The International Journal Of Engineering And Science (IJES)
|| Volume || 5 || Issue || 1 || Pages || PP -53-57 || 2016 ||
ISSN (e): 2319 – 1813 ISSN (p): 2319 – 1805

Study of Page Replacement Algorithms and their analysis with C#

[1]Sourabh Shastri, [2]Anand Sharma, [3]Vibhakar Mansotra
[1]*Department of Computer Science & IT, Bhaderwah Campus, University of Jammu, J&K, India*
[2,3]*Department of Computer Science & IT, University of Jammu, J&K, India*

----------ABSTRACT----------
Page replacement algorithms choose pages to swap out from the memory when a new page needs memory for allocation. A cluster of algorithms have developed for page replacement. Each algorithm has the objective to minimize the number of page faults. With minimum page faults, the performance of the process is increased. In this paper, study and analysis of three algorithms viz. First in First out (FIFO), Least Recently Used (LRU) and Optimal Page Replacement (OPT) is made using C# Visual Studio.net.
Keywords: *FIFO, LRU, OPT, page fault, page replacement algorithm.*

Date of Submission: 22 January 2016 Date of Accepted: 05 February 2016

I. INTRODUCTION

The main objective of a computer system is to execute programs of different sizes. The programs should be in main memory during execution. Since main memory has limited size to accommodate large programs, the computer system must afford secondary storage to backup main memory in this case. A technique that permits the execution of the programs that may not be entirely in main memory is called Virtual memory [1]. The idea behind virtual memory is that the parts of the program which are required during execution are kept in the main memory and the rest on the secondary memory. Virtual memory is commonly implemented by demand paging [2]. Demand paging combines the concept of paging and swapping. The program memory is divided into pages and the available physical memory into frames. Whenever a process is to be executed, it is brought into main memory. Only those pages of a process are brought into the main memory that is required. If the accessed page is not in the main memory, then it is known as page fault. The selection of such a page is performed by page replacement algorithms which try to reduce the page fault rate at the least overhead [3]. Some unused page can be swapped out from the main memory into secondary memory.

There are different policies regarding how to select a page to be swapped out when a page fault occurs to create space for a new page. These policies are called page replacement algorithms. Several page replacement algorithms exist like Optimal Page Replacement, First in First out, Least Recently Used, Least Frequently Used, Most Frequently Used, Not Frequently Used, Second Chance Page Replacement, Clock Page Replacement etc. In this paper, we implement three algorithms i.e. Optimal Page Replacement, First in First out and Least Frequently Used using Visual Studio C#.

II. WORKING PROCEDURE OF PAGE REPLACEMENT ALGORITHMS

The Operating System has to choose a page to remove from memory for creating space for a new page that has to be brought in. Page Replacement Algorithms are used for the selection of the page to replace. In this paper, three page replacement algorithms are discussed.

2.1 FIRST IN FIRST OUT (FIFO) PAGE REPLACEMENT ALGORITHM

The First in First out (FIFO) page replacement algorithm is the simplest approach of replacing the page. The idea is to replace the oldest page in main memory from all the pages i.e. that page is replaced which has been in main memory for the greatest period of time. A FIFO queue can be created to hold all the pages in main memory [4]. The page that is at the front is replaced and when the page is fetched into memory it is inserted at the Rear end. For this reason, FIFO algorithm is seldom used [5].

Let us consider a 12 pages common reference string 1,2,1,3,4,1,5,4,3,2,5,4 for two, three and four page frames in memory to find the page faults using FIFO page replacement algorithm.
2 page frames:

Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	3	3	1	1	4	4	2	2	4
	2	2	2	4	4	5	5	3	3	5	5

Number of Faults:

F	F		F	F	F	F	F	F	F	F	F

In the above reference string with 2 page frames, the number of page faults is 11.

3 page frames:

Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	1	4	4	4	4	3	3	3	3
	2	2	2	2	1	1	1	1	2	2	2
			3	3	3	5	5	5	5	5	4

Number of Faults:

F	F		F	F	F	F		F	F		F

In the above reference string with 3 page frames, the number of page faults is 09.

4 page frames:

Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	1	1	1	5	5	5	5	5	5
	2	2	2	2	2	2	2	2	2	2	2
			3	3	3	3	3	3	3	3	3
				4	4	4	4	4	4	4	4

Number of Faults:

F	F		F	F		F					

In the above reference string with 4 page frames, the number of page faults is 05.

The advantage of FIFO page replacement algorithm is easy to implement and disadvantage is that it suffers from Belady's anomaly. Belady's anomaly is an unexpected result in FIFO page replacement algorithm. In some of the reference strings, increasing the size of the memory increases the page fault rate [6].

2.2 LEAST RECENTLY USED (LRU) PAGE REPLACEMENT ALGORITHM

Least Recently Used (LRU) Page Replacement Algorithm replaces the page in memory that has not been used for the longest period of time [7]. The problem with this algorithm is the complexity in implementation. The implementation may involve hardware or software support. The implementation of this policy can be possible using matrix, counters, linked list etc.

Consider the same reference string 1,2,1,3,4,1,5,4,3,2,5,4 with main memory that can accommodate 2, 3 and 4 page frames respectively to find the page faults using LRU page replacement algorithm.

2 page frames:

Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	1	4	4	5	5	3	3	5	5
	2	2	3	3	1	1	4	4	2	2	4

Number of Faults:

F	F		F	F	F	F	F	F	F	F	F

In the above reference string with 2 page frames, the number of page faults is 11.
3 page frames:
Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	1	1	1	1	1	3	3	3	4
	2	2	2	4	4	4	4	4	4	5	5
			3	3	3	5	5	5	2	2	2

Number of Faults:

F	F		F	F		F		F	F	F	F

In the above reference string with 3 page frames, the number of page faults is 09.
4 page frames:
Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	1	1	1	1	1	1	2	2	2
	2	2	2	2	2	5	5	5	5	5	5
			3	3	3	3	3	3	3	3	3
				4	4	4	4	4	4	4	4

Number of Faults:

F	F		F	F		F			F		

In the above reference string with 4 page frames, the number of page faults is 06.

The advantage of LRU page replacement algorithm is that it does not suffer from Belady's anomaly and the disadvantage is that it needs expensive hardware support or additional data structure to implement.

2.3 OPTIMAL (OPT) PAGE REPLACEMENT ALGORITHM

In Optimal page replacement algorithm, the page that will not be used for the longest period of time is replaced to make space for the requested page. This algorithm is easy to explain theoretically but difficult to implement because it requires to have accurate information of future events. Therefore its use is limited to serving as a benchmark to which other page replacement algorithms may be compared [8]. This algorithm never suffers from belady's anomaly [9].

Consider again the same reference string 1,2,1,3,4,1,5,4,3,2,5,4 with main memory that can accommodate 2, 3 and 4 page frames to find the page faults using OPT page replacement algorithm.
2 page frames:
Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	1	1	1	5	5	5	5	5	4
	2	2	3	4	4	4	4	3	2	2	2

Number of Faults:

F	F		F	F		F		F	F		F

In the above reference string with 2 page frames, the number of page faults is 08.
3 page frames:
Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	1	1	1	5	5	5	5	5	5
	2	2	2	4	4	4	4	4	4	4	4
			3	3	3	3	3	3	2	2	2

Number of Faults:

F	F		F	F		F			F		

In the above reference string with 3 page frames, the number of page faults is 06.
4 page frames:
Reference String:

1	2	1	3	4	1	5	4	3	2	5	4

Page Frames:

1	1	1	1	1	1	5	5	5	5	5	5
	2	2	2	2	2	2	2	2	2	2	2
			3	3	3	3	3	3	3	3	3
				4	4	4	4	4	4	4	4

Number of Faults:

F	F		F	F		F					

In the above reference string with 4 page frames, the number of page faults is 05.

The advantage of optimal page replacement algorithm is that it has least rate of occurrences of page fault and the disadvantage is that it is difficult to implement because it requires accurate knowledge of future events.

III. IMPLEMENTATION OF PAGE REPLACEMENT ALGORITHMS USING C# IN VISUAL STUDIO.NET

The implementation is carried out with C# in Visual Studio. The number of page faults of three algorithms viz. FIFO, LRU and OPT are calculated by entering the same number of pages, the same sequence and the same page frames in main memory. The purpose is to test the performance of the 3 algorithms with the same sequence. We have chosen a reference string of 12 pages having the sequence: 1, 2, 1, 3, 4, 1, 5, 4, 3, 2, 5, 4. Firstly we have tested it for two, three and four page frames. The screenshot for three page frames is shown in figure 1. In the application firstly it requires the number of empty frames, then it requires the length of the reference string and at last the reference string itself. The number of page faults can be calculated for all the three algorithms.

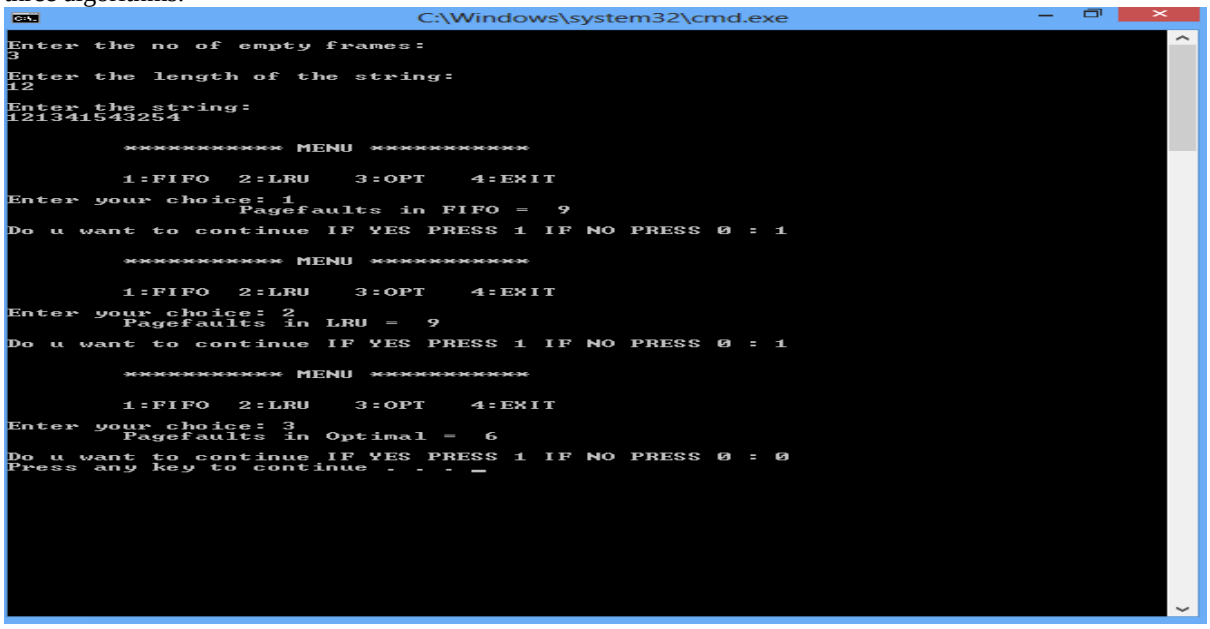

Figure 1: Screenshot of the application.

A table with number of page faults, for each algorithm, with page frame size 2, 3 and 4 is generated as shown in table 1. With 2 page frames, FIFO generates 11 page faults, LRU generates 11 page faults and Optimal generates 8 page faults. Similarly with 3 page frames, FIFO generates 9 page faults, LRU generates 9 page faults and Optimal generates 6 page faults. With 4 page frames, FIFO generates 5 page faults, LRU generates 6 page faults and Optimal generates 5 page faults.

Algorithm	FIFO	LRU	Optimal
Page Frames(2)	11	11	08
Page Frames(3)	9	09	06
Page Frames(4)	5	06	05
Average	8.33	8.66	6.33

Table 1: Page faults for frame size 2, 3 and 4.

The graphical representation is shown in figure 2. From the figure, it is very much clear that the optimal algorithm is the best among all the three page replacement algorithms.

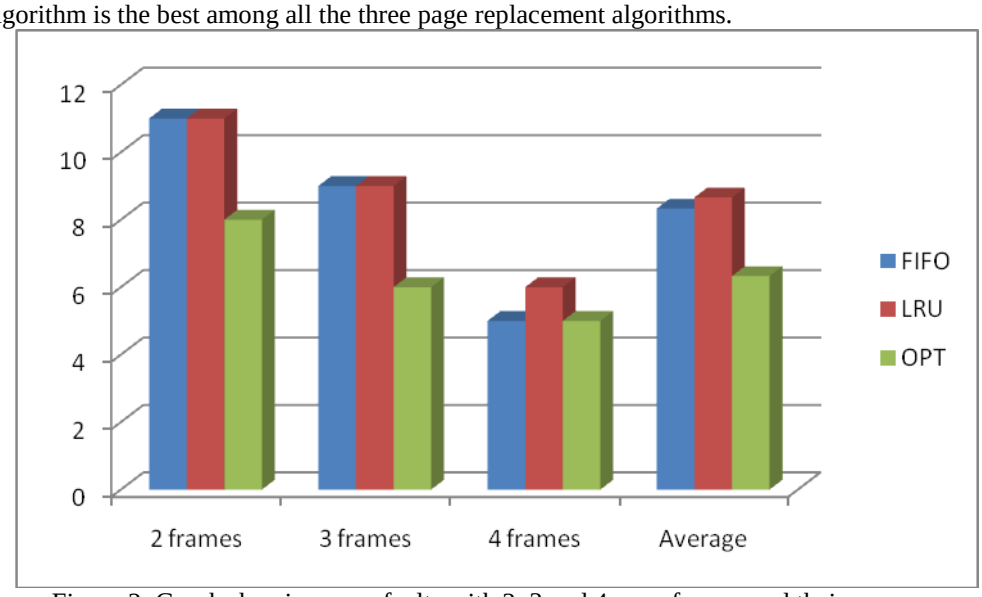

Figure 2: Graph showing page faults with 2, 3 and 4 page frames and their average.

IV. CONCLUSION

In the above study, we have found that optimal page replacement algorithm results the best algorithm because the average page faults in all the three cases with page frame size 2, 3 and 4 is less as compared to FIFO and LRU. A good page replacement algorithm reduces the number of page faults. In FIFO algorithm, some reference strings produces unexpected results called Belady's anomaly i.e. by increasing the number of page frames, the page fault also increases. In that case LRU works better than FIFO. In this paper, three different page replacement algorithms are studied and implemented with C# and compare with respect to page faults.

V. FUTURE WORK

In the future, we will use some other page replacement algorithms for the comparison. Apart from this, we will design a graphical user interface for comparing these algorithms.

REFERENCES

[1] Genta Rexha, Erand Elmazi and Igli Tafa, A Comparison of Three Page Replacement Algorithms: FIFO, LRU and Optimal, *Academic Journal of Interdisciplinary Studies*, *4*(2), 2015, 56-62.

[2] Hasan M H Owda et al., A Comparison of Page Replacement Algorithms in Linux Memory Management, *International Journal of Computer and Information Technology*, *3*(3), 2014, 565-569.

[3] Anvita Saxena, A Study of Page Replacement Algorithms, *International Journal of Engineering Research and General Science*, *2*(4), 2014, 385-388.

[4] Abraham Silberschatz, Peter B. Galvin and Greg Gagne, *Operating System Concepts* (UK: Wiley, 2010).

[5] Andrew S. Tanenbaum, *Modern Operating Systems* (USA: Prentice-Hall, Inc., 2001).

[6] Manisha Koranga and Nisha Koranga, Analysis on Page Replacement Algorithms with Variable Number of Frames, *International Journal Of Advanced Research in Computer Science and Software Engineering*, *4*(7), 2014, 403-411.

[7] William Stallings, *Operating Systems: Internals and Design Principles* (India: Pearson, 2009).

[8] J. Archer Harris, *Operating Systems* (Tata McGraw-Hill Edition, 2002).

[9] Mahesh Kumar M R and Renuka Rajendra B, AN INPUT ENHANCEMENT TECHNIQUE TO MAXIMIZE THE PERFORMANCE OF PAGE REPLACEMENT ALGORITHMS, *International Journal of Research in Engineering and Technology*, *4*(6), 2015, 302-307.