PERTEMUAN 10:

VIRTUAL MEMORI

A. TUJUAN PEMBELAJARAN

Pada bab ini akan dijelaskan mengenai virtual memori, Anda harus mampu:

1.1 Definisi Virtual Memori

1.2 Konsep Dasar

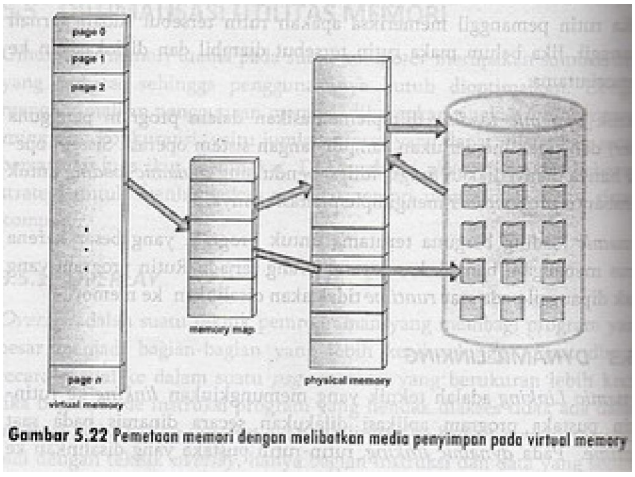
1.3 Algoritma

B. URAIAN MATERI

Tujuan Pembelajaran 1.1:
Virtual Memori

Definisi virtual memori :

- Adalah teknik pemetaan memori yang melibatkan memori sekunder, umumnya disk.
- Secara sistem logika, ukuran memori lebih besar daripada ukuran memori utama secara fisik.
- Melibatkan mekanisme swapping



Gambar 5.22 Pemetaan memori dengan melibatkan media penyimpan pada virtual memory

Keuntungan model virtual memori

- Lebih sedikit memori yang diperlukan per proses.
- System response menjadi lebih cepat, karena tidak semua bagian image proses perlu dialokasikan ke memori utama,

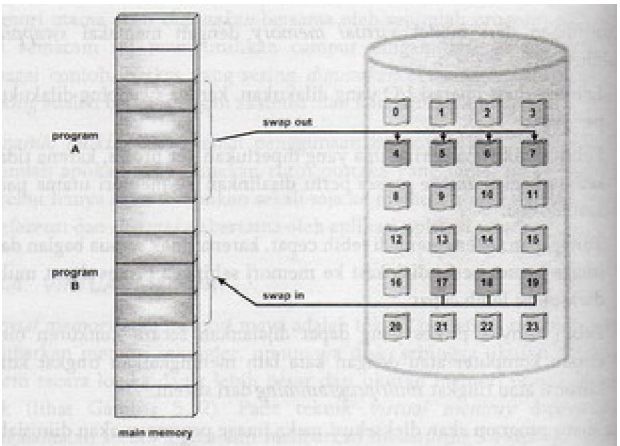
à proses dapat dieksekusi lebih cepat.

- Lebih banyak proses yang dapat dijalankan

à multiprogramming

Konsep Dasar

- Image proses akan diinisialisasi di area swap space, yaitu suatu lokasi di media penyimpan sebagai ekstensi memori utama.
- Swap space dapat berupa suatu berkas atau partisi khusus, dan unit terkecilnya disebut page.
- Pengalihan page dari swap space ke memori utama menggunakan teknik lazy swapper, yaitu hanya page proses yang dibutuhkan yang akan dialihkan ke memori utama.



Gambar 5.23 Pemetaan media penyimpan sebagai senarai berurut pada virtual memory

Bagaimana mengetahui page mana yang masih di swap space dan yang ada di memori utama ?

Mekanisme demand paging

Konsep dasar:

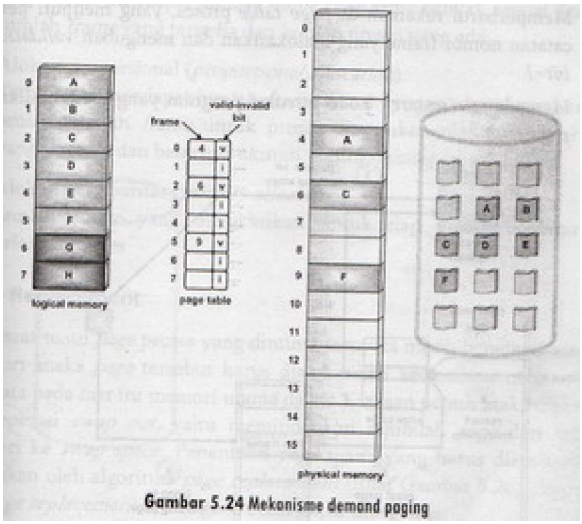
- Jumlah frame di memori utama tergantung tingkat multiprogramming .

  semakin tinggi tingkat multiprogramming, semakin sedikit jatah frame untuk tiap proses
- Menggunakan bit valid/invalid di page table

  misal: bit 1 à berada di memori utama

  bit 0 à berada di swap space

- Jika berstatus invalid, maka trap page fault akan dibangkitkan agar ditangani lebih lanjut oleh rutin SO yaitu: page fault handler.
- Rutin page fault handler akan menangani operasi swap-in terhadap page yang diperlukan.



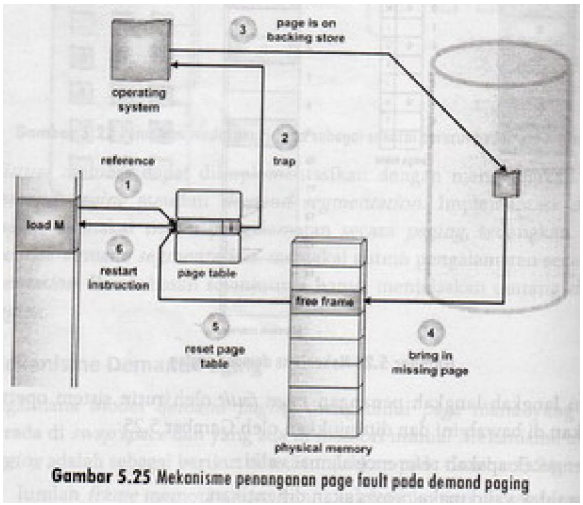
Gambar 5.24 Mekanisme demand paging

Langkah-langkah swap-in :

- Mencari frame memori utama yang kosong.

  jika tidak ada à dipilih salah satu page dalam frame (victim page) untuk di swap-out.
- Swap-in
- Memperbarui rekaman di page table

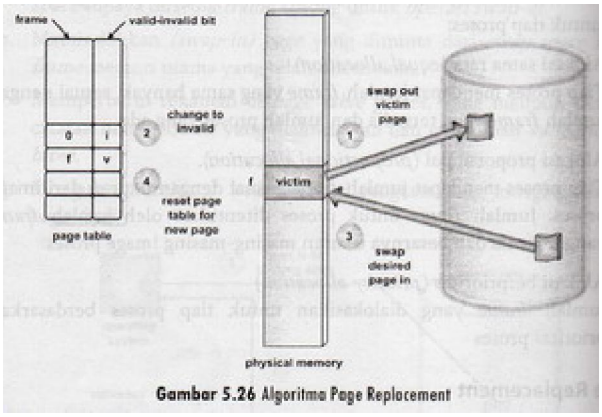
  à mengubah validation=1
- Restart.



Gambar 5.25 Mekanisme penanganan page fault pada demand paging

Secara umum, algoritma dapat dibagi dua:

- Global Replacement
  - Victim frame dapat dipilih dari semua frame yang ada
- Local Replacement
  - Victim frame dapat dipilih dari frame-frame yang sedang ditempati oleh image proses bersangkutan



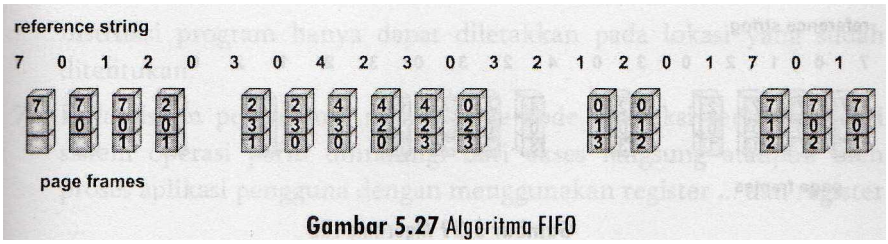
Gambar 5.26 Algoritma Page Replacement

Algoritma page replacement:

- Algoritma FIFO (First In First Out)
- Algoritma Optimal
- Algoritma LRU (Least Recently Used)

Algoritma FIFO

Page yang diganti adalah page yang paling lama berada di memori atau yang pertama kali masuk.

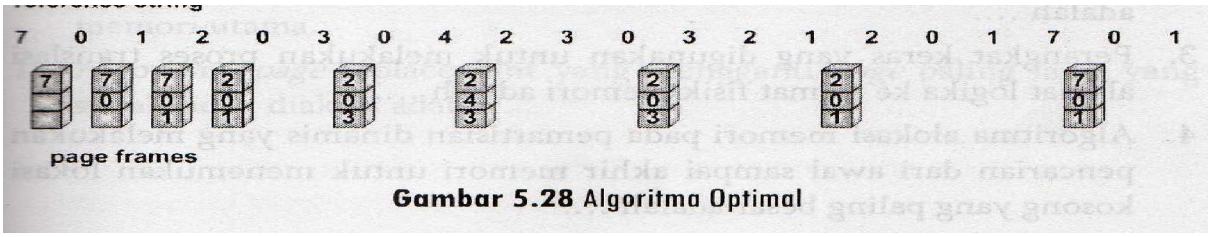


Gambar 5.27 Algoritma FIFO

Algoritma Optimal

a. page yang diganti adalah page yang baru akan dipanggil lagi pada waktu yang masih lama.

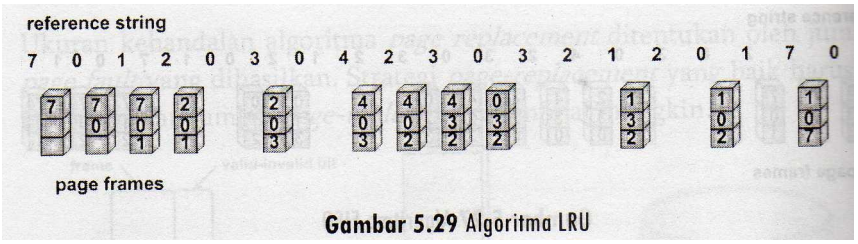
b. Diasumsikan sistem mampu memprediksi page-page yang akan diakses



Gambar 5.28 Algoritma Optimal

Algoritma LRU

Page yang diganti adalah page yang paling lama sudah tidak diakses.



Gambar 5.29 Algoritma LRU

Seberapa banyak jumlah frame yang butuh dialokasikan ke suatu proses yang berjalan?

- Pengalokasian tiap-tiap proses bervariasi tergantung pada tingkat multiprogramming
- Jika tingkat multiprogramming nya semakin tinggi, maka proses akan kehilangan beberapa frame
- Sebaliknya jika tingkat multiprogramming berkurang, maka proses akan mendapat frame melebihi dari yang dibutuhkan.

Alokasi Frame

- Alokasi sama rata (equal allocation)
  - Tiap proses mendapat jumlah frame sama banyak
- Alokasi proporsional (proporsional allocation)
  - Tiap proses mendapat jumlah frame sesuai dengan besarnya image proses itu.

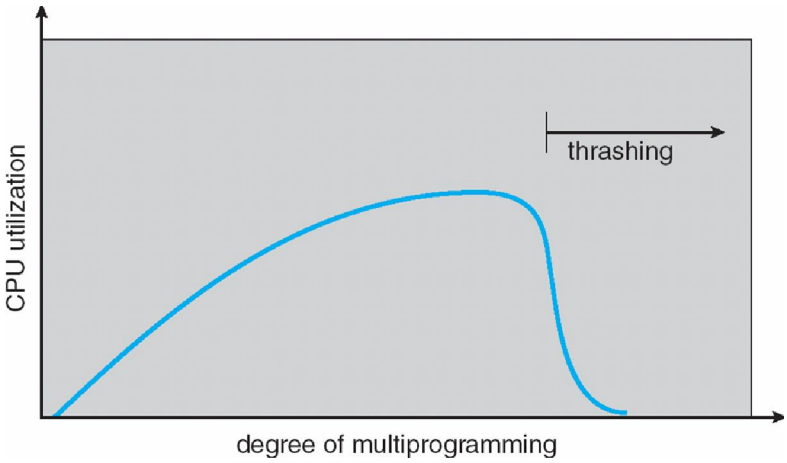
- Alokasi berprioritas (priority allocation)
  - Jumlah frame yang dialokasikan untuk tiap proses berdasarkan prioritas.

Trashing

If a process does not have “enough” pages, the page-fault rate is very high. This leads to:

- low CPU utilization
- operating system thinks that it needs to increase the degree of multiprogramming
- another process added to the system

Thrashing is a process is busy swapping pages in and out



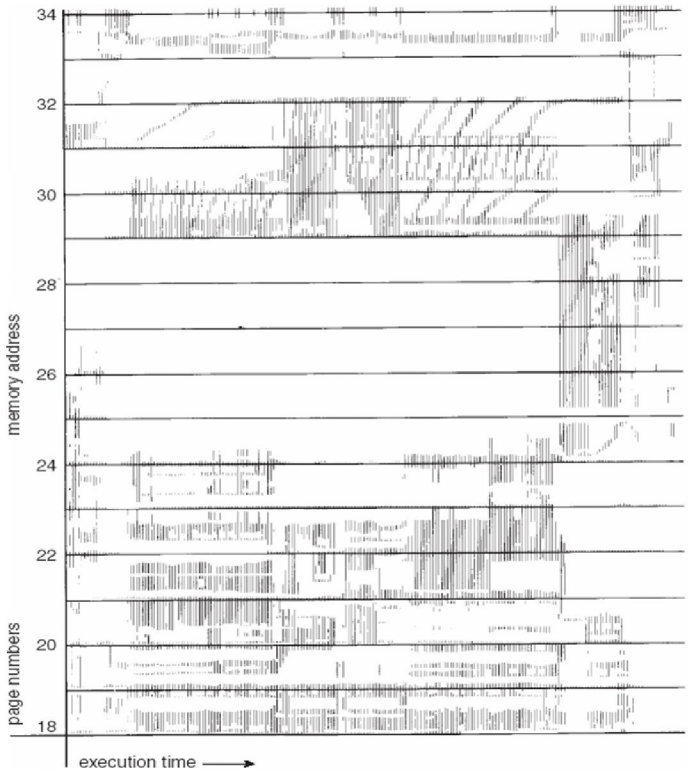
Demand Paging and Thrashing

- Why does demand paging work?

  Locality model
  - Process migrates from one locality to another
  - Localities may overlap
- Why does thrashing occur?

  Σ size of locality > total memory size

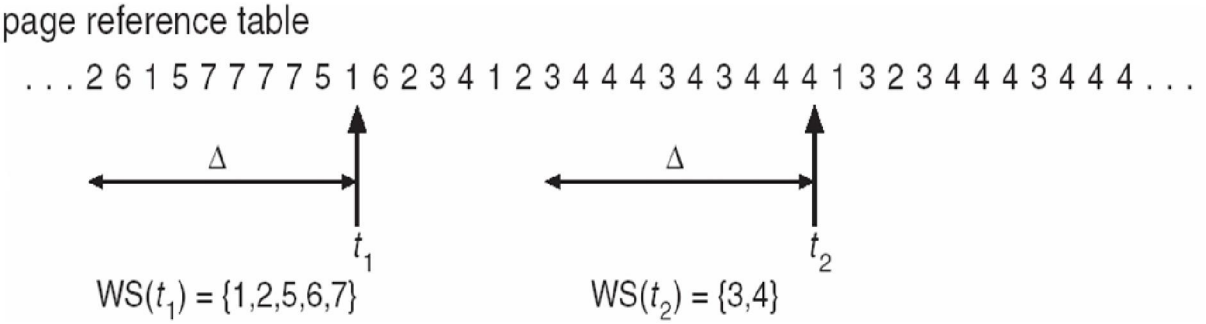
Locality In A Memory-Reference Pattern



Working-Set Model

- $\Delta \equiv$ working-set window $\equiv$ a fixed number of page references
  Example: 10,000 instruction
- WSS_i (working set of Process P_i) = total number of pages referenced in the most recent Δ (varies in time)
  - if Δ too small will not encompass entire locality
  - if Δ too large will encompass several localities
  - if $\Delta = \infty \Rightarrow$ will encompass entire program
- $D = \Sigma WSS_i \equiv$ total demand frames
- if $D > m \Rightarrow$ Thrashing
- Policy if D > m, then suspend one of the processes

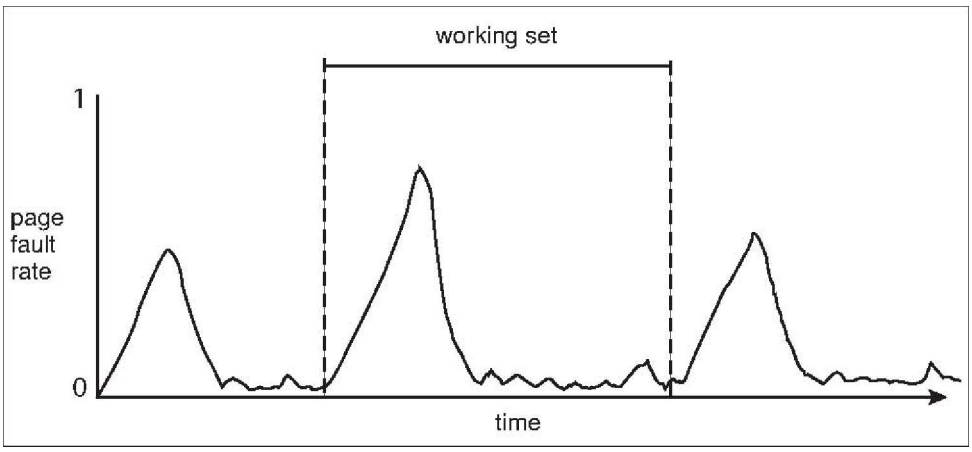
Working-set model



Keeping Track of the Working Set

- Approximate with interval timer + a reference bit
- Example: Δ = 10,000
    - Timer interrupts after every 5000 time units
    - Keep in memory 2 bits for each page
    - Whenever a timer interrupts copy and sets the values of all reference bits to 0
    - If one of the bits in memory = 1 ⇒ page in working set
- Why is this not completely accurate?
- Improvement = 10 bits and interrupt every 1000 time units

Working Sets and Page Fault Rates

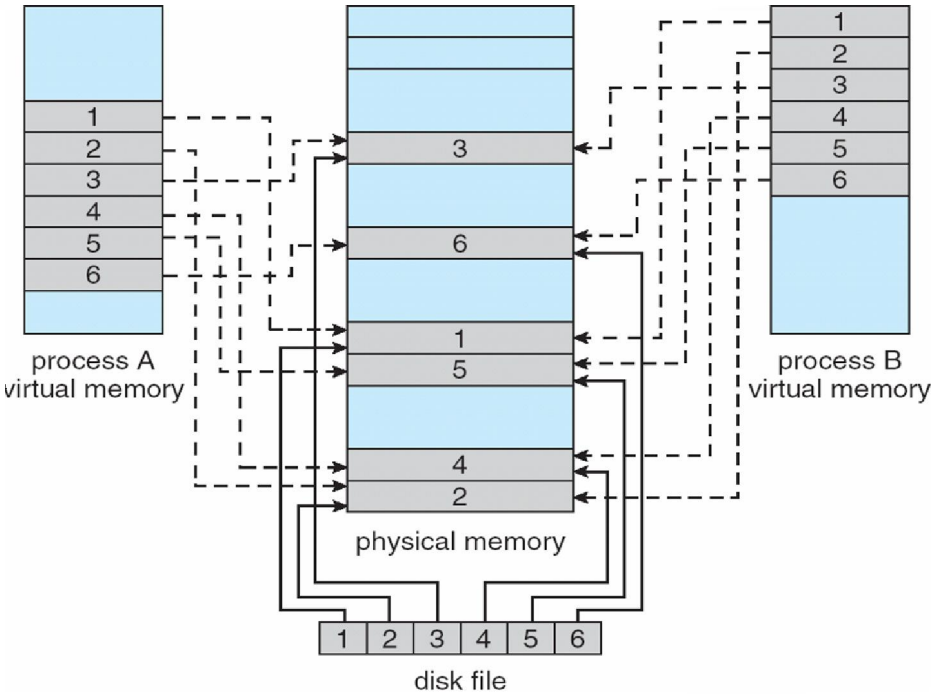


Memory-Mapped Files

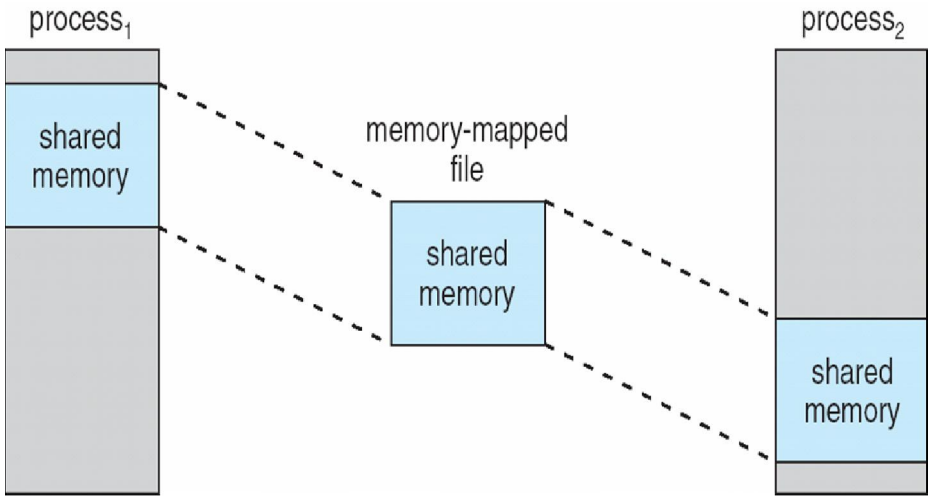
Memory-mapped file I/O allows file I/O to be treated as routine memory access by mapping a disk block to a page in memory.

- A file is initially read using demand paging. A page-sized portion of the file is read from the file system into a physical page. Subsequent reads/writes to/from the file are treated as ordinary memory accesses.
- Simplifies file access by treating file I/O through memory rather than read() write() system calls
- Also allows several processes to map the same file allowing the pages in memory to be shared

Memory Mapped Files



Memory-Mapped Shared Memory in Windows



Allocating Kernel Memory

- Treated differently from user memory
- Often allocated from a free-memory pool
    - Kernel requests memory for structures of varying sizes
    - Some kernel memory needs to be contiguous

C. SOAL LATIHAN/TUGAS

1. Jelaskan algoritma page replacement?
2. Jelaskan memory mapped files?

D. DAFTAR PUSTAKA

Buku

Bambang Hariyanto. 1997. Sistem Operasi, Bandung:Informatika Bandung.

Dali S. Naga. 1992. Teori dan Soal Sistem Operasi Komputer,Jakarta: Gunadarma.

Operating System Concepts (6th or 7th Edition). Silberschatz, Galvin, Gagne, ISBN: 0-471-25060-0. Wiley

Silberschatz Galvin. 1995. 4 Edition Operating System Concepts: Addison Wesley.

Sri Kusumadewi. 2000. Sistem Operasi. Yogyakarta: J&J Learning.

Tanenbaum, A.1992. Modern Operating Systems.New York: Prentice Hall

.

6.

Link and Sites:

http://www.ilmukomputer.com
http://vlsm.bebas.org
http://www.wikipedia.com