

# Page Faults and Page Replacement Algorithms

## Introduction

This activity explores when page faults happen and how page replacement algorithms can affect the number of page faults that happen.

**Student Learning Objectives:** Students will be able to...

- Demonstrate an understanding of the process of handling a page fault.
- Articulate the need for a page replacement algorithm in an operating system.
- Compute how many page faults occur for FIFO, OPT, LRU, and Working Set algorithms, for various inputs.
- Articulate the meaning of Belady's Anomaly.

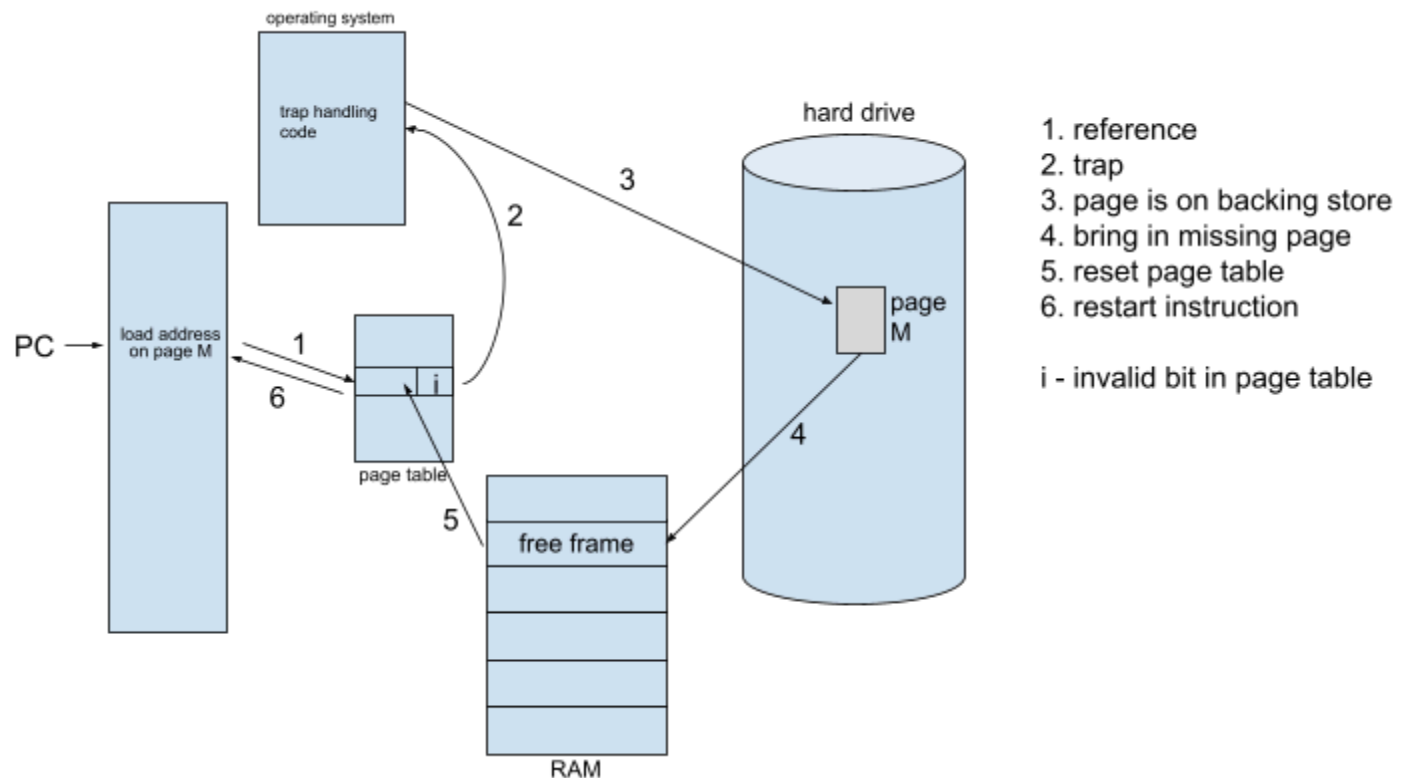
**Prerequisites:** An understanding of page tables.

<b>Date</b>	
<b>Team Members</b>	
<b>Member 1:</b>	
<b>Member 2:</b>	

A **page fault** occurs when a process executes code that tries to access a page (containing text, heap, data, stack, etc.) that has not been brought from disk into memory (yet).


### ***Model I. Handling a Page Fault (10 minutes)***

# Page Faults and Page Replacement Algorithms



Based on figure 8.6 from Operating Systems Essentials, 2nd ed. by Silberschatz, Galvin, Gagne; John Wiley, Inc.

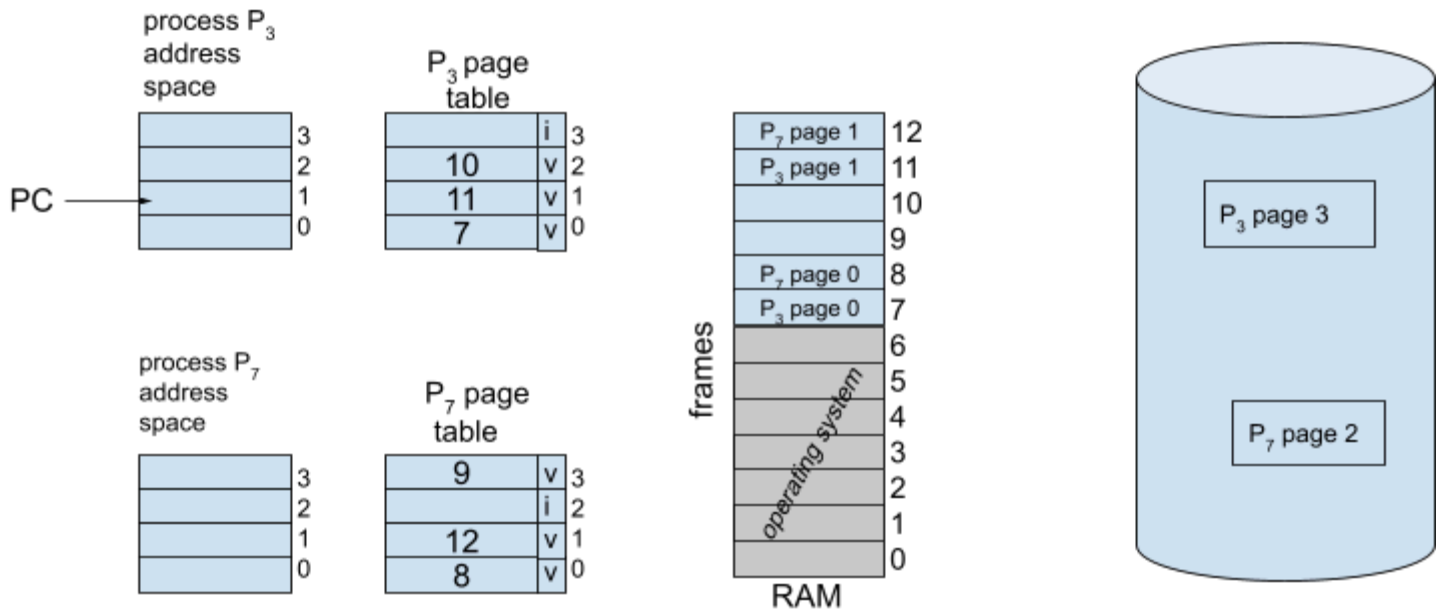
The picture above illustrates the steps to handle a page fault.

1. Describe what step 1 ("reference") means.
2. How does the page table indicate that a trap should be generated?
3. Which entry in the page table does "load the address from page M" access? I.e., what is its index?
4.  What does step 5 ("reset page table") mean? What two things does the OS change in the page table?
5. From your knowledge of memory access and disk access times, which step takes the longest to complete?
6. Speculate: After restarting the instruction, is the system guaranteed to not page fault again? Why or why not?


## Page Faults and Page Replacement Algorithms

7. Only one box is labeled "RAM", but other boxes are actually also in RAM. Which ones?

### Model II: Need for Page Replacement Algorithm (10 minutes)



$P_3$  is executing an instruction on page 1 that says to access memory on page 3.

8. How many total frames exist for use by processes (i.e., not being used by the operating system)?
9. How many pages are there in  $P_3$ 's and  $P_7$ 's address spaces combined?
10. Fill in the labels for  $P_3$  page 2 and  $P_7$  page 3 in the unlabeled frames in RAM.
11.  What happens when the CPU tries to execute  $P_3$ 's instruction on its page 1 that tries to access data on page 3 (see Model I)?
12. What step in Model I cannot execute because there is no free frame?

# Page Faults and Page Replacement Algorithms

## ***Extension Question (if you have time):***

13. What could the operating system do in this case?

## ***Model III: First-In-First-Out (FIFO) Algorithm (10 minutes)***

An OS will, in the case where there is no free frame, typically find a **victim page** either from the same process that causes the page fault or from some other process. The victim page will be written out to disk (if necessary), and the frame will now be free. The free frame then will be filled with the page that needs to be brought in from disk so that the process can continue executing.

Finding a victim page is the responsibility of a **page-replacement algorithm**.

Suppose that a process accesses pages in the order shown in this **reference string** of length 22:

**7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1**

But, the process has only been allocated 3 frames for its use.

The **FIFO** Page Replacement algorithm chooses to victimize the page in memory that was loaded the furthest in the past (i.e., the "oldest" page).

To compute how many page faults result from having 3 frames available and using the given reference string, we can build a table. The first 7 rows have been filled in for you.

Page Reference	Frame 0	Frame 1	Frame 2	Page Fault?
7	7	free	free	Y
0	7	0	free	Y
1	7	0	1	Y
2	2	0	1	Y
0	2	0	1	N
3	2	3	1	Y
0	2	3	0	Y

## Page Faults and Page Replacement Algorithms


4				
2				
3				
0				
3				
0				
3				
2				
1				
2				
0				
1				
7				
0				
1				

14. When a frame is "free" (i.e., empty) and a page is brought into that frame from disk, is that considered a page fault?


15. In the 4th row, 2 replaces 7. Why was page 7 chosen to be replaced?

16. Why is there no page fault for the 5th row?

17. How many page faults are there in the first 7 rows?

18.  Complete the table. When you are finished: how many page faults are there using the FIFO algorithm?

## Page Faults and Page Replacement Algorithms

19.  Suppose the reference string is 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 and 3 frames are available for use. Compute the number of page faults by filling out the table below.

Page Reference	Frame 0	Frame 1	Frame 2	Page fault?
1				
2				
3				
4				
1				
2				
5				
1				
2				
3				
4				
5				

### ***Extension Question (if you have time)***

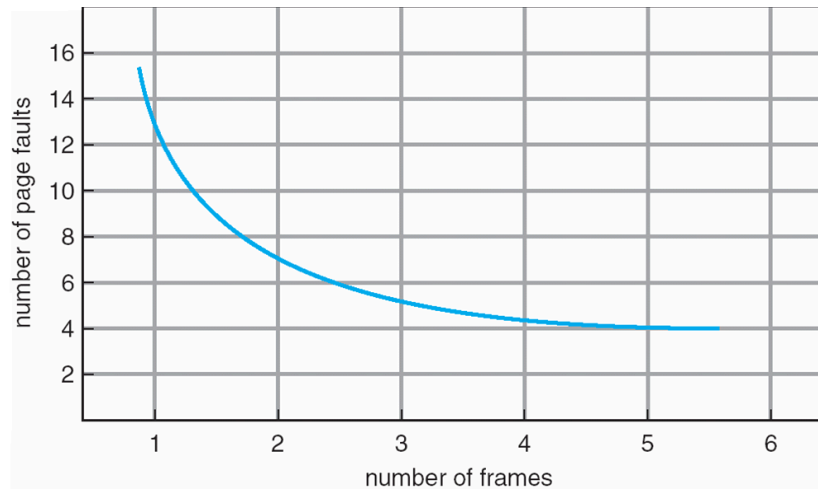
20. How could one implement the tracking of the “age” of the pages so that it is efficient to find the oldest page quickly?

### ***Model IV: Page Faults vs. Number of Frames available (5 minutes)***

To estimate the number of page faults vs. the number of frames allocated to a process, one could use a

## Page Faults and Page Replacement Algorithms

graph like the following.



21. If there are 2 frames allocated, approximately how many page faults do you expect?


22. If there are 4 frames allocated, approximately how many page faults do you expect?

23. Using the reference string from question 19 above, fill out the table, and compute the number of page faults that result when the process is allocated 4 frames instead of 3 frames.

Page Reference	Frame 0	Frame 1	Frame 2	Frame 3	Page fault?
1					
2					
3					
4					
1					
2					
5					
1					
2					
3					

## Page Faults and Page Replacement Algorithms

4					
5					

24.  Is the model correct, in this case? Why or why not?

The FIFO page replacement algorithm suffers from **Belady's Anomaly**, where increasing the number of frames allocated to a process can actually cause the process to have more page faults.

### ***Model V: OPT Algorithm (10 minutes)***

The OPT (or "Optimal") algorithm chooses to victimize a page which will not be used for the longest time in the future. (A page only needs to be victimized, of course, when all frames are full.) The reference string is shown in the left column of the following table and the first 8 rows have been filled in for you.

Page Reference	Frame 0	Frame 1	Frame 2	Page Fault?
7	7	free	free	Y
0	7	0	free	Y
1	7	0	1	Y
2	2	0	1	Y
0	2	0	1	N
3	2	0	3	Y
0	2	0	3	N
4	2	4	3	Y
2				
3				
0				
3				





## Page Faults and Page Replacement Algorithms

0				
3				
2				
1				
2				
0				
1				
7				
0				
1				

25. Why was page 7 chosen to be victimized in row 4? Why wasn't page 1 victimized?

26. Why was page 0 victimized in row 8?

27.  Fill in the rest of the table. How many page faults result?

28.  Why can this algorithm not be implemented in real life?

This algorithm is impossible to implement as-is. However, other algorithms exist which try to approximate this algorithm -- i.e., they try to predict the future behavior of the process.


### ***Model V: Least Recently Used (LRU) Algorithm (10 minutes)***

The LRU algorithm uses the past to try to predict the future. So, the algorithm chooses to victimize the page in memory that has not been used (i.e., accessed) for the longest amount of time in the past.

Page Reference	Frame 0	Frame 1	Frame 2	Page Fault?
----------------	---------	---------	---------	-------------

## Page Faults and Page Replacement Algorithms

7	7	free	free	Y
0	7	0	free	Y
1	7	0	1	Y
2	2	0	1	Y
0	2	0	1	N
3	2	0	3	Y
0	2	0	3	N
4	4	0	3	Y
2				
3				
0				
3				
0				
3				
2				
1				
2				
0				
1				
7				
0				
1				

29.  Complete the table. How many page faults result?

## Page Faults and Page Replacement Algorithms

30. How does the number of page faults compare to your answers for the OPT and FIFO algorithms?

### *Extension Question (if you have time)*

31. How could you implement this algorithm? I.e., how could you find the least recently used page?

### *Model VI: Working Sets (10 minutes)*

The Working Set Model is an algorithm for the Frame-Allocation problem -- how many frames to allocate to each process in memory. The algorithm works as follows:

- For each page reference,
  - if the page is not in the current working set, generate a page fault, load the page, add the page to the working set, and increment the working set size (WSS).
  - If a page in the current set (i.e., in memory) should not be in the working set anymore, release the frame back to the operating system, and decrement the WSS.

The 2nd step requires more explanation to clarify what we mean by "should not be in the working set anymore": the algorithm relies upon a defined **window size ("delta")** that is the last **n** page references (including the current one). So, for  $\delta = 5$ , if one of the pages has not been referenced by the current reference or the previous 4 references, that page should be removed from the working set and the frame released back to the OS (and the WSS decremented).


Note a few things: 1. In one step, a page can be added to the working set, and, another page can be removed from the working set. 2. the maximum number of frames in the working set (i.e., the maximum WSS) equals the delta.

#### **Window size (delta) = 5**


Page reference	Working Set	WSS	Page fault?	Frame released?	Comment
7	7	1	y	n	
0	0, 7	2	y	n	
1	0, 1, 7	3	y	n	Count # of pages in the Working Set to get the WSS.
2	0, 1, 2, 7	4	y	n	
0	0, 1, 2, 7	4	n	n	"Working Set" is a set, so 0 is not added twice, and no page fault.

## Page Faults and Page Replacement Algorithms

3	0, 1, 2, 3	4	y	y	Page 7 is removed because it was not accessed within the last delta accesses (including this one). Page 3 is added.
0					
4					
2					
3					
0					
3					
0					The working set shrinks here because page 4 was not accessed within the last delta accesses (including this one).
3					
2					
1					
2					
0					
1					
7					
0					
1					
<b>Sum</b>					

32.  Fill in the table above. Put a 'y' in the "Page fault?" column when a new page must be brought in and added to the working set. Put a 'y' in the "Frame released?" column when a page that was in memory is released because it is not in the current working set. Fill in the Sum line for the "Page Fault?" and "Frame Released?" columns.

## Page Faults and Page Replacement Algorithms

33.  How many page faults are there? How does this compare to our other algorithms above?

### ***Extension Question (if you have time):***

34. Will this algorithm suffer from Belady's Anomaly?

© Victor Norman at Calvin College, 2022 [ytn2@calvin.edu](mailto:ytn2@calvin.edu)

Version 2.1

This work is licensed under the Creative Commons Attribution 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.