NTFS 索引目录 B-树结构解析及其应用①

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, 昆明 650223)

摘 要: 针对一些有关 NTFS 文件系统的书籍和杂志中认为 NTFS 文件系统对索引目录的管理是采用 B+树结构, 通过对 NTFS 文件系统元文件$MFT 文件夹记录的 90H 属性、A0H 属性和 B0H 属性以及索引结点结构的分析, 以实验的方式对索引文件进行查找、删除和插入运算来观察 NTFS 索引目录结构变化. 实验结果表明: 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.

关键词: NTFS 文件系统; B-树; 索引结点; 索引目录

Analysis and Application For B-Tree Structure of the Index Directories in NTFS

CHEN Pei-De, WU Jian-Ping, WANG Li-Qing

(Digital Media Technology Key Laboratory of Universities and Colleges in Yunnan Province, School of Information Science and Engineering, Yunnan University, Kunming 650223, China)

Abstract: Aiming at managing the index directories in the NTFS file system, it uses B+ tree in some of books and magazines of NTFS, analyzes the 90H attribute, A0H attribute, B0H attribute in the folder recorder in the metafile $MFT and the structure of the index node in NTFS file system, observes to change the index directions in the NTFS by researching, deleting, inserting the file name in it by the mode of experiment. The experiment results indicate that the structure of the index directories in NTFS uses B-tree structure, but it isn't the standard B-tree structure.

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

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, 目前它已经是 Windows 系列操作系统的重要组成部分. 然而在国内的许多有关 NTFS 文件系统的书籍, 普遍认为 NTFS 文件系统对索引目录的管理是采用 B+树结构, 国内的一些期刊杂志也刊登了有关"NTFS 对索引目录管理是采用 B+树结构"的论文,如: NTFS 利用 B+树文件管理方法来跟踪文件在磁盘上的位置[1]、NTFS 文件系统使用 B+树结构大目录对大型文件夹进行管理[2]、NTFS 目录索引 B+树结构是 NTFS 的最要和最复杂的数据结构之一[3]. 笔者按 B+树对 NTFS 文件系统中的文件进行检索时, 发现索引文件名在非叶结点就命中, 而并非是在叶结点命中, 在所有叶结点中并未发现非叶结点中的索引文件名. 因此, NTFS 对索引目录是否采用 B+树结构进行管理还需要实践的进一步检验.

1 B+树和B-树

要验证 NTFS 文件系统对索引目录管理是采用 B+树结构还是 B-树结构? 首先, 要对 B+树结构和 B-树进行深入的研究; 其次, 要对 NTFS 文件系统中的元文件$MFT 文件夹记录及索引结点进行深入的研究分析; 再次, 对 NTFS 文件系统中索引结点中所存储的文件名的逻辑结构进行分析; 最后画出各索引结点中所存储文件名的逻辑结构图, 并与 B+树和 B-树的结构进行比较, 如果 NTFS 索引结点中所存储的文件名的逻辑结构图符合 B+树结构, 则说明 NTFS 对索引目录的管理是采用 B+树结构; 如果符合 B-树结构, 则说明 NTFS 文件系统对索引目录的管理是采用 B-树结构.

一棵 m 阶的 B+树满足下列 3 个条件:

(1)有 k 个孩子的结点必有 k 个关键字; (2)所有叶结点, 包含了全部关键字的信息及指向相应的记录指

① 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技术重点实验室开放基金(2015KFKT002)
收稿时间:2015-12-01;收到修改稿时间:2016-01-11 [doi:10.15888/j.cnki.csa.005265]

针, 且叶子结点本身依照关键字的大小, 自小而大的顺序链接; (3)上面各层结点中的关键字, 均是下一层相应结点中最大关键字的复写(当然也可以采用“最小关键字复写”原则) [4].

一棵 m 阶的 B-树满足下列 5 个条件:

(1)每个结点至多有 m 个孩子; (2)除根结点和叶结点外, 其它每个结点至少有[m/2]个孩子; (3)根结点至少有两个孩子(唯一例外的是只包含一个根结点的 B-树); (4)所有的叶结点在同一层, 叶结点不包含任何关键字信息; (5)有 k 个孩子的非叶结点恰好包含 k-1 个关键字[4].

在 B-树中每个结点的关键字从小至大排列. 因为叶结点不包含关键字, 所以, 叶结点实际上是树中并不存在的外部结点, 且指向这些外部结点的指针为空, 叶结点的总数正好等于树中所包含的关键字总个数加 1[4].

2 文件夹的记录属性

NTFS 文件系统主要由元文件、用户文件、文件夹和数据组成, 在 NTFS 文件系统中最重要的元文件就是$MFT, 它是 NTFS 卷中所有文件和文件夹的集合. 元文件$MFT 由许多记录组成, 每条记录的大小均为 1024 字节, 每条记录由记录头和若干个属性组成[1]. 在文件夹记录中与文件夹相关的属性主要有 90H 属性、A0H 属性和 B0H 属性.

90H 属性即$INDEX_ROOT 属性, 在$MFT 记录项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, 该属性是实现 NTFS 的 B-树的根结点[6].

当文件夹中的文件非常少时, 文件夹中的所有文件名都存储在文件夹记录的 90H 属性中, 此时文件夹记录没有 A0H 属性和 B0H 属性; 如果文件夹中的文件数量不断, 90H 属性无法存放时, NTFS 文件系统将会在 90H 属性后增加一个 A0H 属性和一个 B0H 属性[6].

A0H 属性也就是索引分配属性, 存储着组成索引目录结构的所有结点的定位信息, 它总是非常驻属性, 没有最大最小值限制.

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$BadClus 中会有记载). 一个分配单元在操作系统引导记录中已经有定义, 可以是一个簇, 也可以是 1 个扇区, 也可以是 2 个扇区等等. 该属性目前主要使用在两个地方: 即索引目录和元文件$MFT 中. 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结点号的使用情况. 如果该位为“1”时表示所对应的索引结点有效, 为“0”时表示所对应的索引结点无效.

3 索引结点结构

每个索引结点以“INDX”开头, 对于每个文件夹而言, 每个索引结点均有一个唯一的编号. 经实验研究, 索引结点大小描述、索引结点编号与 NTFS 中每个簇的扇区数有一定的关系, 结果见表 1 所示, 从表 1 可知, 每个索引结点大小为 4096 字节[6].

表 1　每个簇扇区数与索引结点大小及编号表[6]

每个簇的扇区数	索引结点大小描述	索引结点编号
1	8 簇	0、8、16、24、32、…
2	4 簇	0、4、8、12、16、…
4	2 簇	0、2、4、6、8、…
8	1 簇	0、1、2、3、4、…
16	4096 字节	0、8、16、24、32、…
32	4096 字节	0、8、16、24、32、…
64	4096 字节	0、8、16、24、32、…
128	4096 字节	0、8、16、24、32、…

NTFS 的索引结点可以分四种类型: 即有效叶结点、无效叶结点、有效非叶结点和无效非叶结点[6]. 有效叶结点和有效非叶结点在文件夹记录的 B0H 属性中所对应的二进制位的值为“1”. 而无效叶结点和无效非叶结点在文件夹记录的 B0H 属性中所对应的二进制位的值为“0”.

在有效叶结点中, 以“INDX”为开始标志, 以第 1 个“10 00 00 00 02 00 00 00”(存储形式)为结束标志, 之间所存储的文件名为有效文件名, 而之后所存储的文件名为无效文件名. 而在有效非叶结点中, 以“INDX”为开始标志, 在每个文件名后均有一个指针, 以第 1 个“18 00 00 00 03 00 00 00 +指针”(存储形式)为结束标志, 之间所存储的文件名和指针为有效文件名和有效指针, 而之后所存储的文件名和指针为无效. 从有效非叶结点的结构可知, 有效文件名的数量等于有效指

针数量加 1. 该存储结构与“B-树”中的“(5)有 k 个孩子的非叶结点恰好包含 k-1 个关键字”相吻合(注: 文件名为关键字, 文件名后的指针为孩子).

4 NTFS文件系统索引目录结构

为了验证 NTFS 对索引目录的管理是采用 B-树结构, 笔者将索引目录分为小目录、中目录和大目录, 其划分没有明显的界线. 笔者认为当目录中存放的文件名比较少时, 目录中的所有文件名都存储在文件夹记录的 90H 属性中的目录称之为小目录[6]; 当目录中的文件名存储在多个索引结点中, 文件夹记录的 90H 属性为索引目录的根结点, 其他所有索引结点均为叶结点的目录称之为中目录[6]; 当目录中的文件名存储在多个索引结点中, 文件夹记录的 90H 属性为索引目录的根结点, 在该目录中既有非叶结点, 也有叶结点的目录称之为大目录[6]. 下面以实验的形式分别论述小目录、中目录和大目录的 B-树结构.

为了分析 NTFS 索引目录结构, 实验采用 3 个步骤(实验环境: 操作系统: Windows 7, 实验分析软件: WinHex 15.1):

(1) 使用 Windows 7 的虚拟磁盘管理功能, 在 D 盘的根目录下建立一个文件(文件大小为 100MB), 并将该文件附加为一个虚拟逻辑盘(假设盘符为 H:), 将 H 盘格式化成 NTFS 文件系统, 每个簇的扇区数选择 2(即 1024 字节), 索引结点编号为 0、4、8、12、16、....

(2) 在 H 盘的根目录下分别建立名为 A007、A100 和 A1000 的文件夹; 在 A007、A100 和 A1000 文件夹中分别存放 7 个文件、100 个文件和 1000 个文件; 文件名分别 a000~a006; a000~a099 和 a000~a999;

(3) 对 A007、A100 和 A1000 文件夹中所存储的文件名的逻辑结构进行研究.

4.1 小目录的 B-树结构

当文件夹下的文件很少时, 所有的文件名都存储在文件夹记录的 90H 属性中, 而 90H 属性是该文件夹记录的最后一个属性.

实验 1. 在 A007 文件夹中存储了 7 个文件名, 文件名分别为 a000~a006, A007 文件夹的 B-树结构图如图 1 所示, 从图 1 可知, 这是一个只有一个结点的 B-树结构. 此时 A007 文件夹的 B-树结构并不明显.

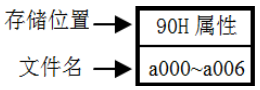

图 1 A007 文件夹的 B-树结构

4.2 中目录的 B-树结构

当文件夹中的文件不断增加时, 文件夹记录的 90H 属性无法存储文件夹中的文件名时, 将文件夹中的一部分文件移动到各索引结点中. 此时, 在文件夹记录 90H 属性后会自动添加一个 A0H 属性和一个 B0H 属性. 在 A0H 属性中存放了各索引结点的数据运行列表, 数据运行列表确定了各索引结点的存放位置. 而 B0H 属性则使用一位来表示各索引结点的状态, 如果该位为“1”表示所对应的索引结点为有效, 如果为“0”表示所对应的索引结点为无效.

实验 2. 在 A100 文件夹中, 存储了 100 个文件, 文件名为 a000~a099. A100 文件夹的 B-树结构如图 2 所示, 对图 2 说明如下:

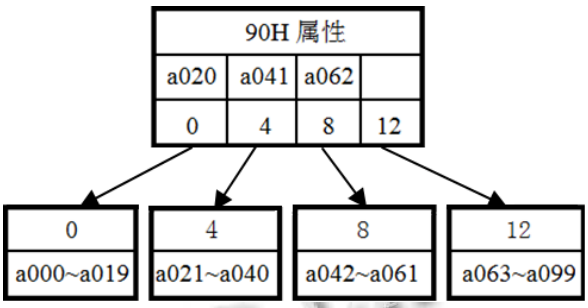

图 2 A100 文件夹的 B-树结构图

(1)A100 文件夹中的所有文件名分别存储在文件夹记录的 90H 属性、0 号、4 号、8 号和 12 号叶结点中; (2)在 90H 属性中存储了 3 个索引文件名和 4 个指针, 此时 k 等于 4; 这与 B-树中所描述的“(5)有 k 个孩子的非叶结点恰好包含 k-1 个关键字”论断相符合; 并不符合 B+树中的“(1)有 k 个孩子的结点必有 k 个关键字”的论断; (3)在 0 号、4 号、8 号和 12 号叶结点中所存储的文件名均未发现 90H 属性中的 3 个索引文件名. 这与 B-树中所描述的“(4)所有的叶结点在同一层, 叶结点不包含任何关键字信息”论断相符合; 并不符合 B+树中所描述的“(2)所有叶结点, 包含了全部关键字的信息”的论断. (4)叶结点的总数等于 4, 而关键字总数等于 3; 这与 B-树中所描述的“叶结点的总数正好等于树中所包含的关键字总个数加 1”论断相符合;

从对图 2 的说明可知, A100 文件夹的总体特性与

B-树的特性相符合, 而并不符合 B+树的特性.

4.3 大目录的 B-树结构

当文件夹下的文件不断增加时, 文件夹的 B-树结构由两层增加到三层.

实验 3. 在 A1000 文件夹中, 存储了 1000 个文件, 文件名为 a000~a999. 文件夹的 B-树结构如图 3 所示, 其说明如下:

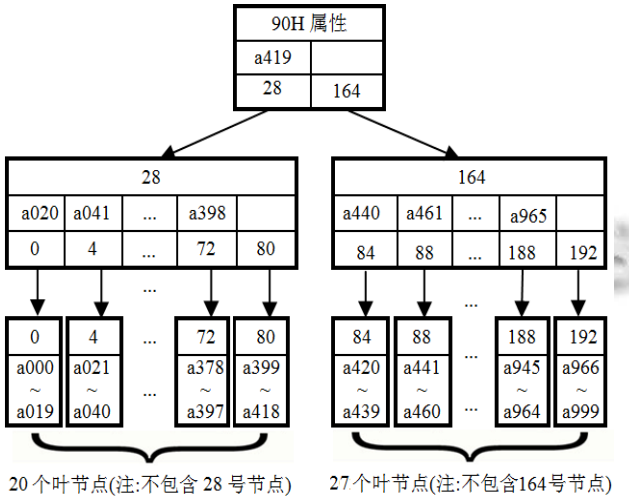

图 3　A1000 文件夹的 B-树结构图

(1) A1000 文件夹中的所有文件名分别存储在文件夹记录的 90H 属性、0 号至 192 号结点中; 各结点中文件名的排列顺序是从小到大. 以 28 号非叶结点为例, 文件名的排列顺序为: a020<a041<…<a398;

(2) 在 90H 属性存储了 1 个索引文件名(文件名为 a419)和 2 个指针(指针号为 28 和 164), 此时 k 等于 2;

(3) 28 号结点为非叶结点, 在 28 号非叶结点中存储了 19 个索引文件名和 20 个指针, 此时 k 等于 20;

(4) 164 号结点为非叶结点, 在 164 号非叶结点中存储了 26 个索引文件名和 27 个指针, 此时 k 等于 27;

(5) 叶结点共有 47 个, 所有的叶结点在同一层(即第 3 层), 在这 47 个叶结点中均未发现 90H 属性、28 号和 164 号非叶结点中所存储的索引文件名, 即叶结点不包含任何关键字信息;

(6) 叶结点的总数为 47, 而树中所包含的关键字总数为 46.

从对图 3 的说明可知, A1000 文件夹的总体特性与 B-树的特性相符合, 而并不符合 B+树的特性.

5　NTFS索引目录B-树的运算

以上分析了 NTFS 索引目录结构采用的是 B-树结构, 以下按 B-树对 NTFS 索引目录进行查询、删除和插入运算, 进一步验证结论的正确性.

5.1 NTFS 索引目录 B-树查找

(1) 在图 3 中, 查找 a419 文件, 在文件夹记录的 90H 属性(即 B-树的根结点)命中;

(2) 查找 a041 文件, 由于 a041<a419, 到 28 号非叶结点中查找, 在 28 号非叶结点命中;

(3) 查找 a379 文件, 由于 a379<a419, 到 28 号非叶结点与该结点中的索引文件名比较, 又由于 a377<a379<a398, a379 文件必然存储在 72 号叶结点, 即在 72 号叶结点命中.

(4) 查找 a461 文件, 由于 a419<a461, 到 164 号非叶结点中查找, 在 164 号非叶结点命中;

(5) 查找 a445 文件, 由于 a419<a445, 到 164 号非叶结点与该结点中的索引文件名比较, 又由于 a440<a445<a461, a445 必然存储在 88 号叶结点, 即在 88 号叶结点命中.

5.2 NTFS 索引目录 B-树删除

(1) 将 90H 属性中的索引文件、28 号和 164 号非叶结点中的所有文件(即 a020、a041、a062、...、a965, 共计 46 个)删除, 其 B-树结构如图 4 所示, 从图 3 到图 4 有如下变化:

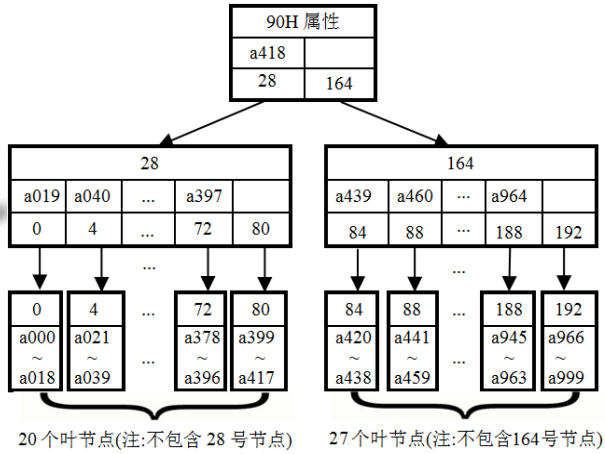

图 4　存储 954 个文件的 B-树结构图

① 将 80 号叶结点中的文件名 a418 上移动到文件夹记录的 90H 属性中(即 B-树的根结点);

② 分别将 0 号至 72 号叶结点中的最后一个文件名 a019、a040、...、a397 上移至 28 号非叶结点中;

③ 将 84 号至 188 号结点中的文件名 a439、a460、...、a964 上移至 164 号非叶结点中.

(2) 删除 0 号~80 号结点中的所有文件后, A1000 文件夹记录的 A0H 属性没有发生变化; B0H 属性中记录索引结点状态值由“FF FF FF FF FF FF 01”变为“00 00 E0 FF FF FF 01”,即 0 号结点至 80 号结点由有效状态变为无效状态; 90H 属性只存储一个指针, 没有存储文件名, 这与“B-树的定义(3)根结点至少有两个孩子(唯一例外的是只包含一个根结点的 B-树)”不相符; 但可以使用“(5)有 k 个孩子的非叶结点恰好包含 k-1 个关键字”来解释, 即在该结点中 k 等于 1. B-树结构图如图 5 所示.

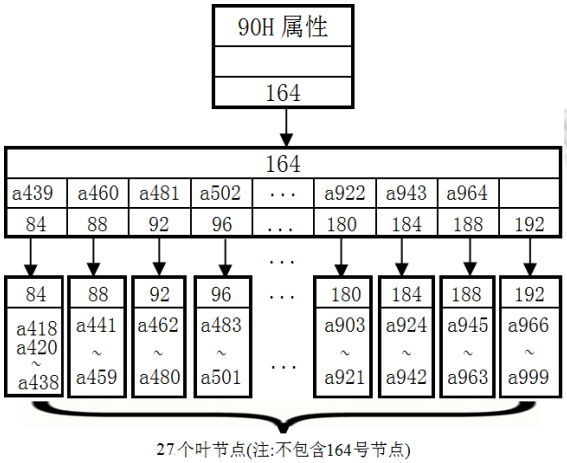

图 5 存储 555 个文件的 B-树结构图

从图 4 到图 5 有如下变化: (1)将 90H 属性中存储的文件名 a418 下移至 84 号叶结点中, 90H 属性只存储了一个指针(指针号为 164); (2)由于 0 号至 80 号结点中的文件被删除, 文件夹记录的 B0H 属性所描述的结点状态值由“1”变为“0”.

5.3 NTFS 索引目录 B-树插入

将 a000~a319(共计 320 个文件)文件复制到该文件夹中. 在该文件夹中存储的文件名为 a000~a0319, a418,a420~a999(除 a440,a461,a483,...,a965 外),共计 875 个, 各叶结点存储的文件名如表 2 所示.

表 2 各叶结点存储的文件名情况表

结点号	文件名	结点号	文件名	结点号	文件名
0	a418, a420~a438	60	a294~a318	136	a693~a711
4	a021~a040	84	a000~a019	140	a714~a732
8	a042~a061	88	a441~a459	144	a735~a753
12	a063~a082	92	a462~a480	148	a756~a774
16	a084~a103	96	a483~a501	152	a777~a795
20	a105~a124	100	a504~a522	156	a798~a816
24	a126~a145	104	a525~a543	160	a819~a837
28	a147~a166	108	a546~a564	168	a840~a858
32	a168~a187	112	a567~a585	172	a861~a879
36	a189~a208	116	a588~a606	176	a882~a900
40	a210~a229	120	a609~a627	180	a903~a921
44	a231~a250	124	a630~a648	184	a924~a942
48	a252~a271	128	a651~a669	188	a945~a963
56	a273~a292	132	a672~a690	192	a966~a999

注: 42 个叶结点(不包括 64,68,72,76 和 80 号空结点), 52 号和 164 号为非叶结点.

从表 2 可知, 各叶结点存储的文件总数为 834 个; 从表 3 可知, 52 号非叶结点存储文件总数为 18 个而指针数为 19 个; 从表 4 可知,164 号非叶结点存储文件总数为 22 个而指针数为 23 个. 而文件夹记录的 90H 属性中存储 1 个文件(文件名为 a586)和两指针(指针号为 164 和 52),其 B-树结构如图 6 所示. 从图 5 到图 6 有如下变化:

(1) 将 164 号非叶结点分离, 即将 164 号非叶结点存储的索引文件名 a607、a628、...、a964 和指针 116、120、...、188、192 移至 52 号非叶结点中, 52 号非叶结点存储的文件名及指针如表 3 表示; 164 号非叶结点存储的文件名及指针如表 4 表示;

表 3 52 号非叶结点所存储的文件名及指针

文件名	指针	文件名	指针	文件名	指针	文件名	指针
a607	116	a712	136	a817	156	a922	180
a628	120	a733	140	a838	160	a943	184
a649	124	a754	144	a859	168	a964	188
a670	128	a775	148	a880	172		192
a691	132	a796	152	a901	176		

表 4 164 号非叶结点所存储的文件名及指针

文件名	指针	文件名	指针	文件名	指针	文件名	指针
a020	84	a146	24	a272	48	a502	96
a041	4	a167	28	a293	56	a523	100
a062	8	a188	32	a319	60	a544	104
a083	12	a209	36	a439	0	a565	108
a104	16	a230	40	a460	88		112
a125	20	a251	44	a481	92		

(2) 将 a586 文件上移至 90H 属性中, 在 90H 属性后放置 2 个指针, 指针号为 164 和 52;

(3) 将 84 号叶结点中的文件移至 0 号叶结点中, 在 0 号叶结点存放了 20 个文件, 文件名为 a000~a019;

(4) 在 4 号至 48 号、56 号叶结点中分别存放 20

个文件, 在 60 号叶结点中存储 26 个文件;

(5) 文件夹记录的 B0H 属性索引结点状态值由“00 00 E0 FF FF FF 01 变为“FF FF E0 FF FF FF 01”, 即 0~60 号结点由无效状态变为有效状态.

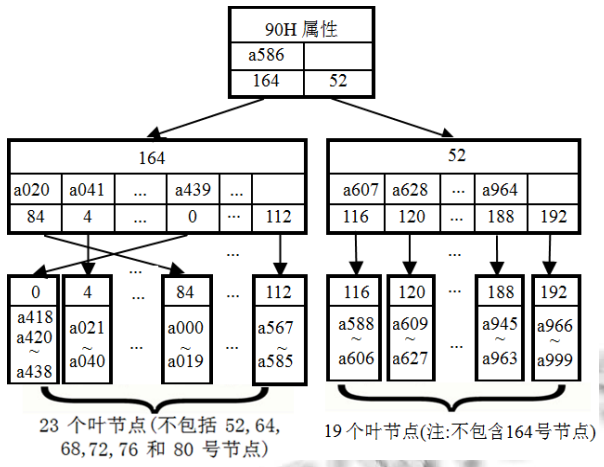

图 6　存储 875 个文件的 B-树结构图

6　结语

NTFS 文件系统中索引文件主要存储在文件夹记录的 90H 属性和各个索引结点中, 90H 属性为 B-树的根结点. 但该结点有时并未完全遵循 B-树的定义“(3)根结点至少有两个孩子(唯一例外的是只包含一个根结点的 B-树)”. 各个索引结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 属性记录了各索引结点的状态. 除根结点外, 每个索引结点的大小固定为 4096 字节, 各索引结点所包含的孩子数取决于文件名的长度, 当索引结点中的文件数量不断增加, 一个索引结点容纳不下时, 该索引结点将会进行分离. 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 刘伟.数据恢复技术深度揭秘.北京:电子工业出版社 2010.

2 吴伟民,刘凯,江达强,苏庆.NTFS B+树大目录结构动态解析.计算机工程与设计,2013,34(4):1376–1382.

3 吴伟民,卢琦,王振华.NTFS 目录下索引 B+树结构动态解析.计算机工程与设计,2010,31(22):4843–4846.

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

5 马林.数据重现—文件系统原理精解与数据恢复最佳实践.北京:清华大学出版社.2009.

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

7 汪中夏,张京生,刘伟.RAID 数据恢复技术揭秘.北京:清华大学出版社,2010.

8 戴士剑,涂彦晖.数据恢复技术.北京:电子工业出版社,2005.

9 刘乃琦,郭建东,张可.系统与数据恢复技术.成都:电子科技大学出版社,2008.

10 Ben Fathi.深入解析 Windows 操作系统 (第 5 版英文版).北京:人民邮电出版社,2009.

11 Carrier B. File System Forensic Analysis. Addison Wesley Professional. 2005.

12 Solomon DA. Inside Windows NT, 2nd Ed. Published by Arrangement with the Original Publisher, Microsoft Press, a division of Microsoft Corporation, Redmond Washington, U.S.A, 1998.

13 Ionescs A. NTFS on-disk structure: Visual basic NTFS programmer’s guide. http://www.alex-ionescu.com. [2009].

14 Microsoft TechNet. Optimizing NTFS. Disabling unnecessary access updates. http://technet.microsoft.com/en-us/library/cc767961.aspx. [2010].