Replacement Strategies:

A virtual memory system is a combination of hardware and software techniques. The memory management software system handles all the software operations for the efficient utilization of memory space. It must decide (1) which page in main memory ought to be removed to make room for a new page, (2) when a new page is to be transferred from auxiliary memory to main memory, and (3) where the page is to be placed in main memory. The hardware mapping mechanism and the memory management software together constitute the architecture of a virtual memory.

When a program starts execution, one or more pages are transferred into main memory and the page table is set to indicate their position. The program is executed from main memory until it attempts to reference a page that is still in auxiliary memory. This condition is called *page fault*. When page fault occurs, the execution of the present program is suspended until the required page is brought into main memory. Since loading a page from auxiliary memory to main memory is basically an I/O operation, the operating system assigns this task to the I/O processor. In the meantime, control is transferred to the next program in memory that is waiting to be processed in the CPU. Later, when the memory block has been assigned and the transfer completed, the original program can resume its operation.

When a page fault occurs in a virtual memory system, it signifies that the page referenced by the CPU is not in main memory. A new page is then transferred from auxiliary memory to main memory. If main memory is full, it would be necessary to remove a page from a memory block to make room for the new page. The policy for choosing pages to remove is determined from the replacement algorithm that is used. The goal of a replacement policy is to try to remove the page least likely to be referenced in the immediate future.

Memory Reference Strings

Assume the following address sequence:
(e.g. recorded by tracing the memory accesses of a process)

0100, 0432, 0101, 0612, 0102, 0103, 0104, 0101, 0611, 0102, 0103
0104, 0101, 0610, 0102, 0103, 0104, 0101, 0609, 0102, 0105

Assuming a page size of 100 bytes, the sequence can be reduced to

1, 4, 1, 6, 1, 6, 1, 6, 1, 6, 1

This memory reference string lists the pages accessed over time (at the time steps at which page access changes).

If there is only 1 frame available, the sequence would cause 11 page faults.

If there are 3 frames available, the sequence would cause 3 page faults.

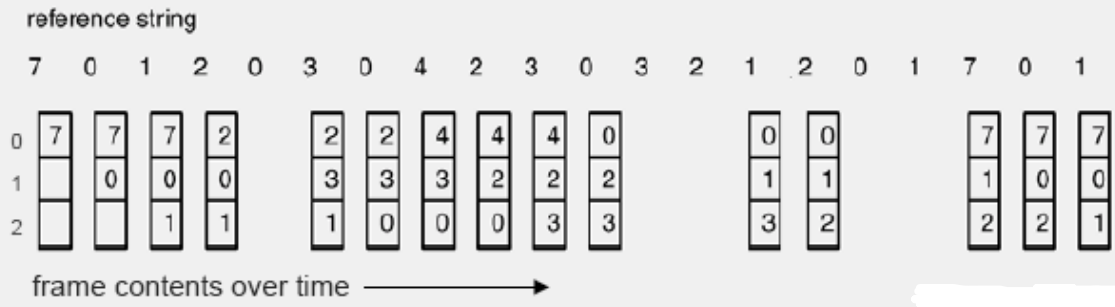
FIFO Page Replacement

Principle: Replace the oldest page (old = swap-in time).

Example VM.1

Memory reference string: 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

Number of frames: 3



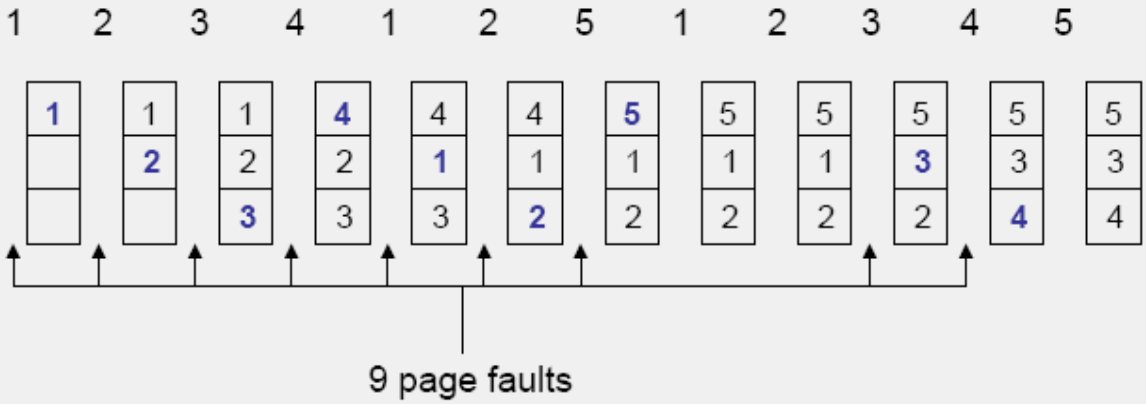
Total: 15 page faults.

FIFO Page Replacement

Example VM.2

Memory reference string: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5

Number of frames: 3

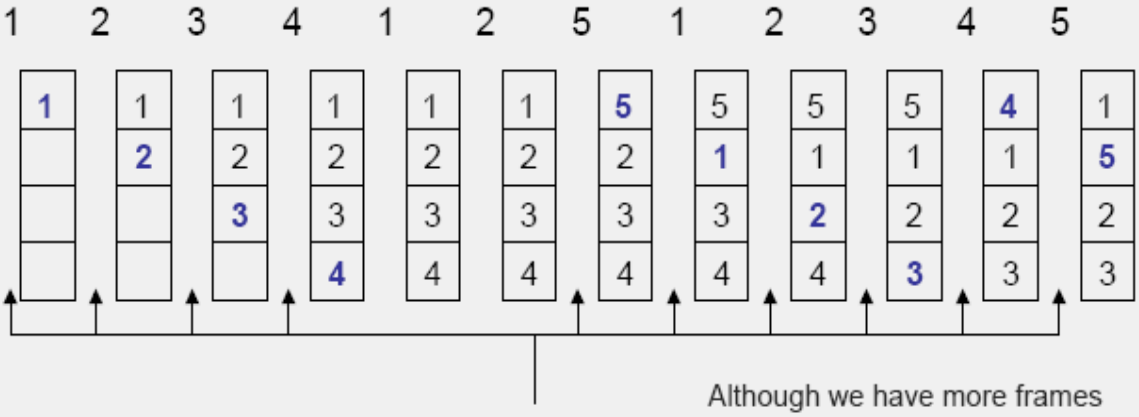


FIFO Page Replacement

Example VM.3

Memory reference string: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5 (as in VM.2)

Number of frames: 4



FIFO Page Replacement

From the examples VM.2, VM.3 it can be noticed that
the number of page faults for 4 frames is greater than for 3 frames.

This unexpected result is known as Belady's Anomaly[1]:

For some page-replacement algorithms the page-fault rate may increase as the number of allocated frames increases.

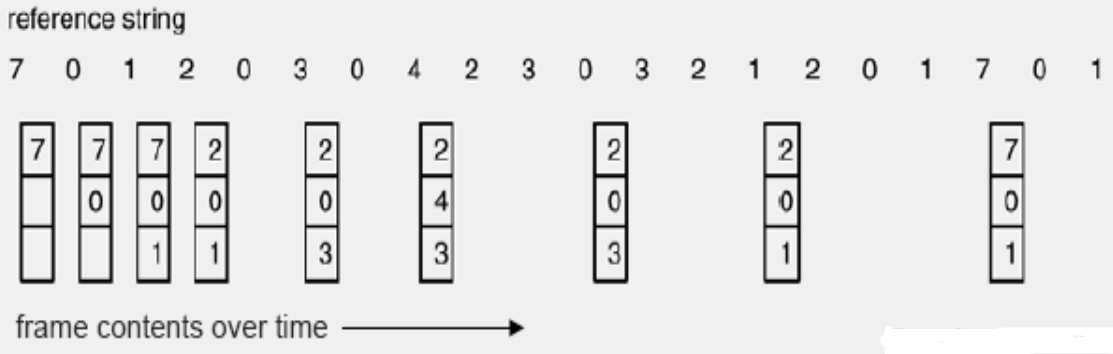
Optimal Page Replacement

Principle: Replace the page that *will* not be used for the longest time.

Example VM.4

Memory reference string: 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

Number of frames: 3



Total: 9 page faults.

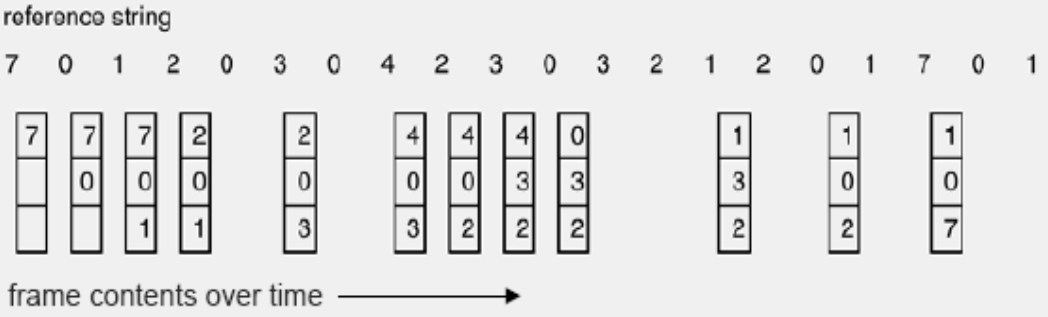
LRU Page Replacement

Principle: Replace the page that *has not been* used for the longest time.

Example VM.5

Memory reference string: 7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

Number of frames: 3



Page Replacement

Algorithms Summary

- First-in first-out (FIFO)

  Simplest algorithm, easy to implement, but has worst performance. The *clock* version is somewhat better as it does not replace busy pages.

- Optimal page replacement (OPT)

  Not of practical use as one must know future! Used for comparisons only. Lowest page fault rate of all algorithms.

- Least Recently Used (LRU)

  The best algorithm usable, but requires much execution time or highly sophisticated hardware.