


操作系统



中国矿业大学计算机学院



文件  系统



目录

CONTENTS

5.1 概述

5.2 文件的组织

5.3 文件目录

5.4 文件系统调用的实现

5.5 文件共享

5.6 文件系统体系结构

5.7 Windows文件系统

5

5.1.1 文件的概念

什么是文件？在计算机系统中，文件是指存储
在外部存储介质上的、由文件名标识的一组相
关信息的集合。

- 文件的概念本身体现了操作系统的一种抽象机制，用于逻辑上描述存储在外存上的数据。
- 把数据组织成文件形式加以管理和控制是计算机数据管理的重大发展。

5

5.1.1 文件的概念

- 在文件这一抽象机制中最重要的是**文件命名**
- 各种系统的文件命名规则不尽相同，但一般来说，文件名称都由文件名和扩展名两部分组成，中间用“.”分隔开来，前者用于识别文件；后者用于标识文件类型。它们都是字母或数字组成的字母数字串。

5

5.1.2 文件系统

◆ 操作系统中负责管理和存取文件信息的软件机构叫做文件管理。

◆ 什么是文件系统？ 操作系统中与文件管理有关的那部分软件和被管理的文件，以及实现管理所需要的一些数据结构的总体。

- ✓ 从系统角度看，文件系统是对文件存储空间进行组织、分配，并负责文件的存储、保护和检索的系统。
- ✓ 从用户角度来看，文件系统主要是实现“按名存取”，并向用户提供简便、统一的使用文件的接口。

1. 文件系统的功能

- (1) 实现文件的“按名存取”功能。
- (2) 实现能够快速定位文件的目录结构；考虑如何组织目录文件。
- (3) 向用户提供一套使用方便、简单的操作命令。
- (4) 管理磁盘、磁带等组成的文件存储器。
- (5) 实现逻辑文件到物理文件的转换。
- (6) 保证文件信息的安全可靠。
- (7) 便于文件的共享。

2. 常用文件系统举例

- ◆ **EXT2**: Linux最为常用的文件系统，设计易于向后兼容，所以新版的文件系统代码无需改动就可以支持已有的文件系统。
- ◆ **NFS**: 网络文件系统，允许多台计算机之间共享文件系统，易于从网络中的计算机上存取文件。
- ◆ **HPFS**: 高性能文件系统，是IBM OS/2的文件系统。
- ◆ **FAT**: 经过了MS-DOS, Windows 3.x, Windows 9x, Windows NT, Windows 2000/XP和OS/2等操作系统的不断改进，它已经发展成为包含FAT12, FAT16和FAT32的庞大家族。
- ◆ **NTFS**: NTFS是微软为了配合Windows NT的推出而设计的文件系统，为系统提供了极大的安全性和可靠性。

5

5.1.3 文件的属性

- ◆ 一个文件包括**文件体**和**文件属性**两个部分：文件体是一系列的记录或字符流，以物理块存放在外存上，也叫文件内容。文件属性是对文件进行说明的信息。
- ◆ 主要属性：
 - ◆ **文件名称**：文件名称是供用户使用的外部标识，也是文件最基本的属性。
 - ◆ **文件物理位置**：具体标明文件在存储介质上所存放的物理位置。
 - ◆ **文件拥有者**：通常文件创建者对自己所建的文件拥有一切权限，而对其它用户所建的文件则拥有有限的权限。

5

5.1.3 文件的属性

- ◆ **文件权限**：通过文件权限，文件拥有者可以为自己的文件赋予各种权限，如可允许自己读写和执行，允许同组的用户读写，而只允许其它用户读。
- ◆ **文件类型**：可以从不同的角度来对文件进行分类，例如普通文件或是设备文件，可执行文件或是文本文件，等等。
- ◆ **文件长度**：长度单位通常是字节。
- ◆ **文件时间**：最初创建时间，最后一次的修改时间，最后一次的执行时间，最后一次的读时间等。

5

5.1.4 文件的分类

◆ 为了有效、方便地组织和管理文件，常按照不同的观点对文件进行分类。常用的几种文件分类方法：

(1) **按照文件的逻辑结构的不同**：流式文件和纪录式文件。

(2) **按照用途**：系统文件、库文件和用户文件。

(3) **按照性质**：可以把文件分为普通文件、目录文件和特殊文件。

5

5.1.5 文件的使用

- ◆ 为了方便用户使用文件系统，文件系统向用户提供了两类操作接口：
 - ✓ 第一类是与文件有关的操作命令或作业控制语言中与文件有关的JCL 语句
 - ✓ 第二类是提供给用户程序使用的文件系统调用

5

5.1.5 文件的使用

◆ 主要的系统调用：

◆ 文件创建

◆ 文件写

◆ 文件删除

◆ 文件的读写定位

◆ 文件截断

◆ 文件打开

◆ 文件读

◆ 文件关闭



目录

CONTENTS

5.1 概述

5.2 文件的组织

5.3 文件目录

5.4 文件系统调用的实现

5.5 文件共享

5.6 文件系统体系结构

5.7 Windows文件系统

5

5.2 文件的组织

- ◆ **存储介质：**用来存储文件信息的载体，例如，磁带、磁盘和光盘等。
- ◆ **块：**存储介质上连续信息所组成的一个区域，也叫做物理记录。
 - ✓ 文件的内容及相关信息都是以“块”为单位存放在外存上的
 - ✓ 外存物理空间的分配是以块为单位的。
 - ✓ 块也是内存和外存进行信息交换的物理单位，每次总是交换一块或整数块信息。

5

5.2 文件的组织

- ◆ **文件组织：**文件中信息的配置和构造方式，同一个文件应该从两个侧面来观察它的文件组织方式：
 - ✓ **文件的逻辑结构：**从用户的观点出发观察到的文件组织形式，用户可以直接处理，独立于文件的物理特性。
 - ✓ **文件的物理结构：**逻辑文件在物理存储空间中存放方法和组织关系，又称文件的存储结构。

5

5.2.1 文件的逻辑结构

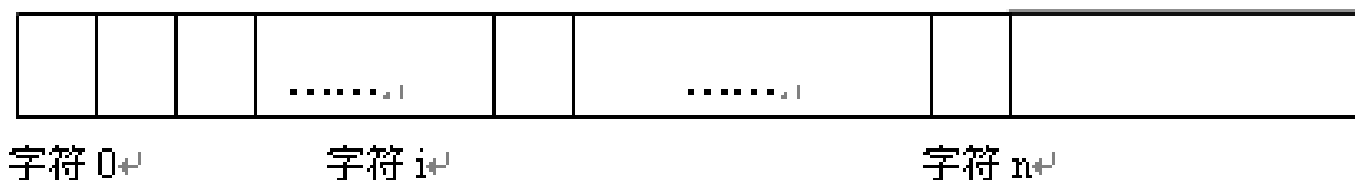
- 文件的逻辑结构分为两种形式：
 1. 流式文件
 2. 记录式文件

5

5.2.1 文件的逻辑结构——流式文件

什么是流式文件？

- ◆ 文件内的数据不组成记录，只是依次的一串信息集合，如字节流或字符流，它也可以看成是无结构的或只有一个记录的记录式文件，所以也称作无结构文件
- ◆ 字节或字符是访问流式文件的基本单位，顺序存取时读/写指针每次步进1个字节或1个字符长度。



5

5.2.1 文件的逻辑结构——流式文件

◆ 流式文件举例：

源程序文件、可执行文件和库函数文件等

◆ 这些类型的文件并不需要分记录

◆ 没有结构并非意味着该类型的文件不能有结构，处理该文件的软件可以按照用户定义的结构来操作文件。

5

5.2.1 文件的逻辑结构——记录式文件

什么是记录式文件？

- ◆ 一种有结构的文件，指文件中的数据由若干条定长或不定长的记录构成，每条记录又由若干数据项构成。
- ◆ 记录是记录式文件进行存取的基本单位，顺序访问时文件读/写指针每次步进一条记录长度。
- ◆ 定长记录文件和不定长记录文件：记录式文件中的所有记录长度一般都相等，也可以不等

5

5.2.1 文件的逻辑结构——记录式文件

- ◆ 记录式文件中记录之间的组织方式非常重要，设计时要综合考虑很多因素，例如，如何提高检索速度，如何便于记录的增删改等。
- ◆ 按照组织方式的不同，记录式文件可进一步分为
 - (1) 顺序文件
 - (2) 索引文件
 - (3) 索引顺序文件

5

5.2.1 文件的逻辑结构——记录式文件

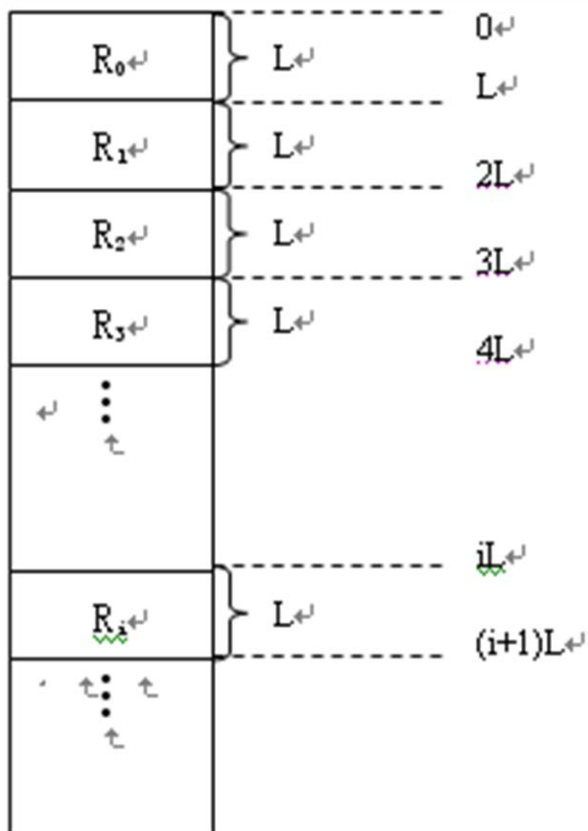
(1) 顺序文件

- ◆ **顺序文件**：记录之间按某种顺序排列组织所形成的文件
- ◆ 在顺序文件中的记录可以按照不同的顺序进行排列。
 - ✓ 一种是按照**存入的时间先后排序**：各记录之间的顺序与记录的关键字或内容无关。（日志文件和各种现场记录文件等场合）；
 - ✓ 一种是**按照记录中的关键字排序**。

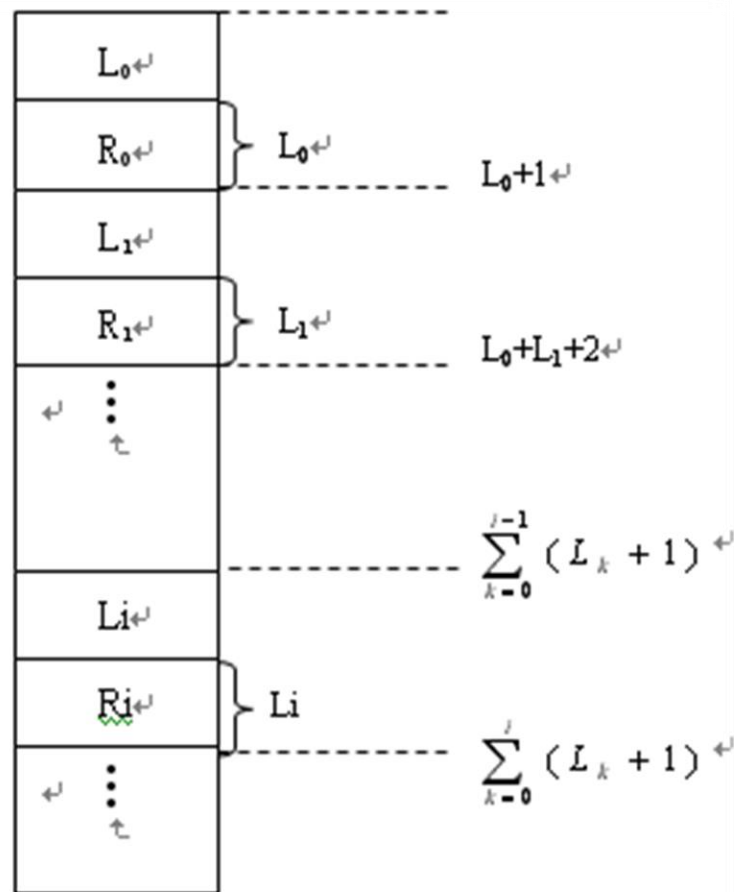
5

5.2.1 文件的逻辑结构——记录式文件

(1) 顺序文件



(a) 定长记录文件



(b) 不定长记录文件

5

5.2.1 文件的逻辑结构——记录式文件

(1) 顺序文件

顺序文件的存取方式：

- ◆ **顺序存取：**只要在系统中分别设置读写指针Rptr和Wptr，读完或写完一条记录后修改该指针指向下一条记录。
- ◆ **直接存取：**也叫随机存取。主要适用于定长记录的顺序文件（任何记录的位置都很容易通过记录长度计算出来）。

5

5.2.1 文件的逻辑结构——记录式文件

(1) 顺序文件

优点:

- ◆ 适合大量记录批量存取的情况，此时对顺序文件的存取效率是所有逻辑文件中最高的。
- ◆ 只有顺序文件才能被存储在磁带上。

缺点:

- ◆ 不适合交互系统中用户要求查找或修改单个记录的情况（但是给出记录位置查找定长记录除外）
- ◆ 增删记录比较困难，因为增加或删除记录后需要重新组织记录的顺序和关键字。

5

5.2.1 文件的逻辑结构——记录式文件

(2) 索引文件

◆ 顺序文件按记录位置查找的问题：

定长记录的顺序文件只要给出记录的位置，通过简单的计算很容易实现随机存取，按关键字检索也较方便。但变长记录的顺序文件都要从第一个记录查起，一直顺序查找到所要查找的记录为止，使查找一个记录的时间非常长。



可不可以做到不管是定长记录还是不定长记录按记录位置查找效率一样呢？

5

5.2.1 文件的逻辑结构——记录式文件

(2) 索引文件

解决思路:

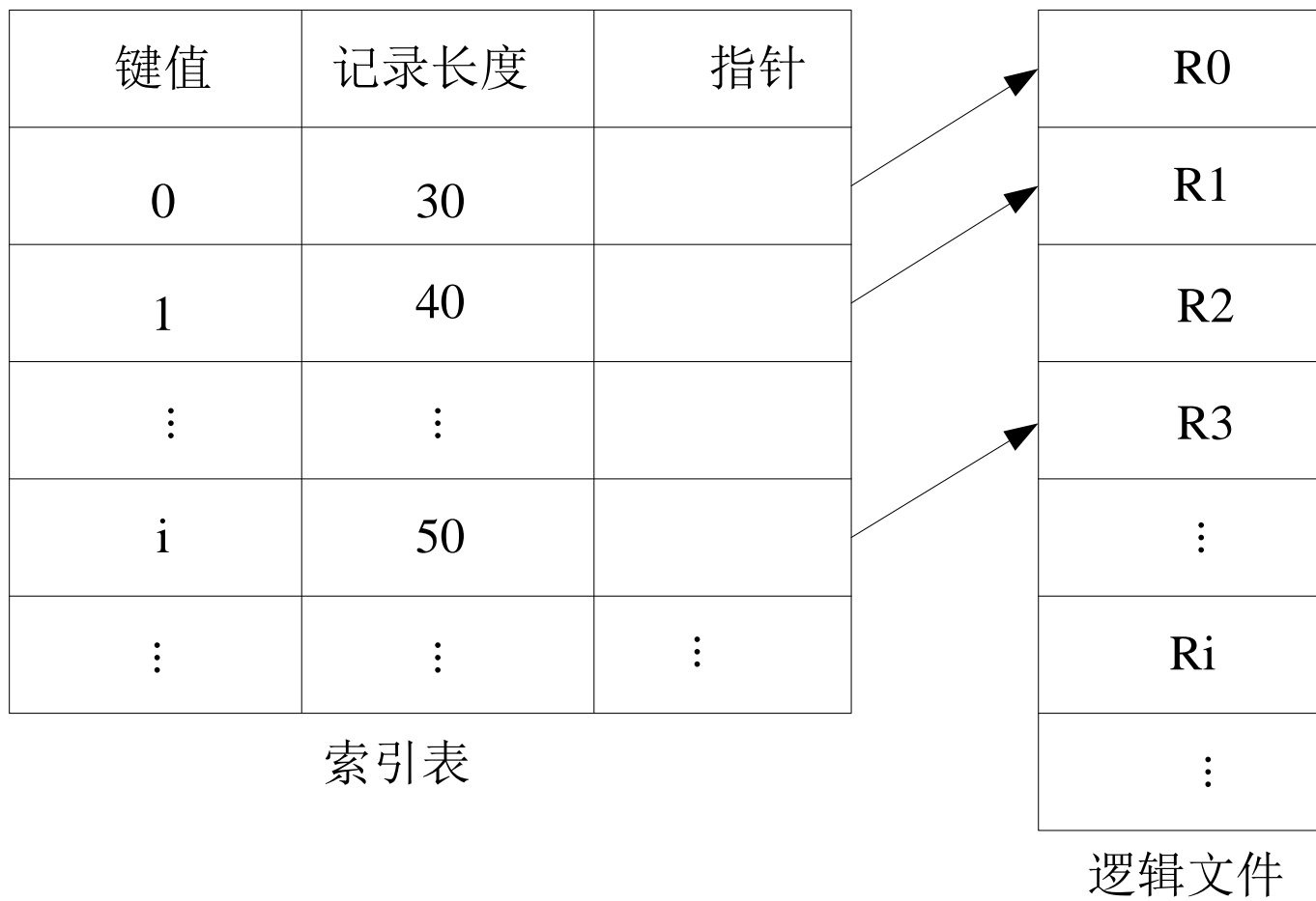
- ◆ 可以为变长记录的顺序文件再建立一张索引表，对主文件中的每个记录在索引表中都有相应的表项，用于记录该记录的长度以及指向该记录的指针。

——索引文件

5

5.2.1 文件的逻辑结构——记录式文件

(2) 索引文件



5

5.2.1 文件的逻辑结构——记录式文件

(2) 索引文件

- ◆ 索引文件可以根据不同的关键字建立索引，形成包含多个索引表的索引文件。
- ◆ **主要优点：**
 - ✓ 通过建立索引极大地提高了对文件的查找速度
 - ✓ 对增加和删除记录也非常方便
- ◆ **主要缺点：** 存储开销变大，增删记录时还需要修改索引表

5

5.2.1 文件的逻辑结构——记录式文件

(3) 索引顺序文件

将顺序文件中的多个记录组合成一组，并对每一组记录建立一个索引，通过索引指针指向该记录组中的第一条记录

因此，索引顺序文件是顺序文件和索引文件相结合的产物，结合两者的优点，同时又能有效克服变长记录文件的缺点，而且所付出的代价也小。

5

5.2.1 文件的逻辑结构——记录式文件

(3) 索引顺序文件

关键字	逻辑地址
An Qi	
Bao Rong	
Chen LIn	
...	...

索引表

姓名	其他属性
An Qi	
An Kang	
...	...
Bao Rong	
...	...

逻辑文件

5

5.2.1 文件的逻辑结构——记录式文件

(3) 索引顺序文件

- ◆ **查找方法：**首先也是利用用户（程序）所提供的关键字，以及某种查找算法去检索索引表，找到该记录所在记录组中第一个记录的表项，从中得到该记录组第一个记录在主文件中的位置。再利用顺序查找法去查找主文件，从中找到所要求的记录。
- ◆ **主要优点：**索引表占用空间小，同时查找效率比顺序文件又高，因此在文件记录比较多时采用索引顺序文件比较适合。

5

5.2.2 记录的成组与分解

- ◆ 文件分逻辑结构和物理结构
- ◆ 记录也分逻辑记录和物理记录，大小不一定相等
- ◆ **逻辑记录**：记录式文件的逻辑结构中的每条记录描述（用户看到的），按信息在逻辑上的独立含义划分的单位
- ◆ **物理记录**：即“块”，存储介质上连续信息所组成的区域，文件的内容及相关信息都是以“块”为单位存放在外存上的（外存存放的单位）

那么，逻辑记录和物理记录一定是一一对应，大小相等吗？

5

5.2.2 记录的成组与分解

- ◆ 因此，一个逻辑记录被存放到文件存储器的存储介质上时，可能占用一块或多块，也可以一个物理块包含多个逻辑记录。
- ◆ 主要讨论后者：当一个物理块包含多个逻辑记录时需要进行记录的成组和分解

5

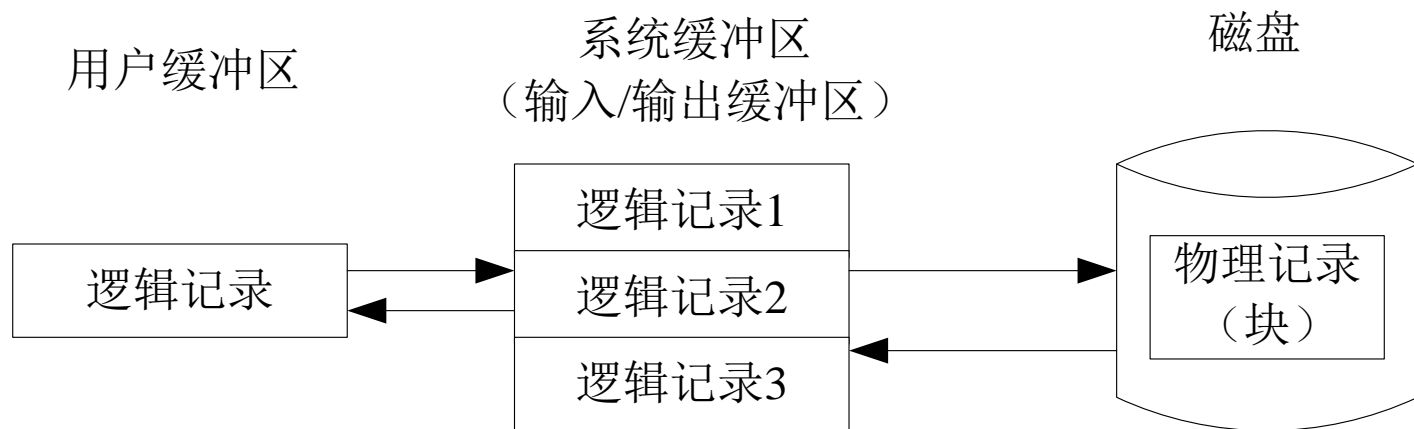
5.2.2 记录的成组与分解

- ◆ **记录的成组**：若干个逻辑记录合并成一组，写入一个块。
- ◆ 每块中的逻辑记录的个数称**块因子**。
- ◆ 成组操作一般先在输出缓冲区内进行，凑满一块后才将缓冲区内的信息写到存储介质上。
- ◆ **记录的分解**：当存储介质上的一个物理记录读进输入缓冲区后，把逻辑记录从块中分离出来的操作叫记录的分解。

5

5.2.2 记录的成组与分解

- 文件被打开后，在主存中分配相应的输入输出缓冲区，用于成组和分解。由于容量有限，操作系统限定同时打开的文件个数，以防止文件缓冲区太多而挤占紧张的主存空间。



5

5.2.3 文件的物理结构

- ◆ 文件的物理结构不仅取决于存储介质的存储特性，还与采用的外存分配方式有关。
- ◆ 考察文件的物理结构时应该把文件看作相关物理块的集合以及如何给文件分配所需要的物理块
- ◆ 按照外存物理块分配方式的不同，文件的物理结构主要分为：
 1. 连续文件
 2. 链接文件
 3. 索引文件
 4. 直接文件

5

5.2.3 文件的物理结构

1. 连续文件

- ◆ 把一个逻辑上连续的文件信息存放在依次相邻的物理块中的组织形式。
- ◆ 连续文件是基于磁带设备的最简单的物理结构，也适合其它各种外存设备。
- ◆ 文件物理地址的存储方式：
目录项的“文件物理地址”字段中，记录该文件第一个记录所在的盘块号和文件长度（以盘块为单位）

5

5.2.3 文件的物理结构

1. 连续文件

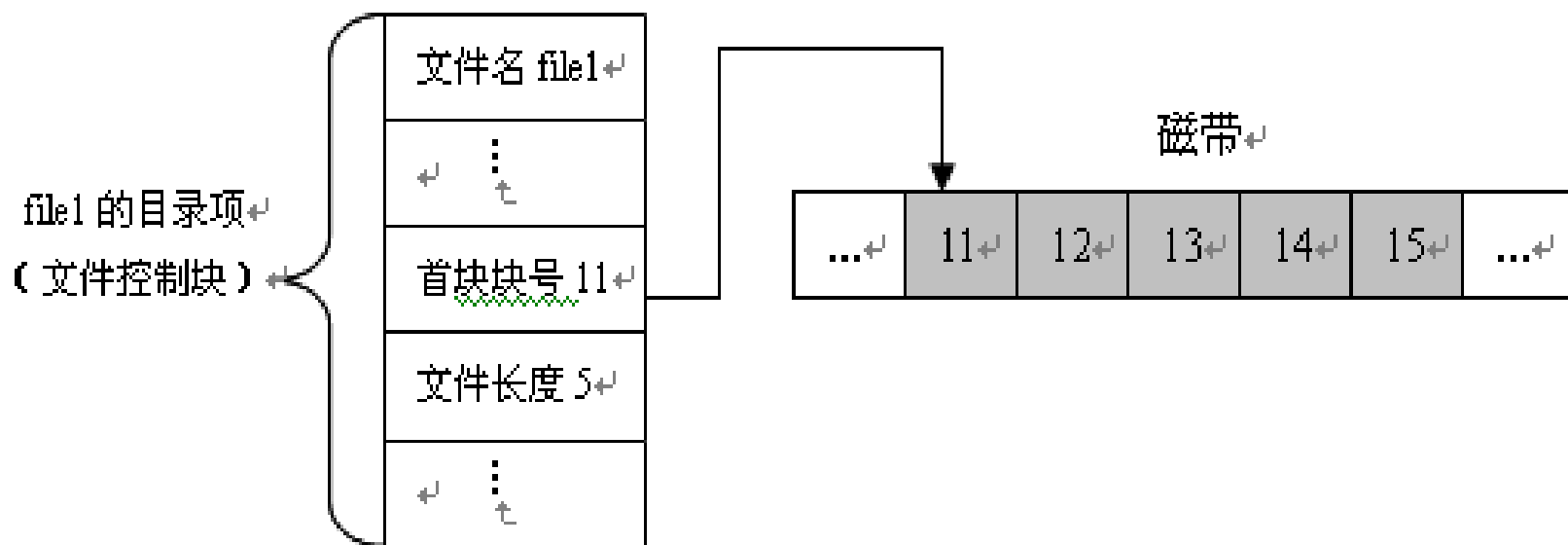


图5.6 连续文件的组织方式

5

5.2.3 文件的物理结构

1. 连续文件

优点:

- ◆ 在顺序存取时速度较快，非常便于顺序访问。
(因为只要在目录中找到顺序文件所在的第一个盘块号，就可以从此开始逐个盘块往下读写。)

应用场合:

- ◆ 常用于存放系统文件，如操作系统文件、编译程序文件和其它由系统提供的实用程序文件，因为这类文件往往被从头至尾依次存取。

5

5.2.3 文件的物理结构

2. 链接文件

如何克服连续文件的缺点？

- ◆ **思路：**可把一个逻辑上连续的文件**分散地存放在不同的物理块**中，这些物理块既不要求连续，也不必按规则排列。——非连续分配
- ◆ 链接分配是非连续分配的一种。
- ◆ 链接文件（链接分配）：通过每个物理块上的链接指针将同属于一个文件的多个离散的盘块链结成一个链表的组织形式，实现了离散分配

5

5.2.3 文件的物理结构

2. 链接文件

优点:

- ◆ 能适应文件的动态增长
- ◆ 消除了磁盘的外部碎片
- ◆ 添加、删除和修改记录更方便。

实现链接组织时有两种方式:

(1) 隐式链接 ; (2) 显示链接

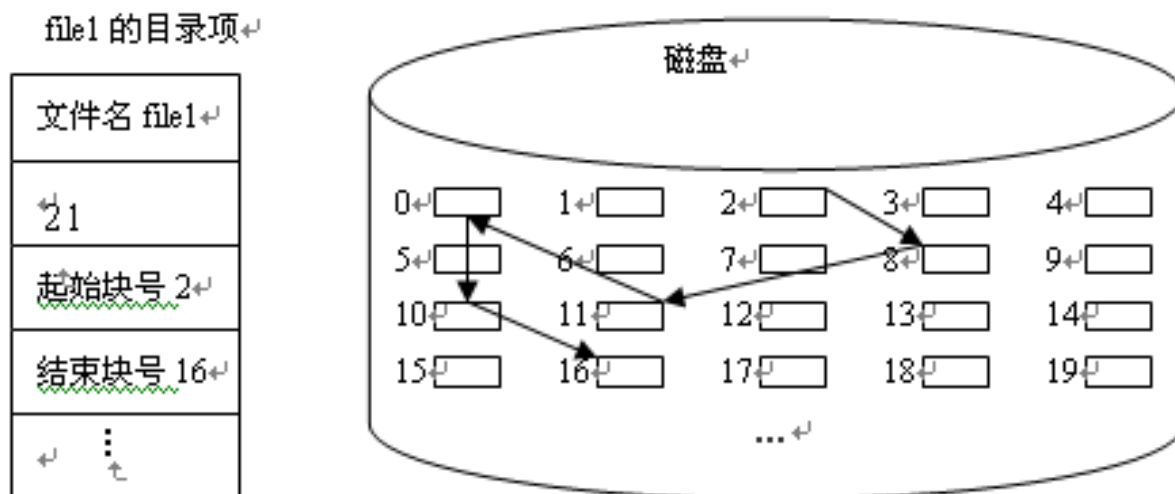
5

5.2.3 文件的物理结构

2. 链接文件

(1) 隐式链接

每个物理块自身存放下一物理块的链接指针



目录项的“物理地址”字段中只要保存该文件的起始块号和结束块号的指针

5

5.2.3 文件的物理结构

2. 链接文件

(1) 隐式链接

◆ 主要缺点:

- (1) 只适合于顺序访问，对随机访问是极其低效
- (2) 每个物理块上增加了一个链接信息，为信息管理添加了一些麻烦。
- (3) 通过链接指针将一大批离散的物理块链接起来，可靠性差，只要其中任何一个指针出现问题，都会导致整个文件信息丢失或出错。

5

5.2.3 文件的物理结构

2. 链接文件

(2) 显式链接

- ◆ 把用于链接文件的各物理块指针显式地放在内存的一张表格中，表格的序号是物理块号，从0开始，直到N-1，N为整个磁盘的盘块的总数，表格的每个表项中存放链接指针即下一个盘块号。
- ◆ 该表格就是FAT（12,16,36）文件系统中的FAT表
- ◆ MS-DOS和windows早中期操作系统一直采用的文件系统技术。

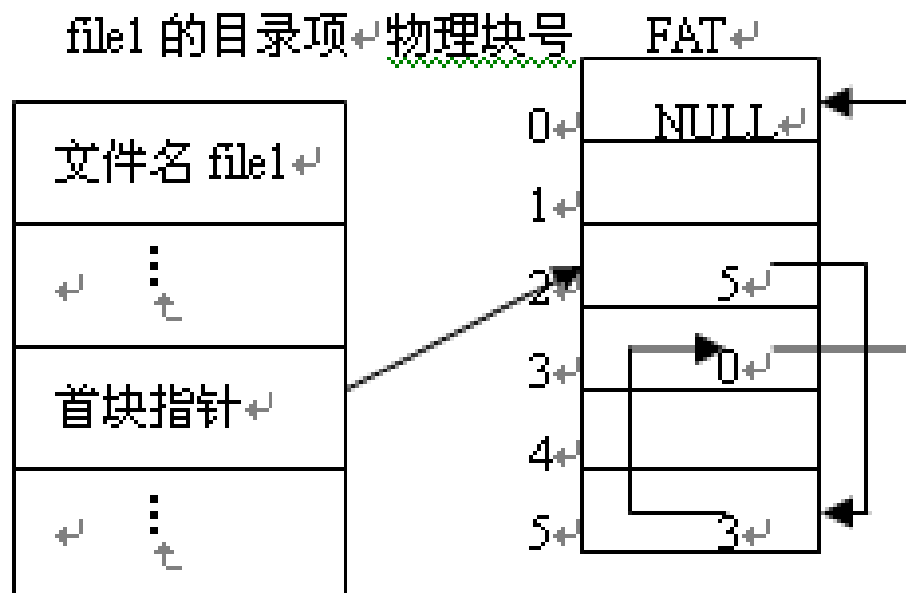
5

5.2.3 文件的物理结构

2. 链接文件

(2) 显式链接

- 在文件的目录项的“物理地址”字段中只要保存该文件的起始块号



5

5.2.3 文件的物理结构

2. 链接文件

(2) 显式链接

◆ 主要缺点：

- (1) 不支持高效的直接存取，对一个较大的文件进行存取时，须在FAT中顺序地查找许多盘块号；
- (2) FAT表本身需要占用较大的内存空间。

5

5.2.3 文件的物理结构

3. 索引文件

- ◆ 索引分配结构：系统为每个文件建立一个索引表，集中记录该文件占用的盘块号。索引表可以直接存放在文件控制块FCB中，但是，大文件的索引表往往很大，所以大多数文件系统让索引表置于单独的物理块中且可驻留在磁盘的任意位置，文件控制块的“物理地址”字段只要保存该索引表所在的盘块号（即索引表地址）。

5

5.2.3 文件的物理结构

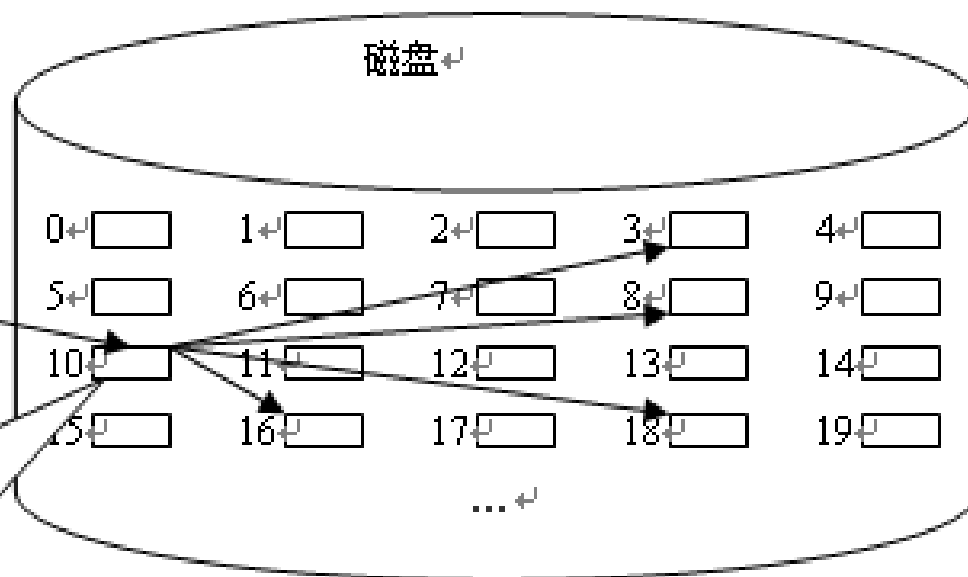
3. 索引文件

无键索引表的组织形式

file1 的目录项

文件名 file1
⋮
索引表地址 10
⋮

3
8
16
18



5

5.2.3 文件的物理结构

3. 索引文件

◆ 索引表可以有不同的索引形式：

- (1) 无键索引表：索引表记录的是组成指定文件的**磁盘块号**，这种索引只是盘块号的序列，适用于**流式文件**；
- (2) 有键索引表：该索引表的**索引表项包含逻辑记录键及其磁盘块号**，指出了每条逻辑记录在磁盘上的存放地址，即物理块号，适用于纪录式文件。

5

5.2.3 文件的物理结构

3. 索引文件

有键索引表的组织形式

file1 的目录项

文件名 file1
⋮
索引表地址 10
⋮

逻辑记录键	物理块号
key1	3
key2	8
key3	16

磁盘

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
...				

5

5.2.3 文件的物理结构

3. 索引文件

优点：

- ◆ 实现了离散分配
- ◆ 有利于直接存取。

缺点：

- (1) 索引表增加了空间开销，
- (2) 不适合中小型文件，对于中小型文件会有空间的浪费。因为往往一个盘块能存放数百个盘块号，但对于中小型文件，本身通常只占用数个到数十个盘块，甚至更少，但仍须为之分配以索引块。
- (3) 索引表的查找增加了时间开销。

5

5.2.3 文件的物理结构

4. 多级索引文件

- ◆ 以上索引文件为**一级索引文件**：存放文件内容所占物理块号的盘块也称作一级索引块
- ◆ 一级索引存在的问题：当索引表本身占用许多物理块时，在查找某键对应的索引项时，可能依次需交换很多块。若索引表占用 n 块，则平均要交换 $(n+1) / 2$ 次，才能找到所需记录的物理地址，当 n 很大时，这是很费时间的操作。

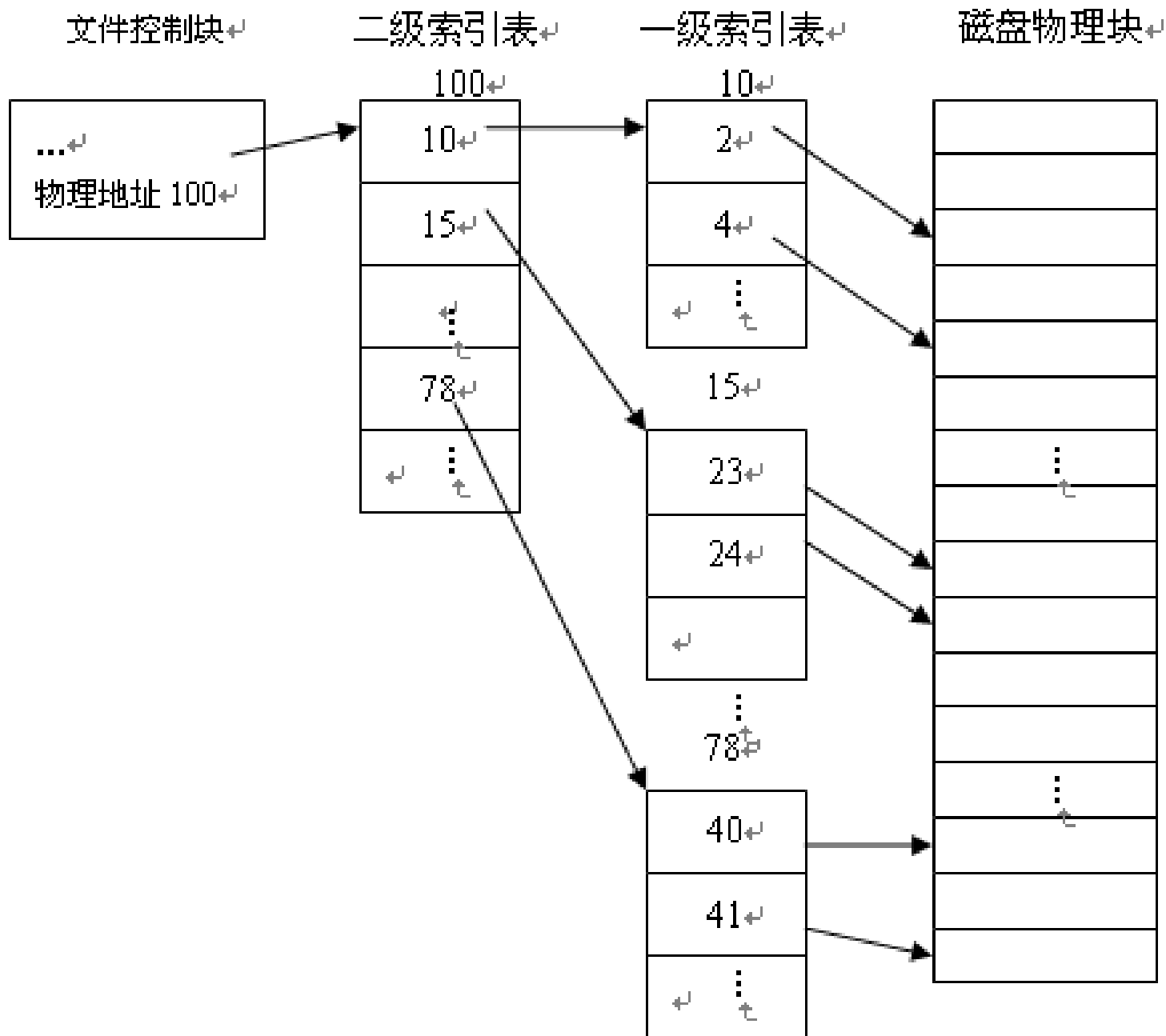
5

5.2.3 文件的物理结构

4. 多级索引文件

- ❖ 解决方法：为了提高大文件的查找速度，可以为这些一级索引块再建立一个索引，称为二级索引，即系统再分配一个盘块记录一级索引块的块号。
- ❖ 如果文件再大，还可以建立三级、四级索引，依次类推。

5

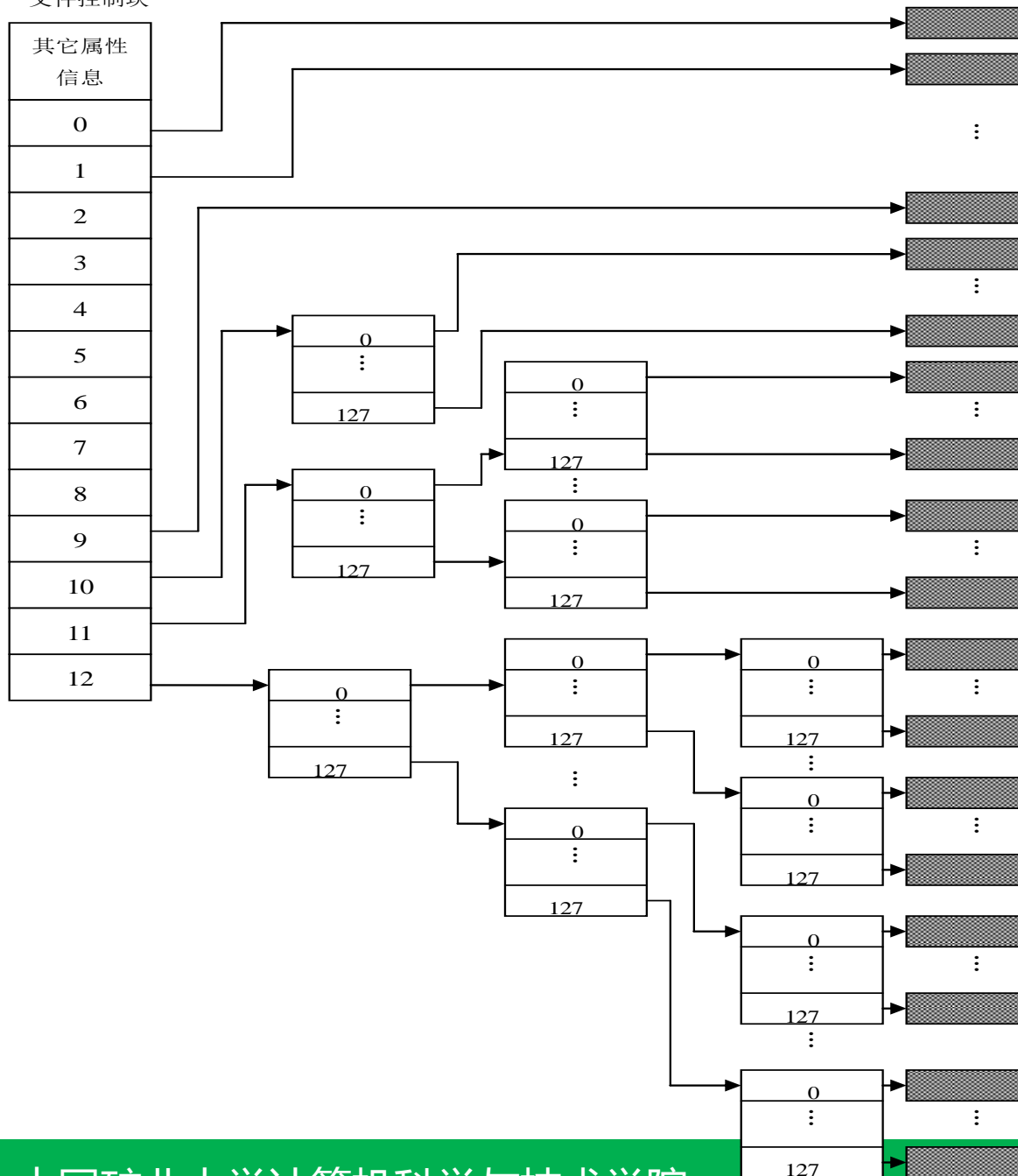


5

5.2.3 文件的物理结构

4. 多级索引文件

- ◆ 在Unix系统中，为了能兼顾到小、中、大以及特大型文件，采取了**混合索引结构**。
- ◆ 它在文件控制块的地址部分设置了13个地址项，即*iaddr(0)-iaddr(12)*，混合了四种寻址方式：**直接寻址、一级索引寻址、二级索引寻址和三级索引寻址**。
- ◆ **iaddr(0)-iaddr(9)**存放了直接寻址指针，即该指针指向的盘块直接存放了该文件的数据内容，如果文件再大就可以用到第11、12和13个地址项（**iaddr(10)-iaddr(12)**），分别存放的是一级索引地址、二级索引地址和三级索引地址。



假设物理块的大小为512字节，每个盘块号（即地址）占用4个字节，则每个物理块最多能存放128个地址，则采用混合索引结构所能容纳的最大文件容量是：
 $10 + 128 + 128^2 + 128^3 = 2113674$ (块)。

5

5.2.3 文件的物理结构

4. 多级索引文件

练习题：如果物理块大小为512字节，每个数据块寻址需要3个字节那么Unix的3级索引方法可以支持一个文件的最多容量是多少？

- (1) 直接块数=10块
- (2) 一级索引块数= $512/3=170$ 块
- (3) 二级索引块数= $170*170=28900$ 块
- (4) 三级索引块数= $170*170*170=4913000$ 块
- (5) 文件容量= $(10+170+28900+4913000) * 0.5k$

$\approx 2.35GB$

5

5.2.3 文件的物理结构

5. 直接文件

- ◆ 利用哈希函数直接建立逻辑记录的关键字与其物理地址的对应关系的文件组织形式。
- ◆ 直接文件只能在直接存取的存储设备（如磁盘）上实现，并且主要用于记录式文件。
- ◆ 优点：（1）记录在介质上不需要按序存放，因为它能根据关键字直接计算出物理地址，所以最适合直接存取；（2）相对于索引文件，它不需索引，节省了索引存储空间和索引查找时间。

5

5.2.3 文件的物理结构

5. 直接文件

- ◆ **适用场合：** 直接文件在不能采用顺序组织方法、次序较乱、又需在极短时间内存取的情况下极为适合，例如操作系统目录文件、实时处理文件、存储管理的页表查找、编译程序变量名表等采用直接文件特别有效。
- ◆ **主要缺点：** 需要解决冲突问题。因为地址的总数和关键字之间不存在一一对应关系。

5

5.2.4 文件的存取方法

- ◆ 存取方法：如何存储和检索文件数据的方法，它是操作系统为用户程序提供的使用文件的技术和手段。
- ◆ 存取方法和文件的逻辑结构、物理结构以及存储介质有密切的关系，许多操作系统都提供多种存取方法，而在一些文件处理密集的系统软件中，如数据库管理系统，文件存取的方法就更加丰富。
- ◆ 常用的方法：**顺序存取，直接存取，按键存取**

5

5.2.4 文件的存取方法

顺序存取

- ◆ 按照记录的排列顺序或字符的先后顺序逐个存取的方法叫做顺序存取，一般要设置存取文件的读写指针。
- ◆ 这种存取方式最为简单，在磁带设备上的存取效率最高。顺序组织的文件，即逻辑结构上的顺序文件或物理结构上的连续文件，都适合顺序存取，特别对于文件的批量存取，效率最高。

5

5.2.4 文件的存取方法

按键存取

- ◆ 按键存取是直接存取的一种特例，它是指按给定的键值存取对应的记录。由于文件中的记录不按它在文件中的位置，而按它的记录键来编址，所以，用户提供给操作系统记录键后就可查找到所需记录。
- ◆ 索引组织的文件最适合该方法，因为索引文件都会按照关键字建立索引表，指定关键字来存取非常方便。

5

5.2.5 文件存储空间管理

- ◆ 文件管理的主要功能之一是如何在外部存储介质上为创建或扩充文件而**分配空间**，为删除文件而**回收空间**以及**对空闲空间的管理**。
- ◆ 磁盘可以随机存取的特性非常适合文件系统的实现，因此磁盘是最常用的文件外部存储介质
- ◆ 文件存储空间管理主要涉及两个问题：
 - 一是**磁盘空间的分配**
 - 二是**磁盘空闲空间的有效管理**

5

5.2.5 文件存储空间管理

一、磁盘空间的分配

- ◆ 文件的物理结构和磁盘空间的分配方法密切相关，当用户要创建文件或扩充文件时决定分配哪些磁盘块是很重要的，这将会影响今后磁盘的访问次数和文件存取效率。
- ◆ 磁盘空间的分配常采用以下两种办法：
 - (1) 连续分配
 - (2) 非连续分配

5

5.2.5 文件存储空间管理

一、磁盘空间的分配

(1) 连续分配

- ◆ 文件被存放在外存空间连续存储区 (连续的物理块号) 中，建立文件时，用户必须给出文件大小，然后，查找到能满足的连续存储区供使用，否则文件不能建立，用户进程必须等待。
- ◆ 优点：顺序访问时通常无需移动磁头，文件查找速度快，管理较为简单
- ◆ 缺点：为了获得足够大的连续存储区，需定时进行‘碎片’收集。因而，不适宜于文件频繁动态增长和收缩的情况，用户事先不知道文件长度也无法进行分配

5

5.2.5 文件存储空间管理

一、磁盘空间的分配

(2) 非连续分配

- ◆ 非连续分配是指以物理块（扇区）为单位，按文件动态要求分配物理块，这些物理块不一定要连续，属于同一文件的块按文件记录的逻辑次序用链接指针连接或用索引表指示，即**链接分配和索引分配**。
- ◆ 优点：辅存空间管理效率高，便于文件动态增长和收缩，访问文件执行速度快，特别是以簇为单位的分配方法已被广泛使用。

5

5.2.5 文件存储空间管理

一、磁盘空闲空间的管理

◆ 为了记录空闲磁盘空间，也就是那些尚未分配给文件或目录的块，需要采用一定的数据结构来实现：

- (1) 空闲区表法；
- (2) 空闲块链表法；
- (3) 位示图法

5

5.2.5 文件存储空间管理

一、磁盘空闲空间的管理

(1) 空闲区表法

- ◆ 系统为外存上的所有空闲块建立一张空闲区表，每个表项记录了一个空闲区，主要包括该空闲区的第一个空闲盘块号、该区的空闲盘块数和状态等信息，再将所有的空闲区按其起始盘块号递增的次序排列。

首块号	空闲块数	表目状态
123	14	未用
137	7	已用
...
456	20	未用

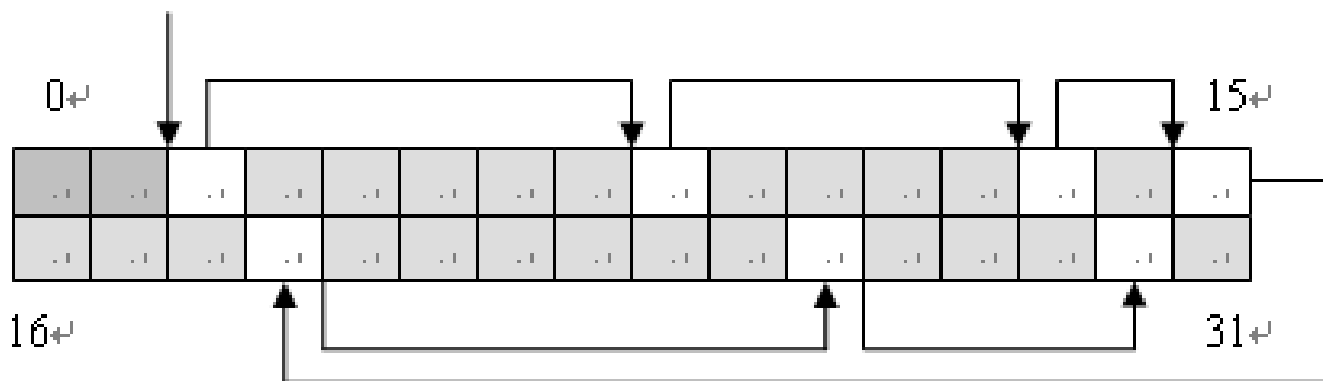
5

5.2.5 文件存储空间管理

一、磁盘空闲空间的管理

(2) 空闲块链表法

- ◆ 空闲块链表法是将所有空闲块连接在一起，组成一个空闲块链表
- ◆ 空闲块链表的一个变种是空闲盘区链，将磁盘上的所有空闲盘区（每个盘区可包含若干个盘块）组成一个链表，也适合连续文件的组织形式。



5

5.2.5 文件存储空间管理

一、磁盘空闲空间的管理

(3) 位示图法

- ◆ 空闲空间表还可由位图或位矢量的方法来实现。每一个磁盘块由1位（bit）来表示。如果该磁盘块是空闲的，这个位就置0；否则，就置1。

00101100 00111111 00000011 11111000



目录

CONTENTS

5.1 概述

5.2 文件的组织

5.3 文件目录

5.4 文件系统调用的实现

5.5 文件共享

5.6 文件系统体系结构

5.7 Windows文件系统

5

5.3 文件目录

◆ 目录管理的主要任务如下：

- 1) 对大量文件实施有效的管理。
- 2) 实现“按名存取”。
- 3) 提高对目录的检索速度。
- 4) 支持文件共享和文件重名。

5

5.3.1 文件目录的基本概念

1. 文件控制块

- ◆ 为了能对文件进行正确的存取，必须为**文件设置用于描述和控制文件的数据结构**，称之为文件控制块（File Control Block, FCB）。
- ◆ 每当创建一个文件，系统都要为其建立一个FCB，FCB和文件是一一对应的。每个文件有两部分组成：

FCB（文件属性信息）

文件体（文件内容信息）

5

5.3.1 文件目录的基本概念

1. 文件控制块

FCB一般包括以下文件属性信息：

- ◆ ① 有关文件存取控制的信息。
- ◆ ② 有关文件结构的信息。文件的逻辑结构，如记录类型、记录个数、记录长度、成组因子数等。文件的物理结构，如文件所在设备名，文件物理结构类型，记录存放在外存的相对位置或文件第一块的物理块号，也可指出文件索引的存放位置等。
- ◆ ③ 有关文件使用的信息。已打开该文件的进程数，文件被修改的情况，文件最大和当前大小等。
- ◆ ④ 有关文件管理的信息。如文件建立日期、文件最近修改日期、文件访问日期、文件保留期限、记帐信息等。

5

5.3.1 文件目录的基本概念

1. 文件控制块

- ◆ **FCB的重要作用：**建立文件名与外存空间中的物理地址的对应关系，从而实现“按名存取”。
- ◆ **主要过程：**系统存取文件时，先找到其**FCB**，从**FCB**中再找到文件信息盘块号或首块物理位置等物理地址信息就能存取文件信息。

5

5.3.1 文件目录的基本概念

2. 文件目录和目录文件

- ◆ 为了加快文件的查找速度，通常把FCB集中起来进行管理，文件控制块的有序集合称为文件目录，即一个文件控制块就是一个文件目录项。
- ◆ 文件目录也是以文件的形式保存在外存上的，这就形成了目录文件。
- ◆ 因此，文件目录的目录项有两种，一种是用于描述子目录（即目录文件）的FCB，一种是普通文件的FCB。

5

5.3.1 文件目录的基本概念

2. 文件目录和目录文件

◆ 目录文件与普通文件不同之处：

目录文件永远不会空，它至少包含两个目录项

当前目录项 “.”

父目录项 “..”

注意：要搜索到某个文件的外存物理地址，必须先
从文件的最外层目录逐层搜索匹配比较。文件目录
和FCB是现实“按名存取”的重要数据结构。

5

5.3.2 目录文件的组织

- ◆ 目录文件的组织是指目录项的设计和FCB的存储组织方法。
- ◆ 不同的组织方法直接影响到检索文件的速度，对整个文件系统的效率、性能和可靠性都有很大的影响。
- ◆ 常用的组织方法主要有三种：
 1. FCB线性表
 2. 索引节点
 3. 哈希表组织

5

5.3.2 目录文件的组织

1. FCB线性表

- ◆ 目录文件中直接存放该目录下所有文件和子目录的FCB信息，组成了一个FCB线性表，即每个目录项记录了该文件或子目录对应的完整FCB信息

文件名	访问权限	属主	物理地址
file1	--	--	--
.....

每一行就是一个FCB

5

5.3.2 目录文件的组织

1. FCB线性表

- ❖ **缺点：**这样组织FCB，当文件很多时，目录文件本身可能要占用大量的盘块数，使得目录检索的速度很慢。
- ❖ **例如：**设目录文件所占用的盘块数为 N ，按此方法查找，则查找一个目录项，平均需要调入盘块 $(N+1)/2$ 次。假如一个FCB为128B，盘块大小为512B，则每个盘块中只能存放4个FCB。若一个文件目录中共有320个FCB，需占用80个盘块，则平均查找一个文件需启动磁盘40次。

5

5.3.2 目录文件的组织

2. 索引节点

- ◆ 基本思想：在检索目录文件的过程中，其实只用到了文件名，仅当找到一个目录项（即其中的文件名与指定要查找的文件名相匹配）时，才需从该目录项中读出该文件的物理地址。而其他一些对该文件进行描述的信息，在检索目录时一概不用。显然，这些信息在检索目录时，不需要调入内存。

索引节点

5

5.3.2 目录文件的组织

2. 索引节点

- ◆ 索引节点：把文件目录项中的文件名和其它描述信息分开，后者单独组成定长的一个数据结构，称为索引节点（i- node或inode也称i节点）
- ◆ 索引节点号：一个文件系统中的所有文件的索引节点都集中存放在外存上的索引节点区，并对每个索引节点进行编号（i节点号）。
- ◆ 文件目录项：不再是完整的FCB，只要存放14个字节的文件名和2个字节的i节点号

5

5.3.2 目录文件的组织

2. 索引节点

◆ 采用索引节点的
目录项 (Unix)

文件名	索引节点号
file1	1
file2	32
...	...

0

13 14

15

检索文件的过程：检索文件时，先从目录文件中找到文件名匹配的目录项，在目录项中找到该文件的索引节点号，根据索引节点号就可以在索引节点区中找到该文件的索引节点，找到了i节点，就获得了它所对应的文件的一切必要属性信息。

5

5.3.2 目录文件的组织

2. 索引节点

索引节点的好处:

- ◆ 通过索引节点的组织方式大大提高了目录的检索速度
- ◆ 举例：通过索引节点的方式，一个512字节的物理块可存放32个文件目录项，可使找到一个文件的平均启动磁盘次数减少到原来的1/8，大大减少了系统开销。

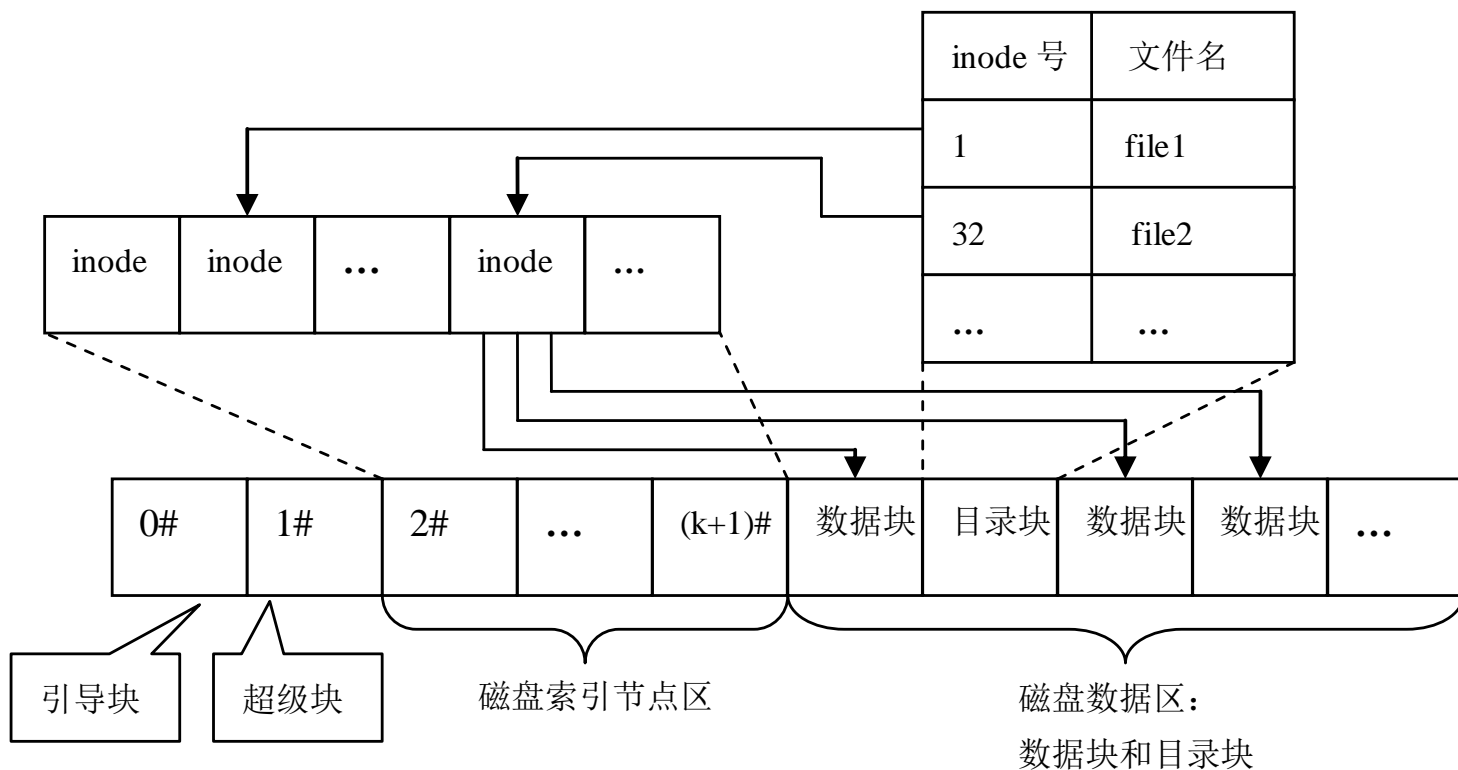
5

5.3.2 目录文件的组织

2. 索引节点

目录项、索引节点和数据块之间的关系

目录文件



5

5.3.2 目录文件的组织

2. 索引节点

- ◆ 外存索引节点存在的问题：由于访问某一文件的过程中，会频繁地涉及到文件的索引节点，不断在内、外存之间引用它，系统消耗很大。
- ◆ 解决方法：内存（活动）索引节点表，正在被使用的索引节点信息保存到内存索引节点表。
- ◆ 在系统占用的内存区里专门开辟一张表（例如100个表目）每个表目称为一个内存活动i节点。

总结：磁盘索引节点反映了文件的静态特性，活动索引节点除了这些已有的静态信息外

5

5.3.2 目录文件的组织

3. 哈希表组织

- ◆ 以上两种目录文件都是以线性表组织目录项的，检索时都是采用线性搜索的算法。
 - FCB线性表方法中，目录文件中存放的是所有文件和子目录的完整FCB信息；
 - 基于索引节点的目录文件存放的是所有文件和子目录的名称和索引节点号，FCB信息通过索引节点的方式专门存放在统一的索引节点区。
- ◆ 为了加快目录检索的速度，目录文件还可以采用哈希表存储（利用散列方式管理FCB）。

5

5.3.2 目录文件的组织

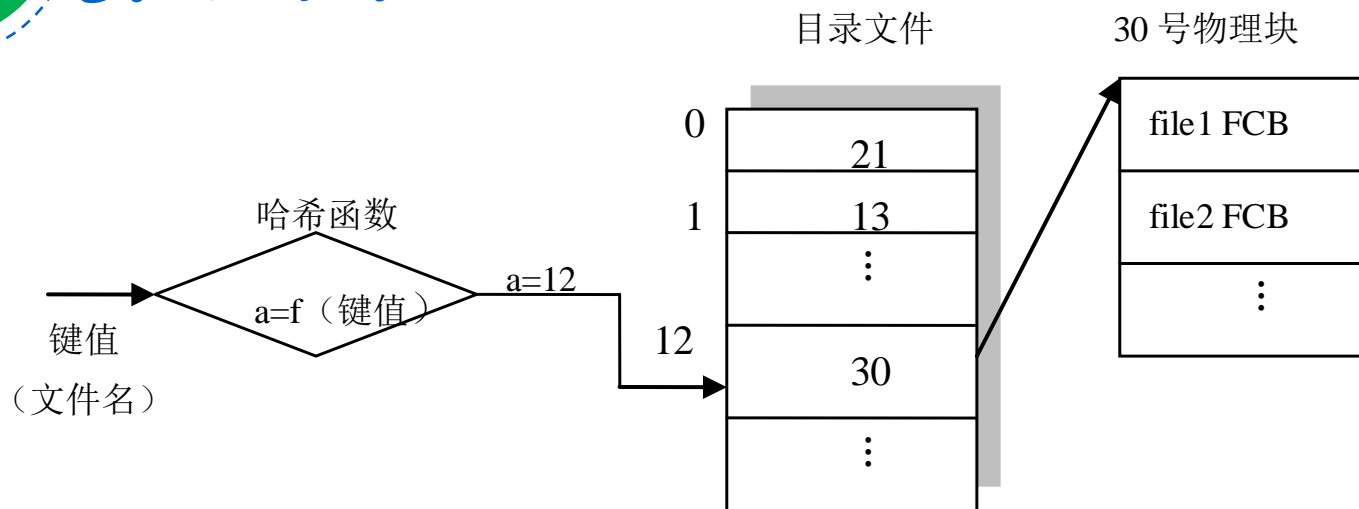
3. 哈希表组织

- ◆ 具体实现：目录文件每个目录项中存放一个块号，每个目录项都设一个序号（类似于数组下标），哈希函数以文件名作为自变量计算散列函数的值，以该值作为序号直接定位的目录文件中的目录项，然后在目录项中找到块号，该块号所指的块内就存放了该文件的FCB。
- ◆ 具有相同散列值的文件的FCB放在一个物理块内，如果相同散列值的文件数目超过一个物理块能存放FCB的数目就产生冲突，需要作溢出处理。

5

5.3.2 目录文件的组织

3. 哈希表组织



该方法可以大幅度地减少目录搜索时间。插入和删除目录项都很简单，只需要考虑两个目录项冲突的情况，即两个文件名返回的数值一样的情形。哈希表的主要难点是选择合适的哈希表长度与适当的哈希函数。

除了线性表和哈希表这两种检索目录的数据结构，还可以采用其它数据结构，如B+树

5

5.3.3 目录的结构

- ◆ 目录结构的组织，不但关系到文件系统的检索速度，还关系到文件的共享性和安全性。
- ◆ 目录结构都是采用层次结构，主要分为：

单级目录

二级目录

多级层次目录结构

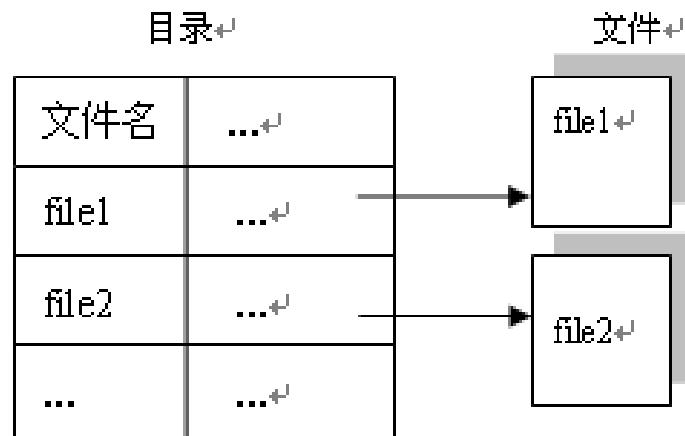
图状目录结构

5

5.3.3 目录的结构

1. 单级目录

- ◆ 整个文件系统中仅建立和维护一张总的目录表，系统中所有文件都在该表中占有一项。



5

5.3.3 目录的结构

1. 单级目录

◆ 优点：实现简单

◆ 缺点：

(1) 不允许文件重名。

(2) 文件查找速度慢。

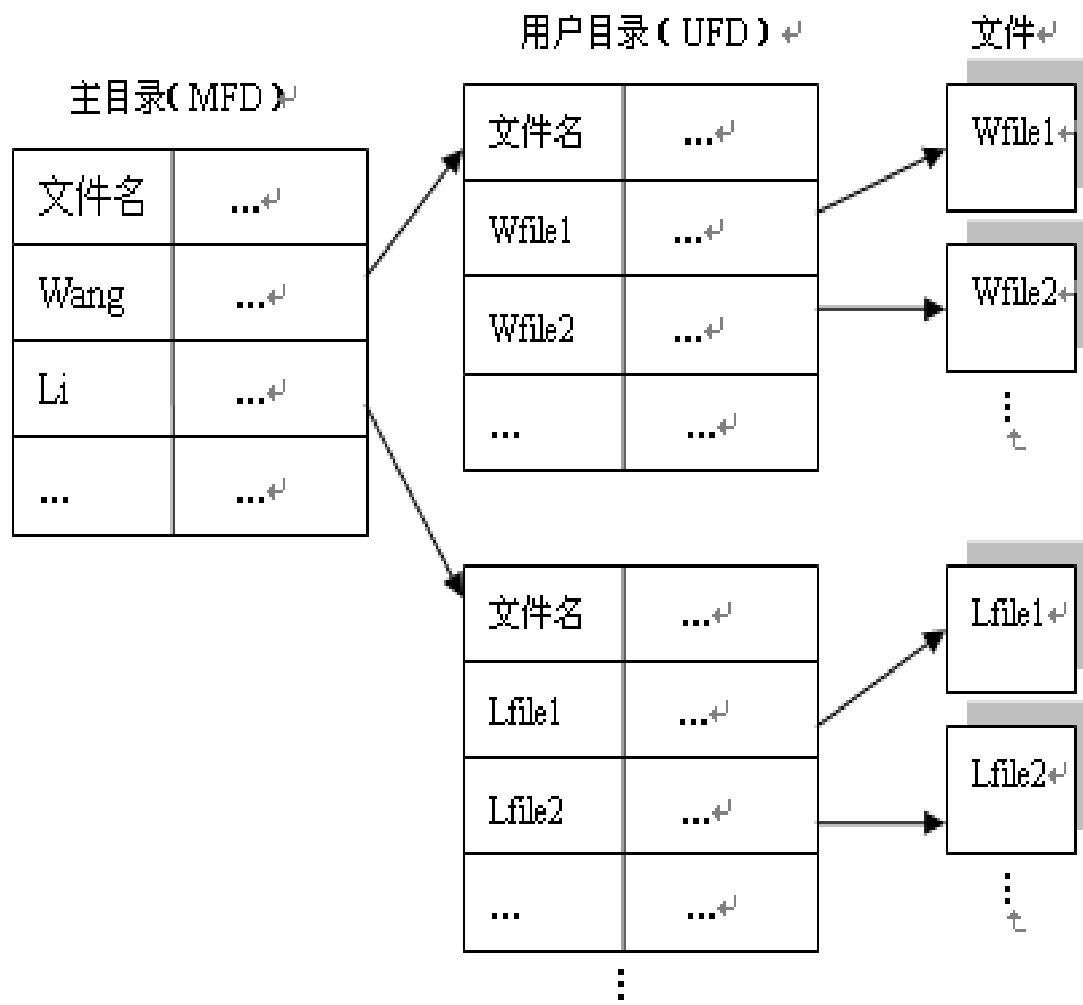
(3) 不便于实现文件共享。单级目录结构只适合于单用户操作系统。

5

5.3.3 目录的结构

2. 二级目录

- 二级目录可以解决文件重名，即把系统中的目录分为一个主目录（Master File Directory, MFD）和多个次目录（User File Directory, UFD）。
- 在多用户系统中，一般每个用户都拥有一个属于自己的次目录UFD，而主目录MFD则存储着各个UFD的信息，标明各个UFD的名称、物理位置等。



5

5.3.3 目录的结构

2. 二级目录

- ◆ 当使用文件时，用户必须给出用户名和文件名。系统根据用户名在主目录中找到该用户目录，再根据文件名在用户目录中找到文件的物理地址。
- ◆ **优点：**
 - （1）提高了文件检索速度（用户名大大缩小了需要检索的文件数量）
 - （2）部分地允许文件名重名。
- ◆ **缺点：**对同一用户，也不能有两个同名的文件存在，限制了共享；多个用户完全隔离开来不便于实现多个用户协同完成任务。

5

5.3.3 目录的结构

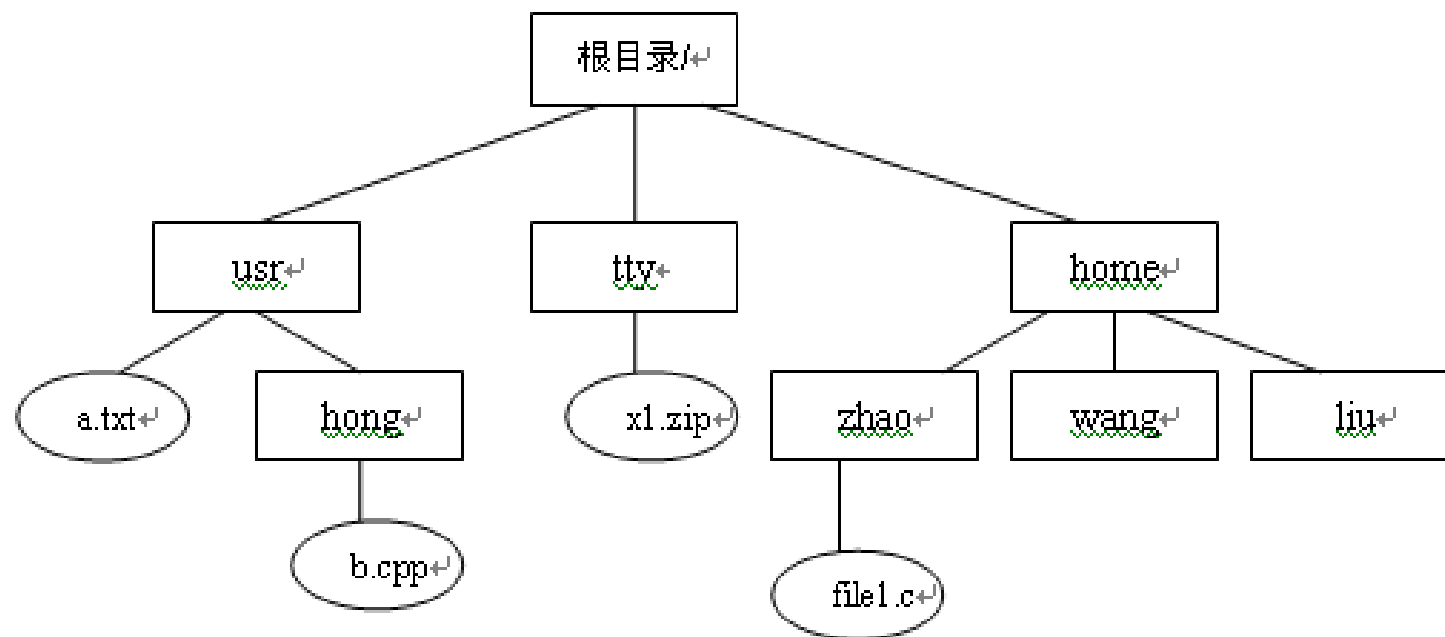
3. 多级层次目录

- ◆ 多级层次目录（树形目录）可以明显地提高对目录的检索速度和文件系统的性能。
- ◆ 在多级层次目录结构中，有一个根目录和许多分目录，根目录是唯一的，每一级目录不但可以包含文件，而且还可以包含下一级的分目录，从而形成了树状目录结构。

5

5.3.3 目录的结构

3. 多级层次目录



5

5.3.3 目录的结构

3. 多级层次目录

几个概念：

- ◇ 路径名（path name）
- ◇ 绝对路径（absolute path name）
- ◇ 当前目录（current directory）
- ◇ 相对路径名（relative path name）

5

5.3.3 目录的结构

3. 多级层次目录

◆ 优点:

(1) 既可方便用户查找文件，又可以把不同类型和不同用途的文件分类。

(2) 允许文件重名。

(3) 利用多级层次结构关系，可以更方便地制定保护文件的存取权限

◆ 缺点: 不能直接支持文件或目录的共享，因为这种纯树型目录中每个文件只能有一个父目录

5

5.3.3 目录的结构

4. 图状目录结构

- ◆ **有向无环图目录结构：**允许一个文件有多个父目录，不同的目录可以以相同或不同的文件名共享同一个文件，形成的目录结构图不会出现环（因为不支持目录共享）
- ◆ **有环图目录结构：**除了允许文件被共享，还支持目录被共享，则目录结构图中就有可能出现环

5

5.3.3 目录的结构

4. 图状目录结构

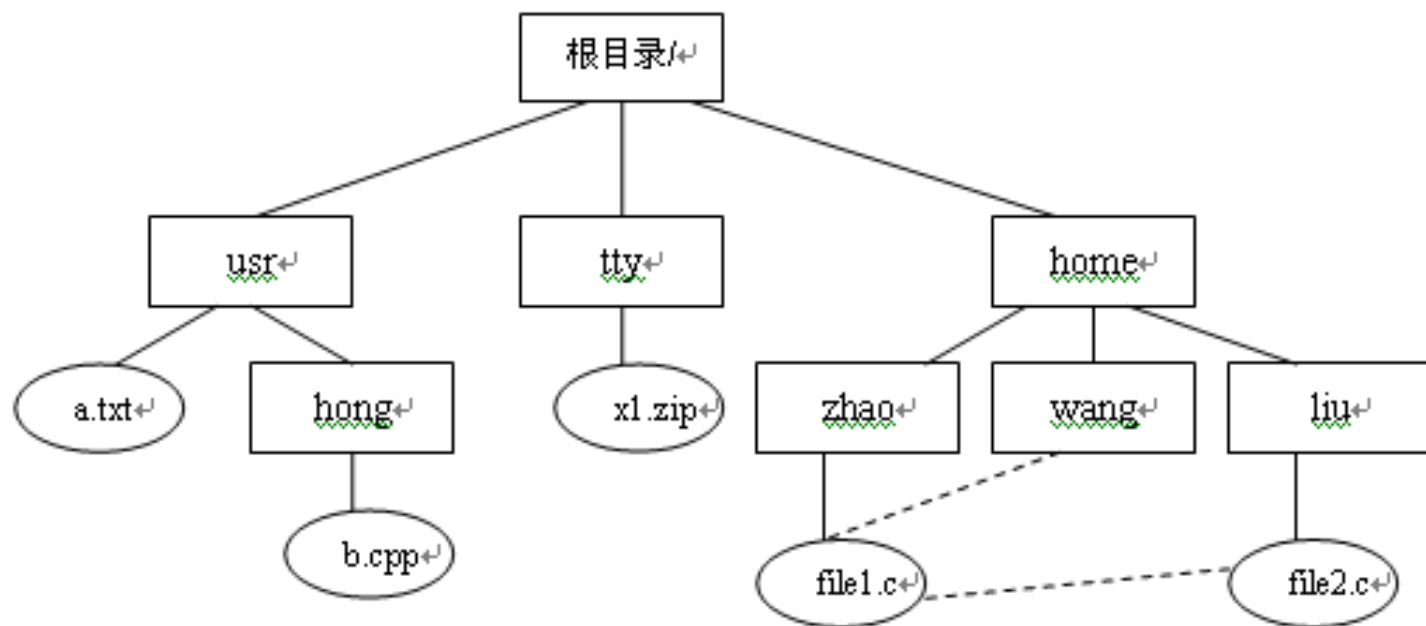


图5.23 (a) 无环图状目录结构 (文件共享)

5

5.3.3 目录的结构

4. 图状目录结构

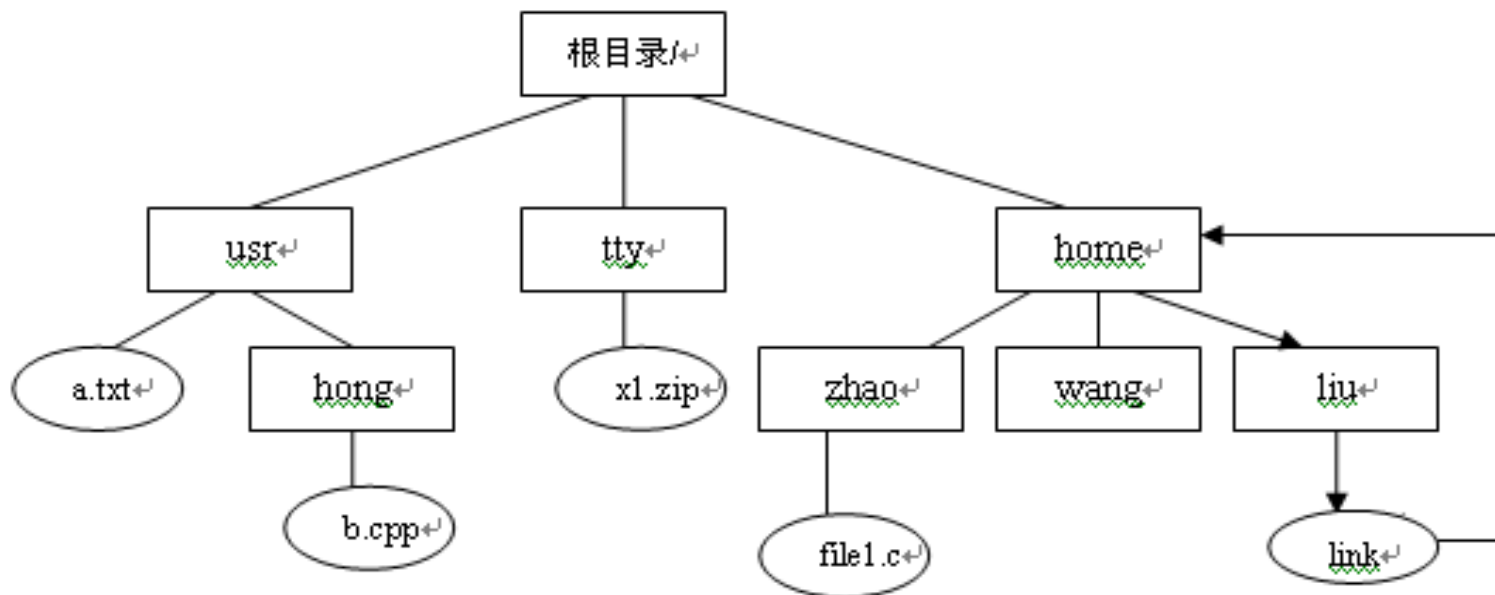


图5.23 (b) 有环图状目录结构 (目录共享)

5

5.3.4 目录的检索

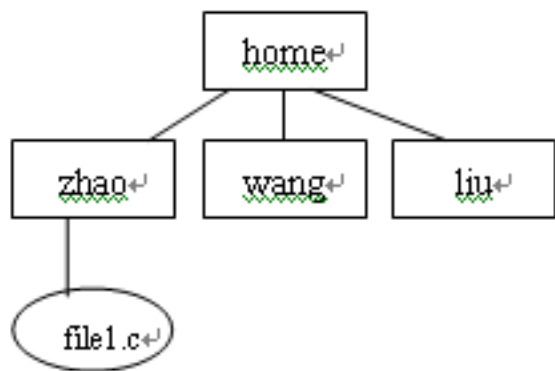
- ◆ 目录的检索是实现**按名存取**的重要基础。
- ◆ 在采用树形目录结构的文件系统中，要根据用户提供的文件路径名，从根目录或当前目录开始，逐级查找路径名中的各子目录名，用它们作为索引，逐层搜索各个目录文件，最后找到匹配的文件目录项。

5

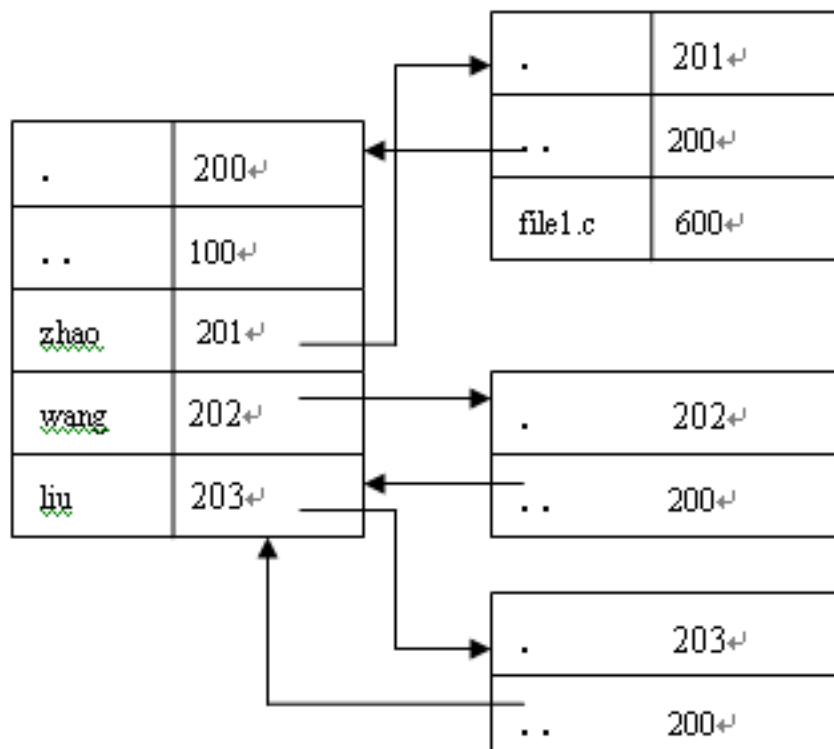
5.3.4 目录的检索

◆ 举例：以索引节点的目录组织方式介绍树型目录检索的过程

(a) 用户角度的目录结构



(b) 系统角度的目录结构



5

5.3.4 目录的检索

◆ 例如用户进程想要打开文件/home/zhao/file1.c时，文件系统开始检索

- (1) 首先，遇到根目录“/”，UNIX/Linux的根目录的索引节点会预先调入内存的活动索引节点表中，在内存活动索引节点表中找到根目录的活动索引节点，把它作为当前的工作索引节点，根据索引节点中的物理地址取出根目录文件内容，然后用字符串home和根目录文件中的目录项一一匹配，直到找到home对应的目录项，取出home目录文件对应的索引节点号，根据索引节点号在索引节点区中找到home目录文件的索引节点。
- (2) 根据home目录文件的索引节点中的物理地址取出home目录文件内容，用字符串zhao与各目录项一一匹配，即可找到zhao目录文件的索引节点。
- (3) 根据zhao目录文件的索引节点中的物理地址取出zhao目录文件内容，用字符串file1.c与各目录项一一匹配，即可找到file1.c文件的索引节点。
- (4) 根据file1.c文件的索引节点中物理地址即可存取该文件。

5

5.3.4 文件目录操作

- ◆ **创建目录：**在外部存储介质中，创建一个目录文件以备存取文件属性信息。
- ◆ **删除目录：**从外部存储介质中，删除一个目录文件。只有当目录为空时，才能删除。
- ◆ **改变目录：**用户可利用改变目录的命令，通过指定目录的绝对或相对路径名，设置当前目录。

5

5.3.4 文件目录操作

- ◆ **检索目录：**首先，系统利用用户提供的文件名，对文件目录进行查询，以找到相应的属性信息；然后，根据这些属性信息，得出文件所在外部存储介质的物理位置；最后，如果需要，可启动磁盘驱动程序，将所需的文件数据读到内存中。
- ◆ **移动目录：**允许文件或子目录在不同的父目录之间移动，文件或子目录经移动后，其文件的路径名将随之改变。

5

5.3.4 文件目录操作

- ◆ **链接操作：**通过链接操作，让指定文件具有多个父目录，从而实现文件共享。
- ◆ **打开目录：**如要使用的目录不在内存中，则需要打开目录，从外存上读入相应的目录文件。
- ◆ **关闭目录：**当所用目录使用结束后，应关闭目录以释放内存空间。



目录

CONTENTS

5.1 概述

5.2 文件的组织

5.3 文件目录

5.4 文件系统调用的实现

5.5 文件共享

5.6 文件系统体系结构

5.7 Windows文件系统

5

5.4.1 实现系统调用的相关数据结构

- ◆ 文件系统向应用程序提供了一组系统调用，包括创建、删除、打开、关闭文件和对文件的读写控制，通过这些系统调用，程序员能获得文件系统的各种服务。

面临的问题？？

文件系统在为用户程序服务时，经常需要沿路径查找目录以获得有关该文件的各种信息，这往往要多次访问外存，使访问速度大大减慢。若把所有文件目录都复制到内存，访问速度是加快了，但又增加了内存的开销。 如何解决？？

5

5.4.1 实现系统调用的相关数据结构

- ◆ 解决办法：把常用的和正在使用的那些文件目录复制进内存，这样，既不增加太多的内存开销，又可明显缩短查找时间。
- ◆ 具体实现：系统为每个用户进程建立一张打开文件表，并在系统中再维护一张记录系统中所有正在使用文件信息的系统打开文件表，正在使用文件的索引节点也会从外存索引节点区复制到内存索引节点表（即活动索引节点表）中。

5

5.4.1 实现系统调用的相关数据结构

- ◆ 使用文件时为什么要打开和关闭？
- ◆ 用户使用文件之前先通过“打开”操作，把此文件的文件目录信息（包括索引节点信息）复制到指定的内存区域，当不再使用这个文件时，使用“关闭”操作撤销该文件存放在内存的使用信息，以切断用户进程和该文件目录的联系。
- ◆ 文件被打开后，很多信息就已经调入内存，可被用户多次使用，直至文件被关闭或撤销，大大减少访盘次数，提高了文件系统的效率。

5

5.4.1 实现系统调用的相关数据结构

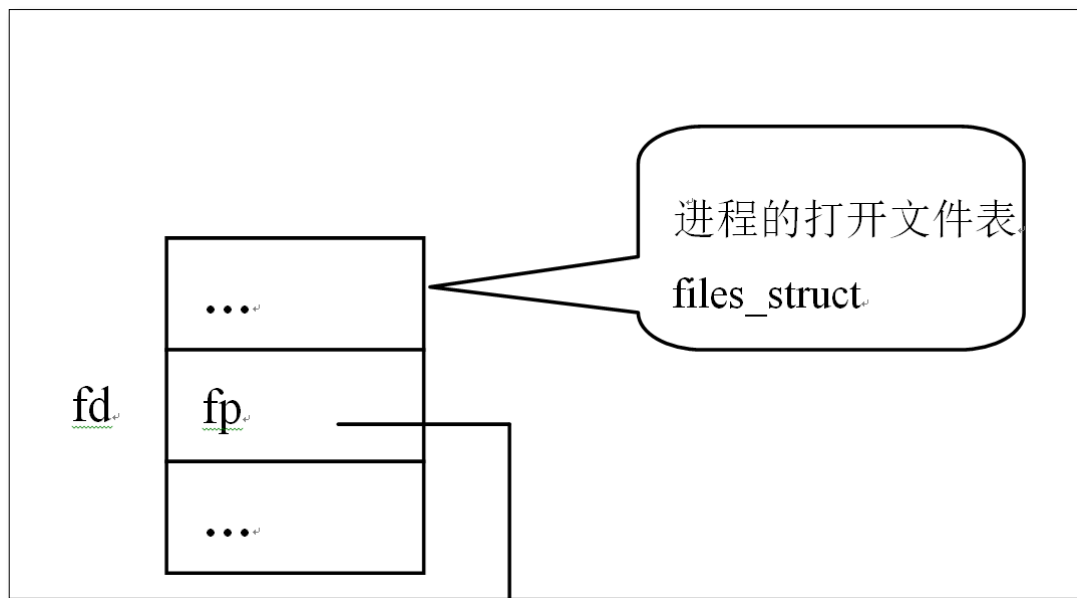
因此，

- ◆ 文件系统外存维护的超级块信息、目录信息和索引节点信息等可以看作是文件系统的**静态描述**，**用户打开文件表**、**系统打开文件表**和**内存活动索引节点表**是表征文件系统管理过程中的**动态特征**的重要数据结构。

5

5.4.1 实现系统调用的相关数据结构

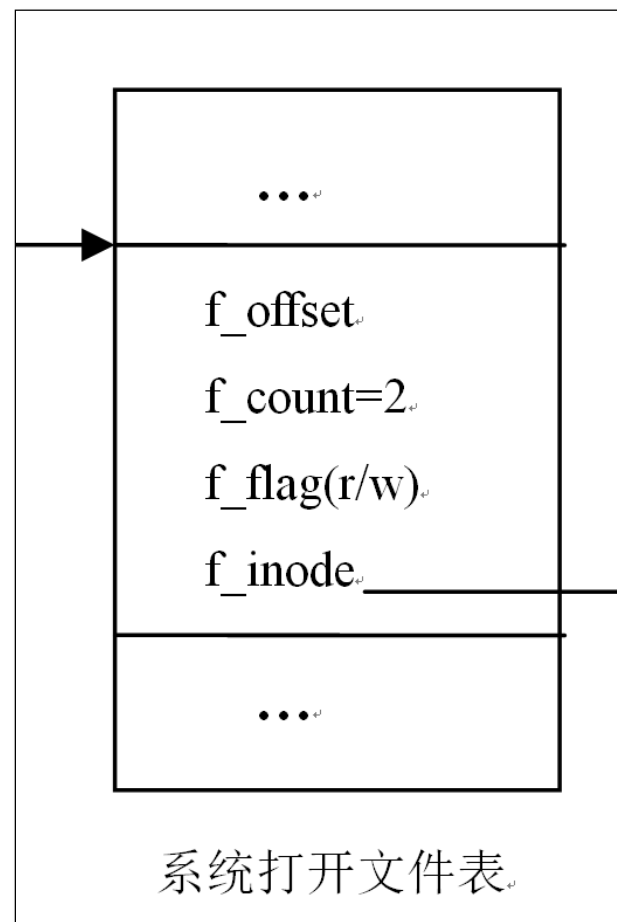
- ◆ 用户打开文件表：进程PCB结构中保留一个**files_struct**，称为用户打开文件表或文件描述符表。表项的序号是文件描述符fd，此登记项内登记系统打开文件表的一个入口指针fp，通过此系统打开文件表项连接到打开文件的活动inode。



5

5.4.1 实现系统调用的相关数据结构

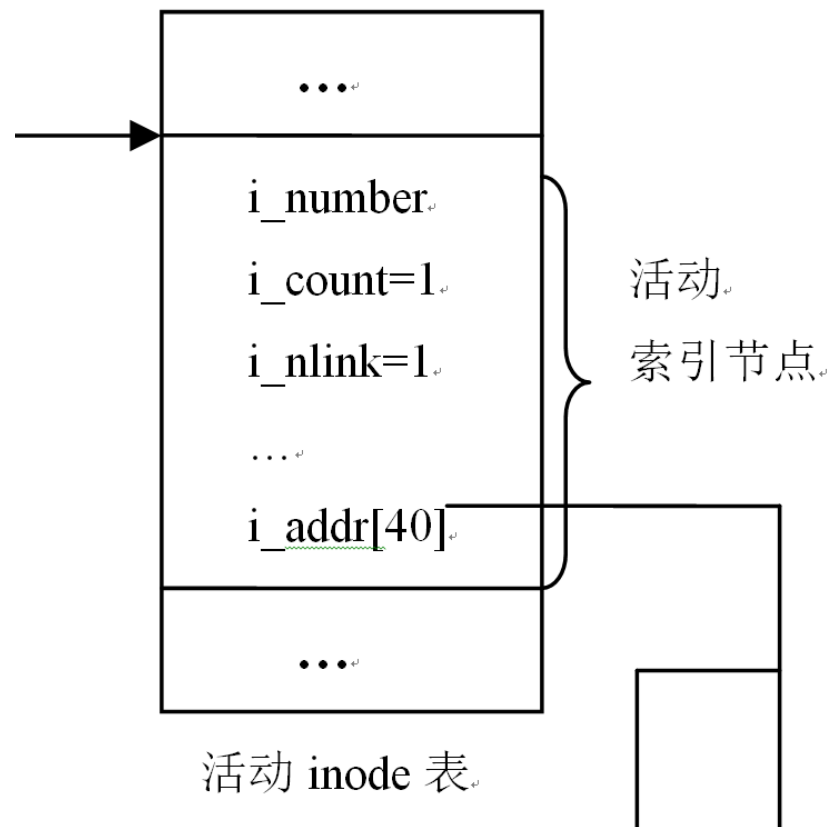
◆ 系统打开文件表：
file_struct, 主存专门开辟最多登记256项的系统打开文件表区，当打开一个文件时，通过此表项把用户打开文件表的表项与文件活动inode连接起来，以实现数据的访问和信息共享。



5

5.4.1 实现系统调用的相关数据结构

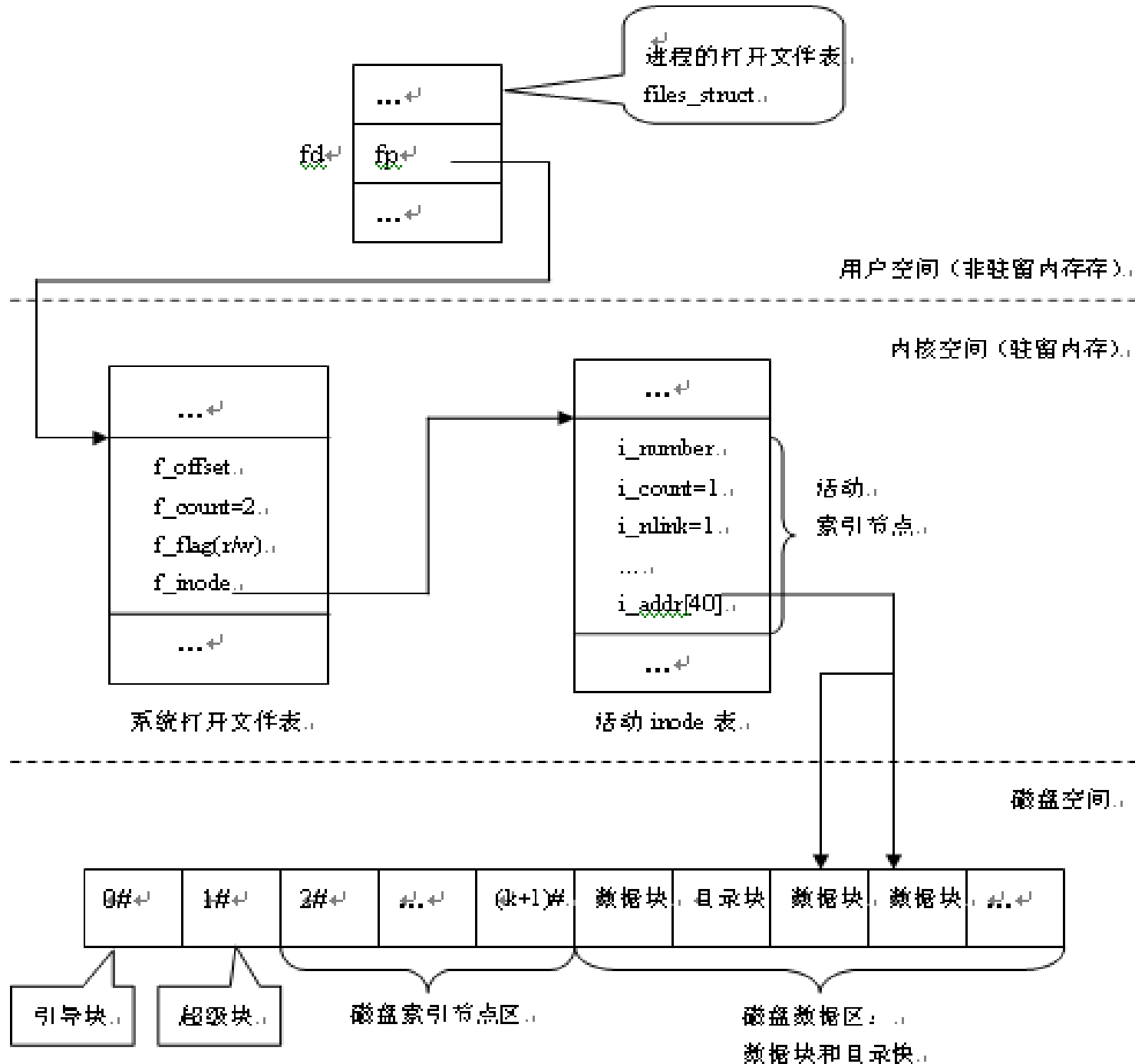
活动索引节点表：由于文件的索引节点中保存系统操作时，都对文件进行各种操作，都离不开文件属性信息。因此，在实现中也将被访问文件在磁盘中的索引节点复制到内存中，构成活动索引节点表，从而使对文件的访问更加便利和快捷。



5

5

内存各数据结构之间的关系



5

5.4.2 创建和删除文件

1. 创建文件:

```
fd=creat(pathname,mode);
```

```
int fd, mode;
```

```
char *pathname;
```

例如: `fd=create("/home/zhao/file1.c",0775);`

5

5.4.2 创建和删除文件

1. 创建文件:

```
fd=create("/home/zhao/file1.c",0775)
```

执行的具体过程:

- (1) 为新文件file1.c分配磁盘inode和活动inode，并把inode号与文件分量名file1.c组成新的目录项，记录到该文件目录路径/home/zhao的目录文件中。显然，在这一过程中，需要执行目录检索程序。
- (2) 在新文件所对应的活动inode中置初值，包括把存取权限i_mode置为0775，链接计数i_count置为“1”等。
- (3) 为新文件分配用户打开文件表项和系统打开文件表项，置系统打开文件表项的初值，包括在f_flag中置“写”标志，读写位移f_offset清0，等等；把用户打开文件表项、系统打开文件表项及file1.c所对应的活动inode用指针连接起来，最后，把文件描述符fd返回给调用者。

5

5.4.2 创建和删除文件

2. 文件的链接、解除链接和删除

link和unlink

- (1) **link**: 链接文件, 实现文件共享, $i_nlink+1$
- (2) **unlink**: 删除文件或解除链接, $i_nlink-1$

注意: 删除的任务是把指定文件从所在的目录文件中去除, 删除时如果没有链接用户(即 i_nlink 为“1”), 还要把文件占用的存储空间释放。

5

5.4.2 创建和删除文件

2. 文件的链接、解除链接和删除

◆ 例如依次执行以下步骤：

(1) 在/home/liu目录下创建一个
/home/zhao/file1.c的共享文件，文件名为file2.c，
则可执行以下系统调用：

```
link ("/home/zhao/file1.c ", "/home/liu  
/file2.c") ;
```

(2) 执行unlink("/home/liu /file2.c")

(3) 再执行unlink("/home/zhao/file1.c")

5

5.4.3 打开和关闭文件

- ◆ 打开：文件在使用之前必须先“打开”，以建立进程与文件之间的联系，而文件描述符唯一地标识了这样一种连接，其任务是把文件的磁盘索引节点复制到主存的活动索引节点表中，同时建立一个独立的读写文件数据结构，即系统打开文件表的一个表项。
- ◆ 关闭：活动索引节点表的大小受到容量的限制，这就要求用户一旦不再对文件进行操作时，应立即释放相应的活动索引节点，以便让其他进程使用

5

5.4.3 打开和关闭文件

1. 打开文件

◆ 打开文件的调用形式为：

```
fd=open (pathname,flags);
```

```
int fd, flags;
```

```
char *pathname;
```

实现过程：

- (1) 检查是否有其他进程已经打开该文件，如果有，则活动inode表中已有此文件的inode，只要把对应的活动inode中的i_count加1，i_count反映了通过不同的系统打开文件表项来共享同一活动inode的进程数目，它是以后执行文件关闭操作时，活动inode能否被释放的依据。

5

5.4.3 打开和关闭文件

1. 打开文件

实现过程:

(2) 如果是第一次打开该文件，则检索目录，要求打开的文件应该是已经创建的文件，它应登记在文件目录中，否则会出错。在检索到指定文件之后，就把它的磁盘索引节点复制到活动索引节点表中。

(3) 根据参数flags所给出的打开方式与活动索引节点中在创建文件时所记录的文件访问权限相比较，如果非法，则这次打开失败。

(4) 当“打开”合法时，为文件分配用户打开文件表项和系统打开文件表项，并为表项设置初值。通过指针建立这些表项与活动索引节点间的联系。把文件描述符fd（即用户打开文件表中相应文件表项的序号）返回给调用者。

5

5.4.3 打开和关闭文件

2. 关闭文件

- ◆ 文件使用完毕，执行关闭系统调用，切断应用进程与文件之间的联系。
- ◆ 调用形式为：`close(fd)`。
- ◆ 要关闭的文件先前已打开，故文件描述符`fd`一定存在，其执行过程如下：
 - (1) 根据`fd`找到用户打开文件表项，再找到系统打开文件表项。释放用户打开文件表项。

5

5.4.3 打开和关闭文件

2. 关闭文件

(2) 把对应系统打开文件表项中的f_count减“1”，如果非“0”，说明进程族（例如父子进程）中还有进程共享这一表项，不用释放此表项直接返回；否则释放表项，并找到与之连接的活动索引节点。

(3) 把活动索引节点中的i_count减“1”，若不为“0”，表明还有其他用户进程正在使用该文件，不用释放而直接返回，否则在把该活动索引节点中的内容复制回磁盘上的相应索引节点中后，释放该活动索引节点。

5

5.4.3 打开和关闭文件

2. 关闭文件

关于f_count和i_count(实现动态共享)

- ◆ f_count反映不同进程通过同一个系统打开文件表项共享一个文件的情况；
- ◆ i_count反映不同进程通过不同系统打开文件表项共享一个文件的情况。
- ◆ 由于文件的读写位移指针是存放在系统打开文件表项中的，所以前者实现的是使用相同位移指针的动态共享文件方式，主要适合同一进程族中的进程之间，即父子进程之间共享文件；后者实现的是使用不同读写位移指针的动态共享文件方式。

5

5.4.4 文件的读和写

◆ 读文件的系统调用为：

```
nr=read(fd, buf, count);
```

◆ 实现过程：首先根据f_flag中的信息，检查读操作合法性，如果读操作合法，按活动索引节点中i_addr指出的文件物理块存放地址，从文件的当前位移量f_offset处开始，读出所要求的字节数到块设备缓冲区中，然后再送到bufp指向的用户主存区中。

5

5.4.4 文件的读和写

- ◆ 写文件的系统调用的形式为:

`nw=write(fd, buf, count);`

- ◆ 其中fd、count和nw的含义类似于read中的含义，buf是信息传送的源地址，即把buf所指向的用户数据区中的信息写入文件中。

5

5.4.5 文件的随机存取

- ◆ 关于f_offset:在文件初次“打开”时，文件的位移量f_offset清空为0，以后的文件读写操作总是根据offset的当前值来顺序地读写文件。
- ◆ 为了支持文件的随机访问，文件系统提供了系统调用lseek，允许用户在读写文件之前，事先改变f_offset的指向。
- ◆ 系统调用的形式为：

lseek(fe, offset, whence);

(其中，当whence值为0时，则f_offset被置为参数offset的值；当whence值为1是，则f_offset被置为文件当前位置值加上offset的值。)



目录

CONTENTS

- 5.1 概述
- 5.2 文件的组织
- 5.3 文件目录
- 5.4 文件系统调用的实现
- 5.5 文件共享
- 5.6 文件系统体系结构
- 5.7 Windows文件系统

5

5.5 文件共享

- ◆ 文件共享可以有多种形式，在UNIX/Linux 系统中，允许多个用户静态共享，或动态共享同一个文件。当一个文件被多个用户进程动态共享使用时，每个进程可以各用自己的读写指针，也可以共用读写指针。
- ◆ 文件共享的方式：
 - 静态共享
 - 动态共享

5

5.5.1 静态共享

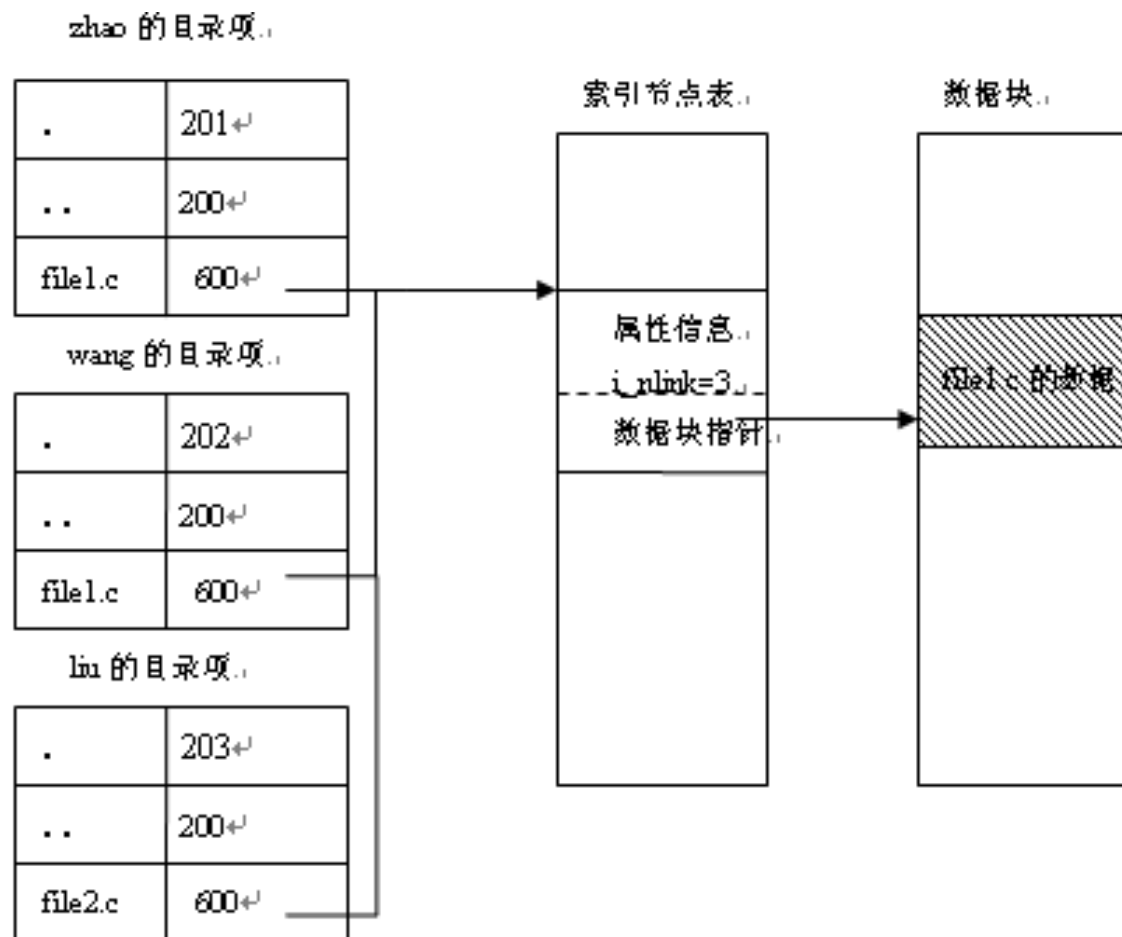
- ◆ 静态共享：文件或目录的共享关系**不管用户是否正在使用系统都存在的**共享方式。
- ◆ 文件链接：允许一个文件或目录有多个父目录，实际上仅有一处物理存储
- ◆ 静态共享主要有两种方式实现链接：
 - 一种是**基于索引节点的方式**，一般不适合目录共享，另一种是**符号链接共享**的方式，适合于文件也适合于目录

5

5.5.1 静态共享

1. 基于索引节点的链接共享

- 通过索引节点实现链接：多个目录共享一个文件，只要把被共享文件的目录项指向同一个索引节点，即存放相同的索引节点号，被共享的文件可以同名，也可以不同名。
- 只适合文件共享



5

5.5.1 静态共享

2. 符号链接共享

- ◆ 符号链接：只有原文件的目录项才拥有指向其索引结点的指针，而共享该文件的其他链接文件**只有该文件的路径名，并不拥有指向其索引结点的指针，这个保存的路径名也称作符号，所以叫符号链接。**
- ◆ 不但可以用于文件共享，也可用于目录共享
- ◆ 符号链接本身是一种只有文件名，不指向inode的文件，是一种link类型的特殊文件。

5

5.5.1 静态共享

2.符号链接共享

举例：

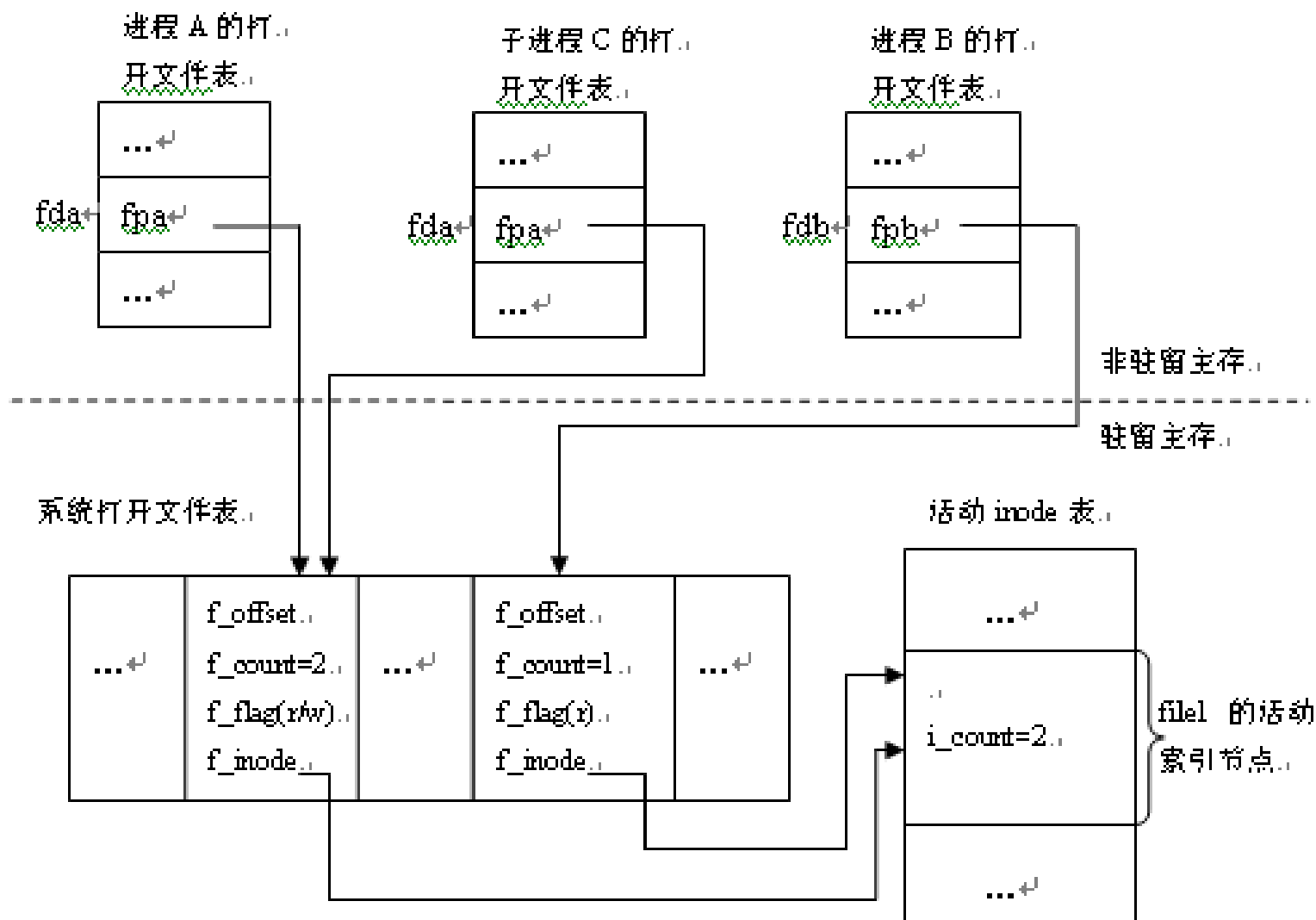
- ◆ 用户A 为了共享用户B 的一个文件F，可以由系统创建一个link 类型的新文件，把新文件写入用户A 的用户目录中，以实现A 的目录与B 的文件F 的链接。在新文件中只包含被链接文件F 的路径名，而文件的拥有者B的目录才具有指向inode 的指针。新文件中的路径名，则仅被看作是符号链，当A 要访问被链接的文件F 且正要读该link类型的新文件时，被操作系统截获，它将依据新文件中的路径名去读该文件，于是就实现了用户A 对用户B的文件F 的共享。

5

5.5.2 动态共享

- ◆ 动态共享：就系统中不同的用户进程或同一用户的不同进程并发地访问同一文件。这种共享关系只有当用户进程存在时才可能出现，一旦用户的进程消亡，其共享关系也就自动消失。
- ◆ 用户打开文件表、系统打开文件表和内存活动索引节点表是实现动态文件共享的重要数据结构
- ◆ 根据是否共享读写位移指针，动态共享分为两种方式：
共享位移指针的动态共享方式
不共享位移指针的动态共享方式

两种动态文件共享方式





目录

CONTENTS

- 5.1 概述
- 5.2 文件的组织
- 5.3 文件目录
- 5.4 文件系统调用的实现
- 5.5 文件共享
- 5.6 文件系统体系结构
- 5.7 Windows文件系统

5

5.6 文件系统体系结构

◆ 文件系统的体系结构通常采用分层结构，有以下三部分组成：

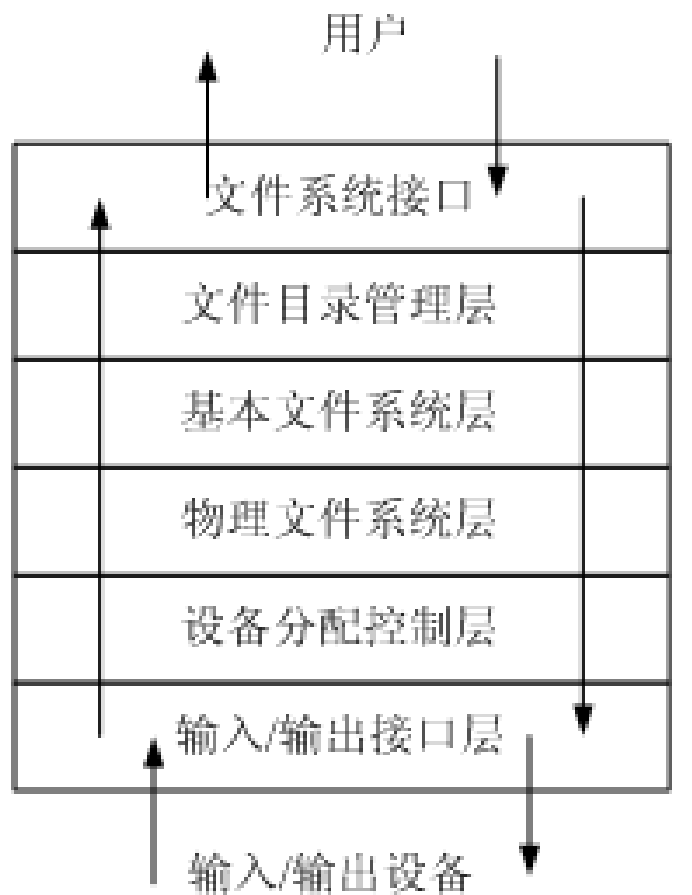
(1) **文件管理层**：实现文件的逻辑结构，为用户提供各种文件系统调用，及文件访问权限的设置等工作；

(2) **目录管理**：负责查找文件描述符，进而找到需要访问的文件，并进行访问权限检查等工作；

(3) **磁盘主存映射管理**：将文件的逻辑地址转换成磁盘的物理地址，即由逻辑块号找到柱面号、磁道号和扇区号，具体的数据传输操作由设备管理实现。

5

5.6.1 文件系统的层次结构模型



5

5.6.1 文件系统的层次结构模型

- ◆ **(1) 文件系统接口层：**主要用于接收用户对文件的操作命令或系统调用，根据用户对文件的存取要求将其转换为统一格式的文件系统内部调用。
- ◆ **(2) 文件目录管理层：**根据文件名或文件路径名建立或搜索文件目录，获得文件内部标志和目录中的文件属性。从目录中的文件属性确定访问文件的用户身份，验证存取权限，判定本次文件操作的合法性，实现文件的存取、共享、保护。如果不允许当前用户访问，则发出操作失败信息。

5

5.6.1 文件系统的层次结构模型

- ◆ **(3) 基本文件系统层：**根据文件内部标志将文件说明信息调入内存，即打开文件，为访问文件做准备。该层根据文件的逻辑结构和存取方法等信息，把指定的逻辑记录地址变换成相应的物理块地址。对于流式文件，只要把用户指定的逻辑地址按块长计算出相对块号；对纪录式文件，先把记录号转换成逻辑地址，再把逻辑地址转换成相对块号。
- ◆ **(4) 物理文件系统层：**根据文件在内存中的物理结构信息，将相对块号和块内地址变换成文件存储器的物理块号和块内地址。

5

5.6.1 文件系统的层次结构模型

- ◆ **(5) 设备分配控制层：**负责文件存储空间的分配，动态地为文件的写操作申请物理块，实现文件缓冲区的管理。该层根据申请的物理块号生成I/O控制系统的地址格式。
- ◆ **(6) 输入/输出接口层：**执行I/O操作，为文件分配磁盘等物理介质空间，实现文件信息的存取，与设备管理功能相联系。

5

5.6.2 文件操作的执行过程

◆ 根据上面文件系统的层次结构，以写一个文件为例简要说明文件写操作发出时，各层是如何工作和相互衔接的。

(1) 当用户写一个文件时，应用程序首先调用文件系统提供的接口，将写文件的请求转换为统一格式的文件系统内部调用。

(2) 文件系统管理根据写文件的文件名和文件路径读内存中相应的目录，修改并更新文件目录。

(3) 基本文件系统根据文件内部标志将文件说明信息放入内存，写入内存中的打开文件表，打开文件，为访问文件做准备。

5

5.6.2 文件操作的执行过程

(4) 物理文件系统根据写文件的结构和存取方法等逻辑结构信息，把指定的逻辑记录地址变换成相应的物理块地址。对于流式文件，只要把用户指定的逻辑地址按块长计算出相对块号；对记录式文件，先把记录号变换成逻辑地址，再把逻辑地址按块长计算出相对块号。

(5) 物理文件系统将相对块号和块内地址变换为文件存储器的物理块号和块内地址。

(6) 设备分配控制为文件的写操作申请物理块，实现文件缓冲区的管理。系统根据申请的物理块号生成I/O控制系统的地址格式。

(7) 输入/输出接口执行I/O操作，为文件分配磁盘等物理介质空间，并将对磁盘的请求信息传递给磁盘管理。

5

5.6.3 虚拟文件系统

◆ 为了同时支持多种文件系统，不同的操作系统采用不同的技术方案提供了虚拟文件系统。

◆ 目标：

(1) 把多种文件系统纳入统一框架，不同的磁盘分区可包含不同的文件系统，对它们的使用和传统的单一文件系统并无区别；

(2) 用户可通过一组系统调用对不同的文件系统及文件进行操作，系统调用可以跨物理介质和跨文件系统执行，如从一个文件系统复制或移动数据到另一个文件系统中，即提供对不同文件系统透明的相互访问；

(3) 对网络文件提供完全的支持，访问远程节点上的文件应与访问本地节点的文件一致；

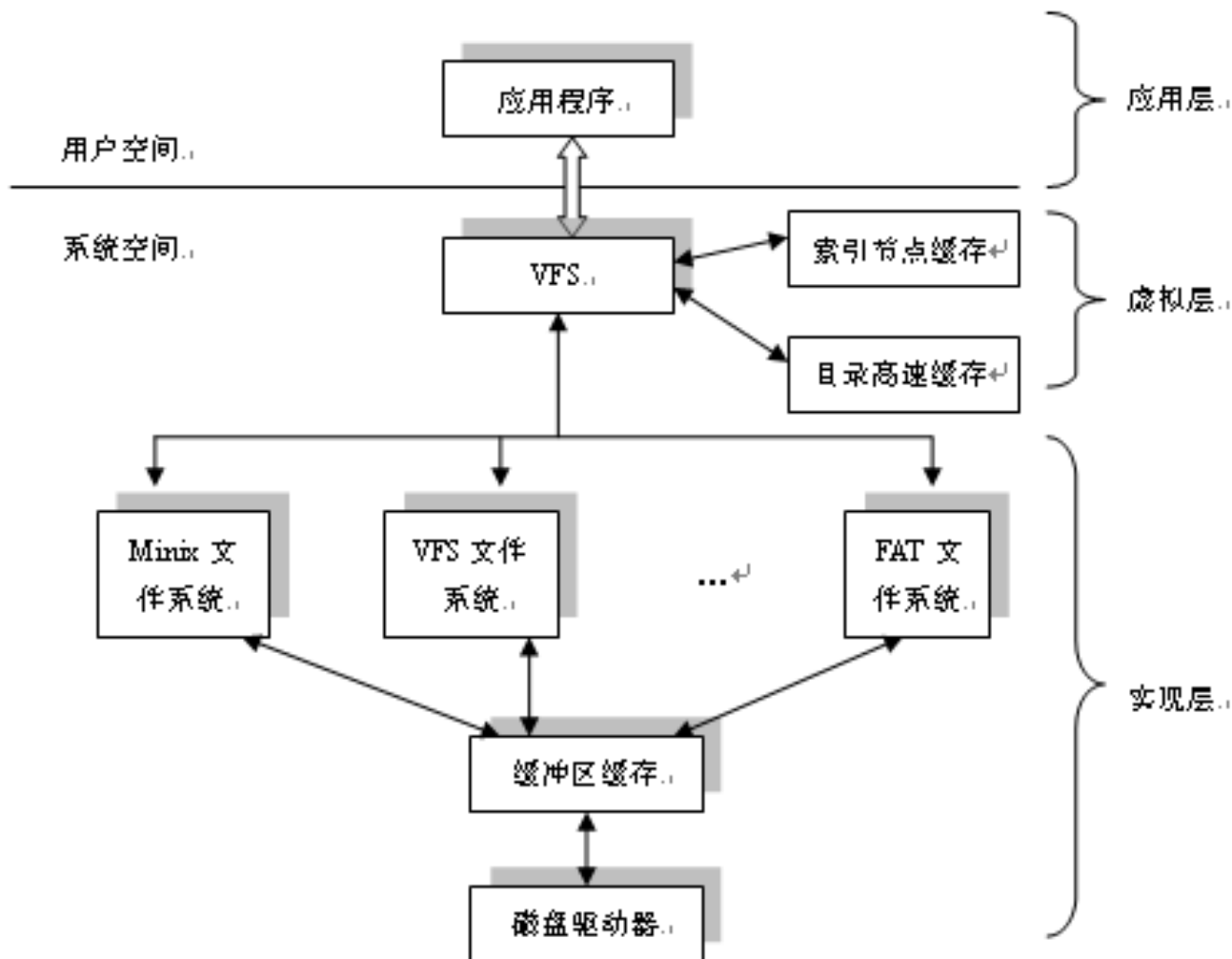
(4) 提供对特殊文件系统的支持。

5

5.6.3 虚拟文件系统

- ◆ 虚拟文件系统也称虚拟文件系统系统开关（Virtual File System, VFS），它是内核的一个子系统，提供一个通用文件系统模型，概括所能见到的文件系统的常用功能和行为，处理一切与底层设备管理相关的细节，为应用程序提供标准接口。具体设计时在原有的具体文件系统层次结构上增加应用层、虚拟层和实现层。各层的功能如下：
- ◆ （1）应用层：VFS模型源于UNIX文件系统，使得用户可以直接使用标准UNIX文件系统调用来操作文件，无须考虑具体文件系统的特性和物理存储介质，通过VFS访问文件系统，才使得不同文件系统之间的协作性和通用性成为可能。
- ◆ （2）虚拟层：在对具体文件系统的共同特性进行抽象的基础上，形成与具体文件系统的实现无关的虚拟层，并在其上定义与用户的一致性接口。
- ◆ （3）实现层：使用类似于开关表的技术进行具体文件系统的转接，实现各种文件系统的细节。

Linux中Ext2的虚拟文件系统结构





THANKS

@CUMTIS