PAGE REPLACEMENT PERFORMANCE

Alec Braynen, Matthew Kramer,
Nghiem Ngo, and Sterling Price

PAGE REPLACEMENT STRATEGIES

1. First-In First-Out [FIFO]
2. Least Recently Used [LRU]
3. Least Frequently Used [LFU]
4. Most Frequently Used [MFU]
5. Optimal

FIRST-IN FIRST-OUT (FIFO)

- Objective: chooses the page that has been in the physical memory the longest time and replaces it with the new page.
- Implemented with a queue-like structure.
- Pages are never moved within the queue, only enqueued and dequeued.
- When a page needs to be replaced, the queue containing the pages in memory has the front element dequeued and the page to be swapped in is enqueued to the back.

LEAST-RECENTLY USED (LRU)

- Objective: replace the page that was last used farthest in the past.
- Implemented with a stack-like structure.
- Pages at the top of the stack have been used recently, while those at the bottom of the stack have been unreferenced for some time.
- When a page is referenced, it is moved to the top of the stack.
- When a page needs to be replaced, the page in memory at the bottom of the stack is removed, and the page to be swapped in is pushed to the top.

LEAST-FREQUENTLY USED (LFU)

- Objective: replace the page that has been referenced the least often.
- Implemented with a standard array.
- When a page is referenced, its associated count is incremented.
- When a page needs to be replaced, the page in memory with the lowest count is selected for replacement, the page to be swapped takes its place in the array, and the associated count is reset.

MOST-FREQUENTLY USED (MFU)

- Objective: replace the page that was referenced the most often.
- Implemented with a standard array.
- When a page is referenced, its associated count is incremented.
- When a page needs to be replaced, the page in memory with the highest count is selected for replacement, the page to be swapped in takes its place in the array, and the associated count is reset.

OPTIMAL

- Objective: replace the page that will be required farthest in the future.
- Implemented with a standard array.
- Pages are never moved within the array, only replaced when needed.
- When a page needs to be replaced, the page in memory that will be referenced the latest is selected for replacement.
- Requires preemptive knowledge about the incoming page references.

CONCLUSION

- Different page replacement strategies perform very differently according to the situations that they are handling.
- Overall, all algorithms are inferior to the optimal algorithm.
- However, while optimal page replacement is very efficient, it is not possible in real-world situations.

THANK YOU