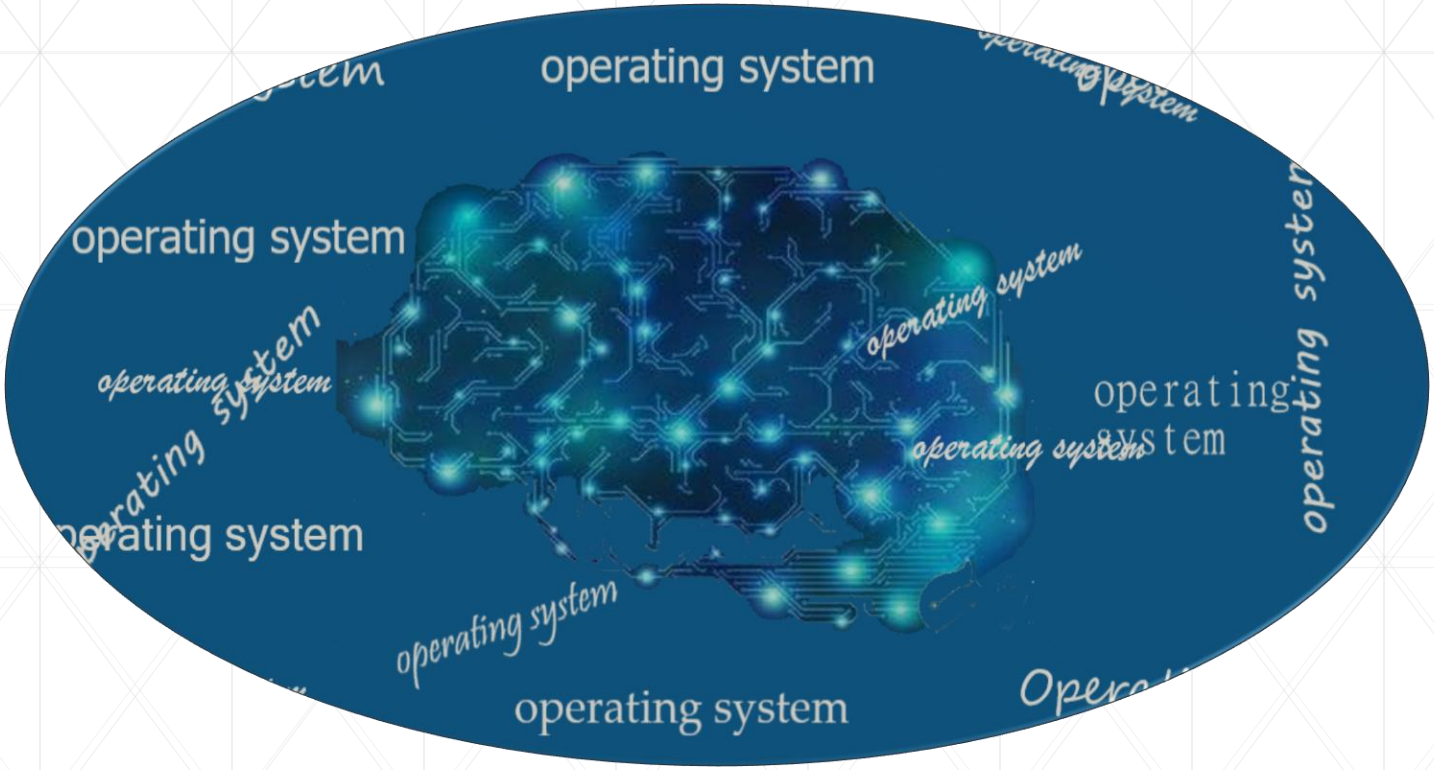


第六章 文件系统

授课教师 电子邮箱：

本章主要内容

- 1）文件的概念，文件系统的层次结构和文件的基本操作。
- 2）目录管理的要求，文件控制块和索引节点，以及查询技术，文件的共享方式和访问控制机制。
- 3）目录管理的要求，文件控制块和索引节点，以及查询技术。
- 4）文件的物理结构和外存空间的管理：外存组织的目的，连续组织和链接组织方式，空闲表法和位示图法和成组链接法。

第六章 文件系统

6.1 文件系统概述

6.2 文件的物理结构

6.3 文件存储空间的管理

6.4 文件目录

6.5 文件共享和访问控制

问题？

- 什么是文件?
- 文件由什么组成?
- 文件如何命名?
- 如何保证文件数据的安全?
- 对文件可以进行哪些操作?
- 文件在磁盘上如何存储?
- 磁盘的空白存储区如何管理 ？

6.1 文件系统概述

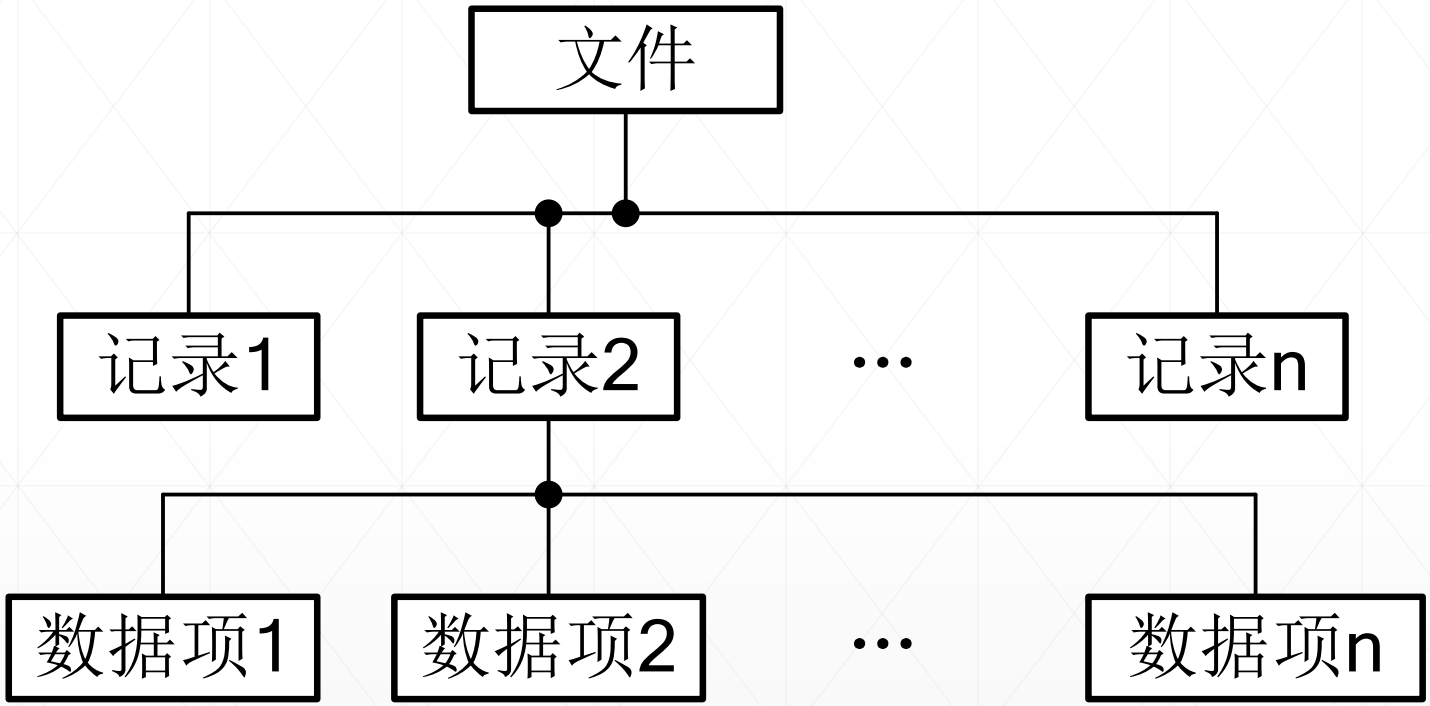
- 文件系统的功能/需解决的问题
  - 从系统角度看： 负责为用户建立、删除、读、写、修改和复制文件。
  - 从用户的角度看：实现了按名存取
- 文件系统的功能

  提供高效、快速、方便的信息存储和访问功能。

6.1 文件系统概述

- **文件系统的功能**
  - 有效地管理文件的存储空间；
  - 管理文件目录；
  - 完成文件的读/写操作；
  - 实现文件共享与保护；
  - 为用户提供交互式命令接口和程序调用接口。
- **定义：操作系统中的各类文件、管理文件的软件，以及管理文件所涉及到的数据结构等信息的集合。**
  - 有少数实时操作系统没有文件系统功能。
  - 绝大多数操作系统都包含文件管理系统部分。

6.1.1 文件、记录和数据项



6.1.1 文件、记录和数据项

- 1.数据项
  - 是最低级的数据组织形式，可把它分成以下两种类型：
  - (1) 基本数据项
  - 这是用于描述一个对象的某种属性的字符集，是数据组织中可以命名的最小逻辑数据单位，即原子数据，又称为数据元素或字段。
  - (2) 组合数据项
  - 它是由若干个基本数据项组成的，简称组项。

6.1.1 文件、记录和数据项

- 2. 记录
- 记录是一组相关数据项的集合，用于描述一个对象在某方面的属性。
- 3.文件
- 文件是指由创建者所定义的、 具有文件名的一组相关元素的集合。
  - **在有结构的文件中，文件由若干个相关记录组成；**
  - **而无结构文件则被看成是一个字符流。**
- 文件在文件系统中是一个最大的数据单位，它描述了一个对象集。

6.1.1 文件、记录和数据项

文件的属性

- 文件的属性可以包括：
  - **(1)文件类型。**
  - **(2) 文件长度。**
  - **(3) 文件的物理位置。**
  - **(4) 文件的建立时间。**

6.1.2 文件类型和文件系统模型

1. 文件类型

1）按用途分类

(1)系统文件 这是指由系统软件构成的文件。大多数的系统文件只允许用户调用，但不允许用户去读，更不允许修改；有的系统文件不直接对用户开放。

(2) 用户文件 由用户的源代码、目标文件、可执行文件或数据等所构成的文件。

(3) 库文件 这是由标准子例程及常用的例程等所构成的文件。这类文件允许用户调用，但不允许修改。

6.1.2 文件类型和文件系统模型

2) 按文件中数据的形式分类

(1)源文件

指由源程序和数据构成的文件。

(2) 目标文件

指把源程序经过相应语言的编译程序编译过，但尚未经过链接程序链接的目标代码所构成的文件。它属于二进制文件。

(3) 可执行文件

指把编译后所产生的目标代码再经过链接程序链接后所形成的文件。

6.1.2 文件类型和文件系统模型

3) 按存取控制属性分类

根据系统管理员或用户所规定的存取控制属性，可将文件分为三类：

(1)只执行文件

该类文件只允许被核准的用户调用执行，既不允许读，更不允许写。

(2) 只读文件

该类文件只允许文件主及被核准的用户去读，但不允许写。

(3) 读写文件

这是指允许文件主和被核准的用户去读或写的文件。

4) 按组织形式和处理方式分类

- 根据文件的组织形式和系统对其的处理方式，文件分为三类：
- **(1) 普通文件：** 由ASCII码或二进制码组成的字符文件。一般用户建立的源程序文件、数据文件、目标代码文件及操作系统自身代码文件、库文件、实用程序文件等都是普通文件，它们通常存储在外存储设备上。
- **(2) 目录文件：** 由文件目录组成的，用来管理和实现文件系统功能的系统文件，通过目录文件可以对其它文件的信息进行检索。由于目录文件也是由字符序列构成，因此对其可进行与普通文件一样的种种文件操作。
- **(3) 特殊文件：** 特指系统中的各类I/O 设备。为了便于统一管理，系统将所有的输入/输出设备都视为文件，按文件方式提供给用户使用

6.1.3 文件系统模型

用户（程序）

↓ ↓ ↓ ↓

<table>
<tr><td colspan="2">文件系统接口（命令接口、系统调用接口）</td></tr>
<tr><td>对对象操纵和管理的软件集合</td><td>对文件存储空间的管理
对文件目录的管理
将文件的逻辑地址转换为物理地址的机制
对文件的读和写管理
对文件的共享和保护</td></tr>
<tr><td colspan="2">对象（文件、目录及磁盘存储空间）及其属性说明</td></tr>
</table>

6.1.3 文件系统模型

1) 对象及其属性

文件管理系统管理的对象有：

① **文件：** 它作为文件管理的直接对象。

② **目录：** 为了方便用户对文件的存取和检索，在文件系统中必须配置目录。对目录的组织和管理是方便用户和提高对文件存取速度的关键。

③ **磁盘(磁带)存储空间：** 文件和目录必定占用存储空间，对这部分空间的有效管理，不仅能提高外存的利用率，而且能提高对文件的存取速度。

6.1.3 文件系统模型

2) 对对象操纵和管理的软件集合

这是文件管理系统的核心部分。文件系统的功能大多是在这一层实现的，其中包括：

- 对文件存储空间的管理、
- 对文件目录的管理、
- 用于将文件的逻辑地址转换为物理地址的机制
- 对文件读和写的管理，
- 以及对文件的共享与保护等功能。

6.1.3 文件系统模型

3) 文件系统的接口

为方便用户使用文件系统，文件系统通常向用户提供两种类型的接口：

(1) 命令接口。这是指作为用户与文件系统交互的接口。 用户可通过键盘终端键入命令，取得文件系统的服务。

(2) 程序接口。这是指作为用户程序与文件系统的接口。 用户程序可通过系统调用来取得文件系统的服务。

示例：文件操作

- **用户通过文件系统提供的系统调用实施对文件的操作。**

1.最基本的文件操作有：创建文件、删除文件。读文件、写文件、截断文件和设置文件的读／写位置。

2.文件的“打开”和“关闭”操作：所谓“打开”，是指系统将指名文件的属性（包括该文件在外存上的物理位置）从外存拷贝到内存打开文件表的一个表目中，并将该表目的编号（或称为索引）返回给用户。 利用“关闭”（close）系统调用来关闭此文件，OS将会把该文件从打开文件表中的表目上删除掉。

3．其它文件操作：对文件属性的操作，改变文件名、改变文件的拥有者，查询文件的状态等；

文件操作实例（Linux）

- **open**：打开一个文件，并指定访问该文件的方式，调用成功后返回一个文件描述符。
- **creat**：打开一个文件，如果该文件不存在，则创建它，调用成功后返回一个文件描述符。
- **close**：关闭文件，进程对文件所加的锁全都被释放。
- **read**：从文件描述符对应的文件中读取数据，调用成功后返回读出的字节数。
- **write**：向文件描述符对应的文件中写入数据，调用成功后返回写入的字节数。

第六章 文件系统

6.1 文件系统概述

6.2 文件的物理结构

6.3 文件存储空间的管理

6.4 文件目录

6.5 文件共享和访问控制

6.2 文件的物理结构

文件系统设计的关键要素：记录构成文件的方法

将文件存储到外存的方法

对任一文件存在着两种形式的结构：

- 文件的逻辑结构（文件的组织）

  从用户观点出发，所观察到的文件组织形式，是用户可以直接处理的数据及结构，它独立于物理特性。

- 文件的物理结构（文件的存储结构）

  是指文件在外存上的存储组织形式，与存储介质的存储性能有关，还和所采用的外存分配方式有关。（分为顺序、链接及索引结构）

注：文件的逻辑结构和物理结构都将影响文件的检索速度。

6.2 文件的物理结构

文件的逻辑结构

对文件的逻辑结构提出的基本要求：

提高检索速度；

便于修改；

降低文件存储费用。

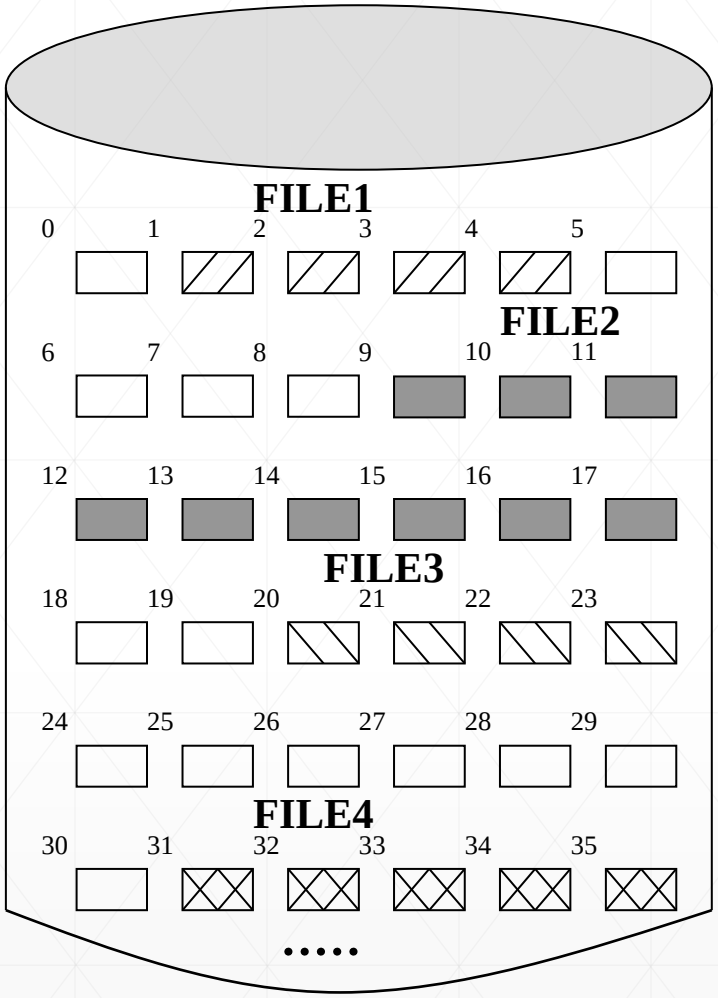
- 文件逻辑结构的类型
- 顺序文件
- 索引文件
- 索引顺序文件

6.2.1 文件的物理组织—存储空间的管理

- 文件的物理结构即文件的外存分配方式，是从系统的角度来看文件，从文件在物理介质上的存放方式来研究文件。
- 要考虑的主要问题：如何有效地利用外存空间
  如何提高对文件的访问速度
- 目前常用的外存分配方法：
  （1）连续分配（顺序分配）—> 顺序文件结构
  （2）链接分配—> 链接文件结构
  （3）索引分配—> 索引文件结构

(1) 连续分配

- 连续分配(Continuous Allocation)要求为每一个文件分配一组相邻接的盘块。一组盘块的地址定义了磁盘上的一段线性地址。
- **把逻辑文件中的数据顺序地存储到物理上邻接的各个数据块中，这样形成的物理文件可以进行顺序存取。**
- **文件目录中为每个文件建立一个表项，其中记载文件的第一个数据块地址及文件长度。**
- **对于顺序文件，连续读/写多个数据块内容时，性能较好。**



目录表

文件名	起始块号	文件长度
FILE1	1	4
FILE2	9	9
FILE3	20	4
FILE4	31	5
…	…	…

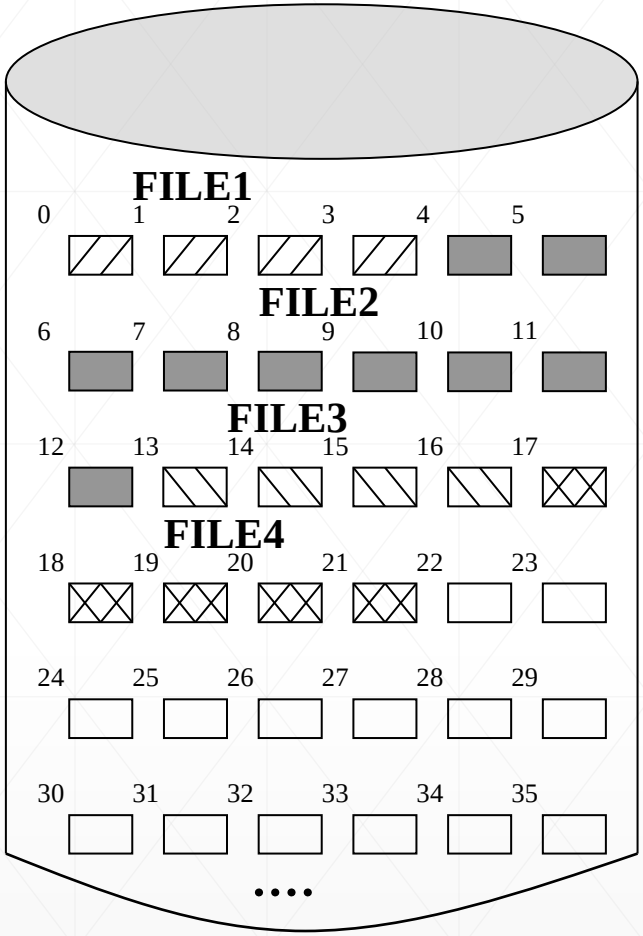
磁盘空间的连续分配

(1) 连续分配

- **连续分配的主要优点：**
  - **（1）顺序访问容易**
    - 可以随机存取，能很快检索文件中的一个数据块。
    - 例如，如果一个文件的第一个数据块的序号为x，需要检索文件的第y块，则该数据块在外存中的位置为x+y-1。
  - **（2）顺序访问速度快**
    - 磁头移动距离短，效率最高

(1) 连续分配

- **连续分配存在的问题**
- （1）要求有连续的存储空间。
  - 该分配方案可能会导致磁盘碎片，严重降低外存空间的利用率。
  - 解决方法之一，系统定期或不定期采用紧凑技术，将小分区合并为大的、连续分区，**将文件占用空间合并在一起**。
- （2）必须事先知道文件的长度。
  - 空间利用率不高；
  - 不利于文件尺寸的动态增长。



目录表

文件名	起始块号	文件长度
FILE1	0	4
FILE2	4	9
FILE3	13	4
FILE4	17	5
…	…	…

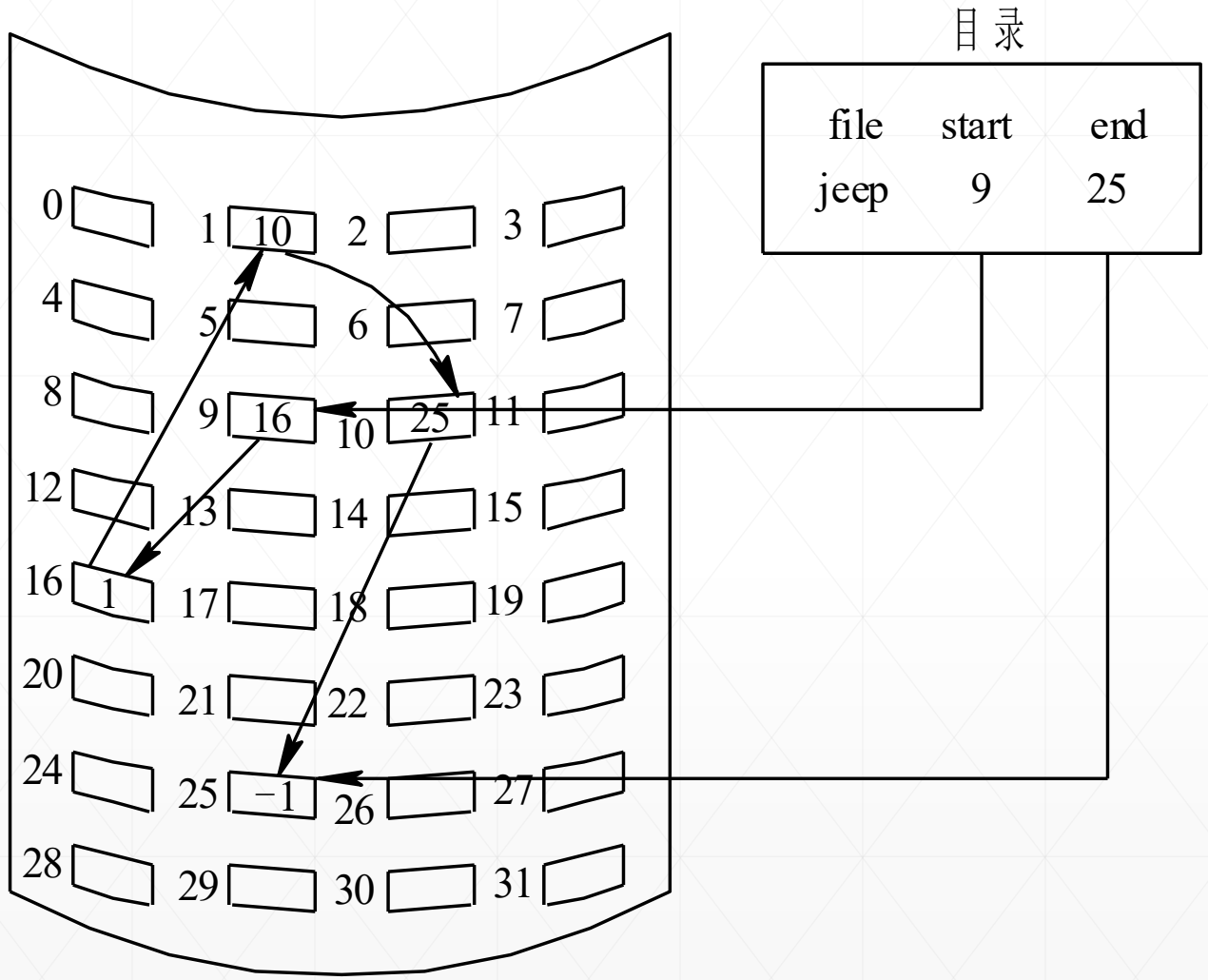
磁盘 连续分配（紧凑以后）

(2) 链接分配

- **连续分配的文件分区太大，不利于存储空间的有效利用。**
- 如果在将一个逻辑文件存储到外存上时，可以考虑将文件装到多个离散的盘块中。
- 链接文件：采用链接分配方式时，可通过在每个盘块上的链接指针，将同属于一个文件的多个离散的盘块链接成一个链表，把这样形成的物理文件称为链接文件。

(2) 链接分配

- **1）隐式链接**
- 在采用隐式链接分配方式时，在文件目录的每个目录项中，都须含有指向链接文件第一个盘块和最后一个盘块的指针。
- 每个盘块中都含有一个指向下一个盘块的指针。



磁盘空间的隐式链接式分配

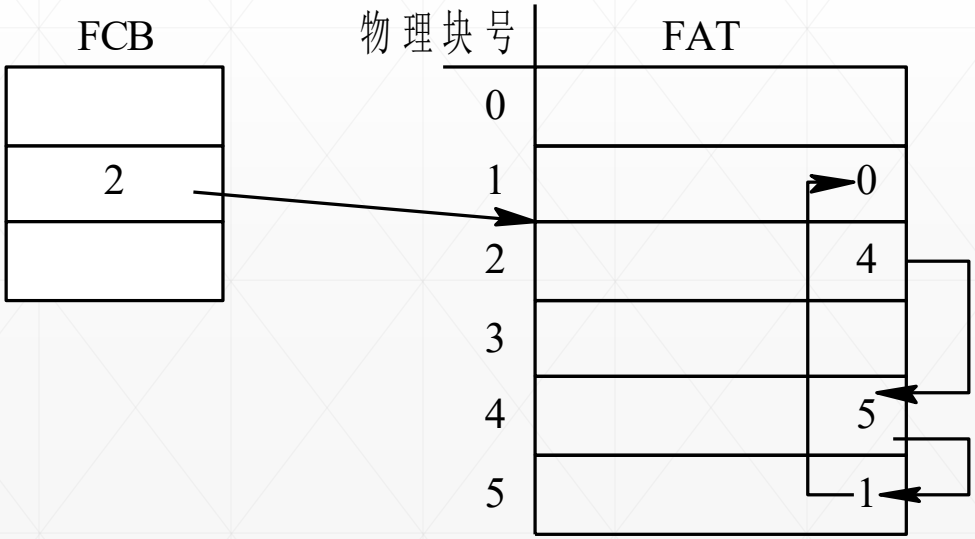
(2) 链接分配

1）隐式链接

- 隐式链接分配方式的主要问题在于：它只适合于顺序访问，它对随机访问是极其低效的。
  - 如果要访问文件所在的第 i 个盘块，则必须先读出文件的第一个盘块……，就这样顺序地查找直至第 i 块。
- 为了提高检索速度和减小指针所占用的存储空间，可以将几个盘块组成一个簇(cluster)。
  - **比如，一个簇可包含4 个盘块，在进行盘块分配时，是以簇为单位进行的。在链接文件中的每个元素也是以簇为单位的。**
  - **减少查找时间和指针所占空间，但增大了内部碎片**
- 这种改进也是非常有限的。

(2) 链接分配

- **2）显式链接**
- **这是指把用于*链接文件各物理块的指针*，显式地存放在内存的一张链接表中。**
- **整个磁盘仅设置一张文件分配表（FAT）。**



(2) 链接分配

2）显式链接

- 在该表中，凡是属于某一文件的第一个盘块号，均作为文件地址被填入相应文件的FCB 的“*物理地址*”字段中。
- 查找记录的过程是在内存中进行的，因而不仅显著地提高了检索速度，而且大大减少了访问磁盘的次数。
- 由于分配给文件的所有盘块号都放在该表中，故把该表称为*文件分配表FAT*(File Allocation Table)。

(2) 链接分配

2）显式链接

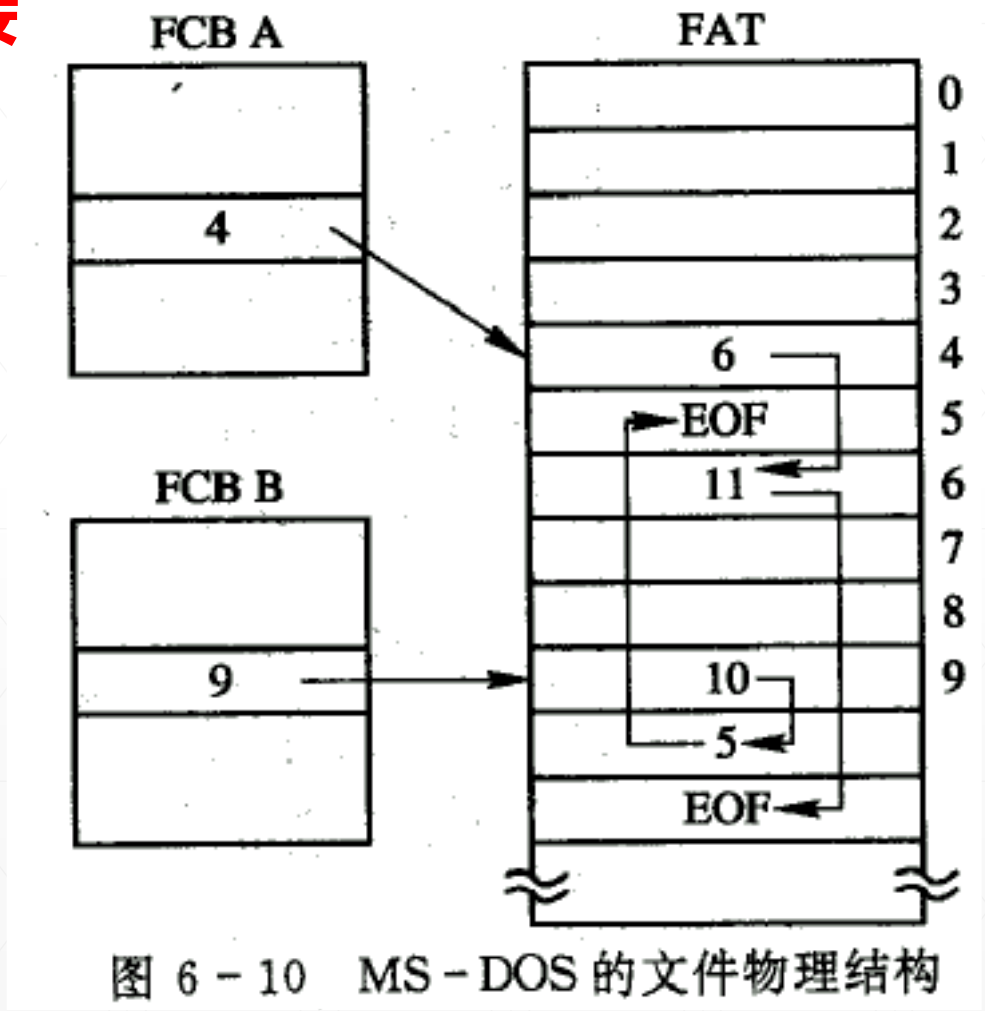


图 6－10　MS－DOS 的文件物理结构

(2) 链接分配

链接分配的优缺点

- 优点

  1、无外部碎片，没有磁盘空间浪费

  2、无需事先知道文件大小。文件动态增长时，可动态分配空闲盘块。对文件的增、删、改十分方便。

- 缺点

  1、不能支持高效随机/直接访问，仅适合于顺序存取

  2、需为指针分配空间。（隐式链接）

  3、可靠性较低（指针丢失/损坏）

  4、文件分配表FAT（显式链接）

  FAT需占用较大的内存空间。

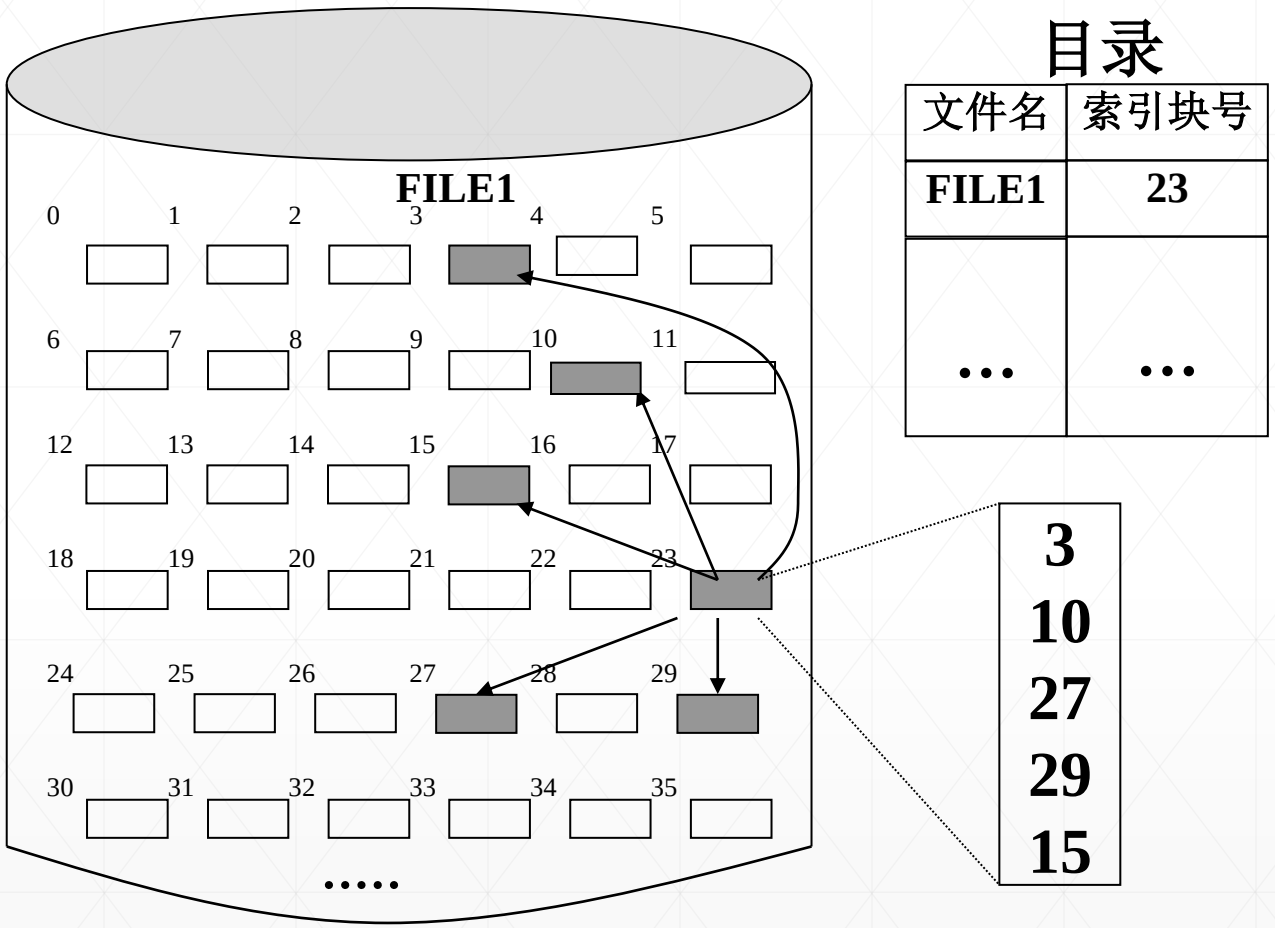
(3) 索引分配

- 链接分配方式虽然解决了连续分配方式所存在的问题，但又出现了另外两个问题，即：
  - **(1) 不能支持高效的直接存取。要对一个较大的文件进行直接存取，须首先在FAT中顺序地查找许多盘块号。**
  - **(2) FAT需占用较大的内存空间。由于一个文件所占用盘块的盘块号是随机地分布在FAT中的，因而只有将整个FAT 调入内存，才能保证在FAT 中找到一个文件的所有盘块号。**

(3) 索引分配

1）单级索引分配

- **索引分配能解决连续分配和链接分配存在的诸多问题。**
- 原理：为每个文件分配一个索引块(表)，再把分配给该文件的所有盘块号都记录在该索引块中，因而该索引块就是一个含有许多盘块号的数组。
- 在建立一个文件时，只需在为之建立的目录项中填上指向该索引块的指针。



基于数据块分区的索引分配

(3) 索引分配

1）单级索引分配

- 优点：
  - 索引分配方式支持直接访问。当要读文件的第*i*个盘块时，可以方便地直接从索引块中找到第*i*个盘块的盘块号；
  - 基于数据块的分区能消除外部碎片
- 缺点：
  - 大文件索引项较多，可能使一个数据块容纳不了一个文件的所有分区的索引。
  - 索引块可能要花费较多的外存空间。每当建立一个文件时，便须为之分配一个**专门的索引块**，将分配给该文件的所有盘块号记录于其中。对于小文件如果采用这种方式，索引块的利用率将是极低的。

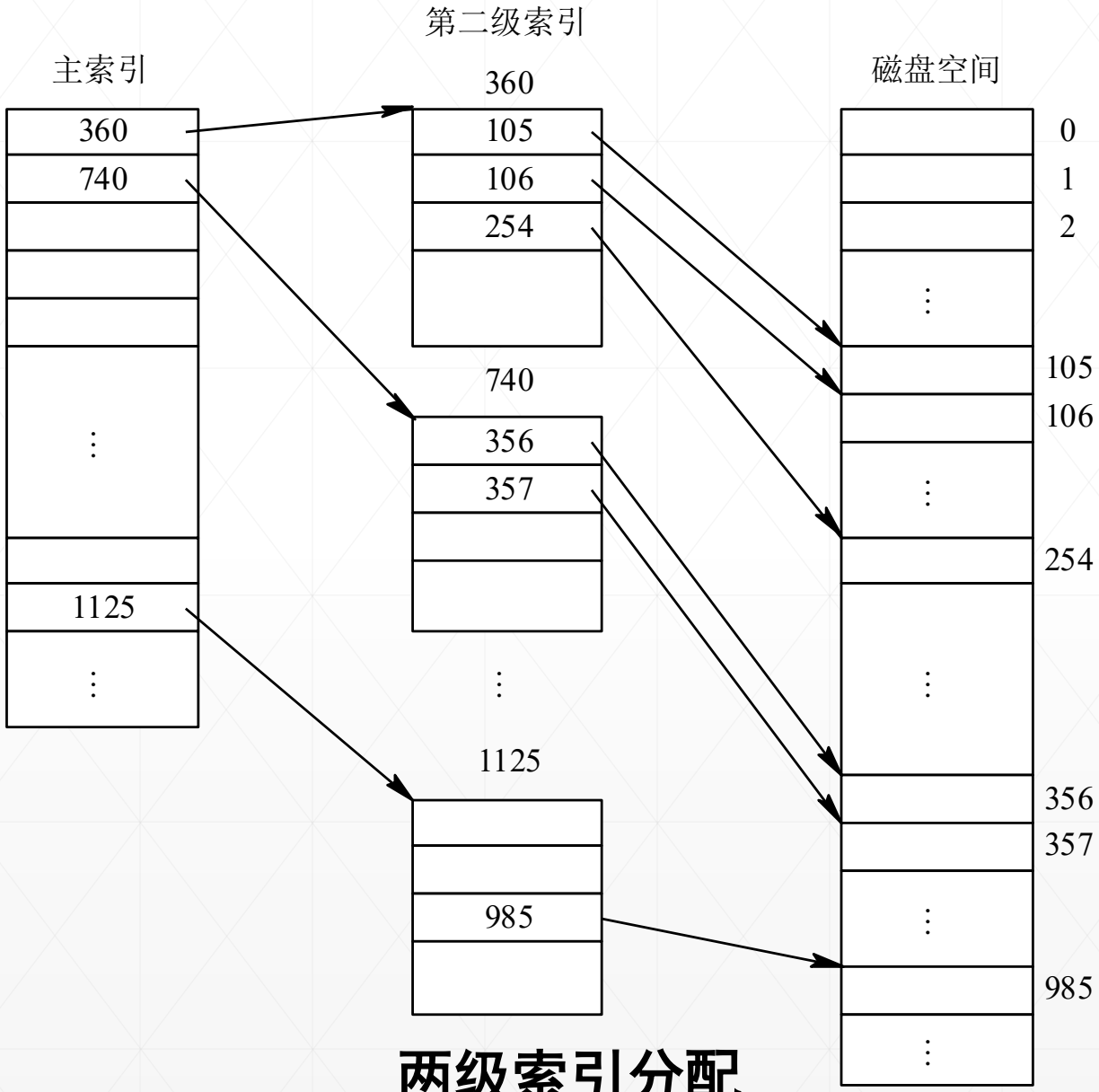
(3) 索引分配

2) 多级索引分配

- 当文件太大，其一级索引块太多时，这种方法是低效的。

- **此时，应为这些索引块再建立一级索引，形成*两级索引分配方式。***
  - **即系统再分配一个索引块，作为第一级索引的索引块，将第一块、第二块……等索引块的盘块号填入到此索引表中**

2) 多级索引分配



两级索引分配

(3) 索引分配

2) 多级索引分配

假设每个盘块大小为1KB，每个盘块号占4个字节，则在一个索引块中可放256个文件物理块的盘块号。在两级索引时，最多可包含的存放文件的盘块的盘块号总数为256*256=64K个盘块号。可以得出：采用两级索引时，所允许的文件最大长度为64MB。

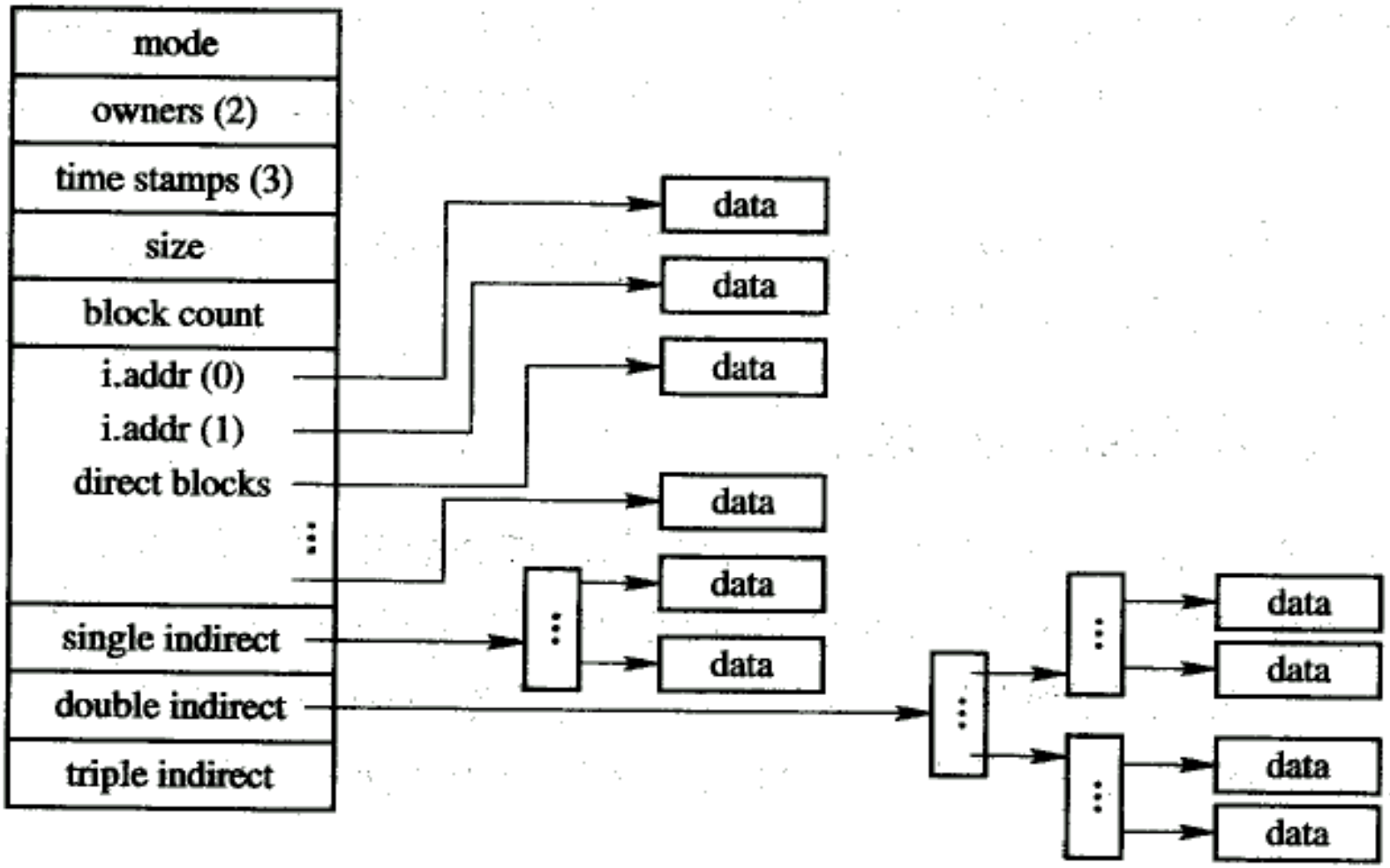
(3) 索引分配

3) 混合索引方式

UNIX文件系统采用的是混合索引结构(综合模式)。每个文件的索引表为13个索引项。最前面10项直接登记存放文件信息的物理块号（直接寻址）

如果文件大于10块，则利用第11项指向一个物理块，假设每个盘块大小为4KB，每个盘块号占4个字节，该块中最多可放1K个盘块号（一次间接寻址）。对于更大的文件还可利用第12和第13项作为二次和三次间接寻址。

(3) 索引分配


mode
owners (2)
time stamps (3)
size
block count
i.addr (0)
i.addr (1)
direct blocks
single indirect
double indirect
triple indirect
data
data
data
data
data
data
data
data
data
data

第六章 文件系统

6.1 文件系统概述

6.2 文件的物理结构

6.3 文件存储空间的管理

6.4 文件目录

6.5 文件共享和访问控制

6.3 文件存储空间的管理

- 文件管理要解决的重要问题之一是如何为新创建的文件分配存储空间。
- 存储空间的基本分配单位是磁盘块。
- 其分配方法与内存的分配有许多相似之处，即同样可采取连续分配方式或离散分配方式。
- 系统应为分配存储空间而设置相应的数据结构；其次，系统应提供对存储空间进行分配和回收的手段。

6.3 文件存储空间的管理

文件存储空间的管理方法

- 空闲分区表
- 空闲链表法
- 位示图
- 成组链接法

6.3.1 空闲分区表

- 空闲表法属于连续分配方式，它为每个文件分配一块连续的存储空间，即*系统也为外存上的所有空闲区建立一张空闲表，每个空闲区对应于一个空闲表项*，其中包括表项序号、该空闲区的第一个盘块号、该区的空闲盘块数等信息。

序　　号	第一空闲盘块号	空闲盘块数
1	2	4
2	9	3
3	15	5
4	—	—

6.3.1 空闲分区表

- 适合于*可变大小分区的连续分配*方式。
- 为文件分配存储空间时，首先顺序查找空闲分区表中的各个表项，直至找到第一个大小适合的空闲分区。
- 可以采用首次适应分配算法、最佳适应分配算法等。
- 然后，将该分区分配给文件，同时修改空闲分区表，删除相应表项。
- 当删除文件释放出空间时，系统回收其存储空间，合并相邻空闲分区.

6.3.1 空闲分区表

- 实现简单。对于最佳适应分配算法，可以将各空闲分区按照（长度）从小到大的顺序进行排列，再利用有效的查找算法，能很快找到需要大小的空闲分区。
- 但是，当存储空间中的空闲分区分布较分散且数量较多时，空闲分区表将会很大。需要很大的内存空间，会降低空闲分区表的检索速度。

6.3.1 空闲分区表

- 对交换分区一般都采用连续分配方式。
- 对于文件系统，当文件较小(1～4个盘块)时，仍采用连续分配方式，为文件分配相邻接的几个盘块；
- 当文件较大时，便采用离散分配方式。

6.3.2 空闲链表法

- **用专门的空闲分区表登记空闲分区信息会浪费一定的存储空间，而且不适合登记分散且数目很多的空闲分区，不利于基于存储块的链接文件和索引文件的存储空间分配。**
- 空闲链表法是将所有空闲盘区拉成一条空闲链。根据构成链所用基本元素的不同，可把链表分成两种形式：
  - **空闲盘块链**
  - **空闲盘区链**

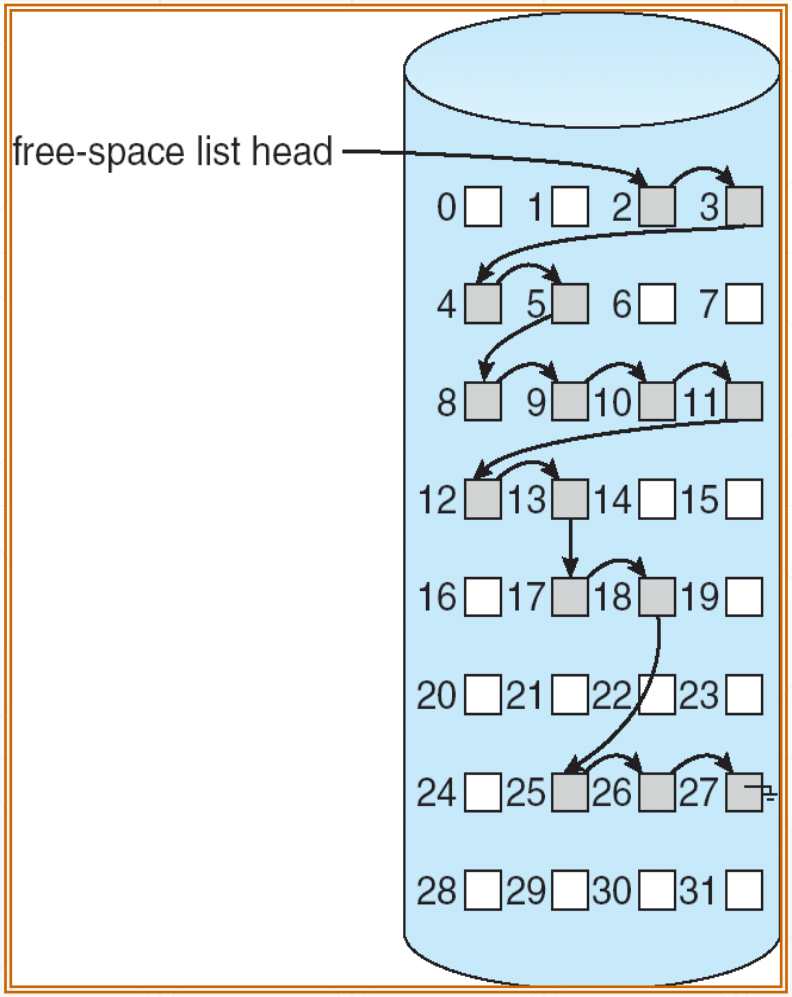
6.3.2 空闲链表法

1）空闲盘块链

- (1) 将磁盘上的所有空闲空间，*以盘块为单位拉成一条链*。
- 当用户因创建文件而请求分配存储空间时，系统从链首开始，依次摘下适当数目的空闲盘块分配给用户。
- 当用户因删除文件而释放存储空间时，系统将回收的盘块依次插入空闲盘块链的末尾。
- 这种方法的优点：用于分配和回收一个盘块的过程非常简单

6.3.2 空闲链表法

1）空闲盘块链



6.3.2 空闲链表法

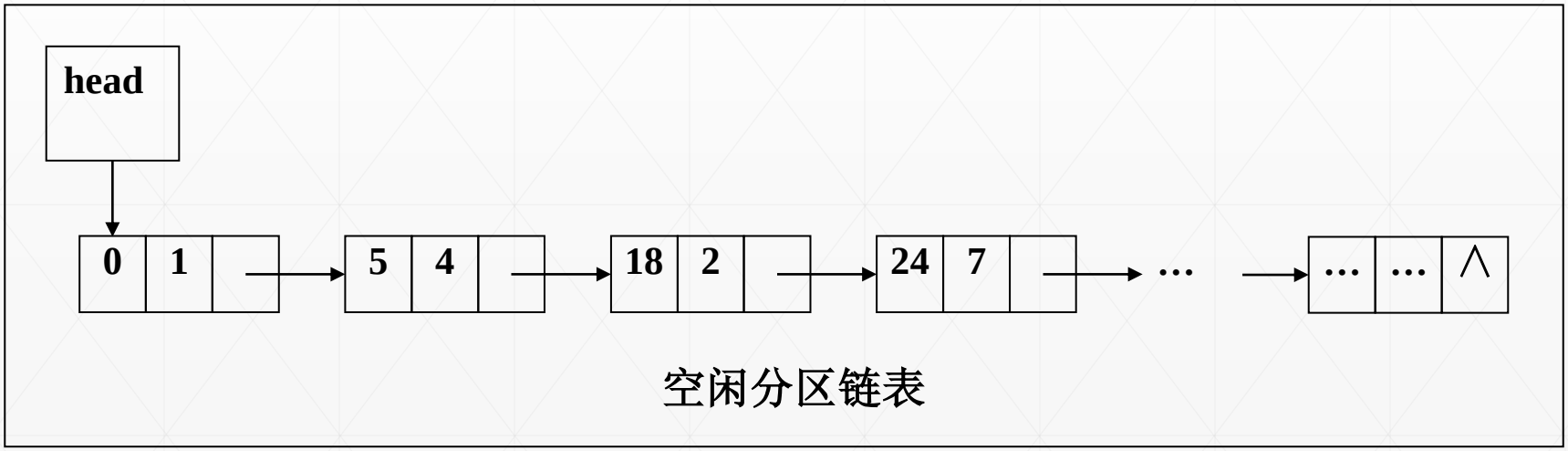
2) 空闲盘区链

- 将磁盘上的所有空闲盘区(每个盘区可包含若干个盘块)拉成一条链。
- 在每个盘区上含有用于指示下一个空闲盘区的指针和能指明本盘区大小(盘块数)的信息。
- 分配盘区的方法与内存的动态分区分配类似，通常采用首次适应算法。
- 在回收盘区时，同样也要将回收区与相邻接的空闲盘区相合并。

6.3.2 空闲链表法

2) 空闲盘区链

- 为了提高对空闲盘区的检索速度，可以采用显式链接方法，亦即，在内存中为空闲盘区建立一张链表。
- 每个分区结点内容：起始盘块号、盘块数、指向下一个空闲盘区的指针。



空闲分区链表

6.3.2 空闲链表法

2) 空闲盘区链：问题

- 一段时间以后，可能会使空闲分区链表中包含太多小分区，使文件分配到的存储空间过分离散。
- 删除一个由许多离散小分区组成的文件时，将回收的小分区链接到空闲分区链表中需要很长时间。
- 若一个文件申请连续存储空间，则需要花费较长的时间查找相邻的空闲分区。
- 因此，这种空闲空间组织方法***适合于非连续存储文件。***

6.3.3 位示图

- 利用二进制位0、1表示存储空间中存储块的使用状态。空闲分区:0，已分配分区:1（或者相反）。
- 磁盘上的所有盘块都有一个二进制位与之对应，这样，由所有盘块所对应的位构成一个集合，称为位示图。
- 通常可用$m \times n$个位数来构成位示图，并使$m \times n$等于磁盘的总块数。
- 位示图也可描述为一个二维数组map:

```
Var map: array of bit;
```

6.3.3 位示图

（1） 位示图描述

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	1	0	0	0	1	1	1	0	0	1	0	0	1	1	0
2	0	0	0	1	1	1	1	1	1	0	0	0	0	1	1	1
3	1	1	1	0	0	0	1	1	1	1	1	1	0	0	0	0
4																
⋮																
16																

6.3.3 位示图

(2) 盘块的分配

- 根据位示图进行盘块分配时，可分三步进行:
- (1) 顺序扫描位示图，从中找出一个或一组其值为“0”的二进制位(“0”表示空闲时)。
- (2) 将所找到的一个或一组二进制位转换成与之相应的盘块号。
  - 假定找到的其值为“0”的二进制位位于位示图的第i 行、第j列，则其相应的盘块号应按下式计算:
  - b = n(i - 1) + j
- (3) 修改位示图，令map[i,j]=1。

6.3.3 位示图

(3) 盘块的回收

盘块的回收分两步：

- (1) 将回收盘块的盘块号转换成位示图中的行号和列号。转换公式为：
  - i = (b - 1)DIV n + 1
  - j = (b - 1)MOD n + 1
- (2) 修改位示图。令map[i,j] =0。

6.3.3 位示图

位示图 优点

- 可以容易地找到一个或一组连续的空闲分区。
  - 例如，我们需要找到8个相邻接的空闲盘块，这只需在位示图中找出8个其值连续为“0”的位即可。
- 一个位示图需要占用的存储空间大小为：

磁盘容量（字节数）/ （8 * 数据块大小）

- 对于容量较小的磁盘，位示图占用的空间会很小。

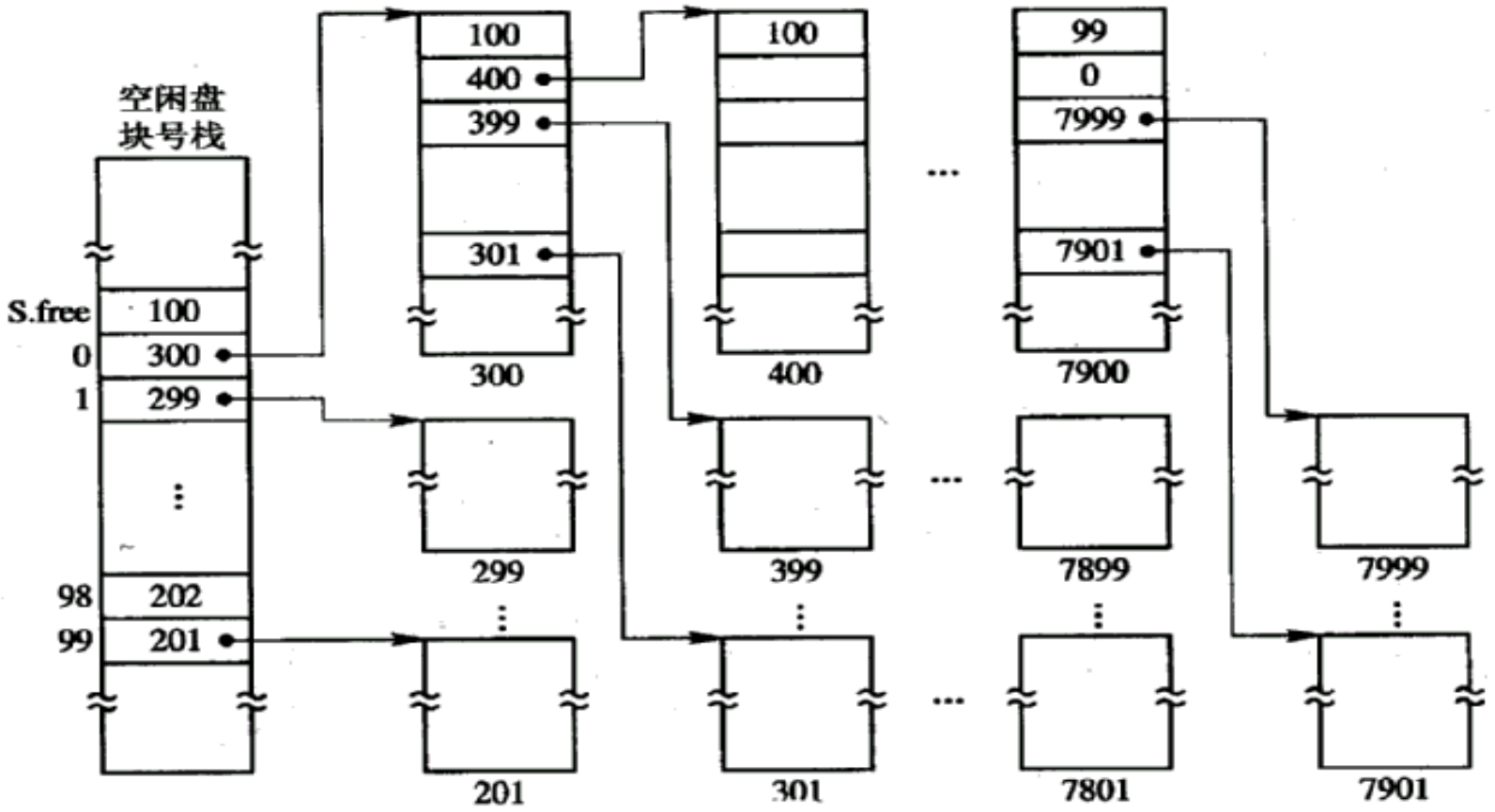
6.3.3 位示图

位示图 缺点

- 但是，对于一个16GB的磁盘，若数据块大小为512字节，则位示图大小为4MB，大约需要占用8000个磁盘块的存储空间。
- 很难一次性将该位示图全部装入内存。即使内存足够大，可以存放全部或绝大部分位示图数据，搜索一个很大的位示图将会降低文件系统的性能。
- 尤其当磁盘空间快用完，剩下的空闲磁盘块很少时，文件系统的性能将严重降低。

6.3.4 成组链接法

- 空闲盘块的组织



6.3.4 成组链接法

- 空闲盘块的分配与回收

- 当系统要为用户分配文件所需的盘块时，应调用盘块分配过程来完成。

1、首先检查空闲盘块号栈是否上锁（互斥访问），如未上锁，便从栈顶取出一空闲盘块号，将与它对应的盘块分配给用户，然后将栈顶指针下移一格。

2、若该盘块号已是栈底，即S.free(0)，这是当前栈中最后一个可分配的盘块号。

6.3.4 成组链接法

- 空闲盘块的分配与回收

3、由于在该盘块号所对应的盘块中记有下一组可用的盘块号，因此，应调用磁盘读过程，将栈底盘块号所对应盘块的内容读入栈中，作为新的盘块号栈的内容，并把原栈底对应的盘块分配出去(其中的有用数据已读入栈中)。

4、然后，再分配一相应的缓冲区(作为该盘块的缓冲区)。

5、最后，把栈中的空闲盘块数减1并返回。

6.3.4 成组链接法

- 空闲盘块的分配与回收

在系统回收空闲盘块时，应调用盘块回收过程进行回收：

将回收盘块的盘块号记入空闲盘块号栈的顶部，并执行空闲盘块数加1的操作。

当栈中空闲盘块号数目已达到100时，表示栈已满，便将现有栈中的100个盘块号，记入新回收的盘块中，再将其盘块号作为新的栈底。

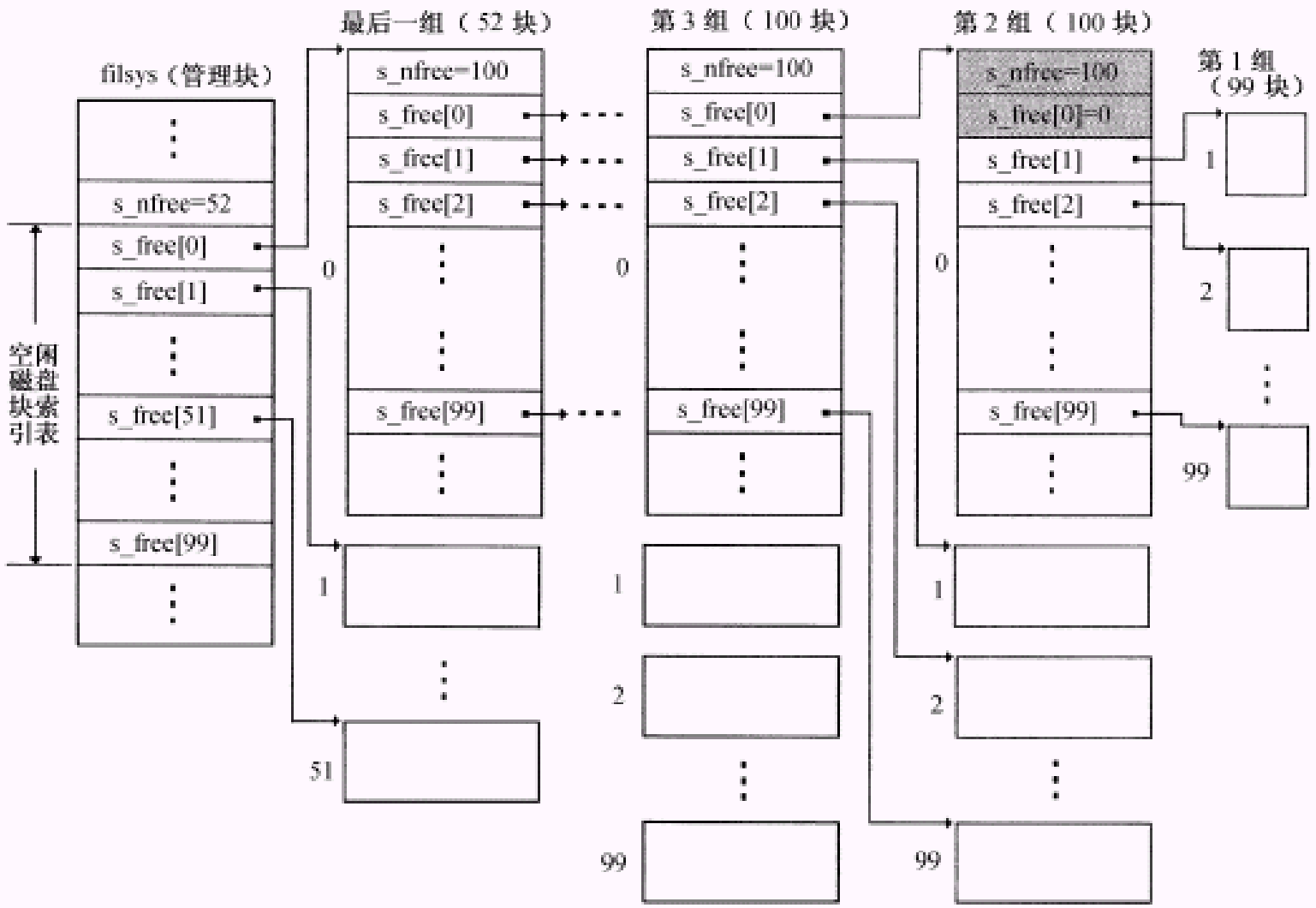
6.3.4 成组链接法

例题

参看右图，在UNIX系统中，现有某进程的文件要释放四个物理块，其块号为150#，152#，172#，177#，试给出其释放过程和释放后的空闲盘块号栈filsys的情况；其后，又有一个文件要分配6个空闲块，试给出其分配过程和分配后的filsys的情况。

6.3.4 成组链接法

例题



UNIX 对空闲磁盘块的成组链接法

第六章 文件系统

6.1 文件系统概述

6.2 文件的物理结构

6.3 文件存储空间的管理

6.4 文件目录

6.5 文件共享和访问控制

6.4 文件目录

- 1、文件目录管理
- 对目录管理的要求如下：
- (1)实现“按名存取”。
- (2) 提高对目录的检索速度。
- (3) 文件共享。
- (4) 允许文件重名。
- 文件控制块（FCB）：用于描述和控制文件的数据结构
- 文件目录：文件控制块的有序集合。

文件控制块（FCB）

文件控制块的内容

从文件管理角度看，文件由FCB和文件体（文件本身）两部分组成。

- 文件控制块（FCB-File Control Block）
  - 文件控制块是操作系统为管理文件而设置的数据结构，存放了文件的有关说明信息，记录了系统对文件进行管理所需要的全部信息。
  - FCB是文件存在的标志。
  - FCB保存在文件目录中，一个FCB就是一个目录项。

文件控制块（FCB）

文件控制块的内容

- 基本信息：文件名、文件类型等；
- 地址信息：*卷*（存储文件的设备）、*起始地址*（起始物理地址）、*文件长度*（以字节、字或块为单位）等。
- 访问控制信息：文件所有者、访问信息（用户名和口令等）、合法操作等；
- 使用信息：创建时间、创建者身份、当前状态、最近修改时间、最近访问时间等。

FCB

文件名	文件标识符
文件结构	文件类型
文件组织	记录长度
当前文件大小	最大文件尺寸
文件设备	物理位置
存取控制	口令
文件建立时间	最近存取时间
最近修改时间	当前存取方式
当前的共享状态	共享访问时的等待状态
进程访问文件所用的逻辑单元号	当前的逻辑位置
访问元素的当前物理位置	下一个元素的物理位置
缓冲区大小	缓冲区地址
指向下一个FCB的指针	文件创建者
临时/永久文件	文件拥有者

文件控制块（FCB）

- 文件目录

把所有的FCB组织在一起，就构成了文件目录，即文件控制块的有序集合。

- 目录项

构成文件目录的项目（目录项就是FCB）

- 目录文件

为了实现对文件目录的管理，通常将文件目录以文件的形式保存在外存，这个文件就叫目录文件。

目录内容的组织方式及分析

- **目录项的两种组织方式：**
- 1. FCB存储全部目录内容
- 2. 存储部分目录信息，如文件名、索引节点指针等，其余部分保存在索引节点（i节点）。打开文件时将索引节点从磁盘读到内存中。

引入索引结点

文件名	索引结点编号
文件名1	
文件名2	
……	……

UNIX系统中，采用了把文件名和文件描述信息分开的方法，把文件描述信息单独形成一个数据结构，叫索引结点。

目录文件及操作

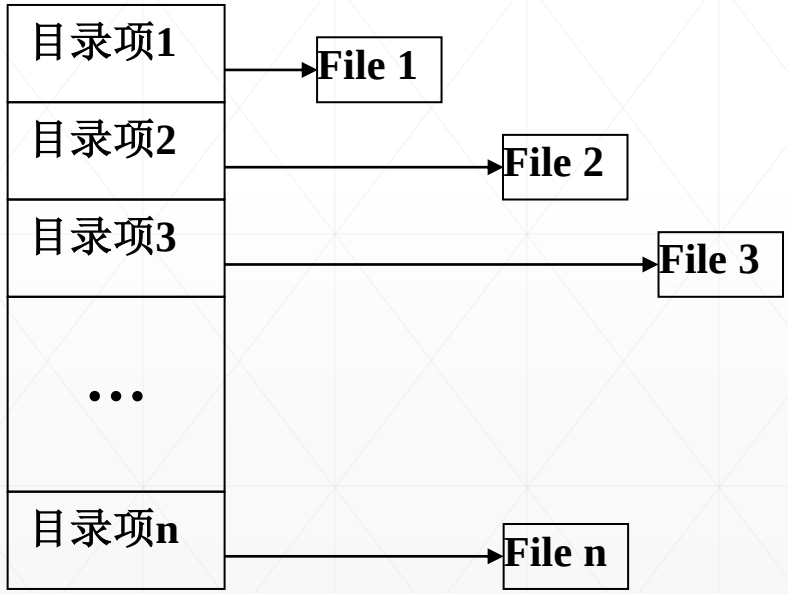
- 目录文件：一个文件目录也被看做是一个文件，即目录文件。是多个文件的目录项构成的一种特殊文件。
- 目录的操作
  - 搜索目录
  - 创建目录
  - 删除目录
  - 显示目录
  - 修改目录

目录结构

- 单级目录结构
- 两级目录结构
- 层次目录结构： 树型目录、无循环图

单级目录结构

- 所有用户的全部文件目录保存在一张目录表中，每个文件的目录项占用一个表项。



单级目录结构

单级目录结构分析

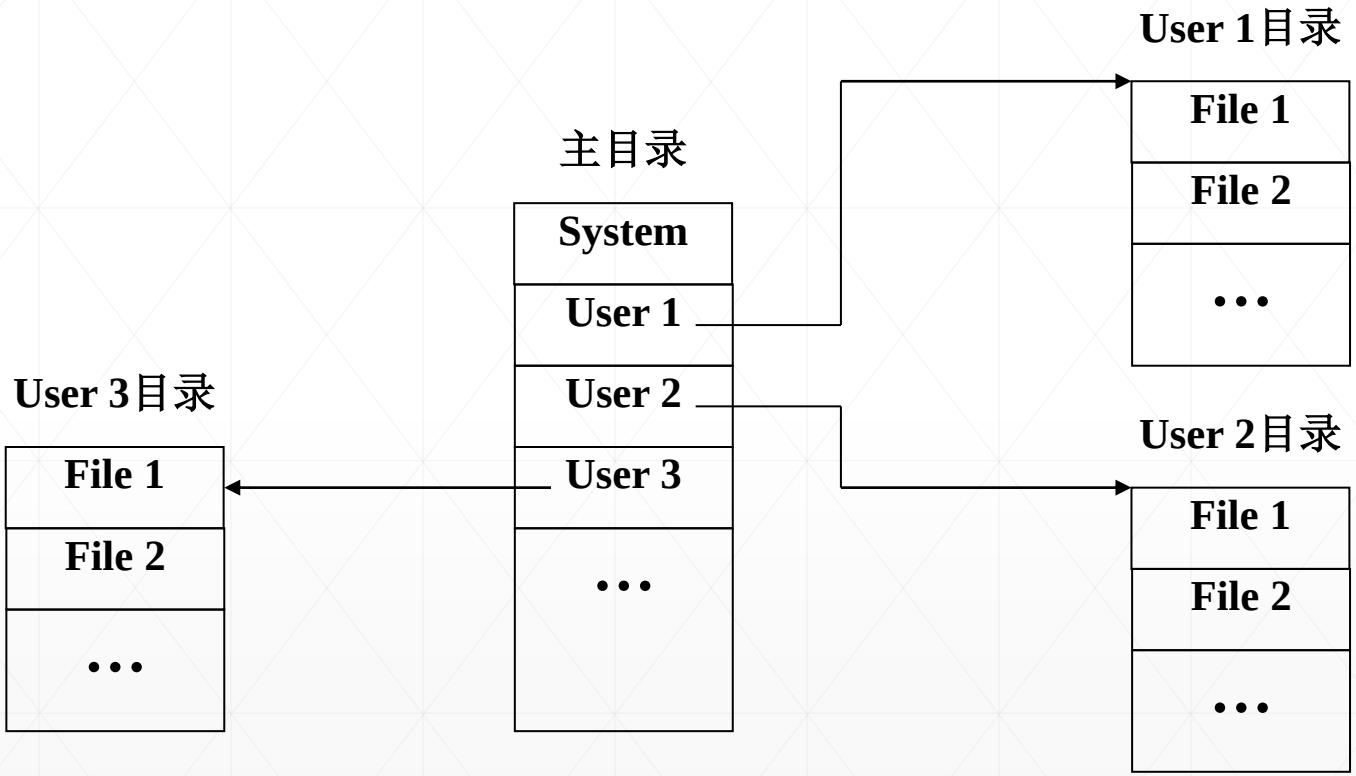
- 单级目录的优点是简单且能实现目录管理的基本功能——按名存取
- 存在下述一些缺点：
- (1) 查找速度慢
- (2) 不允许重名
- (3) 不便于实现文件共享

两级目录结构分析

- 在整个系统中建立两级目录
  - 为每个用户建立一个单独的用户文件目录（UFD）
  - 系统中为所有用户建立一个主文件目录（MFD）

两级目录结构

- 主文件目录MFD、用户文件目录UFD



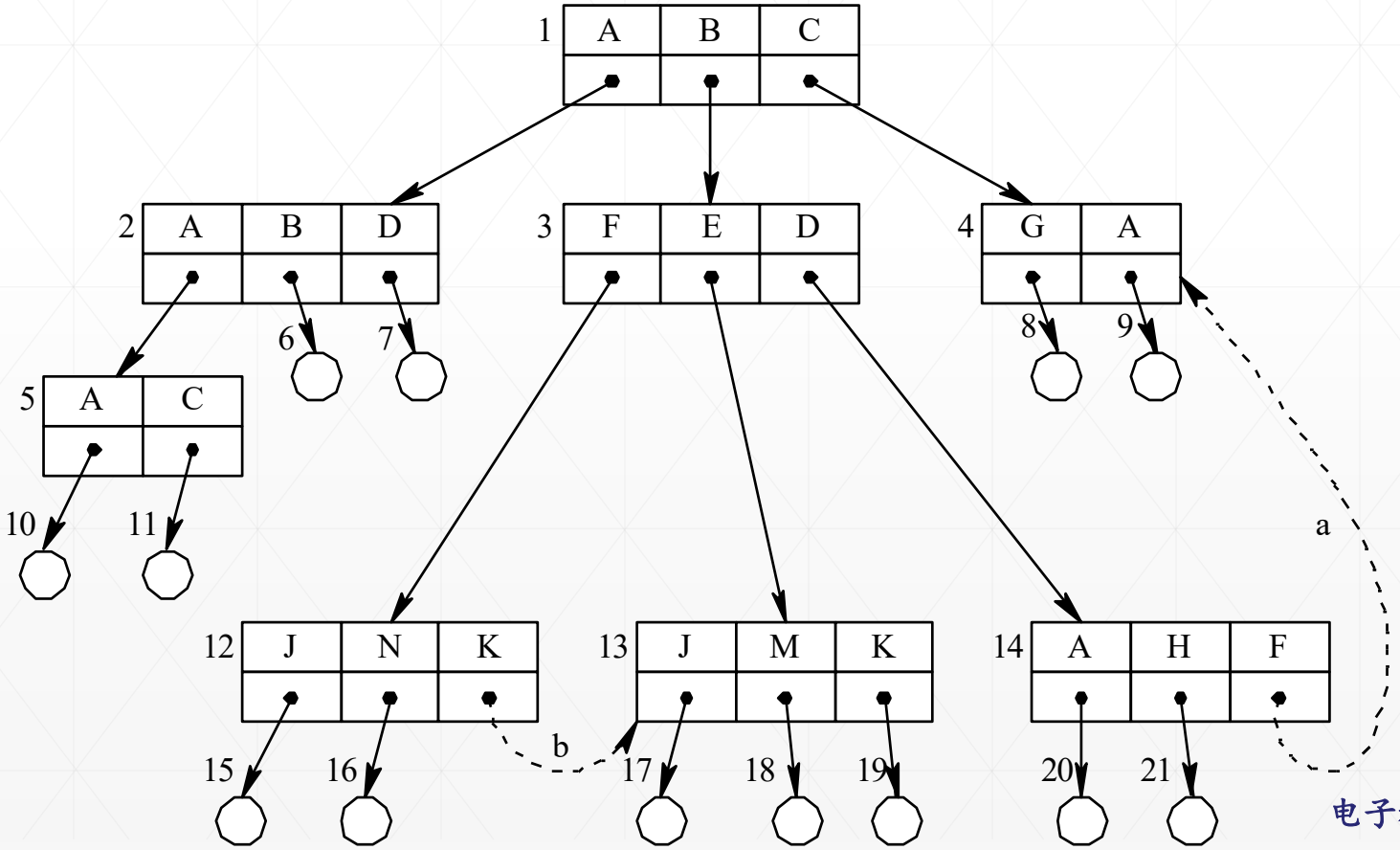
两级目录结构

两级目录结构分析

- 一定程度解决了重名问题
- 提高了文件目录检索效率
- 简单的文件共享
- **问题：**不便用户文件的逻辑分类；进一步解决重名、共享、检索效率等问题

多级目录结构

- 多级目录结构又称为树型目录结构，主目录在这里被称为根目录，把数据文件称为树叶，其它的目录均作为树的结点。



多级目录结构

路径名：从树的根（即主目录）开始，把全部目录文件名与数据文件名，依次地用“/”连接起来，即构成该数据文件的路径名（path name）。

- **系统中的每一个文件都有惟一的路径名。**

当前目录：为每个进程设置一个“当前目录”，又称为“工作目录”进程对各文件的访问都相对于“当前目录”而进行。

- 相对路径名
- 绝对路径名

多级目录结构

- **增加目录：**在用户要创建一个新文件时，只需查看在自己的UFD及其子目录中，有无与新建文件相同的文件名。若无，便可在UFD或其某个子目录中增加一个新目录项。
- **目录删除**采用下述两种方法处理：

(1) 不删除非空目录。

(2) 可删除非空目录。

目录查询技术

★ 对目录进行查询的方式有两种：线性检索法和Hash方法:

1）线性检索法

- 线性检索法又称为顺序检索法。

①在单级目录中，利用用户提供的文件名，用顺序查找法直接从文件目录中找到指名文件的目录项。

②在树型目录中，用户提供的文件名是由多个文件分量名组成的路径名，此时须对多级目录进行查找。

目录查询技术

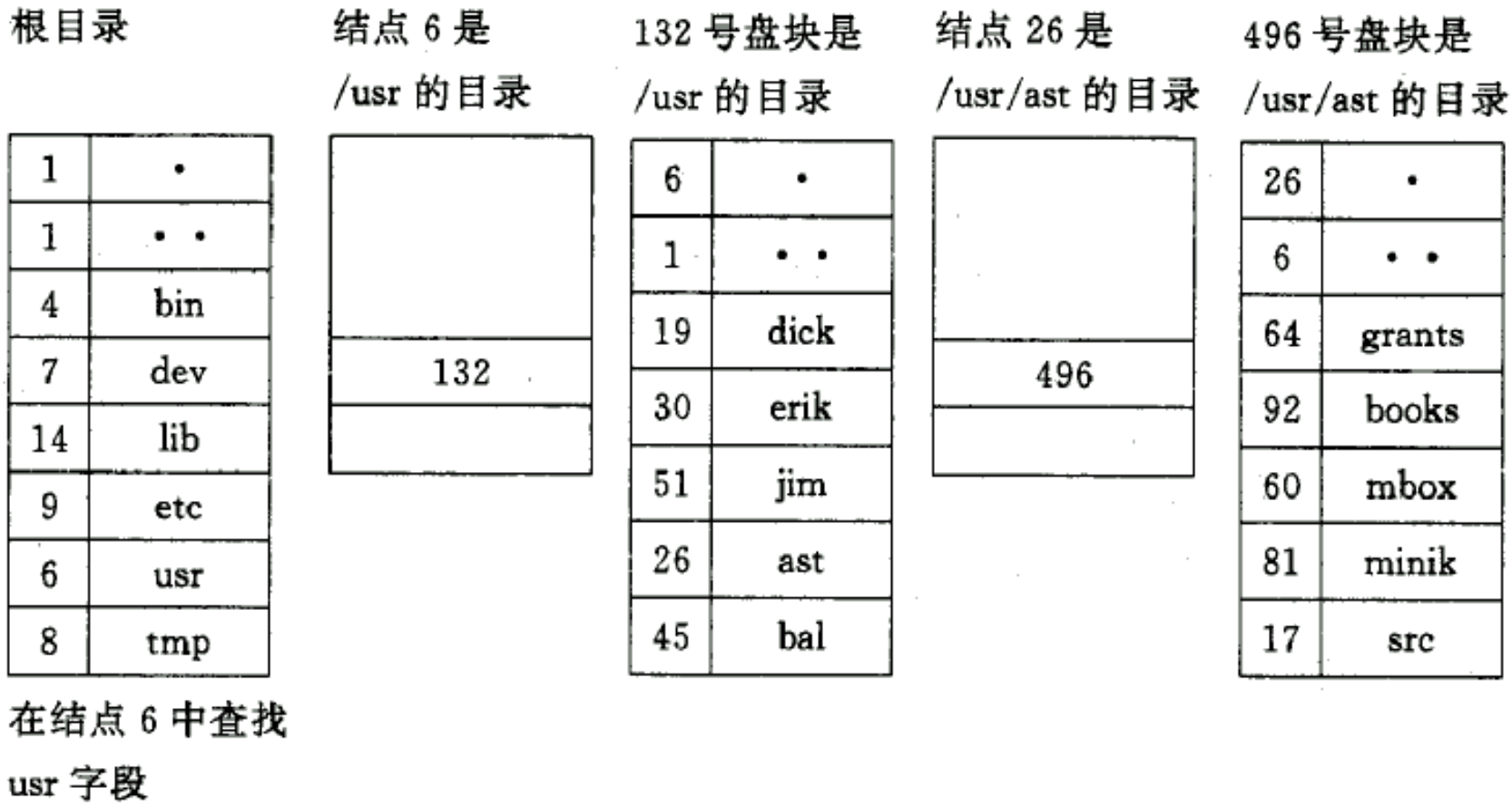


图 6－19　查找/usr/ast/mbox 的步骤

目录查询技术

- 2）Hash方法
- Hash方法: 建立了一张Hash索引文件目录，系统利用用户提供的文件名并将它变换为文件目录的索引值，再利用该索引值到目录中去查找。
- Hash方法将显著地提高检索速度。
- 在文件名中使用了通配符“*”、“?”等，系统便无法利用Hash法检索目录，因此，需要利用线性查找法查找目录。
- 在进行文件名的转换时，有可能把多个不同的文件名转换为相同的Hash值，称谓的“Hash冲突”。

目录查询技术

- Hash查找过程：

①在利用Hash值查找目录时，如果目录表中相应的目录项是空的，则表示系统中并无指定文件。

②如果目录项中的文件名与指定文件名相匹配，则表示该目录项正是所要寻找的文件所对应的目录项，故而可从中找到该文件所在的物理地址。

③如果在目录表的相应目录项中的文件名与指定文件名并不匹配，则表示发生了“Hash冲突”。

- 解决Hash冲突的方法：将其Hash值再加上一个常数（该常数应与目录的长度值互质），形成新的索引值，再返回到第一步重新开始查找。

第六章 文件系统

6.1 文件系统概述

6.2 文件的物理结构

6.3 文件存储空间的管理

6.4 文件目录

6.5 文件共享和访问控制

6.5 文件共享和访问控制

文件共享的控制

文件共享的有效控制涉及两个方面：

- 同时存取（Simultaneous Access）
- 存取权限（Access Rights）

控制同时存取

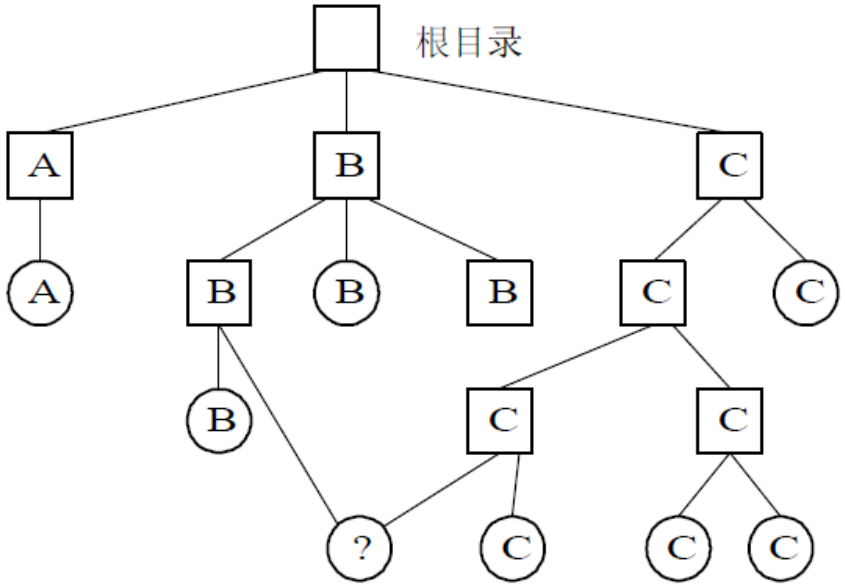
- 允许多个用户同时读文件内容，但不允许同时修改，或同时读且修改文件内容。
- 共享用户之一修改文件内容时，可以将整个文件作为临界资源，锁定整个文件，不允许其他共享用户同时读或写文件。
- 也可以仅仅锁定指定的一条记录，允许其他共享用户读/写该文件的其它记录。后者的并发性能更好。
- 控制对文件的同时存取涉及进程的同步与互斥问题。

控制存取权限

控制授权用户以合法的方式访问文件，包括：

- **执行（Execution）** — 用户可以装载并执行程序，但不允许拷贝程序内容。
- **读（Reading）— 允许用户读文件内容，包括拷贝和执行文件。**某些系统严格地将浏览文件内容和拷贝权限分开，可以控制文件只能被浏览（显示），不能被拷贝。
- **追加（Appending）** — 允许用户向文件添加数据，通常只能将数据添加到文件尾。但是，不能修改或删除文件内容。例如，超市收银员只能将新结帐的数据添加到文件中，不允许其修改或删除已有的数据。
- **更新（Updating）— 允许用户修改、删除、增加文件内容。**包括创建文件、重写文件的全部或部分内容、移动文件的全部或部分数据等操作。
- **更改权限** (Changing protection) **—一般只有文件主才能更改共享该文件的其他用户对该文件的存取权限。**有的系统允许文件主将更改文件存取权限赋予其他某个用户，但必须限制授权用户更改的权限范围。
- **删除** (Deletion) **允许用户删除文件**

文件共享的实现



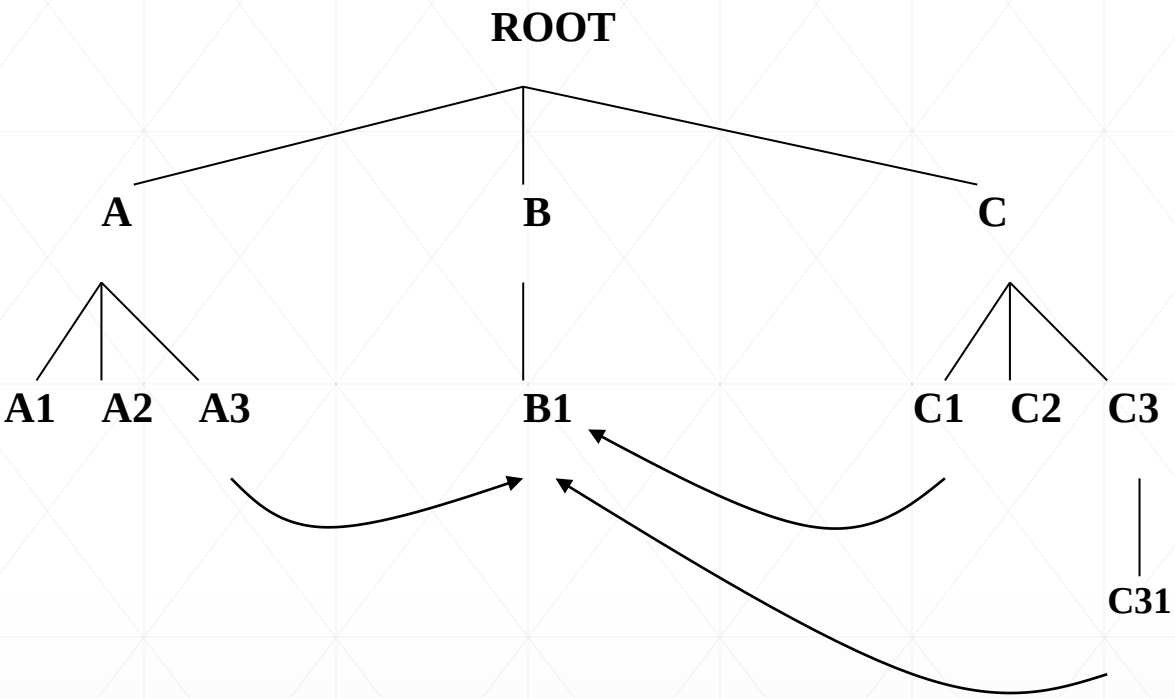
- 在树型结构的目录中，当有两个(或多个)用户要共享一个子目录或文件时，必须将共享文件或子目录链接到两个(或多个)用户的目录中，才能方便地找到该文件。此时该文件系统的目录结构已不再是树型结构，而是个有向非循环图。

文件共享的实现

- 实现文件共享的实质就是可以从不同地方打开同一个文件
- 打开文件的首要步骤就是找到文件的目录项，读取文件在外存的起始地址。
- 实现文件共享的方式：
  - 利用链接目录项实现法、
  - 利用索引节点实现法
  - 利用符号链实现法等。

1.链接目录项实现文件共享

- **文件目录项中设置一个链接指针，用于指向共享文件的目录项。**
- **访问文件时，根据链接指针内容找到共享文件的目录项，读取该目录项中文件起始位置等信息，操作该文件。**
- **每当有用户（进程）共享文件时，共享文件目录项中的“共享计数”加1；当用户不再共享该文件，撤消链接指针时，“共享计数”减1。**
- ***只有当共享文件用户数为1时，才能删除共享文件。***



链接目录项实现文件共享

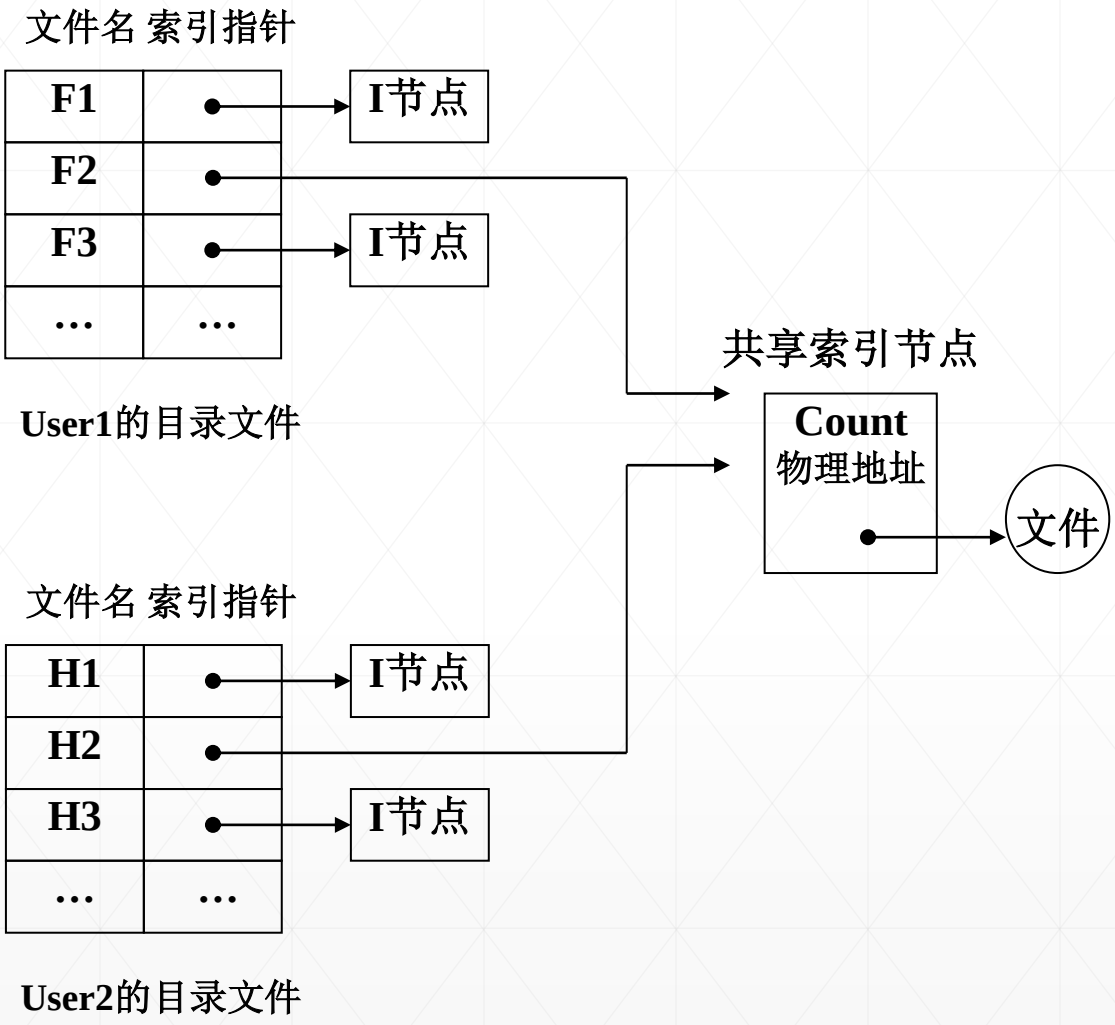
2.利用索引节点实现文件共享

- 文件的物理地址及其它的文件属性等信息，不再是放在目录项中，而是放在索引结点中。在文件目录中只设置文件名及指向相应索引结点的指针。
- 由任何用户对文件进行Append 操作或修改，所引起的相应结点内容的改变(例如，增加了新的盘块号和文件长度等)，都是其他用户可见的，从而也就能提供给其他用户来共享。

2.利用索引节点实现文件共享

- UNIX操作系统的文件目录项中只包含文件名和指向索引节点的指针，文件的物理地址及其它说明信息保存在索引节点中。

- 可以***通过共享文件索引节点来共享文件***，即当用户需要共享文件时，在自己的文件目录中新建一个目录项，为共享文件命名(也可用原名)，并将索引节点指针指向共享文件的索引节点。



利用索引节点实现文件共享

2.利用索引节点实现文件共享

- 在索引结点中还应有一个链接计数count，用于表示链接到本索引结点(亦即文件)上的用户目录项的数目。
  - 当用户C创建一个新文件时，他便是该文件的所有者，此时将count置1。
  - 当有用户B要共享此文件时，在用户B 的目录中增加一目录项，并设置一指针指向该文件的索引结点，此时，文件主仍然是C，count=2。

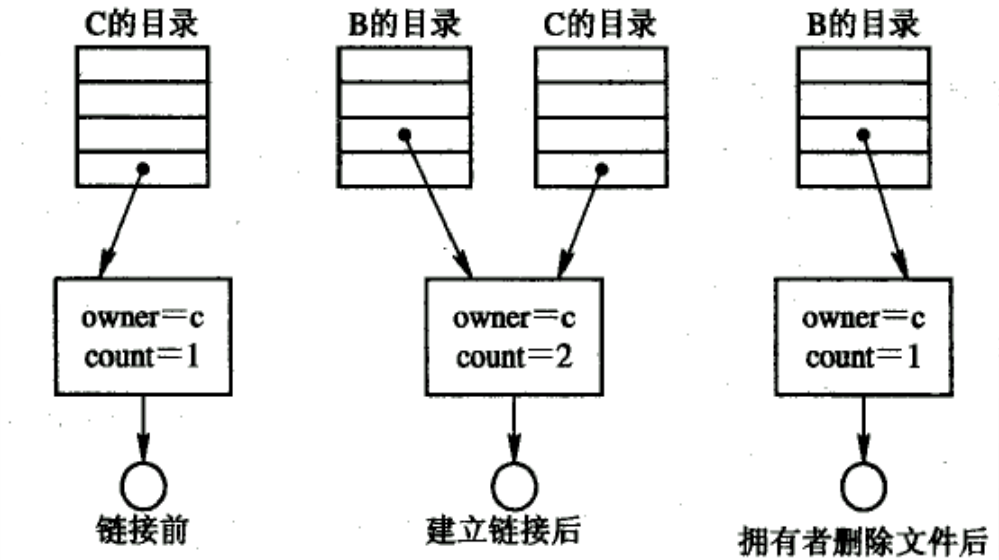


图 6－25　进程 B 链接前后的情况

- *如果用户C不再需要此文件，是否能将此文件删除呢?*

3. 利用符号链实现文件共享

- 为使B能共享C的一个文件F，可以由系统创建一个LINK类型的新文件，也取名为F并将F写入B的目录中，以实现B的目录与文件F的链接；在新文件中只包含被创文件F的路径名。这样的链接方法被称为符号链接.
- 新文件中的路径名，则只被看作是符号链。当B要访问被链接的文件F且正要读LINK类新文件时，将被OS截获， OS根据新文件中的路径名去读该文件，于是就实现了用户B对文件F的共享。

3. 利用符号链实现文件共享

- 在利用符号链方式实现文件共享时，只是文件主才拥有指向其索引结点的指针,而共享该文件的其它用户，则只有该文件的路径名，并不拥有指向其索引结点的指针。
- 符号链方式优点：能连接任何机器上的文件。
- 每增加一个连接，就增加一个文件名，各用户使用自己的名字去共享文件。
- 缺点：备份可能会产生多个拷贝。

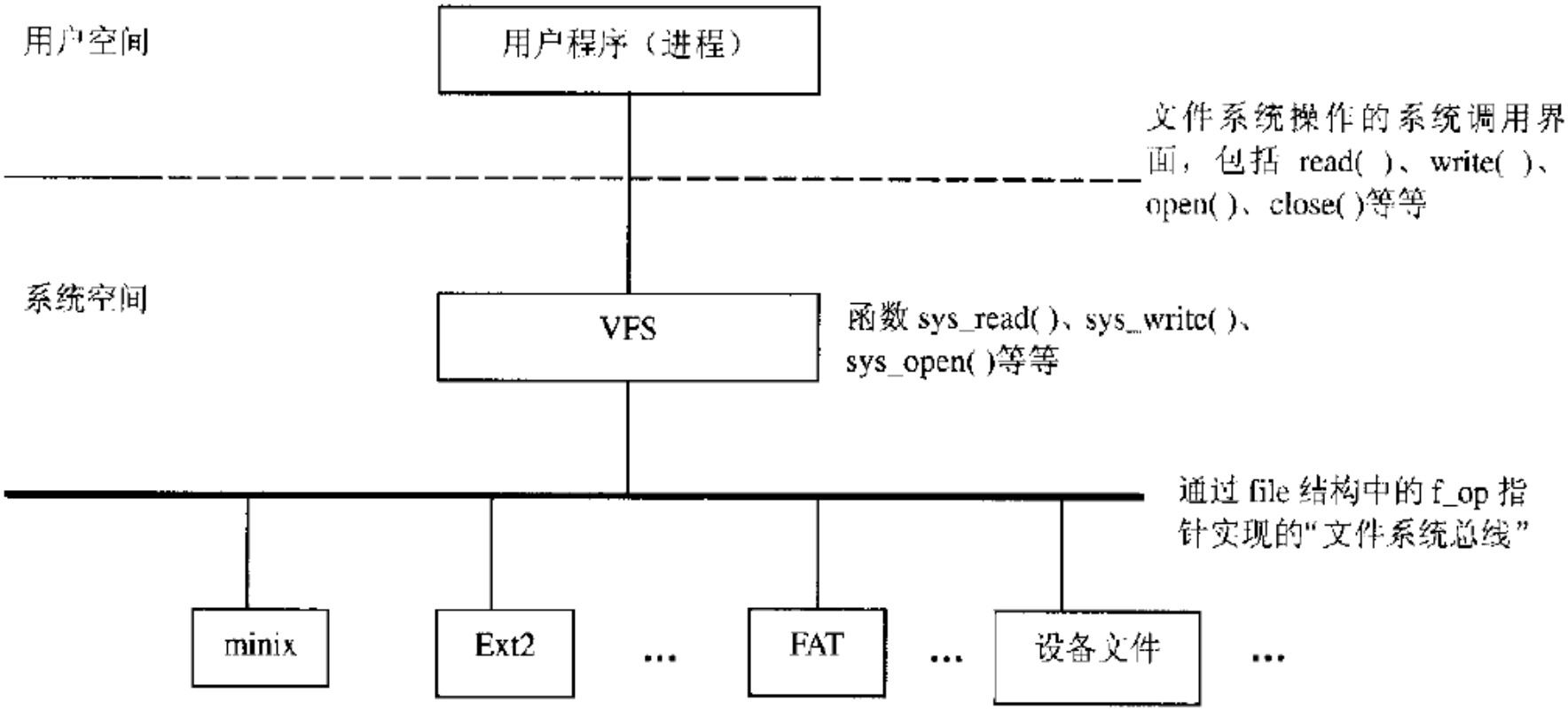
利用URL实现文件共享

- 统一资源定位器URL (Uniform Resource Locator)是Internet上用来链接超文本文件的一种方法。
- 它可以链接同一台计算机中的本地文件，也可链接Internet中任何主机上的远程文件。
- 一个完整的URL包括访问文件的方法（协议）、文件所在的主机域名、目录路径名和文件名几部份。例如，
  http://www.uestc.edu.cn/templates/index2k3/index.html

文件保护

- 不同对象允许实施的操作各不相同。例如，文件可施加读、写、执行等操作，信号量只能施加wait()和signal()操作。
- 因此，***系统为所有对象设置一个允许进程实施操作的操作集***，任何对对象的操作必须符合操作集中的规定，防止未授权进程访问对象。

Linux文件系统


用户空间
用户程序（进程）
文件系统操作的系统调用界面，包括 read( )、write( )、open( )、close( )等等
系统空间
VFS
函数 sys_read( )、sys_write( )、sys_open( )等等
通过 file 结构中的 f_op 指针实现的"文件系统总线"
minix
Ext2
...
FAT
...
设备文件
...

总结

- 重点：
  - 文件和文件系统的概念；文件是具有符号名的信息（数据）项的集合；文件是具有符号名的记录的集合；文件是具有符号名的数据项的集合。
  - 文件逻辑结构；顺序文件、索引顺序文件、索引文件、HASH文件；
  - 磁盘存储分配方法；连续分配、链接分配、索引分配；
  - 文件目录及文件控制块；文件存储器分区和空间管理。
  - 磁盘存储管理；位示图、空闲链表、索引。
- 难点：
  - 磁盘存储管理；
  - 位示图、空闲链表、

练习题：1

1.存放在某个磁盘上的文件系统，对于采用混合索引分配方式，其FCB中共有13项地址项，第0～9个地址项为直接地址，第10个地址项为一次间接地址，第11个地址项为二次间接地址，第12个地址项为三次间接地址。如果每个盘块的大小为512字节，盘块号需要3个字节来描述，则每个盘块最多存放170个盘块地址：（10分）

(1) 该文件系统允许的最大长度是多少？

(2) 将文件的字节偏移量5000、15000、150000转换为物理块号和块内偏移量。

(3) 假设某文件的索引结点已在内存中，但其他信息均在外存，为了访问该文件中某个位置的内容，最多需要几次访问磁盘？

- 答:
- （1）文件的最大长度为:
- 10+170+1702+1703=4942080块=2471040KB（2分）
- （2）
- 5000/512得商9，余数为392。即逻辑块号为9，块内偏移为392。故可直接从该文件的FCB的第9个地址处得到物理盘块号，块内偏移为392。（2分）
- 15000/512得商为29，余数为152。即逻辑块号为29，块内偏移为152。由于10≤29<10+170,而29-10=19，故可从FCB的第10个地址项，即一次间址项中得到一次间址块；并从一次间址块的19项中获得对应的物理盘块号，块内偏移为152。（2分）
- 150000/512得商为292，余数为496。即逻辑块号为292，块内偏移为496。由于10+170≤292，故可从FCB的第11个地址项，即二次间址项中获得第1个一次间址块；并从该一次间址块的112项中获得对应的物理盘块号，块内偏移为496。（2分）
- (3) 由于文件的索引结点已在内存，为了访问文件中的某个位置的内容，最少需要1次访问磁盘（即通过直接地址直接读文件盘块），最多需要4次访问磁盘（第一次是读三次间址块，第二次读二次间址块，第三次读一次间址块，第四次是读文件盘块）（2分）

练习题：2

2．有一个磁盘组共用10个盘面，每个盘面上有100个磁道，每个磁道有16个扇区，假定以扇区为单位，若使用位示图管理磁盘空间，问位示图需要占多少空间？若空闲表的每个空闲表项占用5个字节，问什么时候空闲表大于位示图？

课后作业

教材第266页

练习题2、5、6、16、21

教材第296页

练习题12、13、14