

AHMEDABAD UNIVERSITY

SCHOOL OF ENGINEERING AND APPLIED SCIENCE

---

# Page Replacement Algorithm

---

*Project By:*

Anuj SHAH

Charvik PATEL

Himanshu BUDHIA

Maharsh PATEL

*Instructor:*

Dr. Sanjay CHOUDHARY

*Mentor:*

Purnima SHAH

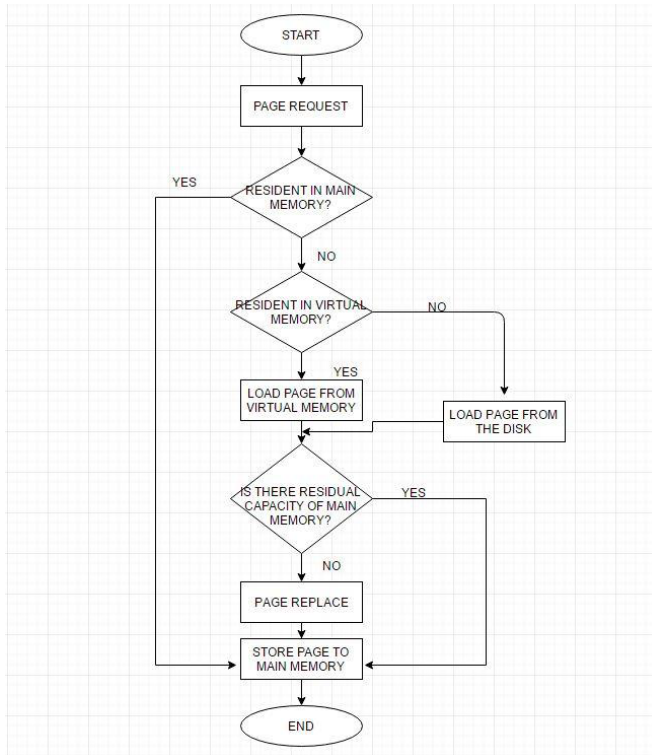
December 11, 2016



# 1 Brief Description

The basic idea of Page Replacement is; if there is a free page in the memory then use it. If there is no free page available, then we select one victim page which we would swap out of the virtual memory to the disk. Then we read the desired page to the new free frame. The page tables are then updated and the process is restarted. The main objective of a good replacement algorithm is to achieve a low page fault rate. For that we need to insure that the heavily used pages stay in the memory, and the replaced page should not be needed for some time. Secondly, we need to reduce the latency of a page fault. That can be assured by writing an efficient code, and replace the pages that do not need to be written out.

# 2 Flow Chart



### 3 Algorithm Implementation

#### 3.1 FIFO

This Page Replacement Algorithm treats the page frames allocated to a process as a circular buffer:

1. when the buffer is full, the oldest page is replaced. Hence First-in First-out
2. we require only one pointer, that circles through the page frames of the process.

The OS keeps the tracks of all the pages in memory in a queue, with most recent arrival at the back, and the oldest arrival in the front. When a page needs to be replaced, the page at the front of the queue(oldest) is selected.

Time	0	1	2	3	4	5	6	7	8	9	10
Requests		<i>c</i>	<i>a</i>	<i>d</i>	<i>b</i>	<i>e</i>	<i>b</i>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
Page Frames	0	<i>a</i>									
1	<i>b</i>										
2	<i>c</i>										
3	<i>d</i>										
Faults											
Time	0	1	2	3	4	5	6	7	8	9	10
Requests		<i>c</i>	<i>a</i>	<i>d</i>	<i>b</i>	<i>e</i>	<i>b</i>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>
Page Frames	0	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>e</i>	<i>e</i>	<i>e</i>	<i>e</i>	<i>e</i>	<i>d</i>
1	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>
2	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>b</i>	<i>b</i>	<i>b</i>
3	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>c</i>	<i>c</i>
Faults						•		•	•	•	•

### **3.2 LFU**

In this Page Replacement Algorithm, we assign a counter to every page. All the counters are initialized by 0. Each time a reference is made to that page, the counter value is incremented by one. When the cache reaches the capacity, and a request for a page that is not present in the memory is made, the system will search for the page with the least number of references (counter value) and replace it from the cache.

### **3.3 LRU Counter**

1. Every page entry has a counter.
2. Every time a page is referenced, increment a global counter and store it in the page counter.
3. For replacement, search the page with the lowest counter value and replace it.

### **3.4 LRU Stack**

1. Maintain a stack of page numbers in a doubly linked list.
2. When a reference to a page is made, move it to the top of the stack.
3. For replacement, choose the bottom page.

Time	0	1	2	3	4	5	6	7	8	9	10
Requests		c	a	d	b	e	b	a	b	c	d
Page Frames	0	a									
	1	b									
	2	c									
	3	d									
Faults											
Time page last used											
Time	0	1	2	3	4	5	6	7	8	9	10
Requests		c	a	d	b	e	b	a	b	c	d
Page Frames	0	a	a	a	a	a	a	a	a	a	a
	1	b	b	b	b	b	b	b	b	b	b
	2	c	c	c	c	e	e	e	e	e	d
	3	d	d	d	d	d	d	d	d	c	c
Faults						•				•	•
Time page last used						a=2 b=4 c=1 d=3		a=7 b=8 e=5 d=3	a=7 b=8 e=5 c=9		
Time	0	1	2	3	4	5	6	7	8	9	10
Requests		c	a	d	b	e	b	a	b	c	d
Page Frames	0	a	a	a	a	a	a	a	a	a	a
	1	b	b	b	b	b	b	b	b	b	b
	2	c	c	c	c	e	e	e	e	e	d
	3	d	d	d	d	d	d	d	d	c	c
Faults						•				•	•
LRU page stack											
Page to replace											
Time	0	1	2	3	4	5	6	7	8	9	10
Requests		c	a	d	b	e	b	a	b	c	d
Page Frames	0	a	a	a	a	a	a	a	a	a	a
	1	b	b	b	b	b	b	b	b	b	b
	2	c	c	c	c	e	e	e	e	e	d
	3	d	d	d	d	d	d	d	d	c	c
Faults						•				•	•
LRU page stack											
Page to replace											

Step of LRU

### 3.5 MFU

In this Page Replacement Algorithm, we assign a counter to every page. All the counters are initialized by 0. Each time a reference is made to that page, the counter value is incremented by one. When the cache reaches the capacity, and a request for a page that is not present in the memory is made, the system will search for the page with the most number of references (counter value) and replace it from the cache.

1	2	3	4	1	2	5	1	2	3	4	5
1 <sub>1</sub>	1 <sub>1</sub>	1 <sub>1</sub>	1 <sub>1</sub>	1 <sub>2</sub>	1 <sub>2</sub>	5 <sub>1</sub>	5 <sub>1</sub>	5 <sub>1</sub>	5 <sub>1</sub>	4 <sub>1</sub>	4 <sub>1</sub>
	2 <sub>1</sub>	2 <sub>1</sub>	2 <sub>1</sub>	2 <sub>1</sub>	2 <sub>2</sub>	2 <sub>2</sub>	1 <sub>1</sub>	1 <sub>1</sub>	1 <sub>1</sub>	1 <sub>1</sub>	5 <sub>1</sub>
		3 <sub>1</sub>	3 <sub>1</sub>	3 <sub>1</sub>	3 <sub>1</sub>	3 <sub>1</sub>	3 <sub>1</sub>	2 <sub>1</sub>	2 <sub>1</sub>	2 <sub>1</sub>	2 <sub>1</sub>
			4 <sub>1</sub>	4 <sub>1</sub>	4 <sub>1</sub>	4 <sub>1</sub>	4 <sub>1</sub>	4 <sub>1</sub>	3 <sub>1</sub>	3 <sub>1</sub>	3 <sub>1</sub>

MFU

Total Page Fault:10

### 3.6 OPT

1. An optimal page-replacement algorithm has the lowest page-fault rate.
2. Replace the page that has not been referenced for the longest period of time.

Page reference stream:

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	6	6	6	6	6	6	6	6	6	2	2	2
	2	2	2	2	2	2	2	2	2	2	2	2	1	1	1	1	1	4	4
		3	3	3	5	5	5	5	5	5	5	3	3	3	3	3	3	3	3
*	*	*			*			*				*	*				*	*	

Optimal

Total 9 page faults

### 3.7 Second Chance

1. This algorithm is a FIFO replacement algorithm with a small modification that causes it to approximate to LRU.
2. When a page is selected according to FIFO order, we check its reference bit. If it is set (the page was referenced), we clear it and look for another page.

Time	0	1	2	3	4	5	6	7	8	9	10
Requests		<i>c</i>	<i>a<sup>w</sup></i>	<i>d</i>	<i>b<sup>w</sup></i>	<i>e</i>	<i>b</i>	<i>a<sup>w</sup></i>	<i>b</i>	<i>c</i>	<i>d</i>
Page Frames	0	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>						
1	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>						
2	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>						
3	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>						
Faults											

Page table entries for resident pages:	10	<i>a</i>									
	10	<i>b</i>									
	10	<i>c</i>									
	10	<i>d</i>									

Time	0	1	2	3	4	5	6	7	8	9	10
Requests		<i>c</i>	<i>a<sup>w</sup></i>	<i>d</i>	<i>b<sup>w</sup></i>	<i>e</i>	<i>b</i>	<i>a<sup>w</sup></i>	<i>b</i>	<i>c</i>	<i>d</i>
Page Frames	0	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>	<i>a</i>
1	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>d</i>
2	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>c</i>	<i>e</i>	<i>e</i>	<i>e</i>	<i>e</i>	<i>e</i>	<i>e</i>
3	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>d</i>	<i>c</i>	<i>c</i>
Faults						*				*	*

Page table entries for resident pages:	10	<i>a</i>									
	10	<i>b</i>									
	10	<i>c</i>									
	10	<i>d</i>									

	11	<i>a</i>	00	<i>a<sup>*</sup></i>	00	<i>a</i>	11	<i>a</i>		11	<i>a</i>	00	<i>a<sup>*</sup></i>
	11	<i>b</i>	00	<i>b<sup>*</sup></i>	10	<i>b</i>	10	<i>b</i>		10	<i>b</i>	10	<i>d</i>
	10	<i>c</i>	10	<i>e</i>	10	<i>e</i>	10	<i>e</i>		10	<i>e</i>	00	<i>e</i>
	10	<i>d</i>	00	<i>d</i>	00	<i>d</i>	00	<i>d</i>		10	<i>c</i>	00	<i>c</i>

## 4 Test Result:

### 4.1 FIFO

```
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ gcc FIFO.c
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ ./a.out
The number of page requested are:8
The pages present in the requested string are
7 4 2 1 7 1 2 9
The elements in the frame are
7
The elements in the frame are
7
4
The elements in the frame are
7
4
2
The page replaced is 7
The elements in the frame are
1
4
2
The page replaced is 4
The elements in the frame are
1
7
2
The page replaced is 2
The elements in the frame are
1
7
9
The number of page faults are 6
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$
```

### 4.2 LFU

```
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ gcc LFU.c
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ ./a.out
The number of page request are:8
The pages present in the requested string are
7 4 2 1 7 1 2 9
The elements in the frame are
7
The elements in the frame are
7
4
The elements in the frame are
7
4
2
The page replaced is 7
The elements in the frame are
1
4
2
The page replaced is 1
The elements in the frame are
7
4
2
The page replaced is 7
The elements in the frame are
1
4
2
The page replaced is 1
The elements in the frame are
9
4
2
The number of page faults are 7
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$
```



## 4.3 LRU Counter

```
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ gcc LRUCTR.c
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ ./a.out
The number of page request are:8
The pages present in the frame are
7 4 2 1 7 1 2 9
The elements in the frame are
7
The elements in the frame are
7
4
The elements in the frame are
7
4
2
The page replaced is 7
The elements in the frame are
1
4
2
The page replaced is 4
The elements in the frame are
1
7
2
The page replaced is 7
The elements in the frame are
1
9
2
The number of page faults are 6
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$
```

## 4.4 LRU Stack

```
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ gcc LRUCSTR.c
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ ./a.out
The number of page request are:8
The pages present in the requested string are
7 4 2 1 7 1 2 9
The elements in the frame are
0
The elements in the frame are
0
1
The elements in the frame are
0
1
2
The page replaced is 7
The elements in the frame are
1
2
0
The page replaced is 4
The elements in the frame are
2
0
1
The elements in the frame are
2
1
0
The elements in the frame are
1
0
2
The page replaced is 7
The elements in the frame are
0
2
1
The number of page faults are 6
```

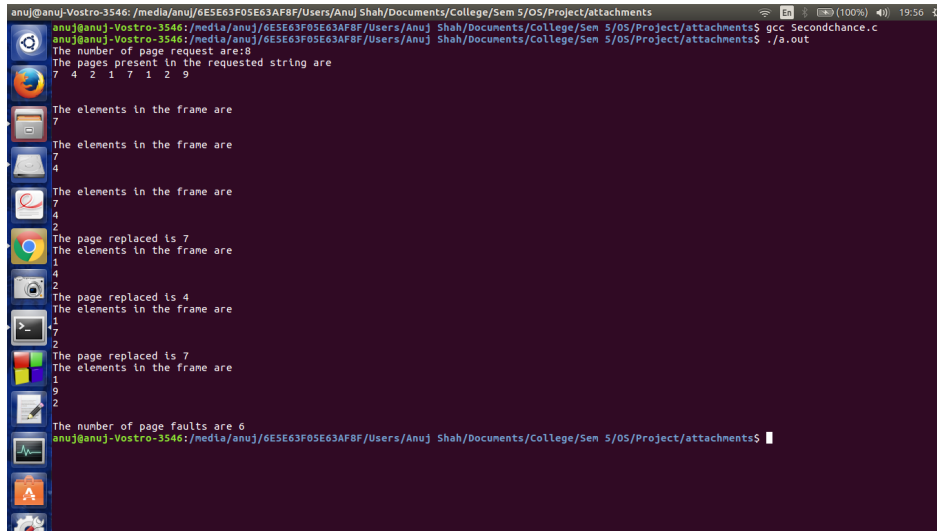
## 4.5 MFU

```
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ gcc MFU.c
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ ./a.out
The number of page request are:8
The pages present in the requested string are
7 4 2 1 7 1 2 9
The elements in the frame are
7
The elements in the frame are
7
4
The elements in the frame are
7
4
2
The page replaced is 7
The elements in the frame are
1
4
2
The page replaced is 1
The elements in the frame are
7
4
2
The page replaced is 7
The elements in the frame are
1
4
2
The page replaced is 2
The elements in the frame are
1
4
9
The number of page faults are 7
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$
```

## 4.6 OPT

```
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ gcc OPT.c
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ ./a.out
The number of page request are:8
The pages present in the requested string are
7 4 2 1 7 1 2 9
The elements in the frame are
7
The elements in the frame are
7
4
The elements in the frame are
7
4
2
The page replaced is 4
The elements in the frame are
7
1
2
The page replaced is 2
The elements in the frame are
7
1
9
The number of page faults are 5
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$
```

## 4.7 Second Chance



```
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ gcc Secondchance.c
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$ ./a.out
The number of page request are:8
The pages present in the requested string are
7 4 2 1 7 1 2 9
The elements in the frame are
7
The elements in the frame are
7
4
The elements in the frame are
7
4
2
The page replaced is 7
The elements in the frame are
1
4
2
The page replaced is 4
The elements in the frame are
1
7
2
The page replaced is 7
The elements in the frame are
1
9
2
The number of page faults are 6
anuj@anuj-Vostro-3546: /media/anuj/6E5E63F05E63AF8F/Users/Anuj Shah/Documents/College/Sem 5/OS/Project/attachments$
```

## 5 Technical Specification

Code Language – C and Python 2.7

Code compatibility – UNIX and Windows

Input – Randomly generated page numbers using Python

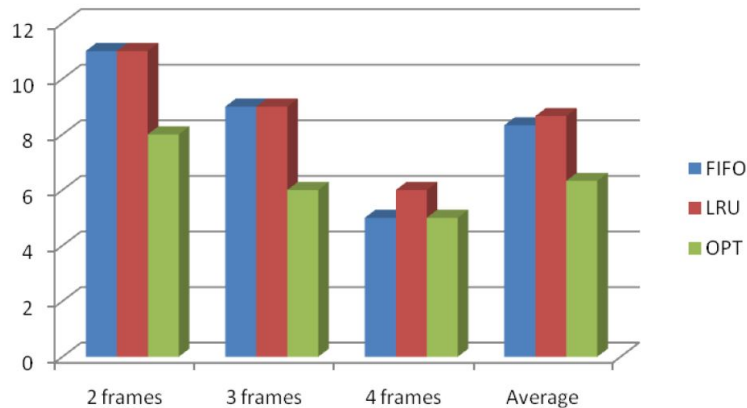
Output – Number of page faults and intermediate pages in frame

Data structures used – Stack, Array, Linked list

## 6 Analysis of most used Algorithm

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



## 7 Anamoly of FIFO page replacement algorithm - Bélády's anomaly

In computer storage, Bélády's anomaly is the phenomenon in which increasing the number of page frames results in an increase in the number of page faults for certain memory access patterns. This phenomenon is commonly experienced when using the First in First Out (FIFO) page replacement algorithm.

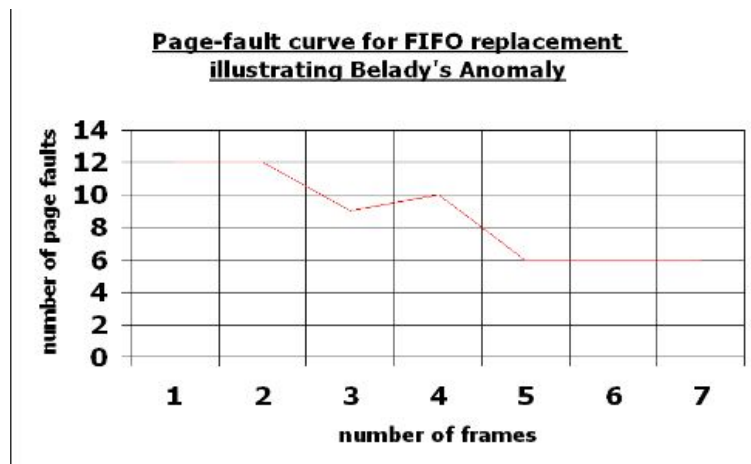


Illustration of Bélády's anomaly

## 8 References

### References

- [1] Operating System internals and Design Principles, 7th ed. Pearson
- [2] "LRU Implementations", [Cs.jhu.edu](http://www.cs.jhu.edu/~yairamir/cs418/os6/tsld021.htm), 2016. [Online]. Available: <http://www.cs.jhu.edu/~yairamir/cs418/os6/tsld021.htm>. [Accessed : 07 – Dec – 2016].
- [3] "Operating System - Virtual Memory", [www.tutorialspoint.com](http://www.tutorialspoint.com), 2016. [Online]. Available: [https://www.tutorialspoint.com/operating\\_system/os\\_virtual\\_memory.htm](https://www.tutorialspoint.com/operating_system/os_virtual_memory.htm). [Accessed : 07 – Dec – 2016].
- [4] "Operating Systems", [Www2.cs.uregina.ca](http://www2.cs.uregina.ca), 2016. [Online]. Available: [http://www2.cs.uregina.ca/~hamilton/courses/330/notes/memory/page\\_replacement.html](http://www2.cs.uregina.ca/~hamilton/courses/330/notes/memory/page_replacement.html). [Accessed : 07 – Dec – 2016].
- [5] "Page replacement algorithm", [En.wikipedia.org](http://en.wikipedia.org), 2016. [Online]. Available: [https://en.wikipedia.org/wiki/Page\\_replacement\\_algorithm](https://en.wikipedia.org/wiki/Page_replacement_algorithm). [Accessed : 07 – Dec – 2016].
- [6] "What's the difference between clock and Second chance page-replacement algorithm?," 2016. [Online]. Available: <http://cs.stackexchange.com/questions/29092/whats-the-difference-between-clock-and-second-chance-page-replacement-algorithm>. Accessed: Dec. 7, 2016.
- [7] "Page Replacement Algorithm", <http://www.cs.utexas.edu/users/witchel/372/lectures/16.PageReplacementAlgos.pdf>, 2016.
- [8] Optimal (OPT) Page replacement Algorithm, [http://faculty.salina.k-state.edu/tim/oss/\\_images/optimal.png](http://faculty.salina.k-state.edu/tim/oss/_images/optimal.png) 2016.
- [9] Most Frequently Used(MFU) Page replacement Algorithm, [http://images.slideplayer.com/20/6026617/slides/slide\\_56.jpg](http://images.slideplayer.com/20/6026617/slides/slide_56.jpg) 2016.