# CSE 306 Operating Systems
# Virtual Memory

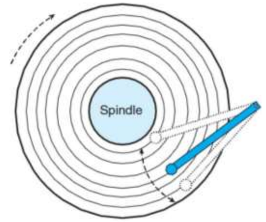
YoungMin Kwon

# Operating System Policies for VM

- Contents
  - Fetch policy (disk → mem)
    - Demand paging, prefetching
  - Placement policy
  - Replacement policy
    - Optimal, LRU, FIFO, clock, page buffering
  - Resident set management
    - Fixed/variable set size, global/local scope
  - Cleaning policy (mem → disk)
    - Demand cleaning, precleaning
  - Load control (degree of multiprogramming)

# Fetch Policy

- Demand paging
  - A page is brought into main memory only when the page is accessed

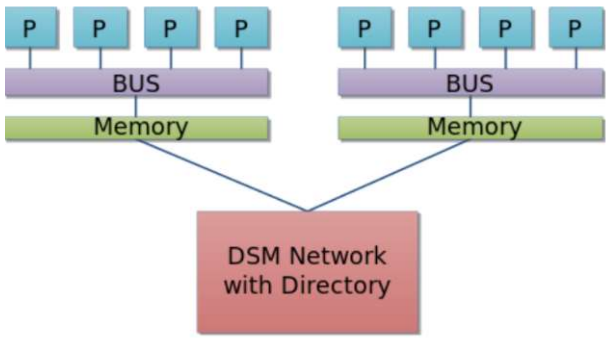



- Prepaging
  - Pages other than the one accessed are brought in
  - Exploits the characteristics of secondary memory (disks have latency and rotational delay)

# Placement Policy

- In pure segmentation system
  - Best-fit, first-fit, …
  - In pure paging or paging + segmentation systems, placement is usually irrelevant

- In NonUniform Memory Access (NUMA) system
  - Access time to a particular memory location varies with the distance between the processor and the memory
  - Need to place data close to the processors that use them

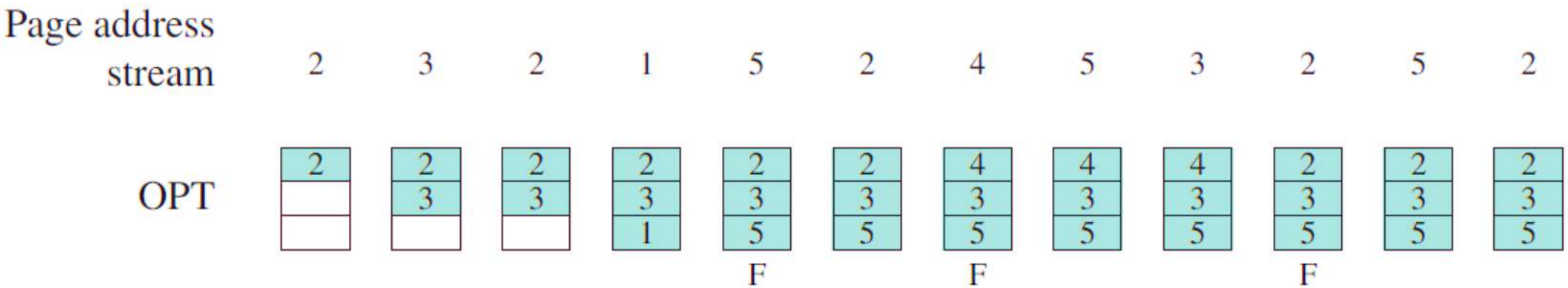


# Replacement Policy

- Replacement Policy
  - Selecting a page frame to be replaced
- Frame Locking
  - When a frame is locked, the page stored in the frame will not be replaced
  - Most kernel code and key control structures are held in locked frames

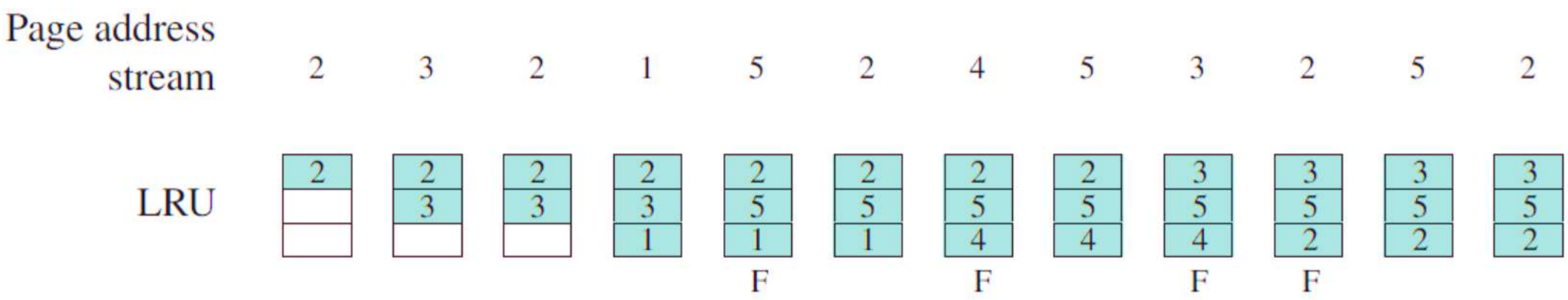
# Replacement Policy

- Optimal algorithm
  - Select the page that will not be referenced longest
  - Not implementable: for the comparison purpose



# Replacement Policy

- Least Recently Used (LRU) algorithm
  - Select the page that has not been referenced longest
  - Tag each page with the time of its last reference
  - Alternatively, maintain a stack of page references



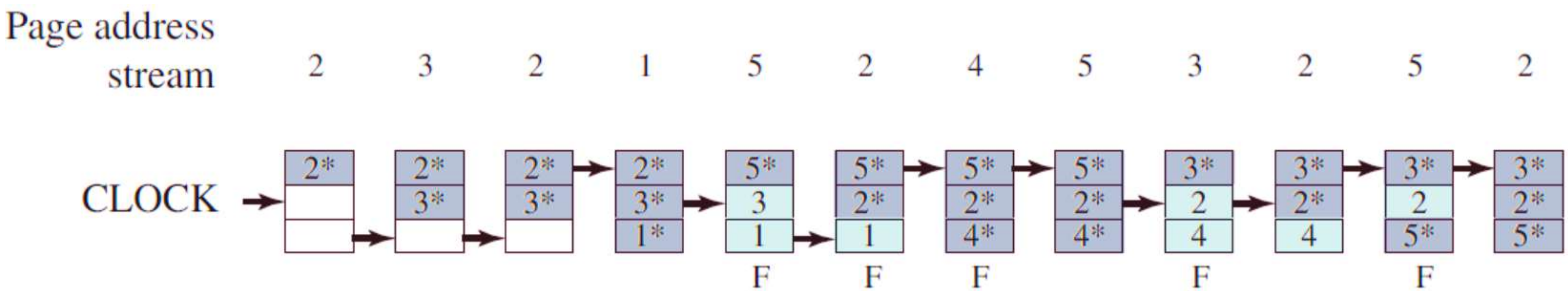
# Replacement Policy

- First In First Out (FIFO) Algorithm
  - Replace the page that has been in memory the longest
  - Simple to implement (a pointer that cycles all frames)

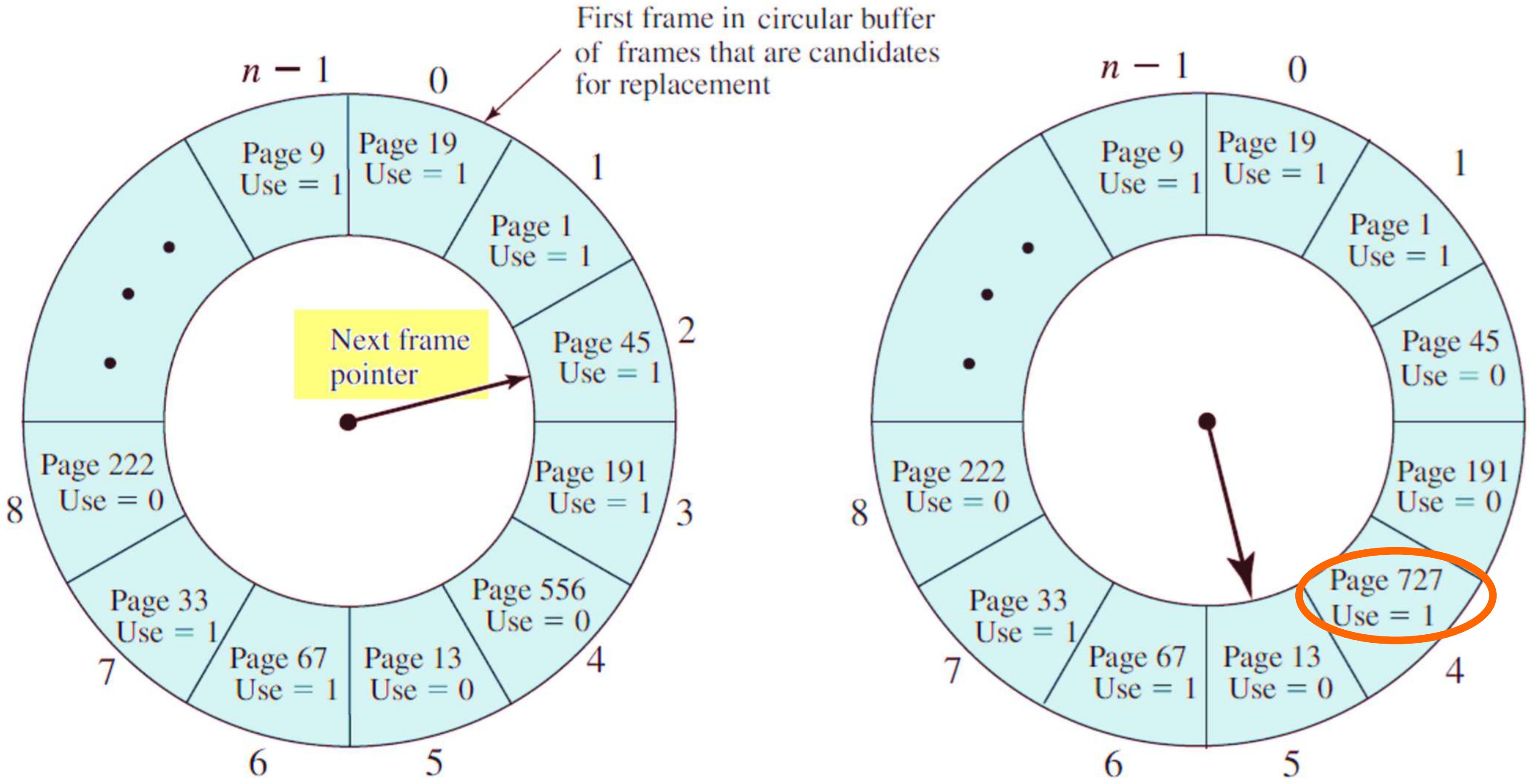
| Page address stream | 2 | 3 | 2 | 1 | 5 | 2 | 4 | 5 | 3 | 2 | 5 | 2 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FIFO | 2 | 2 | 2 | 2 | 5 | 5 | 5 | 5 | 3 | 3 | 3 | 3 |
| | | 3 | 3 | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 5 | 5 |
| | | | | 1 | 1 | 1 | 4 | 4 | 4 | 4 | 4 | 2 |
| | | | | | F | F | F | | F | | F | F |

# Replacement Policy

- Clock algorithm
  - Add a use bit to each frame: whenever a page is referenced its use bit is set
  - Like FIFO, a page pointer cycles frames to find a frame whose use bit is 0
  - While scanning frames, set the use bit to 0



# Replacement Policy (Clock)



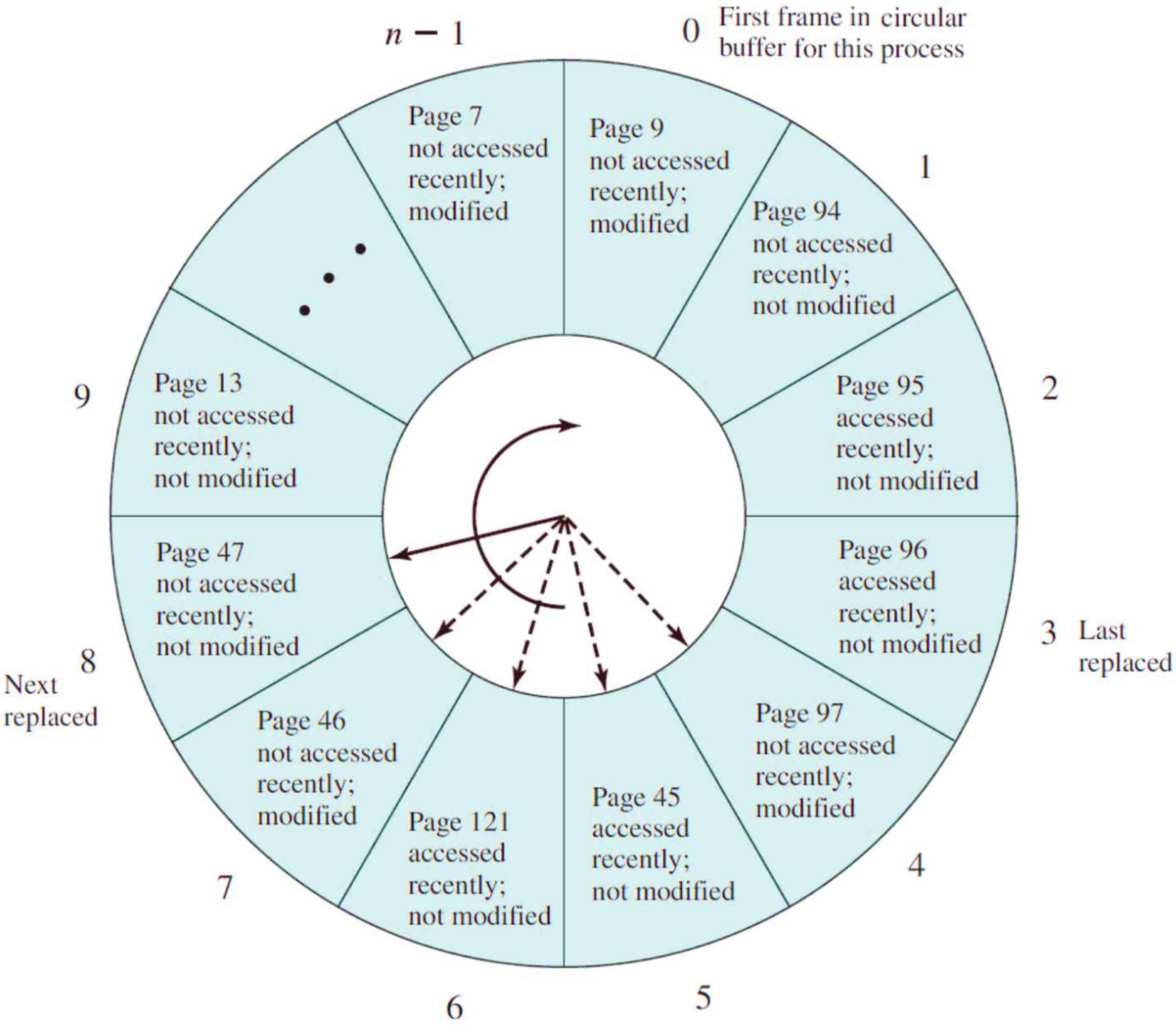

(a) State of buffer just prior to a page replacement

(b) State of buffer just after the next page replacement

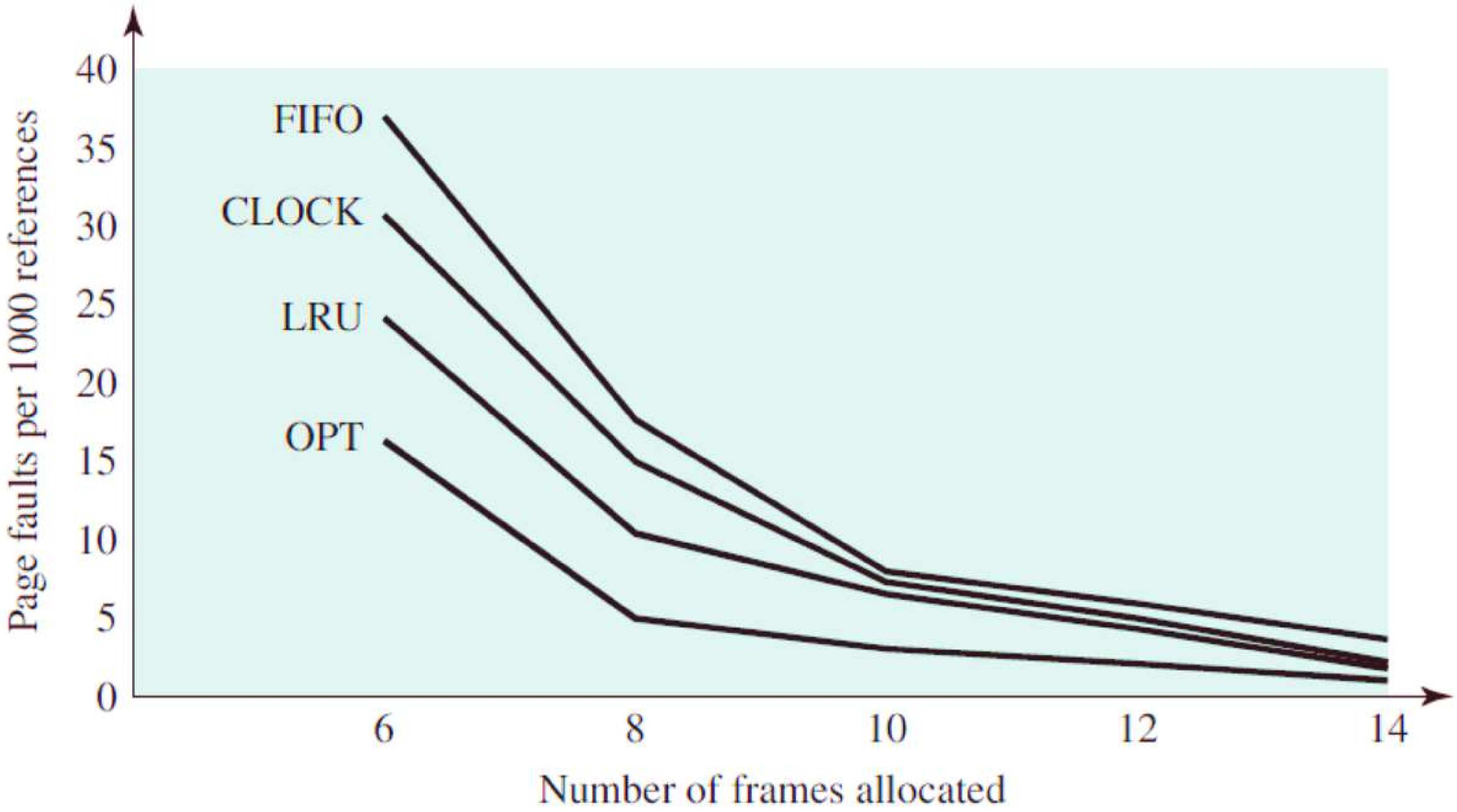
# Replacement Policy

- Clock algorithm with modify bit
  - Each frame falls into
    - Not accessed, not modified (u=0, m=0)
    - Accessed, not modified (u=1, m=0)
    - Not accessed, modified (u=0, m=1)
    - Accessed, modified (u=1, m=1)
  - Step 1: starting from the current position, try to find a frame with (u=0, m=0)
  - Step 2: from the current position try to find a frame with (u=0, m=1). During this scan set u of encountered frames to 0
  - Step 3: Repeat step 1 and step 2 if necessary

# Replacement Policy (Clock with modify bit)



# Replacement Policy: Comparison



# Replacement Policy

- Page buffering
  - Similar to FIFO, but victim frames are moved to one of frame pools: free or modified frame pools
    - Their entries in the page table are removed
    - If a frame in the pools is accessed, only the page table is updated without reading the frame from disk
  - When a pool is full, swap some frames out to disk
    - Modified frames can be swapped out to the disk together considering the seek time and the rotational delay

# Resident Set Management

- Resident set of a process
  - The portion of a process that is actually in main memory at any time

- Resident set size
  - Smaller the size
    - → More processes can reside in main memory
    - → More page fault
  - After a certain point, adding more pages to a particular process will have no noticeable effects

# Resident Set Size

- Fixed-allocation policy
  - Give a process a fixed number of frames
  - On a page fault, one of the frames of the process needs to be replaced

- Variable-allocation policy
  - # of frames allocated to a process will vary over time
  - More frames will be given to processes experiencing a high page fault rates
  - Frames will be taken from processes with exceptionally low page fault rates

# Replacement Scope

- Local replacement policy
  - Find a victim frame among the frames of the process that caused the page fault
- Global replacement policy
  - Consider all unlocked pages in main memory

# Fixed Allocation, Local Scope

- OS must choose one of the frames of the process for the replacement

- Decide the amount of frames to give to a process ahead of time
  - Too small allocation: high page fault rate
  - Too much allocation: low degree of multiprogramming

# Variable Allocation, Global Scope

- Replacement algorithm
  - Typically, OS maintains a list of free frames
  - On page fault, a free frame is added to the resident set of the process causing the fault
  - When no free frame is available, select a frame from any processes

# Variable Allocation, Local Scope

- Strategy
  - When a new process is loaded, allocate to it a certain number of frames
  - Victim page is selected from among the resident set of the process that caused the fault
  - From time to time, reevaluate the allocation and expand or shrink the allocation to improve overall performance

# Variable Allocation, Local Scope

- Working set
  - $W(t, \Delta)$: the set of pages referenced during $(t-\Delta, t]$
    - t: virtual time representing the memory reference count of a process
  - Working set is a non-decreasing function of the window size
    - $W(t, \Delta) \subseteq W(t, \Delta + 1)$
  - The bound of the working set size
    - $1 \leq | W(t, \Delta) | \leq \min(\Delta, N)$, where the entire process is held in N pages

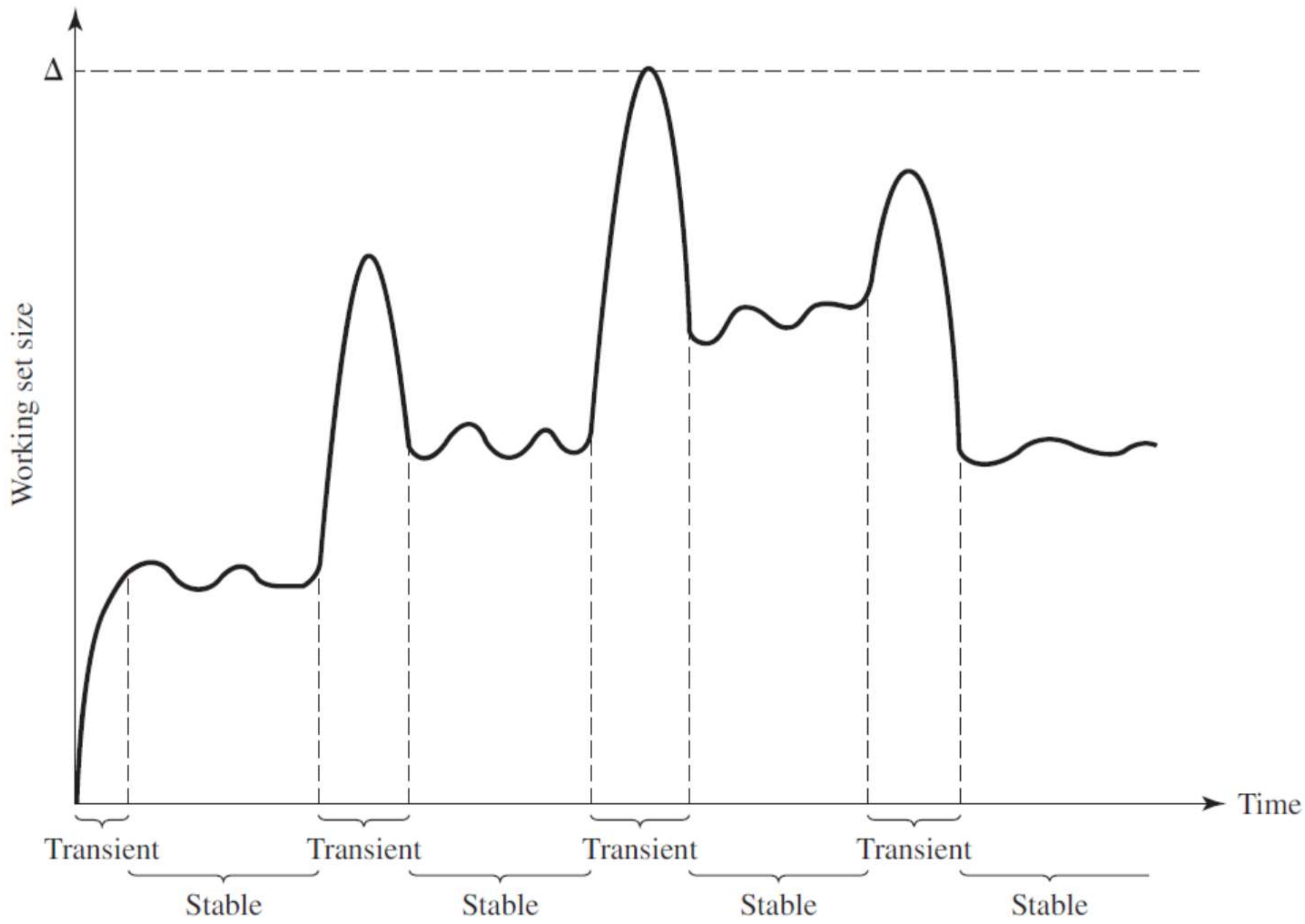
at most 1 page per time window

# Working Set Example

| Sequence of Page References W | Window Size, Δ 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 24 | 24 | 24 | 24 | 24 |
| 15 | 24 15 | 24 15 | 24 15 | 24 15 |
| 18 | 15 18 | 24 15 18 | 24 15 18 | 24 15 18 |
| 23 | 18 23 | 15 18 23 | 24 15 18 23 | 24 15 18 23 |
| 24 | 23 24 | 18 23 24 | • | • |
| 17 | 24 17 | 23 24 17 | 18 23 24 17 | 15 18 23 24 17 |
| 18 | 17 18 | 24 17 18 | • | 18 23 24 17 |
| 24 | 18 24 | • | 24 17 18 | • |
| 18 | • | 18 24 | • | 24 17 18 |
| 17 | 18 17 | 24 18 17 | • | • |
| 17 | 17 | 18 17 | • | • |
| 15 | 17 15 | 17 15 | 18 17 15 | 24 18 17 15 |
| 24 | 15 24 | 17 15 24 | 17 15 24 | • |
| 17 | 24 17 | • | • | 17 15 24 |
| 24 | • | 24 17 | • | • |
| 18 | 24 18 | 17 24 18 | 17 24 18 | 15 17 24 18 |

# Working Set

- For many programs, periods of relative stable working set sizes alternate with periods of rapid changes



# Variable Allocation, Local Scope

- Working set strategy
  - Monitor the working set of each process
  - Periodically remove from the resident set of a process those pages not in the working set (like LRU)
  - A process may execute only if its working set is in main memory

- Problems
  - The past does not always predict the future
  - Computing the working set of processes is impractical
  - Optimal value of $\Delta$ is unknown

# Variable Allocation, Local Scope

- Page Fault Frequency (PFF) algorithm
  - When a page fault occurs, OS records its virtual time
  - If the elapsed time since the last page fault is less than a threshold F, add a frame to the resident set of the process
  - Otherwise, discard all frames whose use bit is 0 and shrink the resident set size accordingly

- Problems with PFF
  - Does not perform well during transition periods (shift to a new locality) $\rightarrow$ resident set size grows

# Variable Allocation, Local Scope

- Variable-interval sampled working set (VSWS)
  - When accessing a frame, set its use bit
  - Any faulted pages are added to the resident set
  - At each sampling time
    - Scan the frames in the resident set and discard the frames whose use bit is not set
    - Clear the use bit of the remaining frames

# Variable Allocation, Local Scope

- Variable-interval sampled working set (VSWS)
  - Parameters
    - M: the minimum duration of the sampling interval
    - L: the maximum duration of the sampling interval
    - Q: the number of page faults allowed to occur between sampling instances
    - E: the virtual time since the last sampling time
  - Suspend the process and scan the use bit
    - If Q page faults occur before L → when E reaches M
    - Otherwise → when E reaches L

# Load Control

- Load control
  - Determines the number of processes that will reside in main memory
- Too few processes
  - All processes can be blocked
  - Processor utilization will be low
- Too many processes
  - Thrashing: page fault will occur frequently
  - Processor utilization will be low