Operating Systems

BITS Pilani
K K Birla Goa Campus

Dr. Lucy J. Gudino
Dept. of CS and IS

Last Class

- Copy on write
- Page replacement algorithm
  - FIFO
    - Belady's Anomaly
  - Optimal
  - LRU
- LRU Implementation
  - using counter
  - using stack

Comparison of OPT with LRU and FIFO

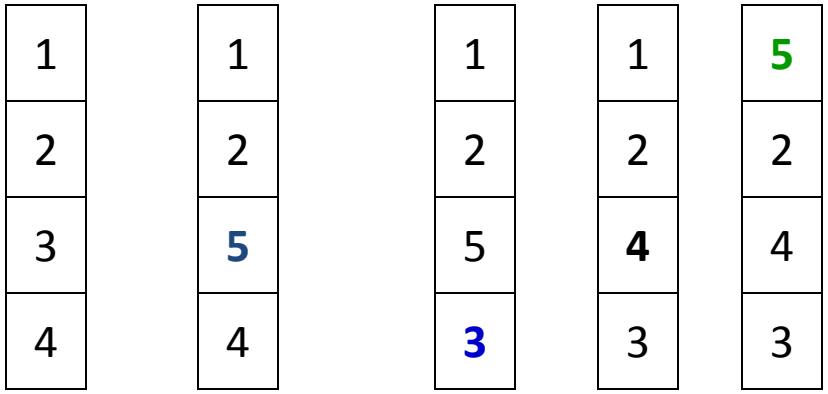
Reference string:

2 3 2 1 5 2 4 5 3 2 5 2

Assume three frames

Least Recently Used (LRU) Algorithm

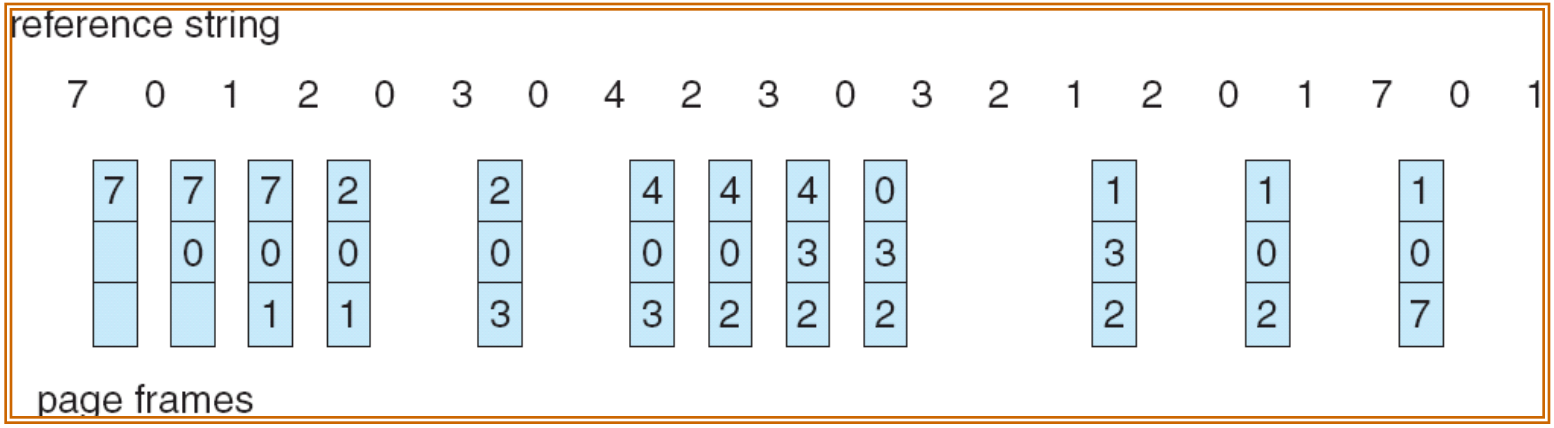
Reference string: 1, 2, 3, 4, 1, 2, **5**, 1, 2, **3**, **4**, **5**



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 determine which are to change

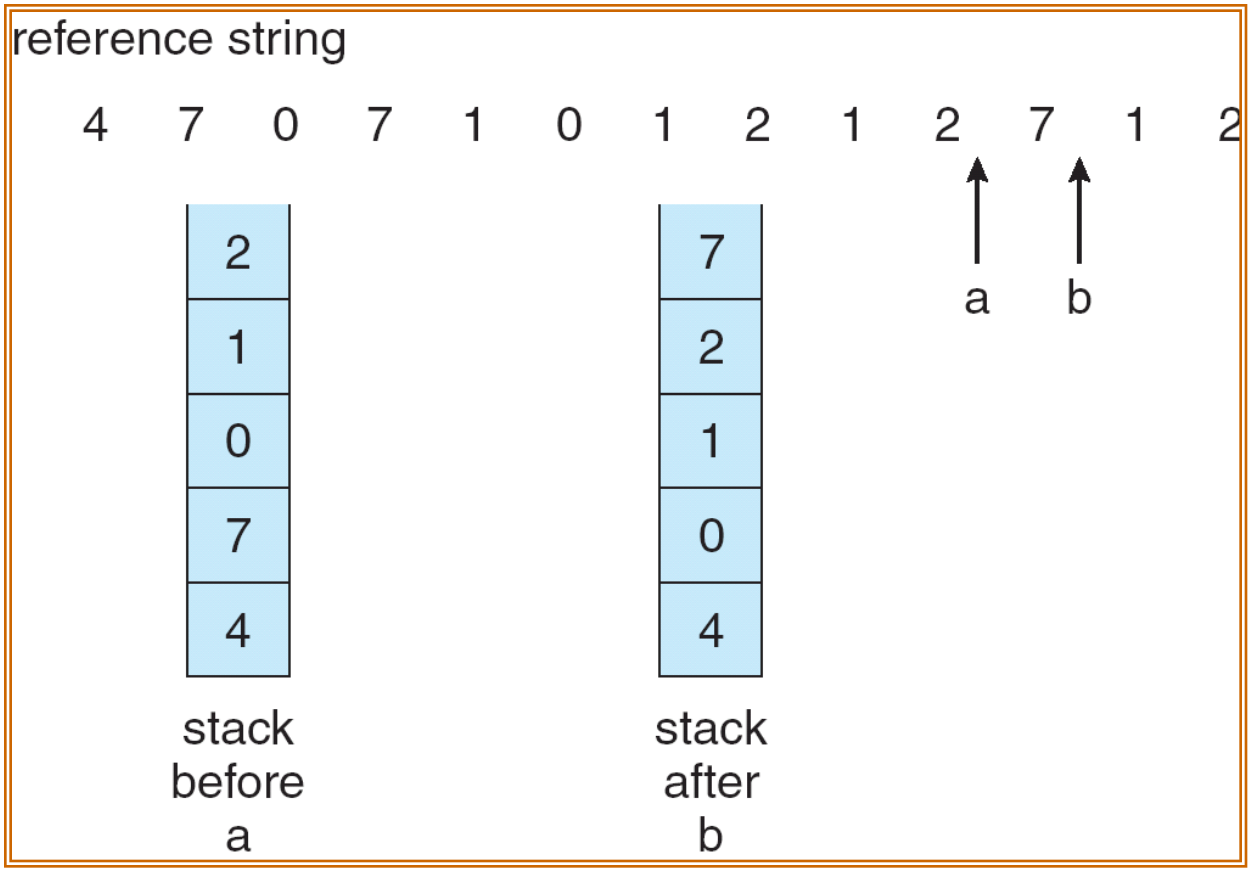
Example



LRU Algorithm (Cont.)

- Stack implementation – keep a stack of page numbers
  - Page referenced:
    - move it to the top
    - implemented using doubly linked list
  - No search for replacement
  - How many pointers are needed ?

Example


reference string
4 7 0 7 1 0 1 2 1 2 7 1 2
a
b
2
1
0
7
4
7
2
1
0
4
stack before a
stack after b

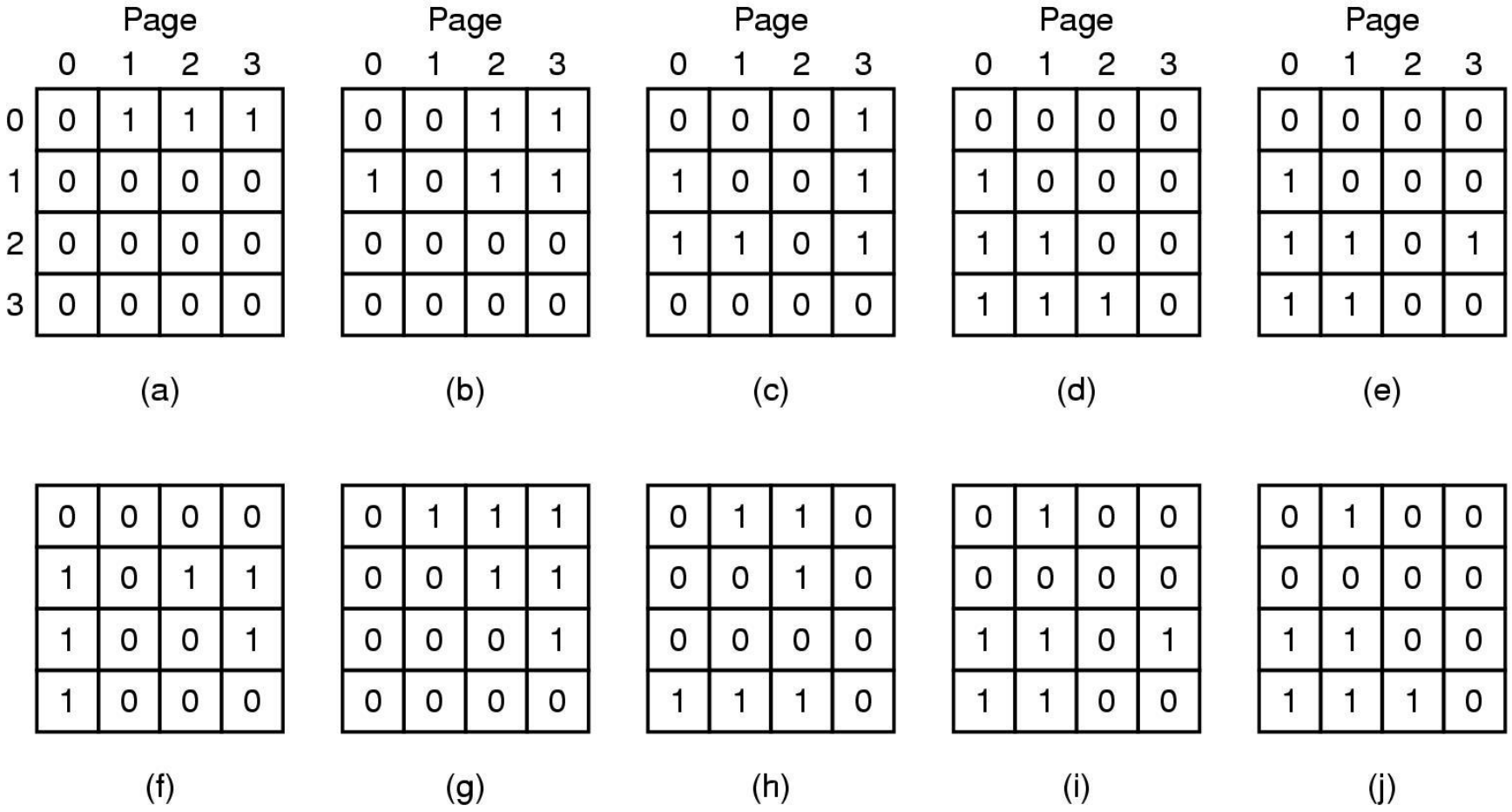
Advantages of Optimal and LRU page replacement algorithm

- never exhibits Belady's anamoly

Hardware Matrix LRU implementation

- For n pages, keep an nxn bit matrix.
- Whenever page frame k is referenced, the hardware first sets all the bits of row k to 1, then sets all the bits of column k to 0.
- At any instant, the row whose binary value is lowest is the least recently used,
- Consider a reference string : 0 1 2 3 2 1 0 3 2 3

Contd..


Page
0 1 2 3
0 0 1 1 1
1 0 0 0 0
2 0 0 0 0
3 0 0 0 0
(a)
Page
0 1 2 3
0 0 1 1
1 0 1 1
0 0 0 0
0 0 0 0
(b)
Page
0 1 2 3
0 0 0 1
1 0 0 1
1 1 0 1
0 0 0 0
(c)
Page
0 1 2 3
0 0 0 0
1 0 0 0
1 1 0 0
1 1 1 0
(d)
Page
0 1 2 3
0 0 0 0
1 0 0 0
1 1 0 1
1 1 0 0
(e)
0 0 0 0
1 0 1 1
1 0 0 1
1 0 0 0
(f)
0 1 1 1
0 0 1 1
0 0 0 1
0 0 0 0
(g)
0 1 1 0
0 0 1 0
0 0 0 0
1 1 1 0
(h)
0 1 0 0
0 0 0 0
1 1 0 1
1 1 0 0
(i)
0 1 0 0
0 0 0 0
1 1 0 0
1 1 1 0
(j)

LRU Approximation Algorithms

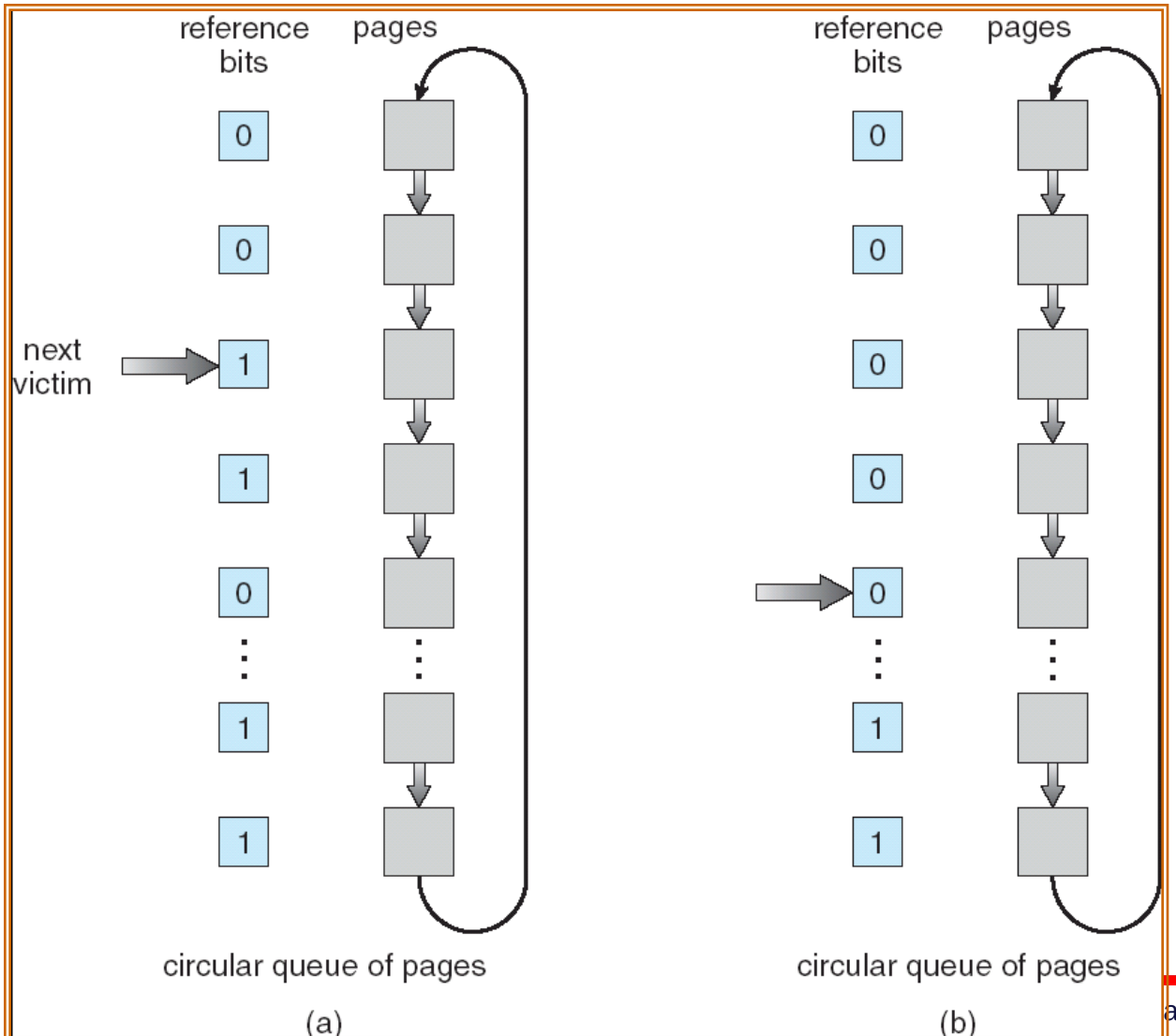
Reference bit

- With each page associate a bit, initially = 0
- When page is referenced bit set to 1
- Replace the one which is 0 (if one exists)
  - We do not know the order, however

Second chance

- Need reference bit
- Clock replacement
- If page to be replaced (in clock order) has reference bit = 1 then:
  - set reference bit 0
  - leave page in memory
  - replace next page (in clock order), subject to same rules

Second-Chance (clock) Page-Replacement Algorithm



Example

Reference string:

0 1 3 6 2 4 5 2 5 0 3 1 2 5 4 1 0

Assume 4 frames

Global vs. Local Allocation

Global replacement – process selects a replacement frame from the set of all frames; one process can take a frame from another

Local replacement – each process selects from only its own set of allocated frames

Thrashing

If a process does not have “enough” pages, the page-fault rate is very high. This leads to:

- low CPU utilization
- operating system thinks that it needs to increase the degree of multiprogramming
- another process added to the system

Thrashing ≡ a process is busy swapping pages in and out