International Conference on "The impact of emerging technologies on global societies: Environment, Ethics, Innovation and sustainability (IETGS 2025)" organized by Government Polytechnic College, Dewas, Madhya Pradesh on 27 Feb 2025.

A Comprehensive Study and Analysis of Page Replacement Algorithms

Tanay Kalmodiya
Department of Computer Science and Engineering
Madhav Institute of Technology & Science Gwalior, India
kalmodiyatanay@gmail.com

Abstract: **The page replacement algorithm selects pages to swap out of memory when new pages require memory for allocation. This study presents a detailed examination of page replacement algorithms, focusing on their design principles, operational mechanisms, and performance trade-offs. It evaluates classical algorithms such as FIFO, LRU, and Optimal, alongside modern variations, using simulation-based performance metrics. Comparative analysis highlights the impact of algorithm choice on system efficiency, memory utilization, and page fault rates. There have been a number of algorithms developed for the replacement of pages. The objective of each algorithm is to minimize the number of page faults. Process performance increases with a minimum number of page faults. Our study will compare page faults for different page frame sizes using the algorithms First in First out (FIFO), Least Recently Used (LRU) and Optimal Page Replacement (OPT).**

Keywords: Page replacement, algorithms, FIFO, LIFO, LRU, Optimal, Operating system.

1. INTRODUCTION

Memory management involves controlling and coordinating a computer's main memory. Memory management ensures that blocks of memory are properly managed and allocated so that the operating system (OS), applications, and other processes can run efficiently. Paging is one way to accomplish this [1].

Paging is a storage mechanism. Paging is the process that fetches processes from secondary storage to primary storage. Program memory is divided into pages and available physical memory is divided into frames. When a process attempts to run, it is brought into main memory. Only the pages of the process that are needed are brought into main memory. If the page is not located in main memory, it can be referred to as having a page fault. This selection of pages is done by a page replacement algorithm that attempts to keep page fault rates as low as possible [2]:

- Disk I/O is very expensive, so designing appropriate algorithms to solve these problems is an important task.
- Even small improvements in demand paging methods can lead to significant improvements in system performance

2. BELADY'S ANOMALY

The Belady's anomaly is the name given to the phenomenon where an increase in the number of page frames leads to an increase in the number of page faults in a certain memory access pattern [3].

Example:

The system has no pages loaded into memory and uses a FIFO replacement algorithm and the reference string is 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Case-1:

1	1	1	2	3	4	1	1	1	2	5	5
	2	2	3	4	1	2	2	2	5	3	3
		3	4	1	2	5	5	5	3	4	4
PF	PF	PF	PF	PF	PF	PF	X	X	PF	PF	X

3 frames, 9 page faults

Case-2:

1	1	1	1	1	1	2	3	4	5	1	2
	2	2	2	2	2	3	4	5	1	2	3
		3	3	3	3	4	5	1	2	3	4
			4	4	4	5	1	2	3	4	5
PF	PF	PF	PF	X	X	PF	PF	PF	PF	PF	PF

4 frames, 10 page faults

Because the number of page frames assigned to each stack-based algorithm is not related to the quantity of frame per second, all such algorithms are not subject to Belady's anomaly.

Optimal, LRU, and LFU are some of the policies that can be used as examples.

The notion that a page is seldom used when it remains dormant for an extended period is the foundation of these algorithms. Therefore, it would be best to forget about this page.The possibility of self-management and the elimination of Belady's anomalies are made possible by this mechanism [4]

Advantage:

1) The Belady anomalies can provide information on the performance and operation of the page replacement algorithm. Developing and optimizing algorithms for particular scenarios can be made easier with this.

2) Sometimes, a process can increase its frame count to improve algorithm performance even though it may cause more page faults. The search rate can be reduced by increasing the number of frames, which can improve the overall performance.

Disadvantages:

1) Unpredictable performance and system instability can result from the Belady anomalies, which makes it challenging for the algorithm to function under various frame and page configurations.

2) In certain situations, a process may be required to allocate more frames, which can result in overhead and resource usage that are detrimental to the system's performance.

3) Users and administrators may find it be perplexing due to Belady's anomaly, which causes an increase in page faults when more frames are allocated to a process. This behavior is not intuitive at all.

4) In certain contexts, the optimization of page replacement algorithms can be challenging due to the unpredictable behavior of algorithm operators. This is a result of the Belady anomaly.

3. WORKING OF PAGE REPLACEMENT ALGORITHMS

The operating system must select pages to remove from memory to make room for inserting new pages via page replacement algorithm [5].

Find a free frame in the absence of any frame.

You can free a frame by writing its contents to swap space and changing the page table to reflect that the page is no longer in memory.

You can now use freed frames to save pages that encounter errors in the process

3.1) First In First Out (FIFO)

This is the simplest algorithm to use for replacing pages.

With this algorithm, the operating system keeps track of all the pages in memory in a queue, with the oldest page being at the front of the queue [6].

If a page needs to be replaced, the page at the top of the queue is selected for deletion

Example:

Memory reference string with three frames and the reference string 6, 1, 1, 2, 0, 3, 4, 6, 0, 2, 1, 6, 1, 1, 2, 0, 3, 4, 6, 0, 2, 1, 2, 1, 2, 0, 3, 2, 1, 2, 0

S. no	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
F3				2	2	2	4	4	4	2	2	2	2	2	2	2	2	2	2	2
F2		1	1	1	1	3	3	3	0	0	0	0	0	0	0	0	0	1	1	1
F1	6	6	6	6	0	0	0	6	6	6	1	1	1	1	1	3	3	3	3	0
Hit (H)/ Fault (F)	F	F	H	F	F	F	F	F	F	F	F	H	H	H	H	F	H	F	H	F

2, 1, 2, 0, 3, 2, 1, 2, 0
Number of Page Hits = 8
Number of Page Faults = 12

Algorithm:

1) The process begins with creating an empty page table and assigning a fixed number of page frames to main memory.
2) When a process requests access to, it is done through requesting page access.
3) Check if the page table is already inside the specified page frame.
4) Continuing with the next access request when the page is already in the frame (Page Hit).
5) If the page is not in the main memory and it's not within the current page frame, use the oldest page from the same location instead.
6) Modify the page from main memory by altering it.
7) Replace the old page with a new one without any charge for the frame.
8) Reorganize the page table to accommodate significant memory modifications.
9) Return to step 2 and ask for permission to return to the next page.

6, 1, 1, 2, 0, 3, 4, 6, 0, 2, 1, 2, 1, 2, 0, 3, 2, 1, 2, 0

S. no	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
F3				2	2	2	4	4	4	2	2	2	2	2	2	2	2	2	2	0
F2		1	1	1	1	3	3	3	0	0	0	0	0	0	0	3	3	3	3	3
F1	6	6	6	6	0	0	0	6	6	6	1	1	1	1	1	1	1	1	1	1
Hit (H)/ Fault (F)	F	F	H	F	F	F	F	F	F	F	F	H	H	H	H	F	H	H	H	F

Advantage:

Some advantages of the FIFO page replacement algorithm are listed below:

1) FIFO is an uncomplicated algorithm that is easy to understand and implement

2) There is minimal overhead involved in storing page reference information by the algorithm, and no additional data structures are needed to do so.

Disadvantages:

The FIFO page replacement algorithm has several drawbacks. These include:

1) Poor performance can be experienced by a FIFO search algorithm, particularly when there are numerous page faults.
2) A problem known as Belady's anomalies can arise from the use of the FIFO search algorithm, which can cause an increase in page faults with an increasing number of frames.
3) Due to the FIFO algorithm's inability to account for page frequency, frequently used pages can be replaced with less frequently utilized ones.

3.2) Least Recently Used (LRU)

This algorithm replaces pages that have not been used for a long time. The problem with this algorithm is implementation complexity. Implementation may include hardware or software support. Implementation of this policy is possible using matrices, counters, linked lists, etc [7,8]

Example

Memory reference string with three frames and the reference string 6, 1, 1, 2, 0, 3, 4, 6, 0, 2, 1, 2, 1, 2, 0, 3, 2, 1, 2, 0

Number of Page Hits = 7
Number of Page Faults = 13

Algorithm:

1) Create an initial instance by utilizing the page table as empty space and assigning a fixed number of page frames to the main memory.

2) The process may request access to a page.

3) Examine the Page Table to ascertain if the page is already inside the frame.

4) If the page is in a page frame (Page Hit), update the information about when this page was most recently visited.

5) The main memory should be used to replace the most recently used page if it is not in a page frame.(Page Hit)

6) The page that needs to be replaced will be removed from the main memory.

7) Move the new page into its place inside the empty page frame.

8) Adapt the page table by updating it to match changes in main memory.
9) Newly added page reaches last access, update information.

10) Revisit step 2 for a subsequent request to be granted access on the following page, again.

Advantage:

Some advantages of the LRU page replacement algorithm are listed below:

1) LRU page replacement algorithm is a good choice for reducing page faults, and its performance is generally satisfactory.
2) The LRU page replacement algorithm does not cause the same problem as its FIFO counterpart, which is increased page faulting with an increase in the number of frames. This technique prevents this from happening.

3) The LRU page replacement algorithm replaces the least used pages by taking into account their page usage frequency.

Disadvantages:

The LRU page replacement algorithm has several drawbacks. These include:

1) The implementation of the LRU page replacement algorithm is a formidable challenge, particularly when using linked lists or stacks, which require additional data structures to store page reference information.
2) The LRU page replacement algorithm may have more load compared to other page replacement algorithms due to the need to track page usage information.
3) Inconsistencies in page usage patterns may cause the LRU page replacement algorithm to perform worse than its other algorithms, such as the optimal one.

3.3) Optimal Page replacement

This algorithm replaces pages that are not used for a long time in the future. One of the consequences of discovering the Belady anomaly was the search for an optimal page replacement algorithm. The optimal page replacement algorithm has the lowest page fault rate of all algorithms and does not suffer from Belady anomalies [8]

Example

Memory reference string with three frames and the reference string 6, 1, 1, 2, 0, 3, 4, 6, 0, 2, 1, 2, 1, 2, 0, 3, 2, 1, 4, 0

6, 1, 1, 2, 0, 3, 4, 6, 0, 2, 1, 2, 1, 2, 0, 3, 2, 1, 4, 0

S. no	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
F3				2	2	2	4	4	4	4	4	4	4	4	4	3	3	3	4	4
F2		1	1	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1
F1	6	6	6	6	0	0	0	6	6	2	2	2	2	2	2	2	2	2	2	0
Hit (H)/ Faul t (F)	F	F	H	F	F	F	F	H	H	F	F	H	H	H	H	F	H	F	F	F

Number of Page Hits = 8
Number of Page Faults = 12

Algorithm:

1) Set the initialization process by creating an empty table and assigning a fixed number of page frames to the main memory.
2) A page is requested to be accessible by a process.
3) Examine the Page Table to ascertain if the page is already inside the frame.
4) The next access request should be made if the page is located in a page frame.
5) In case the page is not within the designated page frame, choose a suitable page to be replaced.
6) Change the page from its main memory and make changes to your preferred page.
7) Replace the old page with a new one without any charge for the frame.
8) Reorganize the page table to accommodate significant memory modifications.

9) Revisit step 2 for a subsequent request to be granted access on the following page, again.

Advantage:

Some advantages of the optimal page replacement algorithm are listed below:

1) Effective performance is typically attained by an optimal retrieval algorithm when it replaces pages that are not used for a prolonged period.
2) Avoid Belady's anomalies by using the best search algorithm, which can help reduce page faults caused by increases in the number of frames.
3) An algorithm designed to optimize page replacement takes into consideration the future use of pages and accurately predicts which pages will be used in the near future.

Disadvantages:

The optimal page replacement algorithm has several drawbacks. These include:

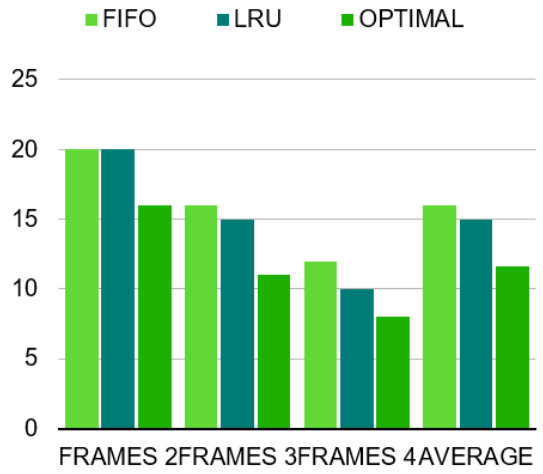
1) A perfect thinning algorithm cannot be implemented as it necessitated knowledge about the page's usage in future, which is not provided.
2) The optimal page replacement algorithm is primarily theoretical and not practical for operating systems, but rather designed to evaluate performance. It has several useful algorithms available on the market.
3) The optimal page replacement algorithm has no real implementation and is utilized solely for comparison with other algorithms.

4. IMPLEMENTATION OF PAGE REPLACEMENT ALGORITHM

The number of page faults for the algorithm was calculated by putting the same number of pages, same order, and same page frames into main memory. The objective is to test the performance of the three algorithms on the same sequence.

Algorithm	**FIFO**	**LRU**	**Optimal**
Page Frames (2)	20	20	16
Page Frames (3)	16	15	11
Page Frames (4)	12	10	8
Average	16	15	11.6

For 2 page frames, FIFO generates 20 page faults, LRU generates 20 page faults, and Optimal generates 16 page faults. Similarly, for a 3-page frame: FIFO generates 16 page faults, LRU generates 15 page faults, and Optimal generates 11 page faults. For a 4-page frame, FIFO generates 12 page faults, LRU generates 10 page faults, and Optimal generates 8 page faults [9].



5. CONCLUSION

The above study found that the optimal page replacement algorithm leads to the best algorithm as it has less average page faults in all three cases of page frame size 2, 3 and 4 compared to FIFO and LRU. A good page replacement algorithm reduces the number of page faults. In the FIFO algorithm, some reference strings cause unexpected results called Belady anomalies I.e. As the number of page frames increases, page errors also increase LRU works better than FIFO in this case

REFERENCES

[1]. Aye Aye Cho (2018). Analysis of page replacement algorithm using C++. International Journal of Advance Research and Development. Vol 3(7).

[2]. Sourabh Shastri, Anand Sharma, and Vibhakar Mansotra (2016). Study of Page Replacement Algorithms and their analysis with C#. The International Journal of Engineering and Science (IJES), Vol. 5(1).

[3]. Genta Rexha, Erand Elmazi and Igli Tafa (2015). A Comparison of Three Page Replacement Algorithms: FIFO, LRU and Optimal. Academic Journal of Interdisciplinary Studies. Vol 4(2), Doi:10.5901/ajis.2015.v4n2s2p56

[4]. Heikki Paajanen (2007). "Page replacement in operating system memory management" Master's thesis, University of Jyvaskyla

[5]. Hitha Paulson and Rajesh R. (2017). Page Replacement Algorithms – Challenges and Trends. International Journal of Computer & Mathematical Sciences IJCMS. Vol 6(9).

[6]. Enayet Kabir, Md. Muniruzzaman, Most.Masuma Akther and Md. Mahbub Alam (2020). Advancement and Analysis of Proficient Page Replacement Algorithm". American Journal of Engineering Research (AJER), vol. 9(03), 2020, pp. 193-197.

[7]. S. Muthusundari, Berlin M.A., J. Geetha Priya, Balasaranya Kirubakaran (2020). A buffer-based page replacement algorithm to reduce page fault. DOI: http://dx.doi.org/10.1016/j.matpr.2020.08.182

[8]. Hasan M H Owda , Munam Ali Shah, Abuelgasim Ibrahim Musa, Manzoor Ilahi Tamimy (2014). A Comparison of Page Replacement Algorithms in Linux Memory Management. International Journal of Computer and Information Technology, 3(3), 2014, 565-569

[9]. Manisha Koranga and Nisha Koranga (2014). 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