Tugas

“Latihan Manajemen Memori”

Diajukan untuk memenuhi salah satu tugas Mata Kuliah Sistem Operasi yang di ampu oleh:

Arfiani Nur Khusna, S.T., M.Kom.

Disusun Oleh:

Mohammad Farid Hendianto 2200018401

UNIVERSITAS AHMAD DAHLAN

FAKULTAS TEKNOLOGI INDUSTRI

PROGRAM STUDI INFORMATIKA

TAHUN 2023

1. Terdapat partisi memori 100K, 500K, 200K, 300K dan 600K, bagaimana algoritma First-fit, Best-fit dan Worst-fit menempatkan proses 212K, 417K, 112K dan 426K (berurutan) ? Algoritma mana yang menggunakan memori secara efisien ?

Untuk melihat cara kerja memory allocation dapat melihat kodingan berikut:

```python
# Python3 implementation of Memory Allocation Algorithms
def memory_allocation(blockSize, m, processSize, n, algorithm):
    print (f"\nMemory Allocation using {algorithm} fit algorithm")
    allocation = [-1] * n
    for i in range(n):
        print(f"\nAllocating process {i + 1}...")
        idx = -1
        for j in range(m):
            if blockSize[j] >= processSize[i]:
                if idx == -1:
                    idx = j
                    print(f"Block {j + 1} is a potential fit")
                elif algorithm == 'best' and blockSize[idx] > blockSize[j]:
                    idx = j
                    print(f"Block {j + 1} is a better fit")
                elif algorithm == 'worst' and blockSize[idx] < blockSize[j]:
                    idx = j
                    print(f"Block {j + 1} is a worse fit")

        if idx != -1:
            allocation[i] = idx
            print(f"Process {i + 1} allocated to block {idx + 1}")
            print(f"blockSize[{idx + 1}] = {blockSize[idx]} K - {processSize[i]} K = {blockSize[idx] - processSize[i]} K")
            blockSize[idx] -= processSize[i]
            print(f"Block {idx + 1} remaining size: {blockSize[idx]} K")
        else:
            print(f"Process {i + 1} not allocated")

    print("\n{:^12} {:^18} {:^10}".format("Process No.", "Process Size (K)", "Block no."))
    for i in range(n):
        if allocation[i] != -1:
            print("{:^12} {:^18} {:^10}".format(i + 1, processSize[i], allocation[i] + 1))
        else:
            print("{:^12} {:^18} {:^10}".format(i + 1, processSize[i], "Not Allocated"))
    print("Remaining block size: ", end='')
    for i in range(m):
        print(f"{blockSize[i]} ", end='')
    print("\n")

# Driver code
if __name__ == '__main__':
    blockSize = [100, 500, 200, 300, 600]
    processSize = [212, 417, 112, 426]
    m = len(blockSize)
    n = len(processSize)

    memory_allocation(blockSize[:], m, processSize, n, 'first')
    memory_allocation(blockSize[:], m, processSize, n, 'best')
    memory_allocation(blockSize[:], m, processSize, n, 'worst')

```

Gambar 1 Kodingan cara kerja alokasi memory, Sumber: geeksforgeeks.org, dengan revisi Mohammad Farid Hendianto

Dengan algoritma First-fit

```python
def firstFit(blockSize, m, processSize, n):
    allocation = [-1] * n

    for i in range(n):
        for j in range(m):
            if blockSize[j] >= processSize[i]:
                allocation[i] = j
                blockSize[j] -= processSize[i]
                break

    print(" Process No. Process Size      Block no.")
    for i in range(n):
        print(" ", i + 1, "         ", processSize[i], "         ", end = " ")
        if allocation[i] != -1:
            print(allocation[i] + 1)
        else:
            print("Not Allocated")

if __name__ == '__main__':
    blockSize = [100, 500, 200, 300, 600]
    processSize = [212, 417, 112, 426]
    m = len(blockSize)
    n = len(processSize)

    firstFit(blockSize, m, processSize, n)
```

Gambar 2 Algoritma First-fit, , Sumber: geeksforgeeks.org

Berikut cara kerja algoritma tersebut:

Memory Allocation using first fit algorithm

Allocating process 1...

Block 2 is a potential fit

Process 1 allocated to block 2

212K is put in 500K partition.

blockSize[2] = 500 K - 212 K = 288 K

Block 2 remaining size: 288 K

Allocating process 2...

Block 5 is a potential fit

Process 2 allocated to block 5

417K is put in 600K partition.

blockSize[5] = 600 K - 417 K = 183 K

Block 5 remaining size: 183 K

Allocating process 3...

Block 2 is a potential fit

Process 3 allocated to block 2

112K is put in 288K partition

blockSize[2] = 288 K - 112 K = 176 K

Block 2 remaining size: 176 K

Allocating process 4...

Process 4 not allocated

Process No.	Process Size (K)	Block no.
1	212	2
2	417	5
3	112	2
4	426	Not Allocated

Remaining block size: 100 176 200 300 183

Dengan algoritma Best-fit

```python
def bestFit(blockSize, m, processSize, n):
    allocation = [-1] * n

    for i in range(n):
        bestIdx = -1
        for j in range(m):
            if blockSize[j] >= processSize[i]:
                if bestIdx == -1:
                    bestIdx = j
                elif blockSize[bestIdx] > blockSize[j]:
                    bestIdx = j

        if bestIdx != -1:
            allocation[i] = bestIdx
            blockSize[bestIdx] -= processSize[i]

    print("Process No. Process Size     Block no.")
    for i in range(n):
        print(i + 1, "         ", processSize[i], end = "         ")
        if allocation[i] != -1:
            print(allocation[i] + 1)
        else:
            print("Not Allocated")

if __name__ == '__main__':
    blockSize = [100, 500, 200, 300, 600]
    processSize = [212, 417, 112, 426]
    m = len(blockSize)
    n = len(processSize)

    bestFit(blockSize, m, processSize, n)
```

Gambar 3 Algoritma Best-fit, , Sumber: geeksforgeeks.org

Berikut cara kerja algoritma tersebut:

Memory Allocation using best fit algorithm

Allocating process 1...

Block 2 is a potential fit

Block 4 is a better fit

Process 1 allocated to block 4

212K is put in 300K partition.

blockSize[4] = 300 K - 212 K = 88 K

Block 4 remaining size: 88 K

Allocating process 2...

Block 2 is a potential fit

Process 2 allocated to block 2

417K is put in 500K partition.

blockSize[2] = 500 K - 417 K = 83 K

Block 2 remaining size: 83 K

Allocating process 3...

Block 3 is a potential fit

Process 3 allocated to block 3

112K is put in 200K partition.

blockSize[3] = 200 K - 112 K = 88 K

Block 3 remaining size: 88 K

Allocating process 4...

Block 5 is a potential fit

Process 4 allocated to block 5

426K is put in 600K partition.

blockSize[5] = 600 K - 426 K = 174 K

Block 5 remaining size: 174 K

Process No.	Process Size (K)	Block no.
1	212	4
2	417	2
3	112	3
4	426	5

Remaining block size: 100 83 88 88 174

Dengan algoritma Worst-fit

Berikut cara kerja algoritma tersebut:

```python
def worstFit(blockSize, m, processSize, n):
    allocation = [-1] * n

    for i in range(n):
        wstIdx = -1
        for j in range(m):
            if blockSize[j] >= processSize[i]:
                if wstIdx == -1:
                    wstIdx = j
                elif blockSize[wstIdx] < blockSize[j]:
                    wstIdx = j

        if wstIdx != -1:
            allocation[i] = wstIdx
            blockSize[wstIdx] -= processSize[i]

    print("Process No. Process Size Block no.")
    for i in range(n):
        print(i + 1, "          ", processSize[i], end = "     ")
        if allocation[i] != -1:
            print(allocation[i] + 1)
        else:
            print("Not Allocated")

if __name__ == '__main__':
    blockSize = [100, 500, 200, 300, 600]
    processSize = [212, 417, 112, 426]
    m = len(blockSize)
    n = len(processSize)

    worstFit(blockSize, m, processSize, n)
```

Gambar 4 Algoritma Best-fit, , Sumber: geeksforgeeks.org

Berikut cara kerja algoritma tersebut:

Memory Allocation using worst fit algorithm

Allocating process 1...

Block 2 is a potential fit

Block 5 is a worse fit

Process 1 allocated to block 5

212K is put in 600K partition.

blockSize[5] = 600 K - 212 K = 388 K

Block 5 remaining size: 388 K

Allocating process 2...

Block 2 is a potential fit

Process 2 allocated to block 2

417K is put in 500K partition.

blockSize[2] = 500 K - 417 K = 83 K

Block 2 remaining size: 83 K

Allocating process 3...

Block 3 is a potential fit

Block 4 is a worse fit

Block 5 is a worse fit

Process 3 allocated to block 5

112K is put in 388K partition.

blockSize[5] = 388 K - 112 K = 276 K

Block 5 remaining size: 276 K

Allocating process 4...

Process 4 not allocated

Process No.	Process Size (K)	Block no.
1	212	5
2	417	2
3	112	5
4	426	Not Allocated

Remaining block size: 100 83 200 300 276

Berdasarkan hasil alokasi memori di atas, algoritma yang paling efisien dalam menggunakan memori adalah algoritma Best-Fit.

Alasan utamanya adalah:

- Algoritma Best-Fit dapat mengalokasikan semua proses (1, 2, 3, dan 4), tidak ada proses yang gagal dialokasikan. Sedangkan algoritma First-Fit dan Worst-Fit tidak dapat mengalokasikan proses 4.
- Algoritma Best-Fit menyisakan fragmen memori terkecil dibandingkan algoritma lainnya. Sisanya adalah:

Best-Fit : 100K, 83K, 88K, 88K, 174K (total = 533K)

First-Fit : 100K, 176K, 200K, 300K, 183K (total = 959K)

Worst-Fit : 100K, 83K, 200K, 300K, 276K (total = 959K)

- Algoritma Best-Fit menempatkan proses di partisi memori yang paling sesuai ukurannya (best fitting), sehingga mengurangi fragmentasi memori internal. Sedangkan First-Fit menempatkan berdasarkan partisi pertama yang cukup, dan Worst-Fit berdasarkan partisi terbesar yang masih tersisa.

Oleh karena itu, **algoritma Best-Fit** adalah yang paling efisien karena dapat mengalokasikan semua proses dengan fragmentasi memori internal dan eksternal paling minimum. Efisiensi penggunaan memori sangat penting untuk kinerja sistem secara keseluruhan.

2. Diketahui ruang alamat logika dengan 8 page masing-masing 1024 word dipetakan ke memori fisik 32 frame. Berapa bit alamat logika ? Berapa bit alamat fisik ?

Soal ini merupakan soal ISRO (Indian Space Research Organisation) | ISRO CS 2013 |

Diketahui:

- Ukuran logical address space adalah 8 halaman (pages), dimana tiap halaman berukuran 1024 kata (words)
- Ukuran physical memory adalah 32 frame

Jawab:

Menghitung ukuran logical address

Jumlah halaman (pages) adalah 8 halaman

Menentukan banyaknya bit yang dibutuhkan untuk mengkodekan nomor halaman (page number) dengan rumus:

$$bit_{PageNumber} = \log_2(jumlah\ pages) = \log_2(8) = \mathbf{3\ bit}$$

Ukuran tiap halaman adalah **1024 words**

Menentukan banyaknya bit offset dengan rumus:

$$bit_{offset} = \log_2(ukuran\_halaman) = \log_2(1024) = \mathbf{10\ bit}$$

Jadi total bit untuk logical address adalah:

$$bit_{PageNumber} + bit_{offset} = 3 + 10 = \mathbf{13\ bit}$$

Menghitung ukuran physical address

Jumlah frame pada physical memory adalah 32 frame

Menentukan banyaknya bit untuk mengkodekan nomor frame (frame number) dengan rumus:

$$bit_{FrameNumber} = \log_2(jumlah\ frame) = \log_2 32 = \mathbf{5\ bit}$$

Ukuran offset tetap 1024 words atau **10 bit** (sama dengan logical address)

$$bit_{offset} = \log_2(ukuran\_halaman) = \log_2(1024) = \mathbf{10\ bit}$$

Jadi total bit untuk physical address adalah:

$$bit_{FrameNumber} + bit_{offset} = \mathbf{5} + \mathbf{10} = \mathbf{15\ bit}$$

Maka ukuran **logical address** adalah **13 bit** dan **ukuran physical address** adalah **15 bit**.