MIT World Peace University

Operating Systems
Second Year B. Tech, Semester 3

Memory Management and Simulation of Paging Algorithms

Assignment 2
Practical Report

Prepared By

Krishnaraj Thadesar
Cyber Security and Forensics
Batch A1, PA 20

November 29, 2022

1 Code

```c
#include <stdio.h>
#define MAX_FRAMES 10
#define MAX_PAGES 20
struct Frames
{
    int page;
    int insert_index;
} frames[MAX_FRAMES];

int pages[MAX_PAGES] = {7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 3, 0, 0, 0, 0, 0};

int frame_size = 4, no_of_pages = 14;
int hits = 0, faults = 0;
int page_search(int page)
{
    for (int i = 0; i < frame_size; i++)
    {
        if (page == frames[i].page)
        {
            return i;
        }
    }
    return -1;
}
void initialize_frame()
{
    for (int i = 0; i < frame_size; i++)
    {
        frames[i].page = -1;
        frames[i].insert_index = -1;
    }
}
void display()
{
    printf("Displaying frame: \n");
    for (int i = 0; i < frame_size; i++)
    {
        printf("%d\n", frames[i].page);
    }
    printf("\n");
}
int where_to_insert()
{
    int min = 1000;
    int min_index = 0;
    for (int i = 0; i < frame_size; i++)
    {
        if (frames[i].insert_index == -1)
        {
            return i;
        }
        else if (frames[i].insert_index <= min)
        {
            min = frames[i].insert_index;
            min_index = i;
        }
```

```
    }
    return min_index;
}
int lru()
{
    for (int i = 0; i < no_of_pages; i++)
    {
        printf("Currently doing : %d\n\n", pages[i]);
        int where = page_search(pages[i]);
        if (where != -1)
        {
            printf("Hit\n");
            hits++;
            frames[where].insert_index = i;
        }
        else
        {
            printf("Miss\n");
            faults++;
            int temp = where_to_insert();
            frames[temp].page = pages[i];
            frames[temp].insert_index = i;
        }
        display();
    }
}
int fifo()
{
    for (int i = 0; i < no_of_pages; i++)
    {
        printf("Currently doing : %d\n\n", pages[i]);
        if (page_search(pages[i]))
        {
            printf("Hit\n");
            hits++;
        }
        else
        {
            printf("Miss\n");
            faults++;
            int temp = where_to_insert();
            frames[temp].page = pages[i];
            frames[temp].insert_index = i;
        }
        display();
    }
}
// int
int main()
{
    printf("Enter how many frames you have\n");
    scanf("%d", &frame_size);
    // printf("Enter how many Pages you have\n");
    // scanf("%d", &no_of_pages);
    // printf("Enter the Pages : \n");
    // for (int i = 0; i < no_of_pages; i++)
    // {
    //     scanf("%d", pages[i]);
    // }
```

```
    printf("Executing First in First Out\n");
    initialize_frame();
    fifo();
    printf("Hits: %d\n", hits);
    printf("Faults: %d\n", faults);

    hits = 0, faults = 0;
    printf("Executing Least Recently Used\n");
    initialize_frame();
    lru();
    printf("Hits: %d\n", hits);
    printf("Faults: %d\n", faults);

    return 0;
}
```

Listing 1: Code

2 Input and Output

```
Enter how many frames you have
3
Executing First in First Out
Currently doing : 7

Hit
Displaying frame:
-1
-1
-1

Currently doing : 0

Hit
Displaying frame:
-1
-1
-1

Currently doing : 1

Hit
Displaying frame:
-1
-1
-1

Currently doing : 2

Hit
Displaying frame:
-1
-1
-1

Currently doing : 0

Hit
Displaying frame:
```

```
-1
-1
-1

Currently doing : 3

Hit
Displaying frame:
-1
-1
-1

Currently doing : 0

Hit
Displaying frame:
-1
-1
-1

Currently doing : 4

Hit
Displaying frame:
-1
-1
-1

Currently doing : 2

Hit
Displaying frame:
-1
-1
-1

Currently doing : 3

Hit
Displaying frame:
-1
-1
-1

Currently doing : 0

Hit
Displaying frame:
-1
-1
-1

Currently doing : 3

Hit
Displaying frame:
-1
-1
-1
```

```
99
100 Currently doing : 2
101
102 Hit
103 Displaying frame:
104 -1
105 -1
106 -1
107
108 Currently doing : 3
109
110 Hit
111 Displaying frame:
112 -1
113 -1
114 -1
115
116 Hits: 14
117 Faults: 0
118 Executing Least Recently Used
119 Currently doing : 7
120
121 Miss
122 Displaying frame:
123 7
124 -1
125 -1
126
127 Currently doing : 0
128
129 Miss
130 Displaying frame:
131 7
132 0
133 -1
134
135 Currently doing : 1
136
137 Miss
138 Displaying frame:
139 7
140 0
141 1
142
143 Currently doing : 2
144
145 Miss
146 Displaying frame:
147 2
148 0
149 1
150
151 Currently doing : 0
152
153 Hit
154 Displaying frame:
155 2
156 0
157 1
```

```
158
159 Currently doing : 3
160
161 Miss
162 Displaying frame:
163 2
164 0
165 3
166
167 Currently doing : 0
168
169 Hit
170 Displaying frame:
171 2
172 0
173 3
174
175 Currently doing : 4
176
177 Miss
178 Displaying frame:
179 4
180 0
181 3
182
183 Currently doing : 2
184
185 Miss
186 Displaying frame:
187 4
188 0
189 2
190
191 Currently doing : 3
192
193 Miss
194 Displaying frame:
195 4
196 3
197 2
198
199 Currently doing : 0
200
201 Miss
202 Displaying frame:
203 0
204 3
205 2
206
207 Currently doing : 3
208
209 Hit
210 Displaying frame:
211 0
212 3
213 2
214
215 Currently doing : 2
216
```

```
Hit
Displaying frame:
0
3
2

Currently doing : 3

Hit
Displaying frame:
0
3
2

Hits: 5
Faults: 9
```

Listing 2: Input and Output

29/11/22

LAB Assignment - 7

Krishnaraj P.T.
PA 20. A: 1

FAQ's

Q.1. Describe Page Table, Frame Table and explain the hardware support required to implement paging.

→ Page Table:

The data structure that is used by the virtual memory system is the OS to store mapping between physical and logical addresses. is commonly known as Page Table. It is stored is main memory.

Generally:

No. of Entries in Page Table = The Number of pages in which the process is divided.

Frame Table:

- Frame Table specifies the frame where the page is stored in the memory.
- The no. of bits in frame number depends on number of frames in main memory.

Hardware Support required to implement Paging.

Paging is a storage mechanism used to retrieve processes from the secondary storage into the main memory, in the form of pages. The main idea behind paging is to divide each process in the form of pages. The main memory will also be divided in the form of frames.

(Q.2) Explain First Fit, Best Fit, Worst Fit.

(1) First Fit:

The approach is to allocate the Best free partition which meets the requirement of the requesting process. This algorithm first searches the entire list of free partitions; and considers the smallest hole that is adequate. It then tries to find which is close to actual size needed.

(2) First Fit: The approach is to find or allocate the first free partition or hole large enough which can accomodate the process. It finishes after finding First Suitable Partition.

(3) Worst Fit : Approach it is to locate largest available free partition so that the portion left will be big enough to be useful. It is reverse of best Fit.

(Q.3) Explain Paging and Page Table.

Paging is a memory management scheme that eliminates the need of contiguous allocation of memory. The process of retrieving processes in the form of pages from the secondary storage into main memory is known as paging.

The basic purpose is to separate each procedure into pages.

Page Table: A page table is the data structure used by a virtual memory system in a computer OS to store the mapping between virtual addresses and physical addresses.

(Q.4) Explain Virtual Memory:

Virtual Memory is a memory management technique that secondary memory can be used as part of the main memory. Virtual memory uses both hardware & software.