B-树在 NTFS 索引目录管理中的应用研究

陈培德,吴建平,王丽清
(云南大学 信息学院 云南省高校数字媒体技术重点实验室,云南 昆明 650223)

摘 要:目前国内一些有关 NTFS 文件系统的书籍或杂志认为 NTFS 对索引目录的管理是采用 B+树结构,而只有少量书籍中认为 NTFS 对索引目录的管理是采用 B-树结构。针对这种争议,以 Windows7 操作系统为平台,以 NTFS 文件系统和文件目录为研究分析对象,用 WinHex 磁盘编辑为分析工具,对 NTFS 文件系统中元文件 $ MFT 文件夹记录的 90H 属性、A0H 属性和 B0H 属性进行分析。以 B-树的定义为衡量标准,对 NTFS 文件系统索引目录中的文件进行查找、删除和插入操作,来观察 NTFS 文件系统索引目录的结构变化。实验结果表明,NTFS 索引目录基本符合 B-树的定义。NTFS 文件系统对索引目录的管理是采用 B-树结构,但并非是标准的 B-树结构。

关键词:NTFS 文件系统;B-树;索引节点;索引目录
中图分类号:TP311.12 **文献标识码**:A **文章编号**:1673-629X(2016)09-0030-04
doi:10.3969/j.issn.1673-629X.2016.09.007

Study on Application of Index Directories in NTFS by B-tree

CHEN Pei-de, WU Jian-ping, WANG Li-qing
(Key Laboratory of Digital Media Technology of Universities and Colleges in Yunnan Province,
School of Information Science and Engineering, Yunnan University, Kunming 650223, China)

Abstract: The method for managing the index directories in NTFS is thought to use the B+ tree data structure in many books and magazines, and the B- tree data structure is used in the less books in the inland. Aiming at the dispute, taking the Windows7 Operation System as platform, the file directories in the NTFS as the analytic object, the WinHex as analytic tool, the attribute of 90H, A0H and B0H is analyzed in folder record for metafile $ MFT in NTFS. The definition of B-tree is used for the judgment standard. The files in the NTFS index directories are found, deleted, inserted, and to be observed the structure change of NTFS index directories. The results of experiment indicate that the NTFS index directories is accorded with basically the B-tree structure definition. The B-tree structure is used for the index directories in NTFS, but it's not a standard B-tree structure.

Key words: NTFS; B-tree; index node; index directories

0 引 言

NTFS 文件系统是随着 Windows NT 的第一个版本推出的一个性能优良的文件系统,目前已是 Windows 系列操作系统的重要组成部分。该系统对索引目录的管理非常复杂,国内少量有关 NTFS 文件系统的书籍认为:NTFS 文件系统对索引目录的管理采用的是 B-树结构。如:NTFS 对文件目录采用 B-树进行管理,这种技术比在 FAT 文件系统中使用链表技术管理优越得多[1];NTFS 利用 B-Tree 文件管理方法跟踪文件在磁盘上的位置[2];一个 NTFS 索引排序属性为一棵树,尤其是一棵 B-树,NTFS 使用 B-树,其与二叉树相似,但每个节点超过了两个孩子[3],等等。

1 B-树的定义

一棵 m 阶的 B-树满足下列 5 个条件:

(1)每个节点至多有 m 个孩子;

(2)除根节点和叶节点外,其他每个节点至少有 $\lceil m/2 \rceil$ 个孩子;

(3)根节点至少有两个孩子(唯一例外的是只包含一个根节点的 B-树);

(4)所有的叶节点在同一层,叶节点不包含任何关键字信息;

(5)有 k 个孩子的非叶节点恰好包含 $k-1$ 个关键字[4]。

在 B-树中每个节点的关键字从小到大排列。因

收稿日期:2015-12-02 **修回日期**:2016-03-09 **网络出版时间**:2016-08-01
基金项目:云南省科技创新强省计划项目(2014AB021);云南省高校数字媒体重点实验室开放基金项目(2015KFKT002)
作者简介:陈培德(1966-),男,工程师,研究方向为文件系统与数据恢复技术。

为叶节点不包含关键字,所以,叶节点实际上是树中并不存在的外部节点,且指向这些外部节点的指针为空,叶节点的总数正好等于树中的关键字总数加 1[4]。

2　文件夹记录的 90H、A0H 和 B0H 属性

在 NTFS 文件系统中最重要的元文件就是 $ MFT,它是 NTFS 卷中所有文件和文件夹的集合[5-7]。在元文件 $ MFT 中的文件夹记录中与文件夹相关的属性主要有 90H、A0H 和 B0H[8-10]。

90H 属性即 $ INDEX_ROOT 属性,在 $ MFT 记录中只有记录项为目录(即文件夹)才有该属性,它总是常驻有属性名的属性。在 90H 属性中常用到的索引项类型为文件名索引,名称为 $ I30。NTFS 对目录的管理采用 B-树结构,该属性是实现 NTFS 的 B-树的根节点[1]。

A0H 属性也就是索引分配属性,存储着组成索引的目录结构的所有节点的定位信息。它总是非常驻属性,没有最大最小值限制[1]。

B0H 属性即位图属性,它是由一系列的二进制位所构成的虚拟分配单元使用情况表。每一位代表一个分配单元的使用情况。该位为“0”时表示所对应的分配单元未使用,为“1”时表示所对应的分配单元已使用或者分配单元已坏(如果分配单元已坏,在元文件 $ BadClus 中会有记载)。一个分配单元在操作系统引导记录(英文全称 DOS Boot Record,缩写为 DBR[11-15])中已有定义,可以是 1 个簇,也可以是 1 个扇区,也可以是 2 个扇区,等等。

B0H 属性目前主要使用在两个地方:即索引目录和元文件 $ MFT 中[11-15]。在索引目录中,该属性一般为常驻属性,每一位表示一个索引节点号的使用情况。如果该位为“1”,表示所对应的索引节点有效;为“0”表示所对应的索引节点无效。

3　NTFS 中的 B-树

实验环境:

(1)操作系统:Windows 7;

(2)分析工具:WinHex 15.08。

实验步骤:

(1)在 Windows 7 下的 D 盘建立一个名为 abcd.vhd 的文件,文件大小为 500 MB;

(2)使用 Windows 7 的虚拟硬盘管理附加 abcd.vhd 文件为虚拟硬盘,在虚拟硬盘上建立一个 MBR 分区,分区大小为 466 MB,将该分区格式化为 NTFS,分配单元大小选择 4 096;

(3)在虚拟硬盘上建立一个文件夹,在该文件夹中建立 1 000 个文件名,文件名为 a000 ~ a999,其 B-树结构如图 1 所示。

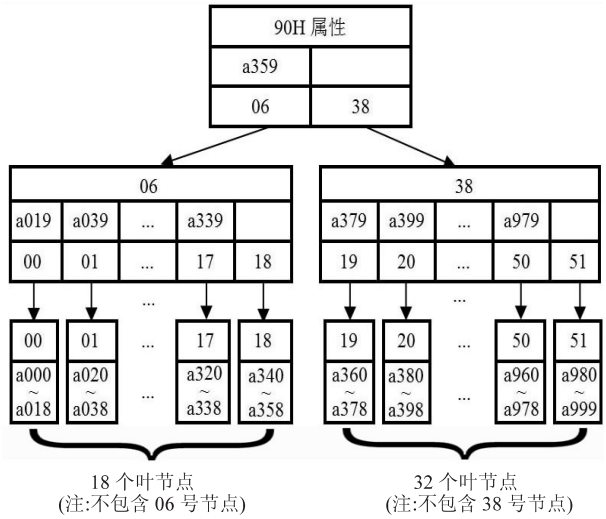

图 1　存储 1 000 个文件的 B-树结构图

对图 1 说明如下:

(1)在文件夹中存储了 1 000 个文件,这 1 000 个文件名分别存储在文件夹记录的 90H 属性、00 ~ 51 号节点中。其中:06 号和 38 号为非叶节点,而 00 ~ 05 号,07 ~ 37 号,39 ~ 51 号为叶节点。

(2)在 90H 属性(即 B-树的根节点)中存储 1 个索引文件名(即 a359)和 2 个索引节点号(即 06 和 38),与 B-树定义的第 3 条相符。

(3)在 06 号非叶节点中存储了 17 个索引文件名(即 a019,a039,…,a339)和 18 个指针(号为 00 ~ 05,07 ~ 18);未发现 90H 属性中的索引文件名 a359;与 B-树定义第 5 条相符,此时 $k = 18$,即有 18 个孩子(即指针)的非叶节点恰好包含 17 个关键字(即文件名)。

(4)在 38 号非叶节点中存储了 31 个索引文件名(即 a379,a399,…,a979)和 32 个指针(指针号为 19 ~ 37,39 ~ 51);与 B-树定义第 5 条相符,此时 $k = 32$,即有 32 个孩子(即指针)的非叶节点恰好包含 31 个关键字(即文件名)。

(5)在各叶节点(即 00 ~ 05 号,07 ~ 37 号,39 ~ 51 号)中均未发现 90H 属性、06 号非叶节点和 38 号非叶节点中的索引文件名,与 B-树定义的第 4 条相符。

(6)叶节点总数为 50 个,而关键字总数为 49 个,这与 B-树的定义最后叙述相符(即叶节点的总数正好等于树中所包含的关键字总数加 1)。

4　NTFS 的 B-查找分析

查找 a359 文件,在文件夹记录的 90H 属性就命中。

查找 a059 文件,将 a059 文件名与 a359 文件名进行比较,由于 a059<a359,而 a059 存储在 06 号非叶节点;通过折半查找的方式在 06 号非叶节点中命中。

查找 a324 文件,将 a324 文件名与 a359 文件名进行比较,由于 a324<a359,所以 a324 又与 06 号非叶节点中的 a339 比较;由于 a324<a339,因此,a324 文件在 17 号叶节点中通过折半方式与 17 号叶节点中所存储的文件进行比较,在 17 号叶节点中命中。

查找 a350 文件,将 a350 文件名与 a359 文件名进行比较,由于 a350<a359,所以 a350 又与 06 号非叶节点中的 a339 文件进行比较;由于 a350>a339,因此,a350 文件在 18 号叶节点中通过折半方式与 18 号叶节点中所存储的文件进行比较,在 18 号叶节点中命中。

5　NTFS 中 B-树的删除

1)将存储在 90H 属性中、06 号和 38 号非叶节点中的所有文件(即 a019,a039,…,a979,共计 49 个文件)删除后,其 B-树结构如图 2 所示。

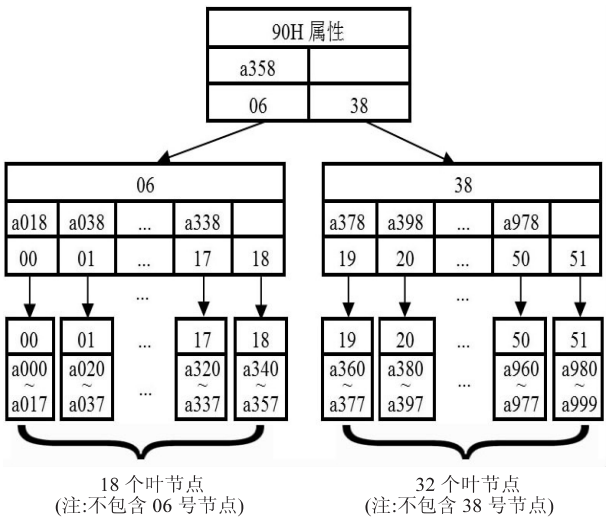

图 2　存储 951 个文件的 B-树结构图

从图 1 到图 2 有如下变化:

(1)将 18 号叶节点中的文件名 a358 上移至文件夹记录的 90H 属性中;

(2)将 00 号 ~ 17 号叶节点中的文件名 a018,a038,…,a338 上移至 06 号非叶节点中;

(3)将 19 号 ~ 50 号叶节点中的文件名 a378,a398,…,a978 上移至 38 号非叶节点中。

2)删除 a340 ~ a377 文件(即存储在 90H 属性、18 号和 19 号节点中的文件)后,B-树结构如图 3 所示。

从图 2 到图 3 有如下变化:

(1)将 06 号非叶节点中的文件名 a338 上移至文件夹记录的 90H 属性中;

(2)存储在 38 号非叶节点中的文件名 a378 下移至 20 号叶节点中;

(3)由于 18 号和 19 号叶节点中的文件已被删除,18 和 19 号叶节点已不再起作用,文件夹记录的 B0H 属性所记录的索引节点状态值由“1”变为“0”。

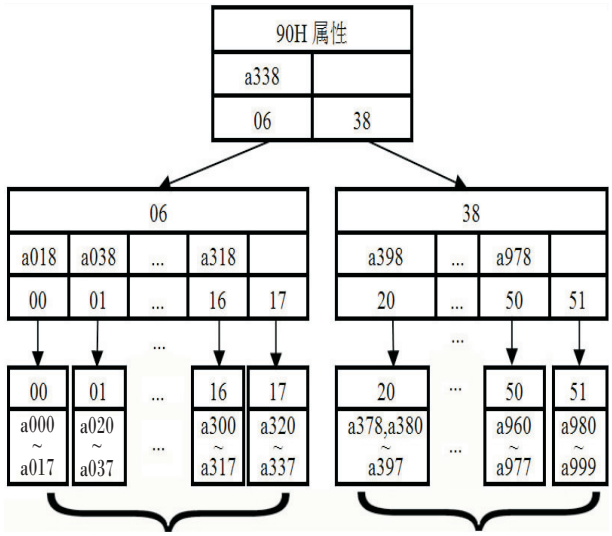

图 3　存储 914 个文件的 B-树结构图

3)删除 a001 ~ a017、a021 ~ a037 共计 34 个文件,其 B-树结构如图 4 所示。

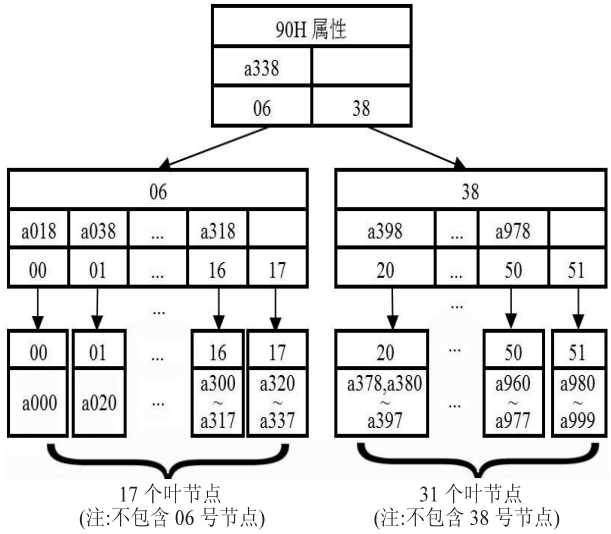

图 4　存储 880 个文件的 B-树结构图(1)

从图 3 到图 4 有如下变化:

(1)将 a001 ~ a017 文件删除后,00 号叶节点只存储一个文件,文件名为 a000;

(2)将 a021 ~ a037 文件删除后,01 号叶节点只存储一个文件,文件名为 a020。

6　NTFS 中 B-树的插入

1)在该文件夹中复制 22 个文件,文件名为 a97700 ~ a97721,其 B-树如图 5 所示。

将 a97700 ~ a97721 文件插入后,由于 a97700>a977,而 a97721<a978,所以,a97700 ~ a97721 文件名存放在 50 号叶节点中,位于 a977 之后,在该叶节点中共存储了 40 个文件名。

2)在该文件夹中复制 1 个文件,文件名为 a97722,其 B-树如图 6 所示。

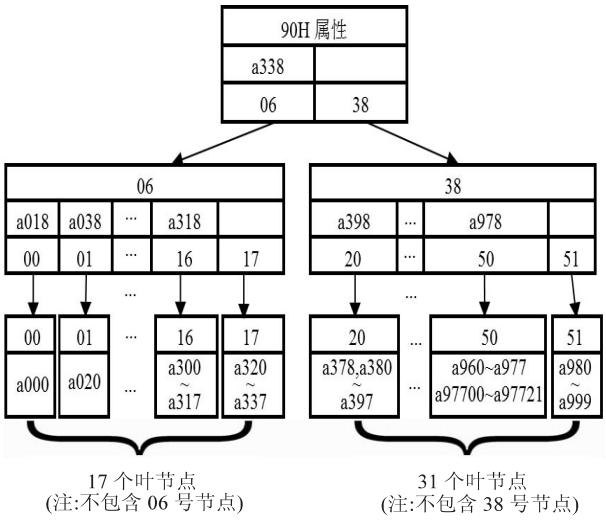

图 5　存储 902 个文件的 B-树结构图

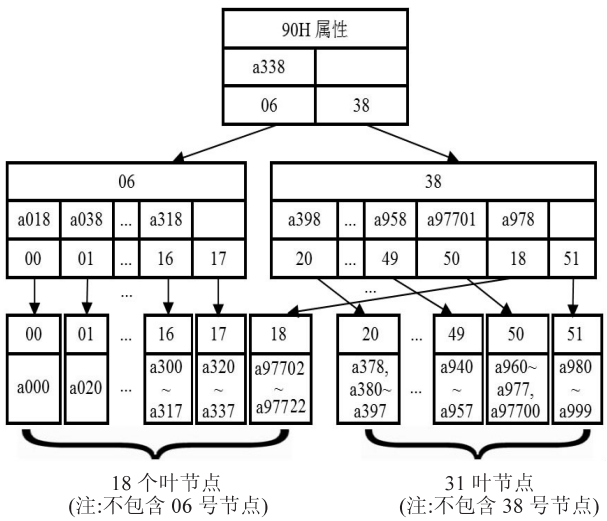

图 6　存储 880 个文件的 B-树结构图(2)

将 a97722 文件插入后,由于 a97722 > a97721,所以,a97722 存放在 a97721 文件名之后,由于该叶节点的空间为 4 096 字节,无法再存储文件名,将该叶节点进行分离。而 18 号叶节点和 19 号叶节点未使用,NTFS 文件系统将使用 18 号叶节点,将 50 号叶节点分离后,50 号叶节点存储的文件名为 a960 ~ a977,a97700,共计 19 个。

将 a97701 文件名上移至 38 号非叶节点,位于 a958 和 a975 之间;而将 a97702 ~ a97722 文件名移动到 18 号叶节点中,完成节点的分离。

7　结束语

在 NTFS 文件系统中索引目录主要存储在文件夹记录的90H属性和各个索引节点中,90H属性为B-树的根节点。但该节点并未完全遵循 B-树的定义"根节点至少有两个孩子(唯一例外的是只包含一个根节点的 B-树)"。各个索引节点的位置通过文件夹记录的 A0H 属性的数据运行列表来定位,文件夹记录的 B0H 属性记录了各索引节点的状态。每个索引节点的大小固定为 4 096 字节,各索引节点所包含的孩子数取决于文件名长度。当索引节点中文件数量不断增加,一个索引节点容纳不下时,将该索引节点进行分离。

总之,NTFS 文件系统对索引目录的管理是采用 B-树结构,但并非是一棵标准的 B-树结构。

参考文献:

[1] 陈培德,吴建平,王丽清. NTFS 文件系统实例详解[M]. 北京:国防工业出版社,2015.

[2] 张钟澍,陈代军,李新萌. 修复和维护你的硬盘[M]. 北京:北京希望电子出版社,2002.

[3] Carrier B. File system forensic analysis[M]. [s. l.]:Addison Wesley Professional,2005.

[4] 唐策善,李龙澍,黄刘生. 数据结构—用 C 语言描述[M]. 北京:高等教育出版社,2006.

[5] 吴伟民,刘　凯,江达强,等. NTFS B+树大目录结构动态解析[J]. 计算机工程与设计,2013,34(4):1376-1382.

[6] 吴伟民,林水宾,江达强,等. 基于 NTFS 大目录的文件创建方法[J]. 计算机应用,2014,34(2):417-420.

[7] 吴伟民,卢　琦,王振华,等. NTFS 目录下索引 B+树结构动态解析[J]. 计算机工程与设计,2010,31(22):4843-4846.

[8] Fathi B. 深入解析 Windows 操作系统(英文版)[M]. 第 5 版. 北京:人民邮电出版社,2009.

[9] Ionescs A. NTFS on-disk structure: visual basic NTFS programmer's guide[EB/OL]. 2009. http://www. alex-ionescu. com.

[10] Microsoft TechNet. Optimizing NTFS: disabling unnecessary access updates [EB/OL]. 2010. http://technet. microsoft. com/en-us/library/cc7679 61. aspx.

[11] 刘　伟. 数据恢复技术深度揭秘[M]. 北京:电子工业出版社,2010.

[12] 马　林. 数据重现—文件系统原理精解与数据恢复最佳实践[M]. 北京:清华大学出版社,2009.

[13] 汪中夏,张京生,刘　伟. RAID 数据恢复技术揭秘[M]. 北京:清华大学出版社,2010.

[14] 戴士剑,涂彦晖. 数据恢复技术[M]. 北京:电子工业出版社,2005.

[15] 刘乃琦,郭建东,张　可. 系统与数据恢复技术[M]. 成都:电子科技大学出版社,2008.