数据筛选技术在并行 I/O 中的应用

李 冀,李晓明,陆桑璐,陈贵海,谢 立
(南京大学软件新技术国家重点实验室,南京 210093)

摘 要: Collective I/O 是并行 I/O 的一种实现方式,然而科学应用中的请求并不总能很好地满足进行 Collective I/O 的前提条件. 根据工作站网络的特点,在并行 I/O 系统 CION(Collective I/O on Now system)中将数据筛选技术与之紧密结合,充分发挥了 Collective I/O 的功效. 性能测试显示了数据筛选对系统性能的显著提高.

关键词: Collective I/O; 数据筛选; 工作站网络
中图分类号: TP316 **文献标识码**: A **文章编号**: 0372-2112 (2001) 02-0250-03

Applying Sieving Technique in Parallel I/O

LI Ji, LI Xiao-ming, LU Sang-lu, CHEN Gui-hai, XIE Li
(State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China)

Abstract: Collective I/O is a kind of effective organization of I/O subsystem. In the paper we analyze the data sieving technique and present the design and implementation of a parallel I/O system. The performance evaluations have shown good I/O performance.

Key words: Collective I/O; data sieving; NOW

1 引言

在工作站网络环境中,为了给应用提供高带宽的 I/O,在硬件结构上一般使用工作站网络的虚拟磁盘阵列. 这时有两个问题需要考虑[1]. 一是大文件的存储结构. 为了实现并行 I/O,提高对文件的访问速度,并行文件系统一般都采用文件分片的方式,将一个大文件分割成多个分片并以某种方式分布在磁盘阵列中. 二是 I/O 子系统的实现方式. 主要是如何实现一个高效的 I/O 子系统,采用什么方式完成应用程序 I/O 请求. 工作站网络的硬件结构在现有条件和环境下是基本固定的,关键是在软件层面上采用什么方式来高效地实现 I/O 子系统.

通过上述分析,在工作站网络环境下提供高性能的 I/O 所采取的主要解决方案是: (1) 循环分片. 将大文件进行分片并将其分布在工作站网络的各结点上,有助于提高文件的并行读写效率. 不同的分片方式对存取模式的 I/O 操作效率的影响很大,关键是选择一种与工作站网络相适应的分片方式. 这解决的是如何为大的 I/O 请求提供高带宽的问题. 工作站网络中的结点是平等关系,性能相近,循环分片简单易管理,因此选择了循环分片方式. (2) Collective I/O 技术. 目的是提高磁盘存取操作的效率,特别是减少大量小的 I/O 请求带来的存取延迟. 关键是充分利用工作站网络的特点设计一种有效的 Collective I/O 的体系结构.

Collective I/O 是一种力图减轻由于映射关系导致的大量小请求所带来的磁盘操作延迟的技术,基本思想是根据磁盘文件数据的分布信息,将各结点上的进程所产生的大量小 I/O 请求合并成少量大的 I/O 请求后再进行磁盘存取,提高了一次磁盘读写的功效. 然而单独的 Collective I/O 对于系统性能的提高并不充分. 在 CION 中将数据筛选与 Collective I/O 相结合,有效地请求合并的效率,初步的测试已经显示了数据筛选对系统性能的显著提高.

本文的组织如下:第二部分详细讨论了数据筛选技术,第三部分给出了 Collective I/O 与数据筛选相结合的 CION 的体系结构,第四部分是性能测试和结果分析,最后是相关工作和结论.

2 数据筛选

Collective I/O 技术将运行于各结点上的并行程序所产生的大量小的且在磁盘上连续存储的 I/O 请求合并成少量大的 I/O 请求,以提高每次磁盘读写的功效. 因此如何提高小请求合并的功效是该技术的关键.

考虑读取矩阵的情况. 假设一个矩阵按行存储在磁盘上,要读取其中的一些元素:[2,1],[2,3],[2,5],[2,7]. 这些元素在磁盘上的存储并不连续,不能对这些请求进行合并,无法采用 Collective I/O 技术,不得不执行四个独立的 I/O 操作. 这种情况造成了 4 个 I/O 调用,每个调用传输的数据量又很小. 可以将 4 个请求加上这些请求之间的间隙作为一个数据块读出,然后从中筛选出用户所需要的数据交给应用程序,这样只要执行一次读操作. 在这种方式下,虽然多读了一些并不需要

收稿日期:2000-04-24;修回日期:2000-11-10

的数据,但对于大多数存取模式而言,请求之间的间隙往往不大,访问一次大块数据节省的时间超过了多读的数据带来的额外操作的代价.这就是数据筛选的基本思想.因此,数据筛选的读分为两个阶段.第一阶段,合并不连续的请求;第二阶段,将读出的数据进行筛选.其过程见图 1.

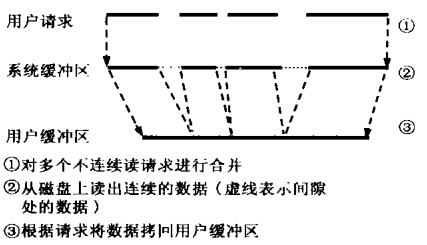

图 1　数据筛选技术

采用数据筛选技术,对于一些间隔不大的分离的读请求,系统只进行一次磁盘操作,读出一块大的连续文件数据,始于所有请求中开始的字节位置最低的,终止于所有请求中结束的字节位置最高的.这一大块数据读入系统内存的缓冲区,然后在根据应用程序的需要对此数据块进行筛选,将有效数据放入应用程序的缓冲区内.数据筛选使得文件访问总是以较大的块进行的,可以减少执行 I/O 的次数,提高每次执行的功效.

数据筛选技术的一个重要问题就是对系统缓冲区的要求.由于这一缓冲区与 Collective I/O 技术中所需的系统缓冲区是一致的,因此并不产生完全新的缓冲区要求(但比不采用数据筛选时的缓冲区要大一些).由于两个请求之间的数据间隙可能很大,如果不对缓冲区的大小做任何约束,会导致非常大的系统缓冲区.因此,在算法中,一方面要规定一个缓冲区上界,一方面要规定何种情况可以采用数据筛选技术,也就是间隙在多大范围内的请求可以合并.从第 4 部分的性能测试可以看出,在内存允许的范围内,应该尽可能使用数据筛选技术,把原先消耗在较慢的磁盘操作上的时间用较快的内存拷贝来替代.

采用数据筛选后写操作的过程与一般的不同,采用先读出再修改最后写回(Read-Modify-Write)的办法.首先,将要写入磁盘的数据所在位置的原磁盘数据读出,然后用要写入的数据覆盖系统缓冲区的相应位置,最后将所有数据写回到磁盘.这里必须考虑文件共享问题.由于此时写操作不是直接写入磁盘的原子操作,可能出现:在此进程的写操作过程中,另一进程修改了处于请求之间的间隙中的数据,但当前进程并不知道而将原先的数据重新写入,导致修改丢失.因此,必须对所要写的范围内的所有数据进行加锁.

3　CION 并行 I/O 系统

在多处理机环境下,根据 I/O 请求是在计算结点还是在 I/O 结点进行全局的分析和合并,Collective I/O 主要包括:Two-Phase I/O[2~5] 和 Disk-Directed I/O (DDIO)[6].较早采用 Collective I/O 的一些并行文件系统或运行库多采用 Two-Phase I/O,由计算结点负责全局的请求合并;后来在 MIMD 多处理机系统提出了由 I/O 结点控制合并更有利于优化,设计出了 Disk-directed I/O.

CION 的体系结构设计为图 2.图中的粗线表示系统中的物理实体.粗线方框表示工作站网络的结点.图中的虚线阴影框表示系统中的逻辑实体,包括 *A*、*B*、*C* 三部分.*A* 代表应用程序;*B* 代表各结点上的进程所组成的实现 Collective I/O 的 I/O 子系统;*C* 是由各结点的磁盘组成的磁盘阵列,存储并行文件.

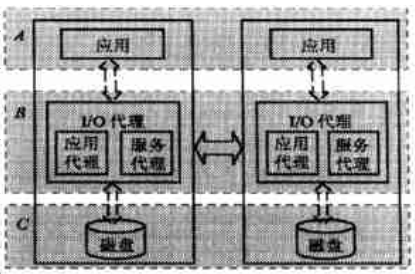

图 2　CION 体系结构

B 是整个系统的核心,连接应用程序与磁盘阵列,由各结点的 I/O 代理组成.I/O 代理以 Collective I/O 和数据筛选相结合的方式完成上层应用程序 I/O 请求.采用将 Two-phase I/O 与 DDIO 相结合的方式实现 I/O 代理.I/O 代理由两部分组成:应用代理和服务代理.

应用代理,是为上层应用提供 I/O 服务的进程,负责接收本地应用程序的 I/O 请求并与此请求所涉及的服务代理进行通信以完成 I/O 操作,它负责:本地应用程序的 I/O 请求的分析和分解;与服务代理的交互.

服务代理,是管理对本地磁盘的 I/O 请求的进程,是小请求的合并者,也是数据筛选的实现者.它负责:I/O 请求的合并;结果数据的分解;数据分析与过程控制;磁盘读写.

其中,数据筛选由服务代理完成.先是各结点的服务代理将自己收到的所有请求进行合并,产生少量较大的连续请求.在读出磁盘数据后,从中筛选出应用真正需要的数据.然后各结点的服务代理返回数据,将读出的数据发送给相应结点的应用代理.

4　性能评测

在 8 台 RS6000 工作站组成的工作站网络环境下对性能进行了测试.RS6000 的 CPU 是 PowerPC,200MHz,内存 128M,每个结点有一个 4.5G的磁盘,操作系统为 AIX4.3.RS6000 之间通过 155M 的 ATM 网相连.用 Java 实现,每个磁盘的读的平均速度是 8.35MB/s,理论上 I/O 总带宽是 66.8MB/s;磁盘的写速度较不稳定,平均是 7.80MB/s,理论上 I/O 总带宽是 62.4MB/s.

为了测试数据筛选的功效,进行了直接存取与数据筛选的对比实验.实验方式是读写一个较大的向量.向量元素为 byte 类型,在磁盘上连续存储.*(m, n, s)* 表示要存取从向量的第 *m* 个到第 *n* 个,每隔 *(s - 1)* 个元素中读取一个.如表中的 (1,8192,2) 表示要从文件中的第 1 个字节开始,隔一个字节读一个,直到文件中的第 8192 个字节.时间单位是毫秒.测试了 8K 和 1M 时各种间隙的情况.表中"筛选"中的时间,对于读,包括了读出后的筛选过程(内存拷贝),对于写,包括了 Read-Modify-Write 的整个过程.

从测试数据可以看出,数据筛选技术对性能有了很大的提高.即使只读或写 8k 的数据时性能的差距也是非常明显.当读或写 1M 的数据时,性能差异可能达到几十倍.

表 1 数据筛选效果测试表

类型 / 数据	读		写	
	直接	筛选	直接	筛选
(1,8192,2)	62	5	71	7
(1,8192,3)	42	5	48	6
(1,8192,4)	33	4	37	6
(1,8192,5)	28	4	31	6
(1,1M,2)	6919	272	8630	324
(1,1M,3)	4624	241	5748	288
(1,1M,4)	3468	228	4320	273
(1,1M,5)	2789	218	3475	264

5 相关工作

Passion[4]采用了 Two-phase I/O,文件的逻辑分片的管理者与文件的物理存储者不一致,无法充分利用数据筛选的优点;而且 Two-phase 方式由于数据的重分布造成的网络负载很大.Dartmouth 提出的 Disk-directed I/O[6]是基于多处理器体系结构的,充分利用文件在磁盘的分布信息,可以优化磁盘控制,但是计算结点与 I/O 结点之间的通信负载较大,而且其实现与操作系统和文件系统密切相关,不易移植.

6 结论

本文分析了数据筛选技术在并行 I/O 中的应用.数据筛选与 Collective I/O 技术的紧密结合,充分发挥了 Collective I/O 技术的效能.在性能测试中,测试了不同的实现方式下系统的 I/O 吞吐量,实验结果显示了数据筛选与 Collective I/O 技术密切相关,应该作为 Collective I/O 技术的必要组成部分.

参考文献:

[1] 李群,靖捷,谢立.基于工作站网络的并行文件系统 [J].软件学报,1997,8(增刊).

[2] R. Bordawekar. Implementation of collective I/O in the Intel Paragon parallel file system: Initial experiences [A]. In Proceedings of the 11th ACM International Conference on Supercomputing [C], ACM Press, July 1997: 20 - 27.

[3] J. Rosario, R. Bordawekar, and A. Choudhary. Improved parallel I/O via a two-phase run-time access strategy [A]. In Proceedings of the IPPS '93 Workshop on Input/Output in Parallel Computer Systems [C], Newport Beach, CA, 1993: 56 - 70.

[4] R. Thakru, A. Choudhary, R. Bordawekar, S. More, and S. Kuditipudi. Passion: Optimized I/O for parallel applications [J]. IEEE Computer, June 1996.

[5] R. Thakur, W. Gropp, and E. Lusk. Data sieving and collective I/O in ROMIO [A]. In Proceedings of the Seventh Symposium on the Frontiers of Massively Parallel Computation, 182 - 189.

[6] D. Kotz. Disk-directed I/O in multiprocessors [J]. ACM Transactions on Computer Systems, February, 1997: 41 - 74.

作者简介:

李 冀 1975 年生,南京大学计算机系硕士研究生,主要研究方向:并行计算与分布式处理.

李晓明 1975 年生,南京大学计算机系硕士研究生,主要研究方向:并行计算与分布式处理.