Analysis of FIFO, OPT, LRU Page Replacement Algorithms with Count Based Page Replacement Technique

Sri Lakshmi Kanagala
Dept of Computer Science and Engineering
Dadi Institute of Engineering and Technology, Visakhapatnam, Andhra Pradesh, India

Abstract- *The page replacement concept can be used in many areas of computer design, there are many page replacement algorithms such as optimal (OPT), first-in-first-out (FIFO), least recently used (LRU) are available in memory management. Different page replacement algorithms suggest different ways to decide which page to replace. The target for all page replacement algorithms is to reduce number of page faults. The existing three page replacement algorithms will be compared with the proposed count based replacement technique. The efficiency of this proposed algorithm is nearer to optimal. It can be used as a standard for other counting based page replacement algorithms.*

Keywords- Page replacement, LRU, Page fault

I. INTRODUCTION

Operating system uses different page replacement algorithms. There are many different page replacement algorithms. We evaluate an algorithm by running it on a particular string of memory reference and computing the number of page faults. Some page replacement algorithms are Optimal , First-In First-Out and Least recently used page replacement algorithms.

A. Optimal Page Replacement Algorithm (OPT)

An optimal page-replacement algorithm has the lowest page-fault rate of all algorithms. It has been called OPT or MIN .It produces the fewest possible page faults. It works by replacing the page that will not be used for the longest period of time. This is best possible page replacement algorithm and it is easy to describe but impossible to implement. The advantages are lowest page fault rate and never suffers from Belady's anomaly It is better than FIFO. The drawbacks are difficult to implement and it needs forecast i.e. Future knowledge.

B. First in First out Algorithm (FIFO)

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. The advantages are FIFO is easy to understand and it is very easy to implement. The drawbacks are not always good at performance. System needs to keep track of each frame and another unexpected side effect is the FIFO anomaly or Belady's anomaly. This anomaly says that the page fault rate may increase as the number of allocated page frames increases.

C. Least Recently Used (LRU)

Page which has not been used for the longest time in main memory is the one which will be selected for replacement. This is based on the principal of locality. This algorithm is easy to implement and to keep a list, replace pages by looking back into time. The advantages are LRU page replacement algorithm is quiet efficient and it does not suffer from Belady's Anomaly. The drawbacks are its implementation is not very easy and its implementation may require substantial hardware assistance.

LRU is considered a good replacement policy, and is often used. The problem is how exactly to implement it. There are two simple approaches commonly used:

Counters: Every memory access increments a counter, and the current value of this counter is stored in the page table entry for that page. Then finding the LRU page involves simple searching the table for the page with the smallest counter value.

Stack: Another approach is to use a stack, and whenever a page is accessed, pull that page from the middle of the stack and place it on the top. The LRU page will always be at the bottom of the stack. Because this requires removing objects from the middle of the stack, a doubly linked list is the recommended data structure.

Note that both implementations of LRU require hardware support, either for incrementing the counter or for managing the stack, as these operations must be performed for every memory access. Neither LRU or OPT exhibit Belady's anomaly. Both belong to a class of page-replacement algorithms called stack algorithms, which can never exhibit Belady's anomaly. A stack algorithm is one in which the pages kept in memory for a frame set of size N will always be a subset of the pages kept for a frame size of N + 1.

II. PROPOSED ALGORITHM

Many research works has been done to improve the performance of the Page Replacement algorithm. The proposed Page Replacement algorithm is based on the small change in Least Recently Used (LRU) page replacement algorithm, using a count median process. The number of page faults for this proposed algorithm is near to optimal algorithm.

The proposed algorithm is as follows,

- First we can find the size of reference string then to calculate the mid value. If the mid value is decimal then take rounded value.
- Using mid value to divide the reference string into two sub partitions and calculate the Count of pages in a given reference string for both left and right partitions separately.
- Next we have to check the number of frames. Initially all the frames are full and the page faults will occur, note that the related pages count values are decremented.
- When a new page enters, to check that the page is present or not. If it is, we should decrease the count of that page otherwise it is a page fault.
- A page fault will occur we have to replace that page using minimum count. Here to check that the page is present in left partition or right. If it is in left partition, to use left side count values otherwise to use right side count values.
- If an ambiguous case occurs using LRU algorithm to replace a page. Repeat this process until to reach the end of the reference string.

Example 1: The Reference string is 1,2,3,2,1,5,2,1,6,2,5,6,3,1,3,6,1,2,4,3

Mid value=10

1,2,3,2,1,5,2,1,6,2 | 5,6,3,1,3,6,1,2,4,3

Left reference string-1, 2, 3, 2, 1, 5, 2, 1, 6, 2
Count values c1=3, c_2=4, c_3=1, c_4=0, c_5=1, c_6=1
Right reference string -5, 6, 3, 1, 3, 6, 1, 2, 4, 3
Count values c1=2, c_2=1, c_3=3, c_4=1, c_5=1, c_6=2
Total number of frames: 3

No. of Frames	Reference String																			
	1	2	3	2	1	5	2	1	6	2	5	6	3	1	3	6	1	2	4	3
1	1	1	1			1			1		1		1					1	4	
2		2	2			2			2		5		3					3	3	
3			3			5			6		6		6					2	2	

Total number of page faults: 9

The number of page faults and comparison for all page replacement algorithms with proposed algorithm is as follows

Table 1: Number of page faults

Page Replacement Algorithms	Page Faults
OPT	9
FIFO	14
LRU	11
Proposed	9

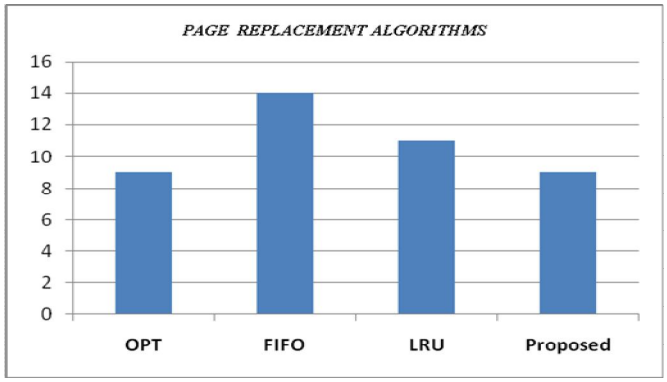

Figure:1 : Page Replacement Algorithms comparison

III. EXPERIMENTAL ANALYSIS

A sample reference string is taken and to calculate the number of page faults for all page replacement algorithms.

A .Test case 1: The Reference string is 1,2,3,2,5,6,3,4,6,3,7,3,1,5,3,6,3,4,2,3,4,5,1
Total number of frames: 3

Optimal page replacement

No. of Frames	Reference String																						
	1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	1	1		1	1		4			7		1	5				5	2				5	1
2		2	2		5	6		6			6		6	6				4	4				4	4
3			3		3	3		3			3		3	3				3	3				3	3

Total number of page faults: 13

FIFO page replacement

No. of Frames	Reference String																							
	1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	1	1		5	5		5		3	3		3	5	5	5		4	4		4		5	5
2		2	2		2	6		6		6	7		7	7	3	3		3	2		2		2	1
3			3		3	3		4		4	4		1	1	1	6		6	6		3		3	3

Total number of page faults: 17

LRU page replacement

No. of Frames	Reference String																							
	1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	1	1		5	5	5	4			7		7	5		5		4	4				4	4
2		2	2		2	2	3	3			3		3	3		3		3	3				3	1
3			3		3	6	6	6			6		1	1		6		6	2				5	5

Total number of page faults: 15

Proposed Algorithm

The Reference string is 1,2,3,2,5,6,3,4,6,3,7,3,1,5,3,6,3,4,2,4,3,4,5,1

Mid value=12, Total number of frames: 3

Left reference string-1,2,3,2,5,6,3,4,6,3,7,3 Count values $c1=1$, $c_2=2$, $c_3=4$, $c_4=1$, $c_5=1$, $c_6=2$, $c_7=1$

Right reference string -1,5,3,6,3,4,2,4,3,4,5,1 Count values $c1=2$, $c_2=1$, $c_3=3$, $c_4=3$, $c_5=2$, $c_6=1$, $c_7=0$

No. of Frames	Reference String																							
	1	2	3	2	5	6	3	4	6	3	7	3	1	5	3	6	3	4	2	4	3	4	5	1
1	1	1	1		5	5		4			7		1	1		6		4	4				4	4
2		2	2		2	6		6			6		6	5		5		5	2				5	5
3			3		3	3		3			3		3	3		3		3	3				3	1

Total number of page faults: 14

The comparison of all page replacement algorithms for the given reference string is as follows

Table 2: Number of page faults

Page Replacement Algorithms	Page Faults
OPT	13
FIFO	17
LRU	15
Proposed	14

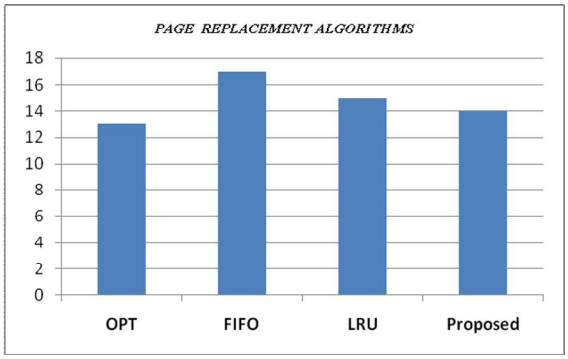

Figure:2 Page Replacement Algorithms comparison

B. Test case 2: The Reference string is 7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1

Total number of frames: 3

Optimal page replacement

No. of Frames	Reference String																				
	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
1	7	7	7	2		2		2			2			2				7		
2		0	0	0		0		4			0			0				0		
3			1	1		3		3			3			1				1		

Total number of page faults: 9

FIFO page replacement

No. of Frames	Reference String																			
	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
1	7	7	7	2		2	2	4	4	4	0			0	0			7	7	7
2		0	0	0		3	3	3	2	2	2			1	1			1	0	0
3			1	1		1	0	0	0	3	3			3	2			2	2	1

Total number of page faults: 15

LRU page replacement

No. of Frames	Reference String																			
	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
1	7	7	7	2		2		4	4	4	0			1		1		1		
2		0	0	0		0		0	0	3	3			3		0		0		
3			1	1		3		3	2	2	2			2		2		7		

Total number of page faults: 12

Proposed Algorithm

The Reference string is 7,0,1,2,0,3,0,4,2,3,0,3,2,1,2,0,1,7,0,1

Mid value=10 , so the reference string is 7,0,1,2,0,3,0,4,2,3 | 0,3,2,1,2,0,1,7,0,1

Left reference string-7,0,1,2,0,3,0,4,2,3 Count values $c_0=3$,$c_1=1$, $c_2=2$, $c_3=2$, $c_4=1$, $c_7=1$

Right reference string -0,3,2,1,2,0,1,7,0,1 Count values $c_0=3$,$c_1=3$, $c_2=2$, $c_3=1$, $c_4=0$, $c_7=1$

Total number of frames: 3

No. of Frames	Reference String																			
	7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
1	7	7	7	2		2		2			2			2				7		
2		0	0	0		0		4			0			0				0		
3			1	1		3		3			3			1				1		

Total number of page faults: 9

The comparison of all page replacement algorithms for the given reference string is as follows

Table 3: Number of page faults

Page Replacement Algorithms	Page Faults
OPT	9
FIFO	15
LRU	12
Proposed	9

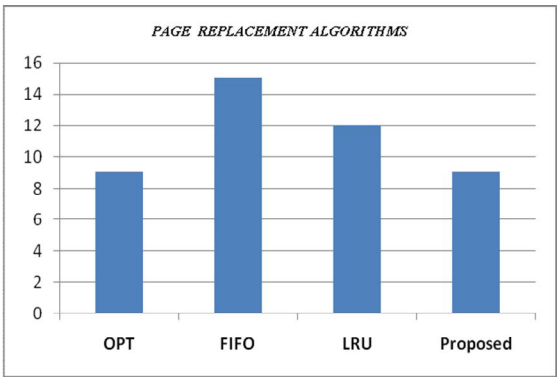

Figure: 3 Page Replacement Algorithms comparison

IV. CONCLUSION

The number of page faults for LRU is near to optimal page replacement. FIFO has more page faults. A good page replacement algorithm can reduce the number of page faults. In this analysis the page fault rate for count based technique comes exactly nearer or same as optimal even though they are both different algorithms. Hence it cannot be used in real life systems but can only be used to compare other algorithms.

REFERENCES

[1] Mr.C.C.Kavar, Mr. S.S.Parmar " Performance Analysis of LRU Page Replacement Algorithm with Reference to different Data Structure" International Journal of Engineering Research and Applications (IJERA) Vol. 3, Issue 1, January –February 2013, pp.2070-2076

[2] A. Janapsatya, A. Ignjatovic, J. Peddersen and S.Parameswaran, “Dueling CLOCK: Adaptive cache replacement policy based on the CLOCK algorithm”, Design,Automation and Test in Europe Conference and Exhibition, pp. 920-925, 2010

[3] Jisha.P. Abraham, Sheena Mathew " A novel approach to improve processor performance with page replacementtechniques" Proceedings of the International Conference on Information and Communication Technologies, ICICT 2014,3-5 December 2014

[4] Chavan A. Nayak K. R. Vora K. D. Purohit M. D. Chawan P. M. (2011). A comparison of page replacement algorithm. IACSIT International Journal of Engineering and Technology. Vol. 3. No. 2

[5] Amit S. Chavan, Kartik R. Nayak, Keval D. Vora, Manish D. Purohit and Pramila M. Chawan, " A Comparison of Page Replacement Algorithms", IACSIT International Journal of Engineering and Technology, Vol.3, No.2, April 2011 pp.171-174

[6] S. Jananee , “ Page Replacement” , International Journal of Computer Science and Information Technology Research, Vol. 2, Issue 3, pp: (90-99), July - September 2014

[7] Hasan M H Owda , Munam Ali Shah, Abuelgasim Ibrahim Musa, Manzoor Ilahi Tamimy " A Comparison of Page Replacement Algorithms in Linux Memory Management"International Journal of Computer and Information Technology (ISSN: 2279 – 0764) Volume 03 – Issue 03, May 2014 pp. 565-569

[8] A.S. Sumant,and P. M. Chawan, “Virtual Memory Management Techniques in 2.6 Linux kernel and challenges”, IASCIT International Journal of Engineering & Technology, pp.157-160, 2010.