Page Replacement Algorithms (PRA) Visualiser

Nirbhay Sharma (B19CSE114), Mayank-Raj (B19CSE053), Mayank Singh Rajput (B19CSE054), Mohit Ahirwar (B19CSE055), Ram Khandelwal (B19CSE116)

Index Terms—React, Firebase, HTML, CSS, Javascript, Page replacement algorithms, LRU, FIFO, Optimal, NRU, Working set, Second chance page replacement algorithm.

✦

1 Introduction & Objective

We all know that virtual memory is lifeline of modern operating systems. Whenever a process is asking or referring to a page and that page is absent in memory so a page fault arises. Hence, page replacement algorithms are very important for virtual memory management.

Tabular and graphical representation of page replacement algorithms are very helpful to understand and grasp concept very well. Hence we come up with an app with same. This app is suitable for desktop, laptop, mobile, tablet, etc because we have used a progressive and responsive framework of React to make it user friendly and easy to use.

Web App Link and Codebase can be found here :

- https://pra-visualizer.web.app/
- https://github.com/nirbhay-design/pra-visualizer

In this project our major contributions are as follows

- We have designed a website for visualizing page replacement algorithms
- We have used ReactJS for implementing pleasing UI animations and the same has been deployed on Firebase platform

The rest of the paper is organized as follows. In section 2, we discuss the Theory of Page replacement algorithms. Methodology is presented in section 3 and the results are presented in section 4. Finally, we conclude the report in section 5.

2 Theory

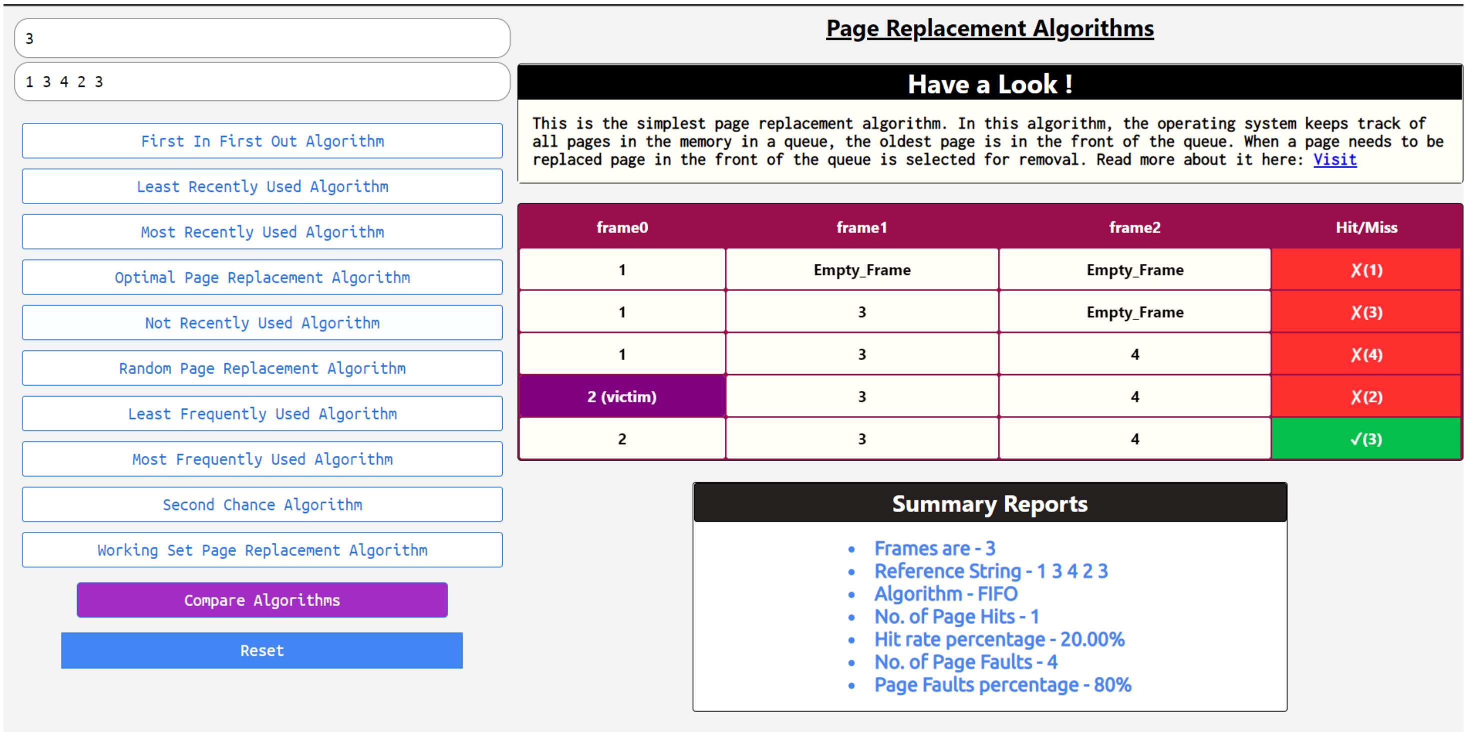
In case when CPU demands a page, then there may be some chance of page faults i.e. page is not present in the main memory and in that case we need to bring the page from the secondary memory and in such a scenario if all the frames in the memory are occupied then we require page replacement algorithms, which basically tries to finds a victim page and then replace it with the demanded page. following are the page replacement algorithms that we have implemented for our web application :

2.1 Algorithms

- **First In First Out (FIFO)** : The page which is assigned the frame first will be replaced first. In other words, the page which resides at the rare end of the queue will be replaced on the every page fault.
- **Least Recently Used (LRU)** : This algorithm replaces the page which has not been referred for a long time.
- **Most Recently Used (MRU)** : MRU page replacement algorithm replaces the most recently used page.
- **Optimal** : This algorithms replaces the page which will not be referred for so long in future. Although it can not be practically implementable.
- **Not Recently Used (NRU)** : This algorithm removes a page at random from the lowest numbered non-empty class.
- **Random** : As the name suggests, this algorithm randomly replaces any page.
- **Least Frequently Used (LFU)** : It select victim page which is least frequently used which means it keeps track of the frequency of pages so far and on page fault it replaces the page having least frequency so far.
- **Most Frequently Used (MFU)** : It select victim page which is most frequently used which means it keeps track of the frequency of pages so far and on page fault it replaces the page having maximum frequency so far.
- **Second Chance** : We provide a second chance to the page using reference bit once a page hit is there we make the ref bit of of that page as 1 if it is 0, and again if this page needs to be replaced then we provide it a second chance by making it ref bit as 0 and replace some other page.
- **Working Set** : In working set algorithm we maintain a working set of last few page numbers and accordingly we replace the pages.

3 Methodology

We have implemented all the algorithms in Javascript in order to connect to the front end part i.e, ReactJS. We have used plenty of animations to help visualize things better like the Frame Table and Comparative Analysis of various algorithms. We have taken special care for Randomized algorithms like Random Page Replacement and Not Recently Used (NRU) so that there is no fixed replacement hence it is randomized in front end. We have successfully deployed our wep application to the Firebase so that it can be used from any device.


First In First Out Algorithm
Least Recently Used Algorithm
Most Recently Used Algorithm
Optimal Page Replacement Algorithm
Not Recently Used Algorithm
Random Page Replacement Algorithm
Least Frequently Used Algorithm
Most Frequently Used Algorithm
Second Chance Algorithm
Working Set Page Replacement Algorithm
Compare Algorithms
Reset

3

1 3 4 2 3

Page Replacement Algorithms

Have a Look !

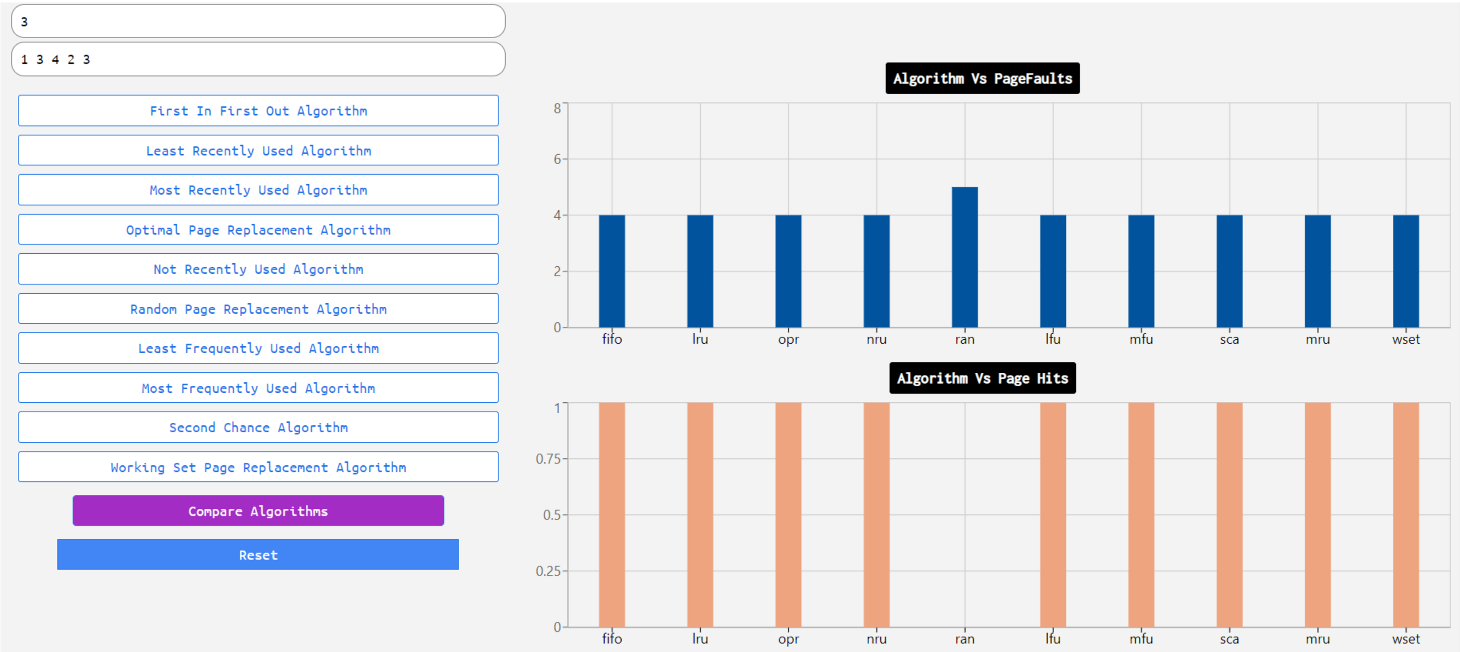
This is the simplest page replacement algorithm. In this algorithm, the operating system keeps track of all pages in the memory in a queue, the oldest page is in the front of the queue. When a page needs to be replaced page in the front of the queue is selected for removal. Read more about it here: Visit

frame0	frame1	frame2	Hit/Miss
1	Empty_Frame	Empty_Frame	✗(1)
1	3	Empty_Frame	✗(3)
1	3	4	✗(4)
2 (victim)	3	4	✗(2)
2	3	4	✓(3)

Summary Reports

- Frames are - 3
- Reference String - 1 3 4 2 3
- Algorithm - FIFO
- No. of Page Hits - 1
- Hit rate percentage - 20.00%
- No. of Page Faults - 4
- Page Faults percentage - 80%

Fig. 1: Description and Visualization of Various Algorithms


First In First Out Algorithm
Least Recently Used Algorithm
Most Recently Used Algorithm
Optimal Page Replacement Algorithm
Not Recently Used Algorithm
Random Page Replacement Algorithm
Least Frequently Used Algorithm
Most Frequently Used Algorithm
Second Chance Algorithm
Working Set Page Replacement Algorithm
Compare Algorithms
Reset
Algorithm Vs PageFaults
8
6
4
2
0
fifo
lru
opr
nru
ran
lfu
mfu
sca
mru
wset
Algorithm Vs Page Hits
1
0.75
0.5
0.25
0
fifo
lru
opr
nru
ran
lfu
mfu
sca
mru
wset

3

1 3 4 2 3

Fig. 2: Comparative Analysis

4 Results

Input to our website is 1) Number of frames in the main memory 2) reference string that CPU demands and on selecting the algorithms, the simulation starts and it will show the page faults / page hit and also the victim page selected due to algorithm and in the end it will also show a summary report showing page hits / faults, hit percentage, page fault's percentage and so on, one can also read about the algorithms which are given above, the results for that can be found at Fig. 1.

One can also visualize the comparison among algorithms using compare algorithm feature and get a clear visualization of page fault vs algorithms and page hit vs algorithms, the reference for that can be found at Fig. 2.

5 Conclusion & Future work

The following website will help the students to visualize the page replacement algorithms in a better manner and will help in clearing their concepts of some of the typical page replacement algorithms like second chance and working set page replacement algorithms.

One Future work can be to include the functionality to visualize Belady's anomaly in FIFO and second chance, this requires an extra database so that it can keep storing the inputs for the constructing the graph of Number of page faults VS Number of frames.

6 Acknowledgement

The members of the team would like to thank the instructor, Dr. Suchetana Chakraborty, for giving us the opportunity to work on a Page replacement algorithms project for helping the students to visualize the algorithms in a better way.

References

[1] Paulson, Hitha and Ramachandran, Dr Rajesh, Page Replacement Algorithms – Challenges and Trends (February 22, 2018). International Journal of Computer Mathematical Sciences IJCMS, Volume 6, Issue 9 September 2017, Available at SSRN: https://ssrn.com/abstract=3128697

[2] 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

[3] React_Documentaion:
https://reactjs.org/docs/getting-started.html

[4] Firebase:
https://firebase.google.com/