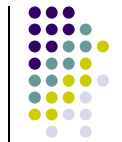


11

Sistem File

Tim Teaching Grant
Mata Kuliah Sistem Operasi



Sistem File

- Konsep File
- Metode Akses
- Struktur Direktori
- Mounting Sistem File
- File Sharing
- Proteksi

Konsep File

- Ruang alamat logik yang berdampingan
- Tipe :
 - Data
 - numeric
 - character
 - binary
 - Program



Aspek Sudut Pandang User

- Media penyimpanan data/informasi:
 - magnetic disk, magnetic tape, dan optical disk
- Untuk kenyamanan user
 - OS menyediakan sudut pandang logik yang sama bagi semua media dengan abstraksi unit-unit storage logik yaitu file



Aspek Isi dan Struktur



- Isi File
 - Representasi program atau data yang terekam dalam secondary storage
- Struktur file
 - Bebas maupun berformat
 - Secara umum file merupakan deretan bit, byte, baris, atau record yang artinya didefinisikan sendiri oleh user perancangnya

File Structure



- Urutan word, bytes
- Struktur record sederhana
 - Lines
 - Fixed length
 - Variable length
- Struktur kompleks
 - Formatted document
 - Relocatable load file
- Yang membuat keputusan :
 - Sistem operasi
 - Program

Atribut File



- **Name** –informasi yang disimpan untuk keperluan identifikasi form oleh pengguna
- **Type** – dibutuhkan sistem untuk mendukung tipe yang berbeda.
- **Location** – pointer ke lokasi file pada device
- **Size** – ukuran file yang sedang digunakan.
- **Protection** – kontrol terhadap pengguna yang sedang melakukan baca, tulis dan eksekusi.
- **Time, date, dan user identification** – proteksi data untuk pengamanan dan monitoring pengguna.
- Informasi yang disimpan file dalam struktur direktori untuk memudahkan pengelolaan disk.

Operasi-operasi File



- Enam operasi dasar yang berkaitan dengan manajemen file sistem:
 - Create file
 - Write file
 - Read file
 - Reposition dalam file
 - Delete file
 - Truncate file

Operasi-operasi File (cont.)



- Create file:
 - (1) menemukan free space; (2) entry baru dibuat dalam tabel direktori yang mencatat nama dan lokasi; serta (3) ukuran yang diinisialisasi 0
- Write file:
 - (1) OS melihat ke direktori untuk mencari lokasinya dalam disk; (2) melakukan transfer dari memori ke lokasi dalam disk (suatu pointer digunakan sebagai penunjuk lokasi penulisan berikutnya); dan (3) entry dalam direktori di update

Operasi-operasi File (cont.)



- Read file:
 - OS melakukan hal yang sama dengan penulisan file kecuali operasinya membaca dari lokasi dalam disk ke dalam memori
- Reposition dalam file:
 - (1) OS melihat ke direktori untuk mencari entry yang dimaksud, (2) pointer di set dengan harga (lokasi) tertentu yang diberikan
- Delete file:
 - (1) OS melihat ke direktori mencari entry dengan nama yang dimaksud; (2) kemudian membebaskan space yang teralokasi; (3) serta menghapus entry tsb
- Truncate file:
 - sama dengan menghapus file kecuali entry tidak dihapuskan tapi ukuran file diisi 0

Operasi-operasi File Lain



- Operasi-operasi lain pada dasarnya dilakukan dengan kombinasi operasi-operasi dasar tadi, contoh:
 - Append file
 - Rename file
 - Get atribut file
 - Set atribut file

Tipe File, Nama, Ekstensi



file type	usual extension	function
executable	exe, com, bin or none	read to run machine-language program
object	obj, o	compiled, machine language, not linked
source code	c, cc, java, pas, asm, a	source code in various languages
batch	bat, sh	commands to the command interpreter
text	txt, doc	textual data, documents
word processor	wp, tex, rrf, doc	various word-processor formats
library	lib, a, so, dll, mpeg, mov, rm	libraries of routines for programmers
print or view	arc, zip, tar	ASCII or binary file in a format for printing or viewing
archive	arc, zip, tar	related files grouped into one file, sometimes compressed, for archiving or storage
multimedia	mpeg, mov, rm	binary file containing audio or A/V information

Metoda Akses



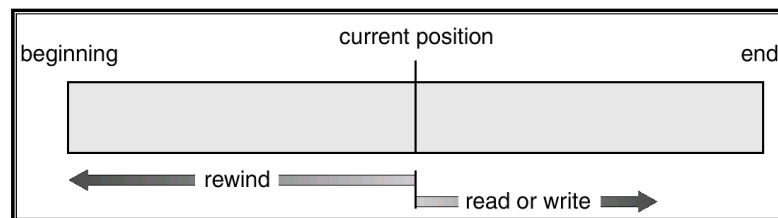
- Sequential Access
 - Akses dilakukan dengan satu arah pembacaan/penulisan (dari awal hingga akhir) jika ingin mundur maka perlu dilakukan rewind
- Direct Access (random access)
 - Akses dilakukan bisa pada posisi mana saja dalam file
- Metoda lain
 - Abstraksi lebih tinggi dari direct access
 - Index file & relative file
 - Tabel Informasi index: record dan pointer ke file direct access.

Metode Akses (cont.)

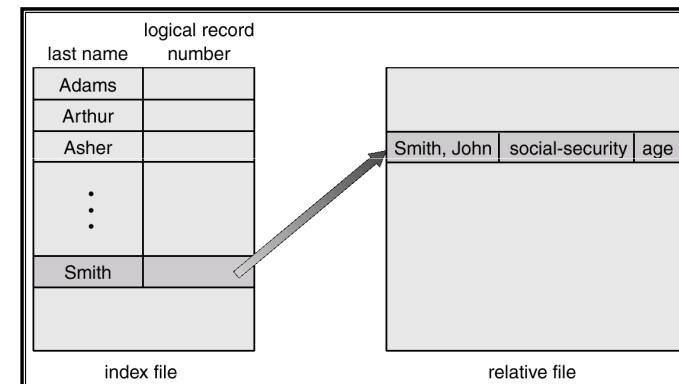


- Sequential Access
 - read next*
 - write next*
 - reset*
 - no read after last write (rewrite)*
 - Direct Access
 - read n*
 - write n*
 - position to n*
 - read next*
 - write next*
 - rewrite n*
- n = relative block number*

Akses File Sequential



Contoh Index and Relative Files

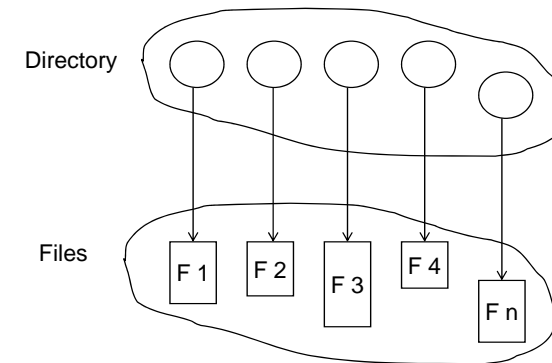


Struktur Direktori

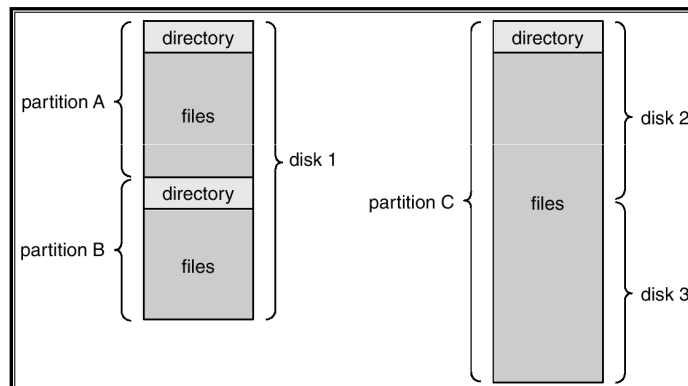


- Directory: kumpulan node yang berisi informasi dari semua file.
- Baik struktur direktori maupun file terletak di disk.
- Backup kedua struktur tersebut dapat disimpan pada tape.
- A collection of nodes containing information about all files.

Struktur Direktori (cont.)



Organisasi Sistem File



Informasi yang Ada pada Device Direktori



- Name
- Type
- Address
- Current length
- Maximum length
- Date last accessed (for archival)
- Date last updated (for dump)
- Owner ID (who pays)
- Protection information (discuss later)

Operasi Direktori

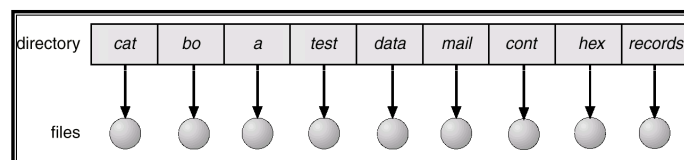
- Pencarian file
- Pembuatan file
- Penghapusan file
- Daftar directory
- Penggantian nama file
- Lintas sistem file

Organisasi Directory (Logik)

- **Efficiency** – menempatkan file secara cepat.
- **Naming** – kenyamanan pengguna
 - Dua pengguna dapat memberikan nama yang sama untuk file berbeda.
 - File yang sama dapat memiliki beberapa nama yang berbeda.
- **Grouping** – pengelompokan file secara logik logical grouping berdasarkan properti (contoh : semua program Java, semua games, ...)

Direktori Satu Tingkat

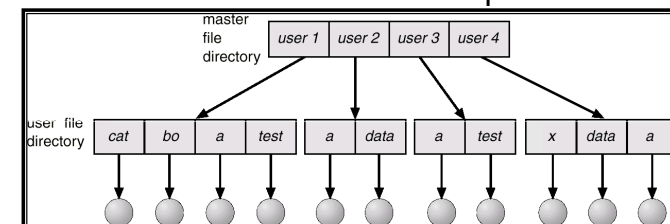
- Hanya ada direktori satu tingkat untuk semua user.



- Naming problem
- Grouping problem

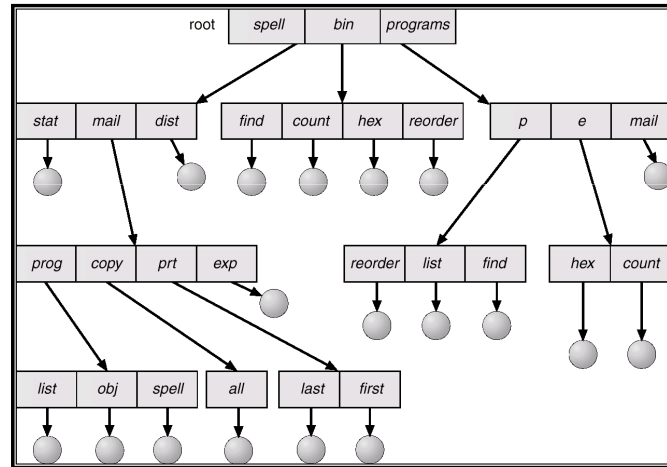
Direktori Dua Tingkat

- Pemisahan Direktori untuk setiap user.



- Path name
- Dapat memiliki nama yang sama untuk user berbeda
- Pencarian yang efisien
- Tidak perlu dikelompokkan

Direktori Struktur Tree



Bab 11. Sistem File

25

Direktori Struktur Tree (cont.)

- Pencarian yang efisien (Efficient searching)
- Menyediakan grouping
- Terdapat Current directory (working directory)
 - `cd /spell/mail/prog`
 - `type list`

Bab 11. Sistem File

26

Direktori Struktur Tree (cont.)

- **Absolute** atau **relative** path name
- Pembuatan file baru pada current directory.
- Delete a file

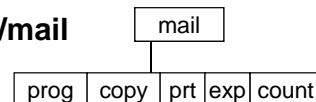
`rm <file-name>`

- Pembuatan subdirektori baru pada current directory.

`mkdir <dir-name>`

Contoh: jika current directory `/mail`

`mkdir count`



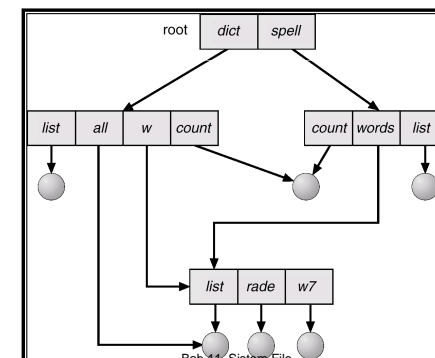
Deleting "mail" ⇒ deleting the entire subtree rooted by "mail".

Bab 11. Sistem File

27

Direktori Acyclic-Graph

- Saling berbagi (share) diantara subdirektori dan file



Bab 11. Sistem File

28

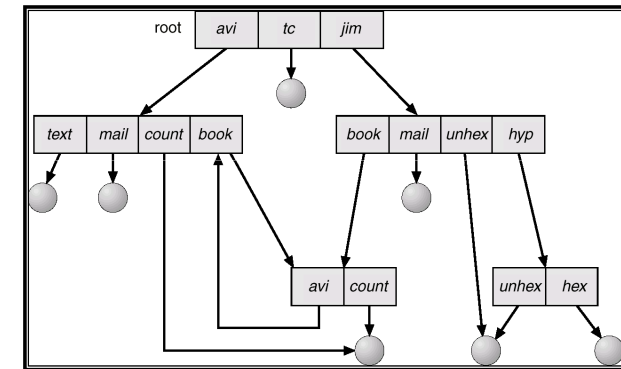
Direktori Acyclic-Graph (cont.)

- Dua nama berbeda (aliasing)
- Jika *dict* dihapus *list* \Rightarrow dangling pointer.

Solusi :

- Backpointer, kita dapat menghapus semua pointer. Masalahnya adalah pada ukuran record yang bervariasi.
- Backpointers menggunakan organisasi daisy chain.
- Solusi : Entry-hold-count.

Direktori General Graph

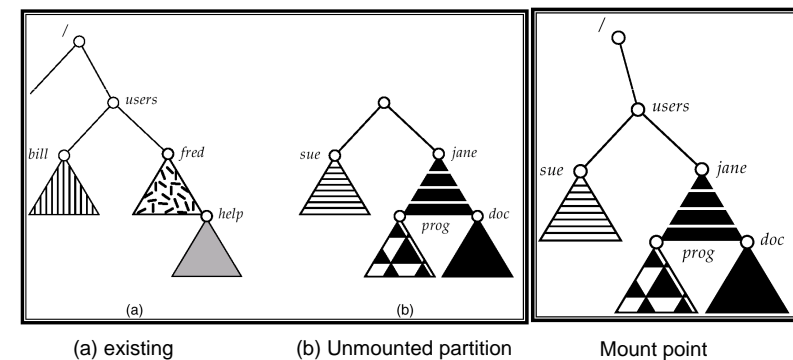


Direktori General Graph (cont.)

- Bagaimana kita menjamin tidak terjadi siklus ?
 - Mengizinkan link hanya pada file dan bukan pada subdirektori.
 - Mengumpulkan sampah (garbage).
 - Setiap kali terdapat link baru ditambahkan menggunakan algoritma penghapusan siklus apakah OK.

Mounting Sistem File

- Sistem file harus di-mount sebelum diakses.
- Sistem file yang tidak di mount akan di mount pada titik mount.



(a) existing

(b) Unmounted partition

Mount point

File Sharing



- Sharing file pada sistem multi user sangat diharapkan
- Sharing dapat dilakukan melalui skema proteksi.
- Pada sistem terdistribusi, file dapat di-share lintas jaringan.
- Network File System (NFS) adalah bentuk umum sharing file terdistribusi.

Proteksi



- Pemilik/pembuat file sebaiknya dapat mengendalikan :
 - File apa yang sedang dikerjakan
 - Siapa yang sedang bekerja menggunakan file
- Jenis Akses :
 - Read
 - Write
 - Execute
 - Append
 - Delete
 - List

File System Security



- Mekanisme proteksi:
 - Ide OS:
 - protection domain:
 - objek dan hak operasi
 - UNIX: uid (user id) dan gid (group id) semua objek file (direktori, I/O) : dikaitkan dengan id untuk hak operasi (read, write, execute)
 - proses dengan uid dan gid yang sama mempunyai hak yang sama untuk setiap objek dalam domain OS UNIX
 - Proses:
 - eksekusi: kode user dan kode kernel
 - SETUID: proses mendapatkan privilege sementara, efektif uid, gid berubah
 - Proteksi: dalam bentuk ring dan gate

Security implementation



- Access Control Lists (ACL)
 - setiap file (i-node) dikaitkan dengan otorisasi dan list hak operasi terhadap file dari berbagai proses
 - otorisasi: identifikasi pemakai, program, terminal etc
 - list dapat beragam: move, delete, open etc
 - UNIX: list: read, write, execute
- Capabilities:
 - setiap proses diasosiasikan dengan objek dapat digunakan:
 - operasi yang diizinkan (capability)
 - jenis objek (tipe), hak, pointer ke objek tsb (i-node)
 - dimana capability disimpan:
 - OS (kernel)
 - user space: disandikan

UNIX: proteksi



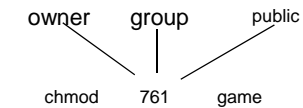
- Mode akses : read, write, execute
- Tiga kelas user

a) owner access	7	⇒	RWX 1 1 1
b) groups access	6	⇒	RWX 1 1 0
c) public access	1	⇒	RWX 0 0 1

Proteksi (cont.)



- Mintalah manager untuk membuat group baru (unique name), katakanlah G, tambahkan beberapa user pada group tersebut.
- Untuk particular file (katakanlah *game*) atau subdirectory, definisikan akses dengan tepat.



- Attach group ke file : **chgrp G game**

Implementasi Sistem File



- Penyimpanan dan pengaksesan file pada media secondary storage disk
- Mengalokasi space, merecover space yang dilepaskan, mencatat lokasi data, memperantarai bagian-bagian OS lain dengan secondary storage.

Struktur Sistem File



- Akibat efisiensi, transfer I/O antara memory & disk dilakukan dalam satuan-satuan block
 - satu block = satu atau beberapa sector
 - sector = bervariasi dari 32 byte hingga 4096 byte (umumnya =512 byte)
- Karakteristik Disk: rewritable in place & direct access

Organisasi Sistem File



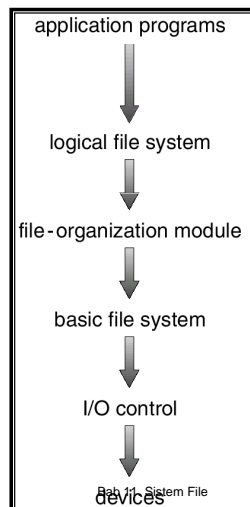
- Sistem file: menyediakan akses yang efisien dan nyaman ke disk
- Masalah-masalah rancangan:
 - penampakan bagi user: definisi dan atribut file, operasi file, struktur directori
 - Algoritma dan struktur data untuk memetakan sistem file logik ke secondary-storage device

Tingkatan /level dari Sistem File



- Sistem file logik: struktur direktori & nama file simbolis => interface
- Modul organisasi file: blok-blok file logik/fisik, manajemen blok-blok bebas => alokasi
- Sistem file dasar: menghasilkan perintah-perintah generik untuk read/write ke block fisik dalam disk
- I/O control: device driver & interrupt handler untuk menterjemahkan perintah-perintah generik ke perintah-perintah spesifik HW

Tingkatan /level dari Sistem File (cont.)



Tingkatan /level dari Sistem File (cont.)



- Contoh ketika program aplikasi memanggil sistem file logik untuk create file
 - Sistem file logik membaca direktori ybs ke memori, mengupdatenya dengan entry baru, dan menuliskan kembali ke dalam disk
 - Sistem file logik memanggil modul organisasi file untuk memetakan I/O direktori ke dalam nomor-nomor blok dari disk

Alokasi Storage Space



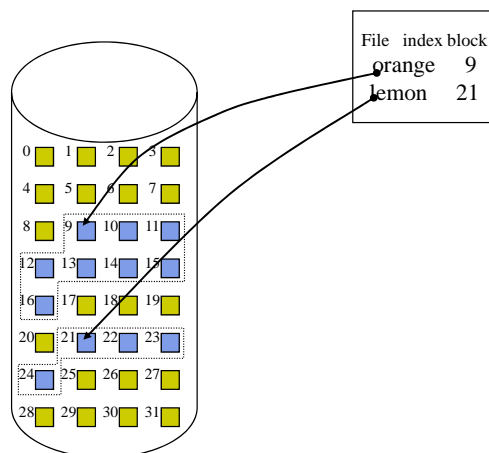
- Contiguous Allocation
- Linked Allocation
- Indexed Allocation

Contiguous Allocation



- Setiap file menempati sejumlah block yang beralamat contiguous dalam disk sehingga meminimisasi gerakan head antara pembacaan block
 - digunakan oleh IBM VM/CMS
 - digunakan oleh komputer mikro untuk floppy disk
- Akses mudah: sequential ataupun direct

Contiguous Allocation (cont.)



Contiguous Allocation (cont.)



- Masalah reliabilitas
 - harga pointer bisa berubah HW failure
 - Solusi: doubly linked list untuk menyimpan nama file dan nomor relatif block pada setiap block (penambahan overhead)
- Masalah jumlah akses memori yang besar
 - Solusi: skema FAT

Masalah Contiguous Allocation



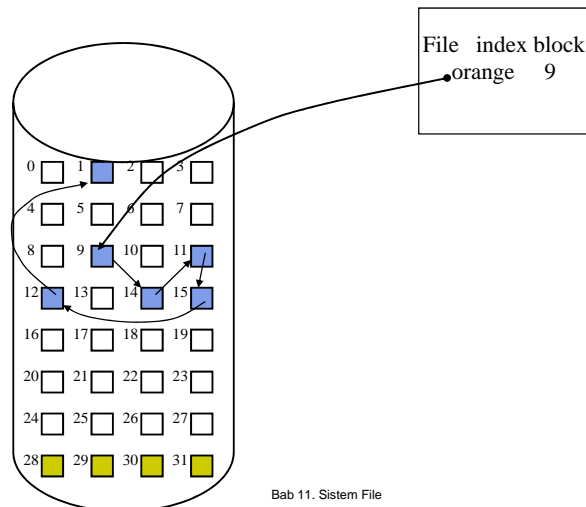
- Penemuan space: first fit atau best fit?
- Fragmentasi eksternal: perlu dilakukan kompaksi
- File output suatu proses tidak dapat diketahui, dan file bisa bertambah besar/kecil

Linked Allocation



- Setiap file menempati sejumlah block yang terangkai secara logik dalam linked-list
- Tidak ada masalah fragmentasi eksternal
- Kerugian
 - tidak efisien dalam penanganan direct access
 - Diperlukan ruang untuk pointer; contoh pointer 4 byte dalam block 512-byte adalah 0.78%
- Solusi: alokasi berbasis kluster

Linked Allocation



Linked Allocation



- Masalah reliabilitas
 - harga pointer bisa berubah HW failure
 - Solusi: doubly linked list untuk menyimpan nama file dan nomor relatif block pada setiap block (penambahan overhead)
- Masalah jumlah akses memori yang besar
 - Solusi: skema FAT

File Allocation Table (FAT)



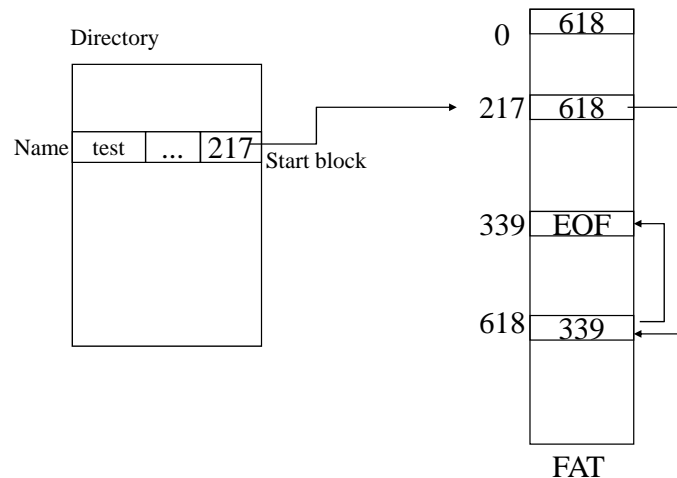
- Skema yang digunakan MS-DOS dan OS/2
- Menempati satu bagian dari disk (biasanya di awal partisi) berisikan satu entry untuk setiap block & berindex dengan nomor block
- Entry dari directory berisi nomor block pertama di dalam file

File Allocation Table (FAT) (cont.)



- Entry dari FAT berisikan nomor block berikutnya dalam file (block yang bebas dinyatakan dengan harga 0)
- Random access dioptimisasi dengan penelusuran nomor block di dalam FAT secara berantai
- Kerugian: head seek, kecuali FAT diload ke dalam cache

File Allocation Table (FAT) (cont.)

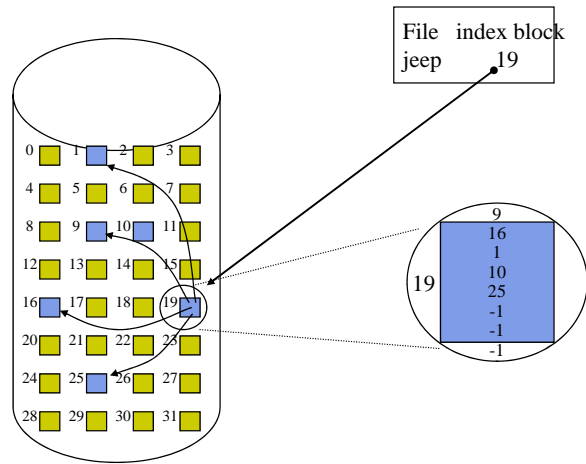


Indexed Allocation



- Setiap file menempati sejumlah block yang terdaftar di dalam suatu block yang berfungsi sebagai index block
- Index block => kumpulan pointer pada satu lokasi block
 - Pendekatan ini mendukung direct access yang lebih efisien
 - entry yang tidak digunakan diisi nil

Indexed Allocation



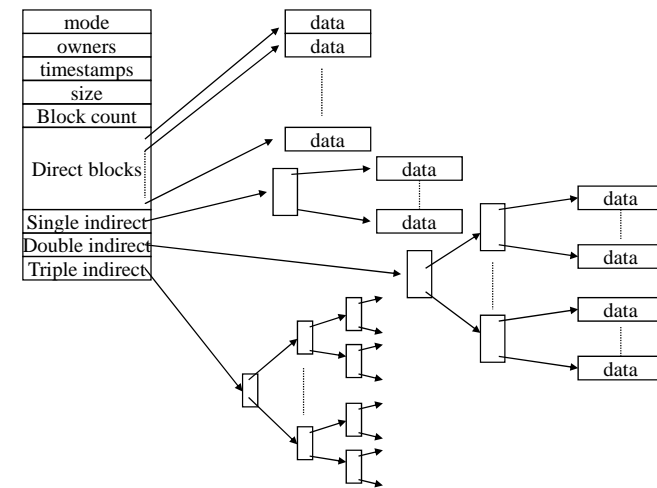
Indexed Allocation File Besar

- Linked scheme: menggunakan beberapa index block yang terangkai dengan pointer (entry terakhir dari tiap index block berisi pointer ke index block berikutnya)
- Multilevel index scheme: digunakan suatu indirect index block yang mendaftarkan semua direct index, jika lebih besar lagi maka terdapat beberapa level indirect block block
 - Contoh ukuran block 4K, pointer 32-bit, maka ada 1024 pointer dalam index block, maka untuk 2 layer index block dapat digunakan untuk 104876 block = 400 M
- Combined scheme, contoh BSD UNIX

BSD UNIX System

- Suatu entry directori menunjuk ke suatu inode dari file yang berisikan
 - atribut-atribut file: owner, timestamp, ukuran, jumlah block
 - 12 pointer direct block ke data data
 - 1 pointer single indirect (1-level) index block
 - 1 pointer double indirect (2-level) index block
 - 1 pointer triple indirect (3-level) index block

BSD UNIX System (cont.)



Disk space management



- Berkas disimpan dalam satuan per blok
- Blok disk
 - besarnya blok (block size) tetap : 512 bytes - 8 Kbytes
 - blok size terlalu besar: space yang terbuang (i.e besarnya bekas rata-rata di UNIX : 1 Kbytes)
- Free blocks:
 - Bagaimana melacak blok yang tidak digunakan?
 - linked list dari nomor blok yang bebas (tabel besar jika disk masih kosong)
 - bit-map (jika terdapat n blok disk maka diperlukan n bits); blok yang bebas diwakili oleh bit 1, blok yang digunakan oleh bit 0

Bab 11. Sistem File

61

Disk management (issues)



- Disk quota (multi-user)
- Keandalan:
 - manajemen bad block
 - backup
 - konsistensi:
 - blok : membandingkan list blok digunakan dan yang bebas
 - direktori/file: membandingkan entry direktori dan i-node/awal blok
 - file system check (fsck: UNIX, scandisk: DOS)
- Unjuk kerja:
 - cache:
 - write-through cache (MS-DOS)
 - write-back/delay cache (UNIX)

Bab 11. Sistem File

62

Implementasi Direktori



Implementasi Direktori



- Linear list pada nama file dengan pointer pada blok data.
 - Sederhana untuk program
 - Memakan waktu yang lama untuk eksekusi
- Hash Table – linear list dengan struktur hash data.
 - Mengurangi waktu cari direktori
 - *collisions* – situasi dimana dua nama file yang di hash pada lokasi yang sama.
 - Ukurannya tetap.

Bab 11. Sistem File

63

Bab 11. Sistem File

64

Efisiensi dan Unjuk Kerja



- Efisiensi tergantung pada :
 - Alokasi disk dan algoritma direktori
 - Tipe data yang dilindungi pada direktori file
- Unjuk Kerja
 - disk cache – memisahkan bagian main memori yang sering digunakan sebagai blok.
 - Teknik free-behind dan read-ahead – digunakan untuk optimasi akses sequensial
 - Meningkatkan unjuk kerja PC dengan menggunakan bagian memori seperti virtual disk atau RAM disk.

Recovery

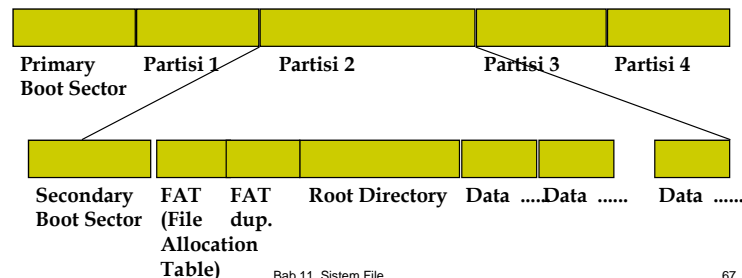


- Pemeriksaan rutin – membandingkan data pada struktur direktori dengan blok data pada disk dan mencoba memperbaiki ketidakkonsistenan.
- Menggunakan sistem *back up* data dari disk ke *storage device* lain (floppy disk, magnetic tape).
- Recovery file yang hilang dengan me-restore data dari *back up*

File System (implementasi)



- MS-DOS (versi: 3.x - 6.x)
 - direktori hirarkis : root, sub-direktori
 - alokasi: linked list (memori) dengan tabel indeks
 - tidak mendukung : proteksi (i.e owner), link (simbolik)
- Sistem berkas PC: hard disk



PC disk layout



- Boot Sector
 - informasi mengenai sistem berkas, struktur disk:
 - parameter: jumlah byte per-sector, jumlah sektor per blok, besarnya root direktori (dibaca pertama kali)
 - tabel partisi: awal dan akhir suatu partisi, setiap partisi dapat mempunyai sistem berkas berbeda,
 - partisi aktif: terdapat kode untuk menjalankan sistem
 - boot-strap: kode awal untuk menjalankan sistem
 - CPU: menjalankan kode ROM (instruksi 0)
 - ROM: load boot sector di memori (sektor 0)
 - transfer ke awal dari kode (jump) : bootstrap (MS-DOS)
 - baca root direktori: load io.sys dan msdos.sys

PC disk layout (FAT)



- File Allocation Table:
 - informasi pemakaian disk
 - duplikasi (back-up)
 - blok disk: 1 sektor - 8 sektor; tergantung besarnya disk
 - entry table: informasi 1 blok disk, besarnya indeks 16 bits (64 ribuan); i.e nomor entry = nomor blok disk
 - isi entry: free atau digunakan bagian indeks list
- FAT & prosedur open file:
 - MS-DOS : mengambil file descriptor yang tidak digunakan
 - cari slot kosong di system file table => indeks disimpan oleh byte pertama pada file descriptor
 - open direktori (path name) dari file
 - ambil nomor awal dari FAT, linked ke blok berikutnya dst.