维普资讯 http://www.cqvip.com



# 进化存储系统中的物理进化*)

刘 艳 谢长生 李怀阳 赵 振

(华中科技大学计算机学院外存储国家专业实验室 武汉 430074)

**摘 要** 系统整体结构不能很好地适应系统磁盘的动态变化,这是现有存储系统中普遍存在的一个问题。而我们提出的进化存储系统 ERAID(Evolving RAID system)是一个动态存储系统,能够在不停止系统 I/O 服务的前提下,采用 DAA、HDAA、e-HDAA 算法分别实现存储系统中同构磁盘的添加、异构磁盘的添加以及异构磁盘的替换,即实现存储系统的物理进化。仿真实验结果表明:在较小的系统开销下,DAA、HDAA、e-HDAA 算法能逐渐吸收添加或替换到 ERAID 系统的(同构或异构)磁盘,使 ERAID 系统获得优化的存储容量和存储性能。

**关键词** 存储系统,RAID,失效,异构磁盘

## Physical Evolution in ERAID System

LIU Yan XIE Chang-Sheng LI Huai-Yang ZHAO Zhen

(Key Laboratory of Data Storage System, School of Computer Science and Technology, Huazhong University of Science and Technology, Wuhan 430074)

**Abstract** The system architecture cannot adapt to the dynamic change of system disks, which is a general problem in traditional storage system. While ERAID system we proposed is a dynamic storage system. To implement the physical evolution of ERAID system, it applies DAA, HDAA, e-HDAA algorithms to cope with the situations of homogeneous disk adding, heterogeneous disk adding, and heterogeneous disk replacement respectively, without stopping the system I/O service. Simulation results demonstrate that through applying DAA, HDAA, e-HDAA algorithms, ERAID system can assimilate the adding/replacing homogeneous/heterogeneous disks gradually with little extra system overhead, and achieve optimized storage capacity and I/O performance.

**Keywords** Storage system, RAID, Failure, Heterogeneous disk

## 1 简介

数字化信息的爆炸性增长给网络环境下的企业级海量存储系统带来一个重要问题:应用对存储系统的容量和性能的要求日益增加;而磁盘技术发展迅速:磁盘存储容量每年翻倍增长,磁盘寻道和旋转延迟的提高相对较低,为每年 7%,磁盘单位容量的价格以每年 40%的速率在降低。以上两种情况共同作用导致存储系统中的磁盘呈现 3 种形式的动态变化:第一种,系统磁盘性能的提高,即系统中原有的旧磁盘发生故障,而系统原有型号的磁盘往往已退出市场或性价比太低不值得购买,从而用容量大、速度高的新磁盘(为了叙述方便,如无特殊说明,本文用"同构磁盘"表示与系统原有磁盘容量、性能相同的磁盘,用"异构磁盘"表示与系统原有磁盘容量或/和性能不同的磁盘);第二种,系统磁盘数量的增加,即向系统添加(原有型号的)磁盘以满足应用对系统更大存储容量的需求;第三种形式是前两种形式的混合,磁盘数量增加的同时磁盘性能也提高,即向系统添加异构磁盘。然而,尽管现有存储系统,如高性能磁盘阵列(RAID)、附网存储(Network Attached Storage, NAS)、存储区域网(Storage Area Network, SAN)的存储容量和性能都在快速增长,这些存储系统却具有一个普遍存在的问题——系统整体结构不能很好地适应系统磁盘的动态变化。对于第一种形式的磁盘动态变化的情况,以 RAID5 技术为例,一旦某个磁盘出现故障,它会在一个新磁盘上重构故障磁盘的数据,由于磁盘技术的发展,新盘的容量和速度都有可能高于 RAID 中原有的旧盘,然而重构后的 RAID 的整体性能并不会因此而提高;对于第二种形式的磁盘动态变化即磁盘数量的增加,RAID5 似乎可以通过简单地往阵列中增加并行磁盘来处理:一方面,磁盘数目的增加使得阵列在不增加冗余开销的前提下扩展存储容量,另一方面,并行磁盘的增加意味着阵列带宽的提高。然而,"向系统添加磁盘"本身也是个难题,当前采用最多的处理方法是"暂停服务"——首先停止系统存取服务,进行数据备份,添加磁盘后重新装载数据,再继续提供系统服务。这种暂停存储系统服务的方法对于许多依赖于数据可用性或要求 24×7h 服务的系统,如科学计算、电子商务等是不可接受的。

我们认为现有存储系统整体结构不能很好适应系统磁盘动态变化的主要原因在于其物理和逻辑的组织是一种静态的结构,而静态组织模型不能很好地刻画处于不断变化之中的系统。而我们提出的进化存储系统(Evolving RAID system, ERAID)[1]是一个由软硬构件组成的动态存储系统。ERAID 实时监测工作构件(或整体结构)状态,一旦发现工作构件(或整体状态)处于不良状态或有向更优状态迁移的可能,就用备

*)本课题受国家自然科学基金(进化存储系统理论和实现技术研究,项目编号:60273073)和"973"课题(海量数据网络存单户系统的研究,项目编号:2004CD318203)资助。刘 艳 博士研究生,主要研究方向为大规模存储、网络存储系统;谢长生 教授、博士生导师,主要研究方向为计算机体系结构、网络存储系统、采用新原理的超高速度、超高速存储技术。

维普资讯 http://www.cqvip.com

用构件(或新结构)构造出一个具有同样功能但处于更优状态的构件代替之,从而使系统整体性能得到进化。(构件可以是磁盘、阵列、互连设备或某种功能的软件。状态可以是物理状态,也可以是数据分布状态。)ERAID 系统从整体上可以分为两种类型的进化,一是逻辑组织结构进化,即磁盘系统中数据的组织以及整个系统中各种不同属性数据的分布可以随着外在的数据流输入输出的变化而相应改变,以期获得最好的系统性能。二是物理进化,即系统在保证系统内部数据可用性的同时,使系统整体性能随部件性能的提高或部件数量的增加而提升。ERAID 系统的逻辑组织结构进化,可参见相关论文[2,3]。本文工作的重点是 ERAID 系统的物理进化,主要研究系统在不停止服务的前提下实现构件(主要是磁盘)的替换和添加,即 ERAID 系统如何适应 3 种类型的磁盘动态变化来实现系统的物理进化。

本文的内容安排如下:第 2 节介绍与存储系统中同构磁盘的引入,以及与系统对异构磁盘的利用相关的研究工作;第 3 节介绍 ERAID 的系统结构;第 4 节详细描述了 ERAID 如何采用适应系统磁盘动态变化,实现系统物理进化的策略;第 5 节用仿真实验对这些策略进行了验证;最后总结全文。

## 2 相关工作

对于系统如何添加同构磁盘,有的工作致力于研究"非限制性"的数据放置技术,这些技术将数据随机放置,使得添加磁盘以后系统重构时所需要从旧磁盘迁移到新磁盘的数据尽量少,同时使得重构后系统各磁盘的负载能够平衡[4,5]。有的工作的研究重点在于增加采用数据分条策略(限制性数据放置)的存储系统的带宽,这些工作主要是针对多媒体环境[6~8],并且并不能增加系统的存储容量。AutoRAID[9]是个多层次的存储系统:上层为 RAID0,底层为 RAID5,通过把更多的存储空间设置为 RAID0,AutoRAID 能够实现系统容量的在线扩充,其缺陷是只有当新磁盘的添加过程完全结束,系统的带宽才得到提高。

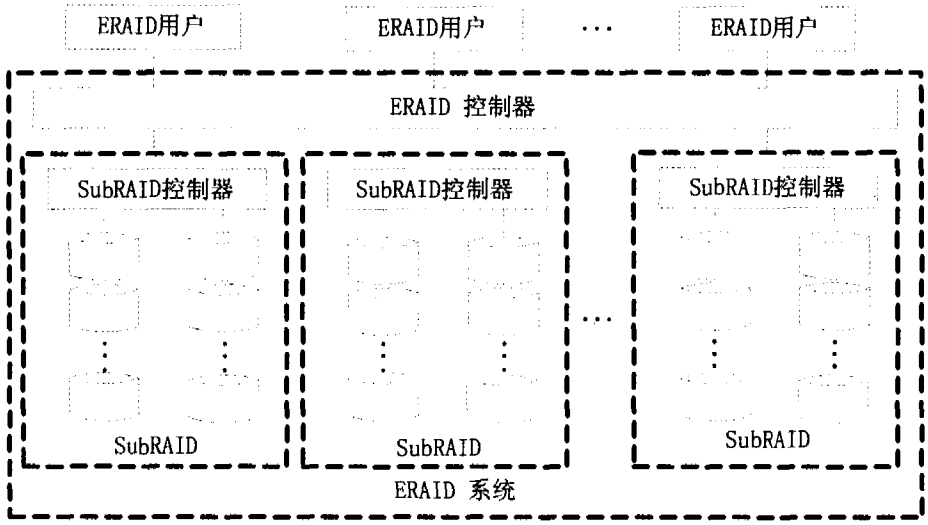


图 1 ERAID 系统结构示意图

为了充分利用系统中异构磁盘的容量和性能,已有研究者提出了一些很好的解决方法,不过这些方法都是针对多媒体系统:Santos 和 Muntz[10]提出一种"带复制的随机分布"策略来提高短期和长期的负载平衡;与此相似,Zimmermann[11]提出由若干个物理磁盘的部分或全部存储空间创建"逻辑磁盘";Dan 和 Sitaran[12]提出利用快磁盘来存放"热"数据,而不重要的数据放置在慢磁盘上。上述针对多媒体的研究都作了一些假定,如请求块大小很大(1Mbyte),读请求比写请求更重要,研究的重点在于系统可以获得一个持续的带宽而不是获得最佳的响应时间。而这些假设在其它环境下(科学计算或通用负载)是不成立的:请求块大小为几千字节,写请求和读请求同样重要,更快的系统响应时间比持续的系统带宽更重要。然而据我们所知,仅有 2 个非多媒体系统对异构磁盘和系统其它磁盘进行了区别对待,它们分别是 HP-AutoRAID[9]和 Linux 系统的 software RAID[13]。AutoRAID 的目的虽然不在于处理磁盘的异构性,可其结构能支持不同型号的磁盘,然而它也只考虑了磁盘容量的不同,并未见其有通过利用不同性能的磁盘来提高整个系统性能的研究。在 Linux software RAID 中,阵列由虚拟磁盘(virtual disk)构成,一个虚拟磁盘由若干个磁盘组成,每个磁盘存放分配到虚拟磁盘的部分数据块;这种"虚拟磁盘"方法的缺陷在于其过于简单,因为它只在能找到一组磁盘,其容量和阵列其他虚拟磁盘的容量相同时才有效(否则将造成磁盘空间的浪费);此外,software RAID 只适用于 RAID0,不适用于 RAID5。还有些研究工作,如 DCD[14],虽然也涉及到异构磁盘,可它们的重点在于提出新的结构,利用不同的磁盘服务不同的任务。然而本文工作的目的不是判断什么是最好、最值得购买的硬件,而是研究当非多媒体系统进行替换异构磁盘或添加(同构或异构)磁盘时,如何能不停止系统服务,并且使重构后的系统获得最佳的存储容量和存储性能。

## 3 ERAID 的系统结构

如图 1 所示,ERAID 由若干个 SubRAID 组成,SubRAID 的数据与校验分布与传统 RAID 相同,而各 SubRAID 的结构可以不同,它们共同组成一个全局的块级存储空间。与传统 RAID 中所有的应用都访问一个统一结构的大阵列不同,对于每个用户,ERAID 只分配一个虚拟卷(Virtual Volume)对其进行服务,虚拟卷是由单个或多个 SubRAID 的部分或全部存储空间构成的逻辑存储空间。用户使用的逻辑地址到磁盘物理地址的转换,即格式为〈虚拟卷号,虚拟卷内偏移量〉的逻辑地址到格式为〈子阵列号,磁盘号,磁盘内偏移量〉的物理地址的转换由 ERAID 驱动器完成,因此,只需要改变分配给用

维普资讯 http://www.cqvip.com

户的虚拟卷所映射的 SubRAID 物理存储空间，就能允许用户的数据随着负载特征的改变在各 SubRAID 间动态迁移，从而在用户看来，系统始终对其提供符合 I/O 需求的虚拟卷。可见，虚拟卷的概念以及数据在 SubRAID 间动态迁移的特征使整个 ERAID 存储系统能对各类应用提供最佳的 I/O 性能。

除上述优势外，ERAID 系统的可扩展性和磁盘的异构性使其能很好地适应磁盘的动态变化，实现系统物理进化——ERAID 系统将用户所见虚拟卷和物理存储空间完全分离的特征使得新磁盘可被加入到 SubRAID 而不会对用户的使用造成影响，SubRAID 也能被系统任意添加、删除或重构，因此 ERAID 系统具有良好的可扩展性：磁盘（和其他资源，如控制器、总线等）能被灵活地添加到存储子系统。另外，SubRAID 的结构可以各不相同，使得异构磁盘（容量、性能不同的磁盘）的性能在系统中能得到充分发挥。ERAID 系统的可扩展性和磁盘的异构性正是本文工作的重点。

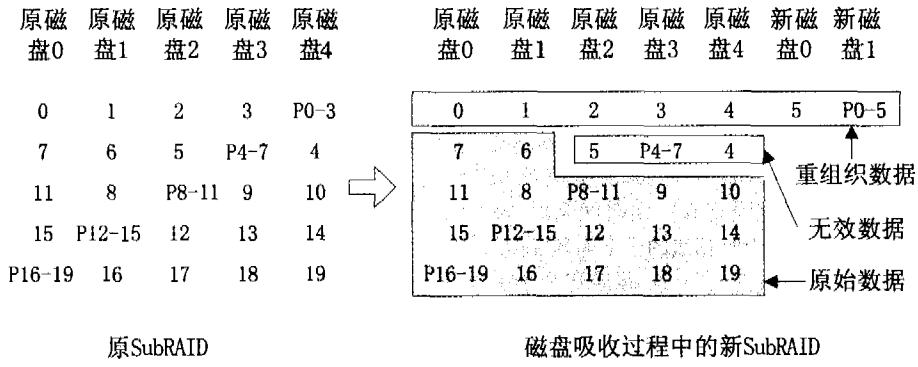


图 2　DAA 算法示意图（级别 level5、磁盘数量为 5 的 SubRAID 采用 DAA 算法添加 2 个同构磁盘的示意图）

图左边为原 SubRAID，右边为已经吸收了一个分条的新 SubRAID。

## 4　ERAID 系统的物理进化

如前所述，系统磁盘动态变化的形式可分为 3 类：同构磁盘的添加、异构磁盘的添加以及异构磁盘的替换，本节就 ERAID 系统如何适应这 3 种类型的磁盘动态变化进行分别介绍。

### 4.1　同构磁盘的添加

ERAID 系统对于同构磁盘的添加，有 2 种处理方式：第一种是当需要增加原有 SubRAID 并行磁盘的数目时，将新磁盘添加到单个 SubRAID，另一种是由新添加的磁盘组成一个或几个新的 SubRAID，添加到 ERAID 系统。后者与新建磁盘阵列的过程基本相同，ERAID 控制器将新 SubRAID 的物理存储空间添加到系统全局存储空间即可，因此本文只对前一种处理方式进行详细描述。

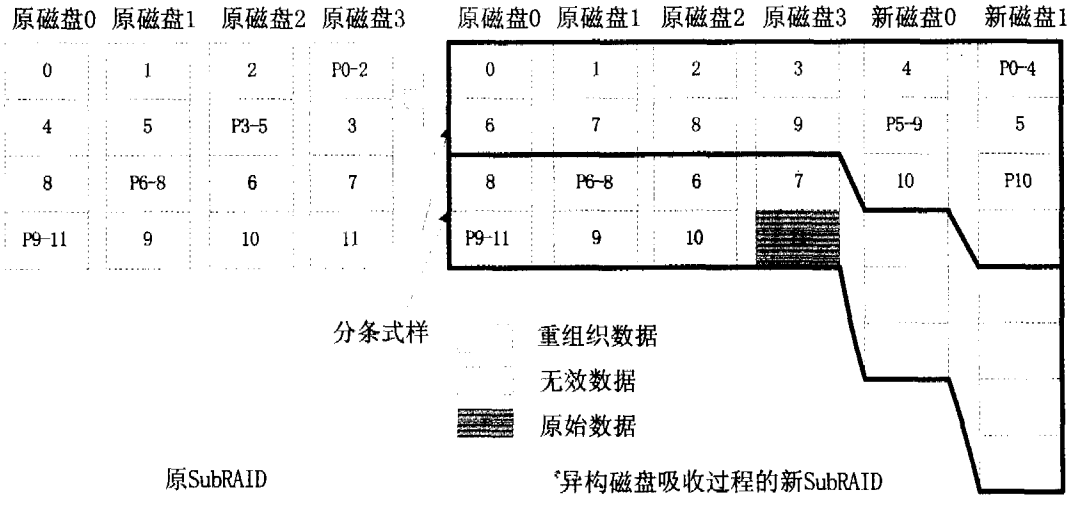


图 3　HDAA 算法示意图（含有 4 个磁盘、级别为 level5 的 SubRAID 采用 HDAA 算法添加 2 个容量分别为原磁盘 1.5 倍和 2 倍的异构磁盘的示意图）

图左边为原 SubRAID，右边为已经吸收了一个分条式样的新 SubRAID。

由于 SubRAID 的结构与传统 RAID 相同，因此如何将新磁盘添加到单个 SubRAID 的问题也即在不停止服务前提下，传统 RAID 如何进行同构磁盘扩展的问题。ERAID 系统采用“磁盘吸收算法（disk assimilation algorithm，DAA）”将原 SubRAID 的数据逐渐进行重新组织，构建成添加磁盘后的新 SubRAID。如图 2 所示的是级别为 level5、磁盘数量为 5 的 SubRAID 添加 2 个新磁盘，已经吸收了一个分条的状态。其它阵列级别、磁盘数量的 SubRAID 添加其它数量新磁盘的情况与此类似。在该算法实现的过程中，SubRAID 的数据分为 3 种：原始数据，即尚未进行重新组织的数据，它只存在于原 SubRAID 的磁盘；重组织数据，即已经由原 SubRAID 重新排布到新 SubRAID 的数据，它是新 SubRAID 中完整的数据分条（stripes），存在于旧磁盘和新添加磁盘；无效数据，即已经重新组织，但仍旧存在于原 SubRAID 磁盘上未被写覆盖的数

维普资讯 http://www.cqvip.com

据。DAA 算法的实现步骤如下：

(1)读取原 SubRAID 中的一定大小的数据，数据的大小可由 ERAID 设置，但是为了避免在新 SubRAID 中引入小写开销，读取数据的大小必须为新 SubRAID 一个分条大小的整数倍。

(2)将读取的数据写到新的 SubRAID。

(3)更新重组织数据对应的 SubRAID 控制器和 ERAID 控制器中的地址映射表和位图(bitmap)信息，并更新无效数据对应的位图信息。

(4)转到步骤(1)，继续读取原 SubRAID 中的固定大小的数据，执行步骤(2)、(3)，直到原 SubRAID 中原始数据都得到处理。

原 SubRAID 吸收新磁盘重建成新 SubRAID 的过程不需要停止 SubRAID 的 I/O 服务，ERAID 系统可以根据该 SubRAID 的负载轻重把整个吸收过程设定在一定时间完成，该时间分为若干时段，每个时段执行吸收算法的一个循环，即步骤(1)、(2)、(3)。如新 SubRAID 有 60 个分条，设定在 1 个小时完成吸收过程，每个循环吸收 10 个分条，则系统每 10 分钟启动一个循环(本例的数字仅用于说明，非实际数字)。这样系统就可以控制磁盘添加过程对系统服务质量造成的影响。当然，DAA 算法也可以在系统空闲时[16]执行。

### 4.2 异构磁盘的添加

与 4.1 节相似，ERAID 系统添加异构磁盘也有 2 种方式，第一种是将异构磁盘添加到系统原有 SubRAID，另一种是由添加的异构磁盘组成新的 SubRAID。在此，我们只叙述前一种方式，后一种方式的处理可以参考 T. Cortes 和 J. Labarta 的研究工作[15]。

T. Cortes 和 J. Labarta 提出了“分条式样”(a pattern of stripes)的概念[16]，以充分利用存储系统中异构磁盘的容量和性能——假设各阵列磁盘的容量按相同的比例缩小，数据再分条存放到“小”阵列，该“小”阵列中的数据分布方式即“分条式样”，在实际阵列中，数据按分条式样的方式重复排布，直至各磁盘空间写满(图 3 中粗线框标示了新 SubRAID 中 2 个分条式样)。本文将“分条式样”的概念从静态的存储系统扩展到动态添加异构磁盘的存储系统，并和 DAA 算法相结合形成“异构磁盘吸收算法(heterogeneous disk assimilation algorithm，HDAA)”，从而将 DAA 算法的使用从系统添加同构磁盘的情况扩展到系统添加异构磁盘的情况。

HDAA 算法的具体步骤如下：

(1)读取原 SubRAID 中的大小为新 SubRAID 一个分条式样整数倍的数据，整数倍数可由 ERAID 设置。

(2)、(3)、(4) 同 DAA 算法中步骤(2)、(3)、(4)。

可见 HDAA 算法和 DAA 算法基本相同，区别之处仅在于 HDAA 算法每次循环吸收的数据量为一个分条式样的整数倍，而后者每次循环吸收的数据大小为新 SubRAID 一个分条大小的整数倍。

图 3 为含有 4 个磁盘、级别为 level5 的 SubRAID 添加 2 个容量分别为原磁盘 1.5 倍和 2 倍的异构磁盘的过程示意图，图左边为原 SubRAID，右边为已经吸收了一个分条式样的新 SubRAID。从图中可看出在 HDAA 算法的执行过程中，SubRAID 的数据也分为原始数据、重组织数据和无效数据三种。其它阵列级别、磁盘数量的 SubRAID 添加其它数量异构磁盘的情况与此类似。

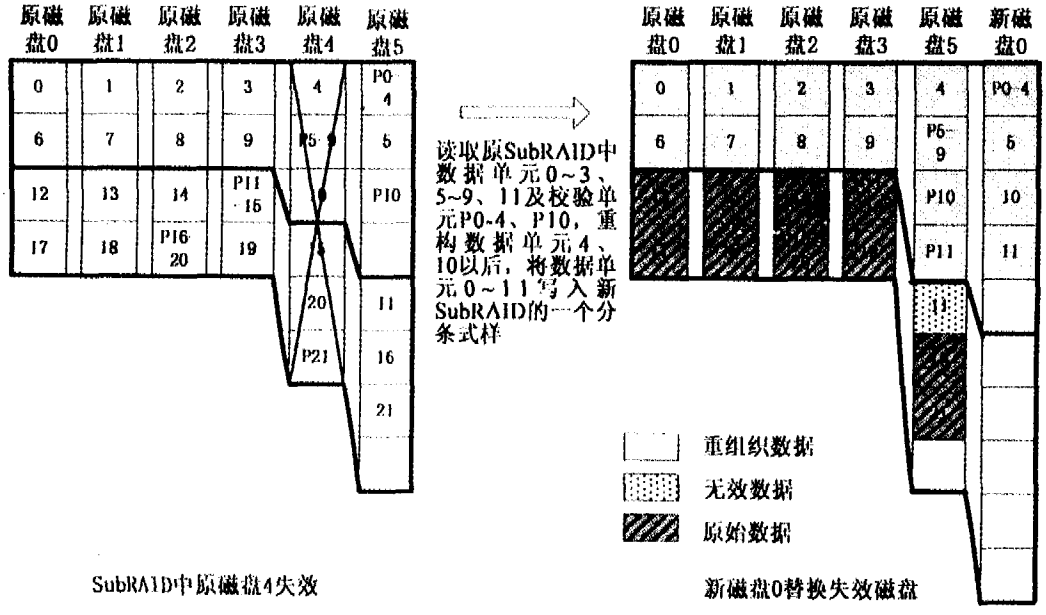


图 4 e-HDAA 算法示意图(阵列级别为 level5 的异构 SubRAID 中原磁盘 4 失效后，用新磁盘 0 替代后，新 SubRAID 采用 e-HDAA 算法吸收替换磁盘的示意图)

图左边为原磁盘 4 失效的 SubRAID，图右边为吸收了一个分条式样的新 SubRAID。

与传统存储系统添加异构磁盘时把其视作同构磁盘，从而造成异构磁盘容量和性能的浪费不同，HDAA 算法中“分条式样”的概念考虑到新加入的异构磁盘的容量和性能，从而使新存储系统的容量和性能得到优化；另外，如上小节所述，整个磁盘吸收过程可以由 ERAID 系统设定在一定时间完成，每个时间段启动执行若干个 HDAA 循环(步骤(1)、(2)、(3))，或在系统空闲时执行，因此 SubRAID 能在不停止 I/O 服务的前提下实现异构磁盘的添加。

### 4.3 异构磁盘的替换

当 ERAID 系统中某个 SubRAID 的磁盘发生失效，而用一个异构磁盘替换失效磁盘时，系统必须在重构失效磁盘数据的同时吸收异构磁盘的存储空间；此时，ERAID 系统采用

维普资讯 http://www.cqvip.com

“扩展异构磁盘吸收算法”( extend HDAA, e-HDAA),该算法在 HDAA 的基础上添加了重构失效数据的过程。e-HDAA 的算法步骤如下:

(1)从原 SubRAID 工作磁盘中读取大小为新 SubRAID 一个分条式样整数倍的数据(整数倍数可由 ERAID 设置),对于其中的失效数据,读取恢复该数据所需要的全部分条单元(stripe unit)及校验单元(parity unit)。

(2)重构失效数据。

(3)、(4)、(5) 同 HDAA 算法中步骤(2)、(3)、(4)。

图 4 为阵列级别为 level5 的异构 SubRAID 中原磁盘 4 失效后,用新磁盘 0 替代后,新 SubRAID 采用 e-HDAA 算法吸收了一个分条式样的示意图——系统首先读取原 SubRAID 中数据单元 0~3、5~9、11 及校验单元 P0-4、P10,重构数据单元 4、10 以后,将数据单元 0~11 写入新 SubRAID 的一个分条式样。

由于在降级模式(degraded mode)工作下的 SubRAID 中各磁盘的负载加倍[],因此如果发生磁盘失效的 SubRAID 负载较大,系统应尽快使该 SubRAID 恢复到正常工作模式,此时,ERAID 系统可以先将失效磁盘的数据重构到替换的异构磁盘(与传统 RAID 重构过程基本相同),再分时段或在系统空闲时采用 HDAA 算法吸收异构磁盘的存储空间。限于篇幅,对于此种情况本文不作详细叙述。

表 1 仿真 trace 的各负载特征参数

| Trace 组成负载 | 总请求个数 | 读/写比率 | 平均请求大小(kB) | 请求大小分布(kB) |
|---|---|---|---|---|
| 负载 0 | 2592000 | 0.43/0.57 | 8.90 | (0.0,40.0) |
| 负载 1 | 2592000 | 0.30/0.70 | 14.31 | (0.0,96.0) |

表 2 磁盘性能参数

| 磁盘型号 | 存储容量(GB) | 转速(rpm) | 数据传输率(MB/s) | 平均寻道时间(ms) |
|---|---|---|---|---|
| 磁盘 a | 73.4 | 10000 | 31.0 | 4.7 |
| 磁盘 b | 110.1 | 10000 | 31.0 | 4.7 |
| 磁盘 c | 146.8 | 15000 | 38.9 | 3.5 |
| 磁盘 d | 183.5 | 10000 | 31.0 | 4.7 |

## 5 仿真实验和结果分析

### 5.1 仿真实验

我们采用 trace 驱动的仿真实验来验证 ERAID 系统适应磁盘动态变化,实现存储系统物理进化。本文的仿真工具为 HRaid[17],它是一个功能强大的存储系统仿真软件。

我们用 HRaid 仿真了一个由 9 个 Seagate Cheetah73[18] 磁盘(标记为磁盘 a)组成 ERAID 系统(我们称之为原 ERAID 系统),该磁盘容量为 73.4GB,转速为 10000r/m,数据传输率为 31MB/s。原 ERAID 系统的控制器具有大小为 256MB,数据传输率为 62MB/s 的非缺失性缓存(NVRAM),总线连接 2 个 SubRAID 控制器,分别管理一个 SubRAID :第一个 SubRAID(SubRAID0)由 5 个磁盘组成,阵列级别为 level5,分条单元大小为 16 kB;第二个 SubRAID(SubRAID1)由 4 个磁盘组成,阵列级别为 level5,分条单元大小为 8 kB。

不失一般性,实验采用了 3 种 HRaid 仿真的异构磁盘(分别标记为磁盘 b、磁盘 c 和磁盘 d),其磁盘容量分别为磁盘 a 容量的 1.5 倍、2 倍和 2.5 倍,4 种实验磁盘的性能参数详见表 2。

本文采用的 trace 为 HRaid 内部生成的 2 种不同特征的负载——负载 0 和负载 1,分别模拟访问 SubRAID0 和 SubRAID1 的 2 个 ERAID 系统用户。负载的特征参数如表 1 所示,其中各负载的请求数均为 2592000(仿真系统运行时间为 24h,平均 30 个请求/s),读/写比率为读写访问次数之比,平均大小分布均服从指数分布,其中(x,y)分别为指数函数的基值(base)和平均值(mean),各请求的到达间隔时间也服从指数分布(0.0,10.0ms),各负载的请求初始地址在对应 SubRAID 上均匀分布。

本文仿真了 3 组实验,分别对应 3 种形式的系统磁盘动态变化(仿真实验用到的 4 种类型的磁盘性能参数见表 2)——第一组实验为向原 ERAID 系统 SubRAID0 添加 2 个同构磁盘 a,ERAID 系统设定用 12h,采用 DAA 算法对同构磁盘进行吸收;第二组实验为向原 ERAID 系统 SubRAID1 添加 2 个异构磁盘:磁盘 b 和磁盘 c,系统设定利用 12h,采用 HDAA 算法对异构磁盘进行吸收;接下来,第三组实验仿真在第二组实验后添加了异构磁盘 b、c 后的 SubRAID1 中磁盘 b 失效,用磁盘 d 替代,系统用 12h,采用 e-HDAA 算法吸收替换磁盘 d,重构 SubRAID 的情况。

### 5.2 实验结果及分析

仿真实验结果如图 5a、5b 和 5c 所示。图 5a 记录第一组仿真实验的结果,即在实验 trace 下第一组实验中 3 个存储系统的平均请求响应时间,它们分别为:原 ERAID 系统、SubRAID0 完全吸收了 2 个同构磁盘后的 ERAID 系统,以及从 SubRAID0 添加到完全吸收同构磁盘过程中的 ERAID 系统(吸收过程为 12h)。与图 5a 相似,图 5b 和图 5c 分别记录了第二组和第三组仿真实验的结果。

从图 5a 中我们看到原 ERAID 系统的平均请求响应时间高于 SubRAID0 完全吸收了 2 个同构磁盘后的 ERAID 系统。此外,我们注意到:在 SubRAID0 刚刚添加磁盘时,ERAID 系统的性能与原 ERAID 系统相同,但随着磁盘吸收过程的进行(仿真前 12h),系统性能逐渐趋向完全吸收了 2 个同构磁盘的 ERAID 系统;在磁盘吸收过程中无显著的系统开销,因为整个吸收过程被分散到 12h,并且原 SubRAID 上的分条一旦被吸收重构到新 SubRAID,该重构的 SubRAID 分条则分布在更多的磁盘上,对该分条数据的访问具有更高的并行性,从而具有更短的请求响应时间。

如图 5(b)所示,ERAID 系统中往 SubRAID1 添加 2 个异构磁盘过程所呈现的规律与图 5(a)基本相似,而异构磁盘吸收过程的开销比前者略大,这是由于:异构磁盘组成的 SubRAID 中的分条花样由若干个不同分条深度(stripe depth)的分条组成,有的分条深度较小(由部分 SubRAID 磁盘组成),导致数据吸收时访问重构 SubRAID 的写请求的磁盘并行性降低。

从图 5(c)中可看到,与图 5(a)、(b)中原 ERAID 系统相比,SubRAID1 中磁盘 b 失效后,ERAID 系统的性能大大降低——由于对 SubRAID1 中磁盘 b 的访问将引起对 SubRAID1 所有工作磁盘的访问,以重构磁盘 b 所存放的数据,这使得 SubRAID1 工作磁盘的负载大大增加。当用异构磁盘 d 替换失效磁盘 b 以后,系统在重构磁盘 b 丢失数据的同时,逐渐吸收新 SubRAID1 的分条花样。从图 5(c)可以观察到:磁盘 b 失效后的 ERAID 系统性能与异构磁盘 d 替换吸收过

维普资讯 http://www.cqvip.com

程中的 ERAID 系统性能并无显著差异，这说明异构磁盘的替换吸收不会给存储系统引入大的额外开销；而随着新 SubRAID1 分条花样吸收过程的进行，更多已经重构的数据可以直接被访问，SubRAID1 的负载逐渐降低至磁盘 b 失效前的大小，存储系统的性能也逐渐趋近异构磁盘 d 完全替换了失效磁盘 b(异构磁盘完全吸收)的系统性能。

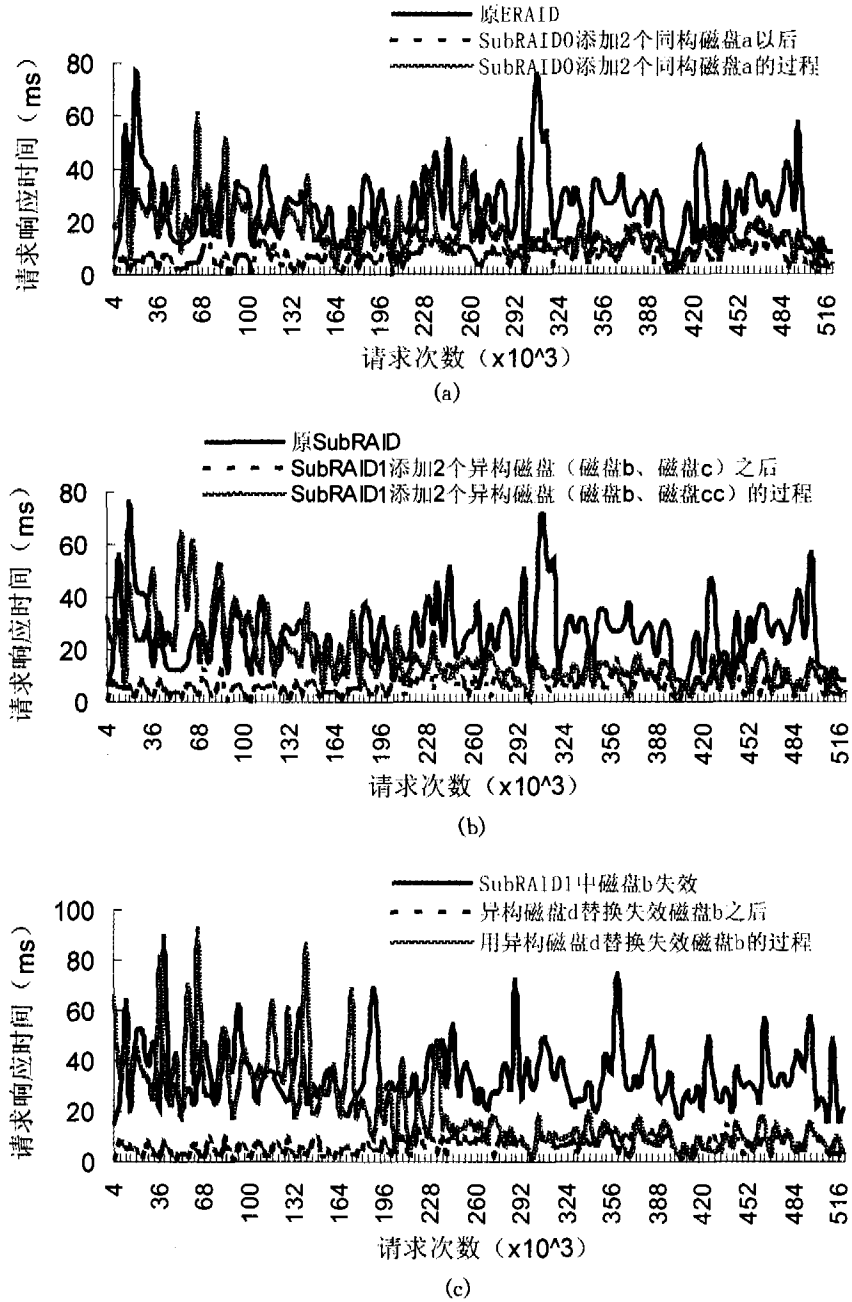


图 5　三组仿真实验结果

(a)原 ERAID 系统、SubRAID0 采用 DAA 算法添加 2 个同构磁盘(磁盘 a)以后的 ERAID 系统，以及 SubRAID0 添加同构磁盘过程中(12h)的 ERAID 系统的请求响应时间，(b)原 ERAID 系统、SubRAID1 采用 HDAA 算法添加 2 个异构磁盘(磁盘 b 和磁盘 c)以后的 ERAID 系统，以及 SubRAID1 添加异构磁盘过程中(12h)的 ERAID 系统的请求响应时间，(c)SubRAID1 中磁盘 b 失效以后的 ERAID系统、SubRAID1 中失效磁盘b采用e-HDAA算法用异构磁盘d替换后的ERAID系统，以及SubRAID1中失效磁盘b用异构磁盘 d 替换过程中(12h)的 ERAID 系统的请求响应时间。

**结论**　本文描述了进化存储系统(ERAID)通过适应 3 种类型的磁盘动态变化来实现存储系统的物理进化——在不停止系统服务的前提下，ERAID 系统采用 DAA、HDAA、e-HDAA 算法分别实现存储系统中同构磁盘的添加、异构磁盘的添加以及异构磁盘的替换。仿真实验表明：采用 DAA、HDAA、e-HDAA 算法添加(同构或异构)磁盘或替换异构磁盘，即进行系统物理进化，不会给运行的存储系统带来太大的开销；此外，实现物理进化后的 ERAID 系统能充分利用添加或替换后的磁盘，获得优化的存储容量和存储性能。

## 参 考 文 献

1　董晓明，谢长生. 基于对象的进化存储系统研究. 计算机科学，2005 (12)

2　Liu Yan, Xie Chang-Sheng, Li Huai-Yang. PMSH: A new algo-

(下转第 277 页)

维普资讯 http://www.cqvip.com

程中。

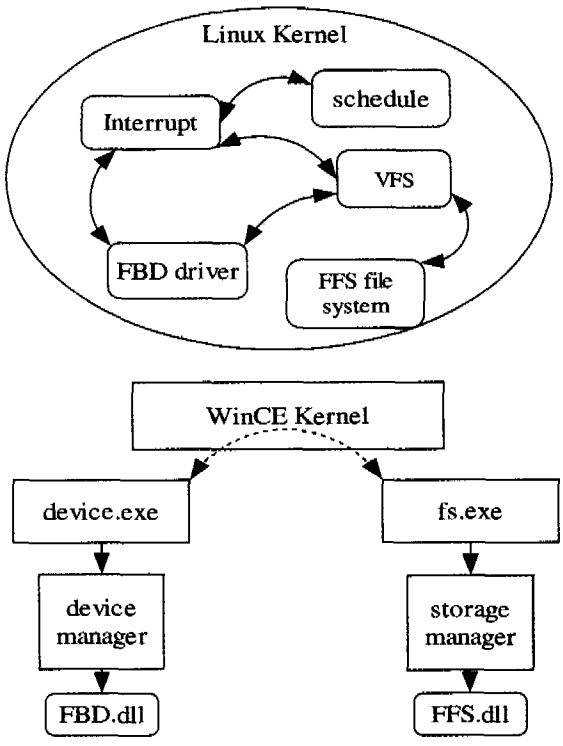


图 7 LINUX 和 WinCE 的结构

(2)模块间通讯

LINUX 各个模块是链接在一起,所以之间的通讯是通过直接调用来实现,如图 7。当然,内核的结构已经比较完善而且清晰,VFS 已经规定了 FBD 和 FFS 的函数接口。FFS 想要执行 FBD 里面的功能的时候,VFS 也提供了对标准块设备的专有操作函数,这些函数将屏蔽具体块设备的细节。

因为 WinCE 下各个模块都属于不同的进程,他们之间的通讯则是依靠进程间通讯实现的。同样,设备管理器(device manager)规定了块设备的 dll 的标准接口,存储管理器(Storage Manager)也给出了 FSDMGR ReadDisk 系列的用于磁盘操作的函数,供 FFS 与 FBD 之间通讯。

另外,模块间的通讯,特别涉及到文件读写的时候,肯定会有内存的传递,在 LINUX 下,传递的内存地址是可以直接操作的,而 WinCE 下,由于内存地址存在于各自的进程访问空间中并且受到保护,所以会不允许其他进程进行操作。WinCE 提供了 MapPtrToProcess 等系列的机制,用于将其他进程的内存地址映射到自己进程中。

(3)网络操作

网络操作的标准函数一般是应用程序来调用的,对于 LINUX,由于驱动程序存在于内核中,不能调用应用层的接口。LINUX 提供了专门用于 kernel 内进行网络操作的函数(sock_create_kern、sock_recvmsg 等)。

对于 WinCE 则没有这个限制,因为 WinCE 下的驱动程序是运行在用户空间的。可以直接像应用程序一样,使用 WinSocket 进行网络通讯。

(4)激活

实现了 FFS 和 FBD 之后,LINUX 的 mount 命令可以通过参数中在 FBD 上挂载 FFS,并且加载成指定的目录。可以通过命令行或者程序执行 mount 来进行激活;WinCE 则是在注册表中设置 FBD 的属性,指定加载的文件系统和加载后的目录。可以在启动时自动激活或者程序通过系统调用来要求激活 FBD。

**结束语** 本文给出的方案可以在当前软硬件平台差异都很大的情况下,实现平台间的数据共享。这个方案通用性强,既能跨各种平台,方便的实现,又能很好的实现与现有软件的兼容,实用价值相当高。本文给出的是一个通用的方案,具体实际应用中,可以在安全、性能等方面进行灵活的扩展,以满足不用平台、不同特性的场合下的需要。

## 参 考 文 献


1 勒伏(美). LINUX 内核设计与实现[M]. 北京:机械工业出版社, 2004
2 Microsoft Windows CE . Net 4. 2 Help. MicroSoft
3 周毓林,宁杨,陆贵强,等. WINDOWS CE. NET 内核定制及应用开发. 北京:电子工业出版社, 2005
4 Corbet J, Rubini A, Kroah-Hartman G. LINUX 应用程序开发第 2 版. 北京:电子工业出版社, 2005


(上接第 248 页)


rithm for raid data migration based on stripe unit heat. In: International Conference on Machine Learning and Cybernetics (ICMLC'2005). 2005. 3421~3427
3 Li Huai-Yang, Xie Chang-Sheng, Liu Yan, et al. The approximate parameters analysis of mS RAID. In:2005 International Conference on Machine Learning and Cybernetics (ICMLC'2005): 2005. 2753~2759
4 Scheideler C, Brinkmann A, Salzwedel K. Efficient, distributed data placement strategies for storage area networks. In: Proceedings of the 12th annual ACM symposium on Parallel algorithms and architectures
5 Ethan L, Miller R, Honicky J. A fast algorithm for online placement and reorganization of replicated data. In: IPDPS 2003 17th International Parallel and Distributed Symposium, 2003
6 Cidi S, Yao R, Zimmermann A, et al. An efficient randomized technique to reorganize continuous media blocks. In: IEEE 18th International Conference on Data Engineering (ICDE'02), 2002. 473~482
7 Ghande H, Shahabi C. Management of physical replicas in parallel multimedia information systems. In: Proceedings of the Foundations of Data Organization and Algorithm (FODO) conference, 1993. 51~58
8 Baird R, Tobagi F, Pang J, et al. Streaming raid: A disk array management system for vedio files. In First ACM Conference on Multimedia, 1993. 393~400
9 Wilkes J, Golding R, Staelin C. The HP AutoRAID hierarchical Storage System. In:Proceedings of the 15th ACM Symposium on Operating Systems Principles, Copper Mountain, 1995. 96~108
10 Santos J R, Muntz R. Performance and analysis of the rio multimedis storage system with heterogeneous disk configurations. In: ACM Multimedia, Bristol, UK, 1998. 303~308
11 Zimmermann R. Continuous media placement and scheduling in heterogeneous disk storage systems: [Ph D Thesis]. University of Southern California, 1998
12 Dan A, Sitaram D. An online video placement policy based on bandwidth to space ratio (bsr). In:Proceedings of the SIGMOD, San Jose, CA, 1995. 376~385
13 Wilkes J. Personal communication. September 1999
14 Hu Y, Yang Q. A new hierarchical disk architecture. IEEE Micro, 1998. 64~75
15 Cortes T, Labarta J. Taking advantage of heterogeneity in disk arrays. Journal of Parallel and Distributed Computing, 2003. 448~464
16 Gonzalez J L, Cortes T. Increasing the capacity of RAID5 by online gradual assimilation. In: The 13th International Conference on Parallel Architectures and Compilation Techniques (PACT), 2004
17 Cortes T, Labarta J. HRaid: a Flexible Storage-system Simulator. In: Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications, CSREA Press, 1999. 772~778
18 Seagate Corporation. Cheetah 73 Family: ST173404LW/LWV/LC/LCV Product Manual, http://www.seagate.com/support/disk/mannuals/scsi/83329478f.pdf