

OPERATING SYSTEMS

Memory Management

Suresh Jamadagni

Department of Computer Science

OPERATING SYSTEMS

Virtual Memory

Suresh Jamadagni

Department of Computer Science

OPERATING SYSTEMS

Slides Credits for all the PPTs of this course

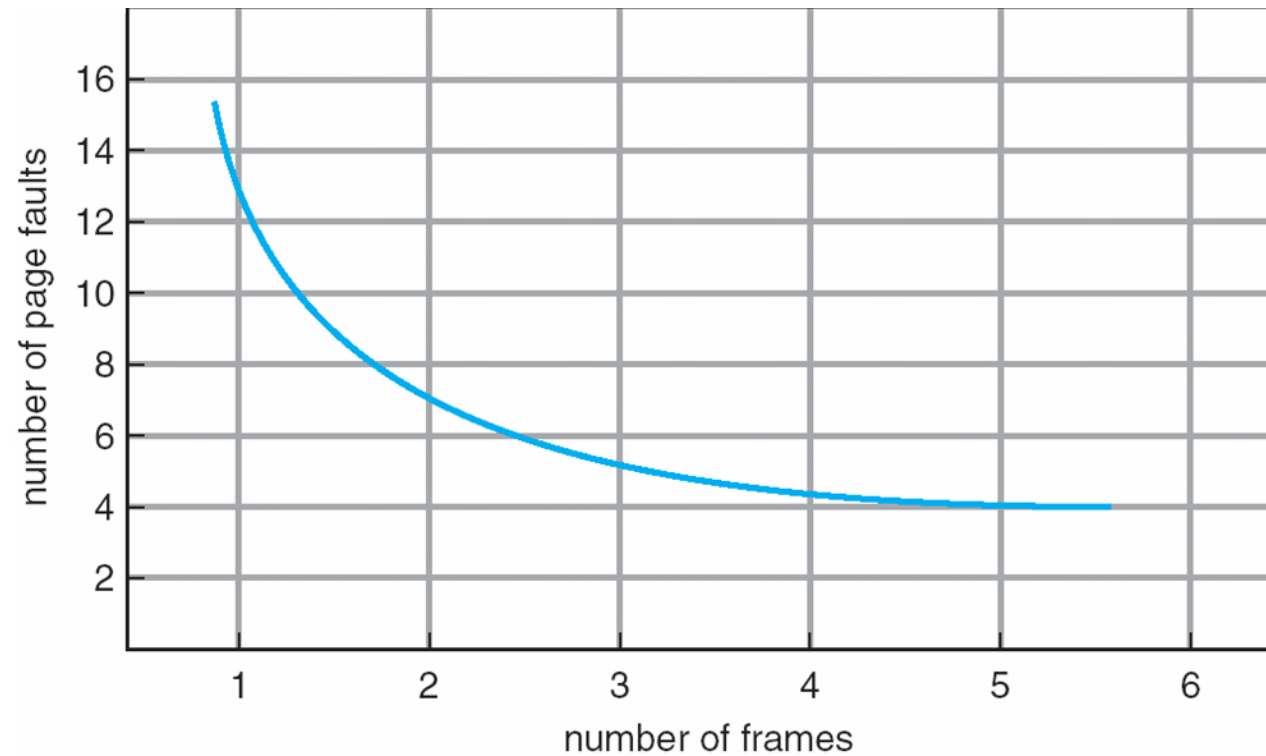


- The slides/diagrams in this course are an **adaptation**, **combination**, and **enhancement** of material from the following resources and persons:
 1. Slides of Operating System Concepts, Abraham Silberschatz, Peter Baer Galvin, Greg Gagne - 9th edition 2013 and some slides from 10th edition 2018
 2. Some conceptual text and diagram from Operating Systems - Internals and Design Principles, William Stallings, 9th edition 2018
 3. Some presentation transcripts from A. Frank – P. Weisberg
 4. Some conceptual text from Operating Systems: Three Easy Pieces, Remzi Arpaci-Dusseau, Andrea Arpaci Dusseau

- **Frame-allocation algorithm** determines
 - How many frames to give each process
 - Which frames to replace
- **Page-replacement algorithm**
 - Want lowest page-fault rate on both first access and re-access
- Evaluate algorithm by running it on a particular string of memory references (reference string) and computing the number of page faults on that string
 - String is just page numbers, not full addresses
 - Repeated access to the same page does not cause a page fault
 - Results depend on number of frames available
- In all our examples, the **reference string** of referenced page numbers is
7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1

OPERATING SYSTEMS

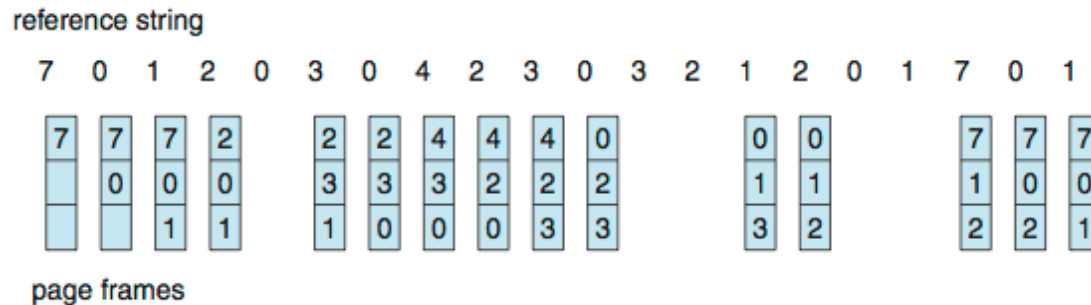
Graph of Page Faults Versus The Number of Frames



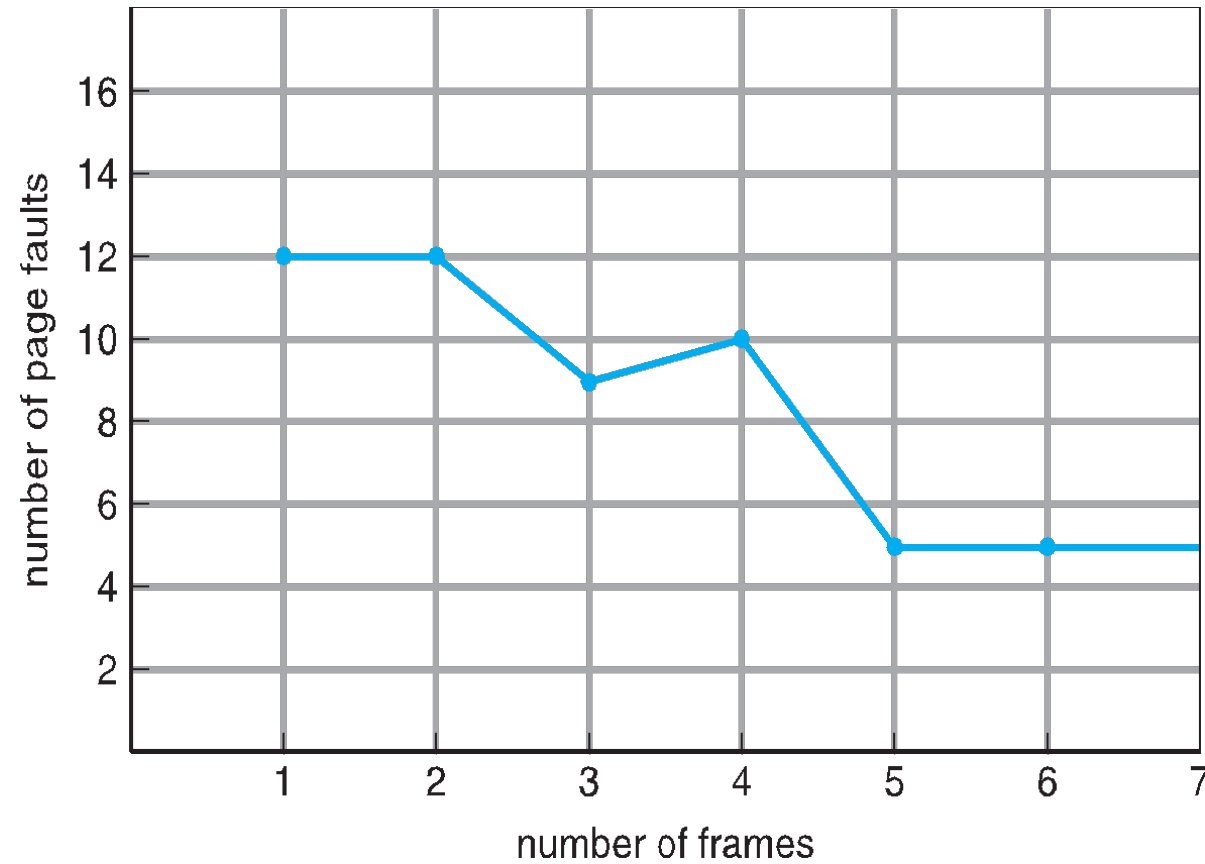
OPERATING SYSTEMS

First-In-First-Out (FIFO) Algorithm

- Reference string: **7,0,1,2,0,3,0,4,2,3,0,3,0,3,2,1,2,0,1,7,0,1**
- 3 frames (3 pages can be in memory at a time per process)



- Can vary by reference string: consider 1,2,3,4,1,2,5,1,2,3,4,5
 - Adding more frames can cause more page faults!
- 4 Belady's Anomaly
- How to track ages of pages?
 - Just use a FIFO queue



OPERATING SYSTEMS

Optimal Algorithm

- Replace page that will not be used for longest period of time
 - 9 is optimal for the example
- How do you know this?
 - Can't read the future
- Used for measuring how well your algorithm performs

reference string

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

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

page frames

OPERATING SYSTEMS

Least Recently Used (LRU) Algorithm

- Use past knowledge rather than future
- Replace page that has not been used in the most amount of time
- Associate time of last use with each page

reference string

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

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

page frames

- 12 faults – better than FIFO but worse than OPT
- Generally good algorithm and frequently used
- But how to implement?

OPERATING SYSTEMS

LRU Algorithm (Cont.)



- Counter implementation
 - Every page entry has a counter; every time page is referenced through this entry, copy the clock into the counter
 - When a page needs to be changed, look at the counters to find smallest value
 - 4 Search through table needed
- Stack implementation
 - Keep a stack of page numbers in a double link form:
 - Page referenced:
 - 4 move it to the top
 - 4 requires 6 pointers to be changed
 - But each update more expensive
 - No search for replacement
- LRU and OPT are cases of **stack algorithms** that don't have Belady's Anomaly

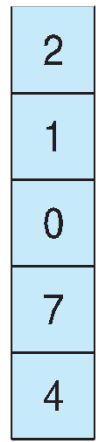
OPERATING SYSTEMS

Use Of A Stack to Record Most Recent Page References

- LRU and OPT are cases of **stack algorithms** that don't have Belady's Anomaly
- Use Of A Stack to Record Most Recent Page References

reference string

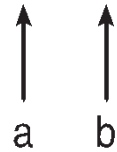
4 7 0 7 1 0 1 2 1 2 7 1 2



stack
before
a



stack
after
b





THANK YOU

Suresh Jamadagni

Department of Computer Science Engineering

sureshjamadagni@pes.edu