(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 103324552 B
(45)授权公告日 2016.01.13

(21)申请号 201310224296.0
(22)申请日 2013.06.06
(73)专利权人 西安交通大学
地址 710049 陕西省西安市咸宁西路28号
(72)发明人 张兴军 朱跃光 董小社 朱国峰 王龙翔 姜晓夏
(74)专利代理机构 西安通大专利代理有限责任公司 61200
代理人 蔡和平
(51)Int.Cl.
G06F 11/14(2006.01)
G06F 17/30(2006.01)
(56)对比文件
CN 101452465 A,2009.06.10, 全文.
CN 103136335 A,2013.06.05, 全文.
US 2007/0179999 A1,2007.08.02, 全文.
审查员 万洋

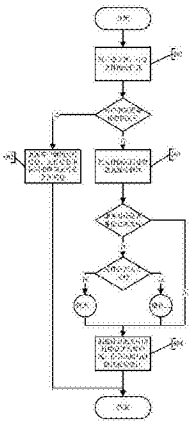
权利要求书3页 说明书17页 附图17页

(54)发明名称

两阶段单实例去重数据备份方法

(57)摘要

本发明公开一种两阶段单实例去重数据备份方法，备份期间来对数据进行两阶段的去重，首先是文件级的重复数据检测，先查询本地日志，判断是否有相同文件存储过，有则通知用户，完成备份操作。本地未存储过，则通知服务器端的备份程序查询数据库，判断是否有相同内容的文件，如果查找到，则只为客户端创建链接指向该文件，服务器端登记该客户端对该文件的引用；如果是新文件，将该文件进行上传，两端记录文件信息；文件上传到服务器端后，后台程序对文件进行下一步的处理，将小文件拼接起来，避免空间浪费；将大文件进行按类型分别存储，定期进行相似文件比较，分组后进行第二阶段的差异去重。



CN 103324552 B

1. 两阶段单实例去重数据备份方法，其特征在于，包括以下步骤：

601)、客户端打开需要备份的文件，计算文件内容，产生文件元信息，检查本地数据库，判断该文件是否已经被客户端存储过：如果该文件已经被客户端存储过则转到步骤 602)；如果对于客户端是没有备份过的新文件则转到步骤 603)；

602)、更新客户端的文件信息，将该文件元信息中的文件标示指向之前保存过的相同文件，备份流程结束；

603)、将需要备份的文件元信息发送到服务器端，包括文件大小、类型、Hash 值、备份时间信息；进一步判断该文件是否在服务器端被存储过；如果服务器端之前存储过相同的文件则转到步骤 604)；如果没有则接着判断此文件的大小是否符合标准：如果是小文件执行备份策略一，执行完成后转到步骤 604)；如果是大文件执行备份策略二，执行完成后转到步骤 604)；

604)、服务器端创建此文件的链接，并且更新服务器端的链接表，将文件链接发送给客户端进行存储，作为将来还原和删除文件的凭证；备份流程结束；

所述小文件小于文件系统分配单元 100 倍；所述大文件大于或等于文件系统分配单元 100 倍；

步骤 603) 中执行备份策略一具体包括：

701)、根据需要备份文件类型判断此类文件是否属于文本类文件：如果是文本类文件则进入到步骤 702)；如果不是文本类文件则进入到步骤 704)；

702)、将该文件追加写入到易压缩文件存储区中的大文件之中，并且更新文件元信息，接着判断该大文件是否超过了标准尺寸：如果该文件尺寸超过了规定的尺寸则转入到步骤 703)；如果该文件尺寸没超过规定的尺寸则直接结束；

703)、将达到规定尺寸的文件进行压缩后转入步骤 705)；

704)、将该文件追加写入到不易压缩文件存储区中的大文件之中，并且更新文件元信息，接着判断该大文件是否超过了标准尺寸：如果该文件尺寸超过了规定的尺寸则转入到步骤 705)；如果该文件尺寸没超过规定的尺寸则直接结束；

705)、根据当前时间创建一个新的空文件，用于存储新到来的文件；

步骤 603) 中执行备份策略二具体包括：

801)、根据文件的类型以及文件的 Hash 值，选择文件的目录：不同的文件类型存储于不同的一级目录，同类型文件中 Hash 值前 12 位不同的文件存储于不同的二级目录；

802)、将文件写入到步骤 801) 指定的目录之中；

803)、更新文件元信息，将文件的元信息写入到服务器端的数据库之中，流程结束。

2. 根据权利要求 1 所述的两阶段单实例去重数据备份方法，其特征在于，所述规定的尺寸为 64MB。

3. 根据权利要求 1 所述的两阶段单实例去重数据备份方法，其特征在于，还包括服务器端对某类型的大文件进行第一次相似文件的归并过程：

1400)、首先在已经按类型分类的文件中挑选一种文件作为备选文件；将该类文件的信息按照文件尺寸的顺序读入到一个数据库的表 t0 中；

1401)、从表 t0 中找到文件尺寸的中位数 Median，以 Median 为参考，寻找文件大小在 0.909*Median ～ 1.1*Median 之间的文件的信息，如果不存在这样的文件信息，则转到

1405）步，如果存在这样的文件信息，则转到 1402）步；

1402）、根据匹配字节串的特征值算法，计算这些文件的特征值，并根据设定的阈值，判断每个文件与尺寸为 Median 的文件之间特征值的差异是否在阈值以内，如果存在差异在阈值以内的文件，则转入到 1403）步，如果没有则转入到 1405）步；

1403）、将步骤 1402）找到的差异在阈值以内的文件的信息存储在表 t2 中，并标示为第 i 组，并且计算文件的平均特征值以及平均尺寸，并且将这些信息以及组号存入到表 t3 之中；表 t0 中删除这组文件所对应的项；检测表 t0 中是否还有其他未处理的文件项，如果没有则转入到 1404）步，如果有则转入 1401）步；

1404）、将表 t2 中的项，在文件信息总表中标示为已分类的文件，并且将表 t2 与表 t3 保存起来，过程完毕；

1405）、将尺寸为 Median 的文件的数据项存入到表 t1 中，从表 t0 中删除此项，之后判断 t0 表是否为空，如果为空则直接结束，如果不为空则转入到 1401）步；

其中特征向量中每个特征值与尺寸为 Median 的文件之间特征值的差异在 10％以内；所述特征向量为差异在阈值以内的文件的特征值的集合；i 初始值为 0，每标示完一组自增 1。

4. 根据权利要求 3 所述的两阶段单实例去重数据备份方法，其特征在于，在进行完第一次相似文件归并过程之后，服务器端定期检查文件系统日志，将新到来的文件进行归并，新到来的文件包括以下两种情况：

1）与之前的分组的文件相似，因此首先计算新到来的文件的特征值与文件尺寸，并且将其与已知文件分组的平均特征值与平均尺寸进行比较，差异在阈值范围内，则划归到一组，并且重新调整该分组的平均特征值与平均尺寸；

2）与之前未被分组的文件或者新到来的文件中的某些文件相似，因此第二步则将这些文件再一次的进行和所述第一次相似文件的归并过程相同的过程，创建新的文件分组。

5. 根据权利要求 4 所述的两阶段单实例去重数据备份方法，其特征在于，第 2）中情况具体的归并过程包括：

1500）、首先通过文件系统将上次进行相似文件判断之后新到来的文件的信息存入到一个数据库表 t0 之中，并且计算这些文件的特征值，将特征值存入到表 t0 中；

1501）、读取 t0 表中的第一个文件的文件尺寸 Size，并且查找平均文件尺寸在 0.909Size ～ 1.1Size 的文件分组，然后判断是否查找到此类分组；如果没有找到此类分组则继续判断表 t0 是否读完，如果否，则转到 1501）步读取表 t0 中的下一个文件；如果是，则继续判断表 t0 所有项是否已经删除了，如果删除完则直接结束，如果还有则转到 1504）步；如果找到此类分组，则转到 1502）步；

1502）、将 t0 表中的第一个文件的特征值与步骤 1501）找到的每个分组的平均特征值进行比较，如果存在差异在阈值之内的分组，则将该文件划归到该分组之中；

1503）、如果存在差异在阈值之内的分组，将 t0 表中的第一个文件的信息存入到已分类表 t2 中，并且重新计算其所属文件分组的平均特征值以及平均文件尺寸，更新分组信息表 t3，并且将该文件的信息从表 t0 中删除；判断表 t0 是否已经读完，如果否，则转到 1501）步读取表 t0 中的下一个文件，如果是，则继续判断表 t0 所有项是否已经删除了，如果删除完则直接结束，如果还有则转到 1504）步；

1504)、将之前未被分组的文件的信息读入到表 t0 中，这样表 t0 中保存着未被分类的新文件以及之前未被分类的文件，并且保证表中的文件信息按文件的尺寸排序，之后按照与第一次相似文件的归并过程相同的过程处理。

6. 根据权利要求 3 所述的两阶段单实例去重数据备份方法，其特征在于，还包括针对每一分组进行差异去重的处理步骤，具体如下：

a)、选择几个连续的字节的取值为文件的分块边界，将某一分组中所有文件在逻辑上进行分块；

b)、首先计算基准文件的每个分块的起始位置 (pos)，偏移量 (ofst)，以及指纹值 (fgpt)，并给每个分块分配一个全局唯一标识符 (uuid)，并将信息储存；将基准文件表示成 (uuid1，uuid2，uuid3，……，uuidN) 的向量形式；

c)、同样方式该处理分组中的其他文件，先将文件逻辑上进行分块，计算文件的每个分块指纹值 (fgpt)，与之前储存的基准文件的指纹值 (fgpt) 比较，判断该分块是否已经存在：如果存在则不再处理，将该分块用基准文件中的对应分块表示；不存在则存储这个分块的起始位置 (pos)，偏移量 (ofst)，以及指纹值 (fgpt)，并分配一个全局唯一标识符 (uuid')；最终将文件表示成向量形式；

d)、最终将文件分组的内容整合成单个连续存储的文件加若干个差异数据块的形式，并且保存每个文件的元信息以及所有分块的信息；所述差异数据块连续存储在一个文件之中。

7. 根据权利要求 6 所述的两阶段单实例去重数据备份方法，其特征在于，步骤 a) 中所述几个连续的字节所组成的字节串的位数由期望分块的长度决定，字节串的位数等于 $\log_2$ChunkSize 的向下取整，ChunkSize 为期望分块的大小，期望分块的大小为文件的尺寸的 1/100 ～ 1/1000。

两阶段单实例去重数据备份方法

技术领域

[0001] 本发明涉及计算机存储技术领域，尤其针对在客户端向服务器备份自身文件期间，提供消除冗余数据并且节省网络带宽的方法，用于提高存储设备的可用性。

背景技术

[0002] 在一般的客户端向服务器端保存自身文件的环境中，服务器端只是接受客户端上传的文件，并不对文件进行过多具体的检查，客户端也不对上传的文件有任何识别。一般应用的环境下，多客户端向服务器端上传文件，经常出现多个用户备份同一个文件，或者单个用户连续备份几个版本连续，内容相似的文件等情况。这种情况下会产生大量的冗余数据。

[0003] 为了处理这种问题，最常用的方法就是在服务器端实现文件级去重技术或者块级去重技术，这种两种方法会有很多的弊端，一是单纯使用文件级去重技术并不能达到很好的去重效果，尤其针对一些内容相似，差异较小的文件，不能检测出文件之间重复数据。二是对于块级去重技术，客户端需要上传大量元数据信息到服务器端，服务器端才能检测出重复数据，服务器端与客户端都需要实时处理这些数据，浪费时间与带宽，而且客户端的工作量很大。三是文件级去重检测是针对所有文件信息进行查询，并未考虑到各种文件相同时的必要条件，块级去重更是将所有文件进行统一的分块，然后使用建立元数据信息库进行查询，这样不仅会使元数据规模非常庞大，降低查询速率。四是传统的块级分块技术很容易将原本在同一个文件的中的连续数据块分散存储，还原时速度很慢。

发明内容

[0004] 本发明的目的在于提供了一种两阶段单实例数据备份方法，以解决上述技术问题。本发明结合文件级去重技术与块级去重技术，利用两种技术的优势，弥补各自的劣势，并且对其中传统的方法进行改进。

[0005] 为了实现上述目的，本发明采用如下技术方案：

[0006] 两阶段单实例去重数据备份方法，包括以下步骤：

[0007] 601)、客户端打开需要备份的文件，计算文件内容，产生文件元信息，检查本地数据库，判断该文件是否已经被客户端存储过：如果该文件已经被客户端存储过则转到步骤602)；如果对于客户端是没有备份过的新文件则转到步骤603)；

[0008] 602)、更新客户端的文件信息，将该文件元信息中的文件标示指向之前保存过的相同文件，备份流程结束；

[0009] 603)、将需要备份的文件元信息发送到服务器端，包括文件大小、类型、Hash值、备份时间信息；进一步判断该文件是否在服务器端被存储过；如果服务器端之前存储过相同的文件则转到步骤604)；如果没有则接着判断此文件的大小是否符合标准：如果是小文件执行备份策略一，执行完成后转到步骤604)；如果是大文件执行备份策略二，执行完成后转到步骤604)；

[0010] 604)、服务器端创建此文件的链接，并且更新服务器端的链接表，将文件链接发送

给客户端进行存储，作为将来还原和删除文件的凭证；备份流程结束。

[0011] 本发明进一步的改进在于：所述小文件小于文件系统分配单元 100 倍；所述大文件大于或等于文件系统分配单元 100 倍。

[0012] 本发明进一步的改进在于：步骤 603）中执行备份策略一具体包括：

[0013] 701）、根据需要备份文件类型判断此类文件是否属于文本类文件：如果是文本类文件则进入到步骤 702）；如果不是文本类文件则进入到步骤 704）；

[0014] 702）、将该文件追加写入到易压缩文件存储区中的大文件之中，并且更新文件元信息，接着判断该大文件是否超过了标准尺寸：如果该文件尺寸超过了规定的尺寸则转入到步骤 703）；如果该文件尺寸没超过规定的尺寸则直接结束；

[0015] 703）、将达到规定尺寸的文件进行压缩后转入步骤 705）；

[0016] 704）、将该文件追加写入到不易压缩文件存储区中的大文件之中，并且更新文件元信息，接着判断该大文件是否超过了标准尺寸：如果该文件尺寸超过了规定的尺寸则转入到步骤 705）；如果该文件尺寸没超过规定的尺寸则直接结束；

[0017] 705）、根据当前时间创建一个新的空文件，用于存储新到来的文件。

[0018] 本发明进一步的改进在于：所述规定的尺寸为 64MB。

[0019] 本发明进一步的改进在于：步骤 603）中执行备份策略二具体包括：

[0020] 801）、根据文件的类型以及文件的 Hash 值，选择文件的目录：不同的文件类型存储于不同的一级目录，同类型文件中 Hash 值前 12 位（十六进制标示时，前三个字段）不同的文件存储于不同的二级目录；

[0021] 802）、将文件写入到步骤 801）指定的目录之中；

[0022] 803）、更新文件元信息，将文件的元信息写入到服务器端的数据库之中，流程结束。

[0023] 本发明进一步的改进在于：还包括服务器端对某类型的大文件进行第一次相似文件的归并过程：

[0024] 1400）、首先在已经按类型分类的文件中挑选一种文件作为备选文件；将该类文件的信息按照文件尺寸的顺序读入到一个数据库的表 t0 中；

[0025] 1401）、从表 t0 中找到文件尺寸的中位数 Median，以 Median 为参考，寻找文件大小在 0.909*Median ～ 1.1*Median 之间的文件的信息，如果不存在这样的文件信息，则转到 1405）步，如果存在这样的文件信息，则转到 1402）步；

[0026] 1402）、根据匹配字节串的特征值算法，计算这些文件的特征值，并根据设定的阈值，判断每个文件与尺寸为 Median 的文件之间特征值的差异是否在阈值以内，如果存在差异在阈值以内的文件，则转入到 1403）步，如果没有则转入到 1405）步；

[0027] 1403）、将步骤 1402）找到的差异在阈值以内的文件的信息存储在表 t2 中，并标示为第 i 组，并且计算文件的平均特征值以及平均尺寸，并且将这些信息以及组号存入到表 t3 之中；表 t0 中删除这组文件所对应的项；检测表 t0 中是否还有其他未处理的文件项，如果没有则转入到 1404）步，如果有则转入 1401）步；

[0028] 1404）、将表 t2 中的项，在文件信息总表中标示为已分类的文件，并且将表 t2 与表 t3 保存起来，过程完毕；

[0029] 1405）、将尺寸为 Median 的文件的数据项存入到表 t1 中，从表 t0 中删除此项，之

后判断 t0 表是否为空，如果为空则直接结束，如果不为空则转入到 1401）步；

[0030] 其中所述位置只特征向量中每个特征值的差异在 10% 以内；i 初始值为 0，每标示完一组自增 1。

[0031] 本发明进一步的改进在于：在进行完第一次相似文件归并过程之后，服务器端定期检查文件系统日志，将新到来的文件进行归并，新到来的文件包括以下两种情况：

[0032] 1）与之前的分组的文件相似，因此首先计算新到来的文件的特征值与文件尺寸，并且将其与已知文件分组的平均特征值与平均尺寸进行比较，差异在阈值范围内，则划归到一组，并且重新调整该分组的平均特征值与平均尺寸；

[0033] 2）与之前未被分组的文件或者新到来的文件中的某些文件相似，因此第二步则将这些文件再一次的进行和所述第一次相似文件的归并过程相同的过程，创建新的文件分组。

[0034] 本发明进一步的改进在于：第 2）中情况具体的归并过程包括：

[0035] 1500）、首先通过文件系统将上次进行相似文件判断之后新到来的文件的信息存入到一个数据库表 t0 之中，并且计算这些文件的特征值，将特征值存入到表 t0 中；

[0036] 1501）、读取 t0 表中的第一个文件的文件尺寸 Size，并且查找平均文件尺寸在 0.909Size ～ 1.1Size 的文件分组，然后判断是否查找到此类分组；如果没有找到此类分组则继续判断表 t0 是否读完，如果否，则转到 1501）步读取表 t0 中的下一个文件；如果是，则继续判断表 t0 所有项是否已经删除了，如果删除完则直接结束，如果还有则转到 1504）步；如果找到此类分组，则转到 1502）步；

[0037] 1502）、将 t0 表中的第一个文件的特征值与步骤 1501）找到的每个分组的平均特征值进行比较，如果存在差异在阈值之内的分组，则将该文件划归到该分组之中；

[0038] 1503）、如果存在差异在阈值之内的分组，将 t0 表中的第一个文件的信息存入到已分类表 t2 中，并且重新计算其所属文件分组的平均特征值以及平均文件尺寸，更新分组信息表 t3，并且将该文件的信息从表 t0 中删除；判断表 t0 是否已经读完，如果否，则转到 1501）步读取表 t0 中的下一个文件，如果是，则继续判断表 t0 所有项是否已经删除了，如果删除完则直接结束，如果还有则转到 1504）步；

[0039] 1504）、将之前未被分组的文件的信息读入到表 t0 中，这样表 t0 中保存着未被分类的新文件以及之前未被分类的文件，并且保证表中的文件信息按文件的尺寸排序，之后按照与第一次相似文件的归并过程相同的过程处理。

[0040] 本发明进一步的改进在于：还包括针对每一分组进行差异去重的处理步骤，具体如下：

[0041] a）、选择几个连续的字节的取值为文件的分块边界，将某一分组中所有文件在逻辑上进行分块；

[0042] b）、首先计算基准文件的每个分块的起始位置（pos），偏移量（ofst），以及指纹值（fgpt），并给每个分块分配一个全局唯一标识符（uuid），并将信息储存；将基准文件表示成（uuid1，uuid2，uuid3，.....，uuidN）的向量形式；

[0043] c）、同样方式该处理分组中的其他文件，先将文件逻辑上进行分块，计算文件的每个分块指纹值（fgpt），与之前储存的基准文件的指纹值（fgpt）比较，判断该分块是否已经存在：如果存在则不再处理，将该分块用基准文件中的对应分块表示；不存在则存储这

个分块的起始位置(pos),偏移量(ofst),以及指纹值(fgpt),并分配一个全局唯一标识符(uuid');最终将文件表示成向量形式;

[0044] d)、最终将文件分组的内容整合成单个连续存储的文件加若干个差异数据块的形式,并且保存每个文件的元信息以及所有分块的信息;所述差异数据块连续存储在一个文件之中。

[0045] 本发明进一步的改进在于:步骤a)中所述几个连续的字节所组成的字节串的位数由期望分块的长度决定,字节串的位数等于 $\log_2$ChunkSize 的向下取整,ChunkSize 为期望分块的大小,期望分块的大小为文件的尺寸的 1/100 ～ 1/1000,字节串不一定要取完整的字节,字节串中最后一个字节可以只取其中几位,但寻找分块边界时进行逐字节比较,而不是逐位比较。

[0046] 本发明用于在客户端向服务器端备份文件期间,消除冗余数据和减少带宽消耗,包括如下的步骤:实现了两阶段的数据去重功能,通过客户端本地检查和服务器端检查的方式,进行在线文件级重复数据删除;对于服务器端的通过相似文件检查,将相似文件进行分组,检查冗余数据,进行差异存储,进行离线块级重复数据删除。客户端使用文件的特征信息与既有文件信息进行比对,在本地检查文件级重复数据。文件的特征信息包括:文件尺寸、文件类型、文件开头数据的指纹值、文件结尾数据指纹值、文件中间数据指纹值、文件全局采样数据的指纹值。客户端在检测不到重复文件信息时,再与服务器端进行交互操作,服务器端开始重复文件检测工作。服务器端使用客户端上传的文件特征信息,在服务器端与既有文件信息进行比对,检查文件级重复数据。服务器端检测不到重复数据,会通知客户端上传文件,记录该文件的信息,并登记客户端对此文件的引用,创建文件链接,发送给客户端,作为客户端对此文件拥有权限的凭证;检测到重复数据,则直接创建文件链接,发送给客户端。客户端的既有文件信息根据文件的全局采样数据的指纹值的不同字段,以 B+ 树结构进行组织。服务器端的既有文件信息,对于大文件,首先根据文件的扩展名中的字段,以 Hash 表或 B+ 结构进行组织进行组织;接着按照文件的尺寸,以 B+ 树结构进行组织。对于小文件,根据文件的全局采样数据的指纹值的不同字段,以 B+ 树结构进行组织。文件的链接信息使用文件的全局唯一标示符中的不同字段建立 B+ 树索引。文件的引用次数使用文件的全局唯一标示符中的不同字段建立 B+ 树索引。第一次相似文件检查工作,对全局所有较大文件进行检查判断;之后的相似文件检查工作会查询文件系统,将上次检查之后到来的文件进行相似文件检测,先检查新文件是否与既有相似文件分组中文件相似,再判断新文件是否与既有未被分组的文件相似。相似文件检查工作首先确定文件的范围只在同类型文件中进行检测,其次选择其中尺寸差异在阈值之内的文件作为候选文件。相似文件的去重工作,首先查找全局相似度最高的文件。全局相似度最高的文件是其特征值与相似文件分组特征值差异最小的文件。相似分组中的文件与全局相似度最高的文件进行比对,只保存差异数据块。

[0047] 相对于现有技术,本发明具有以下有益效果:本发明首先在客户端实现相同文件检测技术,检测客户端是否已经存在相同文件,如果存在则不再进行文件上传;如果不存在则进行第二步操作,将少量文件信息交由服务器端,服务器端进行相同文件检测,存在相同文件,则登记客户端对该文件的引用,不存在相同文件则再与客户端进行交互将文件上传至服务器端,服务器端将文件进行分类,并根据文件的特性,组织并保存该文件的元信息,

针对文件尺寸与类型，采用不同的策略进行存储。在服务器端空闲的时候，服务器端对文件进行相似文件检测，将文件进行分类，对分类后的文件结合块级去重技术，存储文件的差异部分，消除冗余部分。根据本发明，可以有效的降低带宽消耗，降低数据冗余度，提高设备的可用程度。有效地利用服务器的空闲时间，避免大量计算操作同时产生。

附图说明

[0048] 图 1 是本发明所描述方法的部署环境。

[0049] 图 2 是本发明所描述方法的整体架构图。

[0050] 图 3 是本发明所描述方法的客户端的架构视图。

[0051] 图 4 是本发明所描述方法的服务器端的架构视图。

[0052] 图 5 是本发明所描述方法的后台服务系统的架构视图。

[0053] 图 6 是本发明所描述方法的备份整体流程图。

[0054] 图 7 是本发明所描述方法的备份策略一的流程图。

[0055] 图 8 是本发明所描述方法的备份策略二的流程图。

[0056] 图 9 是本发明所描述方法的小文件的组织图。

[0057] 图 10 是本发明所描述方法的大文件的组织图。

[0058] 图 11 是本发明所描述方法的常见文件扩展名的哈希映射方法。

[0059] 图 12 是本发明所描述方法的不常见文件扩展名的 B+ 树组织方法。

[0060] 图 13 是本发明所描述方法的大文件的元数据组织方法。

[0061] 图 14 是本发明所描述方法的小文件的元数据组织方法。

[0062] 图 15 是本发明所描述方法的大文件的链接信息的组织方法。

[0063] 图 16 是本发明所描述方法的小文件的链接信息的哈希表。

[0064] 图 17 是本发明所描述方法的第一次进行相似文件的分类过程。

[0065] 图 18 是本发明所描述方法的新文件到来后的相似文件的分类过程。

[0066] 图 19 是本发明所描述方法的相似文件的去重过程。

[0067] 图 20 是本发明所描述方法的相似文件分组中文件的还原过程。

具体实施方式

[0068] 图 1 所示为本方法的部署实施环境，首先该方法部署环境是 C/S 结构，包括客户端和服务器端，在客户端保存着本地日志，日志中记录着用户曾经保存过的文件的信息和备份任务的信息。客户端通过网络与服务器端进行交互。服务器端包括备份服务器与后台处理系统，备份服务器将备份文件的内容保存在存储介质之中，将备份文件元信息保存入元数据文件之中。而后台处理系统在备份服务器工作任务轻闲或者无任务时，执行对文件进行相似文件分类与去重的操作，对文件进行二次去重。

[0069] 图 2 所示为本方法的整体架构图，包括客户端、备份服务器端、以及后台处理系统三部分，客户端将本地文件进行处理。并将文件元信息与新文件交由到备份服务器端，备份服务器端将文件保存至服务器端的文件系统之中，并且更新文件的元数据，而后台处理系统不定期的从服务器端的文件系统提取元数据与各类文件进行处理。

[0070] 图 3 所示为客户端的架构视图，客户端可以单独完成本地备份，或者与服务器端

进行交互完成文件的远程备份。也可以完成用户提交的还原和删除的文件的操作。对数据进行文件级的全局去重。其中各部分主要完成的工作是：

[0071] 1. 客户端系统界面：

[0072] 客户端的系统界面主要包括三个部分：备份界面、还原与删除界面、请求操作队列。

[0073] 备份界面的主要功能就是方便用户进行操作，提供给用户良好方便的接口，备份界面可以接受用户的选择的文件，将其放入待处理的队列，等待处理，在任务完成后，通知用户操作完成。还原与删除界面的主要功能是接收用户对已备份文件的还原或者删除的操作，将其放入待处理的队列，等待处理，任务完成后，通知用户。请求操作队列是将用户提出的备份、还原或者删除的请求加入到队列之中，等待调用相应的处理逻辑来完成对应的操作。

[0074] 2. 上层软件调用接口：

[0075] 这部分主要是与上层封装的备份还原软件进行交互，可以使上层软件以任务为单位进行备份、删除或者还原操作。主要包括删除任务接口、备份任务接口、还原任务接口、任务处理队列、任务解析器、请求操作队列。

[0076] 删除任务接口、还原任务接口、备份任务接口主要是接受上层软件的调用，接受删除、还原或者备份任务，将任务信息放入到任务队列中等待处理。任务队列主要是将上层软件需要完成的任务暂存在队列之中，等待任务解析器来处理。任务解析器主要是将上层的备份、删除或者还原任务解析成一系列对文件的操作，放入到请求操作队列。请求操作队列主要是接受来自任务解析器的添加的对某个文件的备份、删除或者还原请求操作，调用相应的处理逻辑进行处理。

[0077] 3. 客户端控制台：

[0078] 客户端控制台是与用户交互的直接平台，提供与用户交互的界面，接受用户的请求，同时也是实现在线重复数据删除的关键部分，为用户提供备份、还原与删除的功能。主要包含命令解析器、多线程处理队列、信息汇报三个子模块。

[0079] 命令解析器主要是不断将前面两个模块中的请求操作的队列中的请求，解析成必要的参数，将其交付到多线程处理队列中，进行处理。多线程处理队列主要是搭建一个固定尺寸的处理线程池来和一个阻塞队列来分别接受和完成相应的请求操作。线程接收参数后，调用相应的处理逻辑来进行处理。信息汇报模块主要是在线程完成任务后，在用户界面上提示用户，汇报请求的处理情况。

[0080] 4. 文件备份逻辑模块：

[0081] 文件备份逻辑模块主要包含文件信息处理模块、信息检验与提取模块与文件对象处理模块。分别来实现在备份文件时所需要的各种操作。

[0082] 文件信息处理模块主要是将用户请求进行解析，调用相应的处理模块计算文件的元信息，将各种文件信息进行组合，以便交付到服务器端进行下一步处理。在服务器端处理好后会将文件的信息存储到日志文件之中。信息检验与提取模块是将处理和组合后的文件信息，结合已经备份过的文件信息进行匹配处理，包括针对文件信息将文件进行分类，验证文件的合法性。文件对象处理模块主要是在需要传递文件的时候，针对文件对象进行处理，将文件对象交付给文件传输模块进行发送。

[0083] 5. 局部去重引擎：

[0084] 局部去重引擎在文件备份时进行使用，将新文件的 Hash 值等元信息，与已备份文件的文件信息进行比对，查看是否有相同的项，如果有相同项，则通知客户端控制台与服务器端进行交互，建立需要备份文件的链接。如果没有相同的项，则通知客户端控制器与服务器交互，在全局文件表中查找是否有相同文件。

[0085] 6. 文件还原逻辑模块：

[0086] 文件还原逻辑模块主要是接受用户还原文件的请求，为用户还原文件。主要包括文件信息处理模块，还原信息校验模块以及文件对象处理模块。

[0087] 文件信息处理模块主要是将用户请求解析，在提取已经备份文件的信息，进行组合，等待将其发送到服务器端进行处理。还原信息校验模块主要将用户的信息在日志文件进行校验，查看还原信息是否合法，如果合法，则通知上一模块可以交付给服务器端进行处理。文件对象处理模块主要在服务器端通知可以进行文件还原操作时，通知文件系统为文件预留空间，创建文件对象，接收文件。

[0088] 7. 文件删除逻辑模块：

[0089] 文件删除逻辑模块主要包括文件信息处理子模块和删除信息校验子模块两个部分，来对用户的删除请求进行处理。文件信息处理模块主要是将用户的删除文件的请求解析，在提取已经备份文件的信息，进行组合，将其发送到服务器端进行处理。删除信息校验模块主要功能是提取文件信息，检查文件删除的信息是否合法，如果合法通知上一模块可以将其发送给服务器端。

[0090] 8. 文件元信息产生器：

[0091] 文件元信息产生器是在备份模块调用时，将文件对象的信息计算出来交付给备份模块。主要包括 Hash 产生器、文件类型校验模块、时间产生器。

[0092] Hash 产生器主要针对给定的文件，分别计算文件内容的 Hash 值，将其交付给备份模块，以便在客户端进行重复检查，或在服务器端进行全局去重。文件类型校验模块主要是针对文件对象，检查其文件类型及其文件大小，提示备份模块使用不用的备份策略。

[0093] 9. 数据库管理模块：

[0094] 该方法使用 NOSQL 数据库来处理客户端的本地日志信息，这部分主要是针对数据库编写查询信息、添加信息、删除信息以及遍历信息的代码，方便其他模块调用处理。

[0095] 信息查询模块用于查询已备份文件的元数据信息，交付给备份、还原、删除等模块，进行信息校验，和组合信息等操作。信息添加模块在新文件到来时，需要调用此模块将文件信息写入到本地的数据库之中，以便以后使用。信息删除模块在用户需要删除文件的时候，删除模块在确认操作合法后，此模块将已备份文件信息中关于该文件的信息删除。信息遍历模块主要是将文件的信息全部显示在界面之中，供用户查询。

[0096] 10. 通信模块：

[0097] 通信模块在客户端处理用户的请求时，有时需要与服务器端进行通信。这部分主要包括传输消息模块和接收消息模块。传输消息模块将用户请求的任务交到服务器端进行下一步的处理。接收消息模块获取服务器端的处理的结果，以便进行下一步操作，或者通知用户执行情况。

[0098] 11. 文件传输模块：

[0099] 在客户端与服务器端进行通信后，有些需要备份的文件并不重复，需要发送给服务器端，而确认合法的还原请求，需要接收文件。该部分包括发送文件模块以及接收文件模块。

[0100] 发送文件模块对于不重复的备份文件，发送文件模块需要将文件传送给服务器端进行接收，储存到相应的位置上。接收文件模块在服务器端确认合法的还原请求后，确认文件需要还原到的位置，文件名，所需要的空间后，接收文件内容进行还原。

[0101] 图 4 所示为服务器端的架构视图，服务器端主要完成对用户上传文件的保存，还原文件以及删除过期文件等操作，另外维护文件的全局信息，对数据进行全局文件级别的去重。

[0102] 1. 服务器端控制台：

[0103] 服务器端控制台主要包括命令解析器、多进程处理队列。

[0104] 命令解析器不断地从服务器端的通信模块中提取用户的请求，进行解析，解析完成后将其交付给控制台中多线程处理队列进行处理。

[0105] 服务器端的多线程处理队列与客户端的多线程处理队列的功能类似，是完成用户请求的后期操作，当客户端将文件的信息和请求发送过来后，服务器端主要是将文件信息与全局的数据信息进行匹配，匹配成功后，完成相应的备份、还原或者删除的请求。

[0106] 2. 文件备份逻辑模块：

[0107] 服务器端的文件备份逻辑模块主要是完成后期的备份操作，主要包括文件信息处理模块、链接信息处理模块、文件对象处理模块。

[0108] 服务器端的文件信息处理是将用户发送来的元数据信息，进行重新组合，在全局进行查找是否有匹配的数据。链接信息处理模块：无论文件是否重复，在备份的最后阶段，都要制作一个关于此文件的全局唯一的链接发送给客户端，同时这个链接也将保存到服务器端，这个链接将作为用户将来还原文件的凭证，防止违法操作。文件对象处理模块在文件需要传输到服务器端的时候，需要找到合适的位置，准备合适的空间来存储对应的文件。

[0109] 3. 全局去重引擎：

[0110] 全局去重引擎就是根据文件的 Hash 值、尺寸与类型，查询数据库中是否存在相同的信息。如果存在，则确认该文件为重复文件；否则认定该文件是新文件，需要用户将文件发送过来。

[0111] 4. 元信息模块：

[0112] 元信息模块有效的将文件信息组织起来，将其储存在数据库系统之中，并且在备份、还原以及删除文件的时候，调用出文件的元信息进行匹配，协助完成操作。针对每个文件储存其 Hash 值、类型、大小等信息。

[0113] 5. 链接信息模块：

[0114] 为了有效的为用户还原文件，链接信息模块将文件的链接信息有效地组织起来，将链接信息储存在数据库中，按每个文件的全局唯一标识符来组织文件的链接，针对每个文件建立针对该文件的表或者文档，保存针对该文件的链接号。

[0115] 6. 文件目录组织模块：

[0116] 当需要将客户端的文件保存到服务器端的时候，文件目录组织模块需要针对文件的特性，为其创建一系列文件夹，对其进行分类存储。

[0117] 首先根据文件的尺寸将文件分为大文件与小文件，大文件对空间浪费少可以进行单独存储，小文件需要进行拼接连续存储，避免空间浪费。之后针对大文件，按照文件扩展名与指纹值，建立二级目录，对于小文件，按照文件的到来的时间，将文件存储到对应的文件夹，并且将各类小文件进行组合存储，从而避免空间浪费。

[0118] 7. 文件还原逻辑模块：

[0119] 文件还原逻辑模块主要包括文件信息处理模块、还原信息校验模块、文件对象处理模块。文件信息处理模块将客户端发送过来的文件信息进行处理，重新组合，之后进行校验查找是否有相应的信息匹配，如果匹配，通知服务器端可以进行还原。还原信息校验模块是针对处理后的信息，进行全局校验，确定还原信息的正确性。文件对象处理模块主要是做传输文件前的准备，提取文件对象，准备传输文件。如果文件经过了二次去重，则交由后台处理系统处理。

[0120] 8. 文件删除逻辑模块：

[0121] 文件的删除逻辑模块主要包括文件信息处理模块、文件对象处理模块、删除处理模块三个部分。主要是完成后期的删除操作。

[0122] 文件信息处理模块将客户端发送过来的文件信息进行处理，重新组合，之后进行校验查找是否有相应的信息匹配，如果匹配，则确认可以进行删除。删除信息校验模块主要是校验用户发送过来的文件链接与文件的元数据信息是否合法，如果合法则通知服务器端可以删除文件。删除处理模块的主要工作就是在确认需要删除操作的时候执行任务，首先删除用户的文件链接在链接表中对应的内容，然后查看链接表中是否还有针对该文件的链接，如果还有操作结束，如果没有，则继续将文件表中关于该文件的内容删除，然后再删除文件的实体内容。如果文件经过了二次去重，则交由后台处理系统处理。

[0123] 9. 数据库管理模块：

[0124] 这里的数据库管理模块与客户端数据库管理模块的功能类似，保存着全局的文件信息以及全局的链接信息，其他模块在进行各自操作时，要访问数据库，进行查询，或者更新其中的信息，数据库管理模块主要提供访问数据库的接口，方面其他模块调用。

[0125] 10. 多线程通信模块：

[0126] 多线程通信模块主要是不断地循环监听通信端口，不断地接收每个客户端发送过来的信息，将其交付给服务器端的控制台，另外返回给每个客户端任务处理的情况。主要包括传输消息，接收消息，监听端口三个模块。传输消息模块主要是将处理用户发送过来的请求的结果返回给客户端，或者通知客户端下一步需要进行什么操作。接收消息模块主要是接收每个客户端的备份、还原或者删除文件的请求，将其不断地提交到服务台进行处理。监听端口模块主要是不断地循环监听端口，将每个客户端发送过来的请求添加到队列之中，供接收消息模块不断提取和处理。

[0127] 11. 多线程文件传输模块：

[0128] 这个模块与每个客户端的传输文件模块相对应，主要有传输文件和接收文件两个功能。

[0129] 传输文件模块在还原信息确认正确后，需要发送文件到客户端，这时线程池分配一个线程，将文件发送到客户端。接收文件模块在客户端需要备份的文件确认为新文件之后，线程池分配一个线程，接收由客户端发送过来的文件。

[0130] 图 5 所示为单实例备份的后台处理系统，主要是对经过文件级别去重的数据进行第二阶段的去重。这部分系统会根据文件的现有元信息，选择其中有去重价值的数据，进行分类，并采用不同的策略，对数据进行二次去重。

[0131] 1. 文件扫描器：

[0132] 文件扫描器每次在指定范围内，针对新文件进行扫描，与服务器端数据库管理模块交互，将新文件的信息交给相似文件分类器，进行分类处理。每次记录上次扫描的位置，以及扫描过的文件，保证不重复扫描。

[0133] 2. 相似文件分类器：

[0134] 相似文件分类器是将文件之中大小相近，类型相同等的文件，再次进行相似 Hash 的计算，以多个相似 Hash 值组合成文件的特征值，比较各文件之间的特征值，判断差异是否在阈值之内，从而将其中文件内容相近文件进行归并，然后再将相似文件交付给差异去重模块，让这个模块再对文件进行分组去重。

[0135] 3. 差异去重模块：

[0136] 差异去重模块是针对一组相似文件，选择出其中全局相似度最大的文件（重复数据占据文件比例最大的文件）作为标准文件，其他文件与其比较，将其中的差异部分进行单独且连续存储，标准文件不进行分块处理，其他文件以标示其所含数据位置的方式将元信息储存在数据库之中，这样在尽可能保证数据连续存储的情况下，对文件进行去重。

[0137] 4. 文件元信息模块：

[0138] 由于文件进行了二次去重处理，因此又将产生一系列的文件元信息，针对差异去重会按相似文件分组为单位来存储文件的信息，标示其全局相似度最高的文件，标示差异数据块对应的文件 ID，并且其在原文件之中的位置，针对块级去重的文件，也会表示其分组，以及文件对应分块，保证无错还原。

[0139] 5. 数据库管理模块：

[0140] 将文件元信息模块产生的文件元数据进行有效地组织并永久的储存起来，以便将来其他模块调用文件元信息，方便再次进行相似文件比较时，判断新文件与旧有文件是否相似，以及在还原与删除文件时，提供必要的信息，方便其进行有效的无错的还原。

[0141] 6. 还原模块：

[0142] 再进行完二次去重之后，文件的格式已经发生可很大的变化，这时如果需要还原文件，则需要查询文件的元信息，查询文件对应内容存储的位置，文件的处理方式，然后进行逆向操作，对于进行差异去重的文件，将差异块与全局相似度最高文件的内容提取出来，然后再根据文件的元信息，将文件内容重新组合进行还原。

[0143] 7. 删除模块：

[0144] 同样由于文件的格式发生了变化，这时对于需要删除的文件，将数据库中关于这个文件的各部分引用次数减 1，如果该某一部分的内容不再有文件进行引用，则将此内容进行删除处理。

[0145] 图 6 所示为文件备份过程的整体流程图，下面开始用在图中的数字标示来介绍具体的流程：

[0146] 601、客户端打开需要备份的文件，计算文件内容，产生文件元信息，检查本地数据库，判断该文件是否已经被客户端存储过：

[0147] a) 如果该文件已经被存储过则转到步骤 602。

[0148] b) 如果对于客户端是没有备份过的新文件则进行到步骤 603。

[0149] 602、更新客户端的文件信息，将该文件元信息中的文件标示指向之前保存过的相同文件，流程结束。

[0150] 603、将文件元信息发送到服务器端，包括文件大小、类型、Hash 值、备份时间信息；进一步判断该文件是否在服务器端被存储过。如果服务器端之前存储过相同的文件则转到步骤 604，如果没有则接着判断此文件的大小是否符合标准：

[0151] c) 如果是小文件（小于文件系统分配单元 100 倍）执行备份策略一，执行完成后转到步骤 604。

[0152] d) 如果是大文件（大于或等于文件系统分配单元 100 倍）执行备份策略二，执行完成后转到步骤 604。

[0153] 604、服务器端创建此文件的链接，并且更新服务器端的链接表，将文件链接发送给客户端进行存储，作为将来还原和删除文件的凭证；流程结束。

[0154] 图 7 所示为存储各种小文件使用备份策略一：

[0155] 701、根据文件类型判断此类文件是否属于文本类文件：

[0156] a) 如果是文本类文件则进入到步骤 702。

[0157] b) 如果不是文本类文件则进入到步骤 704。

[0158] 702、将该文件追加写入到易压缩文件存储区中的大文件之中，并且更新文件元信息，接着判断该大文件是否超过了标准尺寸：

[0159] a) 如果大文件尺寸超过了规定的尺寸（64MB）则转入到步骤 703。

[0160] b) 如果大文件尺寸没超过规定的尺寸则直接结束。

[0161] 703、将达到规定尺寸的文件进行压缩后转入步骤 705。

[0162] 704、将该文件追加写入到不易压缩文件存储区中的大文件之中，并且更新文件元信息，接着判断该大文件是否超过了标准尺寸：

[0163] a) 如果大文件尺寸超过了规定的尺寸则转入到步骤 705。

[0164] b) 如果大文件尺寸没超过规定的尺寸则直接结束。

[0165] 705、根据当前时间创建一个新的空文件，用于存储新到来的文件。

[0166] 图 8 所示存储一般大文件使用备份策略二：

[0167] 801、参阅图 10，根据文件的类型以及文件的 Hash 值，选择文件的目录；不同的文件类型存储于不同的一级目录，同类型文件中 Hash 值前 12 位不同的文件存储于不同的二级目录。

[0168] 802、将文件写入到步骤 801 指定的目录之中。

[0169] 803、更新文件元信息，将文件的元信息写入到服务器端的数据库之中，流程结束。

[0170] 下面开始介绍关于文件的元信息，这是用于实现重复数据删除至关重要的部分，下面以表格的形式对客户端与服务器端存储的元数据进行说明。

[0171] 表 1 客户端保存的关于备份任务的信息

[0172]

名称	示例	说明
备份任务号	Task01	备份任务的号码，可以自动命名，也可以随机分配
文件 Hash 组	F1B8F0C69EE32C47D225AAA74EBB9F20 DBFBF1682FD01E14EFF18ADB5F2889C8 ……	备份任务中所有文件的 Hash 值，用于删除任务和还原任务时使用。

[0173] 表 2 客户端保存的关于备份文件的信息

[0174]

名称	示例	说明
文件保存的时间	2011.12.28.16:15:38:031	备份文件的时间
Hash	F1B8F0C69EE32C47D225AAA74EBB9F20	文件内容的 Hash 值，作为文件的特征指纹值（当文件在客户端被重复备份时，该项变为一个针对此文件的链接号）
链接号	75a4c3c5-5bbe-4f1a-9e53-37aba2226cd9	客户端与服务器端进行匹配的标示号（当文件在客户端被重复备份时，该项此文件的 Hash 值）
原文件名	ks(4).rar	用户备份文件时其原文件名
原文件路径	D:\SIS\files\	用户备份文件时其原文件路径
文件大小	1979591	备份文件的大小以字节为单位
文件类型	rar	备份文件的类型
首次备份	True	客户端是否首次备份该文件

[0175] 表 3 服户端保存的关于备份的大文件的信息

[0176]

名称	示例	说明
文件类型	rar	备份文件的类型
文件大小	1979591	备份文件的大小以字节为单位
Hash	F1B8F0C69EE32C47D225AAA74EBB9F20	文件内容的 Hash 值，作为文件的特征指纹值
引用次数	24	文件被多少客户端进行引用次数
二次处理标示	False	文档是否经过了二次处理，如果经过了二次处理需要进行标示，以便还原和删除文件时，交付给后台处理模块进行处理。

[0177] 表 4 服务器端保存的关于大文件的链接信息

[0178]

名称	示例	说明
Hash	F1B8F0C69EE32C47D225AAA74EBB9F20	文件内容的 Hash 值，作为文件的特征指纹值

[0179]

文件的链接号	330af396-8654-47cf-b71a-fea6be11c724	文件的链接号，具有全局唯一的标示性

[0180] 表 5 服务器端保存的关于备份的小文件的信息

[0181]

名称	示例	说明
文件保存的时间	2012.06.21.21:42:44:362	备份文件的时间
所属文件标示	12d4b514-dfea-4192-baf5-30c47816ae6d	小文件是被统一存储在一个连续的大文件之中，此为该文件的文件名
文件位置	10486	小文件包含在大文件之中的起始位置
文件偏移量（大小）	5575	小文件在大文件之中的偏移位置，即是文件大小。
文件类型	zip	备份文件的类型
Hash	9CD919E20CF580C31A78C88902A83882	备份文件整体的 Hash 值
引用次数	24	文件被多少客户端进行引用
二次处理标示	False	文档是否经过了二次处理，即是否经过了压缩处理

[0182] 表 6 服务器端保存的关于小文件的链接信息

[0183]

名称	示例	说明
Hash	9CD919E20CF580C31A78C88902A83882	备份文件整体的 Hash 值
文件的链接号	330af396-8654-47cf-b71a-fea6be11c724	文件的链接号，具有全局唯一的标示性

[0184] 图 9 所示，为小文件的目录组织架构，首先对文件类型的特性进行分类，如果文件属于文本类文件则划入到易压缩文件种类之中；如果属于其它类型文件则划入到不易的压缩文件种类之中。这样做的原因是，文件备份之后如果长期不用，需要对文件进行归档，有些类型的文件有很大的压缩价值，而有些文件本身就经过特殊算法的压缩，如 MP3、RMVB 等各种音视频压缩，再进行压缩不但浪费时间，而且效果并不理想，往往原文件尺寸相差无几。之后根据文件的到来的时间组织若干级目录，将文件按到来的先后顺序，依次写入到一个数据块之中，当写满一个数据块之后，再创建一个空白的数据块，继续写入进去。这样设计的原因是由于每次备份请求都是以任务为单位进行的，每次都需要备份一组文件，而还原文件也是以任务为单位进行还原，当需要进行还原文件时，应该尽量保持磁盘进行连续读写，避免随机 IO，所以应该尽量将每次备份的文件进行连续存储，而每次备份任务都是在一个连续的时间内完成的，所以按照文件到来的先后顺序将文件写入到数据块。

[0185] 图 10 所示为对大文件的目录组织形式，首先根据文件的扩展名对文件进行分类，建立不同的一级目录，因为在第二阶段去重的时候，需要对各种类型的文件进行相似归类，每组相似文件需要在相同类型的文件中候选，将其存储在同一个文件夹中，保证快速查找此类文件。接着按照文件的 Hash 值的前 3 个字段来组织第二级的目录，这样最多可以组织 4096 个文件夹，文件也会比较均匀的分布在每个文件夹之中。

[0186] 另外需要对元数据建立多级索引机制。

[0187] 如图 11 所示，使用 Hash 表进行组织，对扩展名进行 Hash 计算。直接映射到其位置，这样不但占用空间少，而且查找速度快。Hash 表中保存着该文件类型的扩展名，以及其元信息的位置。对于不常见文件类型的扩展名，虽然这类的文件所包含的数据量占据的比例不大，但是不常见的文件类型的非常的多，理论上是没有上限的，一般一台 PC 机中约有几百种文件类型的文件，而且会经常有新的文件类型出现，因此如图 12 所示，使用一个二级 Hash 映射结构对文件的元信息进行组织，第一个表保存着文件扩展名 Hash 值的前两个字段，范围从 00 ～ FF，共有 256 种，这个表是固定的。第一级 Hash 表，key 项 00 ～ FF 中每项对应着一个指针，指向第二级 Hash 表，这个 Hash 表的 key 项是是扩展名 Hash 值的后面的整个字段，对应着该类型文件的元数据的存储区域，由于实际情况中扩展名出现的种类只有几千个，所以二级索引足够快速找到对应项。图 11 与图 12 中的 Hash 表占用内存较小，因此常驻在内存之中，方便快速查询。

[0188] 图 13 所示，对大文件的元信息的组织形式，先以 B+ 树的形式对文件的尺寸信息进行组织，由于绝大部分文件会和文件尺寸值一一对应，因此在文件尺寸后面，直接加入文件的元信息，以便在这一步就进行相同文件的比较，减少再次索引的操作。虽然几率很低，但仍可能出现不多的尺寸不同，但内容不同的文件的情况，因此当出现这种情况的时候，只需要将其在 B+ 树的数据结构中并列存储即可。

[0189] 图 14 所示，对小文件的元信息的组织形式，首先使用文件全局 Hash 值的前 12 位（十六进制表示中前 3 个字段）组织第一级 Hash 表，然后使用文件 Hash 值的第 4 ～ 5 字段组织第二级 Hash 表，第三级是文件的全部 Hash 值。

[0190] 图 15 所示，是对大文件的链接信息的组织形式，每个文件链接代表一个客户端对这个文件进行引用，链接号具有全局唯一性。在图中首先使用文件的 Hash 中的部分字符来构建一个二级索引，第一级索引使用的是文件的 Hash 值的前 2 个字段作为关键码，后面紧接着指向下一级 Hash 表的指针；第二级 Hash 表使用的是 Hash 值的第 3 ～ 4 字段作为关键码，后面紧接着存储两个指针，指针 P 指向一个 Hash 表，这个表保存着文件 Hash 值以这 4 个字段开头的文件的对应的所有链接号，指针 P’ 指向 Hash 值以这 4 个字段开头的文件的对应部分文件信息，包括文件的 Hash 值，文件的引用次数，文件的类型以及文件的尺寸。大文件的元信息是以文件类型和文件尺寸来进行组织的，保存文件的类型和尺寸是为了在文件将被删除的时候，方便查找到之前保存的文件的元信息，并将其进行删除。另外文件的存储路径是根据文件的类型与 Hash 值进行存储的，当需要进行文件的还原时，可以根据这两个值推算出文件的存储路径。二次处理标示是为了判断文件是否经过二次去重处理，如果经过了二次去重处理，则通知后台处理模块来对文件进行还原或者删除，另外由于每个文件可能有多个文件链接，所以文件链接的数量可能远远大于文件的总数量，所以，系统设计一个可选的选项，当文件的链接数量远远大于文件的数量时，如图 15 中的虚线框，使用链

接号的某几位字段(一般 1 ～ 2 字段就已经足够),对文件的链接再做一次 Hash 映射,保证最后一级的 Hash 表的尺寸不至于太大,方便查找。由于 Hash 值的计算方法可以使不同数据比较平均的映射到一个区间之中,链接号是根据时间产生的,各位取值也很随机。所以经过多级映射表的组织,最后一级的 Hash 表的尺寸会比较接近,不会出现某些 Hash 表极大,某些极小的情况。另外,当文件经过第二阶段处理的时候,将文件的保存路径改变为文件所在相似分组的分组号。

[0191] 小文件的文件链接信息的组织形式与大文件类似,但也有不同之处,如图 16 所示在最后一级以文件的 Hash 值组织的 Hash 表保存的链接信息会有差异。表中保存的是文件 Hash 值,引用次数,保存时间,所属文件名(小文件所属大文件的名称),文件位置(小文件在大文件中的起始位置),文件尺寸。文件的 Hash 值是为了在文件被删除时,将文件的元信息也进行删除。文件名与保存时间可以推算出文件的保存路径,而直接保存文件路径长度比较大,二次处理表示是判断单文件时候已经压缩处理,如果经过了压缩处理,则在还原或者删除的时候先进行解压,在执行还原或者删除操作。

[0192] 因为备份还原系统,备份操作远远比还原操作多,极端情况下就是归档系统,交由到服务器的文件很少进行还原操作,除非客户端的文件出现了损坏的情况,因此文件的链接信息不必常驻内存,具体调度可以直接交由 NOSQL 数据库处理,而且文件的链接信息的数目可能是文件信息数目的几十倍,全部导入到内存之中可能会导致溢出。服务器端在执行备份、还原和删除请求时,优先执行备份操作与还原操作,删除请求可以先记录下来等到服务器空闲的时候再进行删除操作。

[0193] 图 11 ～图 16 所示的数据结构可以使用 NOSQL 数据库进行组织,设定数据库参数就能将文件或按照 Hash 表,或按照 B+ 树来组织元数据。

[0194] 图 17 所示为对某类型的大文件进行第一次相似文件的归并过程 :

[0195] 1400、首先在已经按类型分类的文件中挑选一种文件作为备选文件。将该类文件的信息按照文件尺寸的顺序读入到一个数据库的表 t0 中,(由于之前文件的信息是按照文件的尺寸以 B+ 树的形式进行存储,所以可以很快的顺序读出文件信息)。

[0196] 1401、从表中找到文件尺寸的中位数 Median,以 Median 为参考,寻找文件大小在 0.909*Median ～ 1.1*Median 之间的文件的信息,如果不存在这样的文件信息,则转到 1405 步,如果存在这样的文件信息,则转到 1402 步。

[0197] 1402、根据匹配字节串的特征值算法,计算这些文件的特征值,并根据设定的阈值(特征向量中每个特征值的差异在 10% 以内),判断每个文件与尺寸为 Median 的文件之间特征值的差异是否在阈值以内,如果存在差异在阈值以内的文件,则转入到 1403 步,如果没有则转入到 1405 步。

[0198] 1403、将步骤 1402 找到的差异在阈值以内的文件的信息存储在表 t2 中,并标示为第 i 组(i 初始值为 0,每标示完一组自增 1),并且计算文件的平均特征值以及平均尺寸,并且将这些信息以及组号存入到表 t3 之中。表 t0 中删除这组文件所对应的项。检测表 t0 中是否还有其他未处理的文件项,如果没有则转入到 1404 步,如果有则转入 1401 步 ;

[0199] 1404、将表 t2 中的项,在文件信息总表中标示为已分类的文件,并且将表 t2 与表 t3 保存起来,过程完毕。

[0200] 1405、将尺寸为 Median 的文件的数据项存入到表 t1 中,从表 t0 中删除此项,之后

判断是否 t0 表是否为空，如果为空则直接结束，如果不为空则转入到 1401 步。

[0201] 在进行完第一次相似文件归并过程之后，系统定期检查文件系统日志，将新到来的文件进行归并，新到来的文件可能出现以下两种情况：

[0202] 1）与之前的分组的文件相似，因此首先计算新到来的文件的特征值与文件尺寸，并且将其与已知文件分组的平均特征值与平均尺寸进行比较，差异在阈值范围内，则划归到一组，并且重新调整该分组的平均特征值与平均尺寸。

[0203] 2）与之前未被分组的文件或者新到来的文件中的某些文件相似，因此第二步则将这些文件再一次的进行如图 18 所示的归并过程，创建新的文件分组。

[0204] 图 18 所示处理新文件到来后相似文件归并工作的整个过程：

[0205] 1500、首先通过文件系统将上次进行相似文件判断之后新到来的文件的信息存入到一个数据库表 t0 之中，并且计算这些文件的特征值，将特征值存入到表 t0 中。

[0206] 1501、读取表中的第一个文件的文件尺寸 Size，并且查找平均文件尺寸在 0.909Size～1.1Size 的文件分组，然后判断是否查找到此类分组。如果没有找到此类分组则继续判断表 t0 是否读完，如果否，则转到 1501 步读取表 t0 中的下一个文件；如果是，则继续判断表 t0 所有项是否已经删除了，如果删除完则直接结束，如果还有则转到 1504 步；如果找到此类分组，则转到 1502 步。

[0207] 1502、将 t0 表中的第一个文件的特征值与步骤 1501 找到的每个分组的平均特征值进行比较，如果存在差异在阈值之内的分组，则将该文件划归到该分组之中。

[0208] 1503、如果存在差异在阈值之内的分组，将 t0 表中的第一个文件的信息存入到已分类表 t2 中，并且重新计算其所属文件分组的平均特征值以及平均文件尺寸，更新分组信息表 t3，并且将该文件的信息从表 t0 中删除。判断表 t0 是否已经读完，如果否，则转到 1501 步读取表 t0 中的下一个文件，如果是，则继续判断表 t0 所有项是否已经删除了，如果删除完则直接结束，如果还有则转到 1504 步。

[0209] 1504、将之前未被分组的文件的信息读入到表 t0 中，这样表 t0 中保存着未被分类的新文件以及之前未被分类的文件，并且保证表中的文件信息按文件的尺寸排序，之后按照第一次相似文件归并的方式进行处理，在此不再赘述。

[0210] 图 19 所示为相似文件的差异去重方法，首先找到基准文件 file1（是其特征值与分组特征值最为接近的文件），图中黑色的标示块代表各个文件与基准文件的差异部分。差异去重的处理过程如下：

[0211] (1) 选择几个连续的字节的取值为文件的分块边界，将某一分组中所有文件在逻辑上进行分块。

[0212] (2) 首先计算基准文件的每个分块的起始位置(pos)，偏移量(ofst)，以及指纹值(fgpt)，并给每个分块分配一个全局唯一标识符(uuid)，并将信息储存。将基准文件表示成(uuid1，uuid2，uuid3，.....，uuidN)的向量形式。

[0213] (3) 同样方式处理分组中的其他文件，先将文件逻辑上进行分块，计算文件的每个分块指纹值(fgpt)，与之前储存的基准文件的指纹值(fgpt)比较，判断该分块是否已经存在，如果存在则不再处理，不存在则存储这个分块的起始位置(pos)，偏移量(ofst)，以及指纹值(fgpt)，并分配一个全局唯一标识符(uuid)。最终将文件表示成向量形式如(uuid1'，uuid2'，uuid3'，.....，uuidN')。

[0214] (4) 最终将文件分组的内容整合成单个连续存储的文件加若干个差异数据块的形式(差异数据块连续存储在一个文件之中),并且保存每个文件的元信息以及所有分块的信息。

[0215] 注 :

[0216] (1) 如果基准文件内部有重复数据块,文件的块信息不进行重复储存,文件实体不进行调整,在文件的向量中对应位置标识为首先出现的块,例如第 2 个块与第 4 个块内容相同,则表示成(uuid1,uuid2,uuid3,uuid2,.....,uuidN)。

[0217] (2) 文件在还原的时候 :对于基准文件则直接连续读出进行还原,对于其他有差异的文件,则如图 20 所示,将其文件的向量与基准文件的向量进行比较,相同的部分则从基准文件连续读出,不同的部分则从由差异数据块组成的文件中单独读取出来,最后拼接成最终的文件。

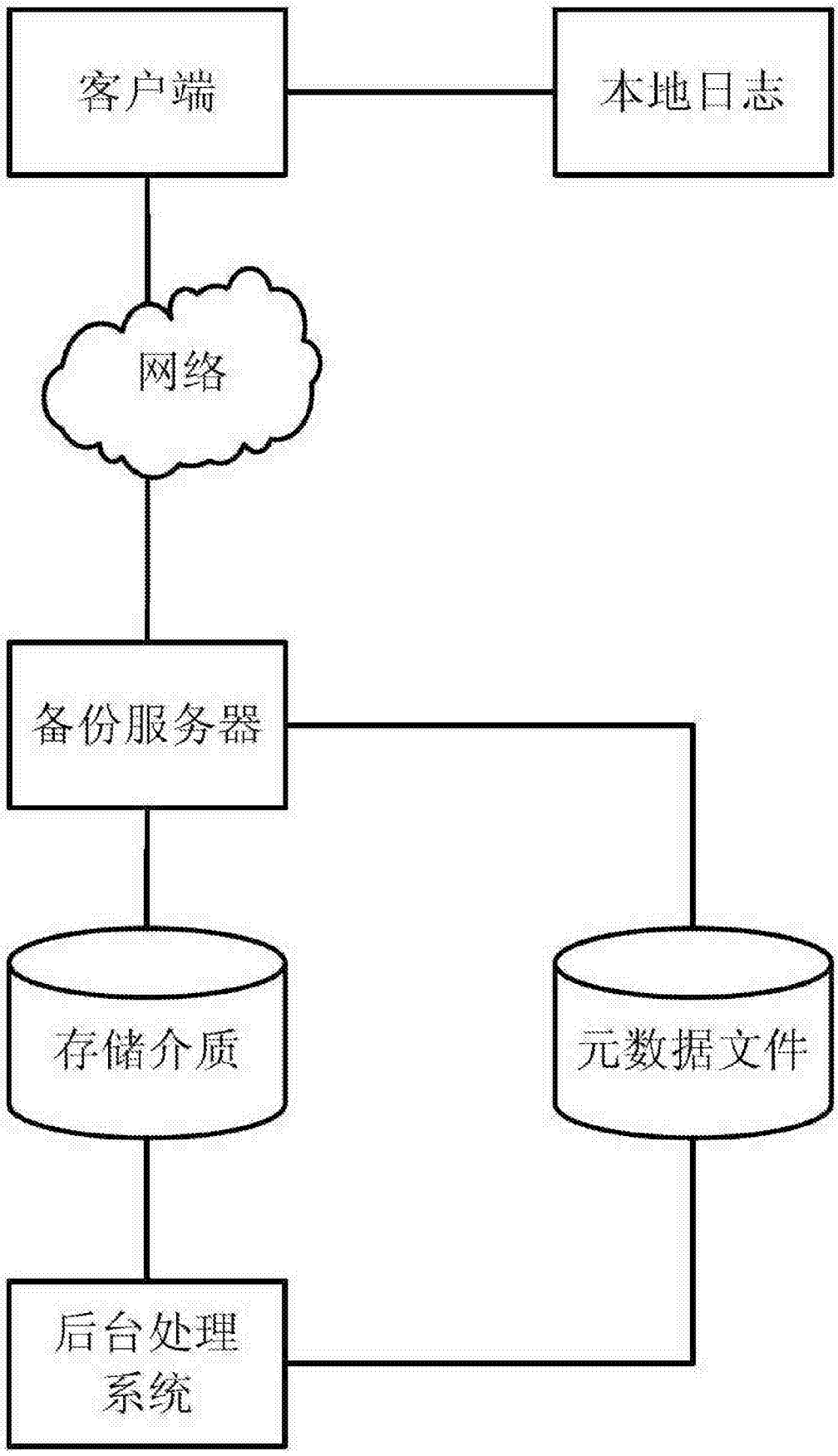


图 1

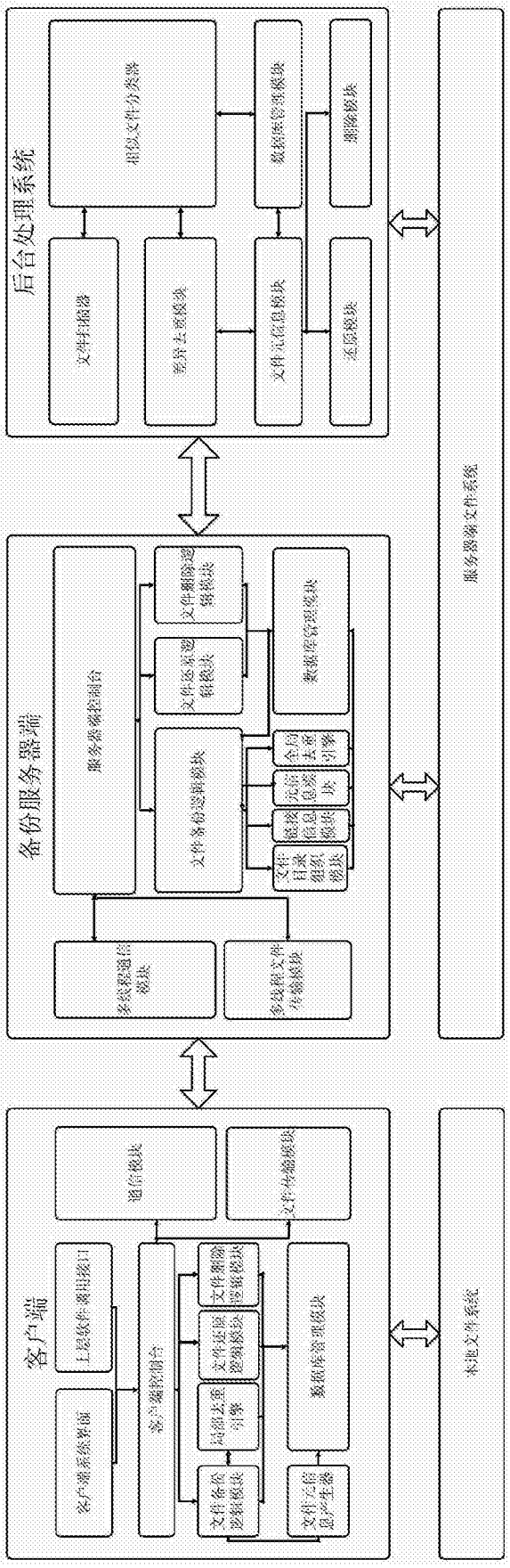

后台处理系统
用户文件分类器
数据库管理模块
删除模块
文件扫描器
数据主要模块
文件元信息模块
还原模块
服务器端文件系统
备份服务器端
服务器端控制台
文件删除逻辑模块
文件还原逻辑模块
数据库管理模块
文件备份逻辑模块
全局主密引擎
元信息模块
链接信息模块
文件目录组织模块
多线程通信模块
多线程文件传输模块
客户端
通信模块
文件传输模块
客户端系统界面
上层软件调用接口
客户端控制台
文件删除控制模块
文件还原逻辑模块
局部主密引擎
数据库管理模块
文件备份逻辑模块
文件元信息生成器
本地文件系统

图 2

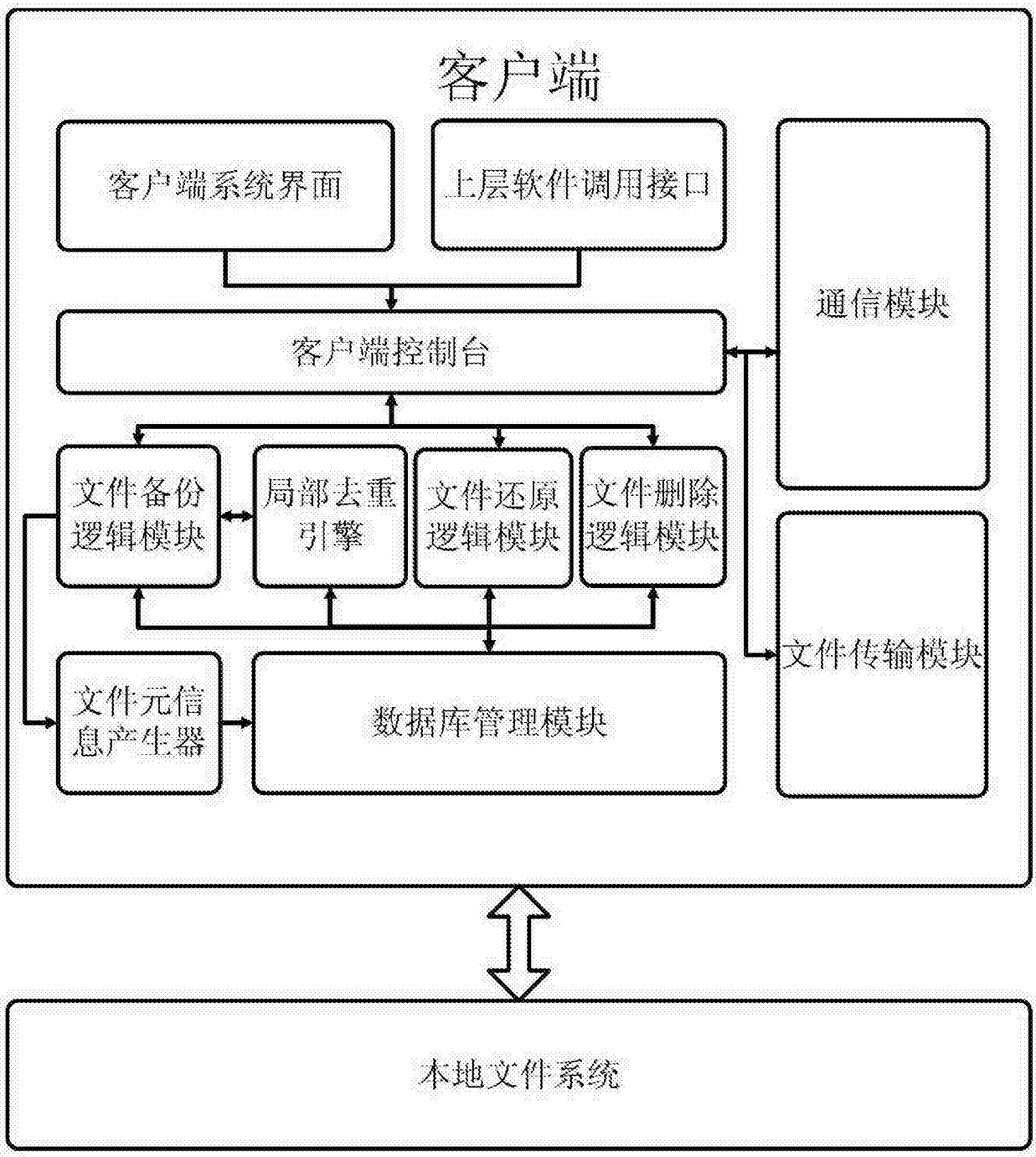

客户端
客户端系统界面
上层软件调用接口
通信模块
客户端控制台
文件备份逻辑模块
局部去重引擎
文件还原逻辑模块
文件删除逻辑模块
文件传输模块
文件元信息产生器
数据库管理模块
本地文件系统

图 3

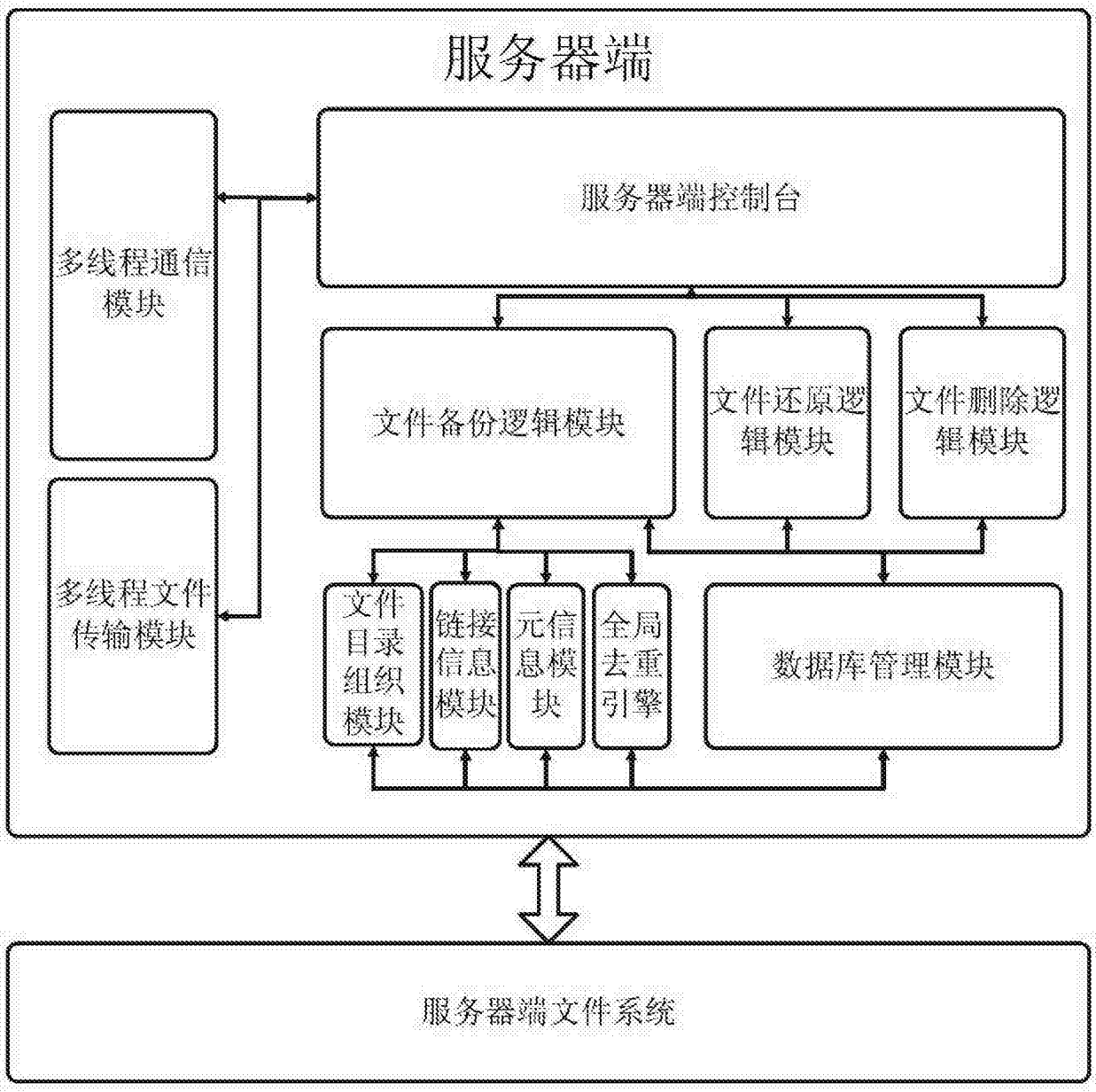

服务器端
多线程通信模块
服务器端控制台
文件备份逻辑模块
文件还原逻辑模块
文件删除逻辑模块
多线程文件传输模块
文件目录组织模块
链接信息模块
元信息模块
全局去重引擎
数据库管理模块
服务器端文件系统

图 4

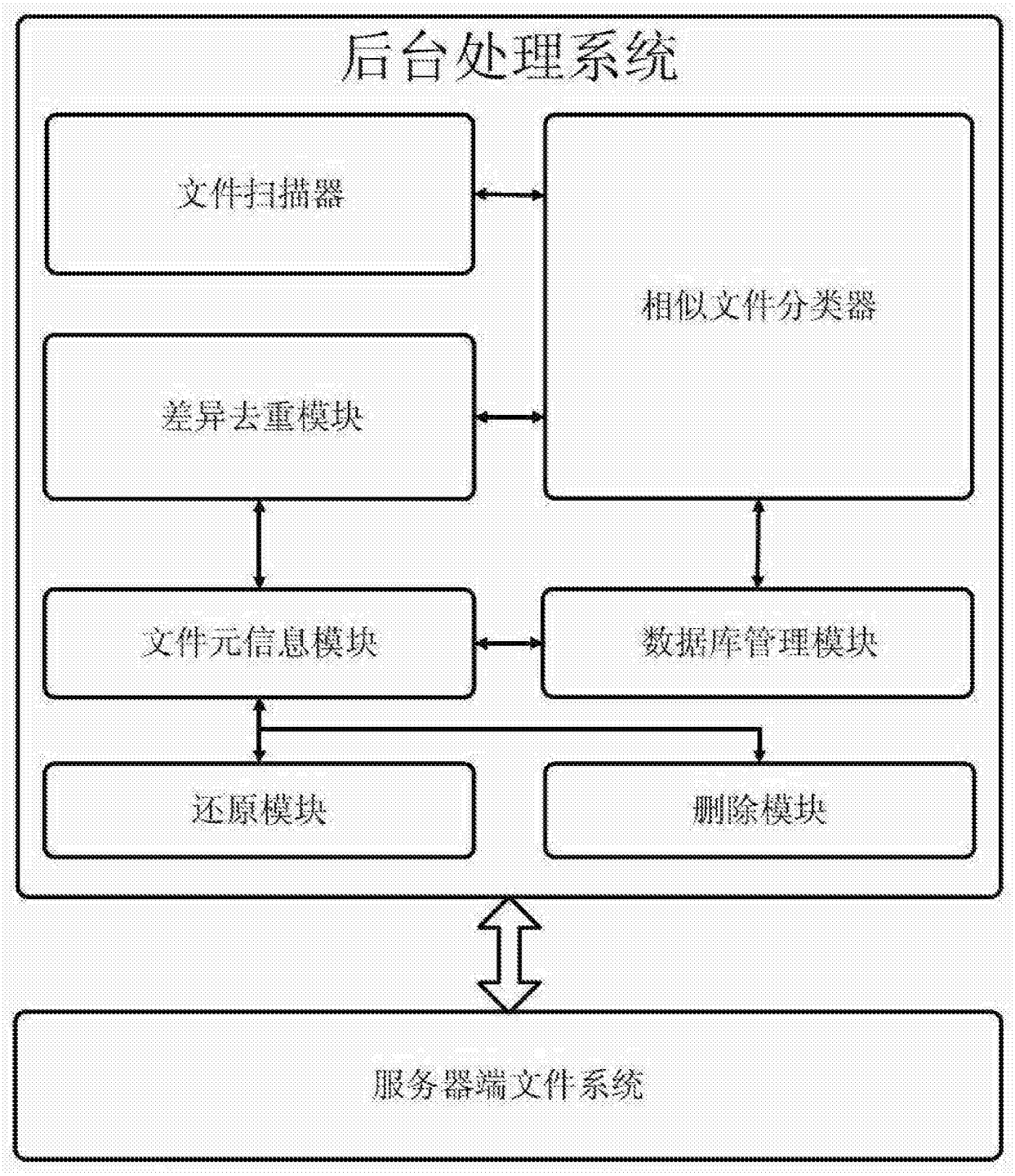

后台处理系统
文件扫描器
相似文件分类器
差异去重模块
文件元信息模块
数据库管理模块
还原模块
删除模块
服务器端文件系统

图 5

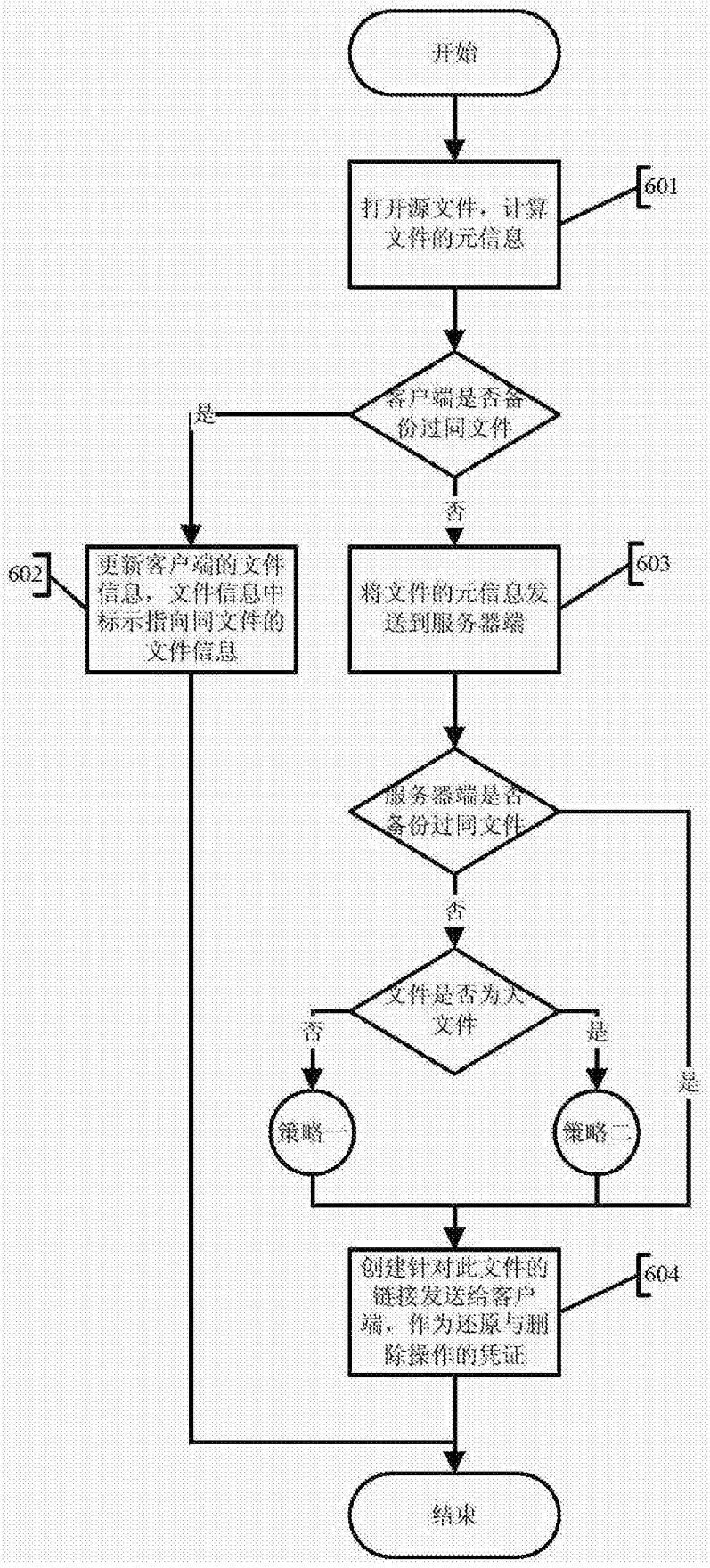

开始
打开源文件，计算文件的元信息
601
客户端是否备份过同文件
是
否
更新客户端的文件信息，文件信息中标示指向同文件的文件信息
602
将文件的元信息发送到服务器端
603
服务器端是否备份过同文件
是
否
文件是否为大文件
否
是
策略一
策略二
创建针对此文件的链接发送给客户端，作为还原与删除操作的凭证
604
结束

图 6

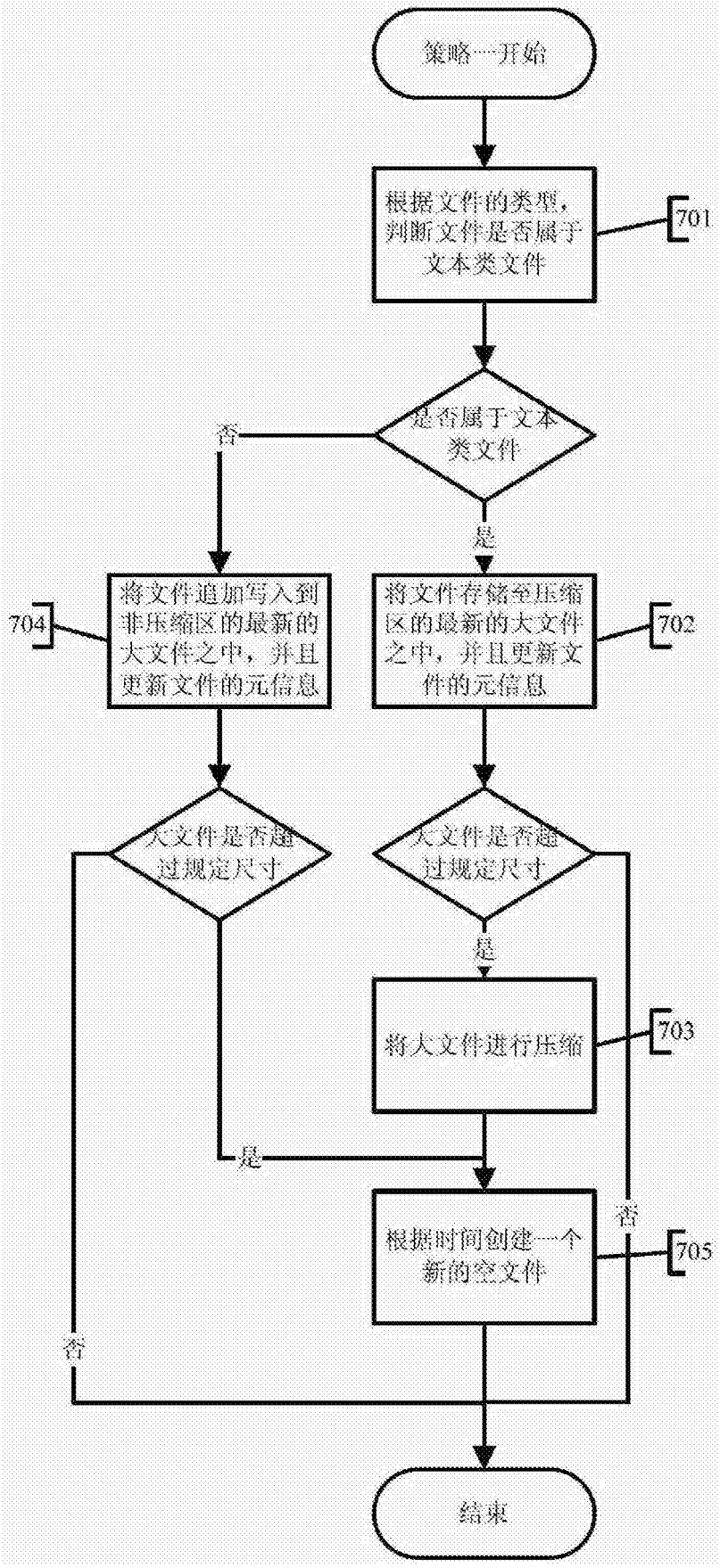

策略一开始
根据文件的类型，判断文件是否属于文本类文件
701
是否属于文本类文件
否
是
将文件追加写入到非压缩区的最新的大文件之中，并且更新文件的元信息
704
将文件存储至压缩区的最新的大文件之中，并且更新文件的元信息
702
大文件是否超过规定尺寸
大文件是否超过规定尺寸
是
将大文件进行压缩
703
是
否
根据时间创建一个新的空文件
705
否
结束

图 7

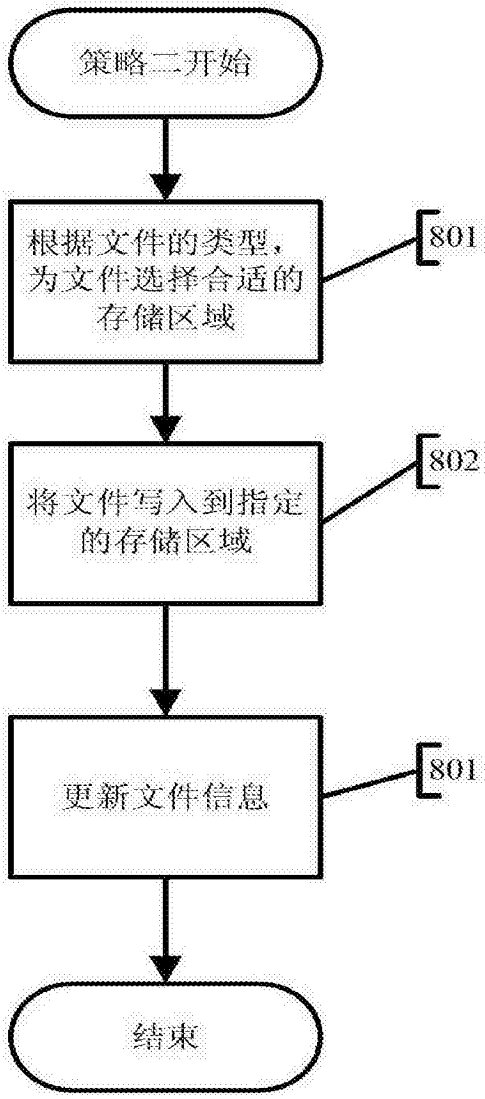

策略二开始
根据文件的类型，为文件选择合适的存储区域
801
将文件写入到指定的存储区域
802
更新文件信息
801
结束

图 8

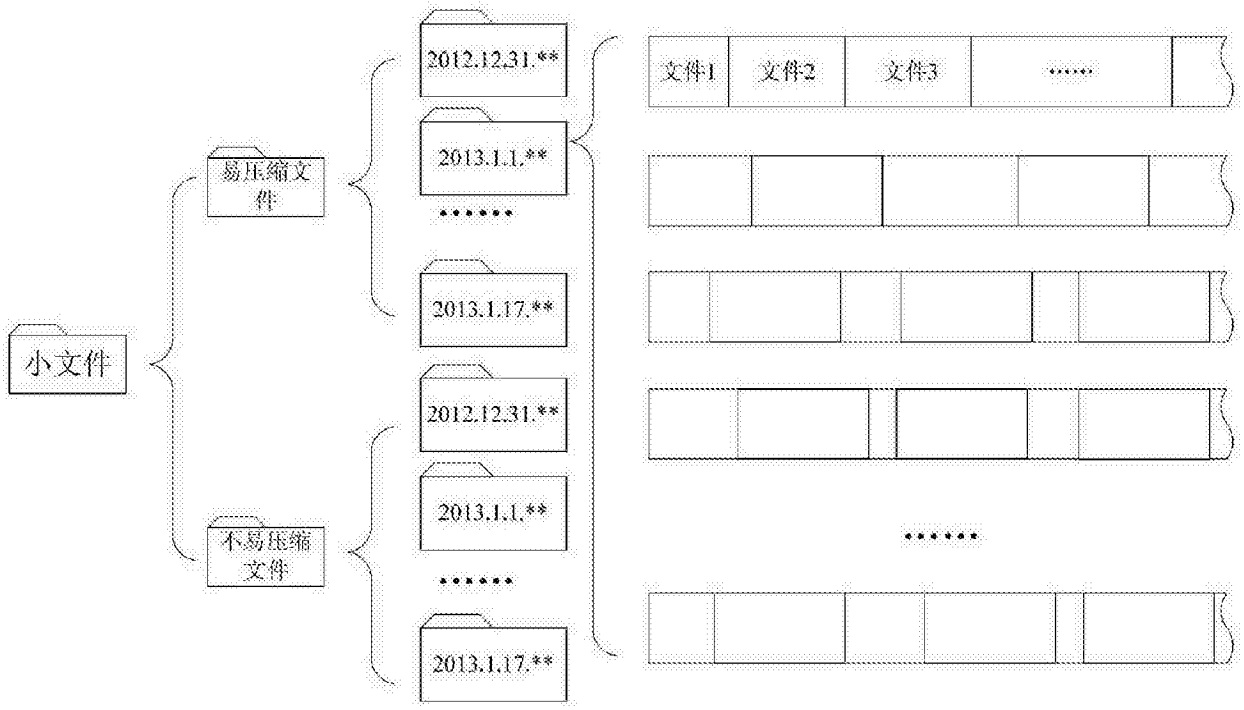

小文件
易压缩文件
不易压缩文件
2012.12.31.**
2013.1.1.**
2013.1.17.**
2012.12.31.**
2013.1.1.**
2013.1.17.**
文件1
文件2
文件3

图 9

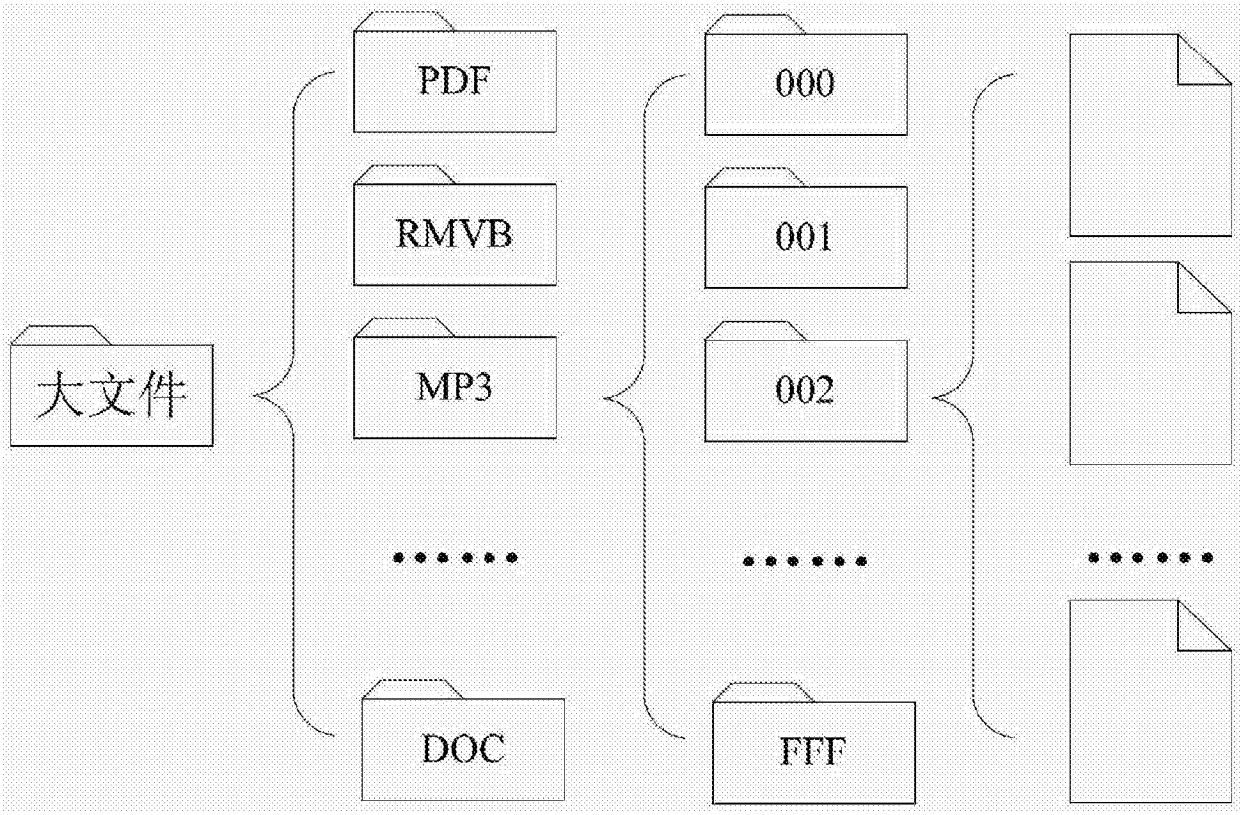

大文件
PDF
RMVB
MP3
DOC
000
001
002
FFF

图 10

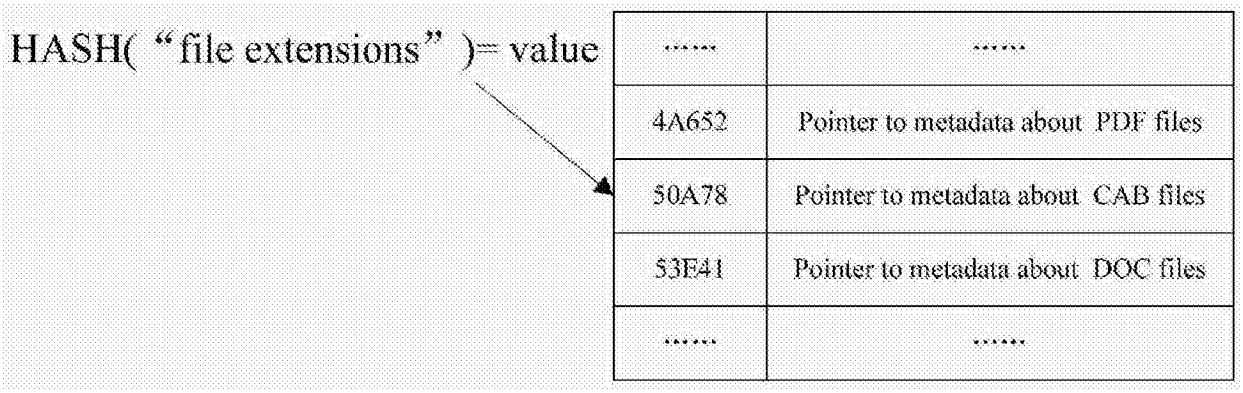

HASH(“file extensions”)= value
……
……
4A652
Pointer to metadata about PDF files
50A78
Pointer to metadata about CAB files
53E41
Pointer to metadata about DOC files
……
……

图 11

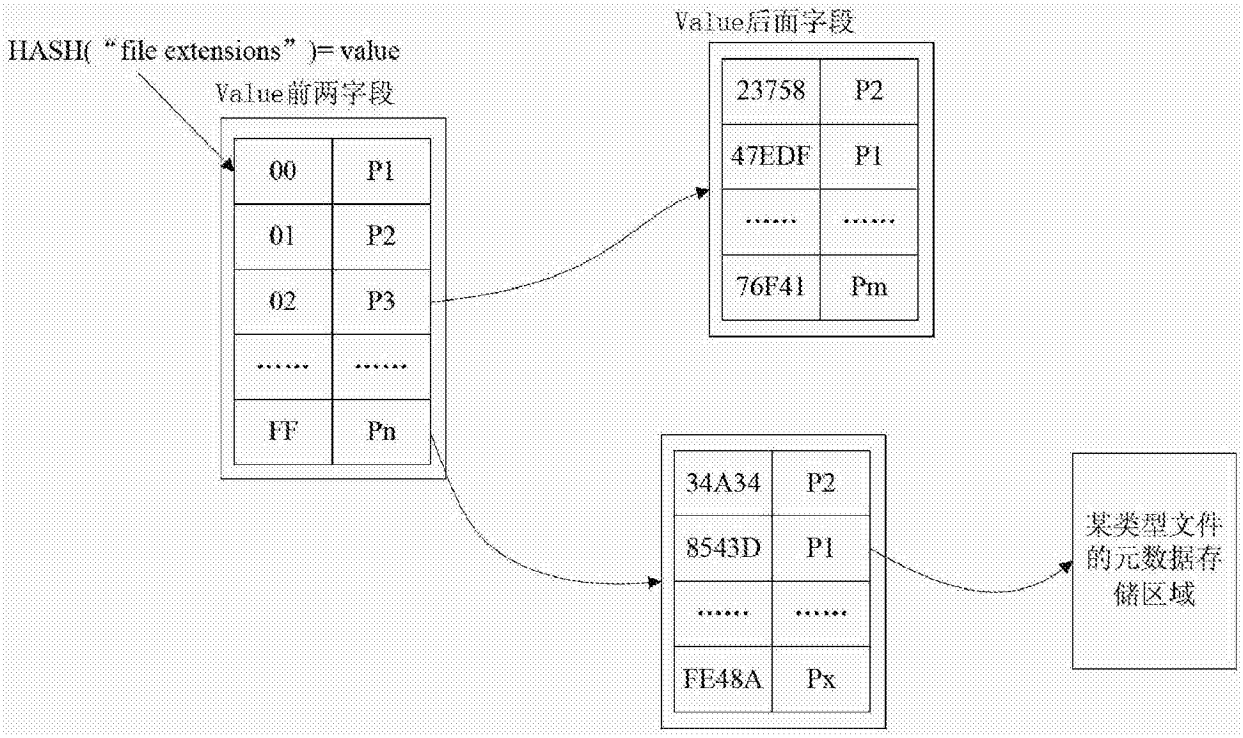

HASH(“file extensions”)= value
Value前两字段
00
P1
01
P2
02
P3
……
……
FF
Pn
Value后面字段
23758
P2
47EDF
P1
……
……
76F41
Pm
34A34
P2
8543D
P1
……
……
FE48A
Px
某类型文件的元数据存储区域

图 12

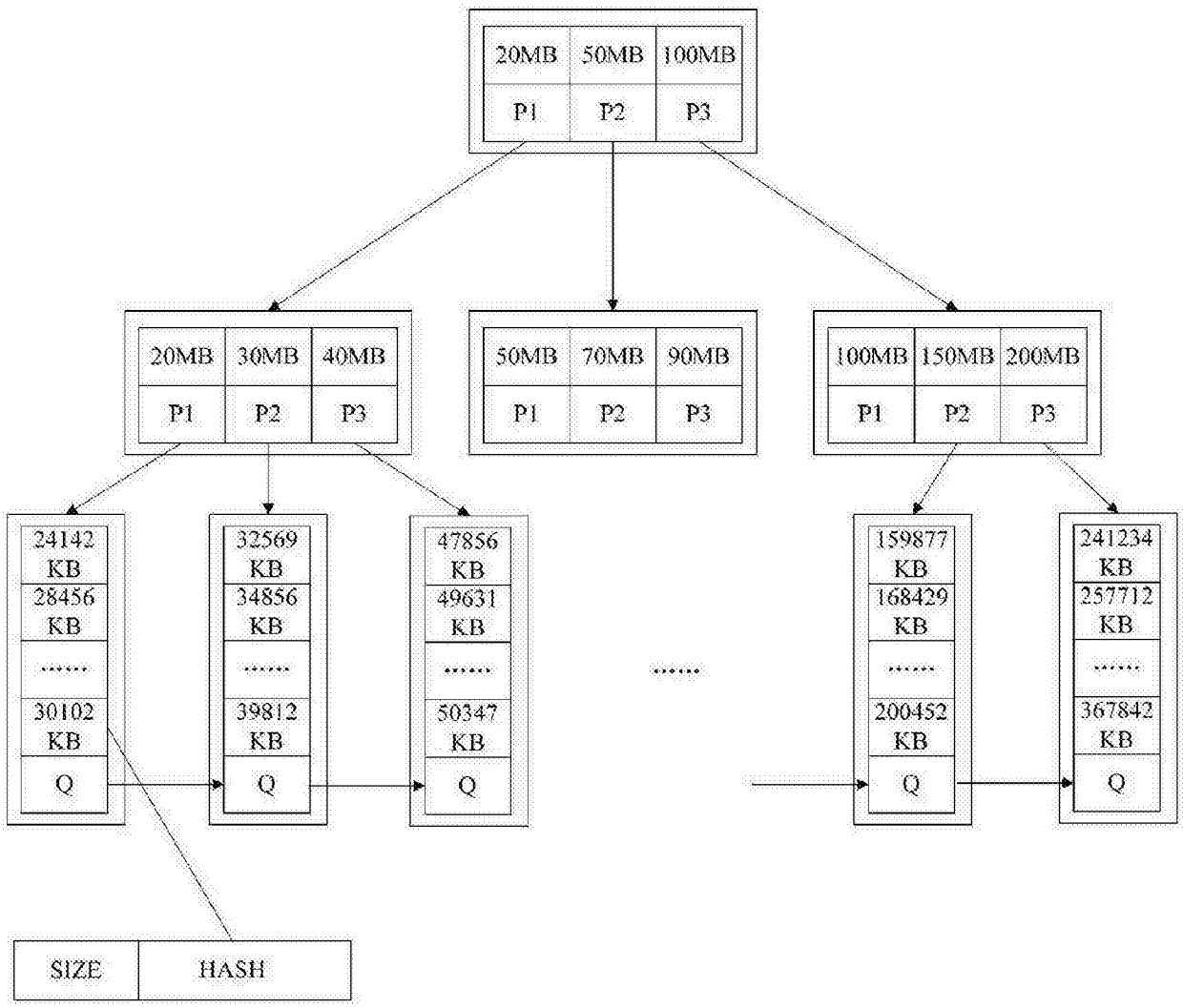

20MB
50MB
100MB
P1
P2
P3
20MB
30MB
40MB
P1
P2
P3
50MB
70MB
90MB
P1
P2
P3
100MB
150MB
200MB
P1
P2
P3
24142 KB
28456 KB
.....
30102 KB
Q
32569 KB
34856 KB
.....
39812 KB
Q
47856 KB
49631 KB
.....
50347 KB
Q
.....
159877 KB
168429 KB
.....
200452 KB
Q
241234 KB
257712 KB
.....
367842 KB
Q
SIZE
HASH

图 13

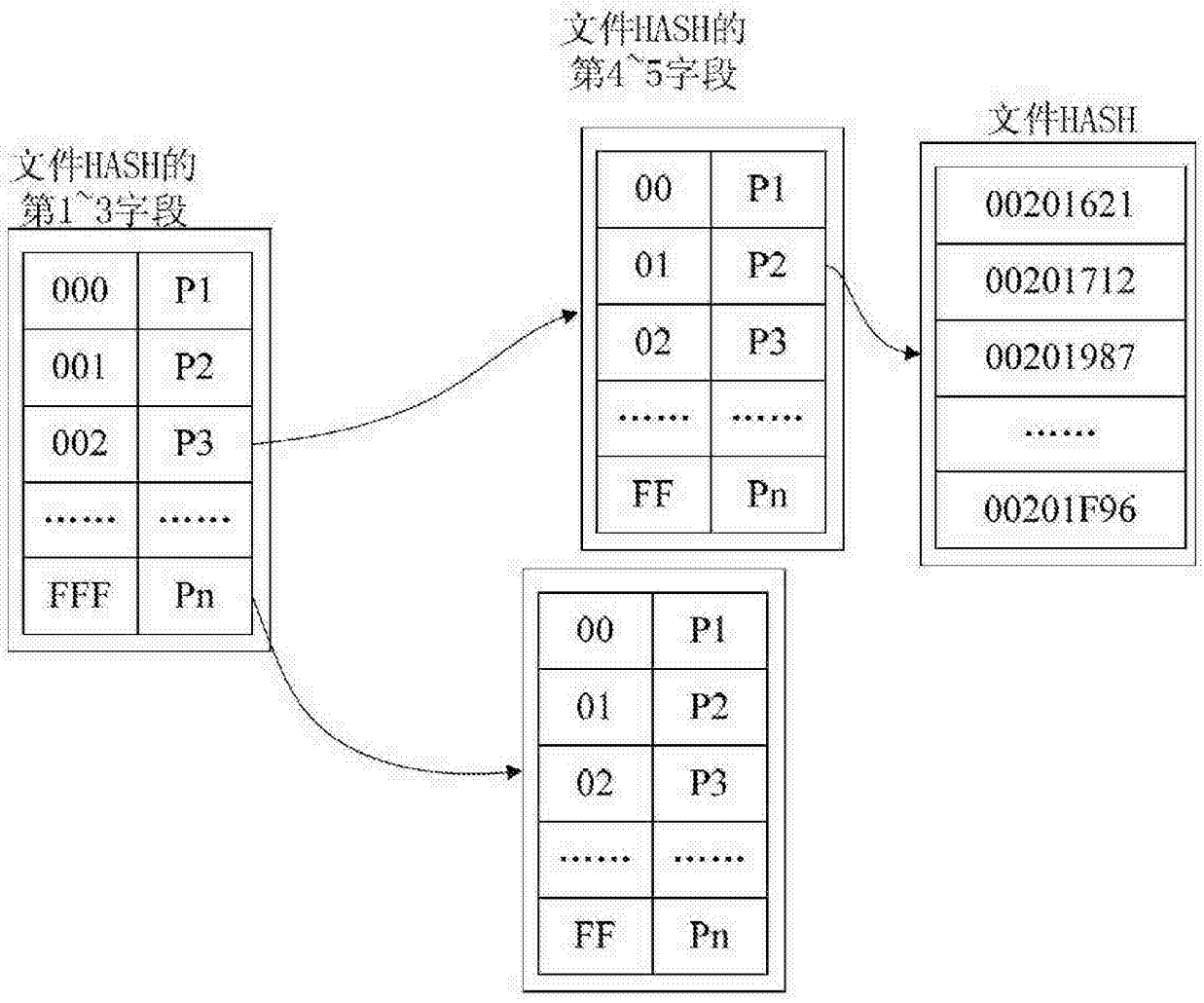

文件HASH的
第1~3字段
000
P1
001
P2
002
P3
.....
.....
FFF
Pn
文件HASH的
第4~5字段
00
P1
01
P2
02
P3
.....
.....
FF
Pn
00
P1
01
P2
02
P3
.....
.....
FF
Pn
文件HASH
00201621
00201712
00201987
.....
00201F96

图 14

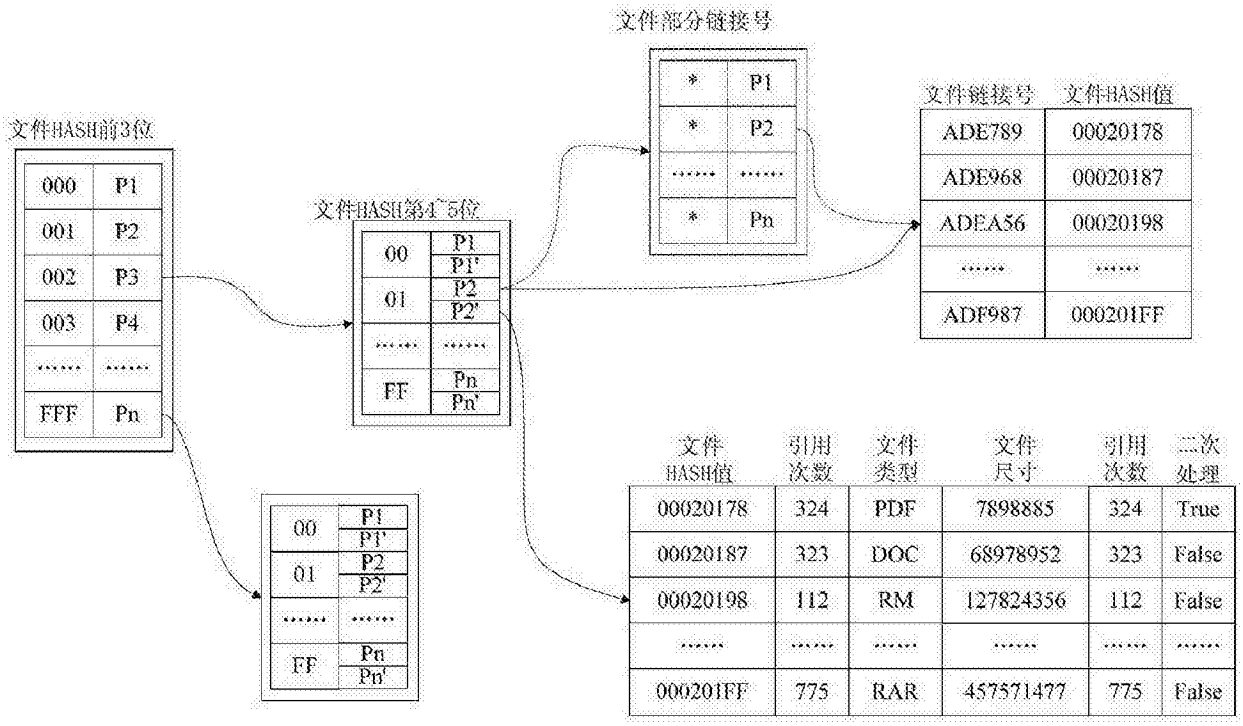


图 15

文件HASH值	引用次数	保存时间	所属文件名	文件位置	文件尺寸	二次处理
00020154	24	2012.09.12	0AE7895	0	7895	True
00020163	32	2012.09.13	0DE986A	31654	68952	False
00020178	2	2012.09.14	18956AD	487	1256	True
……	……	……	……	……	……	……
000201FF	98	2012.09.14	18995ED	99653	8426	False

图 16

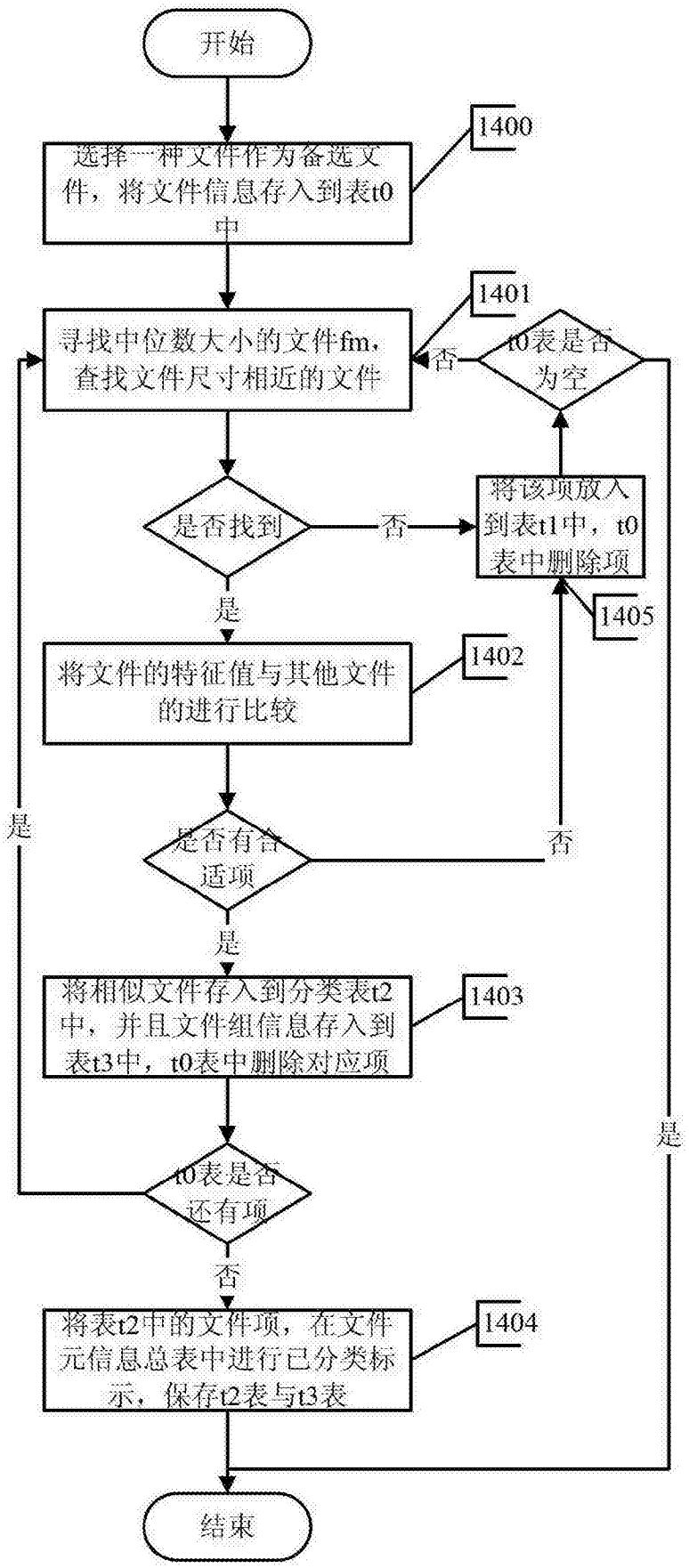

开始
选择一种文件作为备选文件，将文件信息存入到表t0中
1400
寻找中位数大小的文件fm，查找文件尺寸相近的文件
1401
t0表是否为空
否
是否找到
否
将该项放入到表t1中，t0表中删除项
1405
是
将文件的特征值与其他文件的进行比较
1402
是否有合适项
否
是
将相似文件存入到分类表t2中，并且文件组信息存入到表t3中，t0表中删除对应项
1403
t0表是否还有项
是
否
将表t2中的文件项，在文件元信息总表中进行已分类标示，保存t2表与t3表
1404
结束

图 17

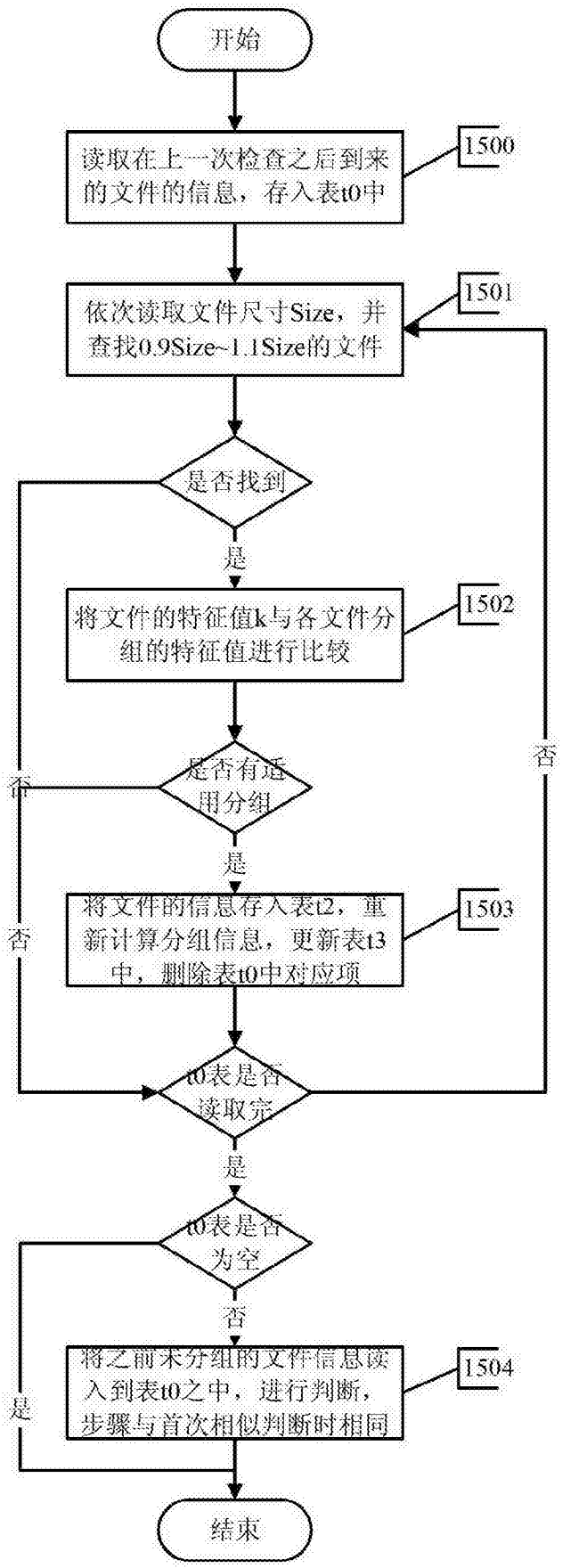

开始
读取在上一次检查之后到来的文件的信息，存入表t0中
1500
依次读取文件尺寸Size，并查找0.9Size~1.1Size的文件
1501
是否找到
是
否
将文件的特征值k与各文件分组的特征值进行比较
1502
是否有适用分组
是
否
将文件的信息存入表t2，重新计算分组信息，更新表t3中，删除表t0中对应项
1503
否
t0表是否读取完
是
否
t0表是否为空
是
否
将之前未分组的文件信息读入到表t0之中，进行判断，步骤与首次相似判断时相同
1504
是
结束

图 18

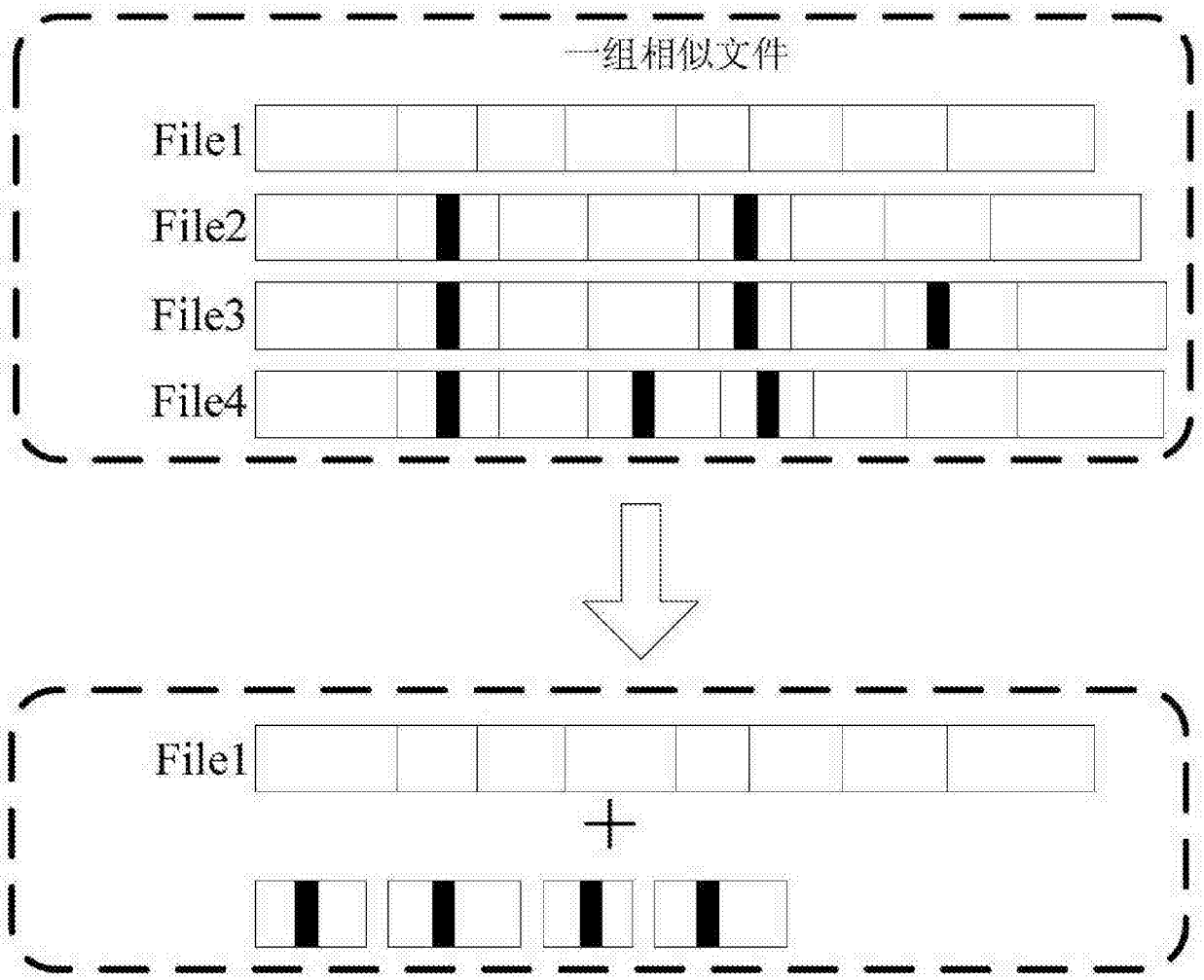


图 19

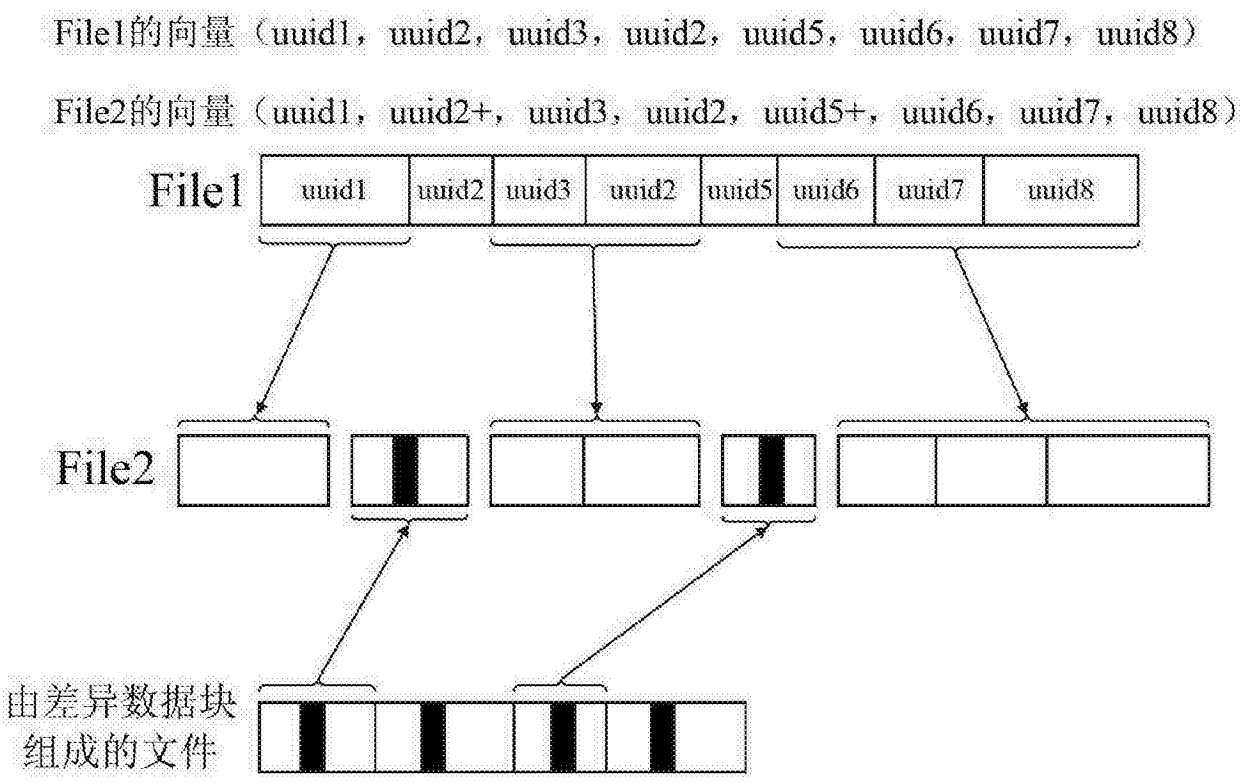

File1的向量（uuid1，uuid2，uuid3，uuid2，uuid5，uuid6，uuid7，uuid8）
File2的向量（uuid1，uuid2+，uuid3，uuid2，uuid5+，uuid6，uuid7，uuid8）
File1
uuid1
uuid2
uuid3
uuid2
uuid5
uuid6
uuid7
uuid8
File2
由差异数据块
组成的文件

图 20