Memory Management: Page Replacement Policies: LRU

CS 571: Operating Systems (Spring 2020)

Lecture 8c

Yue Cheng

Least-Recently-Used (LRU)

Least-Recently-Used Policy (LRU)

- Use the recent pass as an approximation of the near future (using history)
- Idea: evict the page that has not been used for the longest period of time

Least-Recently-Used Policy (LRU)

- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0			
1			
2			
0			
1			
3			
0			
3			
1			
2			
1			

Least-Recently-Used Policy (LRU)

- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0	Miss		LRU→ 0
1	Miss		LRU→ 0, 1
2	Miss		LRU→ 0, 1, 2
0			
1			
3			
0			
3			
1			
2			
1			

Least-Recently-Used Policy (LRU)

- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0	Miss		LRU→ 0
1	Miss		LRU→ 0, 1
2	Miss		LRU→ 0, 1, 2
0	Hit		LRU→ 1, 2, 0
1			
3			
0			
3			
1			
2			
1			

Least-Recently-Used Policy (LRU)

- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0	Miss		LRU→ 0
1	Miss		LRU→ 0, 1
2	Miss		LRU→ 0, 1, 2
0	Hit		LRU→ 1, 2, 0
1	Hit		LRU→ 2, 0, 1
3			
0			
3			
1			
2			
1			

Least-Recently-Used Policy (LRU)

- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0	Miss		LRU→ 0
1	Miss		LRU→ 0, 1
2	Miss		LRU→ 0, 1, 2
0	Hit		LRU→ 1, 2, 0
1	Hit		LRU→ 2, 0, 1
3	Miss	2	LRU→ 0, 1, 3
0			
3			
1			
2			
1			

Least-Recently-Used Policy (LRU)

- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0	Miss		LRU→ 0
1	Miss		LRU→ 0, 1
2	Miss		LRU→ 0, 1, 2
0	Hit		LRU→ 1, 2, 0
1	Hit		LRU→ 2, 0, 1
3	Miss	2	LRU→ 0, 1, 3
0	Hit		LRU→ 1, 3, 0
3			
1			
2			
1			

Least-Recently-Used Policy (LRU)

- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0	Miss		LRU→ 0
1	Miss		LRU→ 0, 1
2	Miss		LRU→ 0, 1, 2
0	Hit		LRU→ 1, 2, 0
1	Hit		LRU→ 2, 0, 1
3	Miss	2	LRU→ 0, 1, 3
0	Hit		LRU→ 1, 3, 0
3	Hit		LRU→ 1, 0, 3
1			
2			
1			

Least-Recently-Used Policy (LRU)

- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0	Miss		LRU→ 0
1	Miss		LRU→ 0, 1
2	Miss		LRU→ 0, 1, 2
0	Hit		LRU→ 1, 2, 0
1	Hit		LRU→ 2, 0, 1
3	Miss	2	LRU→ 0, 1, 3
0	Hit		LRU→ 1, 3, 0
3	Hit		LRU→ 1, 0, 3
1	Hit		LRU→ 0, 3, 1
2			
1			

Least-Recently-Used Policy (LRU)

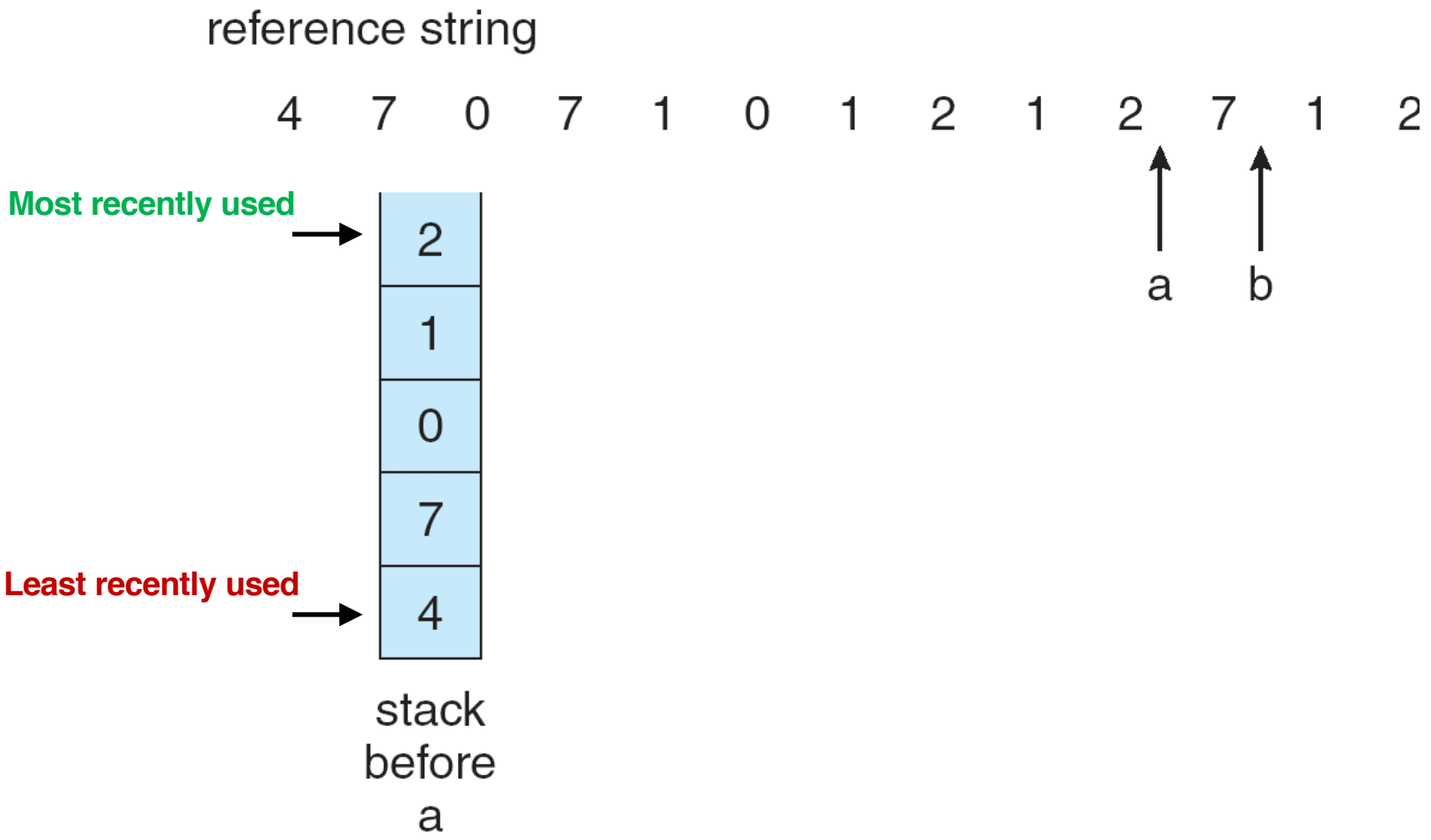
- Idea: evict the page that has not been used for the longest period of time
- Example workload: 0 1 2 0 1 3 0 3 1 2 1

Access	Hit/Miss?	Evict	Resulting Cache State
0	Miss		LRU→ 0
1	Miss		LRU→ 0, 1
2	Miss		LRU→ 0, 1, 2
0	Hit		LRU→ 1, 2, 0
1	Hit		LRU→ 2, 0, 1
3	Miss	2	LRU→ 0, 1, 3
0	Hit		LRU→ 1, 3, 0
3	Hit		LRU→ 1, 0, 3
1	Hit		LRU→ 0, 3, 1
2	Miss	0	LRU→ 3, 1, 2
1	Hit		LRU→ 3, 2, 1

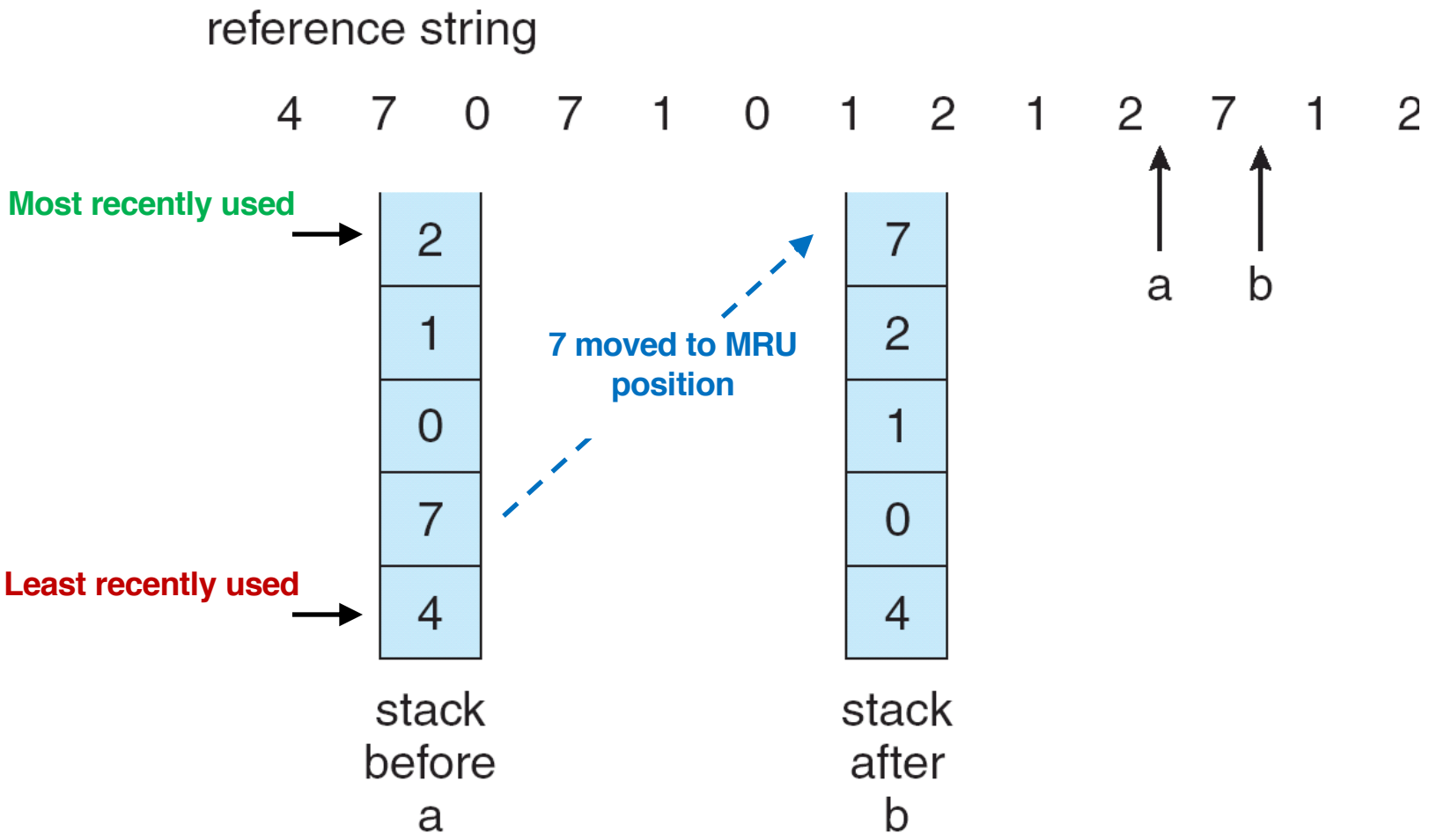
LRU Stack Implementation

- Stack implementation: keep a stack of page numbers in a doubly linked list form
  - Page referenced, move it to the top
  - Requires quite a few pointers to be changed
  - No search required for replacement operation!

Using a Stack to Approximate LRU



Using a Stack to Approximate LRU



LRU Hardware Support

- Sophisticated hardware support may involve high overhead/cost!

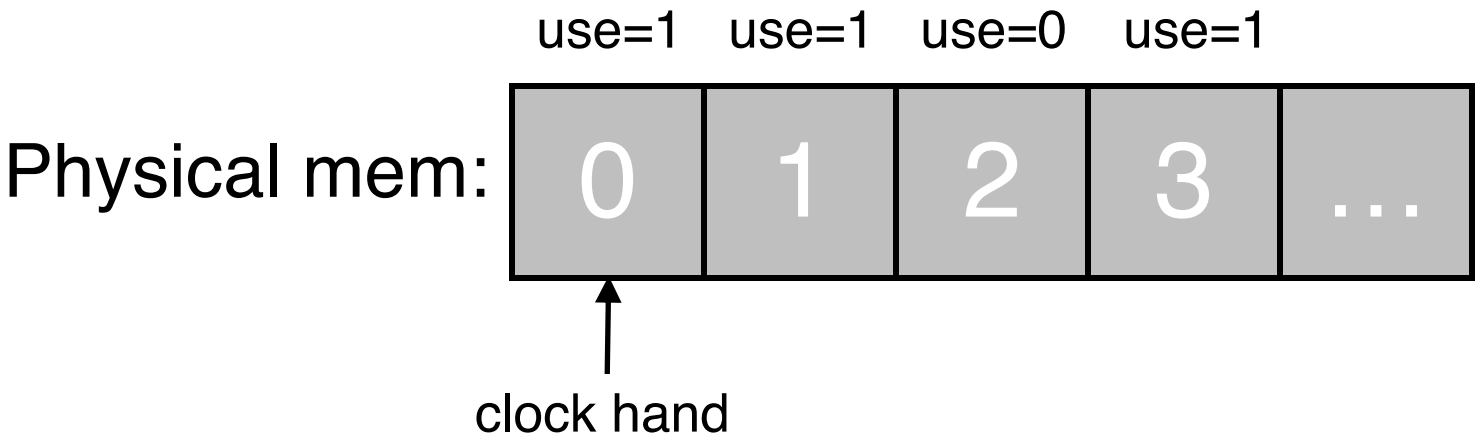
- Some limited HW support is common:

  Reference (or use) bit

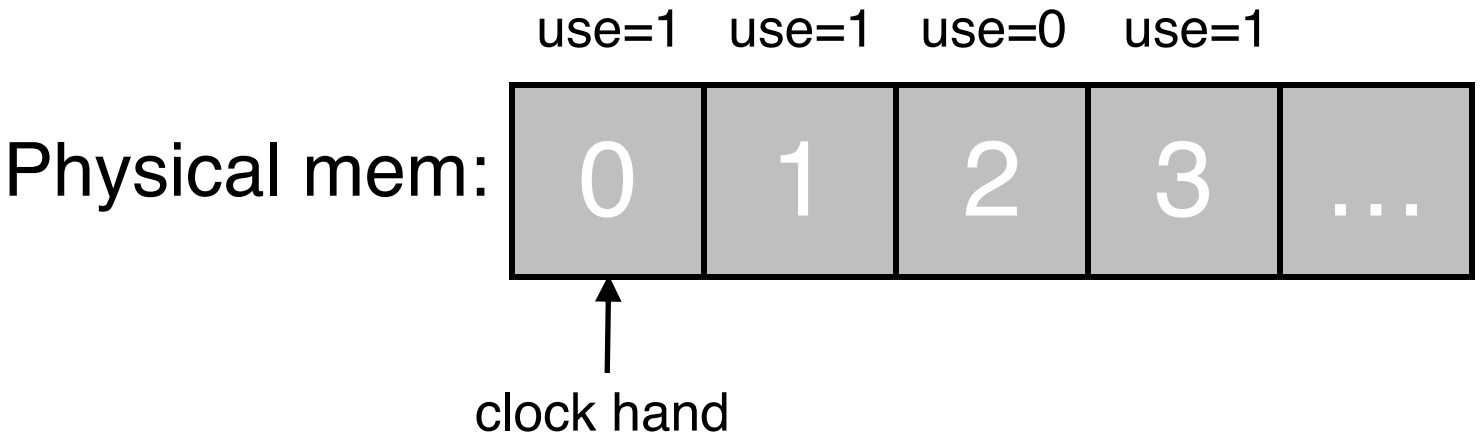
  - With each page associate a bit, initially set to 0
  - When the page is referenced, bit set to 1
  - By examining the reference bits, we can determine which pages have been used
  - We do not know the *order* of use, however!

- Cheap approximation
  - Useful for clock algorithm

Clock: Look For a Page

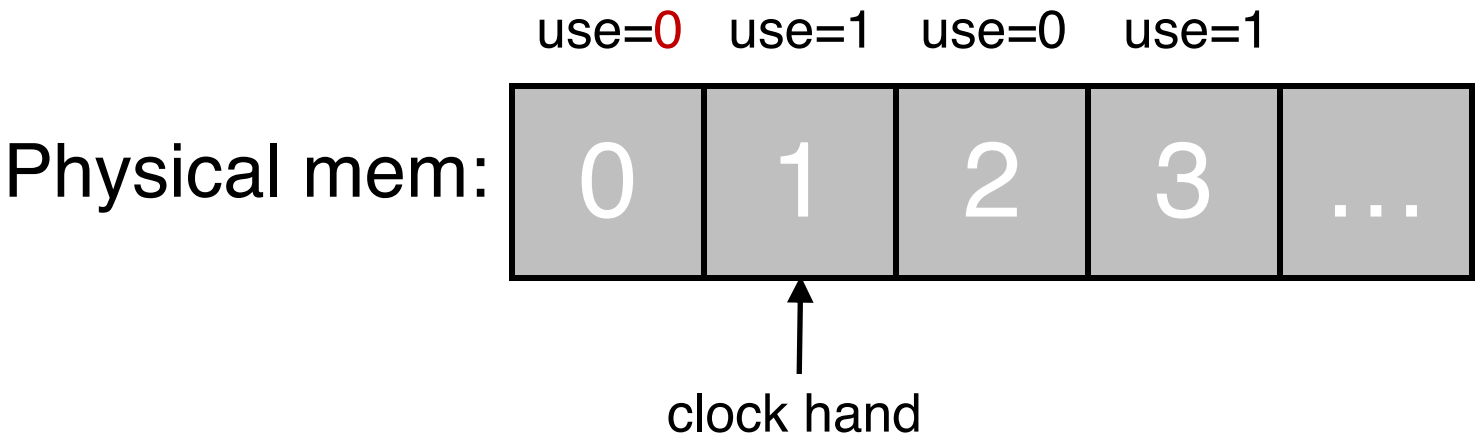


Clock: Look For a Page



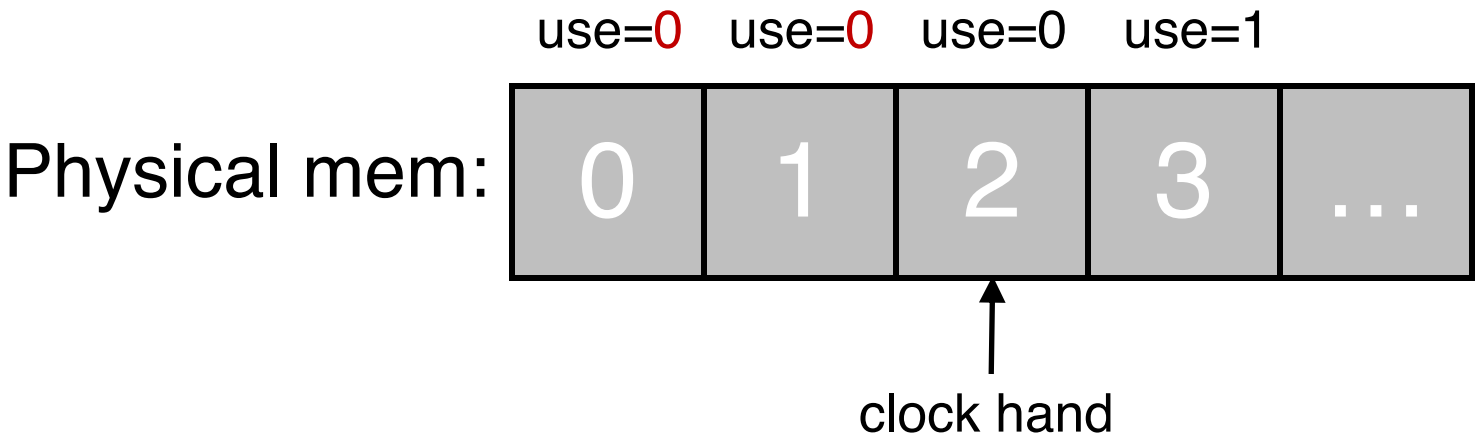
Mem is full, and to evict a page to make room

Clock: Look For a Page



Mem is full, and to evict a page to make room

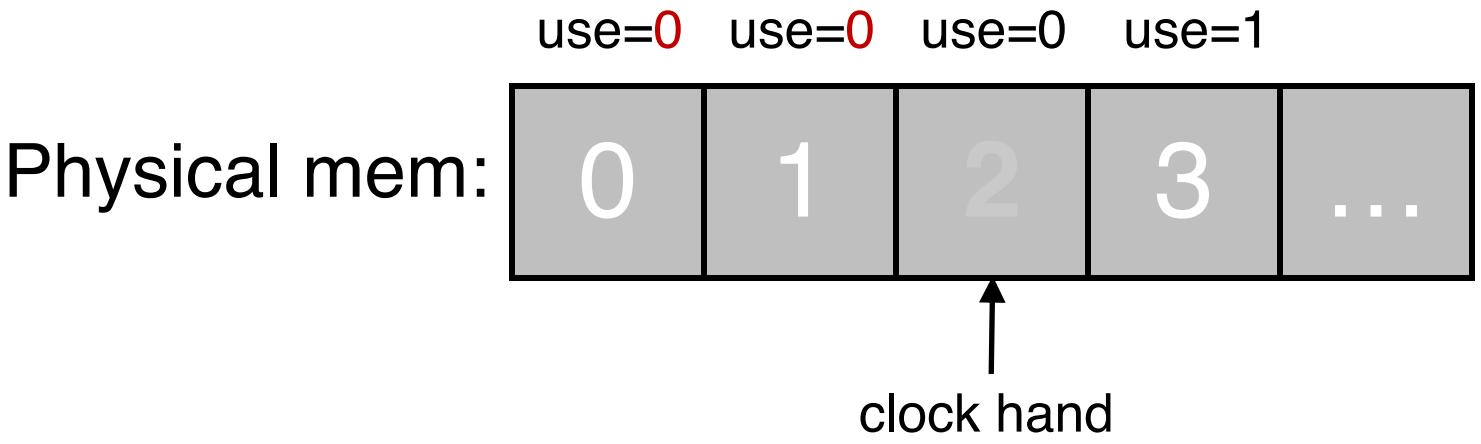
Clock: Look For a Page



Mem is full, and to evict a page to make room

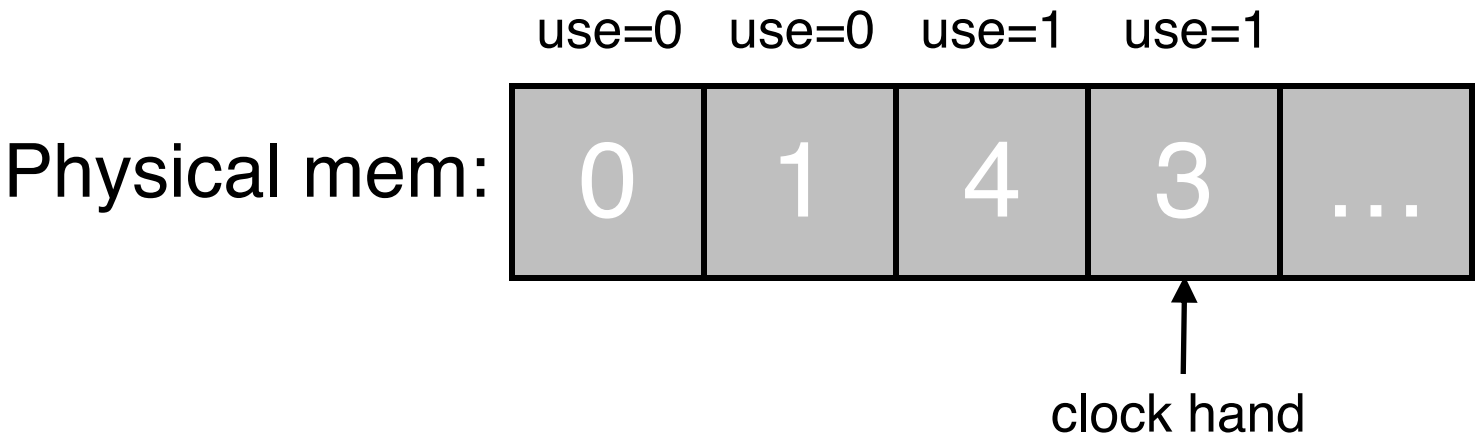
Clock: Look For a Page

Evict **page 2** because it has not been recently used



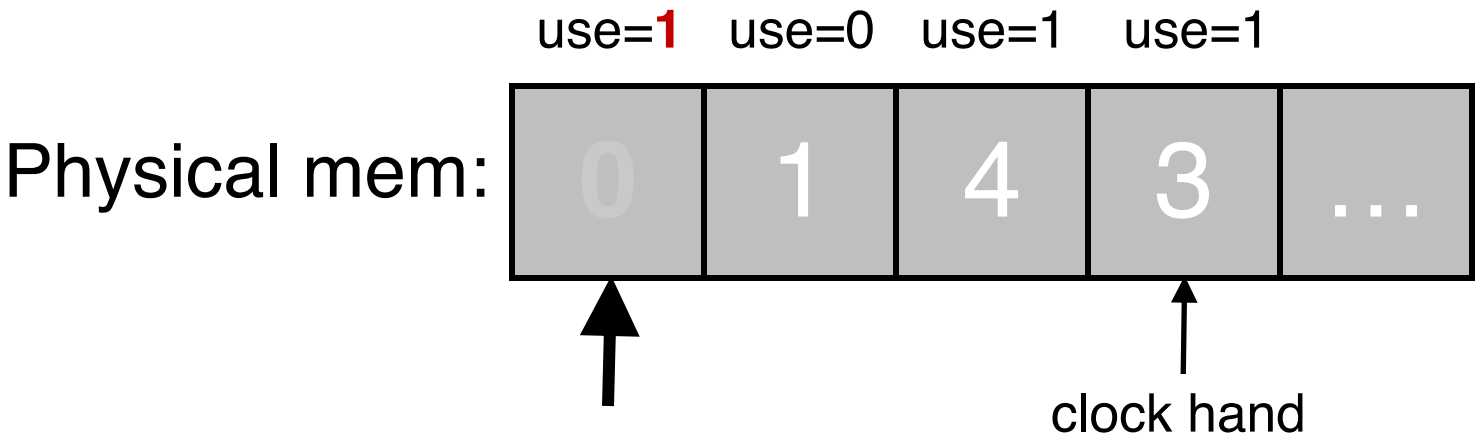
Mem is full, and to evict a page to make room

Clock: Look For a Page

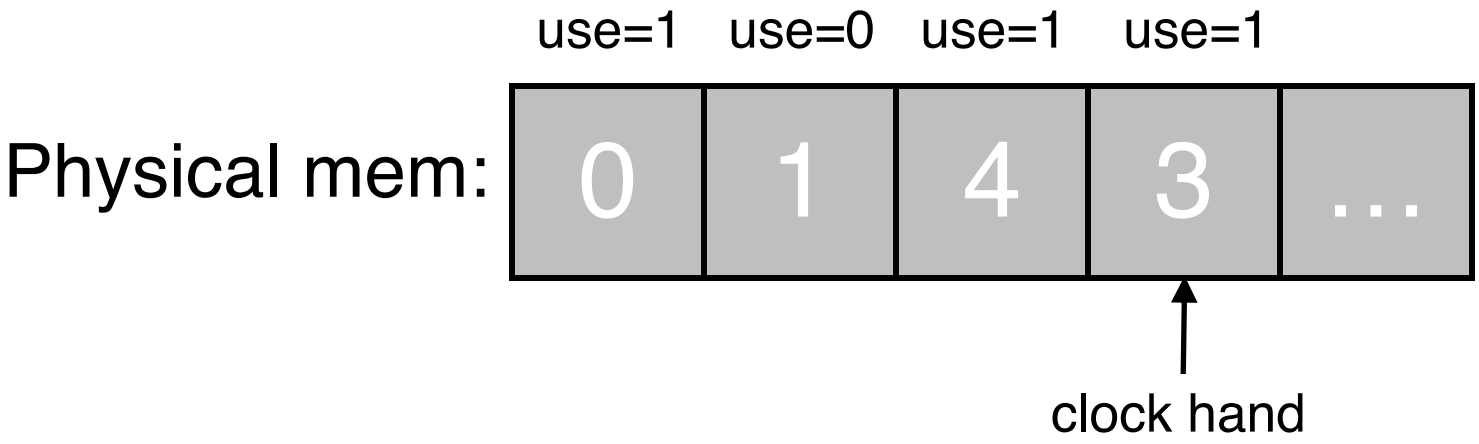


Clock: Access a Page

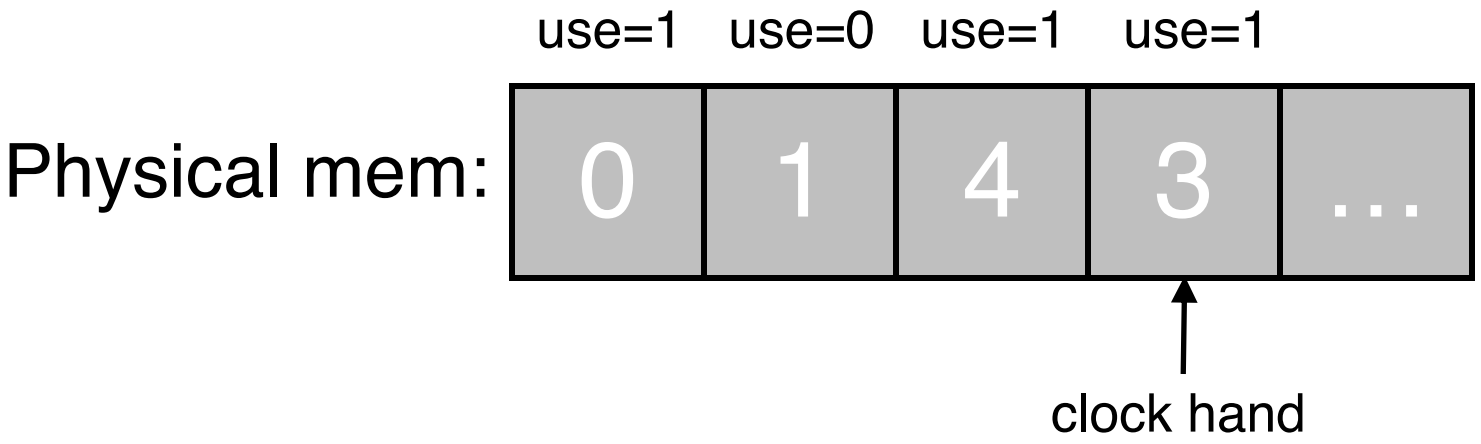
page 0 is accessed



Clock: Look For a Page

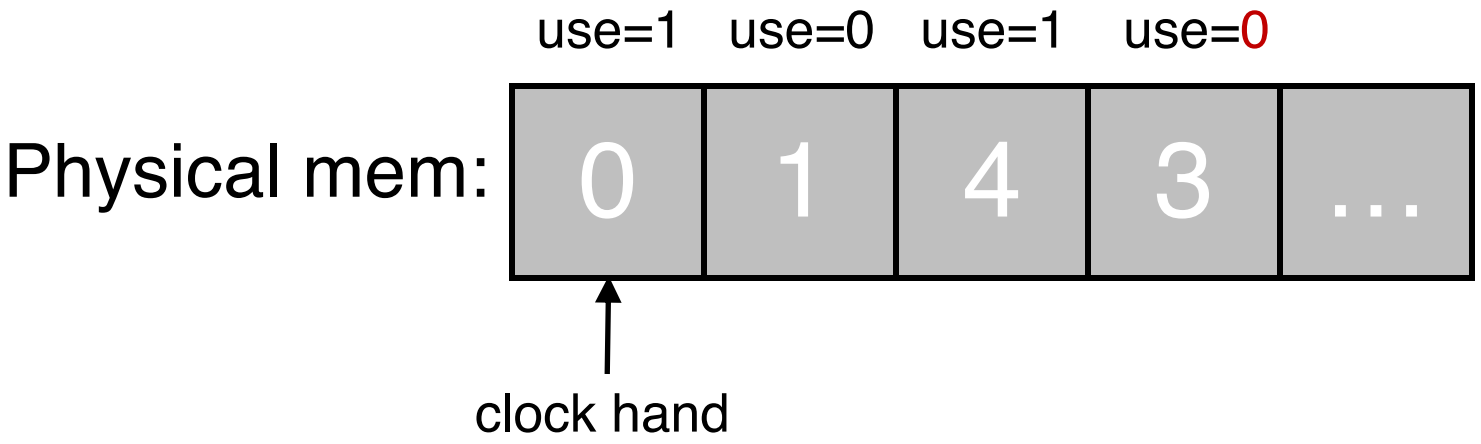


Clock: Look For a Page



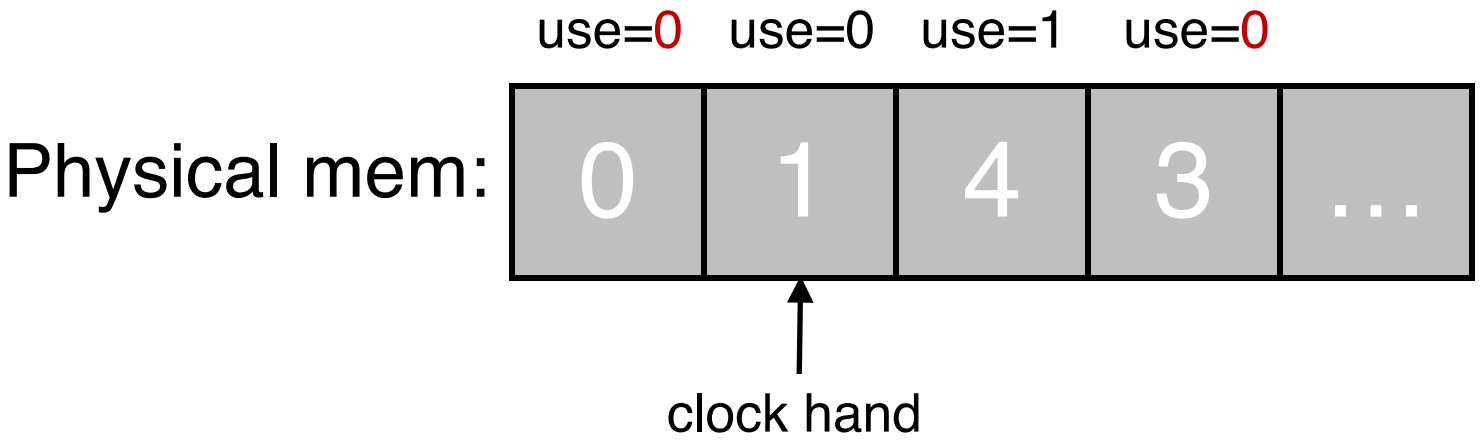
Mem is full, and to evict a page to make room

Clock: Look For a Page



Mem is full, and to evict a page to make room

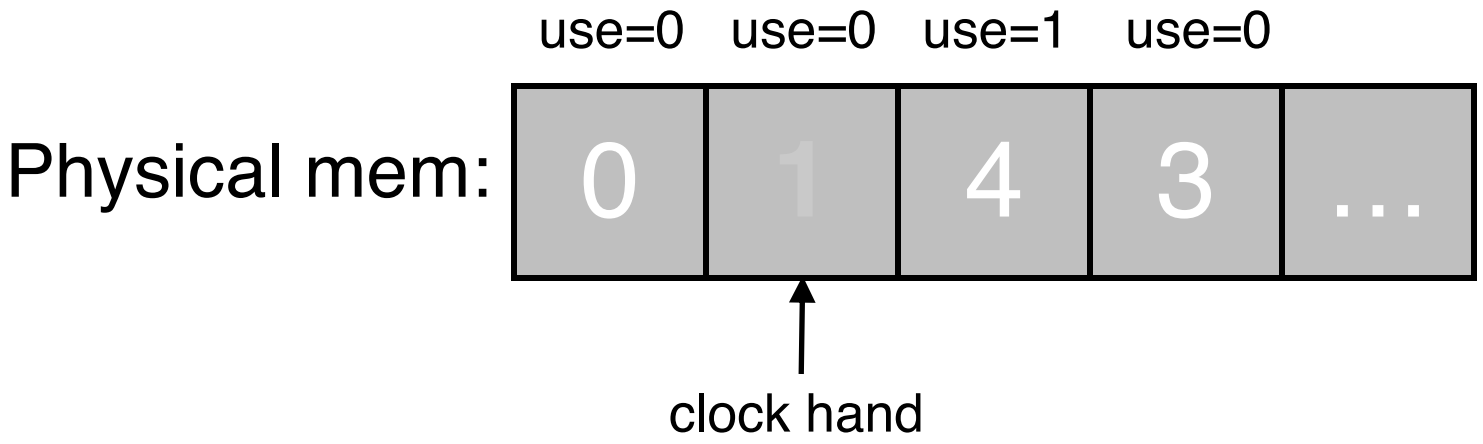
Clock: Look For a Page



Mem is full, and to evict a page to make room

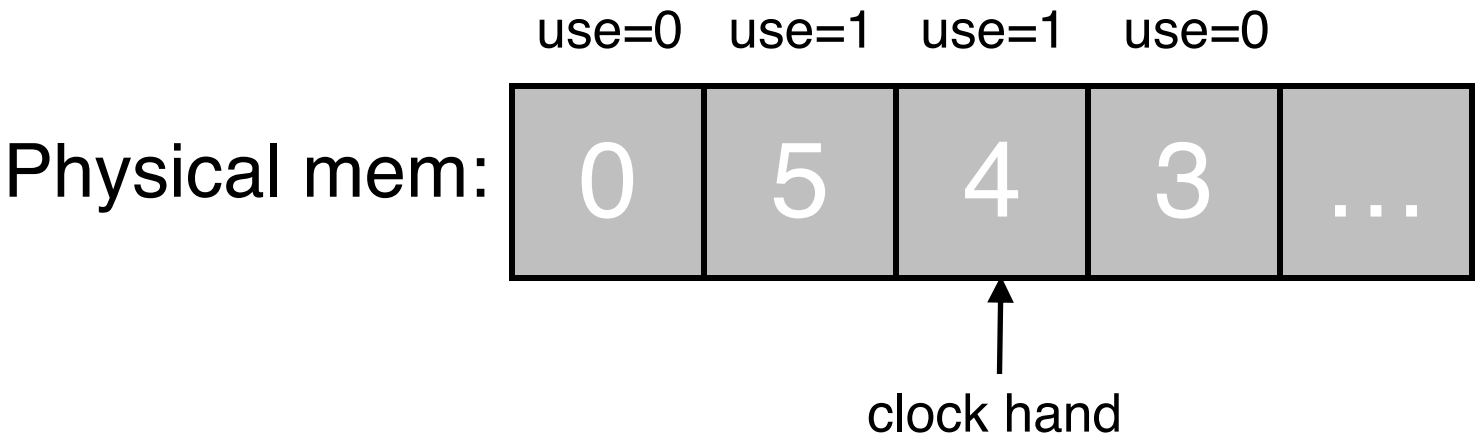
Clock: Look For a Page

Evict **page 1** because it has not been recently used



Mem is full, and to evict a page to make room

Clock: Look For a Page



Summary: Page Replacement Policies

- FIFO
  - Why it might work? Maybe the one brought in the longest ago is one we are not using now
  - Why it might not work? No real info to tell if it's being used or not
  - Suffers "Belady's Anomaly"

Summary: Page Replacement Policies

- FIFO
  - Why it might work? Maybe the one brought in the longest ago is one we are not using now
  - Why it might not work? No real info to tell if it's being used or not
  - Suffers "Belady's Anomaly"
- Random
  - Sometimes non intelligence is better

Summary: Page Replacement Policies

- FIFO
  - Why it might work? Maybe the one brought in the longest ago is one we are not using now
  - Why it might not work? No real info to tell if it's being used or not
  - Suffers "Belady's Anomaly"
- Random
  - Sometimes non intelligence is better
- OPT
  - Assume we know about the future
  - Not practical in real cases: offline policy
  - However, can be used as a best case baseline for comparison purpose

Summary: Page Replacement Policies

- FIFO
  - Why it might work? Maybe the one brought in the longest ago is one we are not using now
  - Why it might not work? No real info to tell if it's being used or not
  - Suffers "Belady's Anomaly"
- Random
  - Sometimes non intelligence is better
- OPT
  - Assume we know about the future
  - Not practical in real cases: offline policy
  - However, can be used as a best case baseline for comparison purpose
- LRU
  - Intuition: we can't look into the future, but let's look at past experience to make a good guess
  - Out "bet" is that pages used recently are ones which will be used again (principle of locality)