# SNS COLLEGE OF TECHNOLOGY

**Coimbatore-35.**
**An Autonomous Institution**

**Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A++' Grade**
**Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai**

## COURSE NAME : OPERATING SYSTEMS

## II YEAR/ IV SEMESTER

## UNIT – III PAGE REPLACEMENT

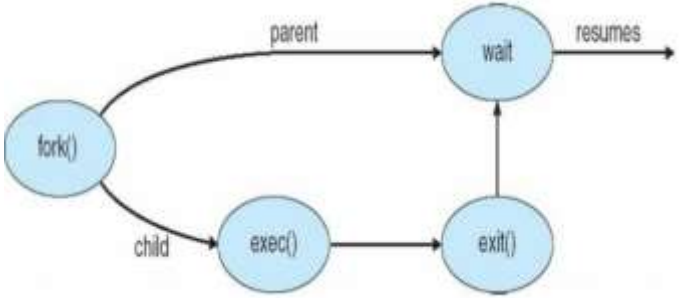
## Topic: Page Replacement

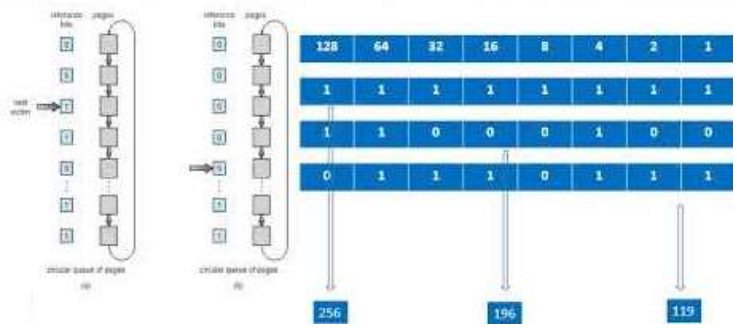
Dr.B.Vinodhini
Associate Professor
Department of Computer Science and Engineering

# SNS COLLEGE OF TECHNOLOGY

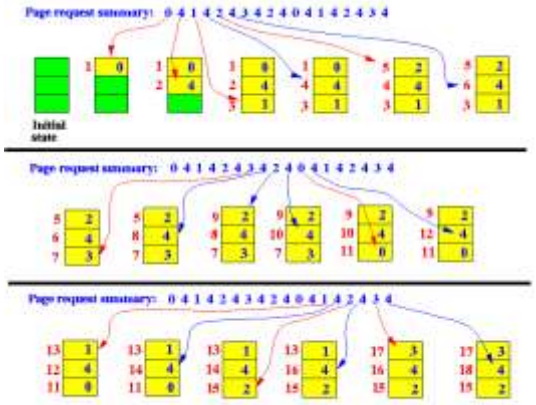
**(Autonomous )**

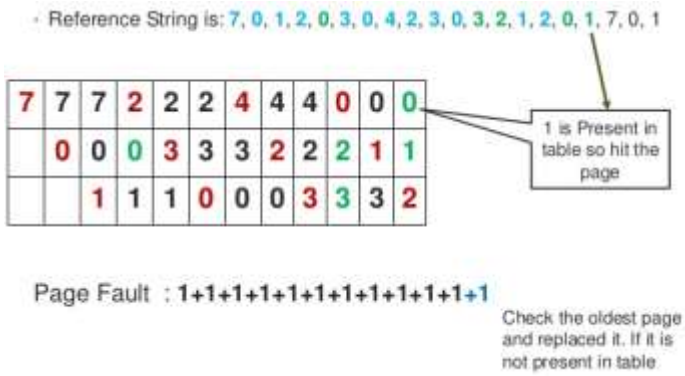
**COIMBATORE-35**



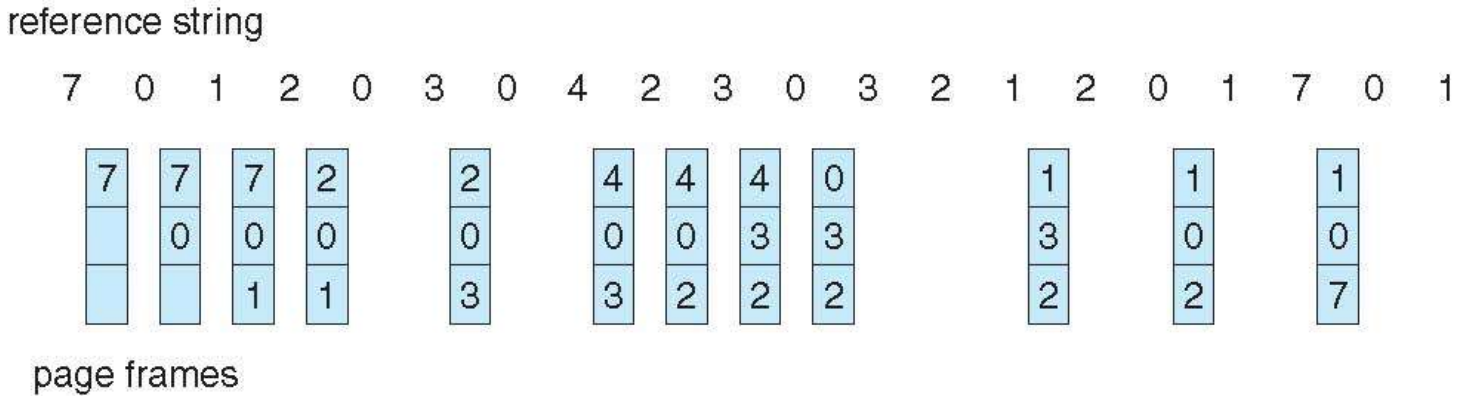

# *Page Replacement*





# *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



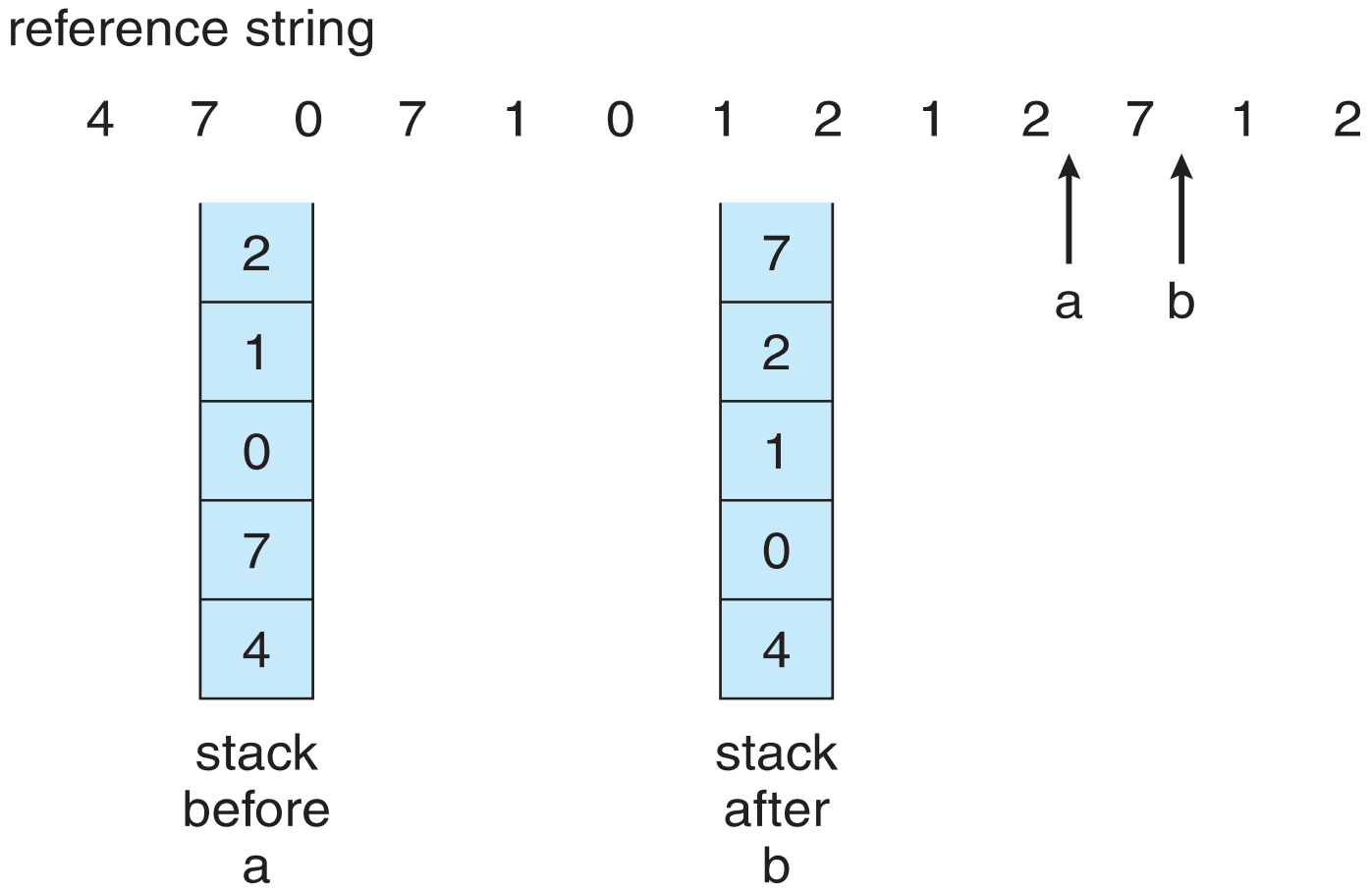

- 12 faults – better than FIFO but worse than OPT

# *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
    - Search through table needed

- LRU and OPT are cases of **stack algorithms** that don't have Belady's Anomaly

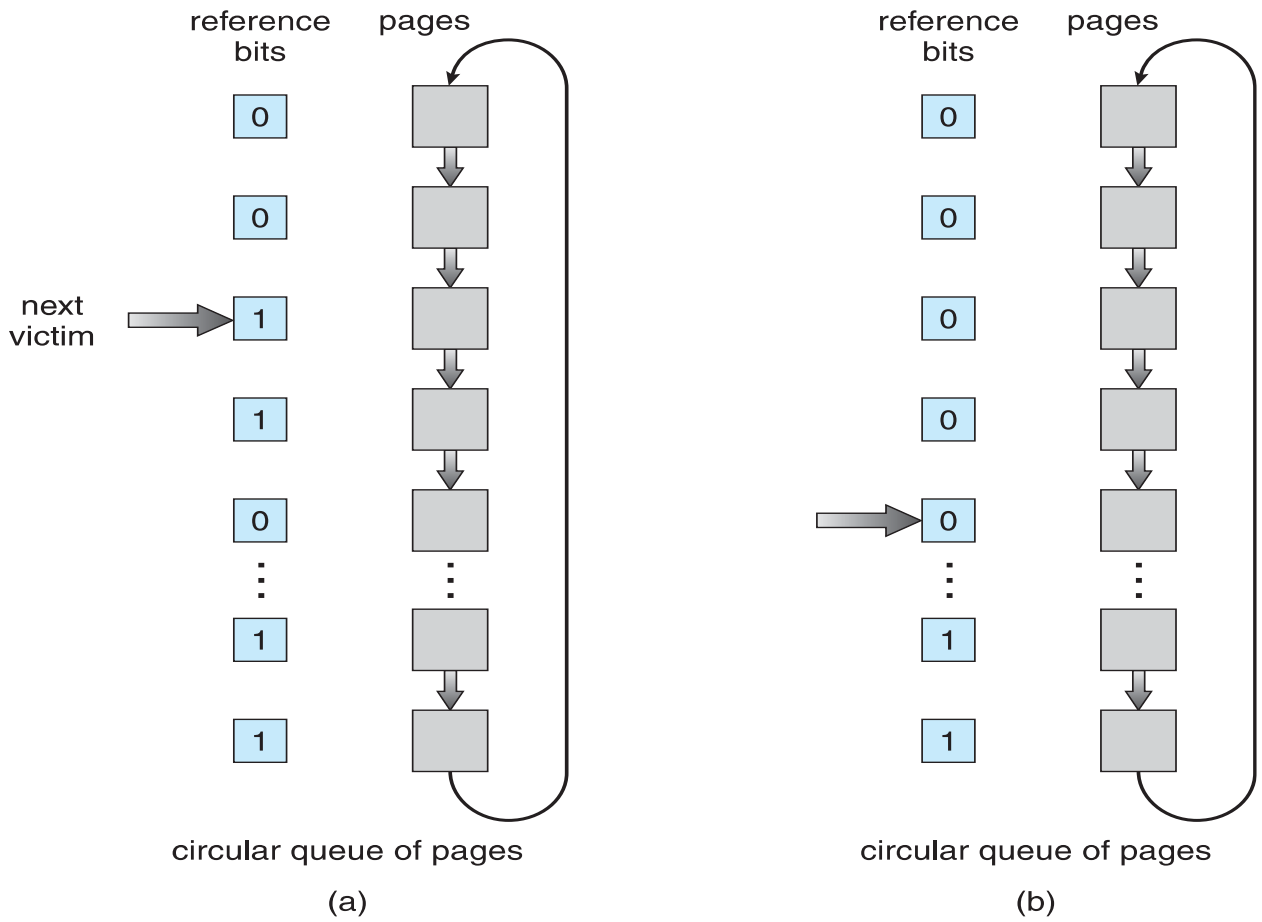
# Use Of A Stack to Record Most Recent Page References



# LRU Approximation Algorithms

- LRU needs special hardware and still slow
- **Reference bit**
  - With each page associate a bit, initially = 0
  - When page is referenced bit set to 1
  - Replace any with reference bit = 0 (if one exists)
    - We do not know the order, however
- **Second-chance algorithm**
  - Generally FIFO, plus hardware-provided reference bit
  - **Clock** replacement
  - If page to be replaced has
    - Reference bit = 0 -> replace it
    - reference bit = 1 then:
      - set reference bit 0, leave page in memory
      - replace next page, subject to same rules

# Second-Chance (clock) Page-Replacement Algorithm



# Enhanced Second-Chance Algorithm

- Improve algorithm by using reference bit and modify bit (if available) in concert
- Take ordered pair (reference, modify)

1. (0, 0) neither recently used not modified – best page to replace
2. (0, 1) not recently used but modified – not quite as good, must write out before replacement
3. (1, 0) recently used but clean – probably will be used again soon
4. (1, 1) recently used and modified – probably will be used again soon and need to write out before replacement

- When page replacement called for, use the clock scheme but use the four classes replace page in lowest non-empty class
  - Might need to search circular queue several times

# Counting Algorithms

- Keep a counter of the number of references that have been made to each page
  - Not common
- **Least Frequently Used** (**LFU**) **Algorithm**: replaces page with smallest count
- **Most Frequently Used** (**MFU**) **Algorithm**: based on the argument that the page with the smallest count was probably just brought in and has yet to be used

# *Summarization*