Файлови Системи

Виктор Кетипов
Николай Димитров
Христо Стефанов
elsys.os.2014@gmail.com

[1]Технологическо училище "Електронни системи"
Технически университет, София

18 май 2015 г.

Съдържание

Въведение

- Освен енергозависимата памет, често се използва и енергонезависима памет
- При енергонезависимата памет съществуват подобни проблеми както при енергозависимата памет:
  - Производителност
  - Защита на паметта
  - Фрагментация

Въведение

- Една енергонезависима памет може да се разглежда като последователност от байтове
- Често те се групират в блокове (твърди дискове, SSD)
- При всяко записване или четене от тази памет трябва да укажем къде в паметта искаме да пишем/четем.
- Трябва да следим дали не пишем върху памет, която вече е била използвана за други данни

Файлове

- За улеснение на потребителите операционната система групира тези последователности от байтове във логическата единица *файл*
- Това ни позволява да дадем име на тази последователност от байтове
- Когато операционната система знае кои байтове на кой файл принадлежат, тя може да гарантира, че един файл не пише върху данните на друг

Файлова система

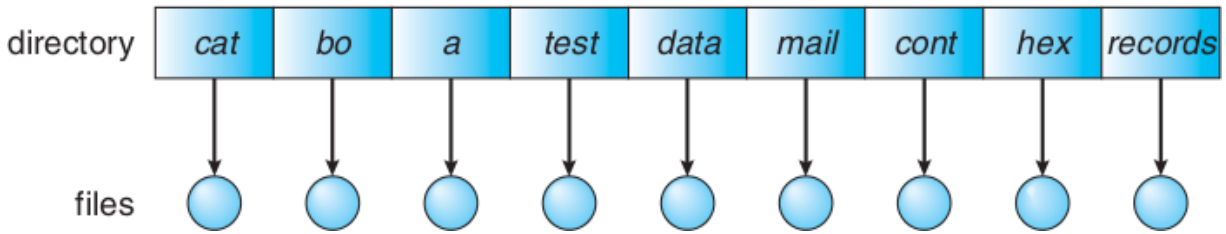
- Файловата система се грижи да организира файловете
- Тя следи къде има свободно място върху паметта и коя част от паметта на кой файл принадлежи

Директории

- В повечето файлови системи е възможно организирането на група от файлове в директория
- Това ни позволява да държим заедно файловете, които имат логическа връзка помежду си

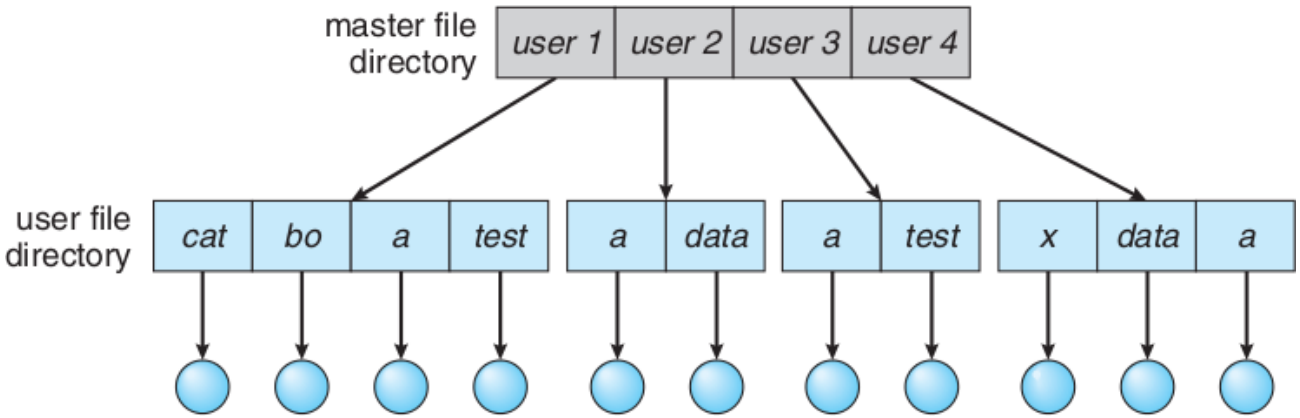
Плоска файлова система

- Не съдържа поддиректории.
- Всички файлове се намират в една обща директория, наречена заглавна директория.



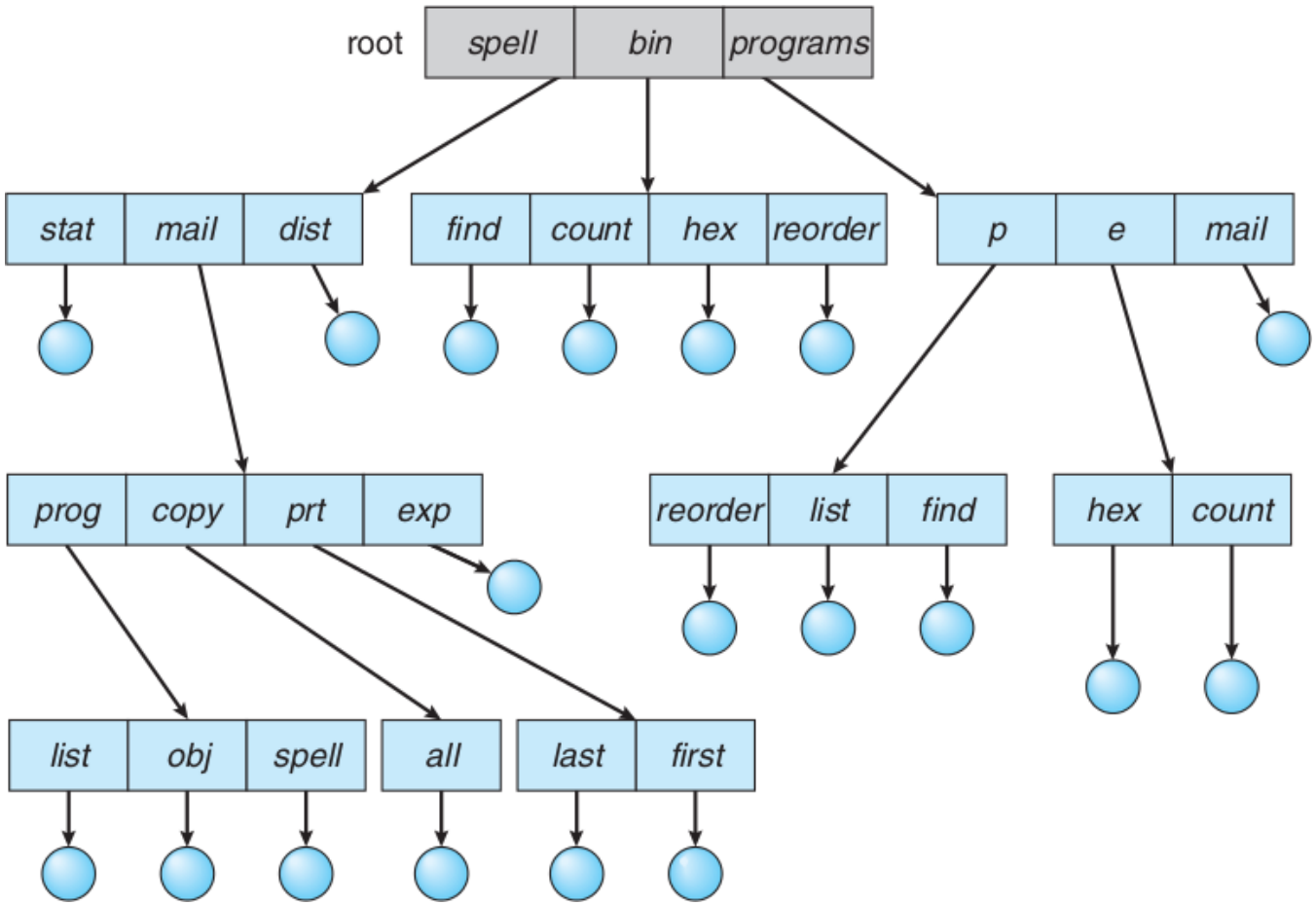
Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

Йерархична структура с фиксиран брой нива



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

Йерархична структура с произволен брой нива



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

Пътища

- За да намерим даден файл трябва да знаем къде в йерархията се намира
- Идентификаторът указващ уникалното местоположение на файл или директория във файловата система се нарича *път*
- Примери за *абсолютен път*:
  - /var/log/apache2/access.log
  - C:\Data\Movies\movie.mkv
- Пътищата може и да са *относителни(релативни)* спрямо текущата директория. Пример:
  - ../../log/apache2/access.log
  - a.txt

Права

- Модерните операционни системи поддържат множество потребители
- Всеки потребител има файлове, които трябва да бъдат достъпни само за него (например: пароли)
- Съществуват файлове, които трябва да са достъпни за множество от потребители или за всички
- Операционната система трябва да има възможност да ограничава достъпа до дадени файлове

Потребители и групи

- За да се улесни управлението на правата, потребителите се разделят в *групи*
- Всеки потребител може да членува в няколко групи

Unix права

- В Unix-базираните операционни системи всеки файл има собственик и асоциирана група
- Не е задължително собственикът да е член на асоциираната група
- Собственикът може да променя правата върху дадения файл

Unix права

- За всеки файл има три вида права:
  - Права на собственика
  - Права на групата
  - Права на всички останали
- Ако даден потребител е собственик на файла, то се гледат правата за собственик
- Ако даден потребител е член на групата на файла, то се гледат правата за групата
- Ако нито едно от двете не е изпълнено, то се взимат предвид правилата за всички останали

Unix права

- Правата биват:
  - Права за четене
  - Права за писане
  - Права за изпълнение
- Дори да сме собственици на даден файл, не е задължително да имаме права върху него. Разбира се, имаме право да ги променим.

Unix права

```
nikidimi@spider:~/tmp$ ls -la
total 48
drwxrwxr-x   3 nikidimi nikidimi  4096 апр 23 21:46 .
drwxr-xr-x 113 nikidimi nikidimi 12288 апр 23 21:45 ..
-rwxrwxr-x   1 nikidimi nikidimi  8374 апр 23 21:45 a.out
-r--------   1 nikidimi nikidimi    14 апр 23 21:46 a.txt
-r--r-----   1 hristo   hristo      14 апр 23 21:46 b.txt
-rw-rw-r--   1 nikidimi nikidimi   525 апр 15 14:18 down.html
-rw-rw-r--   1 nikidimi nikidimi    14 апр 23 21:45 main.c
drwxrwxr-x   2 nikidimi nikidimi  4096 апр 23 21:46 test
```

Физическа организация на файлова система

- Ефективност - Файловата система трябва да осигурява бърз достъп до данните и ефективно използване на дисковата памет.
- Надеждност и сигурност - Файловата система трябва да е устойчива в условия на конкурентен достъп от много потребители, при възможни срирове и да е защитена от неправомерен достъп.
- Мащабируемост - Файловата система трябва да може да работи както на устройства с малък капацитет (16MB), така и на такива с много голям капацитет (1PB = 1000TB)

Физическа организация на файлова система

- Подобно на страницирането на оперативната памет, енергонезависимата памет също се разделя на малки фиксирани части (блокове/сектори)
- При записването на файл, се използват един или повече от тези блокове

Служебна информация за даден файл

- За да достъпим даден файл трябва да знаем върху кои блокове е разположен.
- Освен това къде се намира физически на диска, е необходимо да пазим и друга служебна информация за него като:
  - тип
  - права за достъп
  - дата и час на създаване
  - големина
- Тази информация най-често се пази на специално място на диска.
- В UNIX-базираните операционни системи тази служебна информация се нарича inode.

Директории

- Директориите в повечето случаи се пазят като специален файл
- Всяка директорията трябва да пази информация за всички файлове, които се намират в нея.
- Директорията може да се разглежда като таблица със записи.
- За всеки файл/директория има запис, съдържащ името на файла, както и указател към блока, съдържащ служебната информация (inode-а) за този файл.
- Служебната информация може да се намира и директно в записа за даден файл/директория, например при файловата система FAT.

Достъпване на файл

- На специално място на диска е записана информация за главната (root) директория на файловата система.
- Използвайки информацията от нея, ние можем да стигнем до директорията, в която се намира файлът, който искаме да достъпим.
- Използвайки информацията от тази директория, можем да намерим физическото местоположение на данните на файла.

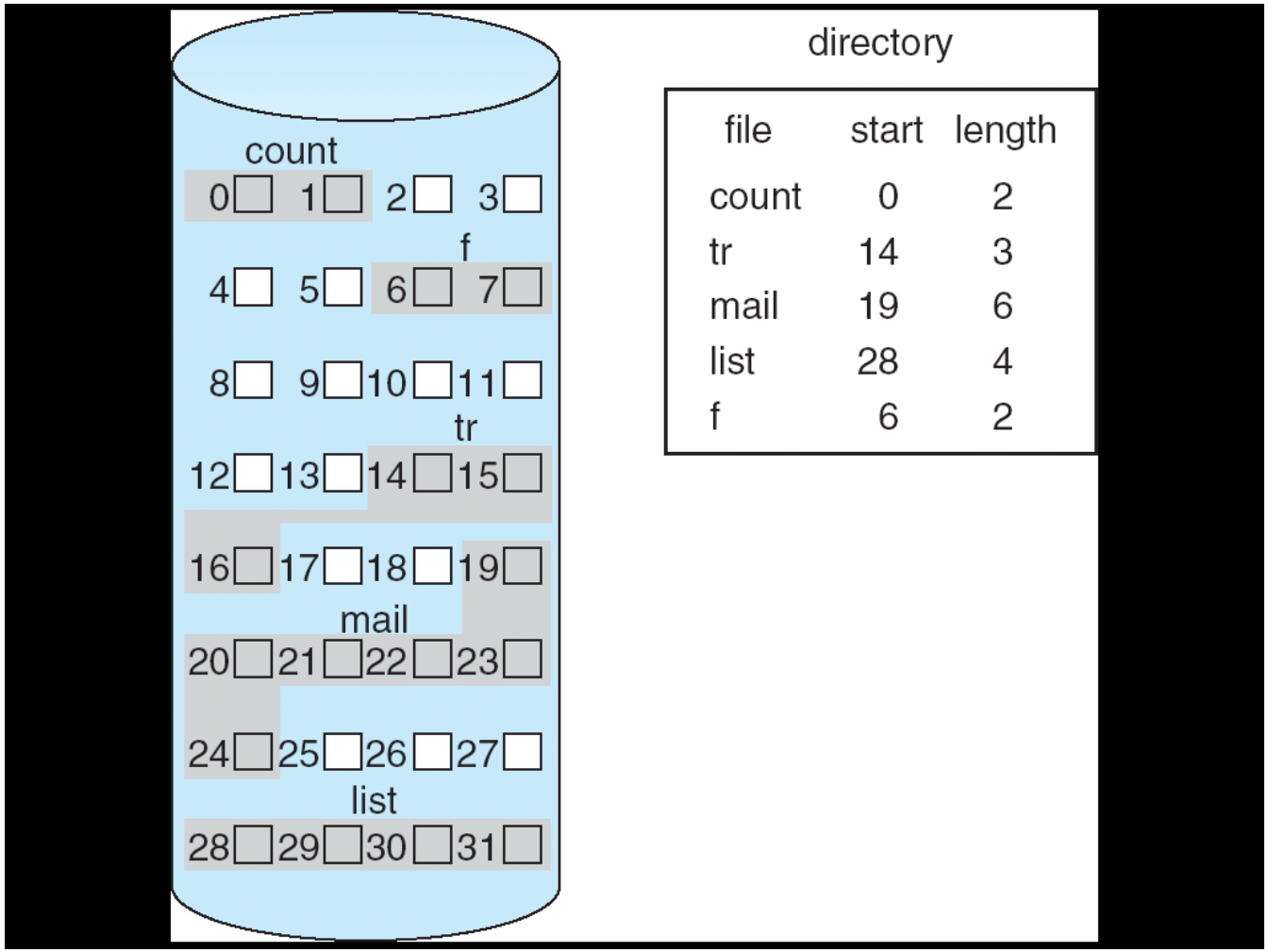
Физическа организация на файлова система

- Има различни начини за разпределяне на блокове за даден файл:
  - Последователно разпределение
  - Разпределение чрез свързан списък
  - Разпределение чрез индексна таблица

Последователно разпределение

- Най-простия начин за разпределение
- Всеки файл се записва в последователни блокове
- Необходимо е да се помни само от кой блок започва даден файл и колко на брой блокове заема

Последователно разпределение



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

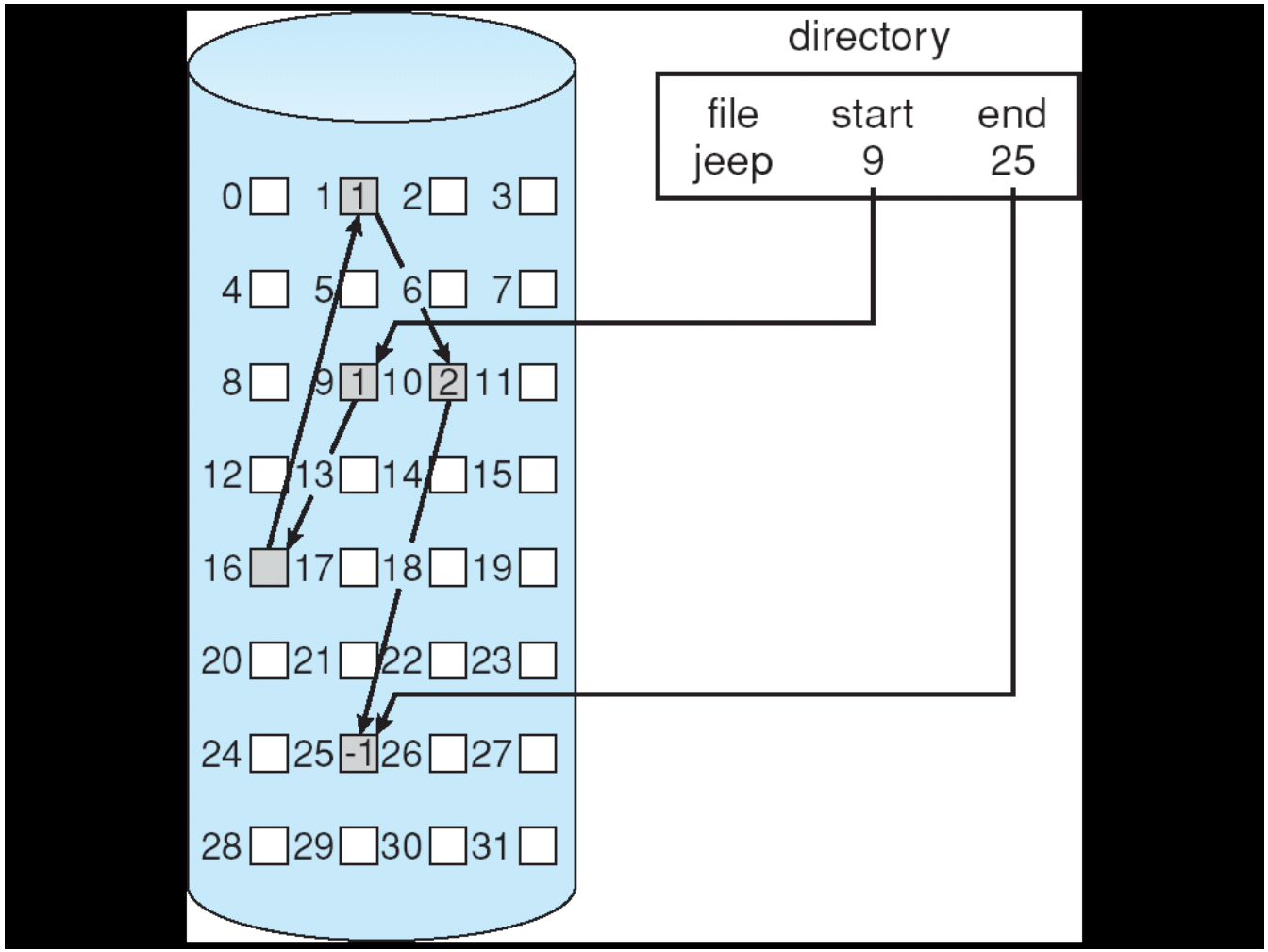
Предимства и недостатъци

- Бърз достъп до произволна част от файла
- Има основни недостатъци:
  - Файловете не могат да нарастват, ако блоковете след даден файл са заети
  - Външна фрагментация - не можем да запазим файл с големина 10 блока, ако например имаме 2 участъка по 5 свободни блока

Разпределение чрез свързан списък

- Всеки файл е разпределен в множество блокове, пръснати из целия диск
- Всеки блок пази къде се намира следващия блок от файла, като така образуват свързан списък

Разпределение чрез свързан списък



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

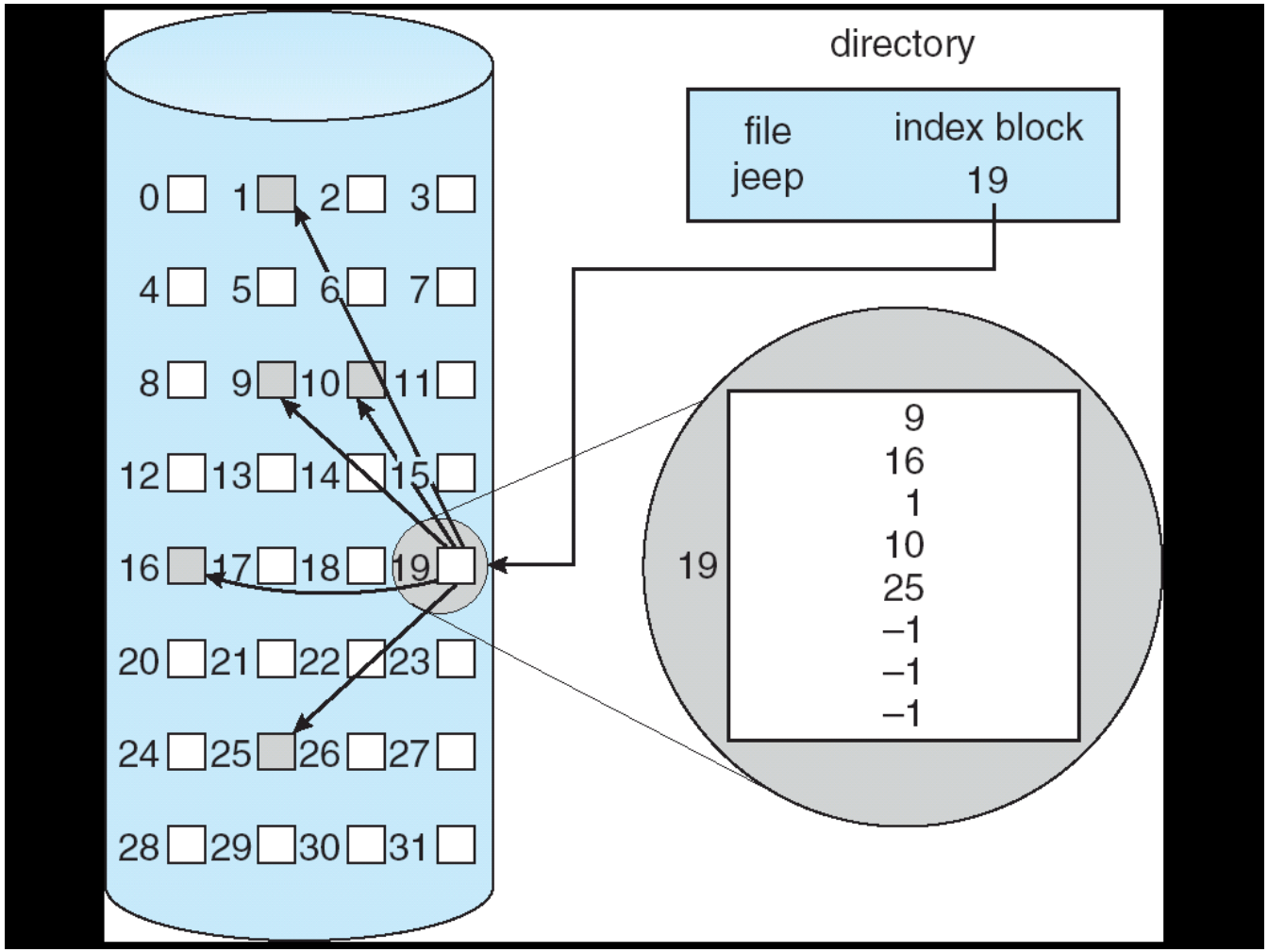
Предимства и недостатъци

- Просто за реализация
- Необходим е само началния адрес за достъпване на файл
- Не страда от външна фрагментация и всички свободни блокове могат да бъдат използвани
- Бавен произовлен достъп - трябва да се обходят всички блокове за да се стигне до желания (подобно на търсене в свързан списък)

Разпределение чрез индексна таблица

- Всички блокове на даден файл се записват в индексна таблица
- Тази таблица се записва в някой свободен блок на диска

Разпределение чрез индексна таблица

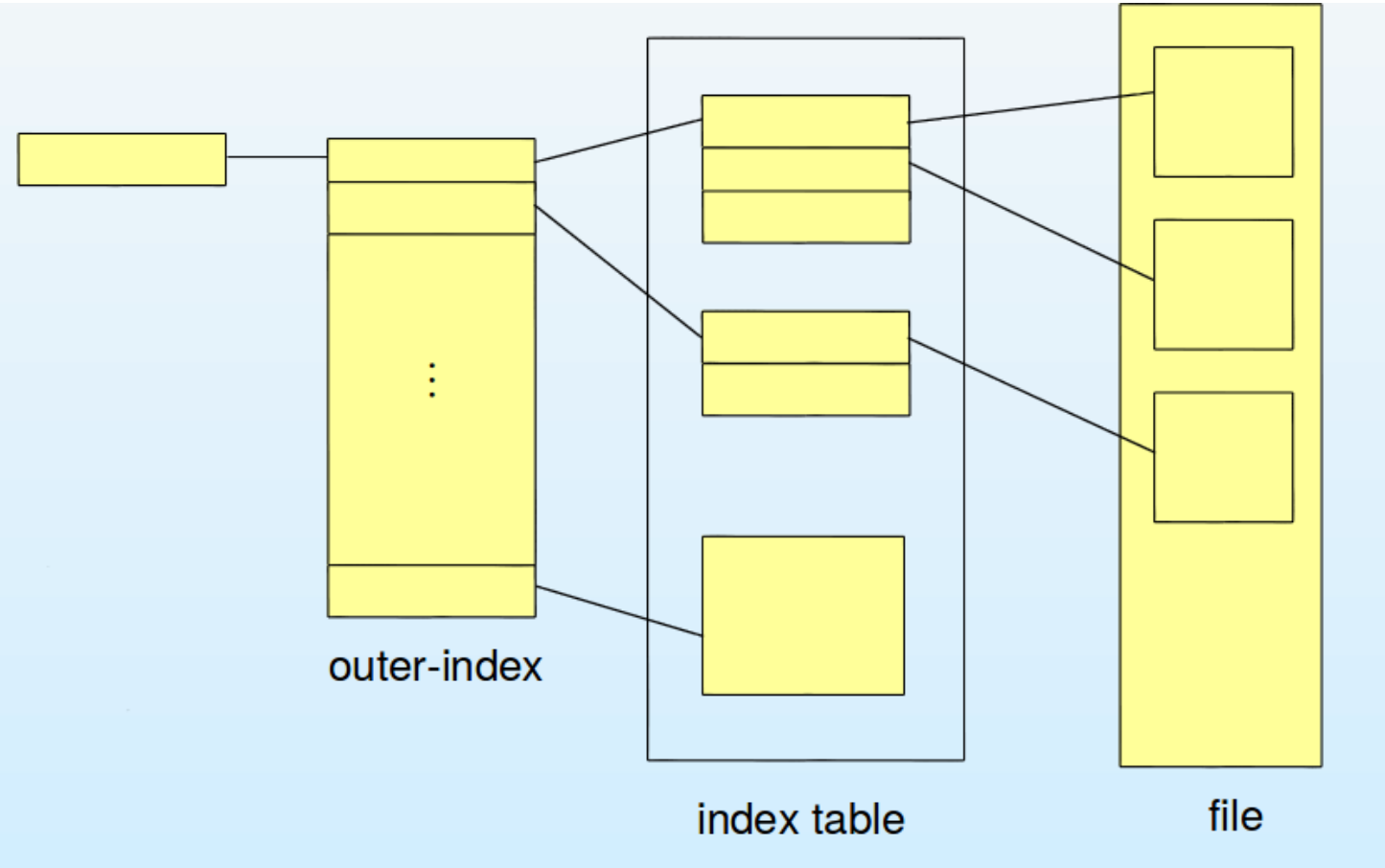


Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

Предимства и недостатъци

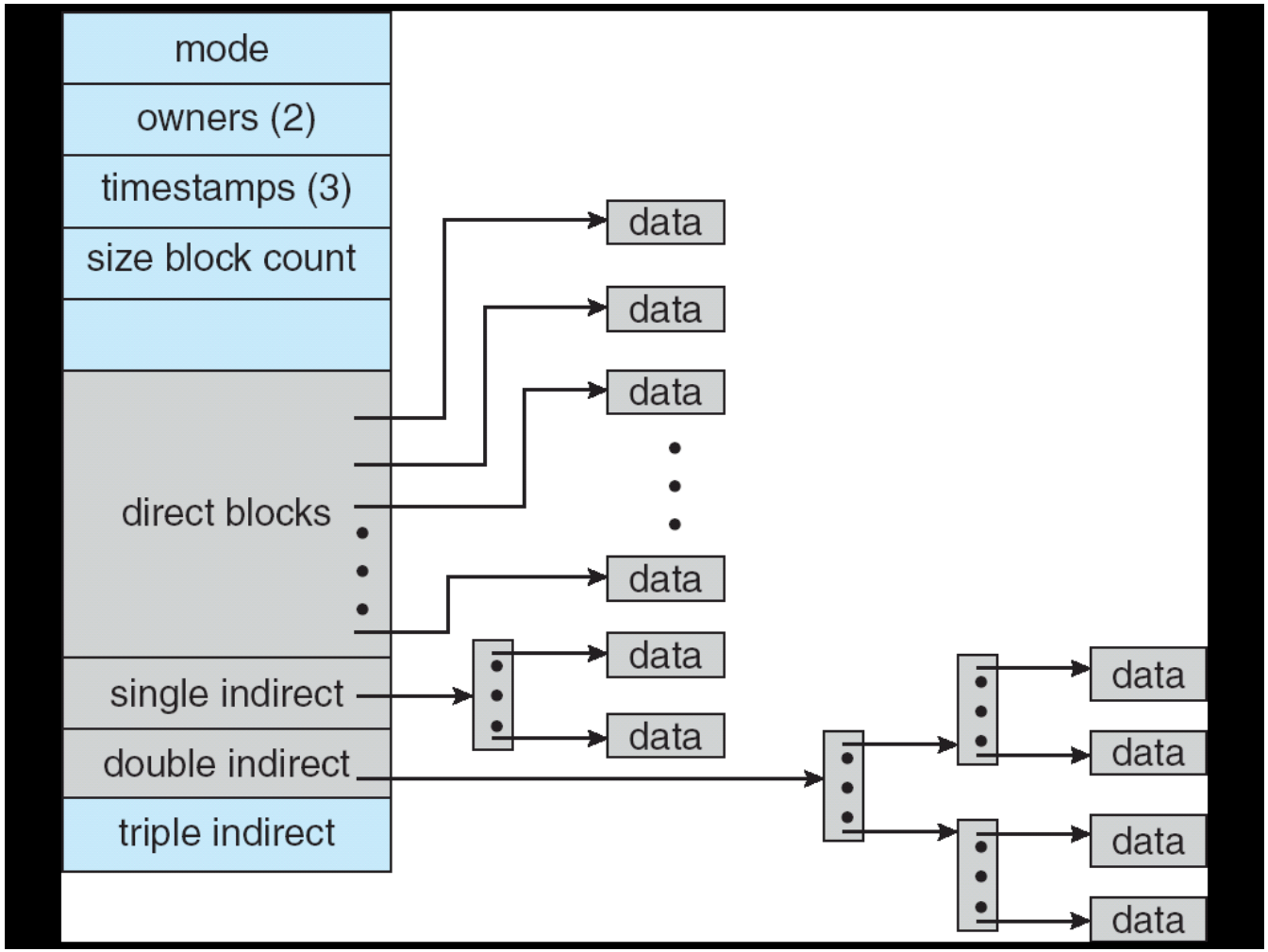
- Бърз произволен достъп без външна фрагментация
- Необходим е допълнителен блок за индексната таблица
- Големината на файла е ограничена от големината на блока, който пази индексната таблица
  - Това може да бъде избегнато като се добавят вторични индексни таблици. Така всеки запис в първата индексна таблица сочи към друга индексна таблица, а всеки запис в нея сочи към блок от файла.
  - При нужда могат да бъдат добавяни и трети, четвърти и т.н. индексни таблици

Множество индексни таблици



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

Комбиниран подход



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

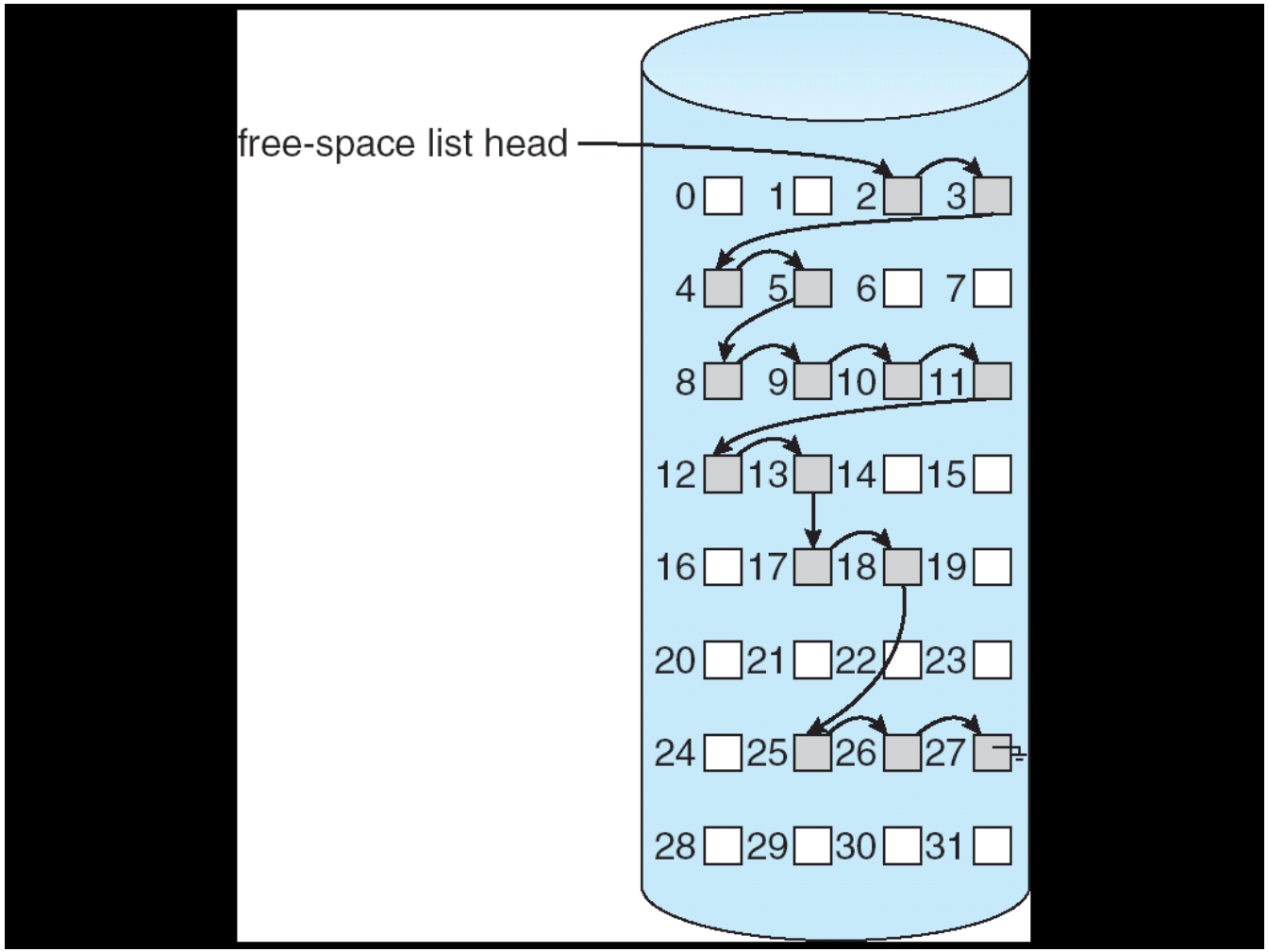
Управление на свободното място

- Съществуват два основни подхода:
  - Свързан списък от свободни блокове
  - Побитова карта (bitmap) на свободните блокове

Свързан списък от свободни блокове

- Подобно на разпределение чрез свързан списък
- Всеки свободен блок пази указател към следващия свободен блок
- Необходимо е да знаем само адреса на първия свободен блок

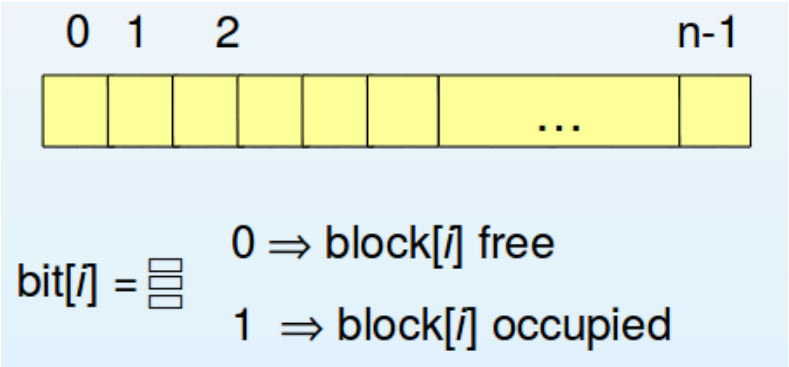
Свързан списък от свободни блокове



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

Побитова карта (bitmap) на свободните блокове

- Използва се побитова карта, в която за всеки блок от диска има заделен един бит
- Стойността на този бит определя дали блокът е зает.
- Необходимо е място, където да се запише тази побитова карта



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*

FAT

- Паметта се разделя на парчета с фиксиран размер, наречени клъстъри (cluster). Това е най-малката единица, която може да се задели за файл или директория
- Използва се разпределение чрез свързан списък
  - Един файл може да се съдържа в един или повече клъстери
  - Всеки клъстър съдържа указател към следващия клъстър от файла или специален указател за край

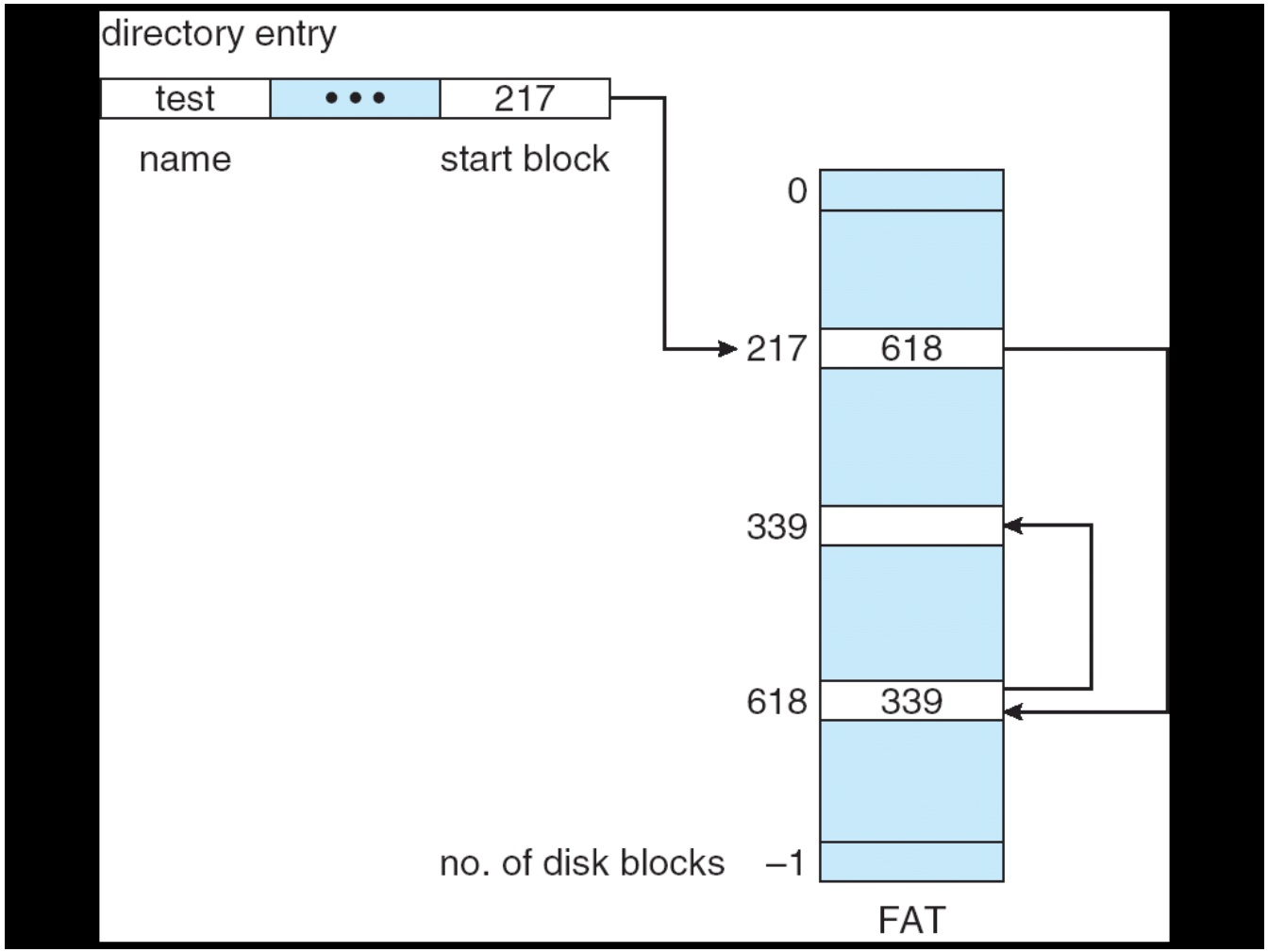
File Allocation Table

- За да се определи състоянието на всеки клъстър, се използва специална таблица.
- Тя се нарича File Allocation Table
- Представлява разширение на побитовата карта на свободните блокове
- Всеки клъстър може да бъде:
  - свободен
  - разпределен за файл
  - маркиран като повреден

Директории

- Директориите се записват като файлове в клъстерите
- Тези файлове съдържат таблица, съдържаща запис за всеки файл или поддиректория
- Всеки запис в таблицата съдържа:
  - Име на файла
  - Големина на файла
  - Дата/час на създаване и последна модификация
  - Начален клъстър на файла
  - Права и други атрибути
- Главната директория е записана точно след File Allocation Table.

File Allocation Table



Фигура : Silberschatz, Gavin, Gagne: *Operating Systems Concepts*