Mawlana Bhashani Science and Technology University

Lab-Report

Report No: 11

Course code: ICT-3110

Course title: Operating Systems Lab

Date of Performance:

Date of Submission: 30/09/2020

Submitted by

Name: Md Humayun Kabir

ID:IT-18026

3rd year 1st semester

Session: 2017-2018

Dept. of ICT

MBSTU.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

Experiment no : 11

Experiment name : Implementation of FIFO page replacement algorithm.

Theory :

In operating systems that use paging for memory management, page replacement algorithm are needed to decide which page needed to be replaced when new page comes in. Whenever a new page is referred and not present in memory, page fault occurs and Operating System replaces one of the existing pages with newly needed page. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all algorithms is to reduce number of page faults.

First In First Out (FIFO) page replacement algorithm –
This is the simplest page replacement algorithm. In this algorithm, operating system keeps track of all pages in the memory in a queue, oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal.
Example -1. Consider page reference string 1, 3, 0, 3, 5, 6 and 3 page slots.
Initially all slots are empty, so when 1, 3, 0 came they are allocated to the empty slots —>
3 Page Faults.
when 3 comes, it is already in memory so —> 0 Page Faults.
Then 5 comes, it is not available in memory so it replaces the oldest page slot i.e 1. —>1 Page Fault.
Finally 6 comes, it is also not available in memory so it replaces the oldest page slot i.e 3 —
>1 Page Fault.

So total page faults = **5**.
Example -2. Consider the following reference string: 0, 2, 1, 6, 4, 0, 1, 0, 3, 1, 2, 1.
Using FIFO page replacement algorithm –

0	2	1	6	4	0	1	0	3	1	2	1
0	0	0	0	4	4			4	4	2	
	2	2	2	2	0		hit	0	0	0	
		1	1	1	1	hit		3	3	3	
			6	6	6			6	1	1	hit

So, total number of page faults = 9.

Given memory capacity (as number of pages it can hold) and a string representing pages to be referred, write a function to find number of page faults.

Working Process :

```
// C++ implementation of FIFO page replacement
#include<bits/stdc++.h>
```

```
using namespace std;
int pageFaults(int pages[], int n, int capacity)
{
	unordered_set<int> s;
	queue<int> indexes;
	int page_faults = 0;
	for (int i=0; i<n; i++)
	{
		if (s.size() < capacity)
		{
			if (s.find(pages[i])==s.end())
			{
				s.insert(pages[i]);
				page_faults++;
				indexes.push(pages[i]);
			}
		}
		else
		{
			if (s.find(pages[i]) == s.end())
			{
				int val = indexes.front();
				indexes.pop();
				s.erase(val);
				s.insert(pages[i]);
				indexes.push(pages[i]);
				page_faults++;
			}
		}
	}
	return page_faults;
}

// Driver code
int main()
{
	int pages[] = {7, 0, 1, 2, 0, 3, 0, 4,
				2, 3, 0, 3, 2};
	int n = sizeof(pages)/sizeof(pages[0]);
	int capacity = 4;
	cout << pageFaults(pages, n, capacity);
	return 0;
}
```

Output :

```
7
Process returned 0 (0x0)   execution time : 0.114 s
Press any key to continue.
```

Discussion :

This lab helps to learn FIFO page replacement algorithm. We have implemented this algorithm using C language. In future we can solve any problem of this algorithm.