Bölüm 10: Dosya Sistemleri

İşletim Sistemleri

Dosya Sistemleri

- Birçok uygulama, sanal adres alanına sığmayacak kadar veri depolar.
- Veriler, sürecin sona ermesinden sonra da kalıcı olmalı.
- Birden çok süreç aynı anda verilere erişebilmeli.

Dosya Sistemi

- Diskteki verilerin düzenlenmesi ve yönetilmesidir.
- İşletim sisteminin kritik bir bileşenidir.
- Tasarlanırken,
  - diskin boyutu,
  - depolanan veri türü,
  - verilere erişen kullanıcı sayısı, gibi
  - sistemin ihtiyaçları dikkate alınır.

Dosya Sistemleri

- Dosyaları saklamak için *disk* kullanılır. Disk *bloklardan* oluşur.
- Dosya sistemi diskten *blok blok* okuma ve yazma yapar.
- Bloklar ile başa çıkabilmek için dosya sistemi bir soyutlama sağlar.
- Dosyalar,
  - Süreçler tarafından oluşturulur,
  - Bir diskte binlerce dosya bulunabilir.
  - Bellek *adres uzayına* benzeyen mantıksal veri *birimleridir*.
- Dosya sistemi dosyaları yönetir.
  - Adlandırma, oluşturma, silme, erişilebilirlik, koruma gibi.

Dosya Sistemleri

- Dosya sistemine iki farklı bakış.
  - Kullanıcı: dosya nasıl adlandırılır, oluşturulur, silinir, korunur?
  - Uygulama: dosya diskte nasıl saklanır? (*organize*)
- Kullanıcı bakış açısıyla
  - Adlandırma (*naming*),
  - Yapı (*structure*),
  - Dizinler (*directories*).

Adlandırma

- Eski Windows sistemlerde *FAT16* ve *FAT32* kullanılmıştır.
- Güncel Windows sistemler *NTFS* dosya sistemi kullanır.
- *New Technology File System (NTFS).*
- *File Allocation Table (FAT).*
- Adın bir parçası olarak sonek (*suffix*) kullanır.
- Sonek (*uzantı*)
  - Unix dosya soneklerinin bir anlam ifade etmesini zorlamazken,
  - DOS sistemde soneklerin bir anlamı vardır.

Sonek Örnekleri

- Belgeler: .doc, .docx, .pdf, .txt, .rtf, .odt
- Görseller: .jpg, .jpeg, .png, .gif, .bmp, .tiff
- Ses: .mp3, .wav, .aiff, .m4a, .wma
- Video: .mp4, .avi, .mov, .wmv, .flv
- E-tablolar: .xls, .xlsx, .csv
- Sunumlar: .ppt, .pptx, .odp
- Web: .html, .css, .js
- Yürütülebilir: .exe, .msi
- Sıkıştırılmış: .zip, .rar, .tar, .gz, .7z

Uzantılar ve Açılımları

doc	Microsoft Word Document
docx	Microsoft Word Open XML Document
pdf	Portable Document Format
txt	Plain Text File
rtf	Rich Text Format
odt	Open Document Text
jpg	Joint Photographic Experts Group Image
jpeg	Joint Photographic Experts Group Image
png	Portable Network Graphics
gif	Graphics Interchange Format

Uzantılar ve Açılımları

bmp	Bitmap Image
tiff	Tagged Image File Format
mp3	MPEG-1 Audio Layer 3
wav	Waveform Audio Format
aiff	Audio Interchange File Format
m4a	MPEG-4 Audio
wma	Windows Media Audio
mp4	MPEG-4 Part 14
avi	Audio Video Interleave
mov	Apple QuickTime Movie

Uzantılar ve Açılımları

wmv	Windows Media Video
flv	Flash Video
xls	Microsoft Excel Spreadsheet
xlsx	Microsoft Excel Open XML Spreadsheet
csv	Comma Separated Values
ppt	Microsoft PowerPoint Presentation
pptx	Microsoft PowerPoint Open XML Presentation
odp	Open Document Presentation
html	HyperText Markup Language
css	Cascading Style Sheets

Uzantılar ve Açılımları

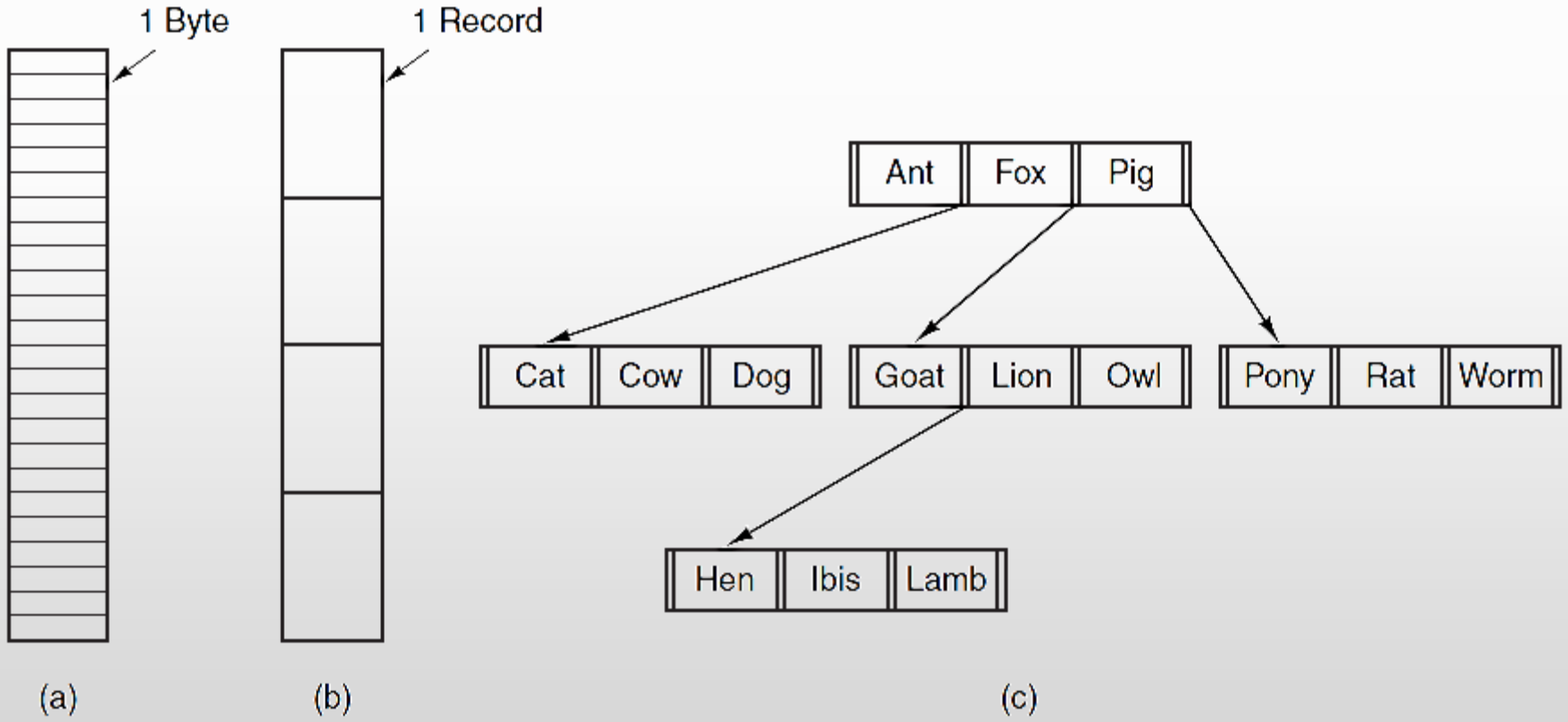
js	JavaScript
exe	Executable File
msi	Windows Installer Package
zip	Zipped File
rar	RAR Archive
tar	Tape Archive
gz	GZIP Compressed Archive
7z	7-Zip Compressed Archive

Dosya Yapısı

- *Maksimum esneklik*: içinde her şey saklanabilir.
- Dosya *bayt* dizilerinden oluşabilir.
- Sabit uzunluklu kayıtlardan (*eskiden kart imajları*) oluşabilir.
- *Kayıt ağacı*: kayıtları bulmak için anahtar (*key field*) kullanılır.

Dosya Yapısı

(a) Bayt dizisi (b) Kayıt dizisi (*sabit boyutlu*) (c) Ağaç (*değişken boyutlu*).



Dosya Tipleri

- **Normal**: Ham veri içerir.
- **Dizinler:** Dosyaların izini tutar.
- **Karakter dosyaları:** seri (*serial*) tabanlı G/Ç cihazları (*yazıcı*).
- **Blok dosyaları:** blok tabanlı modeller (*disk*).

Normal Dosyalar

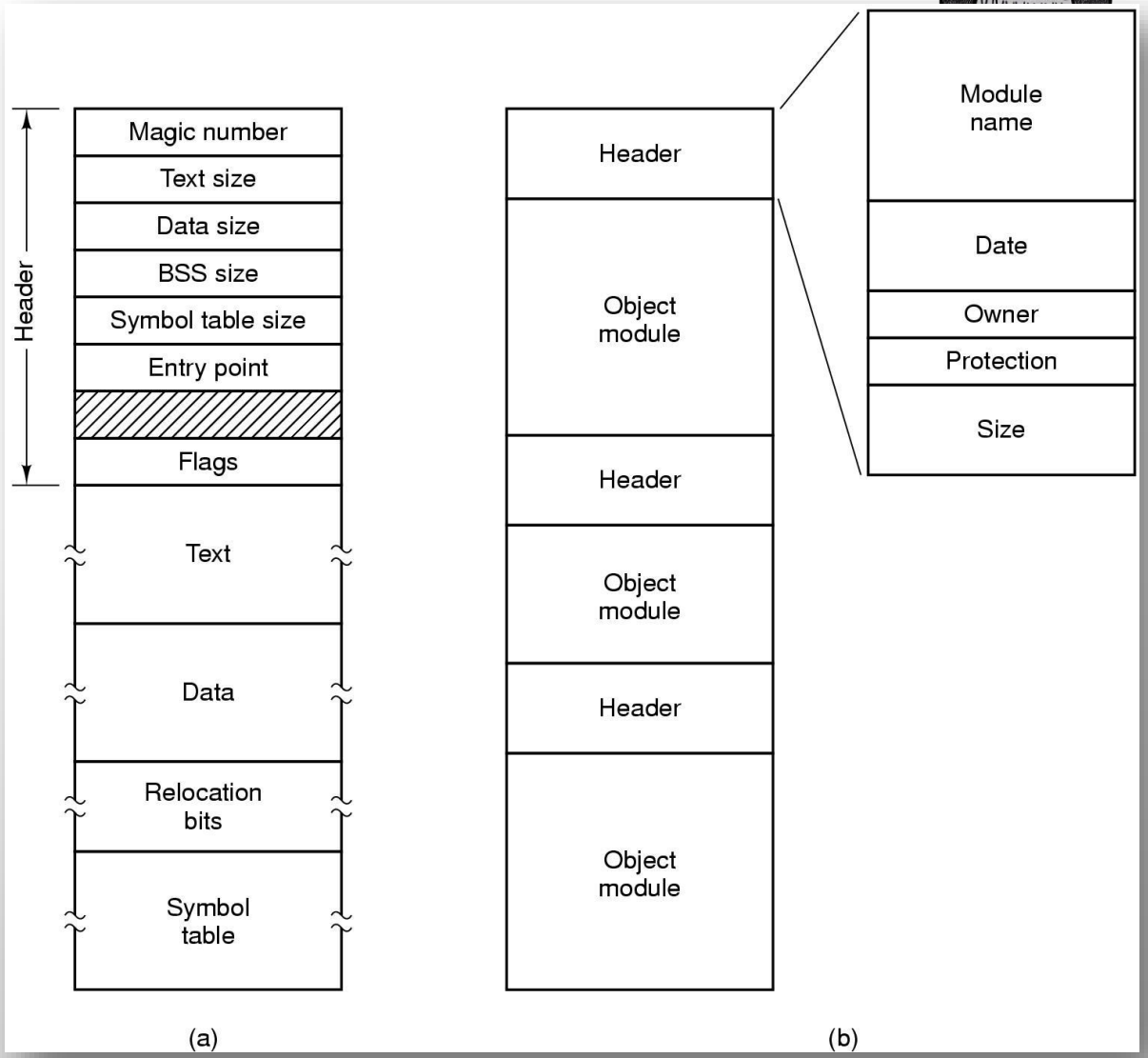
- Metin (*text*) veya ikili (*binary*) formatta olabilir.
- Metin tabanlı dosyalar yazdırılabilir.

İkili Dosya Tipleri

- Yürütülebilir dosyalardır, UNIX'te *magical field* ile tanımlanır.
- Arşiv olarak derlenmiş dosyalar.
  - Bağlanmış (*linked*) kütüphane prosedürleri hariç.
- Her işletim sistemi kendi yürütülebilir dosya formatını tanır.

İkili Dosya Tipleri

- (a) Yürütülebilir dosya (*.exe*)
- (b) Derlenmiş ancak bağlanmamış arşiv kütüphanesi (*.o*)



Dosya Erişimi

- Sıralı erişim: okumaya baştan başlanır, atlama yapılamaz.
  - Manyetik bant, kaset.
- Rastgele erişim: okumaya istenilen yerden başlanır.
  - Diskler.
- Dosya tanımlayıcı (*file identifier*)
  - Çekirdek tarafından yönetilen, bir nesneyi temsil eden tamsayı.
  - Her sürecin, 0'dan başlayan özel bir dosya tanımlayıcı alanı vardır.
  - 0 standart girdiyi, 1 standart çıktıyı ve 2 standart hatayı temsil eder.

Dosya Öznitelikleri

Read-only	Dosya yalnızca okunabilir, değiştirilemez, silinemez.
Hidden	Dosya, dosya gezgininin normal görünümünde görünmez.
System	Dosya işletim sistemi tarafından kullanılır, değiştirilmemeli silinmemelidir.
Archive	Dosya, yedeklendikten sonra değiştirilmiş, sonraki yedeklemeye dahil edilmeli.
Compressed	Dosya, disk alanından tasarruf etmek için sıkıştırılmıştır.
Encrypted	Yetkisiz erişimi önlemek için dosya şifrelenmiştir.
Temporary	Dosya geçici olarak kullanılır ve ihtiyaç kalmadığında otomatik olarak silinir.
Executable	Dosya çalıştırılabilen bir programdır.
Indexed	Dosya, daha hızlı arama için indeksleme hizmetine dahil edilmiştir.
Offline	Dosya anlık kullanım için değildir, çevrimdışı kullanım için hazır hale getirilebilir.
Not content indexed	Dosya, aramayı hızlandırmak için indeksleme hizmetinden çıkarılır.
Reparse point	Dosya, başka bir dosyaya veya dizine bir bağlantı veya referans içerir.

Dosya Öznitelikleri

Sparse file	Dosya, diskte depolanmayan büyük sıfır blokları içeren bir dosya türüdür.
Symlink	Dosya, başka bir dosya veya dizine sembolik bir bağlantıdır.
Device	Dosya, işletim sistemi tarafından kullanılan özel bir aygıt dosyasıdır.
Junction point	Dosya, başka bir birimdeki bir dizine işaret eden bir tür sembolik bağlantıdır.
Creation time	Dosyanın oluşturulduğu tarih ve saat.
Last Access	Dosyaya en son erişilen tarih ve saat.
Last change	Dosyanın en son değiştirildiği tarih ve saat.
Current size	Dosyadaki bayt sayısı.
Max size	Dosyanın büyüyebileceği bayt sayısı.
Owner	Dosyayı sahibinin kimliği.
Creator	Dosyayı oluşturan kişinin kimliği.

Stat Veri Yapısı

```
struct stat {
      mode_t      st_mode;    // file type and mode (permission)
      ino_t       st_ino;     // inode number
      dev_t       st_dev;     // device number
      dev_t       st_rdev;    // device number (special)
      nlink_t     st_nlink;   // number of links
      uid_t       st_uid;     // user ID of owner
      gid_t       st_gid;     // group ID of owner
      off_t       st_size;    // size in bytes
      time_t      st_atime;   // time of last access
}
```

Dosya Öznitelikleri

- drwxr-xr-x 2 root root 4096 Sep 24 2008 Unit2
- drwxr-xr-x 2 root root 4096 May 26 19:21 a
- -rwxr-xr-x 1 root root 10930 Aug 5 22:49 a.out
- -rwxrwx--T 1 root root 81 Aug 2 2008 a.txt
- -rwxr-x--- 1 root root 81 May 26 19:20 b.txt
- -rwx------ 1 root root 81 Jul 30 19:28 c.txt
- -rwxr-xr-x 1 root root 11193 Jul 30 19:27 cp

Dosya Öznitelikleri

- `- rwx  rw-  r--`

- Dosya türü: - bir dosya, *d* dizin anlamına gelir.
- Dosyanın sahibi için okuma, yazma ve yürütme izinleri (*rwx*)
- Dosyanın sahibi olan grup için okuma, yazma ve yürütme izinleri (*rw-*)
- Diğer tüm kullanıcılar için okuma, yazma ve yürütme izinleri (*r--*)

Dosyalar için Sistem Çağrıları

- **Create**: veri olmadan boş dosya oluşturur, öznitelikleri atar.
- **Delete**: disk alanını boşaltmak için dosyayı serbest bırakır.
- **Open**: dosyanın öznitelikleri ve diskteki adresleri belleğe yüklenir.
- **Close**: öznitelikler ve disk adreslerin bulunduğu tablo boşaltılır.
- **Read**: dosya işaretçisinin konumundan belleğe okuma yapılır.
- **Write**: işaretçinin geçerli konumuna bellekten yazma yapılır.

Dosyalar için Sistem Çağrıları

- **Append**: dosyanın sonuna ekleme yapar.
- **Seek**: dosya işaretçisini dosyada belirli bir yere koyar.
- **Rename**: dosya adı güncellenir.
- **GetProperties**: dosyanın öznitelikleri okunur.
- **SetProperties**: dosyanın öznitelikleri güncellenir.

Dosya Kopyalama – copy abc xyz

- *abc* dosyasını *xyz* dosyası olarak kopyalar.
  - Eğer *xyz* adında bir dosya varsa üzerine yazılır.
  - Yok ise boş bir dosya oluşturulur.
- Sistem çağrıları kullanılır. (*okuma, yazma*)
- 4 *KB* boyutunda parçalar halinde okur ve yazar.
  - *abc* dosyasından tampon belleğe okunur. (*read*)
  - Tampon bellekten *xyz* dosyasına yazılır. (*write*)

Dosya Kopyalama – copy abc xyz

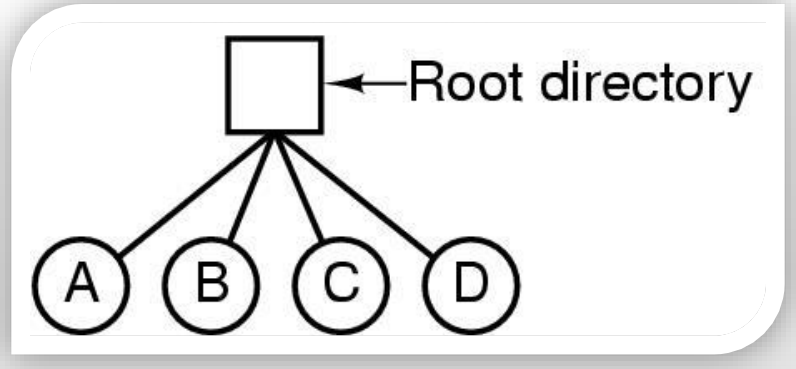
```
#define BUFFER_SIZE 4096  // 4 KB

void copyFile(char *src, char *dest) {
  char buffer[BUFFER_SIZE];
  source = open(src, O_RDONLY);
  destination = open(dest, O_WRONLY | O_CREAT | O_TRUNC);
  // 4 KB'lık parçalar halinde oku ve yaz
  while ((bytesRead = read(source, buffer, BUFFER_SIZE)) > 0) {
    bytesWritten = write(destination, buffer, bytesRead);
  }
  close(source);
  close(destination);
}
```

Dizinler

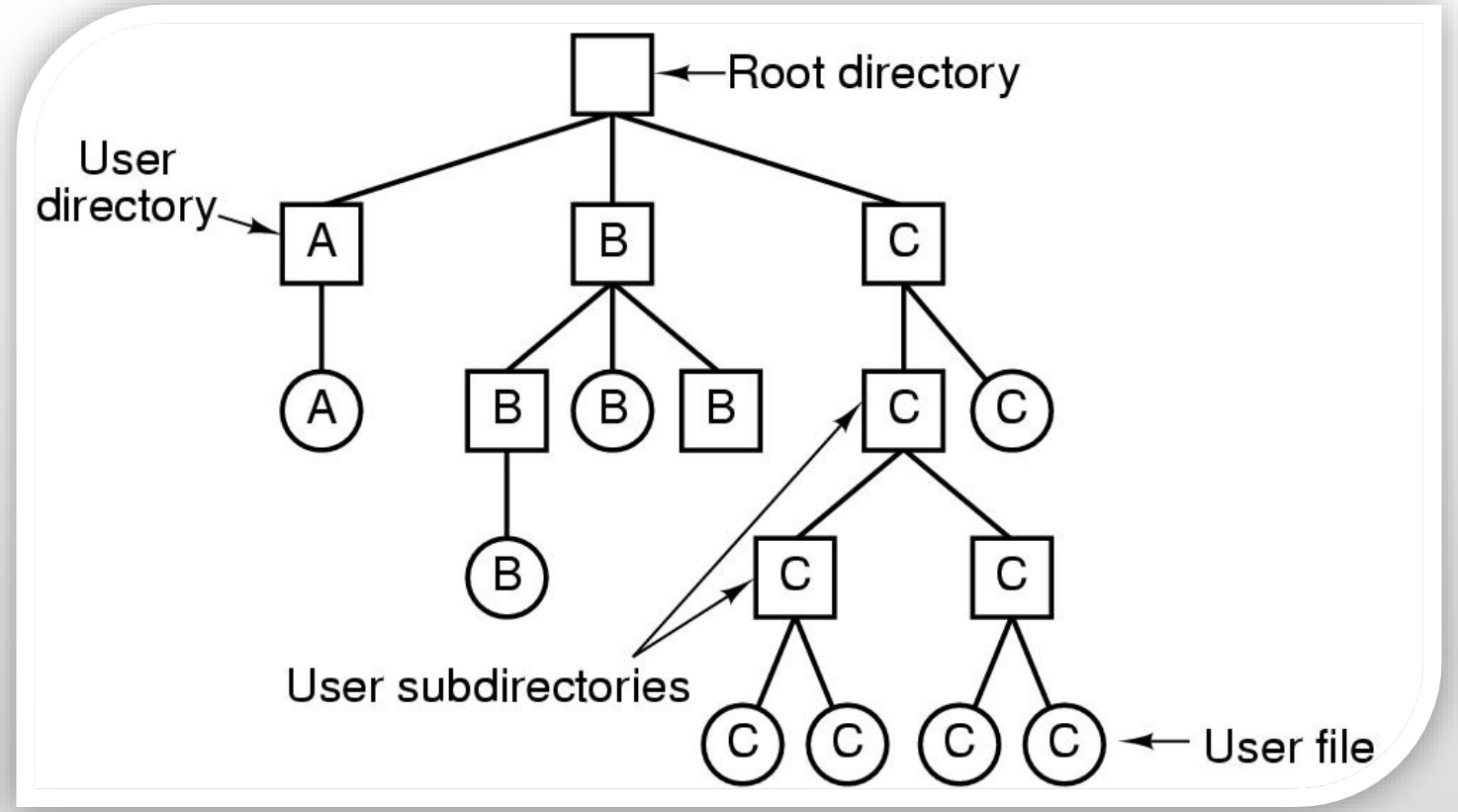
- Bir dosya kümesini düzenleyen dosyalar.
- Klasör (*folder*) olarak da adlandırılır.
- Katı bağlama (*hard link*)
  - Bir dosyanın birden fazla dizinde görünmesini sağlar.
- Sembolik bağlama (*symbolic link*)
  - Başka bir dosyaya işaret (*point*) eden bir bağ (*link*) oluşturur.

Dört Dosya İçeren Tek Düzeyli Dizin

- Tek düzeyli dizinlerde
  - isimlendirme ve
  - gruplandırma sorunları yaşanır.



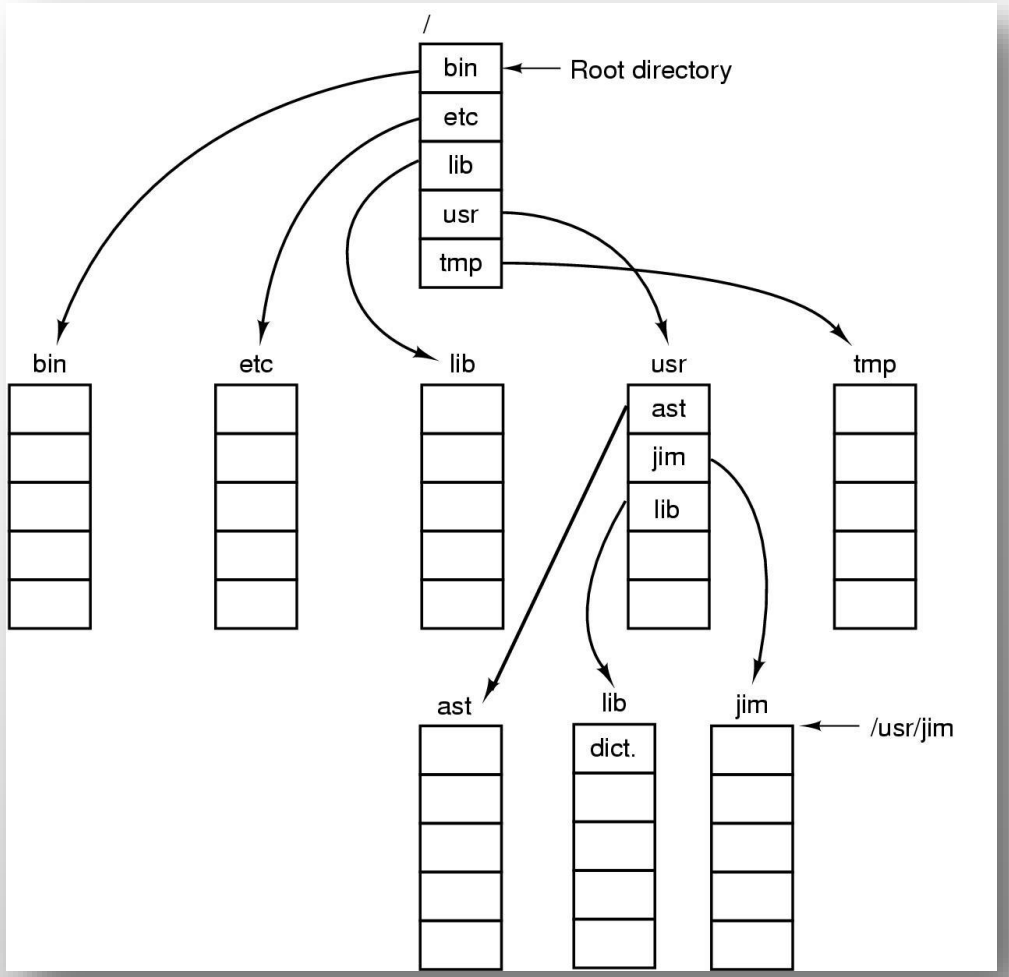
Hiyerarşik Dizin Sistemleri



Yol (Path) Adları

- Mutlak adres (*absolute*)
  - /usr/sercan/os/slaytlar
- Bağıl adres (relative)
  - ../os/slaytlar
- . içinde bulunulan (*working*) dizini ifade eder.
- .. bir üst dizini ifade eder.

UNIX Dizin Ağacı



Dizin İşlemleri

- **`Opendir`**, bir işlem yapılmadan önce, dizin akışı (stream) açılır.
- **`Create`**, dizin oluşturur.
- **`Readdir`**, açılan dizindeki bir sonraki elemanı döndürür.
- **`Delete`**, dizini siler, silinecek dizin boş olmalıdır.
- **`Rename`**, dizini yeniden adlandırır.
- **`Closedir`**, işlemler bittikten sonra dizin akışı kapatılır.
- **`Link`**, dizini başka bir dizine bağlar.
- **`Unlink`**, bağlantıyı kaldırır.

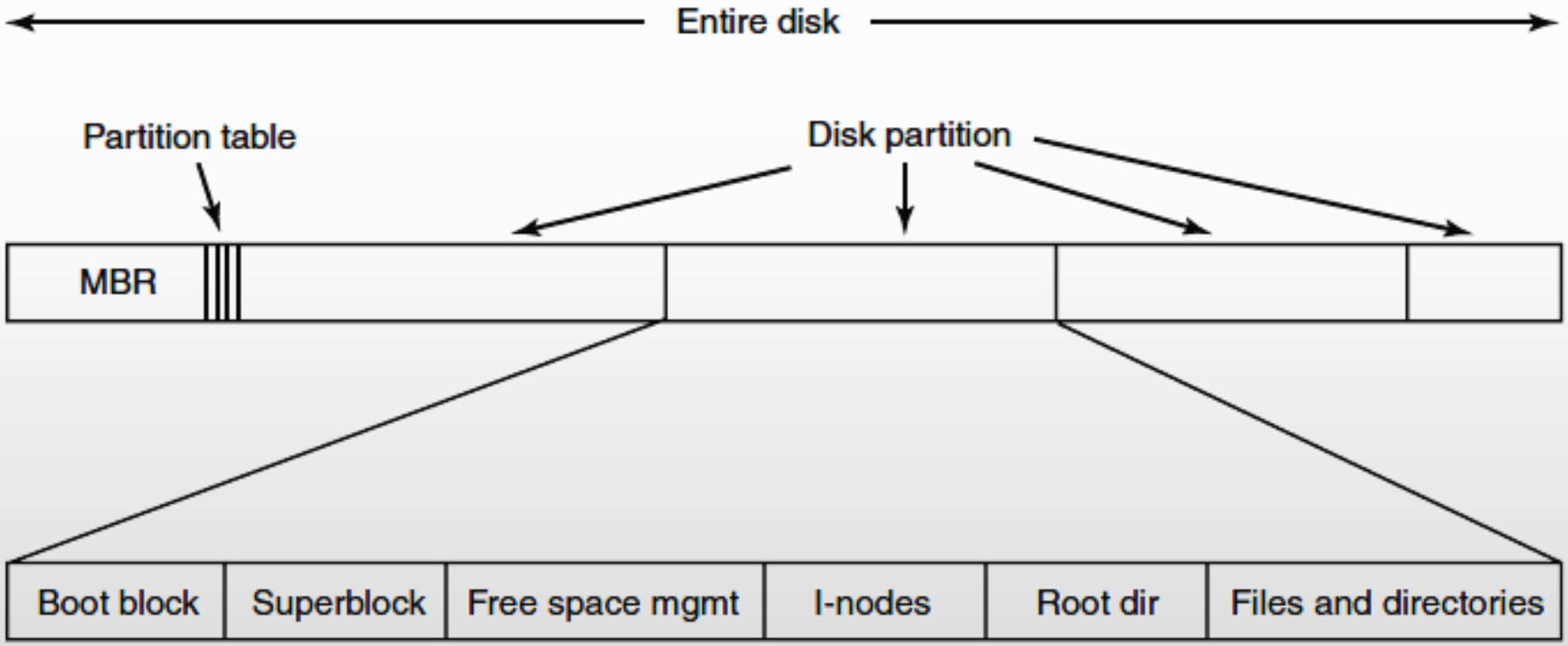
Disk Yapısı

- Dosyalar *diskte* saklanır.
- Diskler bir veya daha fazla bölümden (*partition*) oluşabilir.
- Diskin her bir bölümünde ayrı *dosya sistemi* olabilir.
- Diskin ilk (sıfırıncı) sektörü, ana önyükleme kaydıdır (*master boot record*).
- *MBR*, bilgisayarın açılışında (*boot*) kullanılır.
- *MBR*'nin sonunda bölüm tablosu (*partition table*) bulunur.
- Tabloda her bölümün başlangıç ve bitiş adresleri bulunur.
- Bölümlerden sadece biri, etkin (*active*) olarak işaretlenebilir.

Bilgisayar Açılış (Boot)

- *BIOS (Basic input output system)*, *MBR*'yi okur, yürütür.
- *MBR* aktif bölümü bulur ve ilk bloğu (*önyükleme bloğu*) okur.
- Önyükleme bloğundaki program, bölüm içinde işletim sistemini bulur, okur.
- Tüm bölümler (*partition*) bir önyükleme bloğuyla başlar.

Dosya Sistemi Düzeni (Layout)



Dosya Sistemi Düzeni (Layout)

- **Boot block**:
  - Diskin ilk bloğu. Önyükleyici kodunu içerir.
  - İşletim sistemi çekirdeğini belleğe yükler ve başlatır.
- **Superblock**:
  - Dosya sistemi hakkında bilgi içerir.
  - Boyutu, boş alan yönetimi, blok boyutu gibi.
- **Free Space Management**:
  - Diskte bulunan boş blokları takip eder.
  - Dosya sistemini düzenli tutar, parçalanmayı azaltır.
  - Disk alanını verimli kullanır.

Dosya Sistemi Düzeni (Layout)

- **i-nodes**:
  - Dosya ve dizinler hakkında bilgi içerir.
  - Boyut, izinler, sahiplik, zaman damgaları ve diskteki konumu gibi.
- **Root directory**:
  - Dosya sistemi hiyerarşisinde en üstte bulunan dizin.
  - Dosya ve dizinlere erişim için başlangıç noktası.
- **Files**:
  - Kalıcı olarak saklanması gereken verileri içerir.
- **Directory**:
  - Gezinme ve yönetim için dosyaları yapılandırır ve gruplar.

Blokların Dosyalara Tahsisi

- En önemli uygulama sorunu.
- Yöntemler
  - Bitişik yer (*contiguous*) tahsisi.
  - Bağlı liste (*linked list*) kullanılarak tahsis.
  - Tablo (*table*) kullanılarak tahsis.
  - I-nodes.

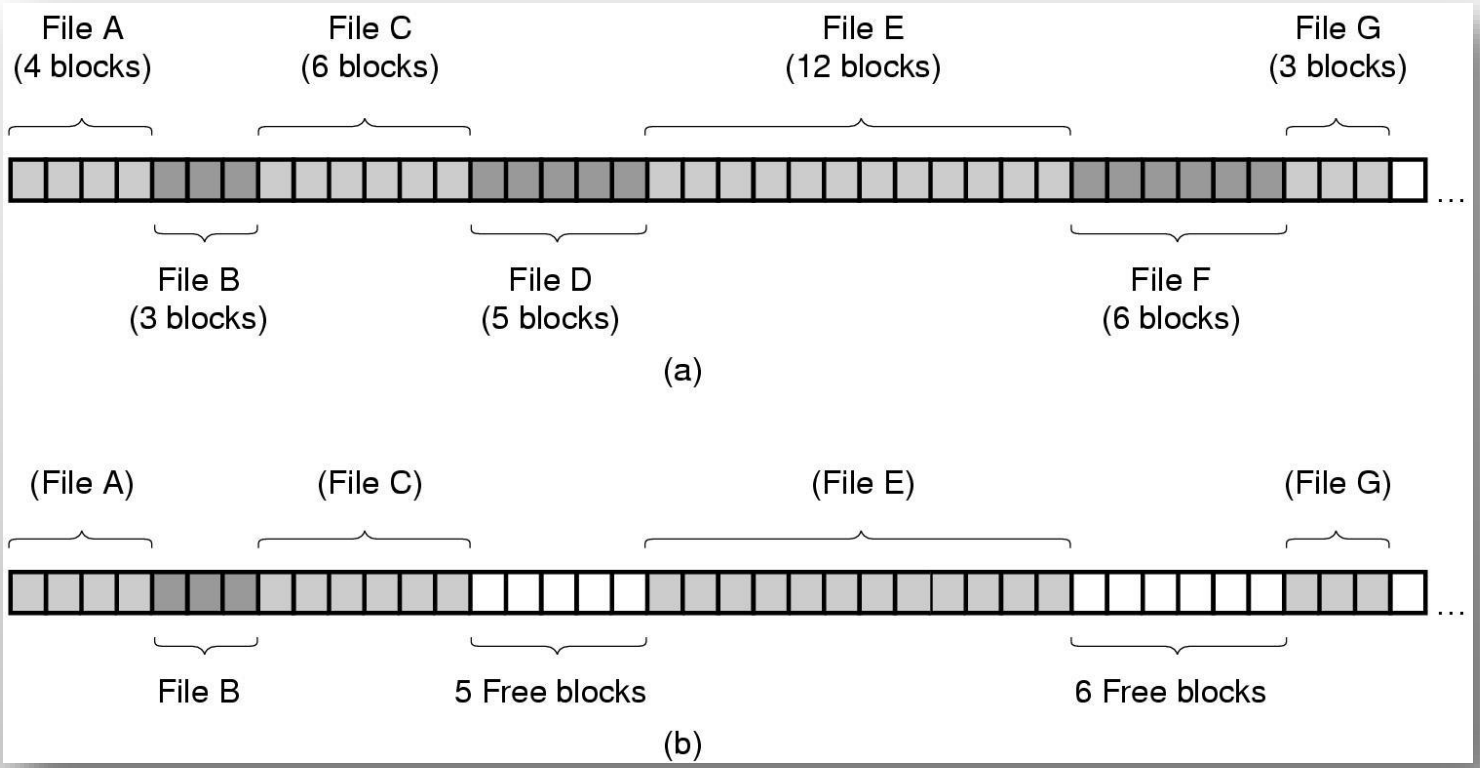
Bitişik Yer Tahsisi

- Dosyaya diskte bitişik boş bloklar atanır.
- Basit bir yöntem, okuma performansı iyi. ☺
- Dahili parçalanma nedeniyle disk alanı boşa harcanır.
- Büyük dosyalara yer tahsisi zordur.
- Dosyadaki ilk bloğu bulmak için sadece bir *seek* komutu yeterli.
- Disk kullanıldıkça zamanla parçalanır (*fragmented*).
- *CD-ROM*, dosya sistemi boyutu sabit olduğundan bitişik yer tahsisi.

Bitişik Yer Tahsisi

(a) 7 dosya için bitişik yer tahsisi.

(b) D ve *F* silindikten sonra diskin durumu.

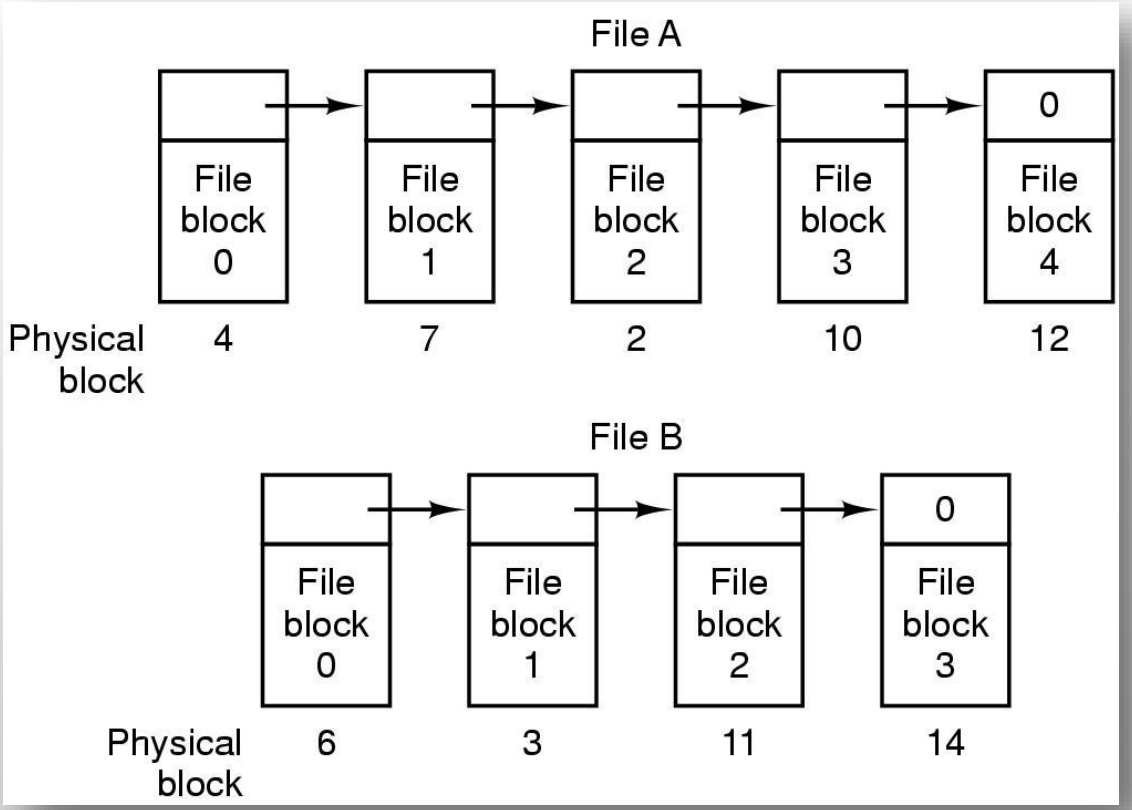


Bağlı Liste Kullanılarak Yer Tahsisi

- İşaretçiler ile disk blokları birbirine bağlanır.
- Bitişik yer tahsisinin kısıtlamalarını çözer.
- Dahili parçalanmayı en aza indirir. ☺
- Disk alanı dinamik ve verimli olarak kullanılır. ☺
- Verilere erişim daha yavaş. ☹
- Uygulanması daha karmaşık.
- İşaretçileri saklamak için ekstra bellek maliyeti gerekir.

Bağlı Liste Kullanılarak Yer Tahsisi

- Bağlı liste yapısında tutulan disk bloklarından oluşan iki örnek dosya.



Tablo Kullanılarak Yer Tahsisi

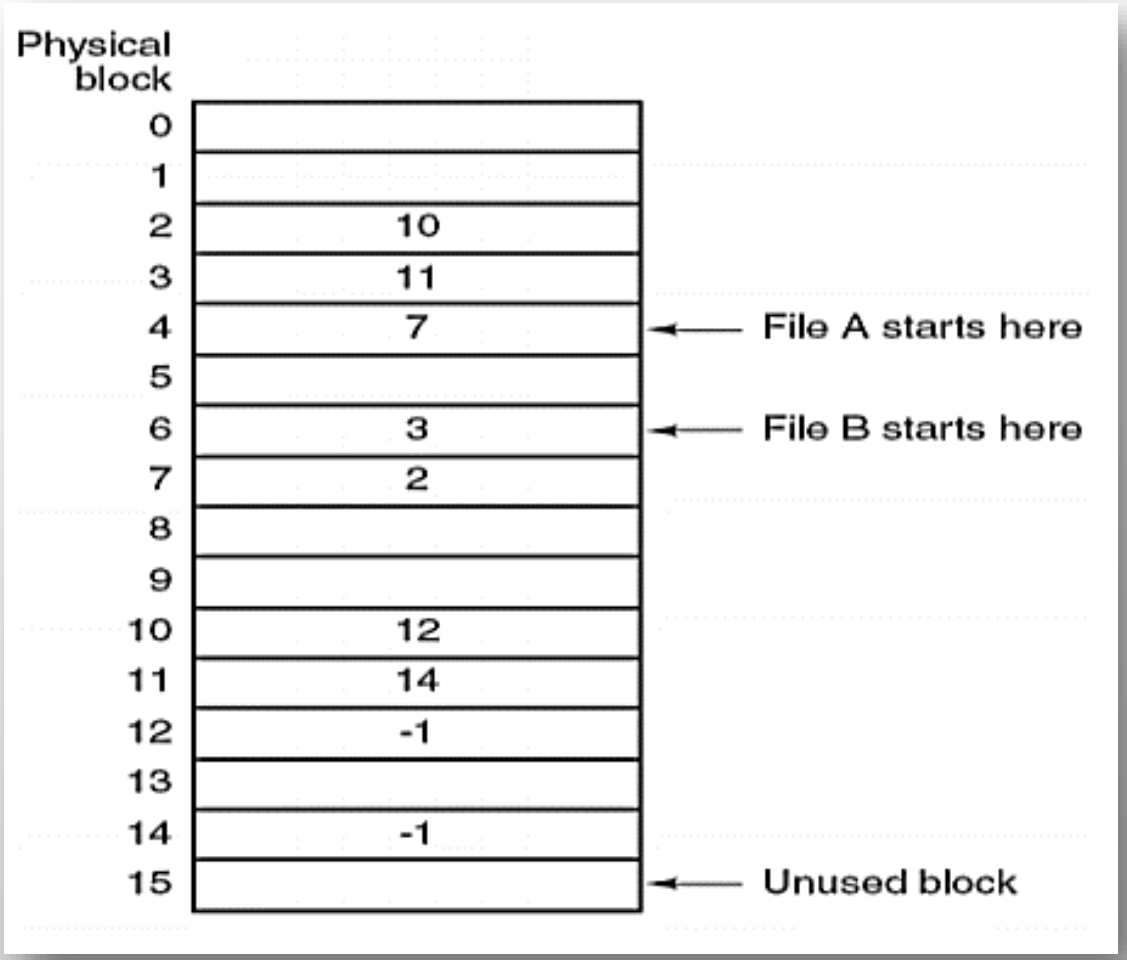
- Disk bloklarını gösteren işaretçiler bir tabloda saklanır.
- Bağlı liste yöntemine kıyasla,
  - Verilere erişim daha hızlı.
  - Bellek gereksinimi daha az.
- Uygulaması daha karmaşık.
- Dahili parçalanmaya neden olabilir.
- **Örnek**: *File Allocation Table (FAT)*

Tablo Kullanılarak Yer Tahsisi

- Tüm tablo bellekte tutulur! ☹
- Ölçeklenebilir değil! (*not scalable*) ☹
- Tablo boyutu, disk boyutu ile doğru orantılı.
- Tablonun boyutu gerçekten *çok büyük*.
- Örneğin,
  - blok boyutu 1 *KB* olsun.
  - disk boyutu 200 *GB* olsun.
  - Tablo için 800 *MB* gerekir.

Tablo Kullanılarak Yer Tahsisi

- Ana bellekte tutulur.
- Tablo kullanılır.
- Örnek iki dosyanın gösterimi.
- A: 7 2 10 12 -1
- B: 3 11 14 -1



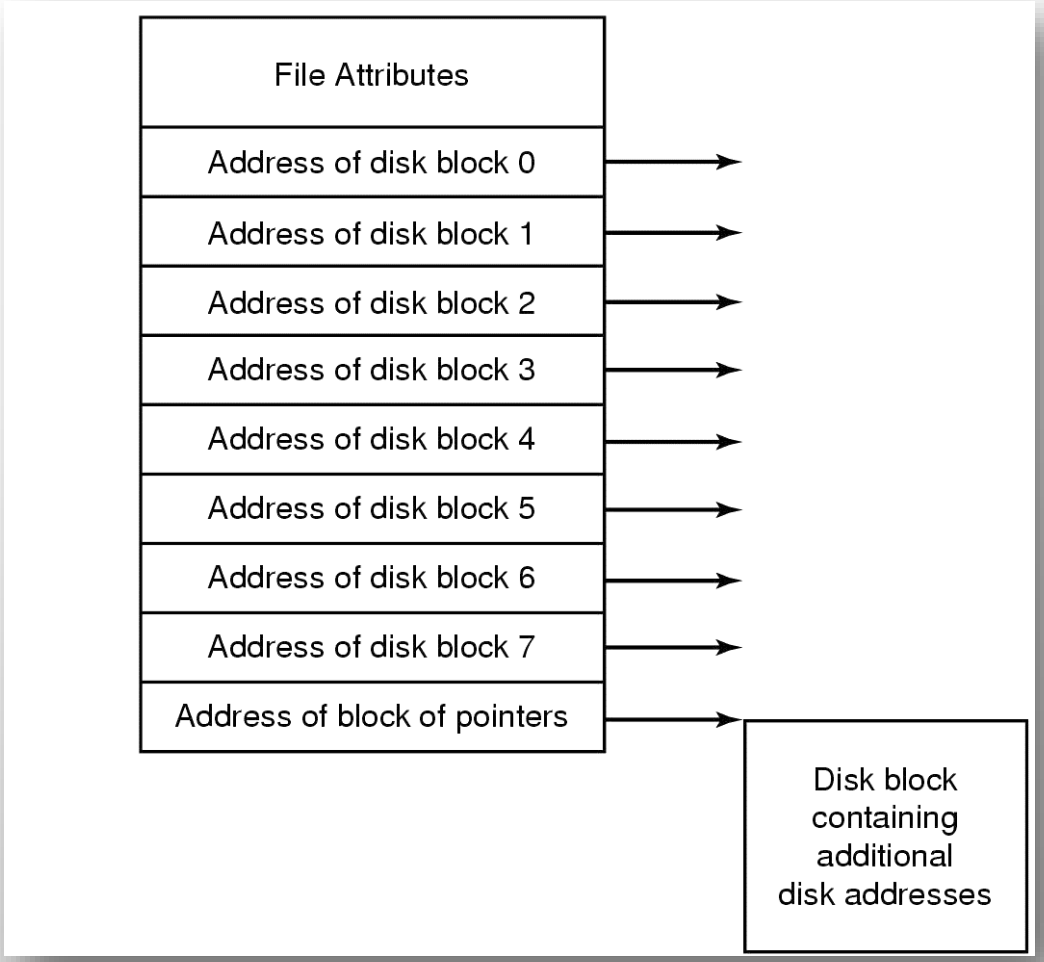
I-nodes

- i-düğüm tablosu kullanılır.
- Dahili parçalanmayı en aza indirir.
- Disk alanının esnek bir şekilde tahsis edilmesini sağlar.
- Meta verileri yönetmek için esnek ve verimli bir yöntem sağlar.
- Uygulanması daha karmaşıktır.
- i-düğümleri saklamak için ekstra bellek gerektirir.
- Disk kafasının daha fazla hareket etmesine neden olabilir.

I-nodes

- Veri yapısını yalnızca açık dosyalar için bellekte tutar. ☺
- Veri yapısı, blokların adreslerini ve dosyaların özniteliklerini saklar.
- *K* aktif dosya, dosya başına *N* blok, en fazla *KN* blok bellekte yer kaplar.
- Disk boyutu ile orantılı olarak büyüme sorununu çözer. ☺
- *N* ne kadar büyük olabilir?
- Tablodaki son satır, diğer disk bloklarına işaret eder.
- i-node büyüklüğü sabittir!

Örnek I-node



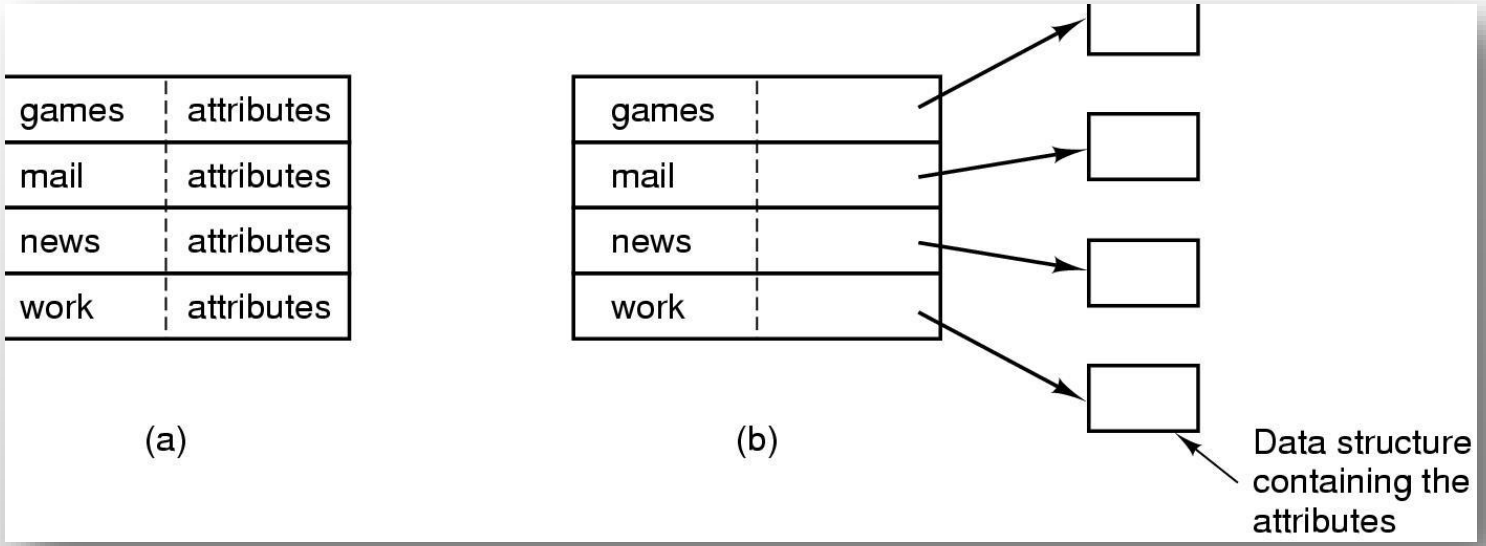
Dizinler

- Dizinin diskteki adresini bulmak için,
  - İlk bloğun adresini (*bitişik yer tahsisi*).
  - İlk bloğun sayısını (*bağlı liste kullanarak yer tahsisi*).
  - i-node sayısını (*i-nodes*) bilmek gereklidir.

Dizinler

(a) disk adresleri ve öznitelikleri tutan sabit boyutlu girdiler. (*DOS*)

(b) her girdi öznitelikleri içeren bir i-node'u işaret eder. (*Unix*)

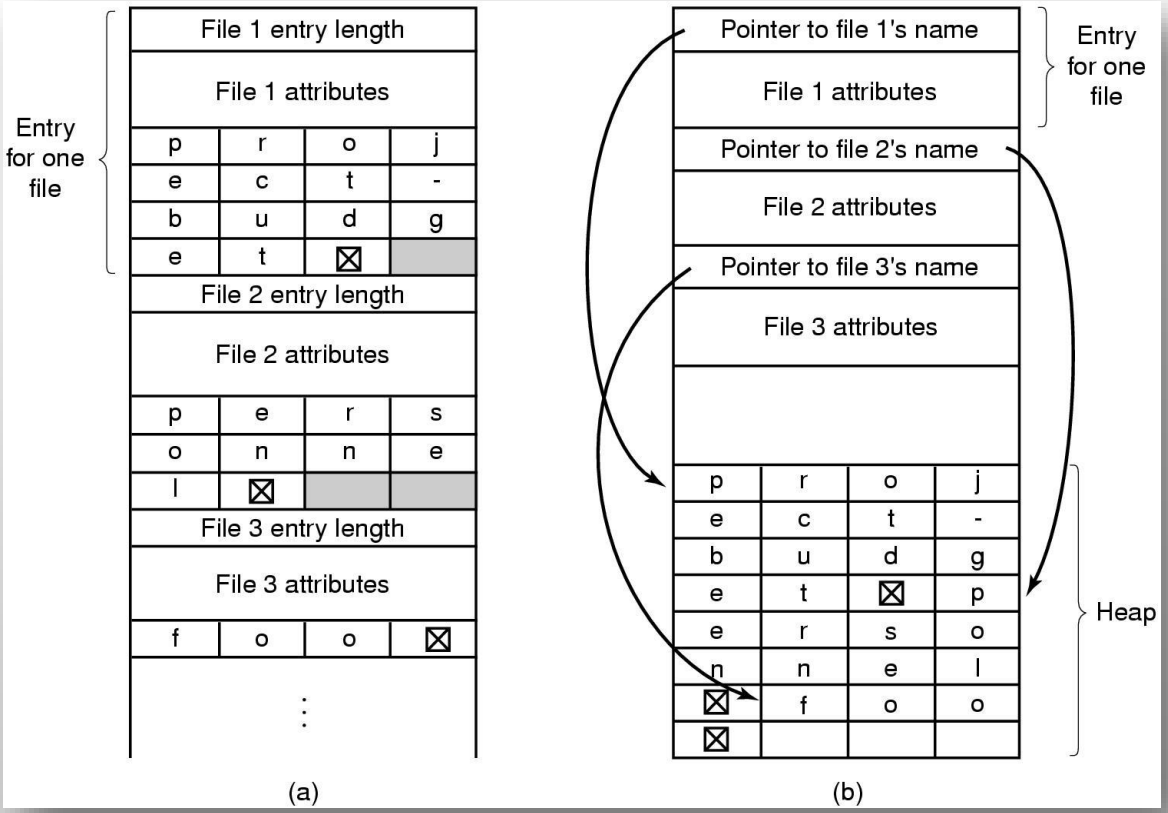


Dizinler

- Değişken uzunluklu dosya/dizin adları?
- Çok uzun adlar problem.! ☹
- İki yaklaşım var.
  - Sabit başlık ve ardından değişken boyutlu alan kullanılır.
  - Yığın kullanılır.

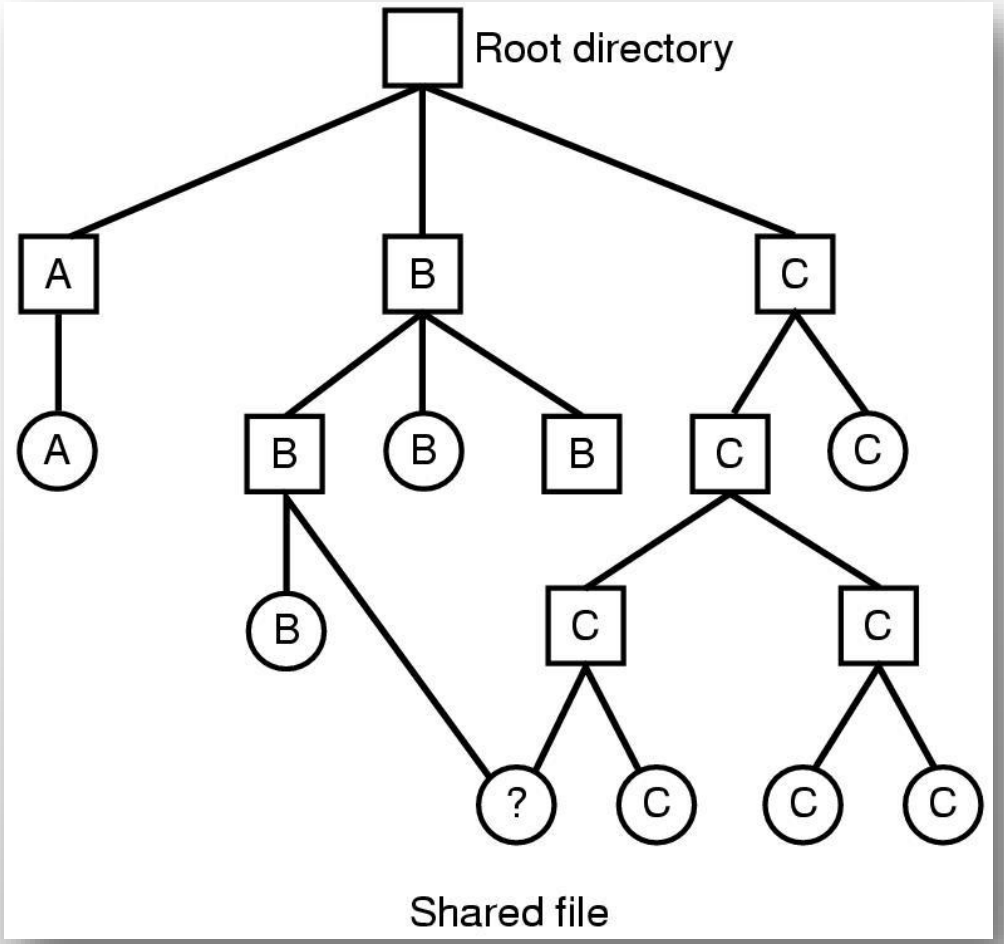
Dizinler - Uzun Dosya Adları

- (a) Sabit başlık, değişken boyutlu alan. (b) Yığın içinde.



Paylaşımlı Dosyalar

- Paylaşımlı dosya içeren dosya sistemi.
- *?* paylaşımlı dosyadır.
- Yönlendirilmiş döngüsüz ağaç.
- Directed acyclic graph (*DAG*).



Paylaşımlı Dosyalar

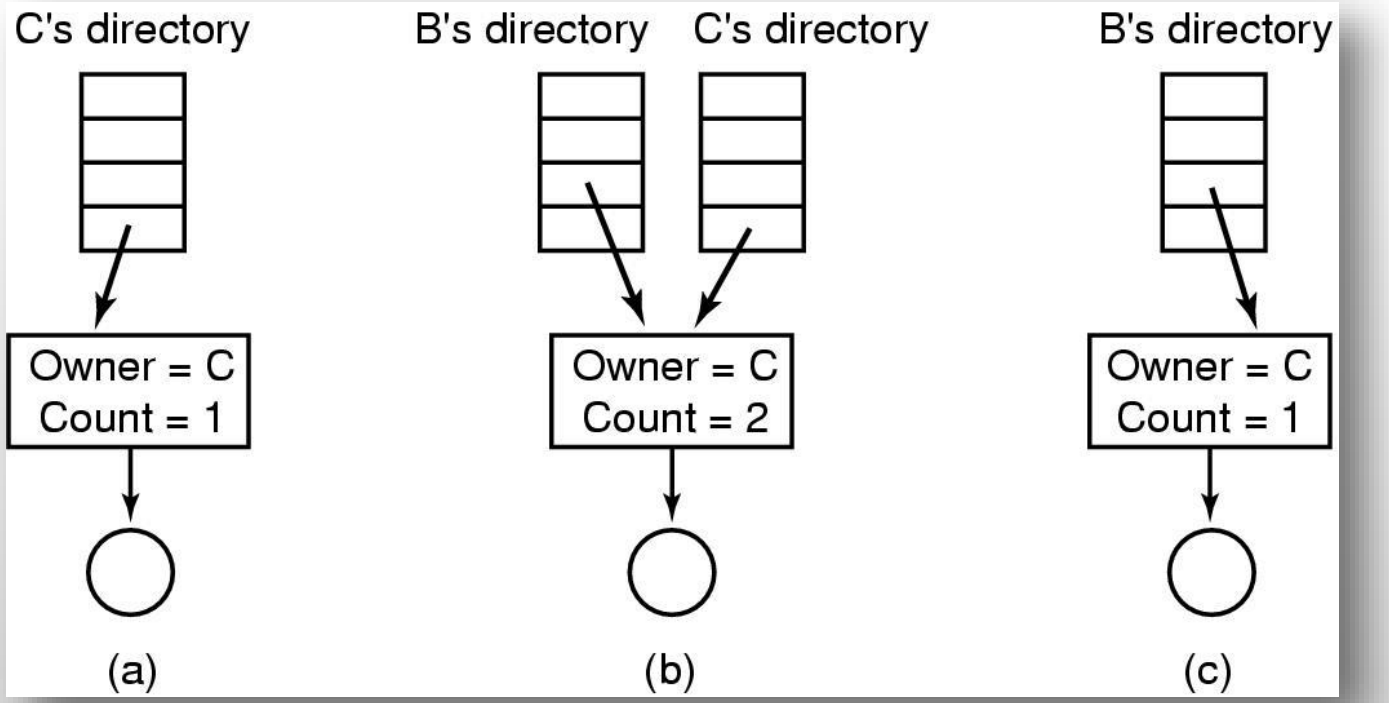
- *B* yeni blok eklerse, *C* nasıl haberdar olur?
- Paylaşımlı dosyalar için özel i-node yapısı kullanılır.
- Sembolik bağlantı (*symbolic link*) kullanılır.
  - Dosyanın sahibi *C* ise, *B* dizinine özel bir dosya konulur.
  - Bağlı (*linked*) olduğu dosyanın yolunu (*path*) içerir.

I-node Problem

- *C* dosyayı silerse, *B* dosya için hala i-node'u işaret etmeye devam eder.
- i-node başka bir dosya için yeniden kullanılırsa, *B* yanlış i-node'u gösterir.
- Çözüm, i-node silinmez, sahip sayısı azaltılır.
- Sembolik bağlantı sorunu çözer. ☺
  - Ancak, dosya çok fazla sembolik bağlantıya sahip olabilir.
  - Bunların takip edilmesi zaman alır. ☹
  - Diğer makinede bir dosyaya işaret edebilir. (*Büyük avantaj*)

I-node Problem

Bağlamadan (a) önce. (b) sonra. (c) sahibi dosyayı sildikten sonra.

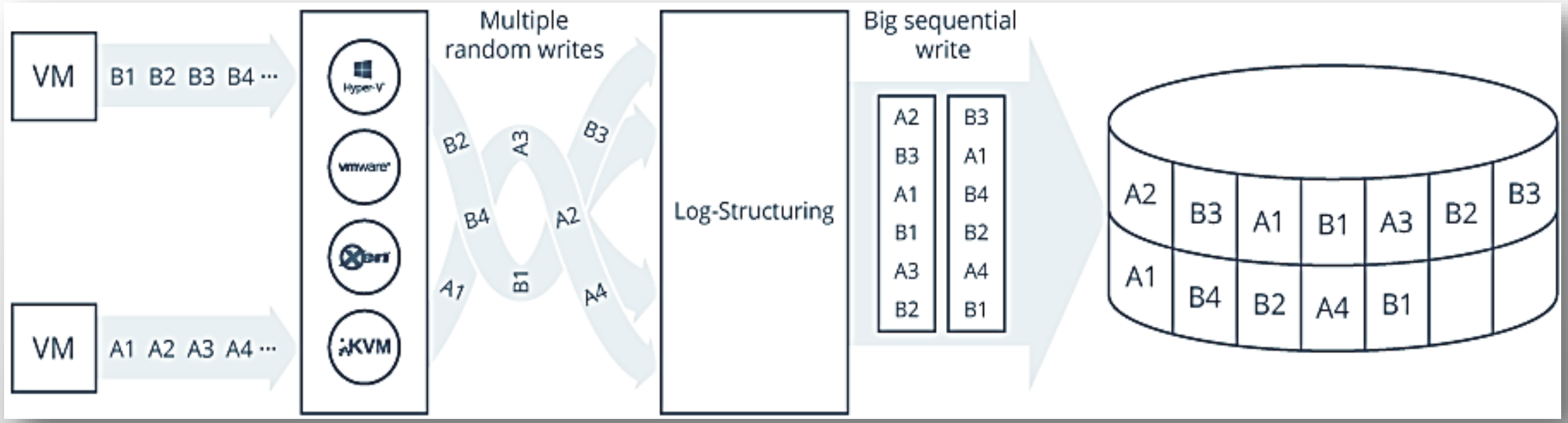


Günlük Olarak Yapılandırılmış Dosya Sistemi

- *Log-structured file system.*
- *Data and metadata are written sequentially to circular buffer, called a log.*
- Disk işlemlerini bir günlük veya güncelleme dizisi olarak kayıt eder.
- Rastgele disk erişiminden kaçınarak, yüksek performans sağlar.
  - Geliştirilmiş yazma performansı, ☺
  - Basitleştirilmiş disk alanı yönetimi ve
  - Verimli disk alanı kullanımı sağlar.
- Günlük düzgün bir şekilde korunmaz ise,
  - Potansiyel veri kaybı. ☹
  - Daha yüksek ek maliyet.

Günlük Olarak Yapılandırılmış Dosya Sistemi

- *A log is a database file that contains the history of operations.*



Günlük Olarak Yapılandırılmış Dosya Sistemi

- İşlemci daha hızlı, disk ve bellek daha büyük,
  - Ancak; disk arama süresi kısa değil.
- Diske yazma işlemleri optimize edilmeli.
- Daha büyük önbellek kullanılarak, okuma işlemi önbellekten yapılabilir.
- i-node eşlemeleri diskte tutulur, önbelleğe alınabilir.
- Disk *log-collect* olarak yapılandırılır, ve
  - Günlükler periyodik olarak diskteki bir kesime (*segment*) gönderilir.
- Kesim, içerik özetine sahiptir (*i-nodes, dizinler ..*).

Günlük Olarak Yapılandırılmış Dosya Sistemi

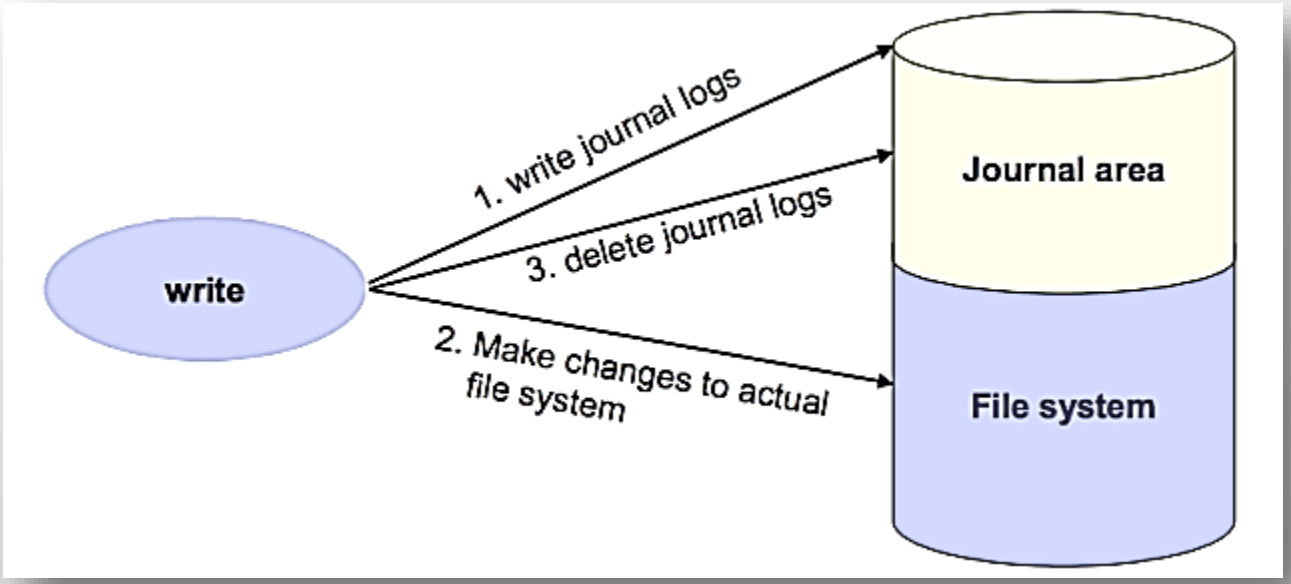
- Temizleyici iş parçacığı,
  - Günlük dosyasını sıkıştırır.
  - Kesimi mevcut i-nodes'lar için tarar.
  - Kullanılmayanları atar.
  - Mevcut olanları belleğe gönderir.
- Yazıcı iş parçacığı,
  - Mevcut olanları yeni kesime yazar.
- Çoğu dosya sistemiyle uyumlu değil. ☹
- Kullanılmıyor. ☹

Günlük (journaling) Dosya Sistemleri

- Diske kaydedilmeden önce, değişikliklerin günlüğünü tutar.
- Çökme veya sistem arızası durumunda dosya sistemi kurtarılabilir.
- Sağlam ve güvenilir bir dosya sistemi sağlar. ☺
- Günlük tutma işlemi nedeniyle,
  - Disk G/Ç maliyeti artar.
  - Performans düşer.

Günlük (journaling) Dosya Sistemleri

- *Journal contains information about changes made to another object.*



Günlük (journaling) Dosya Sistemleri

- İşlemleri gerçekleştirmeden önce günlük tutar, günlüğü diske yazar.
- *Windows (NTFS*), *Linux (ext3), Apple (HFS Plus)* kullanır.
- Bir dosya silineceğinde,
  - Dosyayı bulunduğu dizinden kaldırır.
  - i-node'u *serbest i-node havuzuna* bırakır.
  - Disk bloklarını, *boş disk blokları havuzuna* gönderir.
  - Bu süreçte bir çökme olursa, ortalık karışır. ☹

Günlük (journaling) Dosya Sistemleri

- İşlemler eşgüçlü (*idempotent*) olmalı.
  - Uygun veri yapısı kullanılmalı.
- ***İdempotence***: Bir işlem kaç kere gerçekleştirilirse gerçekleştirilsin, aynı sonuç elde edilir.
- Blok *n*'yi serbest bırak, *idempotent* bir işlemdir.
- Bir listenin sonuna, serbest bırakılmış bloklar eklemek *idempotent* değildir.

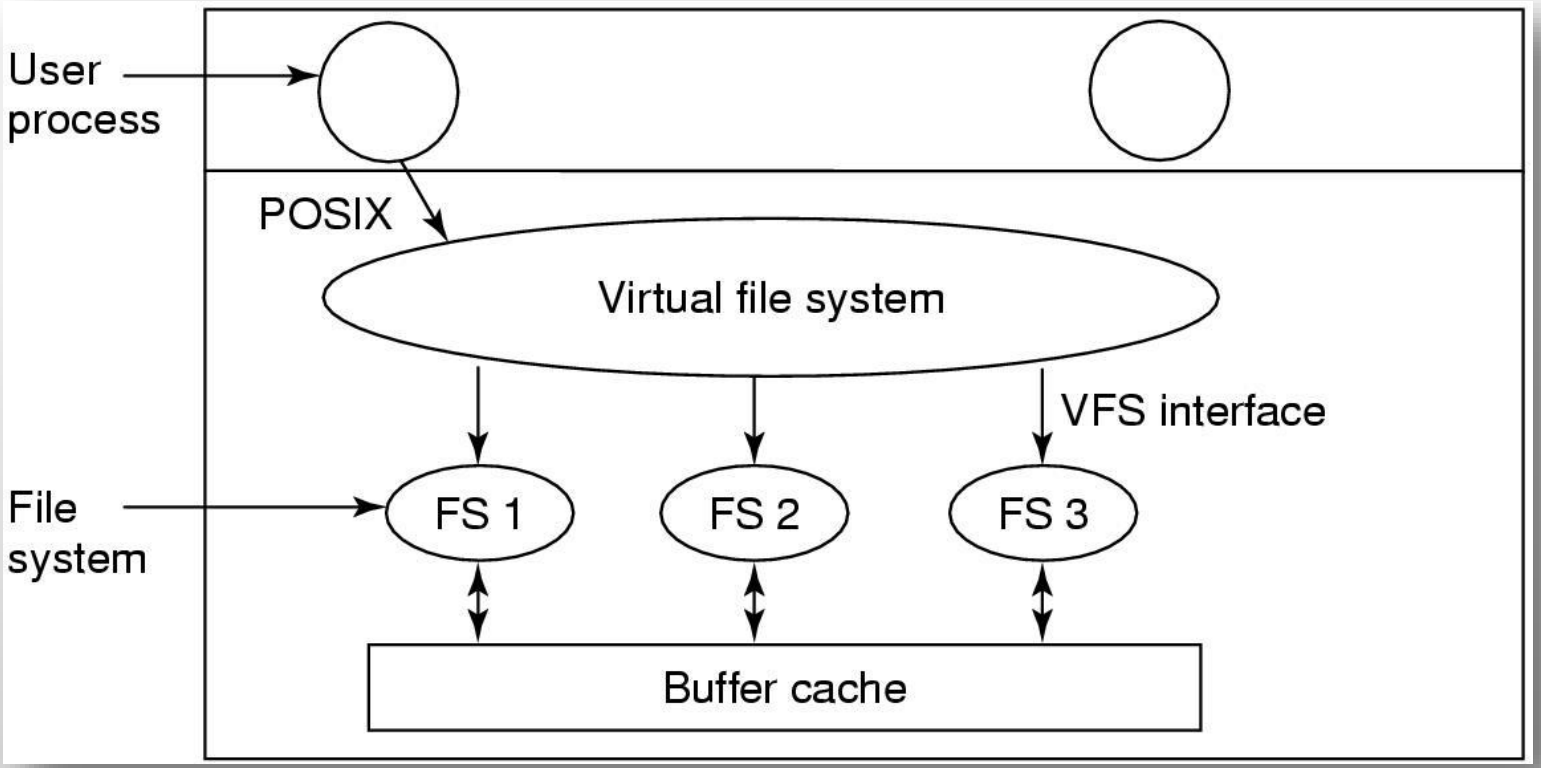
Sanal Dosya Sistemleri

- *Virtual file system*
- Farklı dosya sistemlerine erişim için ortak bir arayüz sağlar.
- Farklı dosya sistemlerinin aynı disk üzerinde bulunmasına izin verir.
- Esnek ve ölçeklenebilir bir dosya sistemi sağlar.
- Farklı disk düzenlerini ve dosya sistemi türlerini işleyebilir,
  - Artan karmaşıklık.
  - Ek bir soyutlama katmanı.
  - Düşük performans.

Sanal Dosya Sistemleri

- Windows,
  - Dosya sistemi sürücüleri tanımlar.
- Unix,
  - Sanal dosya sistemine entegre olur.
  - Kullanıcılar VFS sistem çağrıları yapar.
  - Alt seviye çağrılar, gerçek dosya sistemine yapılır.
- Ağ dosya sistemini (*network file system*) destekler.
  - Dosya başka bir makinede olabilir.

Sanal Dosya Sistemleri

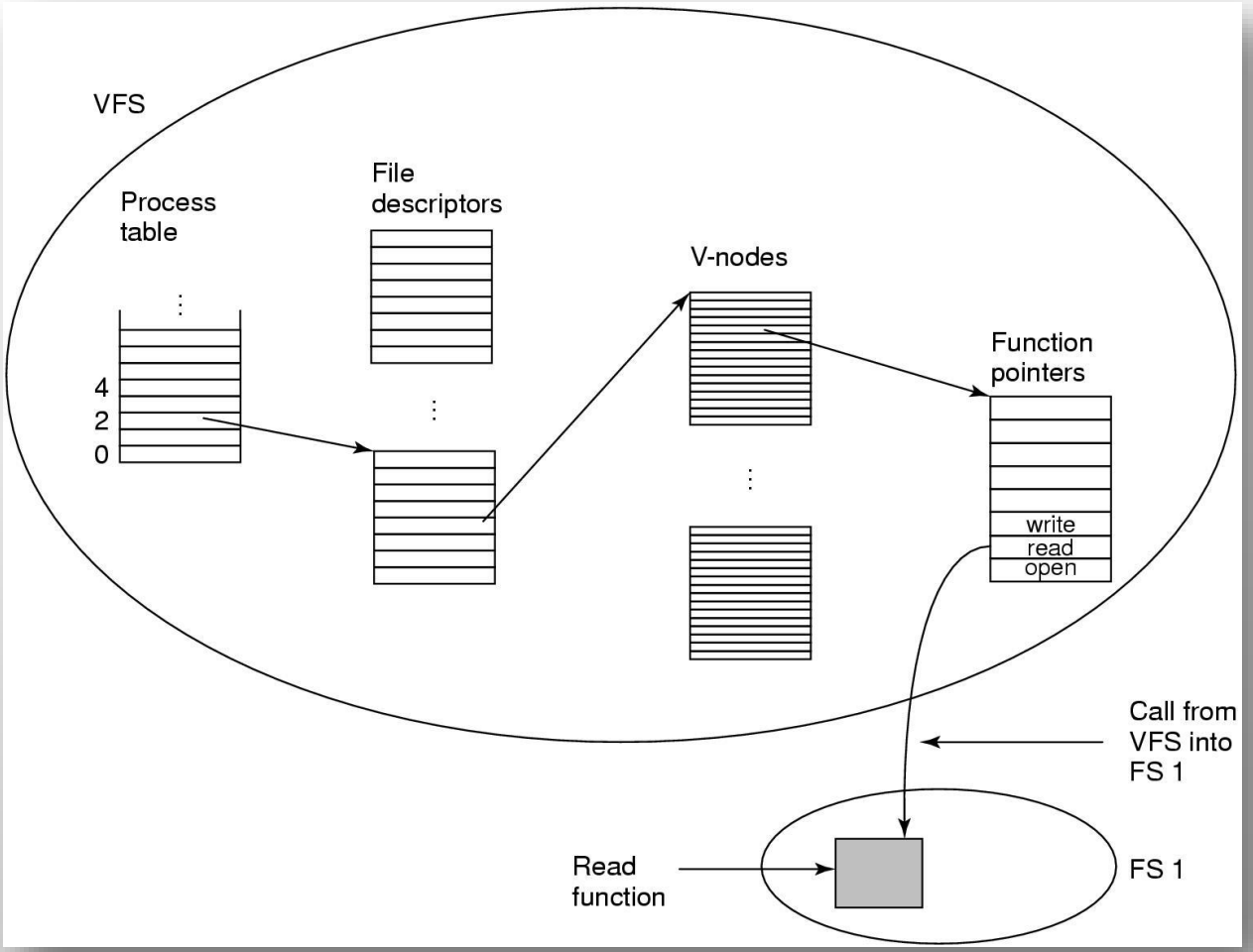


Sanal Dosya Sistemi Nasıl Çalışır

- Dosya sistemi VFS'ye önyükleme sırasında kaydolur.
- Kayıt sırasında dosya sistemi,
  - VFS'nin istediği fonksiyon çağrılarının adres listesini sağlar.
- VFS, dosya sisteminden i-node bilgisini alır ve bir v-node'a yerleştirir.
- Süreç için dosya tanımlayıcı (*file descriptor*) tablosuna ekler.
- Süreç bir çağrı yaptığında (*read, write ..*),
  - Fonksiyon işaretçileri somut fonksiyon çağrılarına işaret eder.

Sanal Dosya Sistemleri

- Kullanılan veri yapıları.
  - V-nodes.
  - Fonksiyon işaretçileri.
  - Süreç tablosu.
  - Dosya tanımlayıcıları.

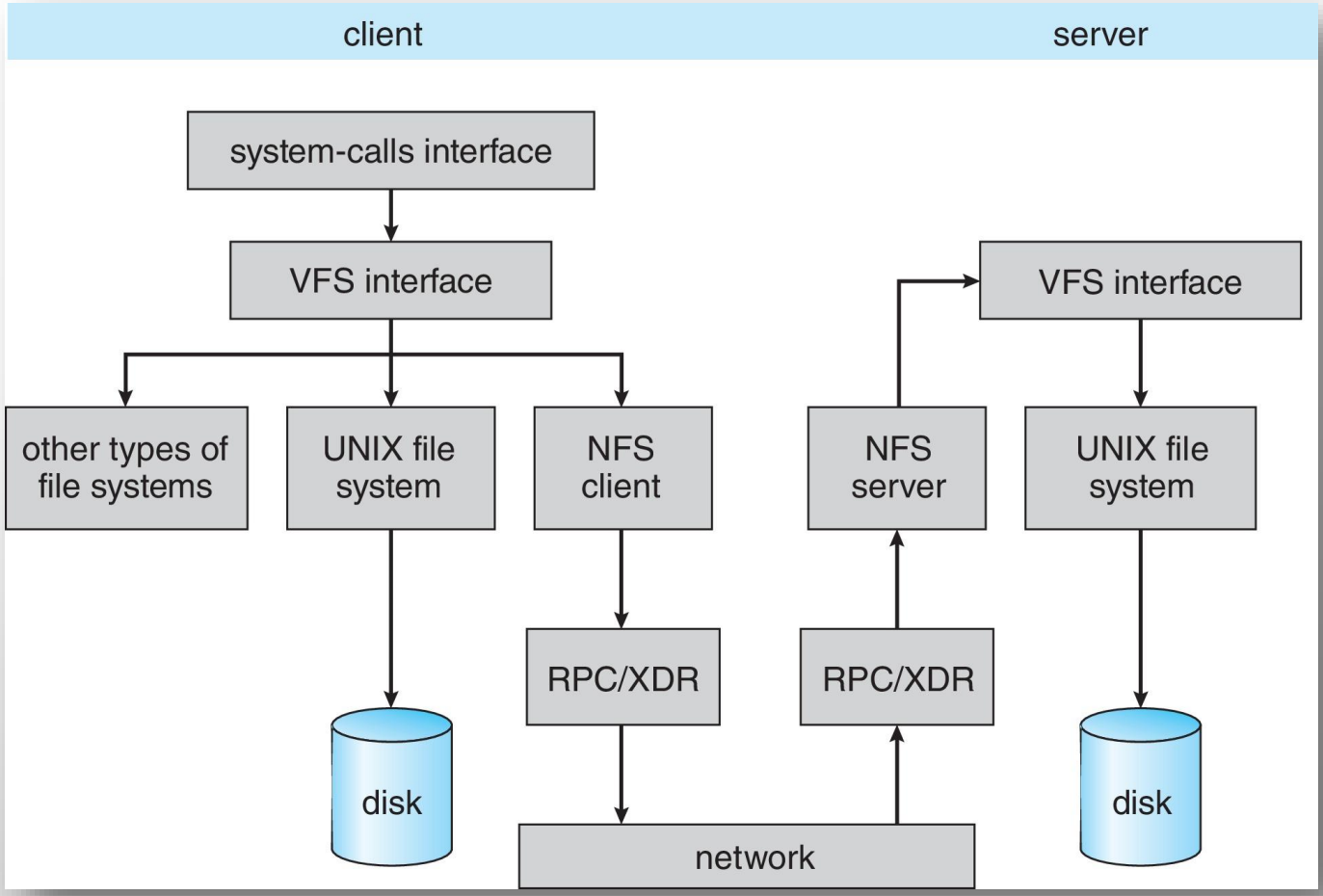


Ağ Dosya Sistemi

- *Network file system*.
- Ağlar üzerinden uzak dosyalara erişim için yazılım arayüzü.
- *SunOS*'un bir parçası iken, endüstri standardı oldu.
- *Ethernet* üzerinden *UDP/IP* veya *TCP/IP* ile kullanılabilir.
- Uzakta bir dizin, yerel bir dosya sistemi dizinine bağlanır (*mount*).
- Farklı makineler, işletim sistemleri ve ağ mimarilerinde çalışabilir.
- Bu bağımsızlık,
  - *XDR (eXternal Data Representation*) protokolü ve
  - *RPC (remote procedure call)* kullanılmasıyla elde edilir..

Ağ Dosya Sistemi

- Network file system.
- VFS arayüzü.
- NFS sunucusu.
- NFS istemcisi.



Dosya Sistemi Yönetimi ve Optimizasyonu

- Disk alanı yönetimi
- Dosya sistemi yedekleme
- Dosya sistemi tutarlılığı
- Dosya sistemi performansı

Disk Alanı Yönetimi

- Bitişik olması gerekmeyen, sabit boyutlu bloklar kullanılmalı.
- Dosyalar ardışık bayt serisi olarak saklanırsa,
  - Dosya büyüdüğünde taşınması gerekir!
- Optimum (en iyi) blok boyutu nedir?
  - Dosya boyutu dağılımı bilinmeli.
- Dosya sistemi tasarlarken genel (*generic*) düşünülmeli.

Disk Alanı Yönetimi

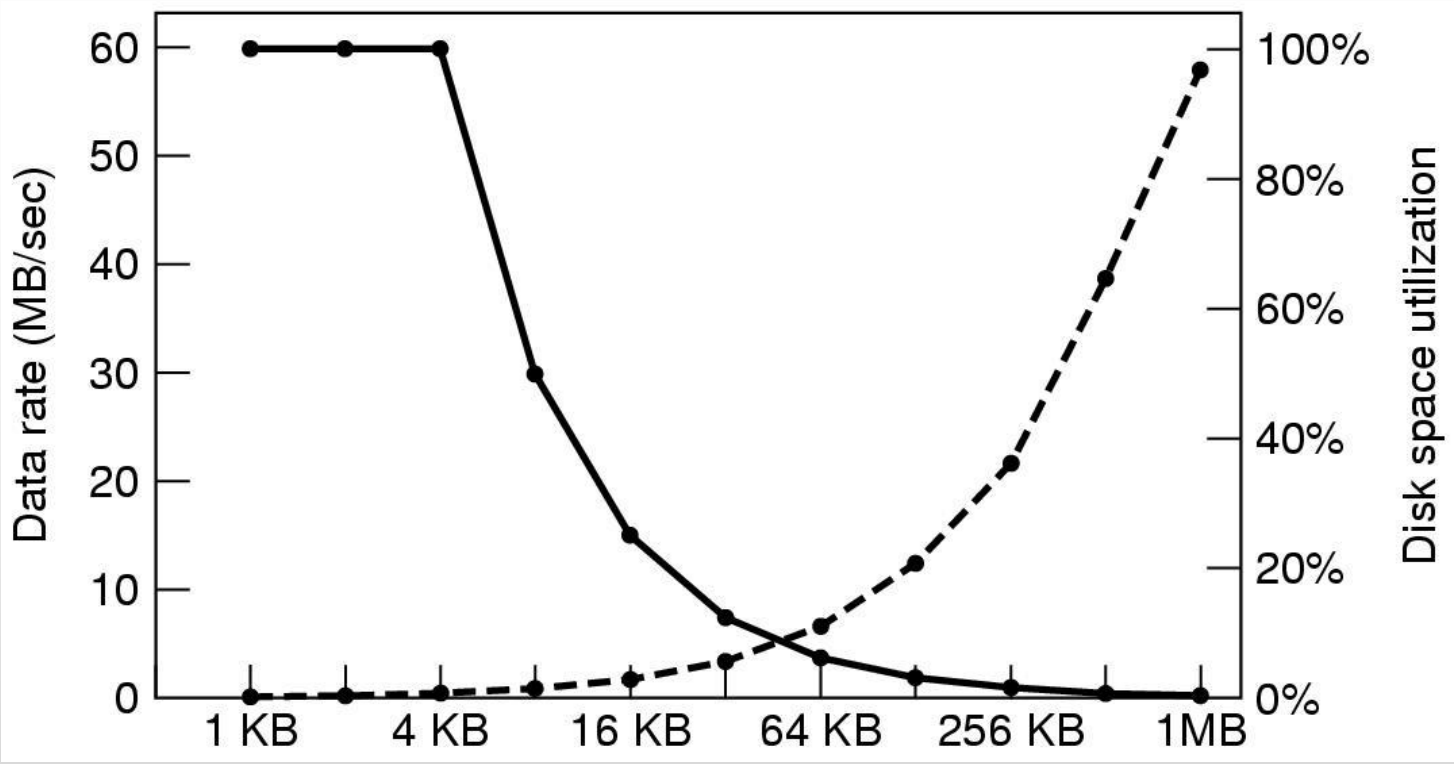
- Verilen boyuttan küçük olan dosyaların yüzdesi. (VU örnek bir sunucu)

Length	VU 1984	VU 2005	Web
1	1.79	1.38	6.67
2	1.88	1.53	7.67
4	2.01	1.65	8.33
8	2.31	1.80	11.30
16	3.32	2.15	11.46
32	5.13	3.15	12.33
64	8.71	4.98	26.10
128	14.73	8.03	28.49
256	23.09	13.29	32.10
512	34.44	20.62	39.94
1 KB	48.05	30.91	47.82
2 KB	60.87	46.09	59.44
4 KB	75.31	59.13	70.64
8 KB	84.97	69.96	79.69

Length	VU 1984	VU 2005	Web
16 KB	92.53	78.92	86.79
32 KB	97.21	85.87	91.65
64 KB	99.18	90.84	94.80
128 KB	99.84	93.73	96.93
256 KB	99.96	96.12	98.48
512 KB	100.00	97.73	98.99
1 MB	100.00	98.87	99.62
2 MB	100.00	99.44	99.80
4 MB	100.00	99.71	99.87
8 MB	100.00	99.86	99.94
16 MB	100.00	99.94	99.97
32 MB	100.00	99.97	99.99
64 MB	100.00	99.99	99.99
128 MB	100.00	99.99	100.00

Disk Alanı Yönetimi

- Düz eğri veri hızını, kesikli eğri verimliliği gösterir. Dosya boyutu 4 *KB*.

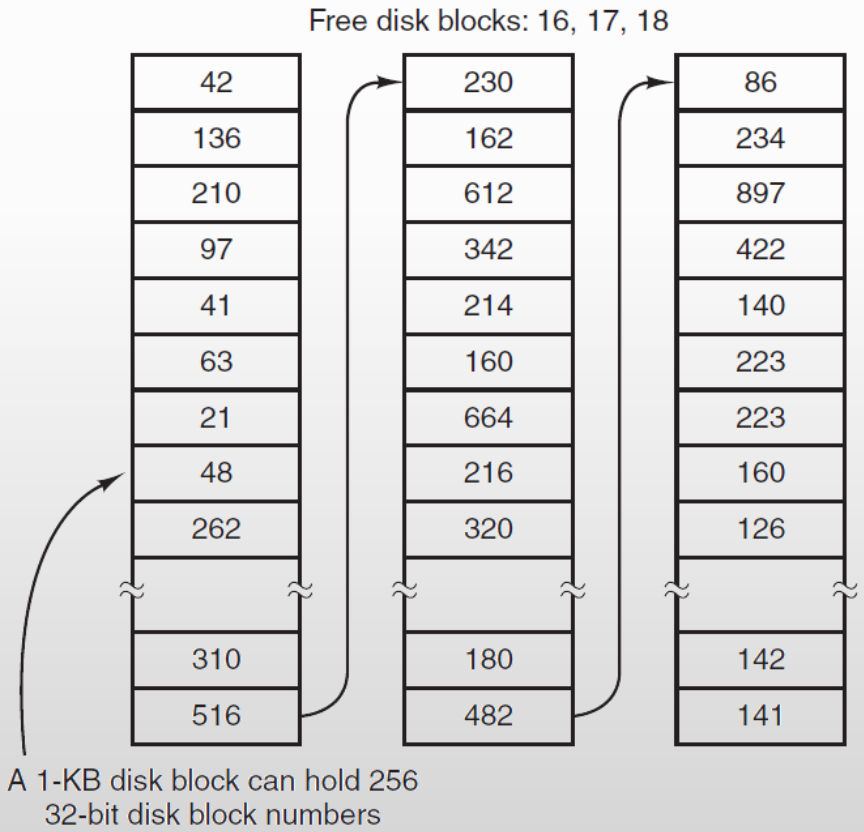


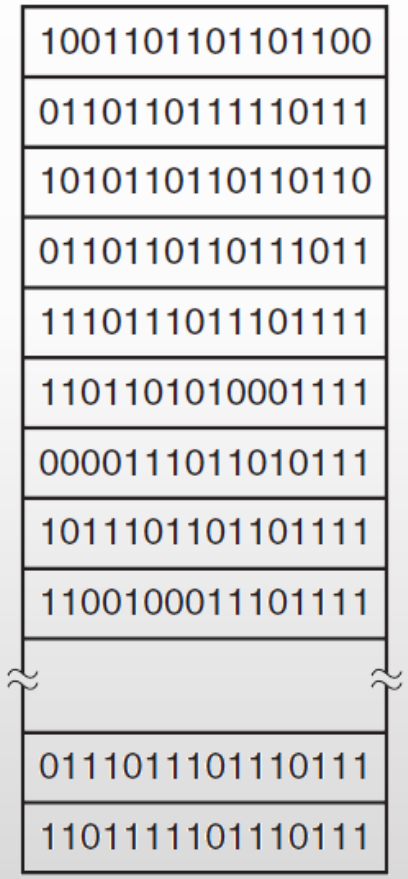
Disk Alanı Yönetimi

- Büyük blok boyutu,
  - Daha iyi alan kullanımına,
  - Daha kötü aktarım (*transfer*) hızına neden olur.
- Alan ve veri hızı birbiriyle ters orantılı (*trade-off*).
- Kesin iyi bir çözüm yok (*Nature wins this time*).
- Disk yeterince büyükse, büyük blok boyutu (64 *KB*) kullanılmalı.

Boş Blokların İzini Tutma

(a) Bağlı liste (b) biteşlem olarak.



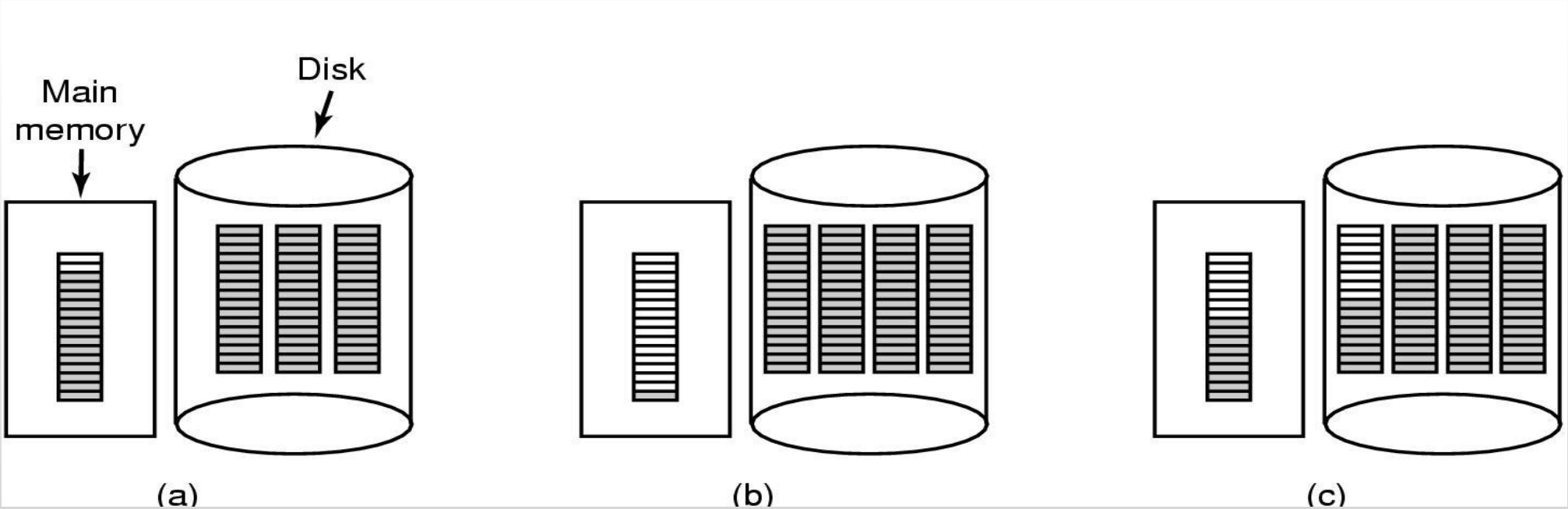


Boş Blokların İzini Tutma

- Bağlı liste yapısı ~1,9 milyon bloğa ihtiyaç duyar.
- Biteşlem yapısı ~60.000 bloğa ihtiyaç duyar.
- Herhangi bir anda bellekte sadece bir işaretçi bloğuna ihtiyaç vardır.
- Bloğu doldur, bir tane daha al.

Boş Blokların İzini Tutma

- (a) Bellekte dolmaya yakın bir işaretçiler bloğu ve diskte üç işaretçi bloğu.
- (b) Üç bloklu bir dosyayı serbest bıraktıktan sonra.
- (c) Üç boş bloğu işlemek için alternatif bir strateji.

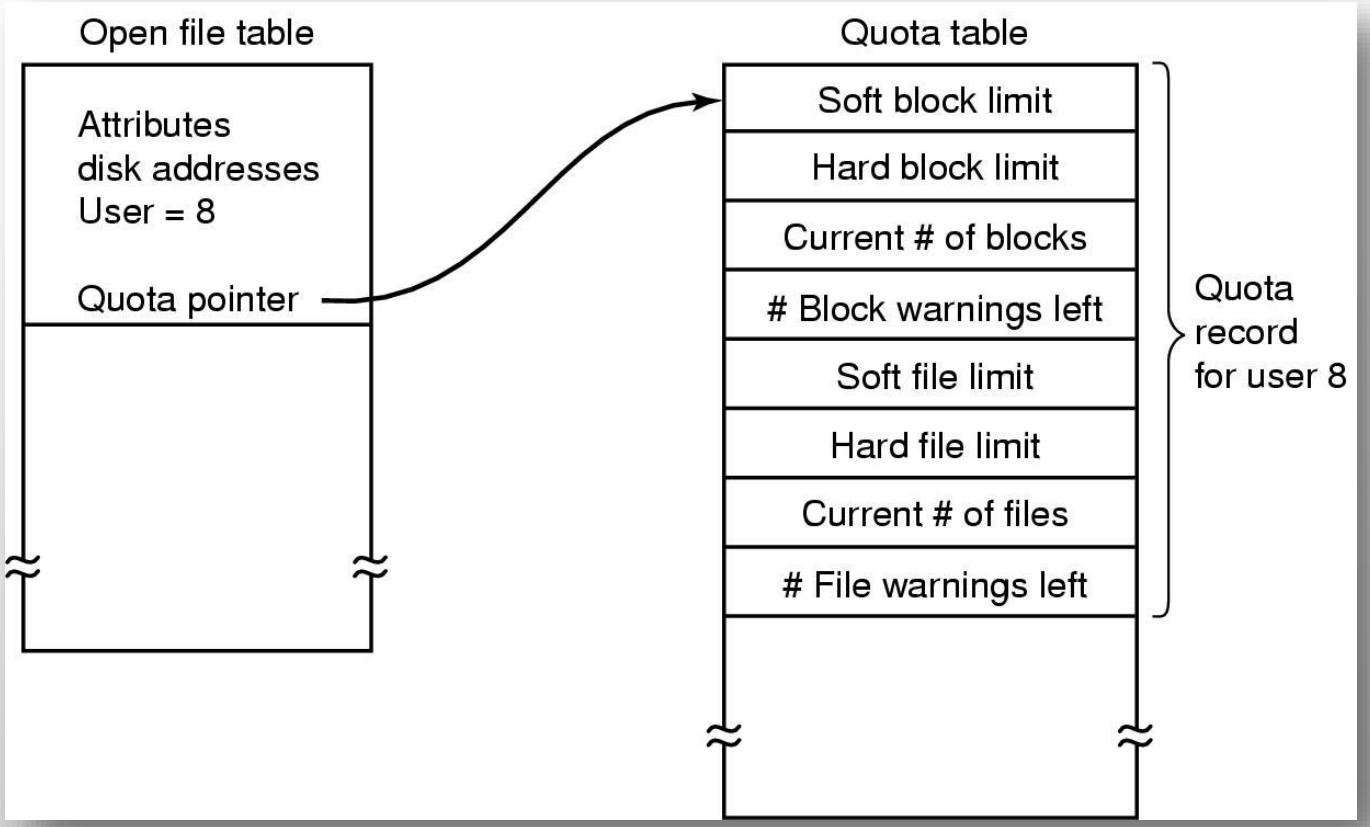


Disk Kotaları

- Açık dosyalar tablosundaki satır (*entry*),
  - Kota tablosuna işaret eder.
- Her açık dosya için bir satır var.
- Kullanıcıların disk kotasına sınır (*soft, hard*) konulabilir.

Disk Kotaları

- Kota tablosunda kullanıcı (*user*) bazlı izlenir.



Dosya Sistemi Yedeklemeler

- Yedeklemeler iki olası sorundan dolayı yapılır.
  - Felaketten (*disaster*) kurtulmak için (*disk çökmesi*).
  - Dikkatsizlik sonucu (*yanlışlıkla silinen dosya*).
- Teypler yüzlerce *GB* veri tutar ve çok ucuzdur.
- Tüm dosyaları yedeklemeye gerek yok.
- Geçici dosyaların yedeklenmesi gerekmez.
- Özel dosyaların (*G/Ç*) yedeklenmeye ihtiyacı yoktur.

Kademeli Olarak Yedekleme

- Son dökümden (**dump**) sonra değiştirilen dosyaların haftalık/aylık listesi.
- Dosya sistemini geri yüklemek için tam döküm gerekli.
- Değiştirilmiş dosyaları döküme dahil etmek için iyi algoritmalar gerekli.
- Sistem kullanılırken döküm performans açısından zor.
- Anlık görüntü (*snapshot*) algoritmaları mevcut.

Döküm Stratejileri - Fiziksel

- Fiziksel olarak tüm her şey dökülür, kaydedilir.
- Uygulaması basit.
- Kullanılan bloklar için blok numaraları teybe yazılmalı.
- İşletim sistemi kullanılmayan blokların listesini tutar.
- Disk denetleyicisi,
  - Kötü (*bozuk*) blokları algılamalı ve değiştirmelidir, veya
  - Nerede olduklarını bilmelidir.
  - İşletim sistemi kötü (*bozuk*) blokların listesini tutar.

Döküm Stratejileri - Fiziksel

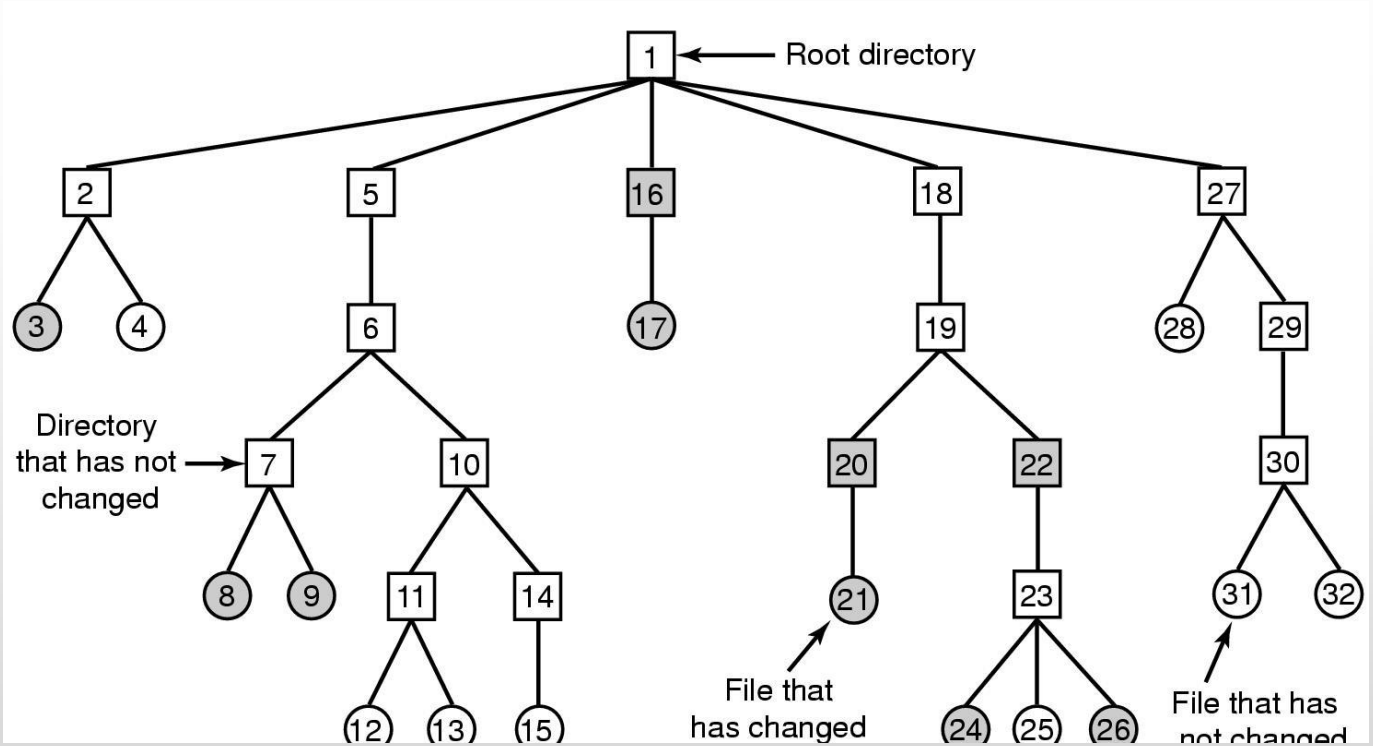
- Uygulaması kolay.
- Belirli bir dizini atlayamaz (*skip*).
- Kademeli dökümler yapamaz (*incremental*).
- Dosyalar tek tek geri yüklenemez.
- Mantıksal döküm stratejisi daha yaygın kullanılır.

Döküm Stratejileri - Mantıksal

- Verilen zamandan sonra değişen dosyaları/dizinleri özyinelemeli döker.
- Dosyaları/dizinleri değiştirilmiş dosya/dizine giden yola döker.
- Bu sayede yolu (*path*) farklı bir bilgisayarda geri yüklenebilir.
- Tek bir dosyayı geri yükleyebilir.

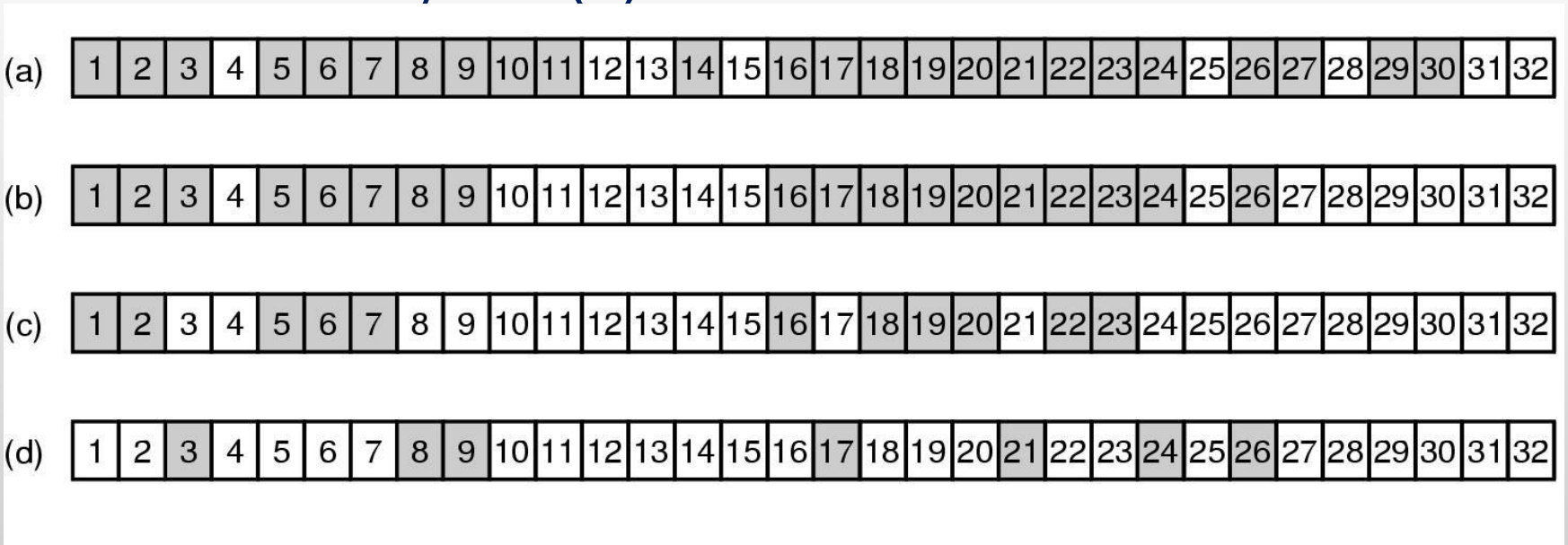
Dosya Sistemi Yedekleme

- Kareler dizinleri, daireler dosyaları gösterir. Gölgeli öğeler, son dökümden bu yana değiştirilenler. Her dizin ve dosya, i-node numarasıyla etiketlenir.



Mantıksal Döküm Algoritması

- i-node tarafından indekslenmiş biteşlem kullanılır.
- Aşama 1: kökte başlar ve değiştirilen tüm dosyalar için bitleri işaretler (a).
- Aşama 2: ağacı gezer, değişiklik olmayan dizinlerin işaretini kaldırır (b).
- Aşama 3 - i-node'ları gözden geçirir ve işaretli dizinlerin dökümünü alır (c).
- Aşama 4 - döküm dosyaları (d).



Diskteki Dosyayı Geri Yükleme

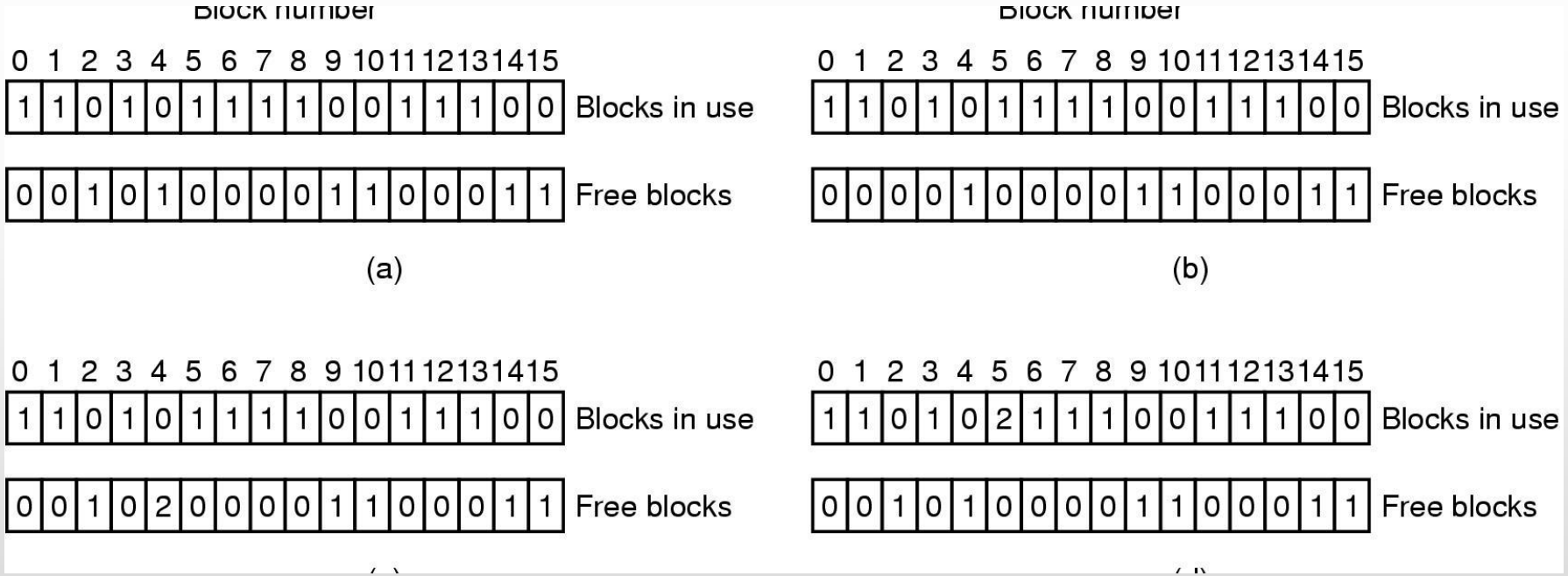
- Diskte boş dosya sistemi ile başlanır.
- Son alınan tam döküm geri yüklenir.
- Önce dizinler, sonra dosyalar.
- Ardından kademeli olarak dökümler geri yüklenir.

Dosya Sistemi Tutarlılığı

- Blokların tümü yazılmadan önce kilitlenme/kaza (crash),
  - Dosya sistemini tutarsız bir durumda bırakır.
- Tutarlılığı kontrol etmek için yardımcı programlara ihtiyaç var.
  - Unix'te *fsck*, Windows'ta *scandisk.*
- Bir dosyada bir blok kaç kez bulunur?
- Bir blok boş alanların tutulduğu listede kaç kez var?
- Cihaz tüm i-node'ları okur, sayaçları artırır.

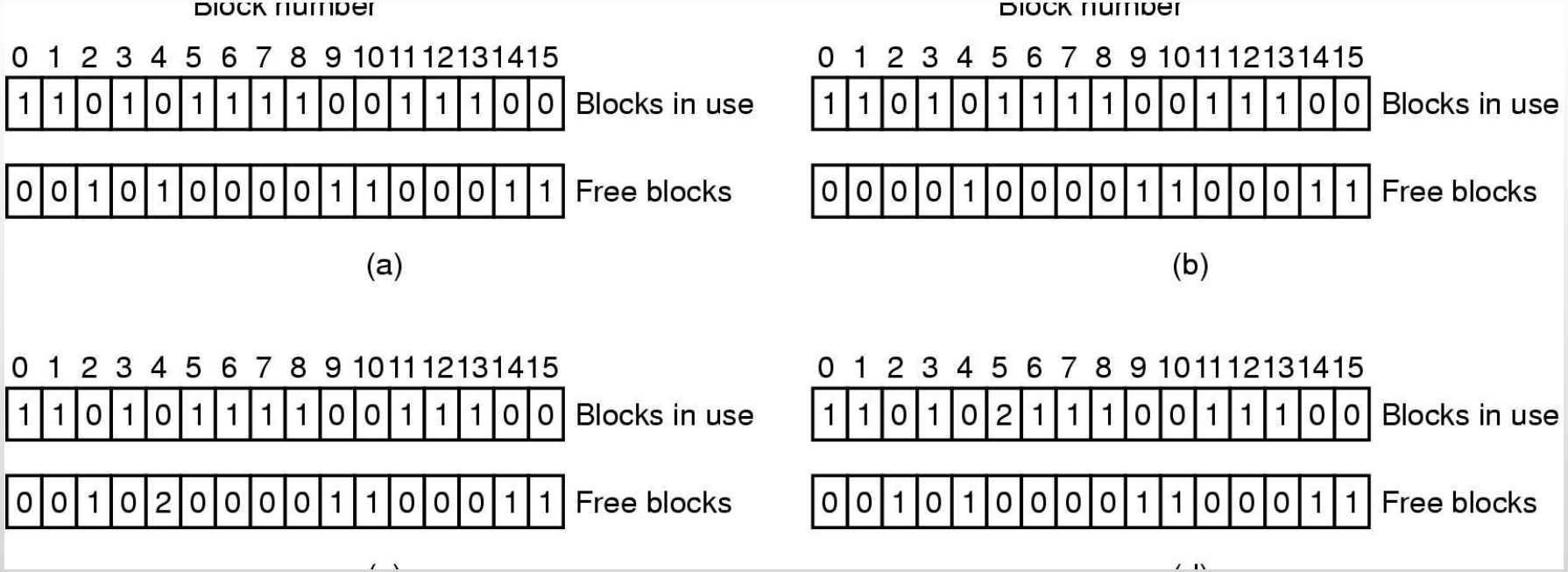
Dosya Sistemi Tutarlılığı

- (a) Tutarlı. (b) Eksik blok. (c) Yinelenen blok. (d) Yinelenen veri bloğu.



Dosya Sistemi Tutarlılığı Çözüm

- Eksik blok (b) - serbest listeye konulur.
- Yinelenen blok (c) - serbest liste yeniden oluşturulur.
- Yinelenen veri bloğu (d) - kullanıcı bilgilendirilir.



Dosya Sistemi Tutarlılığı

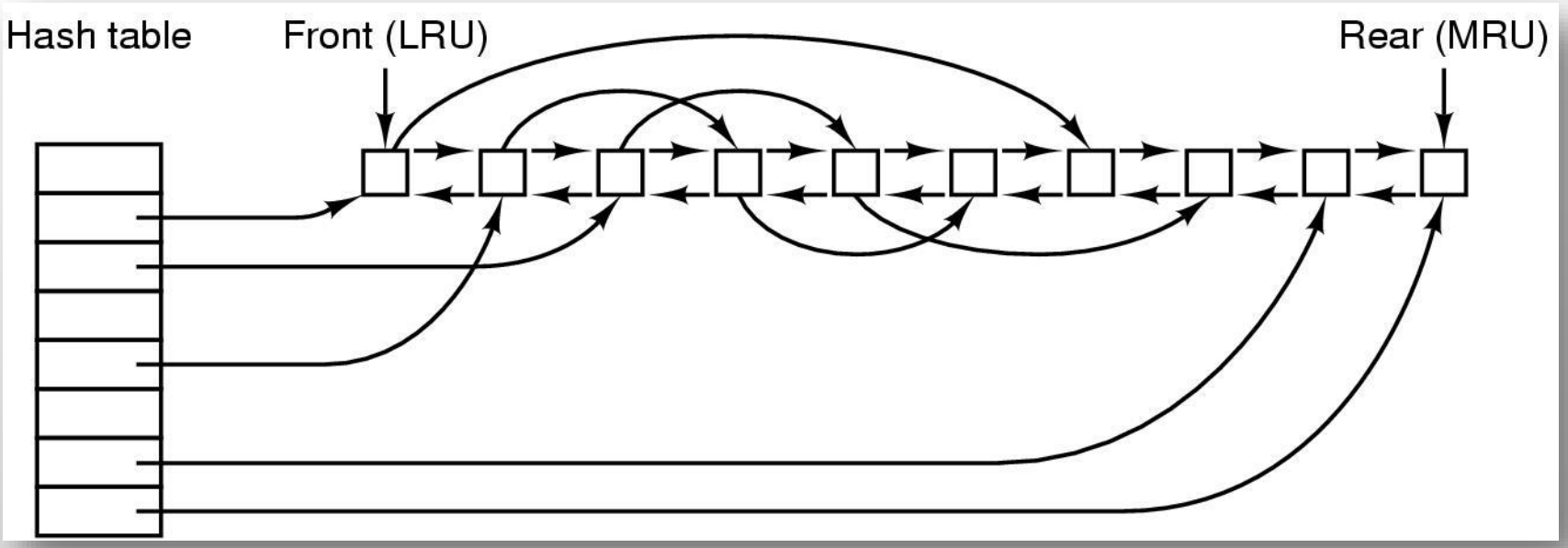
- Bloklar yerine dosyalara bakılır.
- Dosya başına bir sayaç tablosu kullanılır.
- Kök dizinde başlar, dosya bir dizinde göründüğünde sayaç artırılır.
- Sayaçlar i-node'lardan gelen bağ (*link*) sayısıyla karşılaştırılır.
  - Tutarlı olmak için aynı olmak zorunda.

Dosya Sistemi Performansı

- Bellekten sözcük (*word*) okuma: 32 *ns.*
- Disk erişimi: 5-10 *ms* arama (*seek*) + 100 *MB/sn* aktarım (*transfer*).
- Bellekte önbellek blokları tutulur.
- Yönetmek için *hash* tablosu (*cihaz, disk adresi*).
- Önbellek bloklarını değiştirmek için algoritmaya ihtiyaç var.
  - Sayfalama (*page replacement*) algoritmaları kullanılır.

Tampon Önbellek Veri Yapıları

- Buffer cache.



Yer Değiştirme

- Bazı bloklar nadiren kullanılsa da, bellekte olmalı.
- i-node değişiklik olduğunda diske yeniden yazılması gerekir.
  - Çökme durumunda, sistem tutarsız bir durumda kalabilir.
- Blok tekrar kullanılabilir mi?
- Blok, dosya sisteminin tutarlılığı için önemli mi?
- İhtiyaç duyulacak olanlar arka arkaya konur.
- *UNIX,* değiştirilmiş tüm blokları diske yazılmaya zorlar.
- *Windows*, blok değiştiğinde hemen diske yazar. (*Write through cache*)

İleriyi Okuma (Read Ahead)

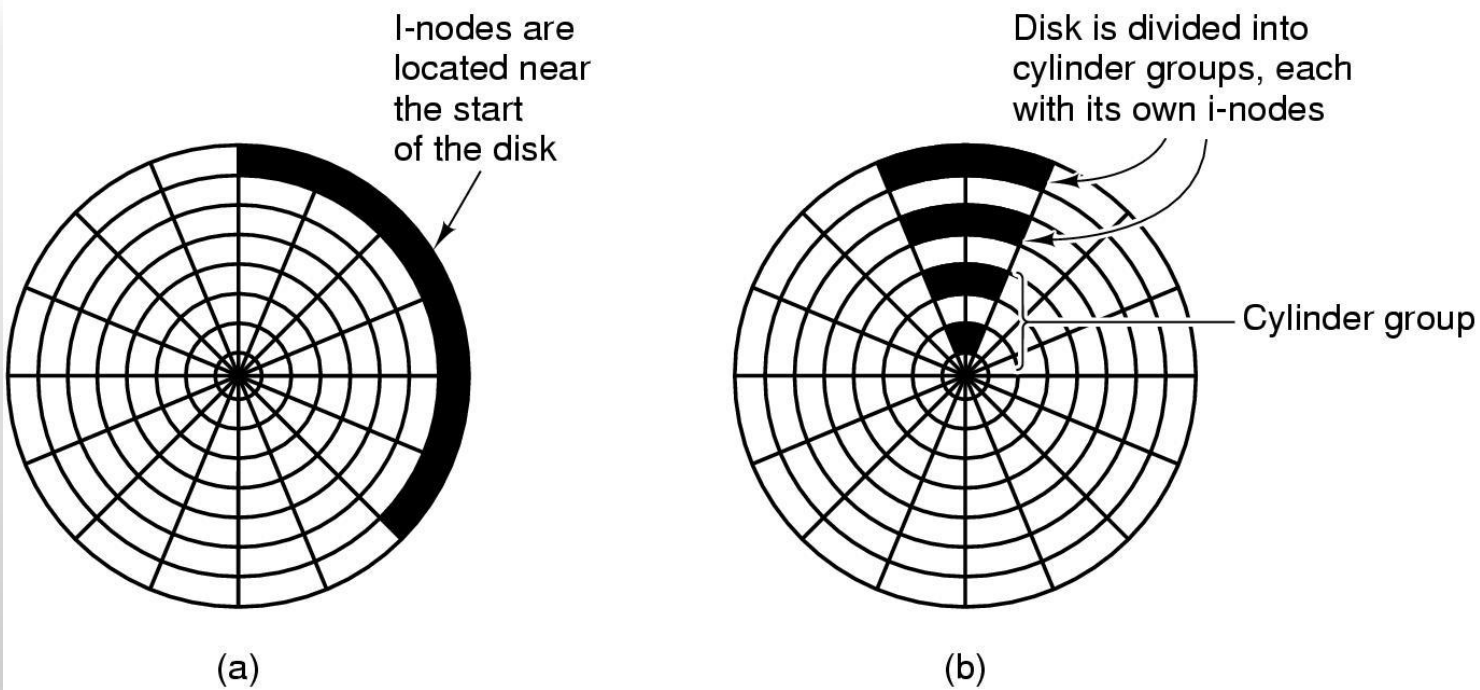
- Önbelleğe almak için *k* bloğu okunduğunda,
  - k+1 bloğu önbellekte değilse, onu da okur.
- Yalnızca sıralı dosyalar için geçerlidir.
- *Dosya sıralı mı, rastgele mi* olduğunu belirlemek için bir bit kullanılır.

Disk Kol Hareketini Azaltmak

- Sıralı olarak erişilecek bloklar birbirine yakın yerleştirilmeye çalışılır.
- Bellekte bir biteşlem tutarak yapmak kolaydır,
  - Blokları boş liste ile arka arkaya yerleştirmek gerekir.
- Önbellek bloğu 1 *KB* ise, yer tahsisi boşlar listesinden 2 *KB* halinde yapılır.
- Ardışık bloklar, aynı silindire konulmaya çalışılır.

Kol Hareketini Azaltmak

(a) Diskin başına yerleştirilen I-nodes. (b) Disk, her biri kendi blokları ve i-nodes'leri olan silindir gruplarına bölünmüştür.

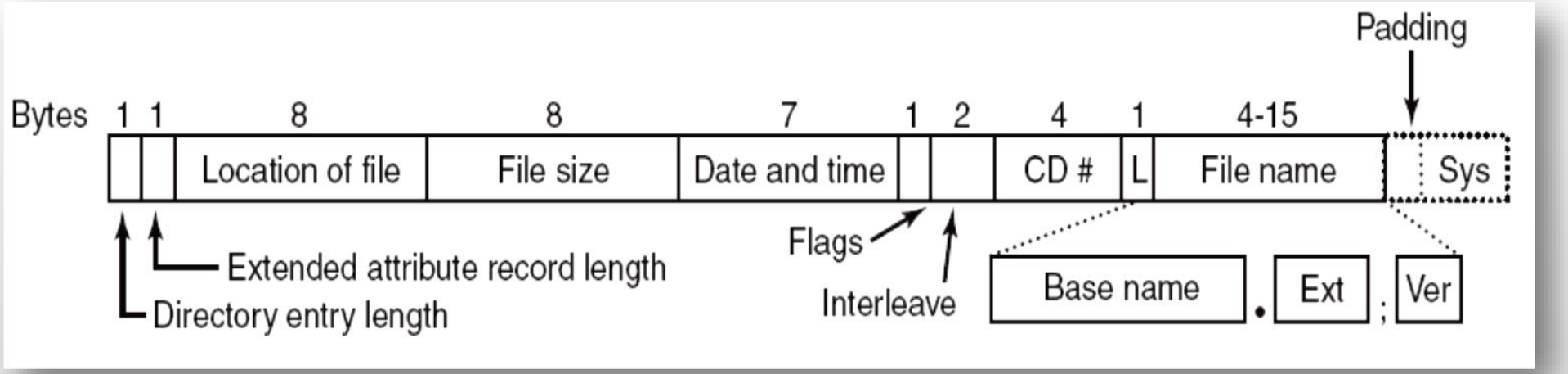


Diski Birleştirme (Defragmentation)

- Başlangıçta, dosyalar diske bitişik olarak yerleştirilir.
- Zamanla delikler, boşluklar (*hole*) oluşur.
- Windows
  - *defrag*, bir dosyanın farklı bloklarını bir araya getirir.
- Linux
  - *defrag* işlemini desteklemez.
  - Farklı dosyalar birbirine uzak yerleştirilir.

The ISO 9660 Dosya Sistemi

- Dizin örneği.



Rock Ridge Interchange Protokolü (RRIP)

- *CD-ROM*'larda kullanılır.
- Dosyalar hakkında ek bilgi sağlayan *ISO 9660* biçiminin bir uzantısı.
- Dosya sahipliği, izinler ve sembolik bağlantılar hakkında ek bilgiler sağlar.
- *CD-ROM*'ların *Unix* tabanlı sistemlerle uyumluluğunu geliştirir.
  - Uzun dosya adları için destek.
  - Unix tarzı sembolik bağlantılar için destek.
- Eski *CD-ROM* sürücüleriyle uyumsuz.

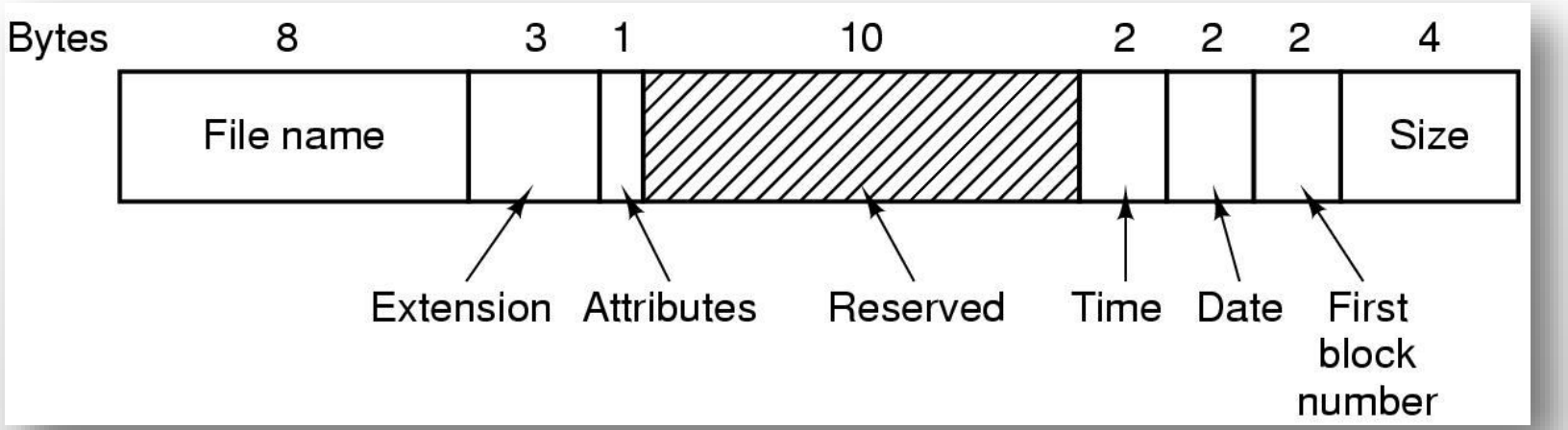
Rock Ridge Interchange Protokolü (RRIP)

- **PX** - POSIX attributes. POSIX öznitelikleri.
- **PN** - Major ve minor cihaz numaraları.
- **SL** - Symbolic link. Sembolik bağ.
- **NM** - Alternative name. Alternatif ad.
- **CL** - Child location. Çocuk konumu.
- **PL** - Parent location. Ebeveyn konumu.
- **RE** - Relocation. Yer değiştirme.
- **TF** - Time stamps. Zaman damgaları.

Joliet Uzantısı (Extension)

- *CD-ROM*'larda kullanılır.
- *Unicode* karakterlerini destekleyen *ISO 9660* biçiminin bir uzantısı.
- *CD-ROM*'ların Windows tabanlı sistemlerle uyumluluğunu geliştirir.
  - *Unicode* karakterleri ve uzun dosya adlarını destekler.
- Windows olmayan sistemlerde sınırlı destek.
- Eski *CD-ROM* sürücüleriyle uyum sorunları var.
- Bir dizin, sekiz seviyeden daha derine inebilir.
- Dizin adlarının uzantıları olabilir.

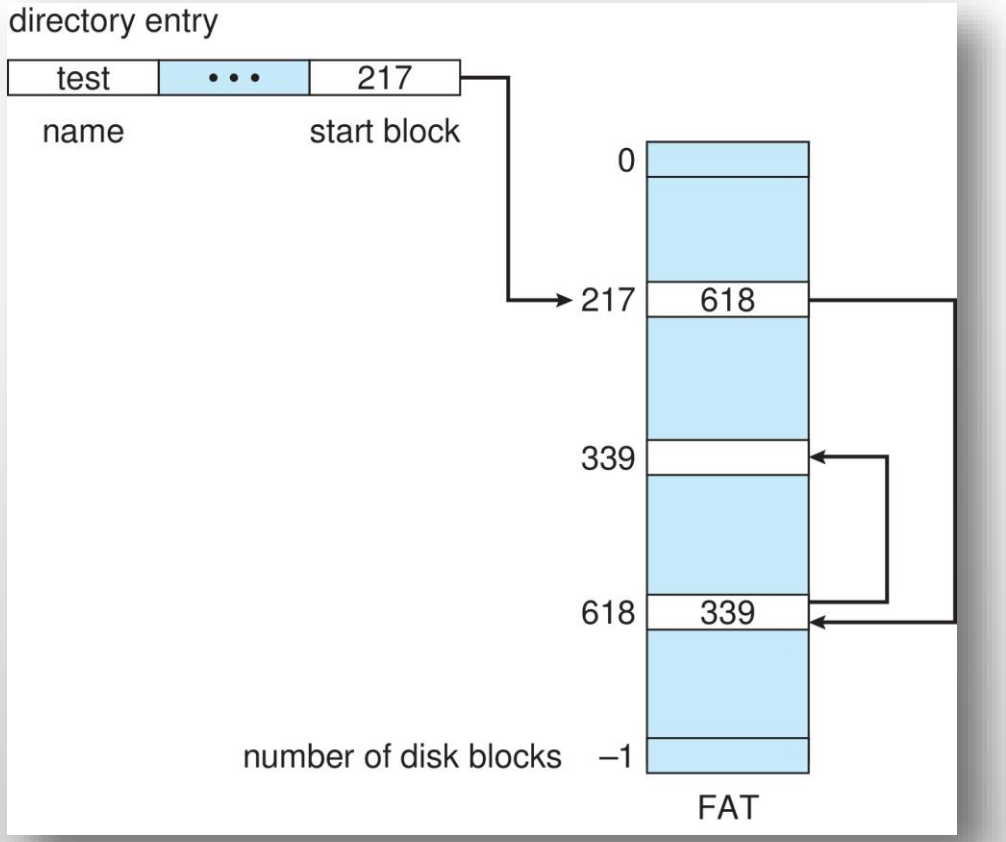
MS-DOS Dosya Sistemi Dizini



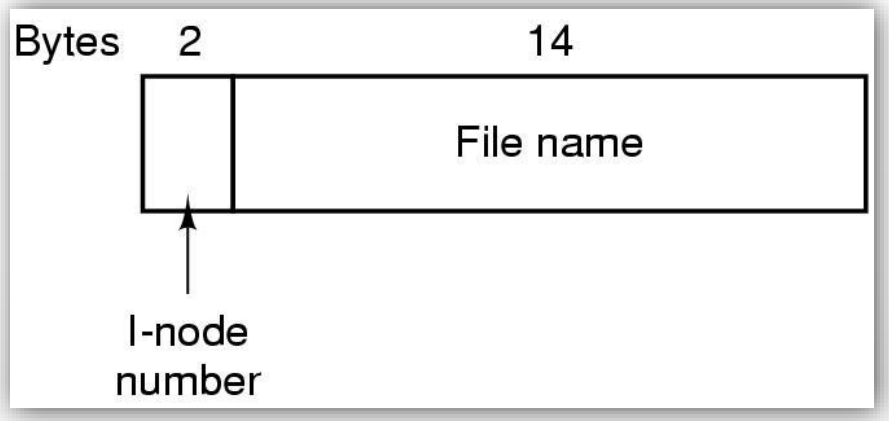
Blok Boyutları için Maksimum Bölüm Boyutu

Block size	FAT-12	FAT-16	FAT-32
0.5 KB	2 MB		
1 KB	4 MB		
2 KB	8 MB	128 MB	
4 KB	16 MB	256 MB	1 TB
8 KB		512 MB	2 TB
16 KB		1024 MB	2 TB
32 KB		2048 MB	2 TB

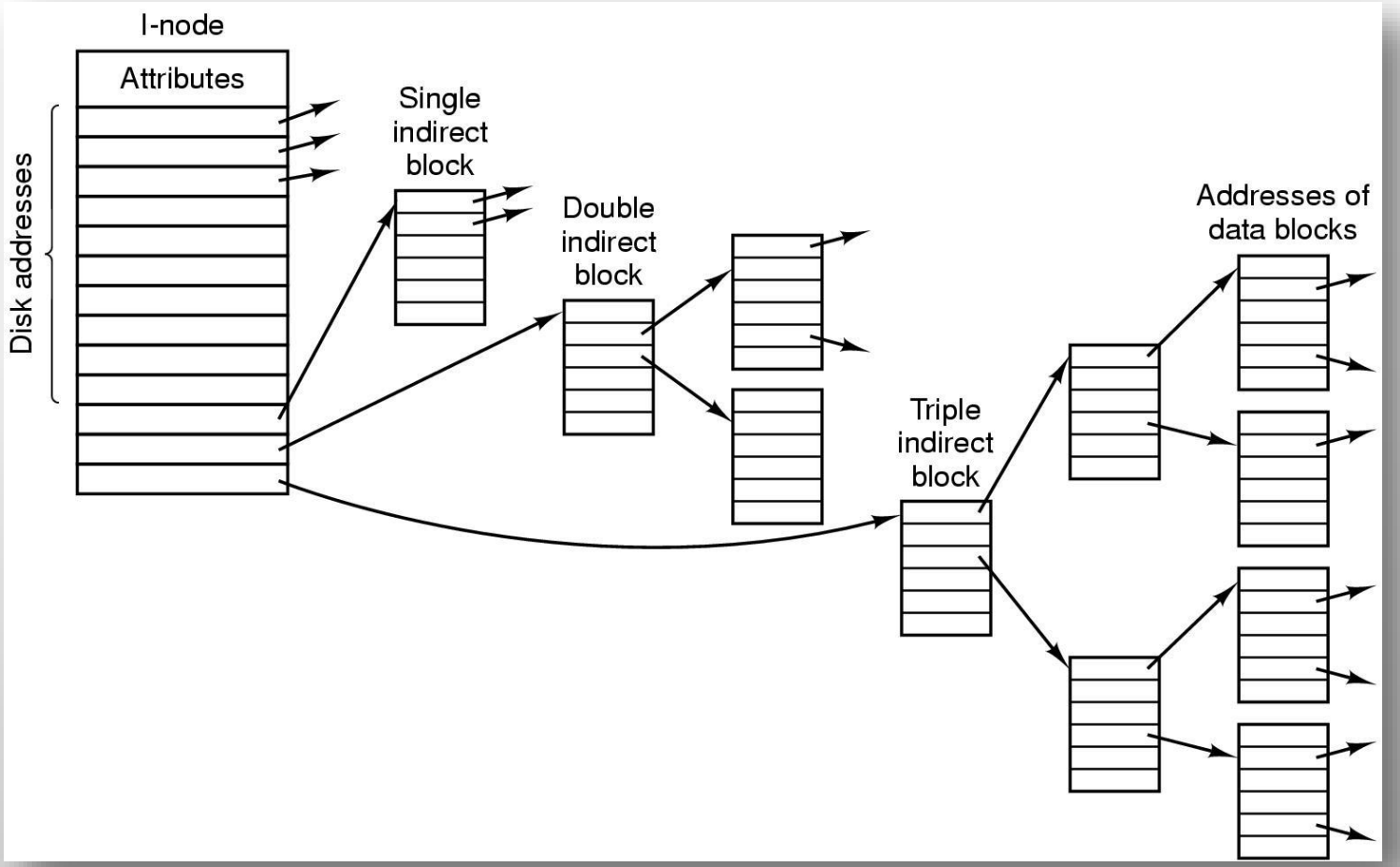
File Allocation Table



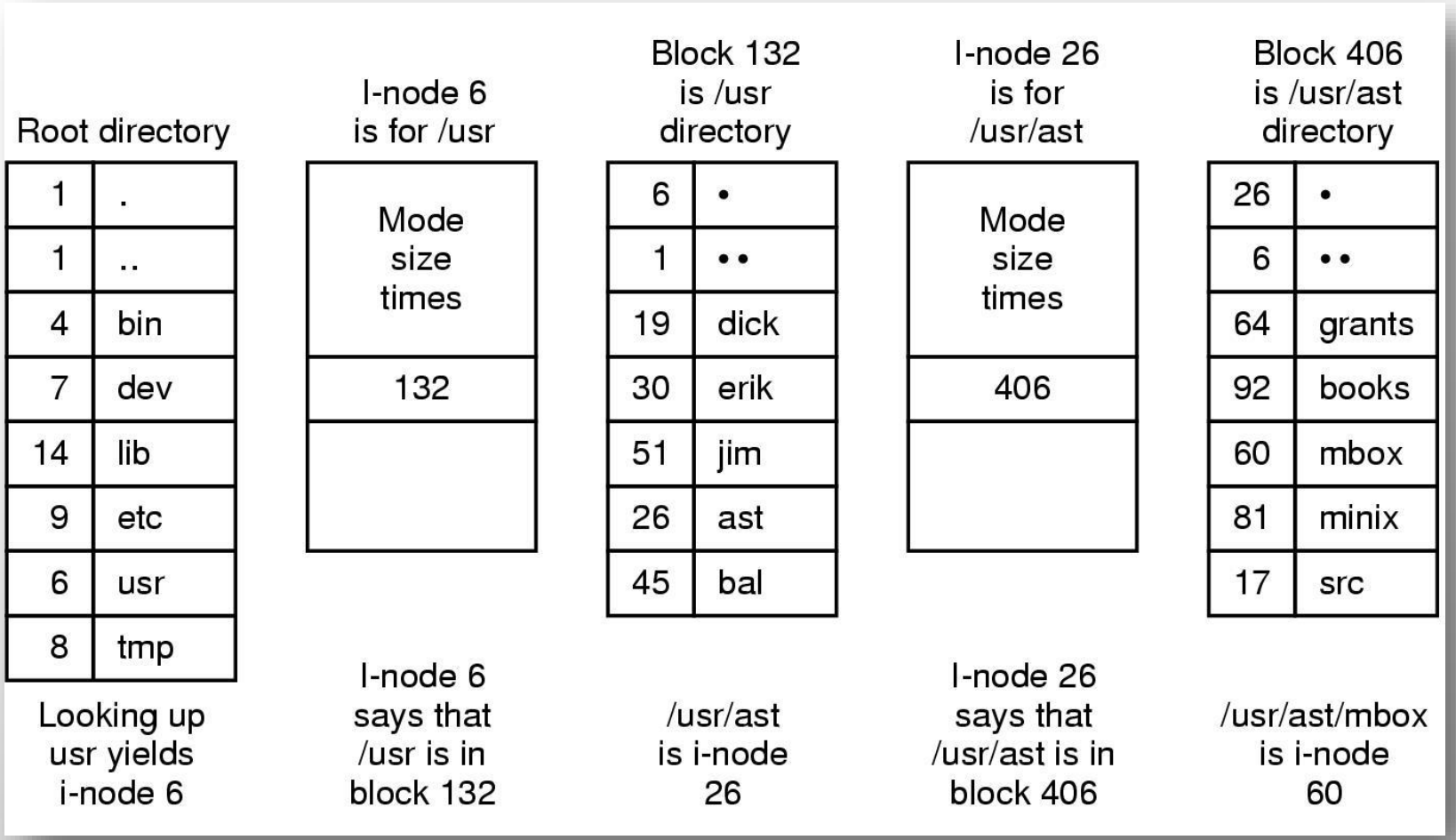
UNIX Version 7 Dizini



UNIX i-node



/usr/ast/mbox.conf Arama Adımları



SON