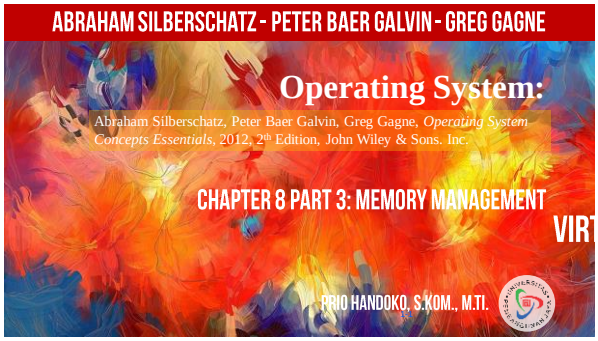


## An Overview

**Virtual memory is a technique that allows the execution of processes that are not completely in memory.** One major advantage of this scheme is that programs can be larger than physical memory. **Virtual memory also allows processes to share files easily and to implement shared memory. Virtual memory also allows processes to share files easily and to implement shared memory.**

**In this chapter, we discuss virtual memory in the form of demand paging and examine its complexity and cost.**



## Ch. 8: Virtual Memory

**Chapter Objectives.**

To explain the concepts of demand paging, page-replacement algorithms, and allocation of page frames.

**Agenda.**

- Page Replacement Algorithms



## Page Replacement Algorithms

- **Tujuan**
  Menentukan algoritma dengan *page fault rate* terkecil.
- **Evaluasi agoritma**
  Menjalankan sekumpulan *string/reference string* yang merujuk ke lokasi memori dan hitung *page fault* dari *string* tersebut.
- **String**
  Penanda nomor *page*, bukan *logical address.*

## Page Replacement Algorithms

**Algoritma Penggantian Page.**

1. FIFO (*first in first out*)
2. OPT (optimal)
3. LRU (*least recently used*)
4. SC (*second chance*)
5. Clock (*circular queue*)
6. ESC (*enhanced second chance*)
7. LFU (*least frequently used*)
8. MFU (*most frequently used*)



## Page Replacement Algorithms

**1. FIFO (First In First Out)**

*Page* yang menempati memori paling lama, dipilih untuk diganti.

*reference string*

| 7 | 0 | 1 | 2 | 3 | 0 | 3 | 2 | 1 | 0 | 3 | 2 | 1 | 2 | 0 | 1 | 7 | 0 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 7 | 7 | 2 | 2 | 2 | | | 1 | | | 1 | | | | | 1 | 0 | 0 |
| | 0 | 0 | 0 | 3 | 3 | | | 3 | | | 2 | | | | | 2 | 2 | 1 |
| | | 1 | 1 | 1 | 0 | | | 0 | | | 0 | | | | | 7 | 7 | 7 |

page fault = 8; page hit = 11



## Page Replacement Algorithms

**2. OPT (Optimal)**

Ganti *page* yang tidak akan digunakan pada periode berikutnya dengan waktu gilir yang terlama.

*reference string*

| 7 | 0 | 1 | 2 | 3 | 0 | 3 | 2 | 1 | 0 | 3 | 2 | 1 | 2 | 0 | 1 | 3 | 1 | 2 | 1 | 2 | 3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 7 | 7 | 2 | 2 | | | | 1 | | | 1 | | | | | 1 | | 1 | 1 | | |
| | 0 | 0 | 0 | 0 | | | | 0 | | | 0 | | | | | 0 | | 2 | 2 | | |
| | | 1 | 1 | 3 | | | | 3 | | | 2 | | | | | 3 | | 3 | 3 | | |

page fault = 12; page hit = 10



## Page Replacement Algorithms

**3. LRU (Least Recently Used)**

Ganti *page* yang memiliki periode terlama yang sudah pernah digunakan sebelumnya diantara *page* yang lain.

*reference string*

| 7 | 0 | 1 | 2 | 3 | 0 | 3 | 2 | 1 | 0 | 4 | 2 | 1 | 5 | 0 | 1 | 3 | 7 | 2 | 1 | 4 | 3 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 7 | 7 | 2 | 2 | 2 | | | 2 | 2 | 4 | 4 | 4 | 5 | 5 | | 3 | 3 | 3 | 1 | 1 | 1 |
| | 0 | 0 | 0 | 3 | 3 | | | 3 | 0 | 0 | 0 | 1 | 1 | 1 | | 1 | 1 | 2 | 2 | 2 | 3 |
| | | 1 | 1 | 1 | 0 | | | 1 | 1 | 1 | 2 | 2 | 2 | 0 | | 0 | 7 | 7 | 7 | 4 | 4 |

page fault = 3; page hit = 19

## Page Replacement Algorithms

### LRU (Counter Clock)

- Setiap *entri page* punya *field time-of-use.*
- Jika ada referensi ke suatu page, nilai register *clock* ditempatkan ke *field time-of-use.*
- Ganti *page* yang mempunyai waktu paling awal.

*reference string*

7 0 1 2 3 0 3 2 1 0 4 2 1 5 0 1 3 7 2 1 4 3



## Page Replacement Algorithms

| page: | fault: | frames: | | | time: | | |
|---|---|---|---|---|---|---|---|
| 7 | 1 | 7 | -1 | -1 | 1 | | |
| 0 | 2 | 7 | 0 | -1 | 1 | 2 | |
| 1 | 3 | 7 | 0 | 1 | 1 | 2 | 3 |
| 2 | 4 | 2 | 0 | 1 | 4 | 2 | 3 |
| 3 | 5 | 2 | 3 | 1 | 4 | 5 | 3 |
| 0 | 6 | 2 | 3 | 0 | 4 | 5 | 6 |
| 3 | 6 | 2 | 3 | 0 | 4 | 7 | 6 |
| 2 | 6 | 2 | 3 | 0 | 8 | 7 | 6 |
| 1 | 7 | 2 | 3 | 1 | 8 | 7 | 9 |
| 0 | 8 | 2 | 0 | 1 | 8 | 10 | 9 |
| 4 | 9 | 4 | 0 | 1 | 11 | 10 | 9 |
| 2 | 10 | 4 | 0 | 2 | 11 | 10 | 12 |
| 1 | 11 | 4 | 1 | 2 | 11 | 13 | 12 |
| 5 | 12 | 5 | 1 | 2 | 14 | 13 | 12 |
| 0 | 13 | 5 | 1 | 0 | 14 | 13 | 15 |
| 1 | 13 | 5 | 1 | 0 | 14 | 16 | 15 |
| 3 | 14 | 3 | 1 | 0 | 17 | 16 | 15 |
| 7 | 15 | 3 | 1 | 7 | 17 | 16 | 18 |
| 2 | 16 | 3 | 2 | 7 | 17 | 19 | 18 |
| 1 | 17 | 1 | 2 | 7 | 20 | 19 | 18 |
| 4 | 18 | 1 | 2 | 4 | 20 | 19 | 21 |
| 3 | 19 | 1 | 3 | 4 | 22 | 19 | 21 |

**Ketentuan:**

Ganti *page* yang mempunyai waktu paling awal.



## Page Replacement Algorithms

### LRU (Stack)

- Setiap ada referensi *page*, pindahkan page ke posisi paling atas.
- *Page* yang paling sering digunakan (*most recently used*) berada di posisi atas.
- *Page* yang paling jarang digunakan (*least recently used*) berada di posisi bawah.
- Umumnya berbentuk *double linked-list.*



## Page Replacement Algorithms

| page: | fault: | frames: | | | stack: | | |
|---|---|---|---|---|---|---|---|
| 7 | 1 | 7 | -1 | -1 | 7 | 0 | |
| 0 | 2 | 7 | 0 | -1 | 7 | 0 | 1 |
| 1 | 3 | 7 | 0 | 1 | 0 | 1 | 2 |
| 2 | 4 | 2 | 0 | 1 | 1 | 2 | 3 |
| 3 | 5 | 2 | 3 | 1 | 2 | 3 | 0 |
| 0 | 6 | 2 | 3 | 0 | 3 | 0 | 3 |
| 3 | 6 | 2 | 3 | 0 | 2 | 0 | 3 |
| 2 | 6 | 2 | 3 | 0 | 0 | 3 | 2 |
| 1 | 7 | 2 | 3 | 1 | 3 | 2 | 1 |
| 0 | 8 | 2 | 0 | 1 | 2 | 1 | 0 |
| 4 | 9 | 4 | 0 | 1 | 1 | 0 | 4 |
| 2 | 10 | 4 | 0 | 2 | 0 | 4 | 2 |
| 1 | 11 | 4 | 1 | 2 | 4 | 2 | 1 |
| 5 | 12 | 5 | 1 | 2 | 2 | 1 | 5 |
| 0 | 13 | 5 | 1 | 0 | 1 | 5 | 0 |
| 1 | 13 | 5 | 1 | 0 | 5 | 0 | 1 |
| 3 | 14 | 3 | 1 | 0 | 0 | 1 | 3 |
| 7 | 15 | 3 | 1 | 7 | 1 | 3 | 7 |
| 2 | 16 | 3 | 2 | 7 | 3 | 7 | 2 |
| 1 | 17 | 1 | 2 | 7 | 7 | 2 | 1 |
| 4 | 18 | 1 | 2 | 4 | 2 | 1 | 4 |
| 3 | 19 | 1 | 3 | 4 | 1 | 4 | 3 |

**Ketentuan:**

- *Page* yang paling sering digunakan (*most recently used*) berada di posisi atas.
- *Page* yang paling jarang digunakan (*least recently used*) berada di posisi bawah.

## Page Replacement Algorithms

### 4. SC (Second Chance)

- Modifikasi algoritma FIFO
  - Menghindari pergantian *old page* yang direferensi
  - Mencari *old page* yang jarang direferensi
- Menggunakan bit referensi (*reference bit*)
  - Nilai bit = 0, page diganti
  - Nilai bit = 1
- Jika page selalu direferensi, maka *page* tak pernah dihapus



## Page Replacement Algorithms

**Ketentuan:**

- Apabila terjadi *page fault* dan tidak ada *frame* yang kosong, maka akan dilakukan razia (pencarian korban) halaman yang *reference* bit-nya bernilai 0 dimulai dari bawah antrian (seperti FIFO).
- Setiap halaman yang tidak ditukar (*swap* - karena *reference* bit-nya bernilai 1), setiap dilewati saat razia *reference* bit-nya akan diset menjadi 0.
- Apabila ditemukan halaman yang *reference* bit-nya bernilai 0, maka halaman itu yang ditukar.
- Apabila sampai di ujung antrian tidak ditemukan halaman yang *reference* bit-nya bernilai 0, maka razia dilakukan lagi dari awal.



## Page Replacement Algorithms

*reference string*

| 3 | 2 | 1 | 0 | 3 | 2 | 4 | 3 | 2 | 0 | 4 | 1 | 4 | 3 | 0 | 3 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3x | 3x | 3x | 3x | 3x | 3x | 2 | 1 | 0 | 4x | 4x | 3 | 2 | 0 | 1x | 1x | 1x |
| | 2x | 2x | 2x | 2x | 2x | 1 | 0 | 4x | 3x | 3x | 2 | 0 | 1x | 4x | 4x | 4x |
| | | 1x | 1x | 1x | 1x | 0 | 4x | 3x | 2x | 2x | 0 | 1x | 4x | 3x | 3x | 3x |
| | | | 0x | 0x | 0x | 4x | 3x | 2x | 0x | 0x | 1x | 4x | 3x | 0x | 0x | 0x |

page fault = 5; page hit = 12



## Page Replacement Algorithms

### 5. Clock (Circular Queue)

- Page dalam bentuk lingkaran dan bergerak searah jarum jam.
- *Pointer* bergerak ke *page* berikutnya jika nomor *page* (*reference string*) telah dimuatkan ke dalam *frame*.

*reference string*

3 2 1 0 3 2 4 3 2 0 4 1 4 3 0 3 0

## Page Replacement Algorithms

*reference string*

| 3 | 2 | 1 | 0 | 3 | 2 | 4 | 3 | 2 | 0 | 4 | 1 | 4 | 3 | 0 | 3 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 3 | 3 | 3 | | | 3 | | | | | 3 | | | 0 | 0 | |
| | 2 | 2 | 2 | | | 2 | | | | | 2 | | | 2 | 3 | |
| | | 1 | 1 | | | 4 | | | | | 4 | | | 4 | 4 | |
| | | | 0 | | | 0 | | | | | 1 | | | 1 | 1 | |

page fault = 9; page hit = 8



## Page Replacement Algorithms

### 6. Enhanced SC

- Menggunakan 2 buah bit yang berfungsi sebagai *status page*.
  - Bit M (*modified bit*): *Page* yang telah dimodifikasi
    - Bit M = 0 → tidak dimodifikasi
    - Bit M = 1 → sudah dimodifikasi
  - Bit R (*referenced bit*): *Page* yang sedang direferensi (*referenced*)
    - Bit R = 1 → sedang direferensi
    - Bit R = 0 → tidak sedang direferensi



## Page Replacement Algorithms

- Adanya dua bit di atas maka akan dapat dikelompokkan menjadi 4 kelas page, yaitu :
  - Kelas 0 → R = 0, M = 0
  - Kelas 1 → R = 0, M = 1
  - Kelas 2 → R = 1, M = 0
  - Kelas 3 → R = 1, M = 1



## Page Replacement Algorithms

- Algoritma *second chance* yang ditingkatkan merujuk kepada kondisi *page frame* algoritma *second chance*.

| 3 | 2 | 1 | 0 | 3 | 2 | 4 | 3 | 2 | 0 | 4 | 1 | 4 | 3 | 0 | 3 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3x | 3x | 3x | 3x | 3x | 3x | 2 | 1 | 0 | 4x | 4x | 3 | 2 | 0 | 1x | 1x | 1x |
| | 2x | 2x | 2x | 2x | 2x | 1 | 0 | 4x | 3x | 3x | 2 | 0 | 1x | 4x | 4x | 4x |
| | | 1x | 1x | 1x | 1x | 0 | 4x | 3x | 2x | 2x | 0 | 1x | 4x | 3x | 3x | 3x |
| | | | 0x | 0x | 0x | 4x | 3x | 2x | 0x | 0x | 1x | 4x | 3x | 0x | 0x | 0x |

## Page Replacement Algorithms

| page: | 3 | fault: | 1 | frames: | 3 | -1 | -1 | -1 | R, M: | 1,0 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 2 | | 2 | | 3 | 2 | -1 | -1 | | 1,0 | 1,0 | | |
| | 1 | | 3 | | 3 | 2 | 1 | -1 | | 1,0 | 1,0 | 1,0 | |
| | 0 | | 4 | | 3 | 2 | 1 | 0 | | 1,0 | 1,0 | 1,0 | 1,0 |
| | 3 | | 4 | | 3 | 2 | 1 | 0 | | 1,0 | 0,0 | 0,0 | 0,0 |
| | 2 | | 4 | | 3 | 2 | 1 | 0 | | 0,1 | 1,0 | 0,0 | 0,0 |
| | 4 | | 5 | | 2 | 1 | 0 | 4 | | 0,1 | 0,1 | 0,1 | 1,1 |
| | 3 | | 6 | | 1 | 0 | 4 | 3 | | 0,1 | 0,1 | 0,1 | 1,1 |
| | 2 | | 7 | | 0 | 4 | 3 | 2 | | 0,1 | 0,1 | 0,1 | 1,1 |
| | 0 | | 8 | | 4 | 3 | 2 | 0 | | 0,1 | 0,1 | 0,1 | 1,1 |
| | 4 | | 8 | | 4 | 3 | 2 | 0 | | 1,0 | 0,0 | 0,0 | 0,0 |
| | 1 | | 9 | | 3 | 2 | 0 | 1 | | 0,1 | 0,1 | 0,1 | 1,1 |
| | 4 | | 10 | | 2 | 0 | 1 | 4 | | 0,1 | 0,1 | 0,1 | 1,1 |
| | 3 | | 11 | | 0 | 1 | 4 | 3 | | 0,1 | 0,1 | 0,1 | 1,1 |
| | 0 | | 12 | | 1 | 4 | 3 | 0 | | 0,1 | 0,1 | 0,1 | 1,1 |
| | 3 | | 12 | | 1 | 4 | 3 | 0 | | 0,0 | 0,0 | 1,0 | 0,0 |
| | 0 | | 12 | | 1 | 4 | 3 | 0 | | 0,0 | 0,0 | 0,0 | 1,0 |

**Ketentuan:**

- Algoritma *second chance* yang ditingkatkan merujuk kepada kondisi *page frame* algoritma *second chance.*



## Page Replacement Algorithms

### 7. LFU (Least Frequently Used)

Mengganti page yang mempunyai jumlah referensi terkecil.

*reference string*

| 3 | 2 | 1 | 0 | 3 | 2 | 4 | 3 | 2 | 0 | 4 | 1 | 4 | 3 | 0 | 3 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 3 | 3 | 3 | | | 3 | | | 3 | | 3 | | | 3 | | |
| | 2 | 2 | 2 | | | 2 | | | 2 | | 2 | | | 2 | | |
| | | 1 | 1 | | | 1 | | | 0 | | 1 | | | 0 | | |
| | | | 0 | | | 4 | | | 4 | | 4 | | | 4 | | |

page fault = 9; page hit = 8



## Page Replacement Algorithms

### 8. MFU (Most Frequently Used)

Mengganti page yang mempunyai jumlah referensi terbanyak.

*reference string*

| 3 | 2 | 1 | 0 | 3 | 2 | 4 | 3 | 2 | 0 | 4 | 1 | 4 | 3 | 0 | 3 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 3 | 3 | 3 | | | 4 | 4 | 4 | | | | | 3 | | | |
| | 2 | 2 | 2 | | | 2 | 3 | 2 | | | | | 2 | | | |
| | | 1 | 1 | | | 1 | 1 | 1 | | | | | 1 | | | |
| | | | 0 | | | 0 | 0 | 0 | | | | | 0 | | | |

page fault = 9; page hit = 8