◎研发、设计、测试◎

实现实时 FAT 文件系统的一种简单方法

刘可嘉[1],梁阿磊[2]
LIU Ke-jia[1],LIANG A-lei[2]

上海交通大学 软件学院,上海 200240
School of Software,Shanghai Jiao Tong University,Shanghai 200240,China
E-mail:donaldliew@gmail.com

LIU Ke-jia,LIANG A-lei.Simple approach to implement real time FAT file system.Computer Engineering and Applications,2008,44(16):70-72.

Abstract: Most embedded smart devices in the market now use the FAT file system because of its compatibility.However,the FAT was not well designed to meet real time requirements.This article proposes a simple solution that greatly improves the response time of the FAT file system,without hurting its compatibility.

Key words: FAT;file system;real time;compressed FAT;free-cluster index

摘 要:出于兼容性等方面的考虑,目前市面上常见的嵌入式智能设备多采用 FAT 文件系统。然而 FAT 文件系统由于设计上的缺陷导致实时性比较差。针对这个问题提出了一套简单易行的方案,在不影响兼容性的前提下大幅度改善了 FAT 文件系统的响应时间(实时性)。

关键词:FAT;文件系统;实时性;压缩 FAT 表;空闲簇索引
DOI:10.3778/j.issn.1002-8331.2008.16.021 **文章编号**:1002-8331(2008)16-0070-03 **文献标识码**:A **中图分类号**:TP316

1 引言

嵌入式系统近年来的发展有目共睹。从传统的工业控制到新兴的数字娱乐,再到医疗保健乃至军事国防,嵌入式系统已经融入人们生产生活的方方面面。在给人们的工作和生活带来巨大便利的同时,嵌入式系统产业还为我国创造了可观的经济价值。根据信息产业部 2004 年的数字,嵌入式软件收入占整个中国软件产业收入的 31.1%,达到了 748.8 亿元。

传统的嵌入式系统结构和功能比较简单,需要存储的数据很有限,因此一般没有完备的文件系统。但随着需求的增长和技术的发展, 智能化成了嵌入式系统的一个主要发展趋势,为了存储和处理更复杂的数据,智能设备中一般都需要实现一套完整的文件系统。相对于一般的计算机系统,嵌入式系统有资源受限、工作环境复杂多样等特点。因此,传统的文件系统并不一定适用于嵌入式系统。特别的,一个适用于嵌入式系统的文件系统应该具有如下三个特性:

(1)兼容性。嵌入式系统特别是便携智能设备作为 PC 机的补充,经常需要与 PC 机交换数据,如果它们的文件系统互相兼容,数据交换就变得非常简单。

(2)实时性。出于节省成本和能源等方面的考虑,嵌入式系统的内存和计算资源通常比较受限,但很多嵌入式应用往往具有一定实时性的要求。例如在进行视频录制等高速率数据采集作业时,文件系统必须保证能够在确定时间内将采集到的数据写入文件,否则容易造成部分数据丢失。

(3)可靠性。嵌入式系统通常采用电池供电,且工作环境复杂多样,很难保证物理上的可靠性,因此嵌入式文件系统必须保证能够在断电或死机重启后继续工作。

正是出于兼容性方面的考虑,市面上大多数嵌入式智能设备如智能手机、数码相机、多媒体播放器(MP3、MP4 等)大多采用微软的 FAT 文件系统, 但是该文件系统由于设计上的不足导致实时性和可靠性比较差。

本文针对 FAT 文件系统的弱点, 提出了一种简单易行的方法,在不影响兼容性的前提下大幅度改善了该文件系统的实时性。关于 FAT 文件系统可靠性方面的研究,可参考文献[1]。

2 现状与背景

2.1 国内外研究现状

改善文件系统实时性的研究一般集中在预读和预分配两种技术上。预读技术有助于改善文件读取时间,T.Niranjan 等人于采用此技术实现了一套称为 MMFS[2]的文件系统,可以满足高质量多媒体实时播放的需求,但该方法需要对文件系统结构进行重新设计,无法保证与原版文件系统的兼容性。

预分配技术则可以减少文件写入时间,韩国的 S.Park 等人采用这种技术实现了一种可满足实时流媒体录制要求的 FAT 文件系统[3],但该方案仍有很大的局限性,同一时刻只允许一个

基金项目:国家高技术研究发展计划(863)(the National High-Tech Research and Development Plan of China under Grant No.2006AA01Z169)。
作者简介:刘可嘉(1983-),男,硕士生,主要研究领域为操作系统,文件系统;梁阿磊(1969-),男,博士,副教授,主要研究领域为操作系统,体系结构。
收稿日期:2007-11-21 **修回日期**:2008-02-25

文件处于可写状态。

国防科技大学的李昊等人提出了一种简化的 FAT 文件系统[4],可以在保证兼容性的前提下实现实时性和可靠性。但这种方法要求文件大小固定,适用范围较窄。

2.2 FAT 文件系统简介

根据微软公司的 FAT 文件系统规范文档[5],FAT 文件系统按簇号的位数分为三个版本:FAT12、FAT16 和 FAT32。一个 FAT 文件系统分区主要由启动信息块(Boot Parameter Block,BPB),文件分配表(FAT 表)和数据簇区(Clusters)三部分组成(如图 1 所示)。其中数据簇区存放文件和目录的内容数据,它被划分为若干个簇,每个簇的大小固定,例如在 FAT32 文件系统中的簇的大小一般为 4 KB,每个文件或目录可以包含若干个簇。FAT 表中包含了数据簇之间的分配和链接信息,每个 FAT 表项与数据簇一一对应。而 BPB 则包括了文件系统分区的全局信息,如分区的大小、FAT 表和根目录的起始位置、簇的大小等。

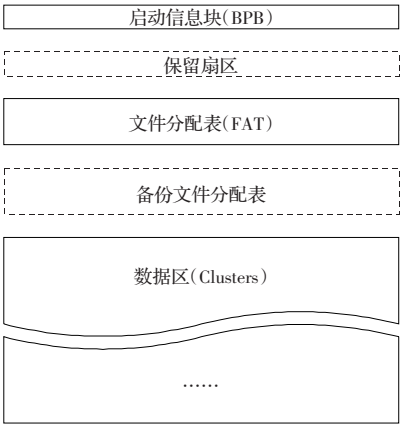

图 1　FAT 文件系统结构示意图

8.3 格式文件名			读写属性、创建时间等
"TEST　DAT"			…
创建、访问日期	首簇高 16 位	修改时间	首簇低 16 位	文件大小
…	0x0000	…	0x0005	0x0000A001

(a)目录项

	0	1	2	3	4	5	6	7	8	9	A	B	B	D	E	F
0	00	00	03	04	$	06	07	13	09	0A	0B	0D	00	0E	$	00
1	00	00	$	14	15	16	17	18	19	1D	00	00	00	$	00	00
⋮	… … … …															

(b)FAT 表

物理簇号 pcn	0x05	0x13	0x1D	… …
连续簇数 count	3	7	1	… …
逻辑簇号 lcn	1	4	11	… …

(c)压缩 FAT 表

图 2　FAT 示意图

在 FAT 文件系统中,一个文件也由三个部分组成:目录项、数据簇链表和文件内容。目录项位于文件所在的父目录[1]中,它记载了文件的文件名、大小、起始簇号和一些属性信息,例如图 2(a)中的目录项就表示了一个名为 TEST.DAT 的文件,它的大小约为 40 KB(十六进制 A001 字节),包含的数据起始簇号是 5,假定这个文件所在的 FAT 文件系统每个簇的大小为 4 KB,那么它就需要占用 11 个簇。文件簇链表位于 FAT 表中,通过一个单向链表的形式记录了文件的所有簇的编号(每个簇号对应的 FAT 表项中存放的是链表中的下一个簇号),并用一个特殊的值表示链表末尾,例如图 2(b)中,从 TEST.DAT 的起始簇号 05 开始,可以依次找到它的所有簇号(05,06,07,13,14,15,16,17,18,19,1D),注意这里用$符号代表链表末尾。文件内容则分散在数据簇区中这 11 个簇号对应的簇中。

3　压缩 FAT 表缓存

在 FAT 文件系统中,文件簇间的链表结构是 FAT 文件系统实时性不佳的一个主要原因。读写文件时,文件系统经常需要进行文件逻辑簇号(即这个文件中的第几个簇)到物理簇号(整个文件系统中的实际簇号)的转换,在这种链表结构中,要进行这种转换必须对 FAT 表进行遍历操作,所以响应时间与读写位置呈线性关系(时间复杂度为 $O(n)$,其中 n 为要访问的数据所在的逻辑簇号)。当文件较大时,遍历 FAT 表所花费的时间很长,难以满足实时需求。

对于这个问题,根本的解决方法是重新设计 FAT 文件系统的结构,改用类似 Linux EXT2 文件系统的树形结构管理文件簇间的分配和链接关系。但是这样一来,文件系统的兼容性就被破坏了。为了不破坏兼容性,可以在打开文件时在内存中建立 FAT 表缓存,并采用效率更高的数据结构"压缩 FAT 表"存放,以降低查询 FAT 表所需的时间复杂度,从而改善文件随机访问的响应时间。下面对压缩 FAT 表的原理做简单介绍。

通常情况下,FAT 文件系统在写文件时,总是试图顺序分配空间,因此 FAT 表中存在大量连续的表项。例如在图 2(b)中的这个例子里,(05,06,07)和(13,14,15,16,17,18,19)这两组簇号都是连续的,一组连续表项一般称为一个碎片,也就是说,文件 TEST.DAT 总共包含(05,06,07),(13,14,…,19)和(1D)这三个碎片。

对于这种情况,只需记录每个碎片中的第一个簇号和碎片的长度即可表示一个碎片,比如上述 TEST.DAT 文件的三个碎片可以简单地表示为(05,2)、(13,7)和(1D,1)。当文件的碎片数较少时,用这种表示法不仅可以大大压缩 FAT 表缓存所占的内存空间,还可以大大减少遍历时间。如果定义文件的碎片率 $\rho=\frac{\text{碎片数}}{\text{簇数}}$,则遍历这种压缩 FAT 表的时间复杂度可以降为 $O(\rho n)$。由于嵌入式文件系统中所存文件数量较少,且写文件的频度远远小于 PC 机,碎片率一般可以控制在 1%以下。

如果以数组形式保存压缩 FAT 表,并在每个碎片中记录起始簇号对应的逻辑簇号,如图 2(c)所示,则可以应用类似二分查找的算法将逻辑到物理簇号转换的时间复杂度降低为 $O(\log \rho n)$。因为 FAT 文件系统限制一个文件的大小至多为 4 GB,也就是说 $n\leqslant\frac{4\times2^{30}}{4\times2^{10}}=2^{20}$,所以利用压缩 FAT 表做一次逻辑到物理簇号的转换至多只需 20 次比较,时间复杂度为 $O(1)$,符合实时要求。

4　空闲簇索引

另一个导致 FAT 文件系统实时性不佳的原因也跟 FAT 表有关。在 FAT 文件系统中,簇的分配和释放也是通过 FAT 表完成的,如果一个 FAT 表项的值为 0,则表示它对应的簇是空闲的。这样,当文件系统给文件分配空间时,需要遍历 FAT 表直到找到空闲的簇。当文件系统空间所剩无几时,查找空闲簇常常需要遍历大半个 FAT 表,因此需要耗费大量的时间,无法满

[1] 在 FAT 文件系统中,"目录"其实是一个特殊文件,其内容包含了该目录中所有文件和子目录的目录项。

足实时要求。

针对这个问题,可以在文件系统加载时扫描整个 FAT 表,把所有空闲的 FAT 表找出来并按照压缩 FAT 表的方式索引起来,就可以把分配空间的时间复杂度降为 O(1),符合实时要求。此外,使用空闲簇索引可以对整个文件系统的空间分配做一些统筹规划,从而减少碎片的产生。例如文件系统 API 层可以提供扩展函数,允许开发者指定文件最终可能达到的大小。如果开发者指定了这一参数,就可以直接预分配这么多连续空间给这个文件,而不是边写边分配,这样就可以避免文件系统在同时写多个文件时产生大量碎片的情况。

本方案的一个缺点是在文件系统加载时需要扫描整个 FAT 表,这个过程一般在操作系统启动过程中完成。但 FAT 表的大小一般不到整个文件系统的 0.1%,因此并不会像 JFFS 那样造成系统启动速度过慢的情况。经实验,对于大小为 32 GB(微软 Windows XP 所允许的最大值)的 FAT 文件系统,建立空闲簇索引所需要的时间仅为 2 s 左右。

5 测试与分析

为测试压缩 FAT 缓存和空闲簇索引对改进 FAT 文件系统实时性的实际效果,本文在原型系统中实现了这两种技术,并使用国际上比较权威的文件系统性能测试软件 IOZone 进行了测试。

由图 3 中的测试结果可以看出,普通的 FAT 文件系统,随着文件增大,随机读写数据所用的时间显著增加。在进行压缩 FAT 缓存和空闲簇索引两项优化之后,这一状况得到了明显改善,无论文件大小,读写时间均基本维持在 100~150 ms 上下。

6 结论

为改善 FAT 文件系统的实时性,本文创新地提出了压缩 FAT 表和空闲簇索引技术。实验证明,这套技术方案可以大幅

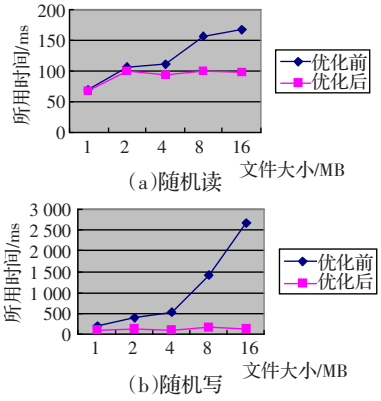

图 3 随机读写 1 MB 数据所用时间与文件大小的关系

度降低文件随机读写的响应时间,在不影响 FAT 文件系统兼容性的前提下基本满足实时性的要求。

参考文献:

[1] Liang A-lei,Liu Ke-jia,Li Xiao-yong,et al.FATTY:a reliable FAT file system[C]//Proceedings of the 10th EUROMICRO Conference on Digital System Design,Luebeck,Germany.[S.l.]:IEEE Computer Society Press,2007,1:390-395.

[2] Niranjan T,Chiuh T,Schloss G A.Implementation and evaluation of a multimedia file system[C]//Proceedings of IEEE International Conference on Multimedia Computing and Systems,1997,1:269-276.

[3] Park S,Ohm S Y.New techniques for real-time FAT file system in mobile multimedia devices[J].IEEE Transactions on Consumer Electronics,2006,52(1):1-9.

[4] 李昊,王跃科,周睿,等.CF 卡在大容量数据存储系统的典型应用[J].微计算机信息,2005,11(1):66-68.

[5] Microsoft Corporation.FAT32 file system specification[S/OL].http://www.microsoft.com/whdc/system/platform/firmware/fatgen.msp.

(上接 65 页)

a)用于检查 V_i 的赋值与 R 中的 $i-1$ 个变量的值的一致性,如果发现 V_i 的赋值与 R 中的变量的赋值不一致,记录下此变量的序号作为变量 V_i 的 $lastest_i$ 的值,之后,重新给变量 V_i 赋值,直到 V_i 的当前域 D 变成空。此时应该回溯到 V 中的第 $lastest_i$ 变量。重新给第 $lastest_i$ 个变量赋值。发生回溯是因为之前给第 $lastest_i$ 的赋值引起的。

5 结论

目前国内外对空间关系推理的研究取得了一定的进展,但还存在一些问题。例如在实际应用中,单独的方向关系有时不足以推导出人们希望的结果,而方向与距离的结合或与拓扑的结合有助于提高表达能力及推理结果的准确性。本文研究了基于投影区间的主方向关系表示方法以及拓扑关系表示方法,提出了利用约束满足问题求解结合方向关系和拓扑关系的推理求解算法,该算法能够解决拓扑和方向关系的约束满足问题,在基于 GIS 的系统中有一定实际应用价值。

参考文献:

[1] Randell D A,Cui Z,Cohn A G.A spatial logic based on regions and connection[C]//Nebel B,Swartout B,Rich C.Proc of the 3rd Int'l Conf Knowledge Representation and Reasoning.Los Allos: Morgan Kanfmann Publishers,1992:165-176.

[2] Goyal R,Egenhofer M J.The direction-relation matrix:a representation for directions relations between extended spatial objects[C]//The Annual Assembly and the Summer Retreat of University Consortium for Geographic Information Systems Science,1997.

[3] 曹菡.空间关系推理的知识表示与推理机制研究[D].武汉:武汉大学,2002.

[4] 郭平.定性空间推理技术及应用研究[D].重庆:重庆大学,2004.

[5] Clementini E,di Felice P,Hernandez D.Qualitative representation of positional information[J].Artificial Intelligence,1997,95(2):317-356.

[6] Gerevini A,Renz J.Combining topological and size information for spatial reasoning[J].Artificial Intelligence,2002,137(1/2):1-42.

[7] Xie Q,Liu Da-you,Yu Qiang-yuan,et al.Integrating orientation, distance and time for qualitative spatio-temporal reasoning[C]//Proceedings of 2004 International Conference on Machine Learning and Cybernetics,2004:2269-2273.

[8] 何建华,刘耀林.GIS 中拓扑和方向关系推理模型[J].测绘学报,2004,33(2):156-160.

[9] 许小艳.距离与方向关系的定性推理研究[D].重庆:重庆大学,2007.

[10] Dechter R,Frost D.Backjump-based backtracking for constraint satisfaction problems[J].Artificial Intelligence,2002,136:156-165.

[11] 谢琦.空间方位关系模型与时空结合推理的研究[D].吉林:吉林大学,2006.