并行科学计算应用的动态 I/O 模式分析*

卢凯　金士尧

(国防科大计算机系　长沙　410073)

摘　要　设计高性能并行文件系统的关键前提是对 I/O 访问模式的分析和研究。并行文件系统的界面、组织结构、cache 预取算法等都与具体应用的 I/O 模式紧密相关。目前,大多数对 I/O 模式的分析局限于其静态模式,而动态 I/O 模式对并行文件系统的设计至关重要。本文通过分析并行科学计算应用的动态 I/O 访问模式,总结出科学计算应用在访问数据量、时间间隔、访问顺序性等空间、时间上的一般访问特性,并在此基础上提出设计面向并行科学计算应用的并行文件系统应考虑的因素。

关键词　并行科学计算, I/O 模式, 并行文件系统

分类号　TP393

The Analysis of Parallel Scientific Application's Dynamic I/O

Lu Kai　Jin Shiyao

(Department of Computer Science, NUDT, Changsha, 410073)

Abstract　It is necessary to understand the mode of I/O mode before designing a parallel file system (PFS). The structure, interface and algorithm of cache/prefetch is closely related to application. Now, most analyses are based on static I/O model, but the dynamic mode is much more important to PFS. This paper analyses the dynamic I/O mode of parallel scientific application, and summarizes the characteristics of I/O size, interval, sequence and so on. The conclusion of what should be considered when designing a PFS is finally drawn.

Key words　scientific application, I/O mode, parallel file system

大规模并行计算机处理系统(MPP)面对科学计算的巨量 I/O 需求广泛采用了并行 I/O 子系统和并行文件(PFS)技术。由于用户的 I/O 访问模式差别很大,不同应用环境下用户所能获得的实际 I/O 带宽与系统提供的差距甚远。因此,针对不同的应用开发合适的 I/O 服务模型成为人们研究的热点。

分析和研究 I/O 访问模式是开发合适并行 I/O 服务模型的前提,目前主要采用的是静态分析法,即统计应用整个执行过程中的文件打开数、访问数据量、次数、时间等。但还必须分析文件访问的动态特征,通过分析 I/O 访问在时间和空间上的动态关系,才能了解系统的瓶颈和需要重点优化的部分。并且,PFS 的用户界面、并行服务模型和 cache 预取机制等都与应用动态I/O模式密切相关。所以只有在分析应用动态 I/O 访问模式的基础上才能设计出高性能的并行文件系统。

1　并行科学计算应用的 I/O 模式特性

目前,天气预报、地球数据处理、海洋模型应用、遥感数据处理、可视化处理等科学计算应用[1]都有巨量 I/O 需求。我们从上述科学计算应用中选取了几个有代表性的应用: 并行 Render、SAR、PRISM 和 ESCAT,通过分析这些应用的实际 I/O 访问轨迹获取其静态、动态 I/O 模式特性。

并行 Render 应用是一个动态视景生成的可视化处理应用。该应用可根据用户的交互文件来动态地显示用户所希望看到的景象。其 I/O 数据主要是景象的原始信息和着色后的景象数据。SAR(合成孔径雷达信息处理)应用是雷达信息处理应用,广泛应用于研究地球表面物理特征等。其 I/O 数据量根据雷达数据信道数不同而不同。PRISM 和 ESCAT 分别是三维海洋流模拟应用和化学实验模拟应用。

* 1998 年 9 月 11 日收稿
第一作者: 卢凯, 男, 1973 年生, 博士生

2 应用的静态 I/O 特性

并行程序可分为数据并行的单程序流多数据流(SPMD)和控制并行的多程序流多数据流(MPMD)。目前,并行科学计算应用的主要编程模式是SPMD方式。但一个应用往往同时包含数据并行和控制并行方式。有经验的程序员更喜爱MPMD方式[2],因为它灵活、资源利用率高。因此,在设计并行文件系统时应兼顾SPMD和MPMD方式。SPMD和MPMD方式中,I/O主要有三种访问模式:所有节点同步读取相同数据集的广播方式;所有节点同步执行I/O操作,但I/O数据集不同的同步方式;各个节点独立I/O的独立方式。SPMD中,I/O模式以前二种为主,也有少量独立方式,这主要是运行了不同条件分支产生的。MPMD中,同样广泛存在同步I/O方式,但独立I/O方式大大增多。

表1和表2显示了并行Render和SAR应用的I/O静态特性。并行Render总共打开106个文件。其中,只读文件占总数的5.6%,而输入数据量有880M,占总I/O量的89%;只写文件100个,输出的数据量仅98M。并行Render应用不存在读写文件。SAR总共打开文件18个,其中只读文件14个,读入的数据量为336M。占总I/O量的86%。只读文件4个,写回数据54M。无读写文件。并行Render的写时间为31.7s,占19.3%;读和异步读的时间仅为4.8s,占总时间的2.89%。实际上由于大量异步I/O与计算并未完全覆盖,由此产生的等待时间为88.4s,故因读操作引起的I/O等待时间应为93.2s,占56.6%。SAR的总I/O时间为112s,其中读操作有55s,占49%;异步写操作4.7s,占4.3%。

表1 并行Render静态I/O特性

Table 1 Parallel Render's static feature

	只读文件	只写文件	读写文件
文件数	6	100	0
I/O量	880M	98M	0
时间	93.2s	31.7s	0

表2 SAR静态特性

Table 2 SAR's static feature

	只读文件	只写文件	读写文件
文件数	14	4	0
I/O量	365M	54M	0
时间	55.25s	4.76s	0

3 I/O 模式的动态特性

3.1 并行 Render 的动态 I/O 模式

并行Render应用的I/O操作和时间关系分布图(图1)显示其执行过程可分为三个阶段。

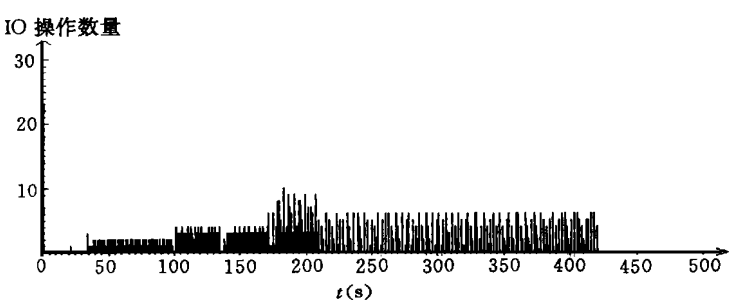

图1 并行Render应用的I/O操作时间分布图

Fig. 1 Time distribute of Parallel Render's I/O access

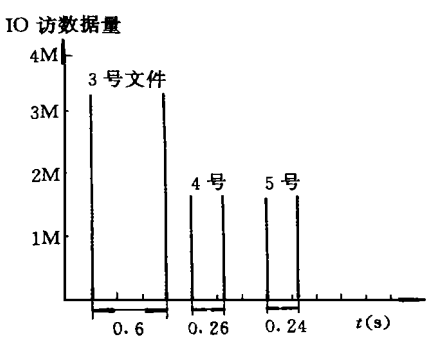

图2 异步读时间分布图

Fig. 2 Time distribute of Asynch read

第一阶段是应用初始化阶段。该阶段的I/O操作是节点128读入初始化数据。节点128采用19个短读结束该过程。第二阶段并行Render的初始化阶段,I/O操作十分频繁。0号节点从三个文件中读取大量的Render原始信息,再通过异步通讯机制广播给其它所有节点。每个节点则提取自己所需的数据处理。这是一种同步的I/O操作方式。第三阶段是应用的着色阶段。应用根据用户的观测轨迹执行着色过程。其I/O操作主要是从轨迹文件中读取数据和输出结果。表3显示了并行Render三个阶段的

I/O动态特性。

表 3 并行 Render 的动态特性

Table 3 Parallel Render´s dynamical feature

	第一阶段		第二阶段		第三阶段	
	读	写	读	写	读	写
open 的文件数	1	0	3	0	2	100
I/O 量	555	0	880M	0	7902	98M

表 3 表明文件的生存期往往仅限制在一个阶段中。并且一个应用打开的文件数有限。

并行 Render 应用的主要 I/O 行为发生于第二阶段。该阶段中异步读取的数据有 880M。应用异步读的I/O 量的时间分布图如图 2 所示。0 号节点先后从三个文件中顺序读取数据。在读取 3 号文件时, 读间隔约 0.6s, 读取数据量均为 3,244,800 字节。读取 4 号 5 号文件时, 间隔约 0.25s, 每次读取信息量相同, 为 1,622,400 字节。该过程是各个节点同步获取数据的同步I/O 过程, 因此I/O 数据量较大。从图 2 可知, 该应用的 I/O 操作十分有规律。

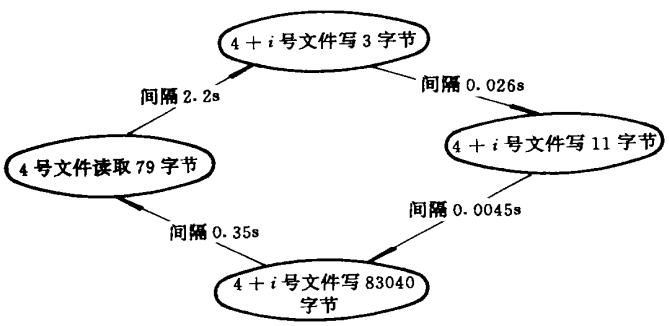

图 3 Render 阶段中第 i 个 I/O 操作循状态图

Fig. 3 No. i I/O access loop in Render Phase

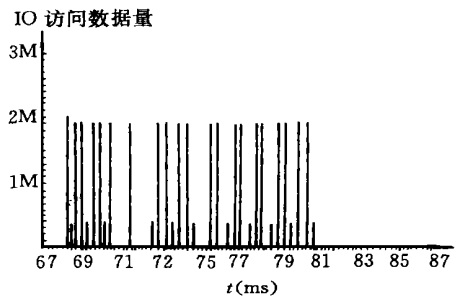

图4 191、159 号节点读操作时间分布图

Fig. 4 Time distribute of 191 and 159 node´ read

第三阶段的主要 I/O 操作是 128 号节点读取轨迹文件, 各个节点根据轨迹着色并输出景象数据。该阶段中存在大量小尺寸 I/O, 并且其 I/O 时间分布、访问数据量十分有规律: 表现为多个循环反复出现(图 3)。在第 i 个循环中, 当从轨迹文件 4 号文件读取数据 79 字节后 2.2s 向4+ i 号文件输出 3 字节信息, 然后又分别间隔 0.026s 和0.0045s 向该文件写入 11 字节和约 80K 的数据。系统在 0.35s 后进入下一循环。

3.2 SAR 的 I/O 动态模式

SAR 应用将处理机节点分成若干个处理机组, 每组 32 个节点, 各处理一路信号。各组间的相互通讯较少。分析表明各个处理节点组的 I/O 模式基本相同。

SAR 应用同样可分为三个阶段: 预处理阶段、主计算阶段和后处理阶段。

主要的I/O 操作发生在后二阶段。第二阶段的主要 I/O 操作是读。8 个节点组共打开 14 个文件。从以 191 和 159 节点为代表的节点组读操作时间分布图(图 4) 可看出其 I/O 行为在时间间隔和访问数据量上的规律性。图 5 显示了 159 号节点读取文件的空间特征。产生规则跳跃现象的原因是处理机组按块处理数据矩阵。各组中所有节点的数据都通过代表节点读取后再分发, 所以该同步 I/O 操作的数据量也较大。

第三阶段是后处理阶段, 其主要 I/O 行为是将处理结果写回磁盘。4 个节点按顺序方式写回各自的结果文件中, 不存在交叉现象。其中 191 号节点的写操作时间分布图如图 7 所示。每个写操作的尺寸相同。

3.3 ESAT 和 PRISM 应用

在 ESCAT 和 PRISM 应用中, I/O 以独立 I/O 为主。PRISM 的 I/O 主要集中在第一阶段, 各个节

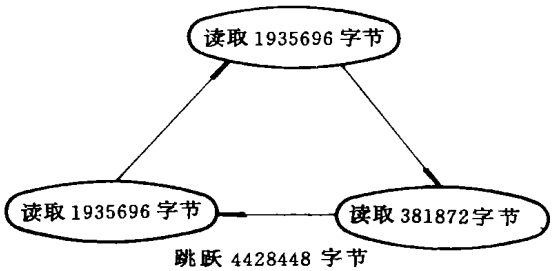

图5 159号节点读操作空间分布图

Fig. 5 Space distribute of 159 node's read

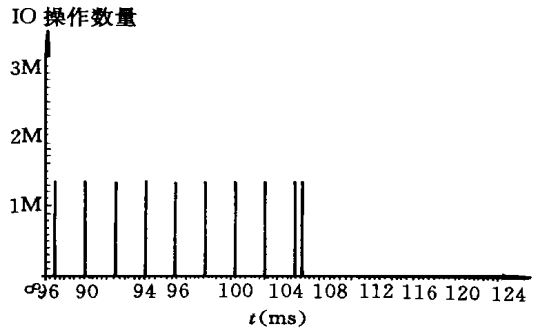

图6 191号节点的写操作数据量/时间分布图

Fig. 6 191 node's write bytes/time

点采用共享方式读取初始数据文件。由于是独立I/O方式,故其一次I/O的数据量较小。其中8字节的read有299,328个,占总read的99.95%。独立I/O时各个节点按共享方式顺序读取文件。同样,各节点的合作读也存在很好的时间、空间规律性。以上这种特性在ESCAT中也存在。

4 结论

(1) 科学计算应用同时存在同步I/O和独立I/O方式。由于同步I/O是多个节点统一操作,因此I/O数据量往往较大。独立I/O是单个节点独立操作,其I/O尺寸较小(David发现有90%的I/O尺寸小于4K[3])。同步I/O要求系统提供阵发性的大I/O带宽,独立I/O则对服务延迟较为敏感。

(2) 科学计算应用可以明显地划分成若干阶段,各阶段内的I/O访问模式都有较强的规律性。这种规律性体现在时间和空间上:

· 每个I/O操作都有一定的时间间隔,并且在同一阶段内间隔相似。I/O访问时间间隔一般大于1ms,这就允许在服务开销不大的情况下采用一定的集中式服务结构,并且该结构不会成为瓶颈。

· I/O操作在空间上呈现良好的整体顺序性。这种顺序性以连续方式和等跨度顺序方式为主,并且每次访问的数据量相同。从各个节点看,I/O访问的顺序性又分为单一访问顺序性和合作访问顺序性。在前方式中,一节点对一文件单独顺序访问;在后方式中,多个节点共享文件,整体上顺序访问。

(3) 在科学计算应用中,大多数文件按只读或只写方式打开。即使是读写文件,其在同一阶段内也呈现单一状态。各个节点I/O数据很少存在互覆盖现象。科学计算应用I/O的主要方式是读,所以并行文件系统必须重点对读提供优化。文件生存期往往在一个阶段内,并且一阶段内文件访问又呈单一的规律性。所以,我们可以认为一个文件的访问规律性较单一,可按文件来设置预取和cache策略。

综上所述,高性能的并行文件系统应该对不同I/O请求类型提供优化。对于同步I/O,系统应该提供同步能力。并且因此类I/O操作数据量大,对带宽要求高,故应回避cache直接访问磁盘。对于独立的分散小尺寸I/O,并行化的I/O服务方式无法提供优化,系统则应该通过cache机制提供低延迟服务。故面向科学计算应用的并行文件系统应该设计两套界面、两套处理方案,并将二者有机地结合起来。在设计cache和预取算法时,应充分考虑应用的I/O特性,同时针对整体顺序和单一顺序以及连续访问和等跨度顺序访问等提供优化。

参考文献

1 David Kotz, File-System Workload on a Scientific Multiprocessor, IEEE Parallel & Distributed Technology, 1995: 51～67

2 Cherri M. Pancake, Is parallelism for you? IEEE Computational Science & Engineering

3 David Kota. Applications of Parallel I/O, ftp://ftp.cs.dartmouth.edu/TR/TR96-297.ps.Z