

文章编号:1001-9081(2009)02-0421-03

RAID 小数据随机访问性能分析与优化

刘 冬,王丽芳,蒋泽军,刘志强

(西北工业大学 计算机学院,西安 710072)

(yoskie@gmail.com)

摘 要: RAID 采用条纹结构,使多块磁盘可并行访问,提高了带宽,适合于大块数据顺序访问,而对小块数据随机访问影响不大。针对 Stripe 条纹大小对 RAID 的读写性能进行分析,探讨多用户小数据访问模式下的 IOPS (IOs per second) 问题,提出粗粒度条纹布局模型。仿真实验表明:该模型的性能优于现有布局方式,显著提高小块数据随机访问性能。

关键词: 独立磁盘冗余阵列技术;条纹;IOPS;粗粒度

中图分类号: TP333.3⁺5 **文献标志码:** A

Analysis and optimization for RAID small data random access

LIU Dong, WANG Li-fang, JIANG Ze-jun, LIU Zhi-qiang

(School of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an Shaanxi 710072, China)

Abstract: In order to make multiple disks accessible parallelly, the stripe layout is adopted by RAID for high-bandwidth. The approach is appropriate for big sequential data access, but is not that effective for small data random access. This paper analyzed the I/O performance based on RAID Stripe size, explored the IOPS (IOs per second) of multiple user small data access, and proposed a new stripe layout architecture—Thick Granularity Model. The simulation indicates: the I/O performance of Thick Granularity Model has been improved, especially in small data random access.

Key words: Redundant Arrays of Independent Disks (RAID); stripe; IOs PerSecond (IOPS); thick granularity

0 引言

独立磁盘冗余阵列技术 (Redundant Arrays of Independent Disks, RAID)^[1] 通过条纹划分,对大块数据,可并行地从多个磁盘进行读写操作,极大地提高了设备带宽,在广电、高清视频等流媒体领域可大幅度提高性能。但是对小块数据读写性能的提高不大,尤其是对于有校验信息的 RAID5、RAID6 而言,存在小写问题,使得性能大幅度下降,因此,提高磁盘阵列的小块数据读写性能具有重要意义。

文献[5-6]对 I/O 响应时间及吞吐量做了分析,并提出了条纹热度的概念,使得在顺序访问中得到良好的性能,但并没有很好地解决小块数据访问的并发性问题。本文从并发访问时间分析,针对 Stripe 条纹大小对 IOPS 的影响,提出一种新的粗粒度条纹布局模型,可以显著提高多用户小数据访问的并发性,使得 RAID 系统可以更好地应用于对 IOPS 性能要求较高的银行、证券等领域。

1 磁盘访问时间

磁盘作为随机访问设备,从 CPU 将 SCSI 命令经由总线递交到设备以后,访问时间 T_{access} 主要由寻道时间 T_{seek} 、延迟时间 T_{delay} 和数据传输时间 T_{data} 三部分构成^[2]。

$$T_{\text{access}} = T_{\text{seek}} + T_{\text{delay}} + T_{\text{data}} \quad (1)$$

对于大量读写的存储系统, T_{seek} 寻道时间服从 $[0, T_{\text{seek-max}}]$ 之间的独立随机分布,寻道时间取决于磁头到目的位置的磁道数,与磁道数的出现概率,满足泊松

分布,因此寻道时间 T_{seek} 同样满足泊松分布。每次的寻道时间记为 T_1, T_2, \dots, T_n , 当 n 取值足够大时有:

$$\sum_{n=1}^{\infty} T_n = n * E(T_{\text{seek}}) \quad (2)$$

同理可以计算平均延迟时间和平均数据传输时间。

2 RAID 并发分析

2.1 RAID 访问时间分析

RAID 采用条纹将磁盘进行条纹划分,其结构如图 1 所示。将各个磁盘物理地址相同的部分划分为一个条纹,条纹所包含的物理磁盘数称为条纹宽度,条纹在每个磁盘所划分的区域称为一个条纹单元 strip,条纹单元包含的扇区数称为条纹深度。

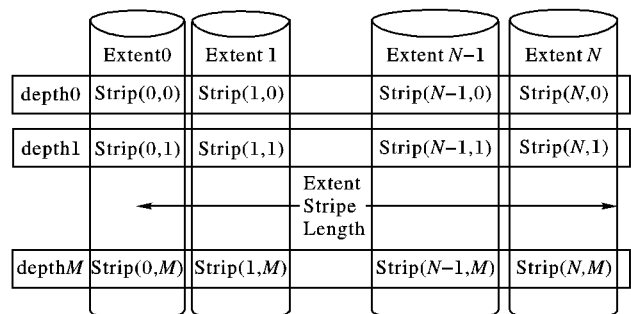


图 1 条纹划分结构

磁盘阵列通过条纹共同控制各个磁盘,当访问大块数据 (假定数据足够大到包含至少一个条纹),各磁盘同时访问数

收稿日期:2008-08-07;修回日期:2008-09-28。

作者简介: 刘冬(1985-),男,河南延津人,硕士,主要研究方向:分布式计算、计算机存储系统; 王丽芳(1964-),女,内蒙古齐齐哈尔人,教授,CCF 会员,主要研究方向:嵌入式系统、分布式计算; 蒋泽军(1964-),男,安徽合肥人,教授,CCF 会员,主要研究方向:嵌入式系统、网络安全、计算机存储系统; 刘志强(1975-),男,内蒙古赤峰人,博士,CCF 会员,主要研究方向:传感器网络、复杂网络、网络存储。

据,这样增加了磁盘的吞吐量,减少了数据传输时间。对于 n 块磁盘构成的无冗余阵列结构, I/O 响应时间为^[3]:

$$S = t_1 + n * t_2 + t_3 + t_4 \quad (3)$$

其中 S 为 I/O 响应时间, t_1 为 CPU 命令处理时间, t_2 为启动器时间, t_3 为 I/O 处理时间, t_4 为中断处理时间。由于 t_3 为 ms 级别, 而 t_1, t_2, t_4 均为 ns 级别, 对 S 影响较小, 因此本文仅针对 t_3 做出分析处理。在 I/O 处理过程中, 有 n 块磁盘同时进行磁盘 I/O 操作, 磁盘的同步使得处理时间取各磁盘处理时间的最大值。RAID 中 n 台磁盘独立处理, $T_i (i = 1, 2, \dots, n)$ 之间相互独立有:

$$t_3 = \max_{1 \leq i \leq n} T_i = \max_{1 \leq i \leq n} (T_{\text{seek}} + T_{\text{delay}} + T_{\text{data}}) = \max_{1 \leq i \leq n} T_{\text{seek}} + \max_{1 \leq i \leq n} T_{\text{delay}} + \max_{1 \leq i \leq n} T_{\text{data}} \quad (4)$$

n 台磁盘同步所需的时间开销^[4] $\Delta T_{(n)} = (0.5 - \frac{1}{(n+1)}) \times s$, 由式(3)、(4) 可得:

$$S \approx t_3 = T_{\text{seek_average}} + \Delta T_{(n)} + T_{\text{delay_average}} + \max_{1 \leq i \leq n} T_{\text{data}} \quad (5)$$

假定现有 n 个任务需要进行处理, 每个任务访问数据块大小均为 K , 那么磁盘阵列设备处理任务所需要时间为:

$$T = n \times S = n \times T_{\text{seek_average}} + n \times \Delta T_{(n)} + n \times T_{\text{delay_average}} + n \times \max_{1 \leq i \leq n} T_{\text{data}} (\lceil K/n \rceil) \quad (6)$$

假定 K 为条纹整数倍, 有:

$$T_1 = n \times T_{\text{seek_average}} + n \times \Delta T_{(n)} + n \times T_{\text{delay_average}} + T_{\text{data}}(K) \quad (7)$$

假定 $\text{Stripe} < K < 2 \times \text{Stripe}$, 有:

$$T_2 = n \times T_{\text{seek_average}} + n \times \Delta T_{(n)} + n \times T_{\text{delay_average}} + 2 \times T_{\text{data}}(\text{Stripe}) \quad (8)$$

显然 T_2 耗用的时间大于 T_1 , 尤其在 $\text{Stripe} < K \leq \text{Stripe} + \text{strip}$ 时, 对数据处理时间会造成极大的浪费。

假定有 n 块磁盘来存储需要访问的数据, 而每个任务访

问的数据都不在同一磁盘, 此时处理该 n 个任务, 所需要时间为:

$$T_3 = T_{\text{seek_average}} + T_{\text{delay_average}} + T_{\text{data}}(K) \quad (9)$$

而 T_3 又远小于 T_1 。由此可见, RAID 结构采用条纹布局后, 虽然提高了设备的整体吞吐量, 但是并没有发挥出设备的最佳性能, 尤其是对于小块数据的访问, 造成不同磁盘子任务的同步等待, 降低了磁盘设备的利用率。不难得出采用条纹布局的 RAID 结构其 IOPS 明显低于相互独立的磁盘结构。

2.2 RAID I/O 并发访问分析

以小块数据访问为主的联机交易系统, 例如银行、证券等领域, 对设备的 IOPS 会有较高的要求。对于这一领域, IOPS 的提高会极大地加快系统的处理能力, 而大多数设备通常通过磁盘联动技术^[1]、Cache 技术^[5], 提高对大块数据访问的设备带宽, 对随机访问的 IOPS 影响并不大。

在 N 块数据磁盘的 RAID 结构中 (RAID0 为无冗余结构, 结构最为简单, 本文分析的理论数据均以 RAID0 为准), 做如下假定:

①数据随机访问, 长度 K 满足泊松分布;

②命令队列中任务相互独立, 且有足够多的命令可供操作;

③RAID 结构中请求数据为 K 时访问延迟时间与独立磁盘访问时间比记为 S ,

$$S'(data, n) = T_{\text{seek_average}} + \Delta T_{(n)} + T_{\text{delay_average}} + T_{\text{data}}(data) \quad (10)$$

$$S'' = T_{\text{seek_average}} + T_{\text{delay_average}} + T_{\text{data}}(N) \quad (11)$$

$$S = S' / S'' \quad (12)$$

④在出现大小为 K 的命令时, 命令足够多。

现在比较两个不同条纹大小的 RAID 结构, $\text{Stripe}_1 \geq 2 \times \text{Stripe}_2$ 。表 1 给出了不同请求数据 K 值对两个 RAID 结构的 IOPS 影响。

表 1 数据块大小对不同条纹结构的 IOPS 影响

数据块	S_{Stripe1}	$IOPS_{\text{Stripe1}}$	S_{Stripe2}	$IOPS_{\text{Stripe2}}$	S_1/S_2
$K \leq \text{strip}_2$	S''	N/S_1	S''	N/S_2	1
$\text{strip}_2 < K \leq \text{strip}_1$	S''	N/S_1	$S'(\text{strip}_2, 2)$	N/S_2	$(0.5, 1)$
$\text{strip}_1 \leq K \leq \text{Stripe}_1$	$S'(\text{strip}_1, \lceil K/\text{strip}_1 \rceil)$	N/S_1	$S'(\text{strip}_1, n)$	N/S_2	< 1
$\text{Stripe}_1 \leq K$	$S'(\lceil K/\text{strip}_1/N \rceil, n)$	N/S_1	$S'(\lceil K/\text{strip}_2/N \rceil, n)$	N/S_2	≈ 1

由表 1 可以得出, 在小块数据访问时条纹越大其 IOPS 性能越好, 尤其当数据块 K 在两个条纹单元之间时, 效果更加明显; 当 K 值大于较大的条纹后, 随着 K 值的增大, 采用不同条纹大小布局的并发性渐趋相同, 对 IOPS 性能影响的差别也逐渐降低。

2.3 测试与验证

表 2 为在 Iometer 在 DFTraid 3016F4R 磁盘阵列测出的 IOPS 值。

由表 2 可以看出, 随着访问数据块的增大, 不采用条纹布局的独立磁盘其 IOPS 会呈现下降趋势, 幅度比较平缓; 在采用条纹布局的 RAID 结构中, 当数据大于 strip 单元后, IOPS 性能会大幅度下降; 而在数据增大到一定范围后, 条纹大小对性能的影响不再明显, 不同条纹配置的 IOPS 值相近。设备中所测得的数据表明的三种迹象都与文中理论分析的情况相吻合。

表 2 不同 Stripe 的性能表现

Block	Stripe				
	16K	32K	64K	128K	NonRaid
512 B	519.37	520.56	516.19	514.48	517.72
1 KB	507.38	514.47	513.75	518.81	514.37
4 KB	452.49	476.97	487.13	493.74	509.24
8 KB	397.27	432.37	464.89	482.85	510.24
16 KB	304.02	371.71	413.54	450.98	499.88
32 KB	218.20	285.91	342.70	400.63	484.59
64 KB	163.14	202.64	262.51	330.72	458.99
128 KB	147.38	153.81	178.60	245.94	416.42

3 粗粒度条纹模型

文献[6]详细介绍了磁盘磁道距离对 I/O 性能的影响, 在此基础上, 本文提出一种基于区域划分的粗粒度条纹布局模型, 条纹布局采用原 RAID 结构, 在原有布局结构基础上调

整 strip 单元大小,即条纹深度。

现有磁盘大多采用分区技术^[7],靠近磁盘轴心的近区,扇区数较远区少,数据量小, N 个磁盘采用较小条纹布局,当数据块需要跨盘操作时,式(4)中的寻道时间、延迟时间以及同步时间均会降低,具有较短的响应时间,进而提高 IOPS,对于大块数据以及顺序访问,保证了多块磁盘的并发,可以提高系统带宽;在远区磁道中,扇区数较多,采用较大条纹布局,寻道、旋转等待时间都会相应降低,可以更好地提高响应速度,不会过于降低系统的 IOPS,由于模型仍然采用条纹布局方式,在顺序访问操作时,也不会降低吞吐量;而根据磁盘的具体磁道的扇区配置,可以将多个磁道组合为一个条纹单元,增加条纹大小,进一步增加 IOPS。

4 实验结果

采用广泛应用于磁盘阵列的仿真工具——RAIDframe^[8]进行仿真实验。对该软件包进行了改进,修改条纹划分算法,改变布局方式,使其支持本文设计的粗粒度条纹布局模型;采用写穿策略,以随机写为主要访问方式,减小 Cache 对实验性能的影响。

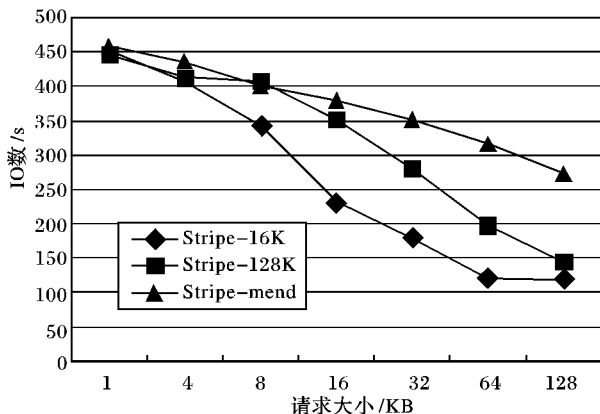


图2 不同数据块请求下的性能比较

(上接第420页)

其中,数字 i 表示 F_i 属性。为使得结果更直观,属性按 F_i 中的 i 的大小从小到大排好序。

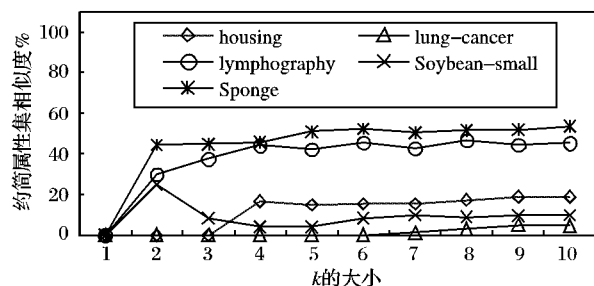


图1 本文算法对表1选取的数据表的实验效果

通过实验发现,本文的算法在 k 较大时,得到的约简属性集的相似度趋于稳定。在条件属性集的个数比较多而各个约简所用到的属性个数比较少,得到的效果较佳。在决策表有较多核属性或者约简属性集的属性个数比较多的情况下效果较差,而这种情况下,各个约简之间的相似度本来就有比较大的相似度。总体上说,本文的算法在计算差异化属性约简问题上可以得到一个较好的效果。

4 结语

本文针对文献[8]提出的如何计算差异化属性约简问

题,提出了一个计算方法并实现了它。把层次聚类的思想应用到差异化属性约简的计算过程中。先对决策表的条件属性集按定义好的距离进行聚类,得到条件属性集的若干个划分;接着,对这些属性子集进行后处理操作,从而得到若干个有较大差异的属性约简集。最后,为说明算法的效果,利用UCI上的数据集对本文算法进行实验,效果良好。

5 结语

本文针对 RAID 条纹布局在随机访问的不同数据块大小下的性能做了整体分析,总结出条纹的增大对 IOPS 性能的影响会逐渐降低,根据磁盘的磁道分布思想提出了粗粒度条纹布局模型。通过实验仿真,验证了模型在小块随机数据请求下具有良好性能,可以减缓 IOPS 的下降趋势。在仿真实验中没有采用缓存,如果为模型设计一种适合的缓存方式,可以使模型的整体性能得以更加显著的提高。

参考文献:

- [1] PATTERSON D A, GIBSON G, KATZ R H. A case for redundant arrays of inexpensive Disks (RAID) [C]// Proceedings of ACM SIGMOD. New York: ACM, 1988: 109 - 116.
- [2] RUEMLER C, WILKES J. An introduction to disk drive modeling [J]. IEEE Computer, 1994, 27(3): 17 - 29.
- [3] 王芳, 张江陵, 冯丹. RAID 的并行 I/O 调度算法分析[J]. 计算机工程与科学, 2003, 25(3): 3 - 4, 30.
- [4] 骆新国, 张江陵. 一种 RAID 评价新方法[J]. 华中理工大学学报, 1994, 25(10): 98 - 100.
- [5] 周可, 张江陵, 冯丹. 带 Cache 的磁盘阵列 I/O 响应时间及吞吐量分析[J]. 微电子学与计算机, 2003, 20(8): 66 - 68.
- [6] 谢长生, 刘艳, 李怀阳, 等. 基于分条单元的 RAID 数据分布优化[J]. 计算机科学, 2006, 33(3): 275 - 278.
- [7] METER R V. Observing the effects of multi-zone disks [C]// Proceedings of the annual conference on USENIX Annual Technical Conference. Berkeley: USENIX Association, 1997: 19 - 30.
- [8] COURTRIGHT II W V, GIBSON G, HOLLAND M, et al. RAIDframe: Rapid Prototyping for Disk Arrays [EB/OL]. [2008 - 06 - 10]. <http://www.pdl.cmu.edu/PDL-FTP/RAID/Sigmetrics96.pdf>.

题,提出了一个计算方法并实现了它。把层次聚类的思想应用到差异化属性约简的计算过程中。先对决策表的条件属性集按定义好的距离进行聚类,得到条件属性集的若干个划分;接着,对这些属性子集进行后处理操作,从而得到若干个有较大差异的属性约简集。最后,为说明算法的效果,利用UCI上的数据集对本文算法进行实验,效果良好。

参考文献:

- [1] HU XIAOHUA, CERCONE N. Learning in relational databases: a rough set approach [J]. Computational Intelligence, 1995, 11(2): 323 - 337.
- [2] 王珏. Rough sets 约简与数据浓缩 [J]. 高技术通讯, 1997, 7(11): 40 - 45.
- [3] 叶东毅. 属性约简算法的一个改进 [J]. 电子学报, 2000, 28(12): 81 - 82.
- [4] 王国胤, 于洪, 杨大春. 基于条件信息熵的决策表约简 [J]. 计算机学报, 2002, 25(7): 759 - 766.
- [5] ZIARKO J. Variable precision rough set model [J]. Journal of Computer and System Sciences, 1993, 46(1): 39 - 59.
- [6] 贺玲, 吴玲达, 蔡益朝. 数据挖掘中的聚类算法综述 [J]. 计算机应用研究, 2007, 24(1): 10 - 13.
- [7] TAN PANG-NING, STEINBACH M, KUMAR V. Introduction to data mining [M]. New Jersey: Addison-Wesley, 2006.
- [8] 汤周文, 叶东毅. 差异化属性约简计算的两个算法 [J]. 计算机科学, 2008, 35(8A): 37 - 40.