



第6章 文件系统

6.1 文件系统概述

6.2 文件的结构与存取方式

6.3 文件目录

6.4 空闲存储空间的管理

6.5 文件的使用、共享和保护

6.6 小结



6.1 文件系统概述

一、为什么要引入文件？为什么引入文件系统？

- ✓ 长期保存（大量的）数据
- ✓ 方便用户使用（包括共享）



6.1 文件系统概述

二、什么是文件？文件系统有何作用？

1. 什么是文件？

文件是**有名字的**记录在**外存中**的一组有逻辑意义的数据项的序列

- ✓ 名字：文件名
- ✓ 数据项：构成文件内容的基本单位
- ✓ 长度：文件的字节数
- ✓ 文件内容的含义：由文件的建立者和使用者解释

什么是文件？

编号： 0 1 i n-1



读写指针

- 各数据项之间具有顺序关系



什么是文件？

文件命名：

每个操作系统都有自己的文件命名规则

- ✓ 长度
- ✓ 数字和特殊字符
- ✓ 是否大小写敏感
- ✓ 文件扩展名（一个或多个）

例子：.bak .c .exe .gif

.hlp .html .mpg .o

.doc .java .txt .zip



6.1 文件系统概述

2. 什么是文件系统

文件系统是OS中用来管理文件的那一部分软件

文件系统的功能：

- ✓ 统一管理文件的存储空间，实施存储空间的分配与回收
- ✓ 实现文件信息的共享，提供文件的保护和保密措施
- ✓ 实现文件的**按名访问**

访问的透明性：用户不关心文件的物理位置和存储结构

- ✓ 向用户提供一个方便使用的**接口**，提供对文件系统操作的命令
- ✓ 提供与I/O的**统一接口**

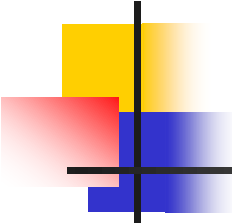


6.1 文件系统概述

3. 文件的分类

文件分类角度很多，比如：

- ✓ 按文件的性质和用途：系统文件；用户文件；库文件
- ✓ 文件中数据的形式：文本文件；二进制文件
- ✓ 文件的保护方式：只读文件；读写文件；可执行文件
- ✓ **文件的逻辑结构：流式文件；记录式文件**
- ✓ **文件的物理结构：顺序（连续）文件；链式文件；索引文件**



文件分类

Unix系统将文件分为3类:

- ✓ 普通文件(regular) : ASCII或二进制文件
- ✓ 目录文件(directory)
- ✓ 特殊文件 : 设备文件 , 管道 , 套接字 (socket) , 符号链接等



6.1 文件系统概述

三、看待文件系统的两种观点

✓ 用户观点：

文件系统如何呈现在用户面前：一个文件由什么组成，如何命名，如何保护文件，可以进行何种操作等等

尽可能方便用户使用

✓ 系统观点：

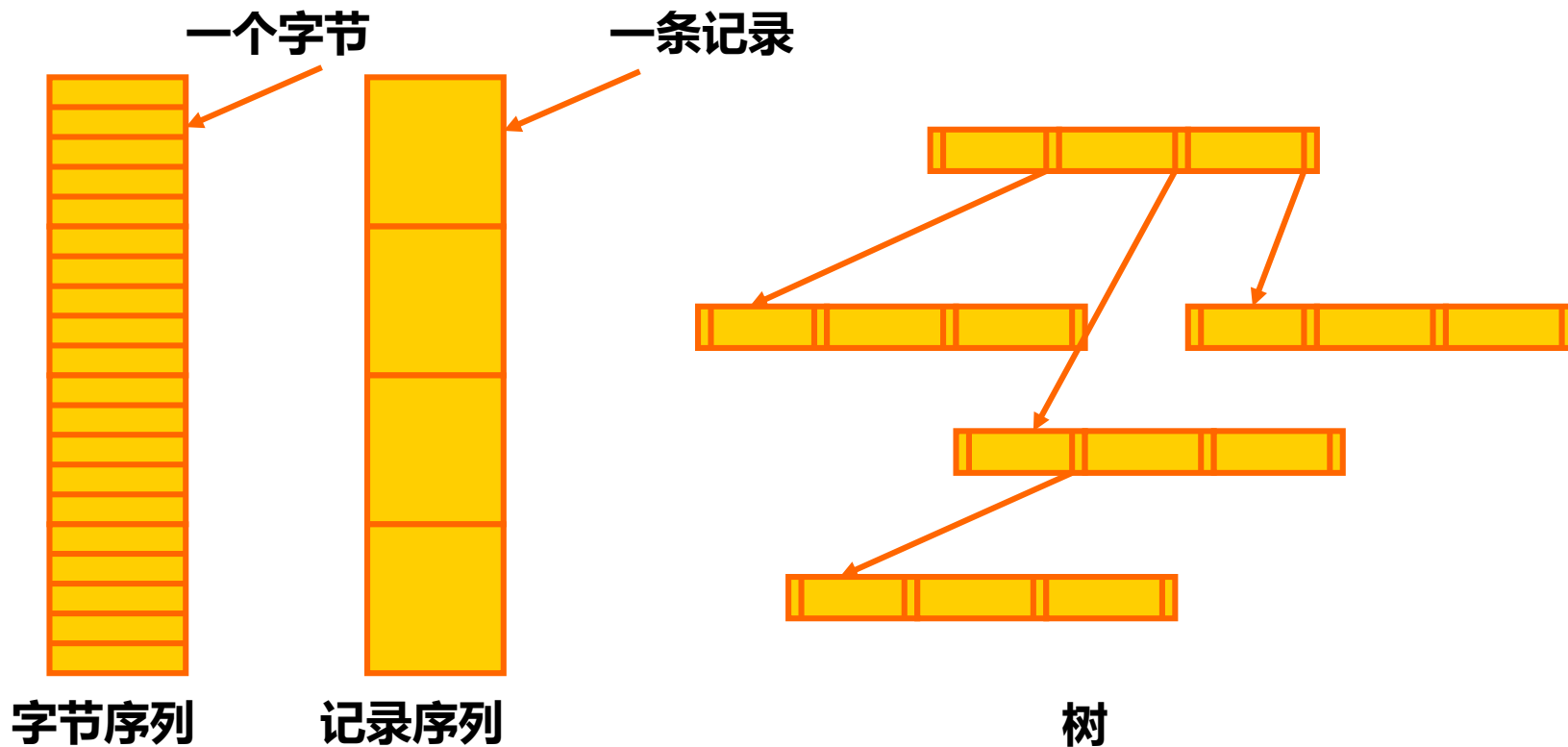
文件目录怎样实现，怎样管理文件存储空间，文件存储位置，与设备管理的接口，等等

尽可能提高效率（时间，空间）

6.2 文件的结构与存取方式

一、文件的逻辑结构

从用户的观点看文件的组织形式





文件的逻辑结构

一般分为2类：

(1) **字节流文件** (流式文件) ：无结构文件

文件：**一个无结构的字节序列，其含义由使用者解释**

Unix的所有文件都看作字节流文件

好处：非常灵活

(2) **记录文件**：有结构文件

文件：记录的序列，每条记录有其内部结构

记录：定长，不定长



6.2 文件的结构与存取方式

二、文件的物理结构

文件的存储结构，指文件在外存上的存放方式

即从系统的角度看文件的组织方式

基本结构有3种：

- ✓ 连续结构（顺序结构）
- ✓ 链式结构（串联结构）
- ✓ 索引结构



文件的物理结构

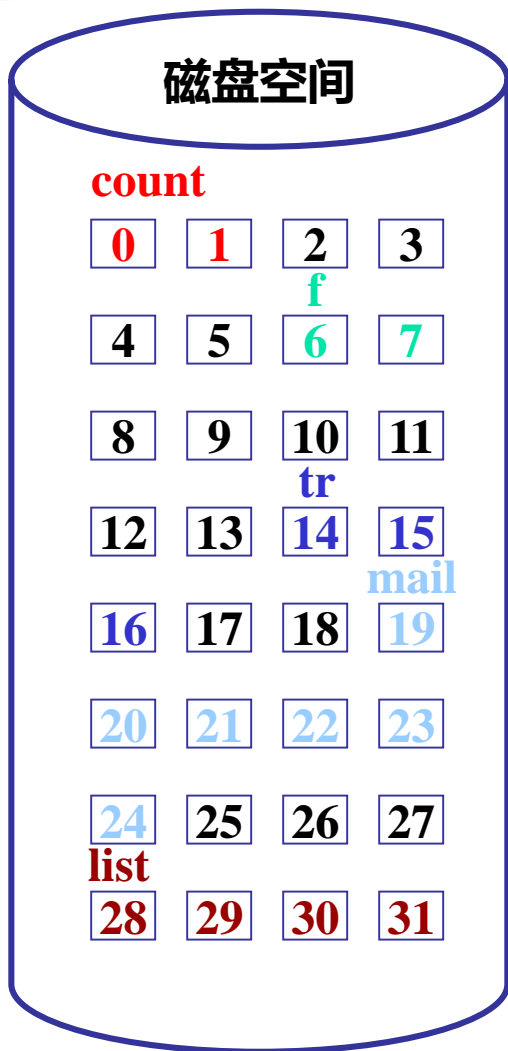
(1) 连续结构

文件的数据存放在若干连续的物理块中

优点:

- ✓ 简单，只要记住首块的地址和文件长度即可
- ✓ 支持顺序存取和随机存取
- ✓ 顺序存取速度快
- ✓ 所需的磁盘寻道时间最少

连续文件



文件目录

文件名	始址	块数
count	0	2
tr	14	3
mail	19	6
list	28	4
f	6	2



连续文件

缺点:

- ✓ 不利于文件的动态增长

若预留空间：浪费，而且预先不知道文件的最大长度

否则需要重新分配和移动

- ✓ 不利于文件内容的插入和删除
- ✓ 存在外部碎片问题



文件的物理结构

(2) 链式结构

一个文件的数据存放在若干不连续的物理块中，各块之间通过指针连接，每个物理块指向下一个物理块

优点：

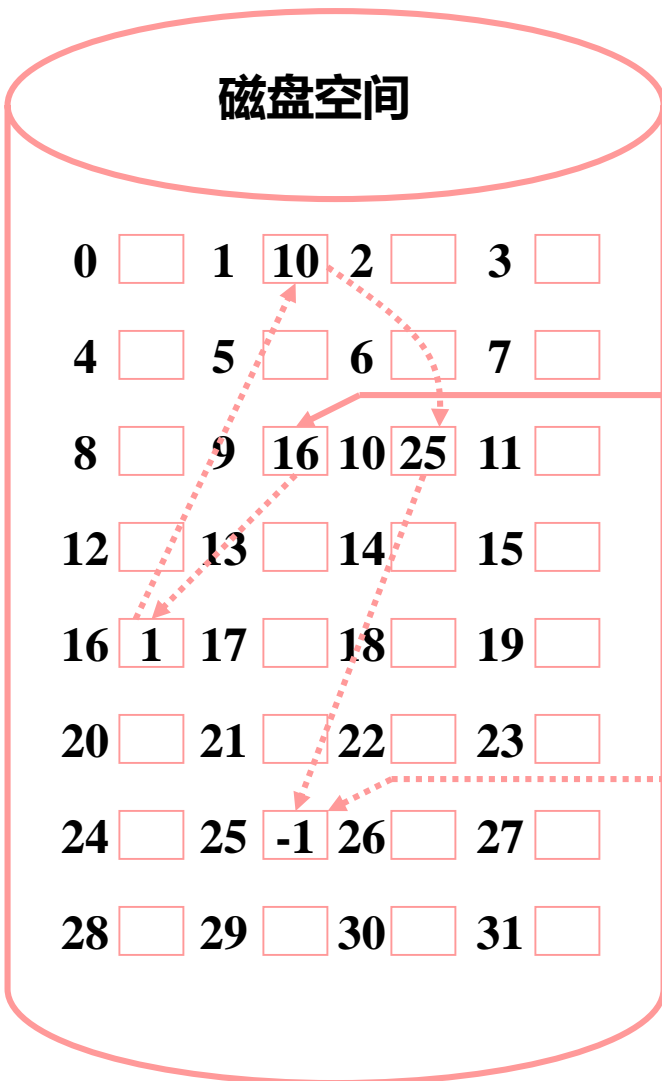
- ✓ 提高了磁盘空间利用率
- ✓ 不存在外部碎片问题
- ✓ 有利于文件的插入和删除
- ✓ 有利于文件的动态扩充

链式文件

文件目录

文件名	始址	末址
jeep	9	25

磁盘空间





链式文件

缺点：

- ✓ 随机存取相当缓慢
- ✓ 需要更多的寻道时间
- ✓ 链接指针占用一定的空间

链接结构的变形:

文件分配表FAT (File Allocation Table)

一般以簇为单位分配空间，簇由若干连续的物理块组成



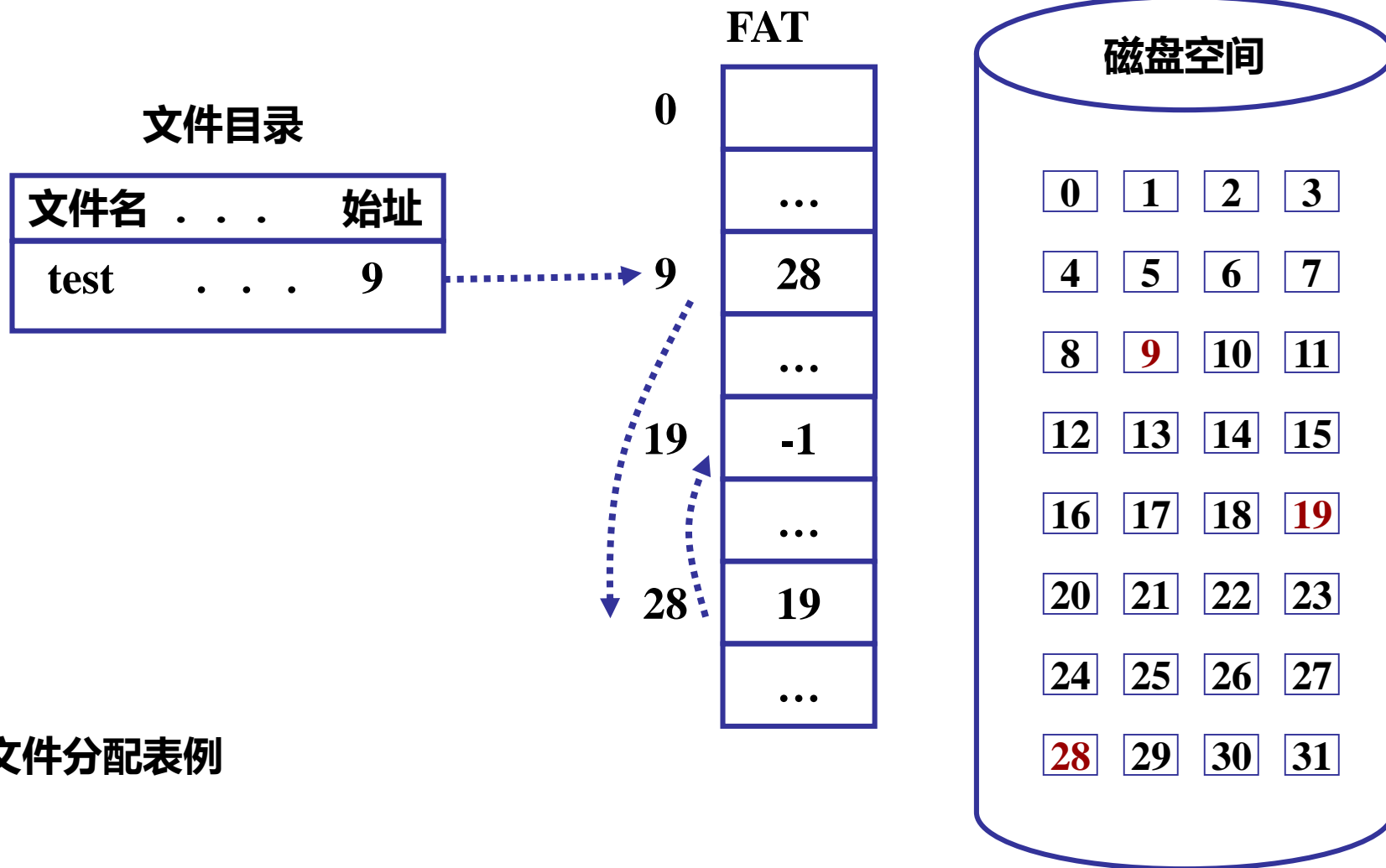
链式文件

文件分配表FAT的一种实现：

磁盘的每个分区包含一个FAT，分区中的每个盘块在其中占有1项（以块号为索引），指出文件中下一块的块号。

在目录项中包含文件首块的块号。

链式文件





文件的物理结构

(3) 索引结构

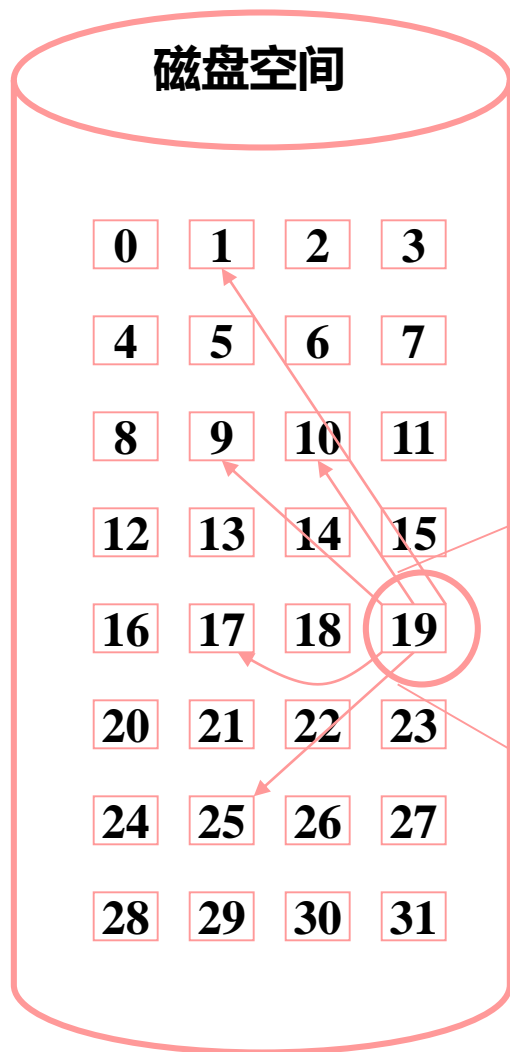
一个文件的数据存放在若干不连续的物理块中，系统为每个文件建立一个专用数据结构--索引表。

索引表存放逻辑块号与物理块号的对应关系

一个索引表就是磁盘块地址（块号）数组，其中第 i 个条目存放的是逻辑块号 i 对应的物理块号

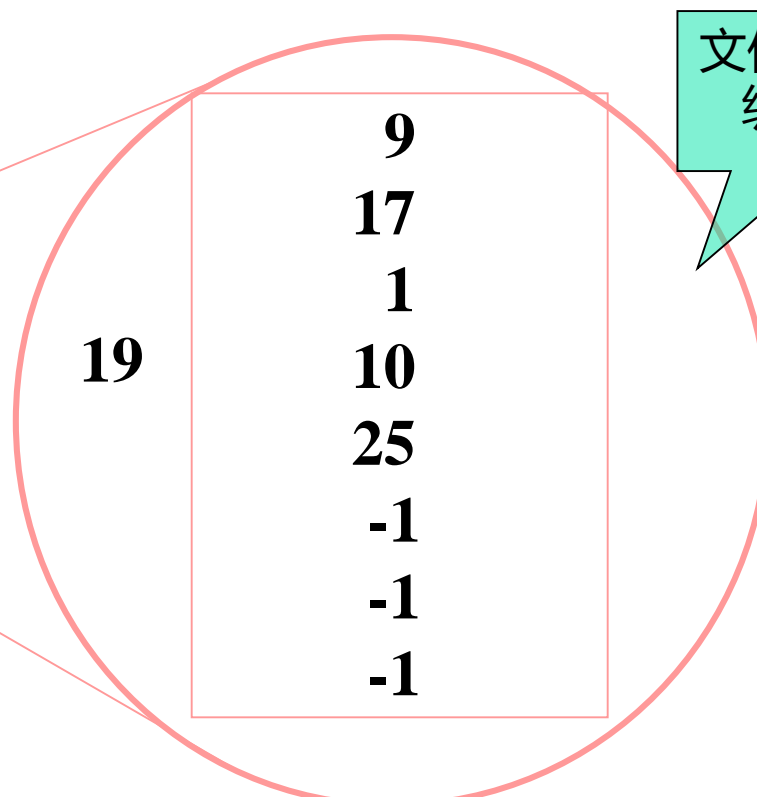
文件目录的目录项中指出索引表的物理地址

索引结构



文件目录

文件名	索引表地址
test	19



文件test的单
级索引表



索引结构

优点：保持了链接结构的优点，又避免了其缺点

- ✓ 既能顺序存取，又能随机存取
- ✓ 能满足文件动态增长、插入、删除的要求
- ✓ 能充分利用外存空间

缺点：

索引表本身带来了系统开销



索引结构

一般来说，小文件的索引表保存在一个单独的物理块中。

如果文件很大，索引表较大，超过了一个物理块，就必须考虑索引表的组织方式

索引表的组织:

(1) 连续方式

索引表占用多个连续的盘块

(2) 链接方式

索引表按照链式结构组织，占用多个不连续的盘块

(3) **多级索引** (多重索引)

例如，二级索引：将一个大文件的所有索引表（二级索引）的地址放在另一个索引表（一级索引）中

此外，还有三级索引等。



索引结构

UNIX文件系统采用的是多级混合索引结构。

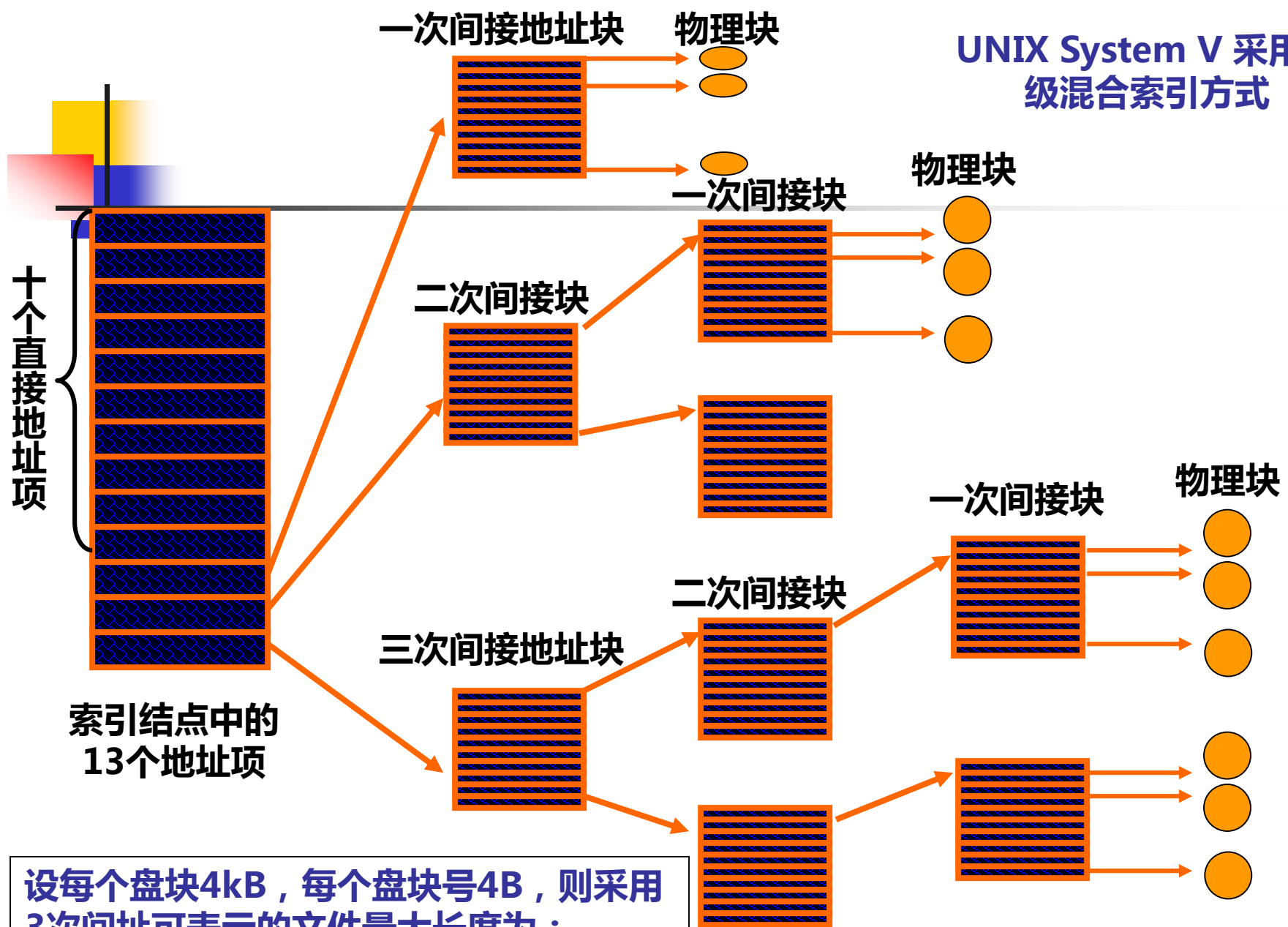
每个文件的索引表为13个索引项

最前面10项直接登记存放文件数据的物理块号（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，该块中最多可放256个文件物理块的块号（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

UNIX中采用了三级索引结构后，文件最大可达16M个物理块

UNIX System V 采用多级混合索引方式



设每个盘块4kB，每个盘块号4B，则采用3次间址可表示的文件最大长度为：
 $4T + 4G + 4M + 40K(B)$



6.2 文件的结构与存取方式

三、文件的存取方式

主要有顺序存取和随机存取两种。

- ✓ 顺序存取

对文件中的数据按逻辑顺序进行读/写的存取方式

- ✓ 随机存取

对文件中的数据按任意顺序进行读/写的存取方式

- ✓ 按键 (key) 存取：如DBMS



6.2 文件的结构与存取方式

四、文件结构、存取方式与存储介质的关系

文件的存取方式不仅与文件的结构有关，还与文件所在存储介质的特性有关，如下表所示：

存储介质	磁带	磁盘		
物理结构	连续结构	连续	链接	索引
存取方式	顺序存取	顺序	顺序	顺序
		随机		随机



6.3 文件目录

一、几个基本概念

1. 文件控制块 (File Control Block, FCB)

文件控制块是操作系统为管理文件而设置的数据结构，存放了为管理文件所需的所有相关信息（**文件属性**）

也称为**文件说明**，或**文件目录项**

文件控制块是文件存在的标志

文件控制块的内容：

- ✓ 基本信息：**文件的名字、地址**（起始物理块号）、长度、结构（逻辑结构、物理结构）、类型
- ✓ 存取控制信息：文件属主（owner）、存取权限或口令
- ✓ 使用信息：共享计数，文件的建立、修改日期等



6.3 文件目录

2. 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

3. 目录项

构成文件目录的项目（目录项就是FCB）

4. 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

目录主要是为了系统快速实现“按名访问”而引入的，

查目录是文件系统最频繁的操作，因此目录的合理组织很重要



6.3 文件目录

二、目录结构

1. 单级目录结构

为所有文件建立一个目录文件（组成一线性表）

优点：简单，易实现

缺点：

- ✓ 限制了用户对文件的命名（容易出现“命名冲突”）
- ✓ 顺序检索文件时，平均检索时间长
- ✓ 不利于对文件的共享



目录结构

2. 二级目录结构

目录分为两级：

一级称为主文件目录，给出用户名，用户子目录所在的物理位置；

二级称为用户文件目录（又称用户子目录），给出该用户所有文件的FCB

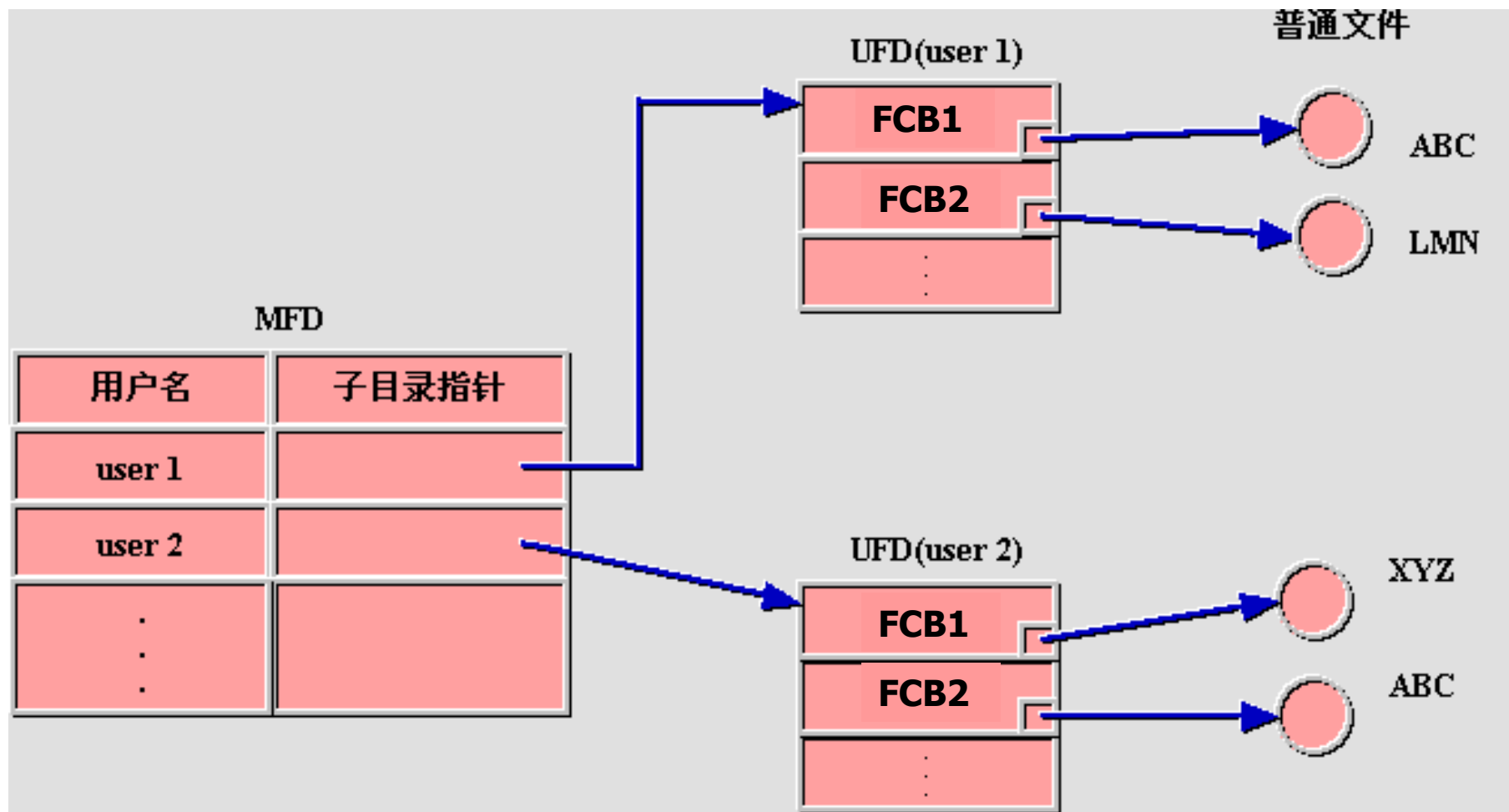
优点：一定程度上解决了文件的重名和文件共享问题

用户名|文件名

查找时间缩短

缺点：增加了系统开销

二级目录结构





目录结构

3. 多级目录结构（树型目录）

对二级目录简单扩充可得三级或三级以上的多级目录结构，
即允许每一级目录中的FCB要么指向文件，要么指向下一级子目录。

这是当今主流OS普遍采用的目录结构

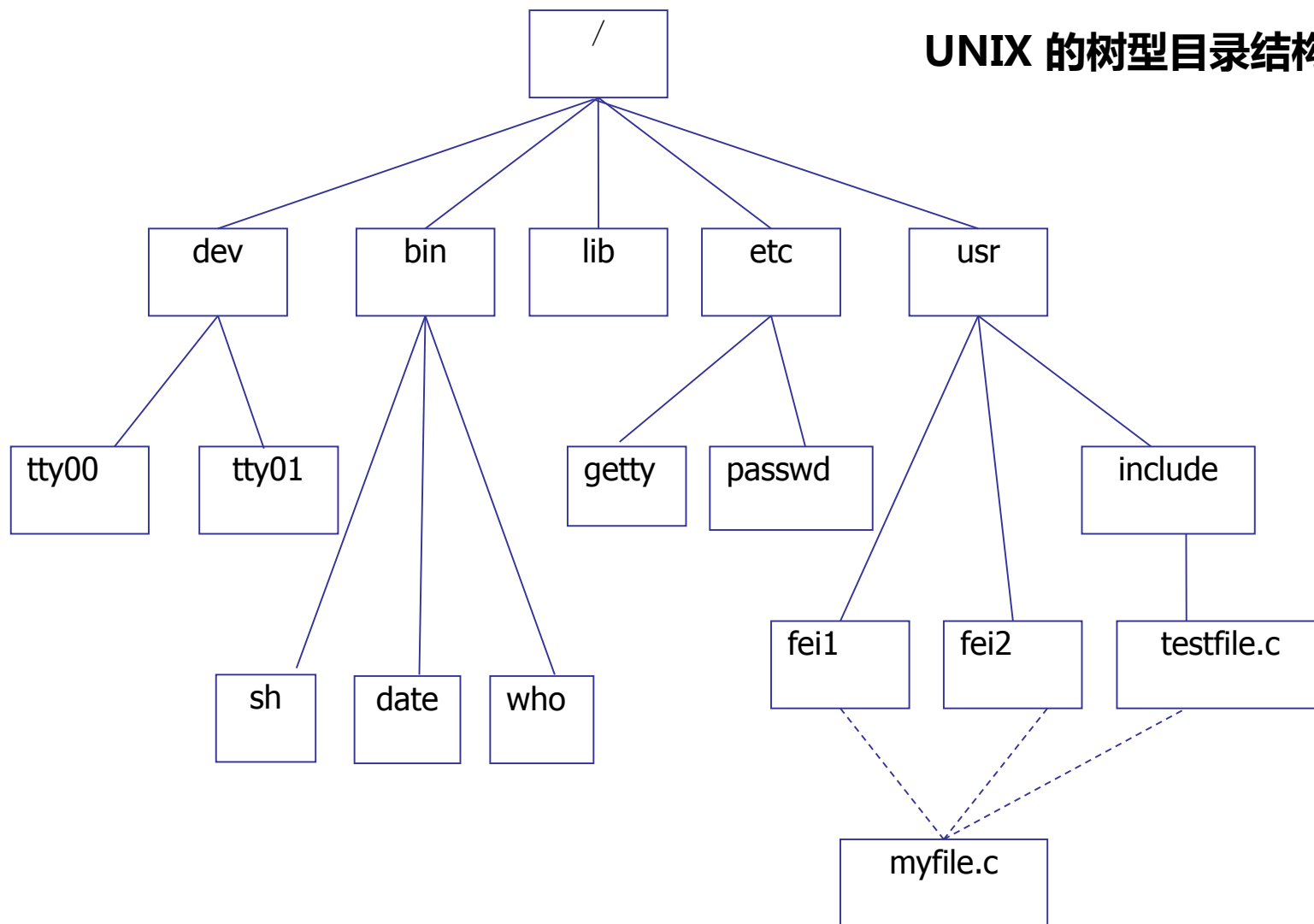
优点：

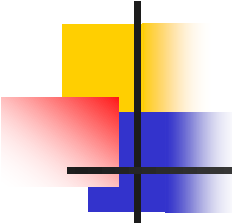
层次结构清晰，便于管理和保护；有利于文件分类；

能较好地避免重名问题；提高了文件检索速度；有利于访问权限的控制

树型目录结构

UNIX 的树型目录结构





6.3 文件目录

三、文件目录检索

访问文件时，必须首先确定读写文件的地址，需要下列2步：

- (1) **目录检索**：根据文件名，查目录，确定文件的起始地址。
- (2) **文件寻址**：确定所要访问文件内容的起始位置（地址）。



6.3 文件目录

1. 目录检索

文件的“按名存取”是通过查目录实现的，系统按照文件的路径名检索

基本的目录检索技术主要有：

- ✓ 线性检索法
- ✓ Hash方法

为了加快目录检索，许多系统引入当前目录（工作目录）、相对路径名等。



6.3 文件目录

2. 文件寻址

根据目录项（FCB）中记录的文件物理地址等信息，

求出文件的任意记录或字节在存取介质上的地址

文件寻址与文件的物理结构和逻辑结构以及设备的物理特性有关

文件的内容是以块为单位存储的。

但存取文件时，对于记录式文件，是以逻辑记录为单位提出存取要求的，因此，

存储介质上的物理块长度与逻辑记录的长度是否匹配直接影响到对文件的寻址



6.3 文件目录

四、文件目录的实现

1. 把文件说明信息（FCB）都放在目录项中

当查找文件时，需要依次将存放目录的物理块装入内存，逐一比较文件名，直到找到为止。

例如，文件说明占128B，每块512B，则每块可存放4个目录项，100个文件就需要25块。

设目录文件占用的盘块数是N个，则要找到一个目录项，

平均需要读入多少个盘块？

$(N+1)/2$ 块



文件目录的实现

把文件说明信息（FCB）都放在目录项中的缺点：

查找文件缓慢，因为目录项较大

文件目录平常放在外存中，当文件很多时，可能占用大量的外存物理块

检索目录时，需要文件说明中的什么信息？

文件名即可。



文件目录的实现

2. 将文件说明分成2部分（目录项分解法）

把FCB分成两部分：

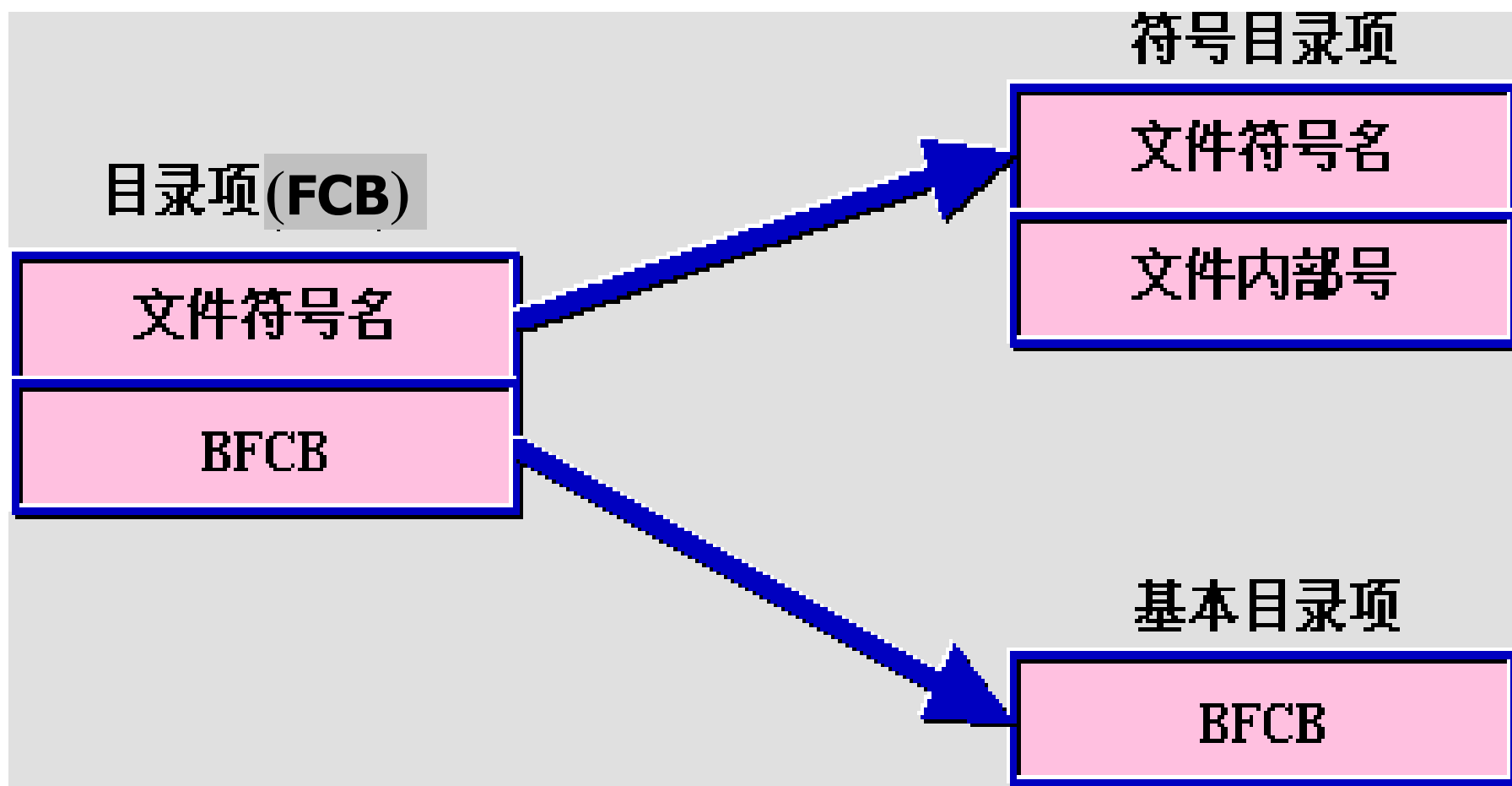
① 符号目录项

文件名，文件号（基本目录项编号）

② 基本目录项

除文件名外的所有文件说明

目录项分解法





目录项分解法

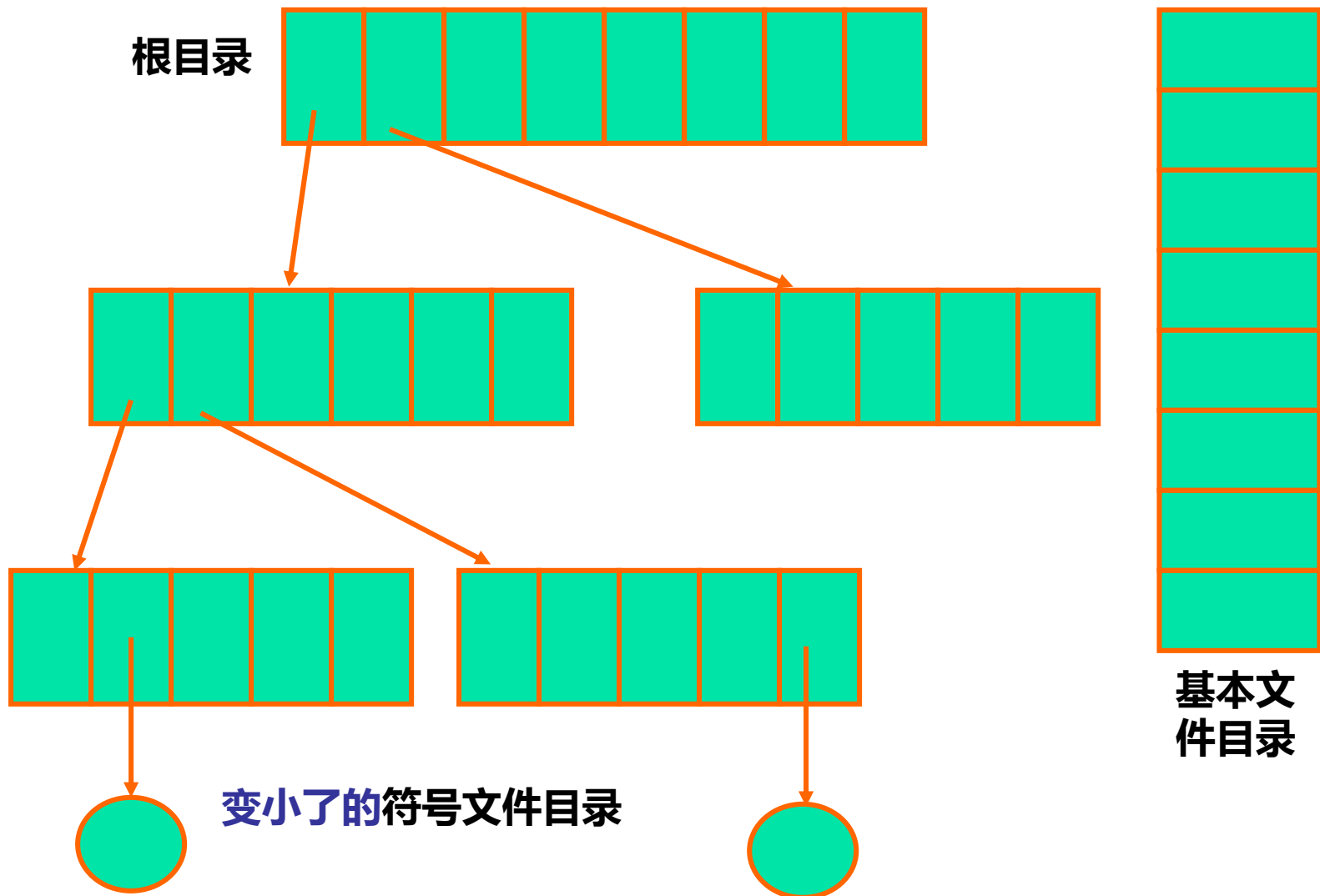
目录项分解法的典型实现：

(1) 基本文件目录 + 符号文件目录

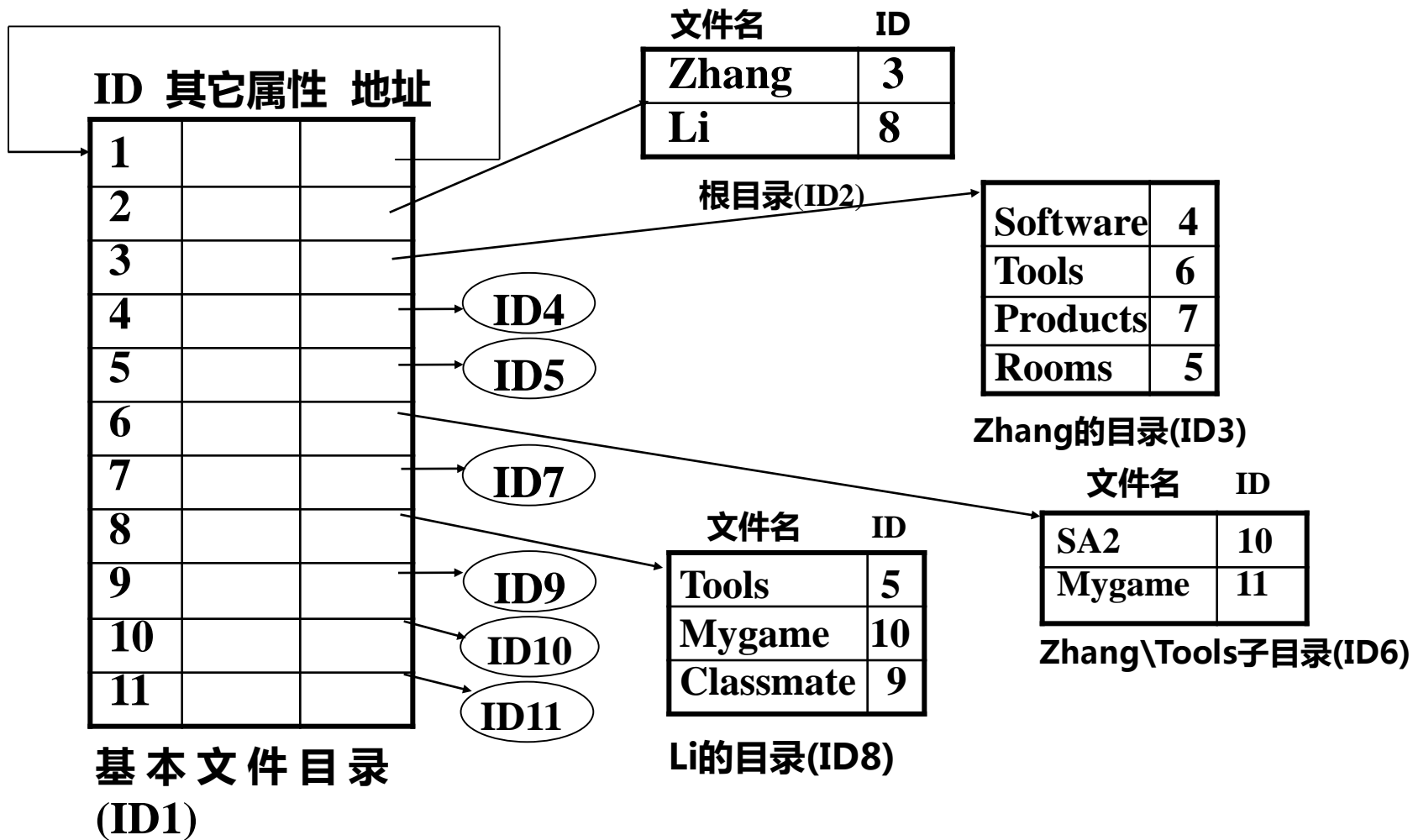
(2) 目录项 + I节点

Unix/Linux采用此方法，它把符号目录项称为目录项，
而把基本目录项称为**I节点**（ Index node，**索引节点**），
这样，目录项中的文件号就是**I节点号**。

目录项分解法



目录项分解法





文件目录的实现

【例】 设磁盘物理块大小为512个字节，一个FCB有64个字节，符号目录项占8个字节，其中文件名占6个字节，文件号占2个字节，基本目录项占 $64 - 6 = 58$ 个字节。若把含有256个目录项的某目录文件改造成符号文件目录和基本文件目录的结构，试说明改造前后查找一个文件的平均访问磁盘块数。



文件目录的实现

解：分解前：1块含 $512/64=8$ 个FCB

分解后：1块含 $512/8=64$ 个符号目录项

该目录文件含有256个目录项

分解前占 $256/8 = 32$ 块

分解后其符号文件目录占 $256/64 = 4$ 块

故查找一个文件的平均访问磁盘块数：

分解前： $(1 + 32)/2 = 16.5$

分解后： $(1 + 4)/2 + 1 = 3.5$

由此可见：改造后减少了访问磁盘的次数，提高了检索速度。



文件目录的实现

目录的其他实现方法：

✓ Hash表

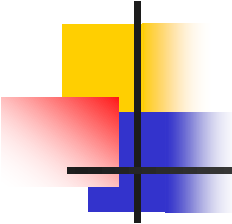
目录文件按目录项键（key）的Hash值进行顺序组织。

创建或搜索时，根据文件名计算Hash值，得到一个指向目录表中相应表目的指针

✓ 其他方法：

如B+树，这是一种将大的单级索引目录文件组织成有序的树型多级索引目录文件的方法，是索引顺序文件中实际采用的基本索引结构，支持随机访问和顺序访问，多见于DBMS中。

NTFS文件系统就采用了B+树



6.4 空闲存储空间的管理

一、空闲区表

将所有空闲区记录在一个表中。

适合连续文件的外存分配与回收。如今很少用

6.4 空闲存储空间的管理

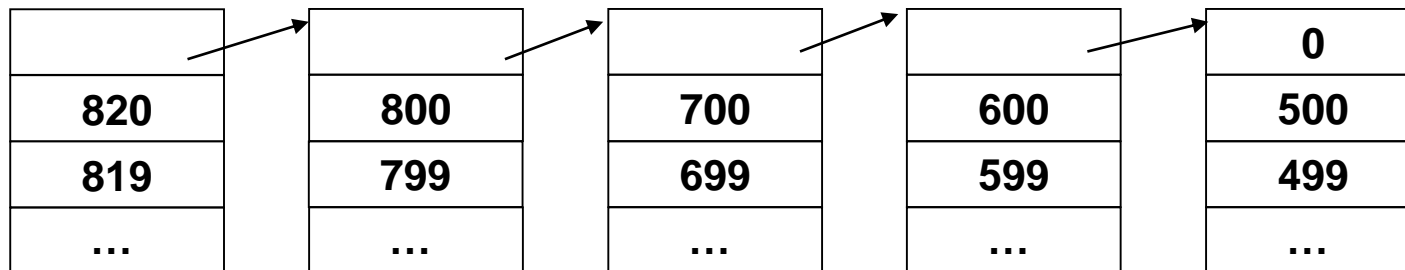
二、空闲块链

把所有空闲块链成一个链。适合离散分配

Windows、Unix使用类似方法

空闲块如何链接？

- (1) 每个空闲块中指出下一个空闲块的块号。
- (2) 采用多个空闲块构成的链表存放空闲块号。





空闲块链

扩展：

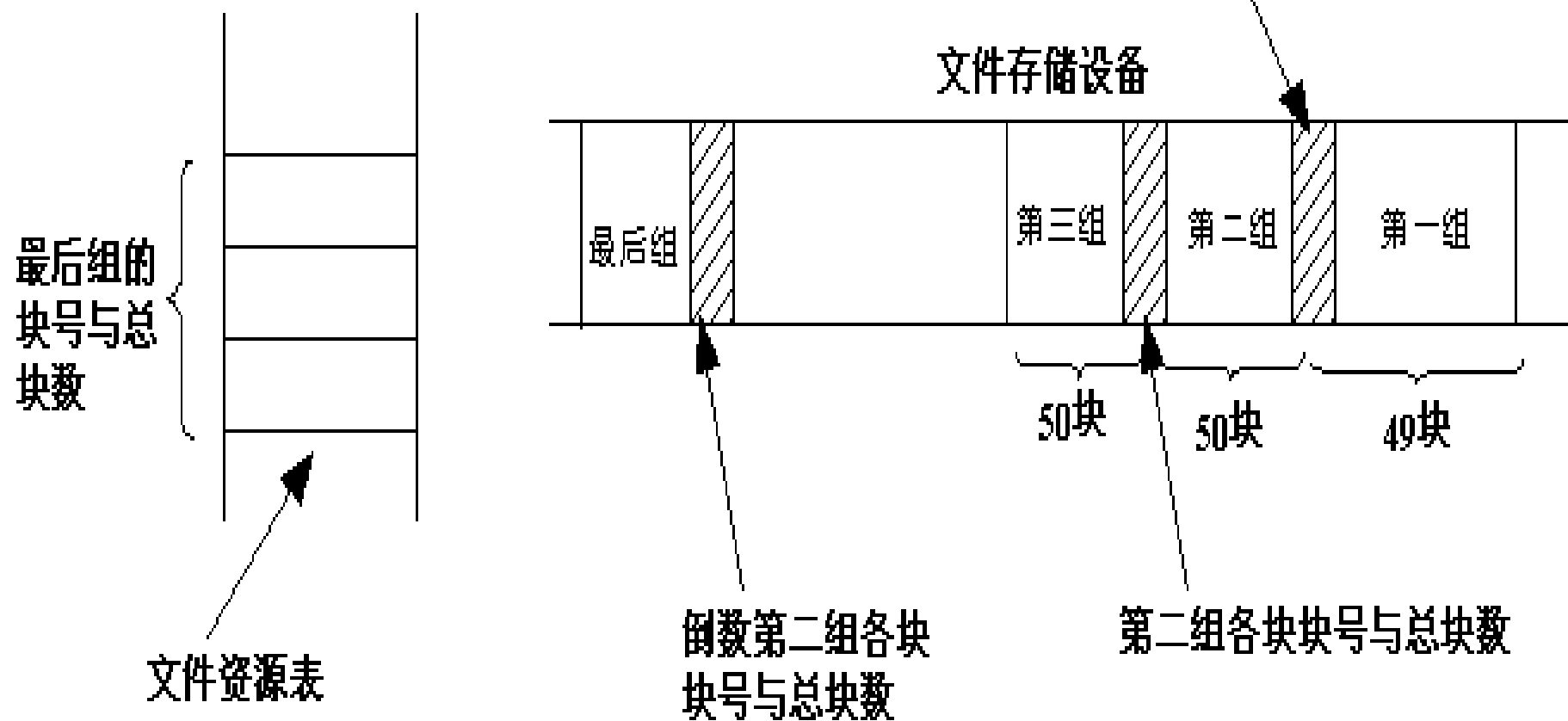
①不断地适度增加块大小

从最早的512B \Rightarrow 1KB \Rightarrow 2KB \Rightarrow 4KB \Rightarrow 8KB \Rightarrow 16KB \Rightarrow 32KB \Rightarrow 64KB。

②成组链接法

链上每个节点记录1组空闲块。适合大型文件系统，分配、释放快，链占用空间少（除首组外均隐藏在空闲块中）。UNIX用之

成组链接法





6.4 空闲存储空间的管理

三、位图

- ✓ 用一串二进制位反映磁盘空间的分配情况，每个物理块对应1位，已分配的物理块为1，否则为0
- ✓ 申请物理块时，可以在位图中查找为0的位，返回对应的物理块号
- ✓ 归还时，将对应位设置为0
- ✓ 描述能力强，适合各种物理结构



6.5 文件的使用、共享和保护

一、文件的使用

为方便用户使用文件，文件系统提供对文件的各种操作，使用的形式包括系统调用或命令

- ① 提供设置和修改用户对文件访问权限的操作
- ② 提供建立、修改、删除目录的操作
- ③ 提供文件共享、设置访问路径的操作
- ④ 提供创建、**打开、读、写、关闭**、删除文件等操作

其中，最基本的操作是：打开、关闭、读、写文件等



文件的使用

1. 对文件的基本操作方式





文件的使用

2. 为什么要open/close文件？

open：把文件说明信息（FCB）装入内存，便于以后的快速访问。

- （1）根据指定的文件路径名，查目录，找到相应文件的目录项，检查权限；
- （2）将文件说明信息装入内存；
- （3）分配一个文件id（整数）。后面通过该id实施对该文件的操作。

close：

- （1）释放文件说明信息所占的内存空间；
- （2）把文件缓冲区中已修改的内容写回文件。

很多系统限制进程打开文件的个数，用户尽可能要关闭不再使用的文件。



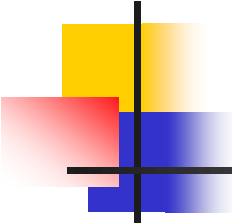
6.5 文件的使用、共享和保护

二、文件共享

文件共享：一个文件被多个用户或进程使用

共享的目的：

- ✓ 节省时间和存储空间，减少用户工作量
- ✓ 进程间通过文件交换信息



文件共享

1. 普通的文件共享方法

(1) 按路径名访问共享文件

实现简单，不需要建立另外的目录项

但路径名可能长，检索较慢

(2) 链接法

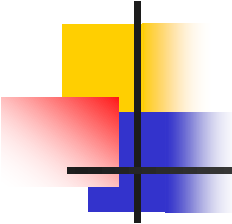
在相应目录项之间建立链接。即一个目录项中含有指向另一个目录项的指针。

实现方法：

在目录项中设置一个“链接属性”，

表示目录项中的“物理地址”是指向另一目录项的指针。

同时，在共享文件的目录项中包含“用户计数”。



文件共享

(3) 基本文件目录BFD

- ✓ 整个文件系统有1个基本文件目录BFD：

每个文件（及目录）有1个目录项，包含系统赋予的唯一标识符ID（整数）
以及其他的文件说明信息

- ✓ 每个目录有1个符号文件目录SFD：除了ID = 0, 1, 2外,

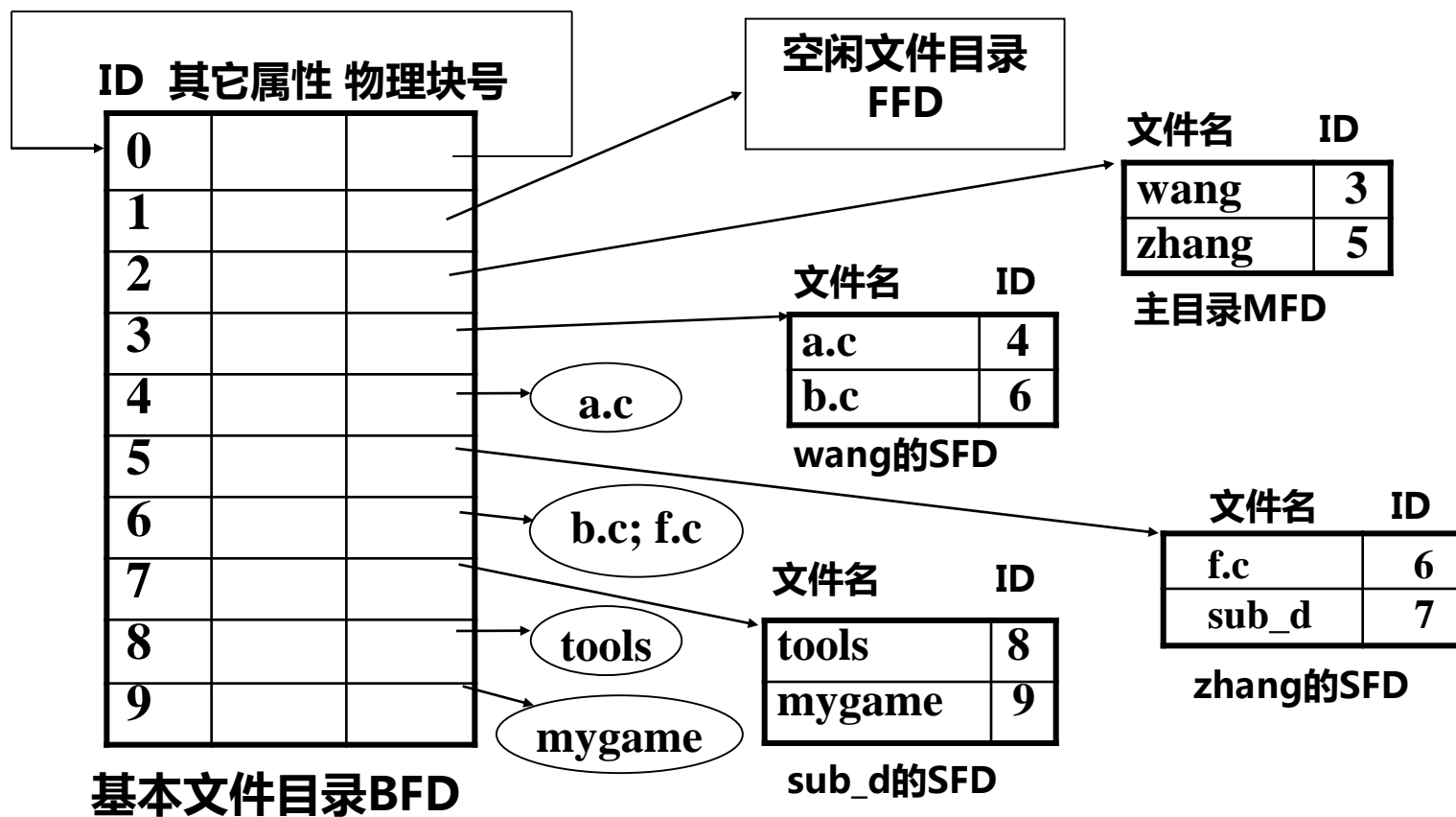
每个目录项仅包含文件名和ID

- ✓ 系统把ID = 0, 1, 2的目录项分别作为BFD、FFD、MFD的标识符

- ✓ **共享方法：**

若一个用户想共享另一用户的文件，只需在自己的目录文件中增加一个目录项，
填上自己起的文件名和该共享文件的唯一ID即可。如ID = 6的文件。

文件共享





文件共享

2. 基于I节点的文件共享方法（Unix采用）

（1）硬链接

多个目录项指向同一个I节点。

引入链接计数count：表示链接到本I节点的文件数

硬链接

【例】

1) 用户A创建1个新文件name1

/dirA中的目录项

文件名 **I节点号**

name1	1234
-------	------

...
owner = A
count = 1
5678

I节点1234

文件内容

"This is
the text
in the file"

块5678

硬链接

2) 用户B共享该文件，命名为name2

/dirB中的目录项

文件名	I节点号
name2	1234

/dirA中的目录项

文件名	I节点号
name1	1234

...
owner = A
count = 2
5678

文件内容

"This is
the text
in the file"

I节点1234

块5678

将I节点1234中的count置为2。

硬链接

3) 用户A删除文件name1

/dirB中的目录项

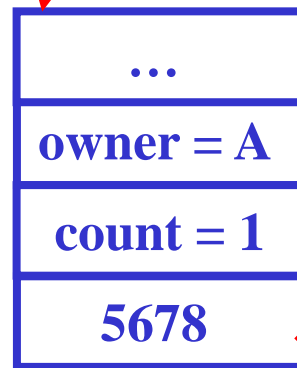
文件名 I节点号

name2	1234
-------	------

删除了文件name1后，并不会删除I节点1234，只是将count减1。

缺点：

由于文件主是A，若要计费收费，A即使把文件name1删除了，还要为B付费，直到B把name2删除为止。



I节点1234

文件内容

"This is
the text
in the file"

块5678



基于I节点的文件共享

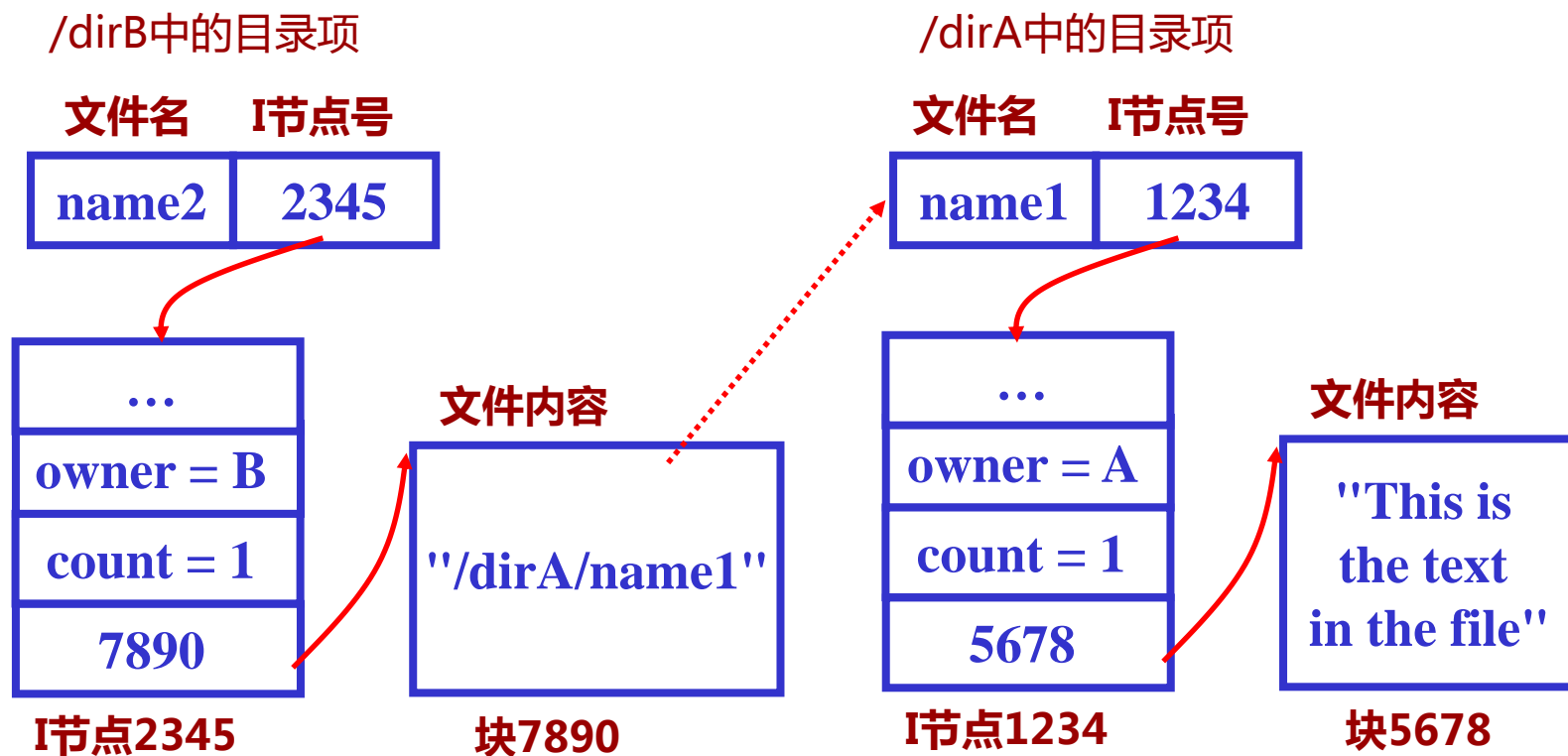
(2) 符号链接

又称软链接

文件内容是所链接文件的路径名

符号链接

【例】文件name1和它的一个符号链接name2





符号链接

符号链接的缺点：

- ① 访问共享文件时，要根据路径名查目录，直到找到I节点，开销大；
- ② 若执行

```
rm /dirA/name1
```

则符号链接依然存在，此时对/dirB/name2的引用将出错。

然而，若后来创建一新文件/dirA/name1，这个符号链接将指向该文件。

符号链接在Windows中叫快捷方式。



6.5 文件的使用、共享和保护

三、文件存取控制

1. 存取控制矩阵

给出每个用户对每个文件的访问权限。

一维是所有用户，另一维是所有文件，
对应的矩阵元素是用户对文件的访问权限。

例如，访问操作分为：

- ✓ 读操作 (r)
- ✓ 写操作 (w)
- ✓ 执行操作 (x)
- ✓ 不能执行任何操作 (-)

当用户和文件较多时，很庞大。



存取控制矩阵

用户	文件			
	File1	File2	File3
User1	rwX	r	rw
User2	x	rw	—
User3	r	—	r
User4	rwX	r	rw



文件存取控制

2. 存取控制表 (Access Control List , ACL)

每个文件一张ACL，将用户分类，规定每类用户的访问权限。

例如，Unix/Linux将用户分类为：

- ✓ 文件主 (owner)
- ✓ 文件主的同组用户 (group)
- ✓ 其他用户 (other)



存取控制表

用户	对File1的访问权限
owner	rwX
group	rx
other	-

这样UNIX索引节点中可用9个二进制位

rwX rwX rwX

对各类用户设访问权限。

文件主可修改访问权限，例如

```
chmod 711 file1
```

```
chmod 755 file2
```



文件存取控制

3. 存取权限表 (Capability List , CL)

每个用户一张CL，规定对每个文件的访问权限。

文件	某用户的访问权限
file1	r
file2	rw
file3	rwX



文件存取控制

4. 口令

用户创建文件时，设置一个口令，放在文件目录中。

5. 密码

写入时加密，读出时解密。



6.6 小结

一、文件系统及其作用

- ✓ 什么是文件和文件系统？为什么引入文件和文件系统？
- ✓ 文件系统有何作用？用户观点？系统观点？

二、文件的逻辑结构和物理结构

掌握每种结构的特点，会一般的计算。例如读盘次数等。

三、文件目录

- ✓ 几个概念：文件控制块FCB；文件说明；目录项；文件目录
- ✓ 文件目录的实现方法：FCB都放在目录项中；目录项分解
- ✓ 目录检索与文件寻址
- ✓ 会一般的计算



6.6 小结

四、基于I节点的文件共享

- ✓ 硬链接
- ✓ 软链接

五、为什么要打开/关闭文件？Open/close有何作用？