Page replacement

S.Rajarajan

SASTRA

- When all of the frames in main memory are occupied and it is necessary to bring in a new page to satisfy a page fault, the replacement policy determines which page currently in memory is to be replaced.
- All of the policies have as their objective that the page that is removed should be the page least likely to be referenced in the near future.
- Because of the principle of locality, there is often a high correlation between recent referencing history and near-future referencing patterns.
- Thus, most policies try to predict future behavior on the basis of past behavior

Basic Algorithms

- Regardless of the resident set management strategy (discussed in the next subsection), there are certain basic algorithms that are used for the selection of a page to replace.
  - Optimal
  - Least recently used (LRU)
  - Least frequently used (LFU)
  - First-in-first-out (FIFO)
  - Clock

- The **optimal** policy selects for replacement that page for which the time to the next reference is the longest.
- It can be shown that this policy results in the fewest number of page faults.
- Clearly, this policy is impossible to implement, because it would require the operating system to have perfect knowledge of future events.
- However, it does serve as a standard against which to judge real world algorithms

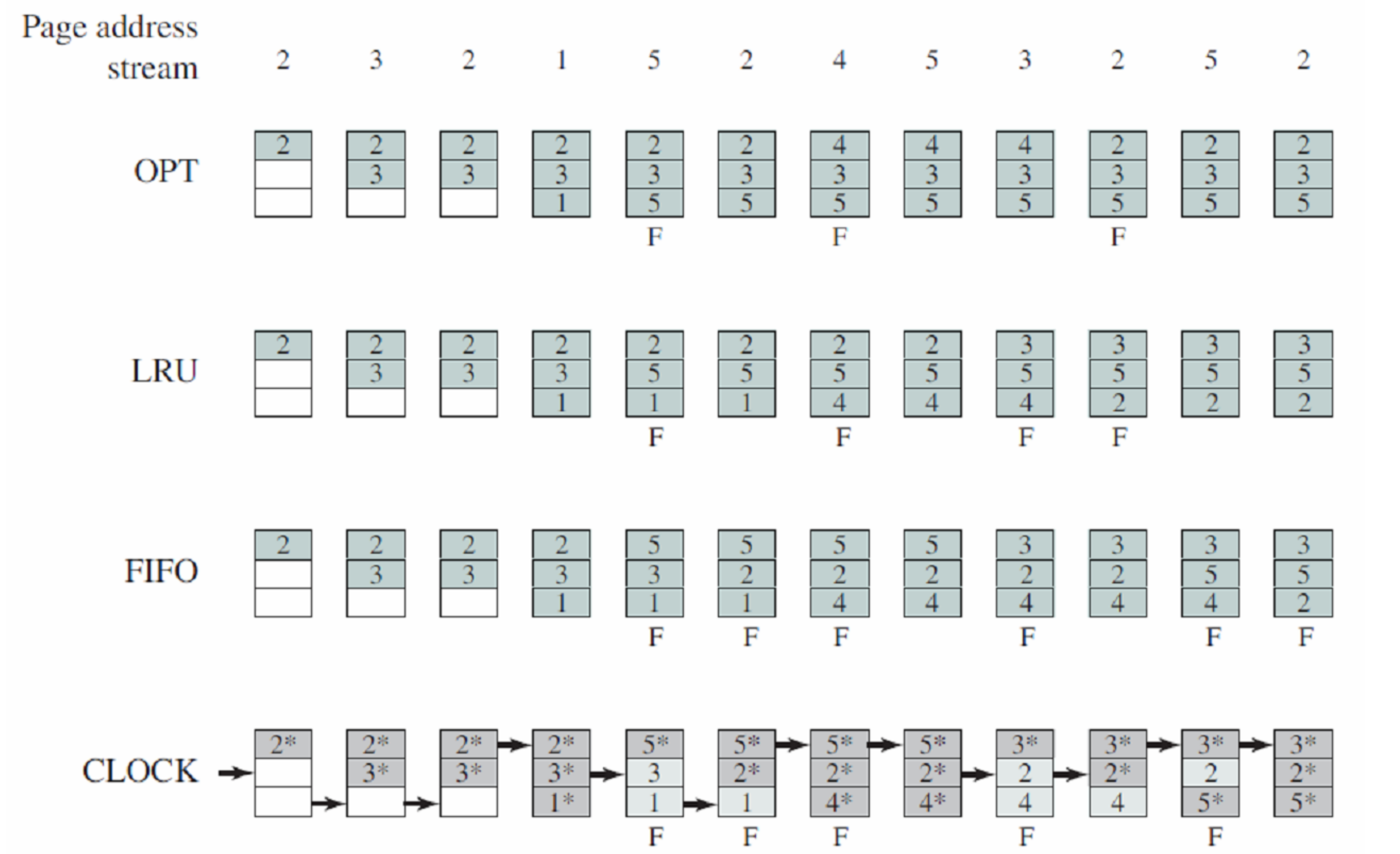
- The **least recently used** (LRU) policy replaces the page in memory that has not been referenced for the longest time. By the principle of locality, this should be the page least likely to be referenced in the near future.
- And, in fact, the LRU policy does nearly as well as the optimal policy.
- The problem with this approach is the difficulty in implementation.
- One approach would be to tag each page with the time of its last reference; this would have to be done at each memory reference, both instruction and data.
- Even if the hardware would support such a scheme, the overhead would be tremendous. Alternatively, one could maintain a stack of page references, again an expensive prospect

- The **first-in-first-out** (FIFO) policy treats the page frames allocated to a process as a circular buffer, and pages are removed in round-robin style.
- All that is required is a pointer that circles through the page frames of the process.
- This is therefore one of the simplest page replacement policies to implement

- Over the years, operating system designers have tried a number of other algorithms to approximate the performance of LRU while imposing little overhead. Many of these algorithms are
- variants of a scheme referred to as the **clock policy (second chance algorithm)**.
- The simplest form of clock policy requires the association of an additional bit with each frame, referred to as the use bit.
- When a page is first loaded into a frame in memory, the use bit for that frame is set to 1.Whenever the page is subsequently referenced (after the reference that generated the page fault), its use bit is set to 1.
- For the page replacement algorithm, the set of frames that are candidates for replacement is considered to be a circular buffer, with which a pointer is associated.

- When a page is replaced, the pointer is set to indicate the next frame in the buffer after the one just updated.
- When it comes time to replace a page, the operating system scans the buffer to find a frame with a use bit set to zero. Each time it encounters a frame with a use bit of 1, it resets that bit to zero and continues on.
- If any of the frames in the buffer have a use bit of zero at the beginning of this process, the first such frame encountered is chosen for replacement.
- If all of the frames have a use bit of 1, then the pointer will make one complete cycle through the buffer, setting all the use bits to zero, and stop at its original position, replacing the page in that frame

Given a sequence of page references and 3 frames, calculate the no. of page faults under each scheme:


Page address stream
2 3 2 1 5 2 4 5 3 2 5 2
OPT
F F F
LRU
F F F F
FIFO
F F F F F F
CLOCK
2* 3* 1* 5* 2* 4* 3* 5*
F F F F F