需要进一步的研究.

综上所述，减少数据预取过程中网络通信时延和对客户端文件系统的干预，对于提高分布式文件的整体性能至关重要. 因此，本文提出了一种在分布式文件系统中存储服务器端的推送式预取机制，以克服传统数据预取机制的不足. 在不需要客户端文件系统干预的情况下，存储服务器通过记录和分析磁盘 I/O 操作，预测将来的 I/O 操作并预取数据，主动将数据推送给相应的客户端文件系统，从而达到提高文件系统性能的目标.

1　推送式数据预取机制

1.1　系统结构图

图 1 为本文提出的在分布式文件系统中存储服务器端的推送式预取机制的基本结构. 即在存储服务器端通过捕获和分析物理 I/O 访问历史记录，预测将来的 I/O 操作，准确地支持数据预取并推送至相应的客

① 收稿日期：2017－07－07
基金项目：基于生物影像分析和多层次代理模型的脑癌发育模拟及优化方法的研究(20130695).
作者简介：夏　苑(1980－)，女，四川南溪人，讲师，博士研究生，主要从事复杂系统和分布式系统方向的研究.

户端文件系统，从而提高I/O系统的性能.

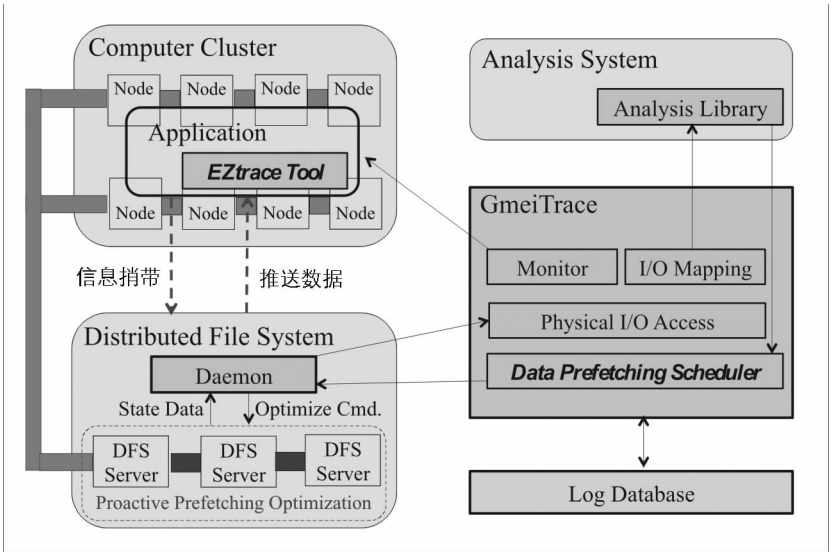

图1 支持推送式数据预取的分布式文件系统结构图

1.2 基于概率计算的预测模型

本文提出的推送式数据预取机制是在存储服务器端通过分析磁盘I/O访问历史记录，预测下一次访问的具体位置，从而准确地帮助存储服务器预取数据. 图2为基于概率计算的预测模型.

X 表示I/O访问模式，$X \in \{X_{se}, X_{re}, X_{st}, X_{ser}, X_{rer}, X_{str}, X_{ra}\}$，其中：$X_{se}$ 表示顺序，X_{re} 表示反向，X_{st} 表示等差跳跃，X_{ser} 表示顺序随机(最开始一段为顺序，后面为随机)，X_{rer} 表示反向随机，X_{str} 表示等差跳跃随机，X_{ra} 表示全随机.

P 表示某种访问模式在磁盘I/O访问历史记录中发生的概率，对应访问模式的表示方法，$P \in \{P_{se}, P_{re}, P_{st}, P_{ser}, P_{rer}, P_{str}, P_{ra}\}$

基于概率计算的预测模型(图2)算法步骤如下，在存储服务器端：

1) 计算出不同I/O访问模式在磁盘I/O访问历史记录中的概率值，即计算(X, P).

2) 设读数据量的大小为 M，将 M_n 等分为 $M_1, \cdots, M_n$(在本文实验中读数据量为8 Gb，取 n 为10).

3) 在 M_1 中先读取 q 个存储块数据(q 取 M_1 大小的1/10)，如果这 q 个存储块的访问模式为 X_{se} 且 $P_{se} >= P_{ser}$，那么 M_1 中剩余存储块的预取方式为 X_{se}，反之取消预取操作；如果这 q 个存储块的访问模式为 X_{re} 且 $P_{re} >= P_{rer}$，那么 M_1 中剩余存储块的预取方式为 X_{re}，反之取消预取操作；如果这 q 个存储块的访问模式为 X_{st} 且 $P_{st} >= P_{str}$，那么 M_1 中剩余存储块的预取方式为 X_{st}，反之取消预取操作；如果这 q 个存储块的访问模式为 X_{ra}，那么取消预取操作.

4) 对 $M_2, \cdots, M_n$ 重复操作3)，完成对 M 大小数据的读操作.

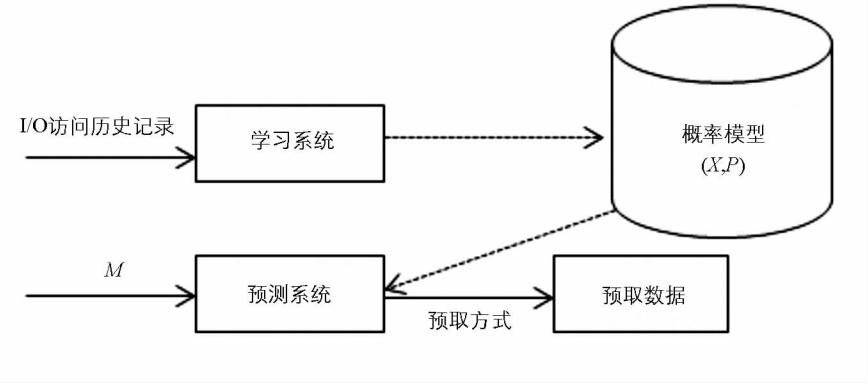

图2 基于概率计算的预测模型

1.3　推送式数据预取

与传统的数据预取机制不同，本文提出的推送式数据预取机制不需要客户端文件系统执行 I/O 访问的预测任务也不需要发送数据预取请求. 具体来说，推送式数据预取机制不需要客户端文件发送预取请求，预取到的数据是由存储服务器主动地推送至文件系统客户端；而且，由存储服务器通过捕获和分析 I/O 访问的历史记录来预测下一次 I/O 访问，避免了客户终端消耗有限耗能执行预测操作(图 3).

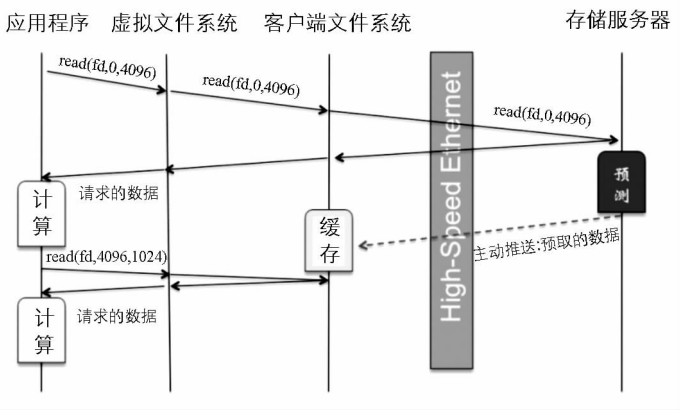

图 3　推送式数据预取机制

2　实　　验

2.1　实验平台

采用服务器集群和两个局域网络来进行评估实验. 我们在服务器集群上部署了由 1 个元数据服务器和 4 个存储服务器组成的分布式文件系统. 所有的客户端文件系统安装在彼此隔离的 2 个局域网的 12 台终端上. 为了模拟一个分布式计算机环境，其中 6 个终端安装在一个千兆局域网络，另外 6 个终端安装在另一个百兆局域网络. 所有的客户端都安装了 MPICH2-1.4.1. 存储服务器和客户端的配置规格表见表 1.

表 1　存储服务器和客户端的配置规格表

项　目	存储服务器	客户端
CPU	2×Intel E5410 2.33 GHz	Intel E5800 3.20 GHz
Memory	1×4 GB 1 066 MHz/DDR3	4 GB DDR3-SDRAM
Disk	6×114 GB 7 200 rpm SATA	500 GB 7 200 rpm SATA
Network	Intel 82 598 EB，10 GB	1 000 MB or 100 MB
OS	Ubuntu 13.10	Debian 6.0.4

我们采用下面几种方式来进行评估实验：①Non-prefetching 表示没有预取功能的分布式文件系统；②Readahead prefetching 表示传统的顺序数据预取机制(如果服务器没有非连续的数据丢失，文件系统会向前预取一定的数据长度，在本实验中向前预取的数据大小为 32 个存储块)；③Server-side prefetching 表示本文提出的数据预取机制，即基于概率计算在存储服务器端预取数据存储块.

本实验采用的基准测试程序为 IOzone(http：//www.iozone.org). IOzone 是一种流行的基准测试程序，可以有不同的 I/O 访问模式，如顺序、随机、反向、跳跃等，因此比较适合用作本实验的基准测试程序.

2.2　实验结果

图 4 为不同 I/O 访问模式下的 IOzone 基准测试程序的实验结果. 由图 4 知，本文提出的数据预取机制在运行基准测试程序 IOzone 时表现得最好. 特别是在反向和跳跃两种读取模式下，与不采用数据预取及采用传统的顺序预取机制相比，本文的数据预取机制达到了较高的读吞吐量. 预测命中率结果见表 2. 由表 2 可知，本文提出的数据预取机制在随机访问模式中命中率不高，因此它更适合于固定访问模式下的数据预取工作. 总之，实验结果证明，本文提出的数据预取机制预测精度较好，能更好提升系统的性能.

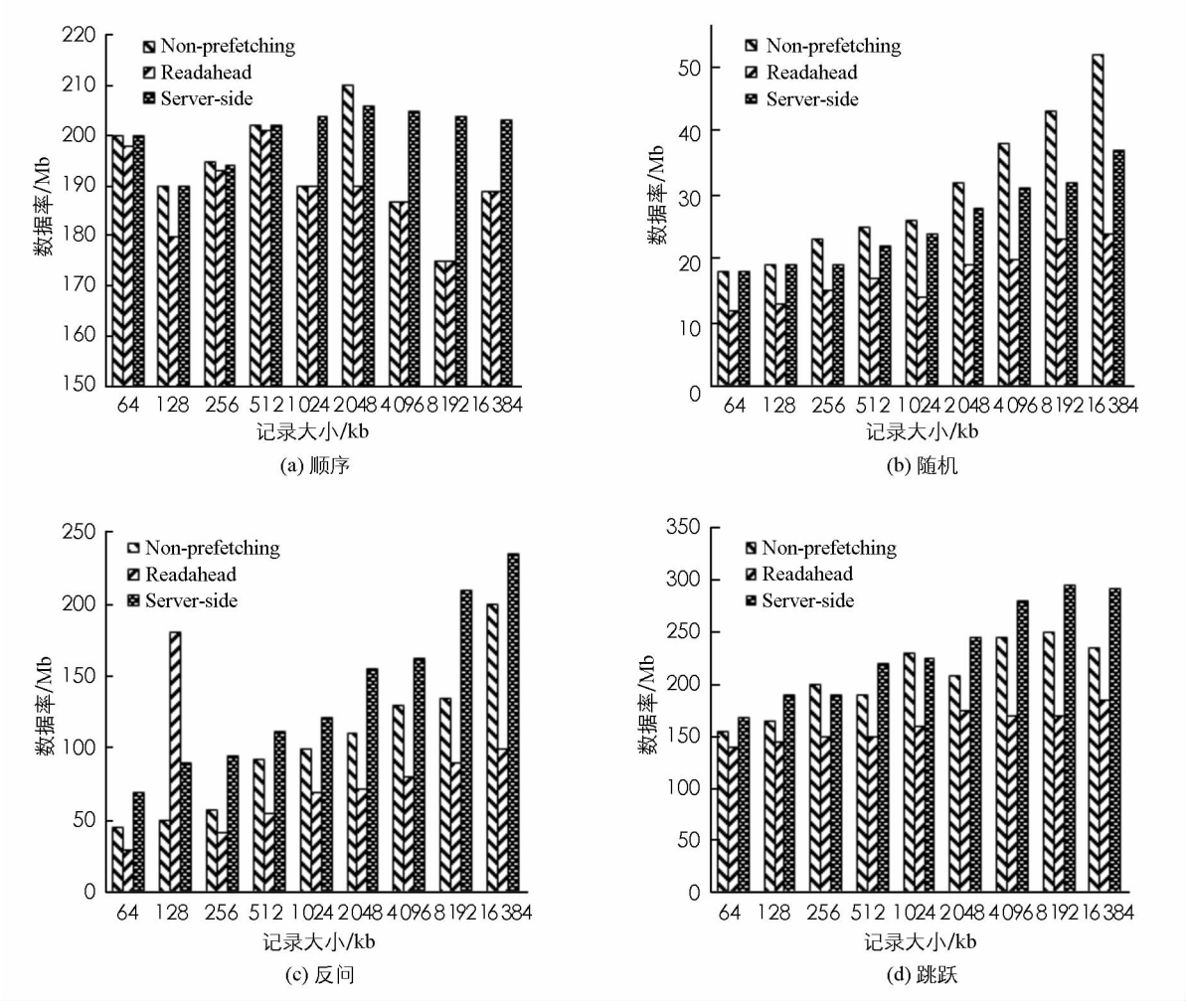

图4 不同I/O访问模式下的IOzone基准测试程序的实验结果

表2 预测命中率

基准测试程序	读出数据量/kb	预测数据量/kb	命中数据量/kb	命中率/%
IOzone：read	2 035 748	1 131 672	916 541	80.99
IOzone：random-read	2 035 720	199 871	96 117	48.09
IOzone：backward-read	1 138 679	981 371	857 521	87.38
IOzone：stride-read	1 138 679	961 380	840 053	87.38

3 结 语

提出了一种在分布式文件系统中存储服务器端的推送式数据预取机制，由存储服务器负责数据预取工作，并主动地把预取到的数据推送给相应的客户端文件系统. 在达到减少网络通信的数据量和通信时延目的的同时，可以让运行客户端文件系统的计算结点从跟踪I/O操作和预测I/O操作的工作中“脱离”出来，从而更关注其自身的工作. 简言之，该研究成果能够满足商业应用和科学应用对存储系统的较高性能的要求.

参考文献：

[1] YU W, VETTER J, CANON R S, et al. Exploiting Lustre File Joining for Effective Collective I/O [C]//IEEE International Symposium on CLUSTER Computing and the Grid (CCGRID '07). New York: IEEE Computer Society Press, 2007.

[2] SEHRISH S, SON S W, LIAO W K, et al. Improving Collective I/O Performance by Pipelining Request Aggregation and File Access [C]// Proceedings of the 20th European MPI Users' Group Meeting. Providence: ACM Press, 2013.

[3] 周　斌，毕传美. 一种基于频繁项挖掘的大量小文件动态合并算法 [J]. 中南民族大学学报(自然科学版)，2016，42(5)：708－714.

[4] KALLURKAR P，SARANGI S R. pTask：A Smart Prefetching Scheme for OS Intensive Applications [C]// IEEE/ACM International Symposium on Microarchitecture. New York：IEEE Computer Society Press，2016：1－12.

[5] HE J，BENT J，TORRES A，et al. I/O Acceleration with Pattern Detection [C]// International Symposium on High－Performance Parallel and Distributed Computing. Providence：ACM Press，2013：25－36.

[6] 吴峰光，奚宏生，徐陈锋. 一种支持并发访问流的文件预取算法 [J]. 软件学报，2010，21(8)：1820 - 1833.

[7] 郝　杰，逯彦博，刘鑫吉，等. 分布式存储中的再生码综述 [J]. 重庆邮电大学学报(自然科学版)，2013，25(1)：30－38.

[8] CHANG F，DEAN J，GHEMAWAT S，et al. Bigtable：A Distributed Storage System for Structured Data [J]. Acm Transactions on Computer Systems，2008，26(2)：1－26.

[9] DEAN J，GHEMAWAT S. MapReduce：Simplified Data Processing on Large Clusters [J]. Communication of ACM，2008，51(1)：107－113.

[10] 温昱晖，陈广勇，赵劲涛，等. 基于 CP-ABE 在云计算中实现数据访问控制的方案 [J]. 重庆邮电大学学报(自然科学版)，2013，25(5)：658－664.

Push Data Prefetching Scheme in a Distributed File System

XIA　Yuan[1]，　HE Ying-si[2]

1. College of Economics and Management，Southwest University，Chongqing，400715，China；

2. School of Computer and Information Science，Southwest University，Chongqing，400715，China

Abstract：In order to overcome the shortcomings of the traditional data prefetching scheme，this paper proposes a push data prefetching scheme in a distributed file system. It can reduce the amount of data communication and network communication delay. At the same time，it can reduce the client's burden of work by freeing the computing nodes running the client file system from tracking I/O operation and predicting I/O operation. So our data prefetching scheme improves the performance of storage system.

Key words：distributed system；data prefetching；cloud computing；big data

责任编辑　张　枸