重复数据删除算法在 VTL 系统中的应用研究*

孙虎威,靳嘉伟,张 晶,龚 鸣
(重庆大学 光电技术及系统教育部重点实验室,重庆 400044)

摘 要:为了使 VTL(虚拟磁带库)系统能更有效地利用磁盘空间,存储更多的数据信息,介绍了一种带有重复数据删除算法的虚拟磁带库应用方法。该方法从性能和效率等多方面考虑,首先把磁带按文件级去重,再将文件切分成块,通过 Bloom Filter 和 MD5 算法双重计算,经查找和存储实现数据块级的重复删除。实验测试证明,该方案稳定地实现了数据的去重及加密功能,能有效节省虚拟磁带库的存储空间。

关键词:虚拟磁带库;重复数据删除;Bloom filter;MD5
中图分类号:TP309.3 **文献标识码**:A **文章编号**:1674-7720(2013)06-0082-04

Application and implementation in the VTL system based on data de-duplication algorithm

Sun Huwei, Jin Jiawei, Zhang Jing, Gong Ming
(Key Laboratory of Optoelectronic Technology and Systems of the Education Ministry of China, Chongqing University, Chongqing 400044, China)

Abstract: In order to let the virtual tape library system make good use of disk space, so as to store data more efficiently, this article described an application of VTL with data de-duplication algorithm. Considering the capability and efficiency, firstly, it ensured the tape files are not repeated, then cut file into blocks, calculated by Bloom Filter and MD5 algorithm, seek and stored. Finally, achieve data de-duplication. Tests showed the method achieve stable capabilities of data de-duplication and encryption, saving storage space of VTL system efficiently.

Key words: virtual tape library; data de-duplication; Bloom filter; MD5

进入 21 世纪以来,在科技飞速发展的同时,数据信息的产生也在急剧增长。据悉,企业的数据量平均年度增长率为 50%左右,部分数据的冗余率却在 60%以上。这使得备份时需消耗大量的时间和空间去存储重复的数据,资源浪费十分严重。为了实时存储大量有效的信息,针对物理磁带库存储容量小和效率低等不足,人们引进了虚拟磁带库技术,将高速磁盘阵列仿真成磁带格式,节省了磁带机上带、定位、退带等机械动作时间,同时无需担心机械手故障、磁头耗损或磁带受潮等问题。节省成本的同时提高了备份和恢复速度,实现了实时有效地存储海量数据信息。

尽管虚拟磁带库在应对数据存储时发挥了巨大作用,但是仍不能满足市场需求。如何对存储在虚拟磁带库系统中的数据进行重新压缩从而更有效地利用存储空间,便成为了如今研究的热门课题。而重复数据删除技术作为目前企业热捧的技术之一,在数据压缩处理和存储领域具有很大的应用空间。本文提出了重复数据删除算法在虚拟磁带库系统中的一种应用方案。

1 相关概念和算法介绍

1.1 重复数据删除算法

重复数据删除算法又名智能压缩算法,是一种通过消除冗余重复数据减少存储需求的方法。

重复数据删除算法有多种分类方法。按照重复内容识别方法分类可分为三种:基于内容散列识别、基于内容识别和基于 Hyper-factor 识别;而基于消除冗余执行次序的分类则可以分为在线式消冗和后处理式消冗两种;基于去重粒度分类可分为文件级、数据块级和字节

* 基金项目:重庆市科技攻关重点项目(CSTC2009AB2231);重庆市自然科学基金(CSTC2009BB2195)

级消冗三种[1]。本文在虚拟磁带库系统的应用主要采用基于散列识别方法的数据块级后处理式消冗方案。

1.2 数据分块算法

基于数据块级的分块算法主要有定长切分、CDC 切分和滑动块切分三种[2]。

定长分块算法(Fixed-Size Partition)主要采用预先分配好的块对文件进行切分，并计算弱校验值和 MD5 强校验值。该算法的优点是简单、性能高，但它对数据插入和删除非常敏感，处理十分低效，不能根据内容变化作调整和优化。

CDC(Content-Defined Chunking)算法是一种变长分块算法，它应用数据指纹将文件分割成长度大小不等的分块。CDC 算法对文件内容变化不敏感，插入或删除数据只会影响到较少的数据块，其余数据块则不受影响。该算法也有缺陷，数据块大小的确定比较困难。

滑动块(Sliding Block)算法结合了定长切分和 CDC 切分的优点，数据块大小固定。它对定长数据块先计算弱校验值，如果匹配则再计算 MD5 强校验值，两者都匹配则认为是一个数据块边界。该数据块前面的数据碎片也是不定长的数据块。如果滑动窗口移过一个块大小的距离仍无法匹配，则认定其为一个数据块边界。滑动块算法对插入和删除问题的处理非常高效，并且能够检测到比 CDC 更多的冗余数据，但它容易产生数据碎片。

1.3 哈希查找和存储算法

1.3.1 MD5 算法

MD5 算法即消息摘要算法第 5 版，由 MIT 计算机科学实验室和 RSA 数码保安公司联合开发，经 MD2、MD3 和 MD4 延伸而来[3]。它将文件的任意一段内容通过一系列算法压缩成一段 128 bit 的信息摘要(哈希值)。其本质即为一种哈希函数，具有单向性、抗弱碰撞性和抗强碰撞性等特点。

在 MD5 算法操作中，先对元数据信息进行填充，使得其字节长度对 512 求余结果为 448；接着填充 64 bit 数据段长度信息，凑齐为 512 的整数倍；然后用 4 个固定的链接变量作为参数对 MD 缓冲器进行初始化；最后用 4 种不同的非线性函数进行轮换计算，结果输出 4 个 32 bit 即 128 bit 的哈希值[4-5]。算法过程如图 1 所示。

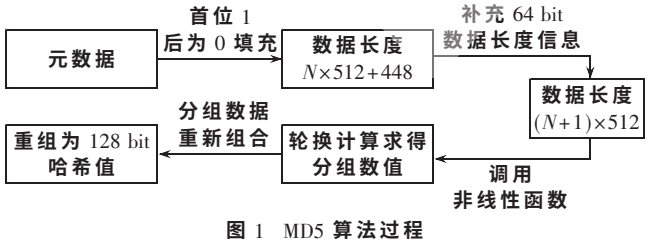

图 1 MD5 算法过程

1.3.2 Bloom Filter 算法

Bloom Filter 由 Howard Bloom 在 1970 年提出。它利用位数组很简洁地表示一个集合，并能通过一组哈希映射函数判断一个元素是否属于这个集合。该算法具有很好的空间效率和时间效率，但是却有一定的误识别率(假阳性误判)，并且删除操作比较困难。

该算法主要包括数据元素的查找和插入两部分。在查找操作中，首先将目标信息存储到一个集合 S 中，接着设计多个相互独立的哈希函数及适度大小的哈希表，并设其初始值全为 0。在集合 S 中任取一个元素，经哈希函数分别映射到哈希表中。如果所对应哈希表位置的值都为 1，则说明该元素可能已经存在，但也有误判的可能。若有任意其中一个位置不为 1，则说明该元素必不存在。同样插入操作经哈希函数计算并映射后，把相应位置的值都置为 1。

2 方案设计及实现

2.1 应用场景

图 2 所示为常见的一种应用虚拟磁带库进行数据备份的场景。各个客户端所产生的数据通过网络传送到服务器端，在服务器中备份软件的操作下，将数据备份到虚拟磁带库所模拟成磁带格式的磁盘阵列中，该磁盘阵列由相应的 RAID 组构成，从而进行容灾保护。该数据可以实时导入、导出到相应的物理磁带库中。同样，数据流的逆向即可实现数据恢复作业。在虚拟磁带库系统中可以对所备份的数据进行重新扫描和重复数据删除，并存储压缩后的数据，选择是否删除原有数据，进而节省大量的磁盘空间。

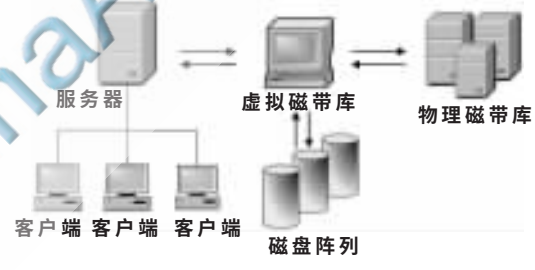

图 2 常见备份数据流

2.2 系统结构设计

带有重复数据删除功能的虚拟磁带库系统结构设计如图 3 所示。上层为包含有支持 NFS/CIFS、OST 及 VTL 等文件协议的文件协议读取层，该层将存储子系统进行网络化，实现存储内容的高速共享访问。下一层为文件管理层，该层主要实现对数据存放文件及命名空间的管理和设置。文件管理层下面为重复数据删除模块，

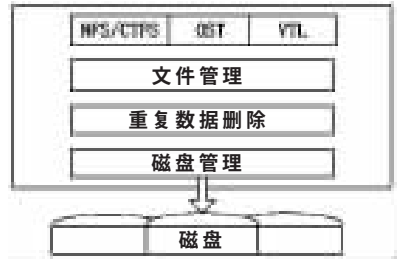

图 3 系统结构设计图

主要对搜寻到的数据文件进行分块处理、哈希计算和查找并归类存储等操作。下一层为磁盘管理模块，主要负责对磁盘阵列数据元数据和哈希值的分类存放和获取，以及磁盘访问顺序的优化处理等。

2.3 重复数据删除功能详细设计

为实现文件中重复数据的删除功能，本文进行了如图4所示的详细设计。首先该模块对虚拟磁带库中需处理的磁带文件进行查找和获取，然后计算出相应的哈希值，先用 Bloom Filter 算法进行快速计算和查找，如果位数组 A 中已存在相关的文件，则再次进行 MD5 算法计算和查找，如果位数组 A 中的确存在该文件，则只存储该文件相关哈希值，接着进行下个文件的处理。如果在 Bloom Filter 算法的位数组 A 中不存在该数据的信息，则进行添加和更新，接着完成对该文件哈希值的存储，然后对该文件进行数据块级的处理。由于在 Bloom Filter 中可能出现误判，故而当 MD5 再次校验不存在时，同样也会进入数据块级处理中。

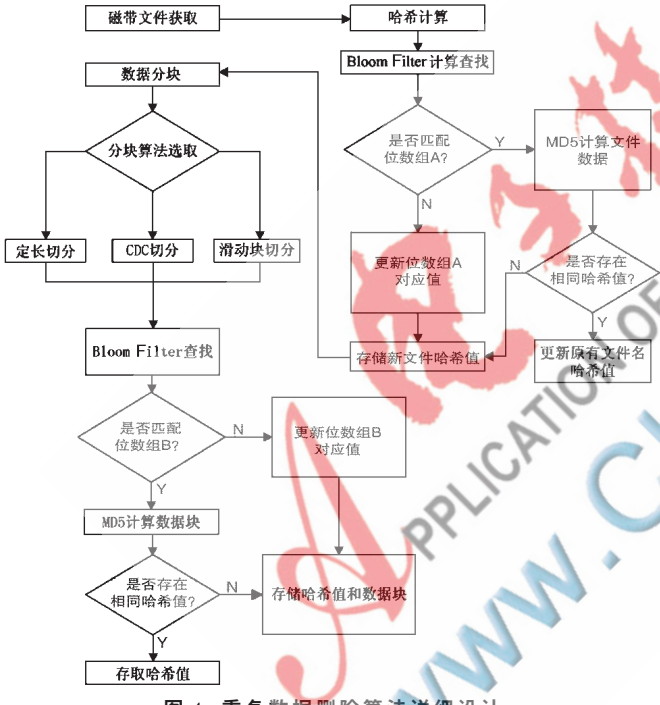

图4 重复数据删除算法详细设计

本文应用可以根据需要选择定长、CDC、滑动块任意一种切分方式来进行数据块划分。接着对所切分的数据块进行如同文件级别的 Bloom Filter 和 MD5 双重验证。首先对数据块进行 Bloom Filter 计算，当结果不匹配位数组 B 中相关位时，则表明该数据块必不存在，对位数组中相关位进行插入和更新，并分别存储该数据块和相关的哈希值；如果该数据块匹配该位数组 B 时，则再次进行 MD5 计算和校验。如果仍然匹配，则说明该数据块重复，只存储该数据块的哈希值；如果出现不匹配情况，则说明前面计算出现误判，分别存储该数据块和相应的哈希值。

数据块及相应哈希值存储及检索如图5所示。当文件 A 进入计算时，会生成相应哈希值并指向对应数据块。当首次查找数据块 N 不存在时，则先存入数据块，然后再把数据块 N 的索引指向该数据块所在位置，当再次查找时，仅存储对应哈希值。文件 A 检索完毕后同样对文件 B 进行相关操作。而当 A'经计算与文件 A 内容相同时，则文件 A'的索引会指向文件 A 的索引，当文件 A'数据恢复时，通过指引直接检索调用文件 A 中的索引值，从而进一步加快效率，节省存储空间。

图5 磁盘中数据块及索引存储

2.4 关键参数分析

2.4.1 Bloom Filter 误判率

假设在 Bloom Filter 算法中，有 k 个相互独立的 Hash 函数，待处理的元素数为 n，位数组位数为 m，此时需满足 $kn<m$。当所有元素都被 k 个 Hash 函数映射到位数组中，可以求得误判率 f 为：

$$f=\left[1-\left[1-\frac{1}{m}\right]^{kn}\right]^{k}\approx(1-e^{-kn/m})^{k}$$

容易看出，当位数组 m 增加时，误判率会下降。经计算可得当 $k=(\ln 2)\frac{m}{n}$ 时，误判率取得最小值，即：

$$f=\left[\frac{1}{2}\right]^{k}\approx(0.618)^{\frac{m}{n}}$$

当误判率为 e 时，若 $f\leqslant e$ 成立，则需

$$m\geqslant n\times\log_2(1/e)$$

若使 $f\leqslant 0.01$，则需 $m\geqslant 9.567n$，此时取 $k=7$[6]。表1中所示数据可获得不同 k 值和 m/n 下对应的误判率的

表1 k 值和 m/n 值对误判率的影响

m/n	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	最佳 k
2	0.393	0.400				1.39
3	0.283	0.237	0.253			2.08
4	0.221	0.155	0.147	0.160		2.77
5	0.181	0.109	0.092	0.092	0.101	3.46
6	0.154	0.080 4	0.060 9	0.056 1	0.057 8	4.16
7	0.133	0.061 8	0.042 3	0.035 9	0.034 7	4.85
8	0.118	0.048 9	0.030 6	0.024	0.021 7	5.55

大小以及 m/n 固定时取得最小误判率的最佳 k 值。

假设本文处理的数据为 16 TB，平均分块大小为 8 KB，则数据块的数量大概为 2×10^9 个，位数组 B 占用内存空间大概为：$93\ 567\times2\times10^9\div8\approx2.4$ GB。这对于虚拟磁带库系统是完全可以实现的。

2.4.2 重删率

重复数据删除算法的效果可用重删率 DDR(Data De-duplication Ratio)来表示。即为元数据在重复数据删除之前的字节数与处理之后的字节数之比：

$$DDR=\frac{initalbytes}{finalbytes}$$

重删率同样依赖于数据集自身的特征、数据划分策略以及平均数据分块大小等因素。故而有专家对上述公式进行了修正：

$$DDR'=\frac{DDR}{1+\frac{元数据大小}{平均分块大小}}$$

由上式可以看出，修正后的重删率在平均分块大小较小时与元数据大小成正比，而当平均分块大小较大时，与元数据大小成反比。因此设定合适的数据分块策略和分块大小界限很关键。

3 实验结果和分析

本文在以下实验环境进行了测试：CPU 为 Intel core 2 双核处理器，2 GB 内存，磁盘空间为 200 GB；Win7 系统下 VMware 虚拟运行 Red Hat 5 Linux 系统；虚拟磁带库系统应用软件选用开源软件 MHVTL0.18。实验根据以上设定的规格分别对不同的数据进行备份，然后进行重删处理，其结果如表 2 所示。

表 2 测试结果分析

文件	内容	文件数	大小/MB	重复块	删后值/MB	DDR
1	TXT	464	30.7	1 201	15.2	2.037 9
2	照片	47	63.4	2	62.2	1.023 8
3	PDF	114	83.9	995	79.9	1.050 8
4	音频	31	162.7	1	165.3	0.994 2
5	视频	1	276.6	0	279	0.991 3

实验中采用分块大小为 4 KB，共对 5 组大小及内容不同的文件进行了数据的重复删除处理。由表 2 可知，文件 1 中 TXT 文件和文件 3 中 PDF 文件存在相当数量的重复块；而照片、音频和视频等文件存在较少重复数据块。由于测试环境限制，本次测试的子文件都不相同，且数据量小，所以重删率较低，甚至出现小于 1 的情况。不过数据经还原处理后，与原始数据相比完全相同，安全性能有保障，当出现大量重复文件时，效果更好。

本文主要介绍了一种重复数据删除算法在虚拟磁带库系统中的应用方法。该应用采用后处理式的数据分块哈希计算方法来进行数据的重复删除。数据分块可选择使用任一种常用的 3 种分块方法，数据查找和存储采用 Bloom Filter 和 MD5 算法双重计算，经过设置参数有效地降低了 Bloom Filter 的误判率和 MD5 算法的碰撞率。有效提高了存储的时间效率和空间效率，并获得良好的重删率，同时完成了数据的压缩和加密双重功能。

参考文献

[1] 付印芳，肖侬，刘芳.重复数据删除关键技术研究进展[J].计算机研究与发展，2012，49(1)：12-20.

[2] 敖莉，舒继武，李明强.重复数据删除技术[J].软件学报，2010，21(5)：916-929.

[3] RIVEST R.The MD5 message digest algorithm[M].RFC 1321，1992.

[4] 陈少晖，翟晓宁，阎娜，等.MD5 算法破译过程解析[J].计算机工程与应用，2010，46(19)：109-112.

[5] 张裔智，赵毅，汤小斌.MD5 算法研究[J].计算机科学，2008，35(7)：295-297.

[6] HOROWITZ E，SAHNI S，MEHTA D.Fundamentals of data structures in C++[M].Computer Science Press，1995.

(收稿日期：2012-11-20)

作者简介：

孙虎威，男，1986 年生，在读硕士研究生，主要研究方向：嵌入式系统与虚拟网络仪器技术。

靳嘉伟，男，1983 年生，在读硕士研究生，主要研究方向：测试与控制技术。