Page Replacement

Reading: Section 4.4 of Tanenbaum or 10.4 of Siberschatz

When a page fault occurs, the OS has to choose a page to remove from memory to make the room for the page that has to be brought in.

Which one page to be removed ?

What happen if the page that required next, is removed?

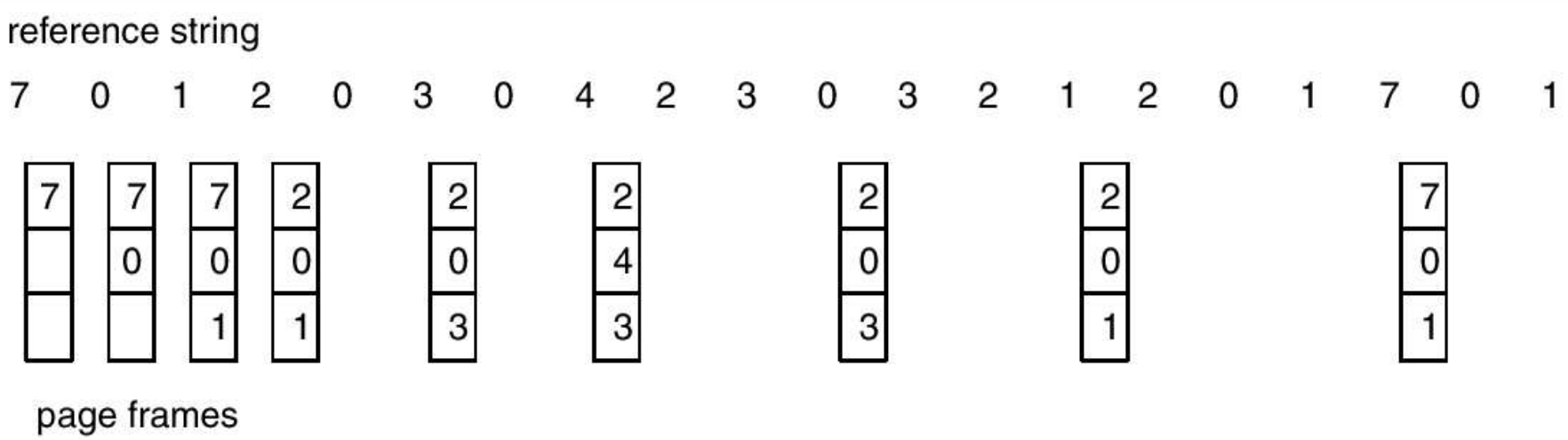
Principle of Optimality: *To obtain optimal performance the page to replace is one that will not be used for the furthest time in the future.*

Optimal Page Replacement (OPR)

Replace the page that will not be used for the longest period of time.

Each page can be labeled with number of instructions that will be executed before that page is first referenced.

Ex: For 3 - page frames and 8 pages system the optimal page replacement is as:



Optimal Page Replacement

Advantages:

An optimal page-replacement algorithm; it guarantees the lowest possible page fault rate.

Problems:

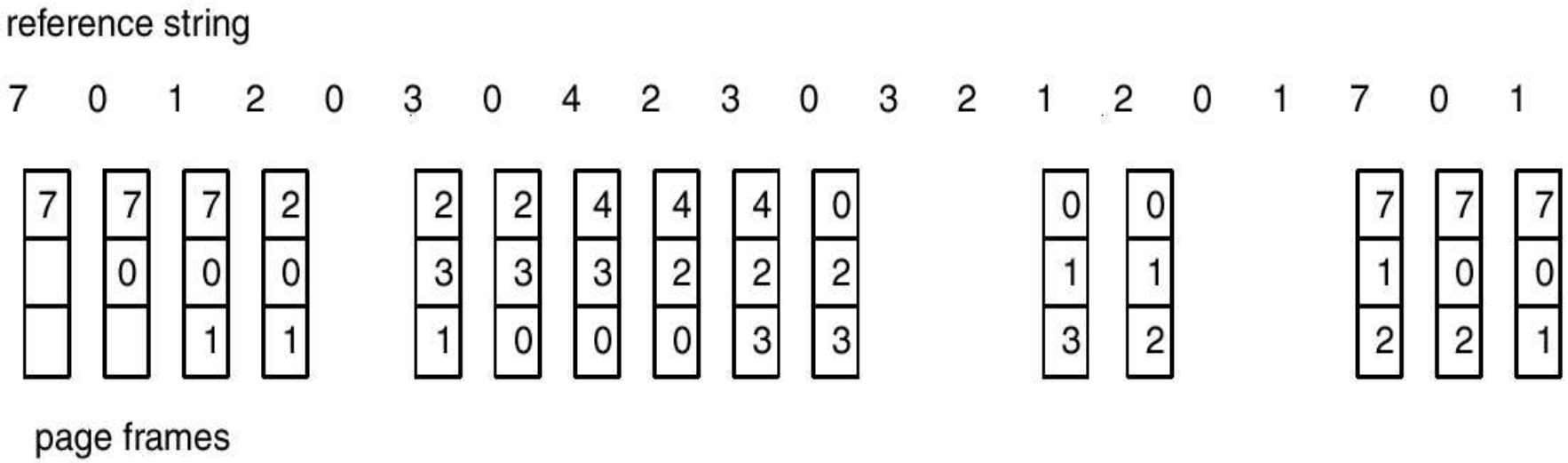
Unrealizable, at the time of the page fault, the OS has no way of knowing when each of the pages will be referenced next.

This is not used in practical system.

First-In-First-Out (FIFO)

This associates with each page the time when that page was brought into the memory. The page with highest time is chosen to replace.

This can also be implemented by using queue of all pages in memory.



First-In-First-Out (FIFO)

Advantages:

Easy to understand and program.

Distributes fair chance to all.

Problems:

FIFO is likely to replace heavily (or constantly) used pages and they are still needed for further processing.

Second Chance

FIFO to avoid the mis-replacing of heavily used pages.

The reference bit is also checked and if it is present then its entry is updated as it has been just arrived into the system.
When a page gets a second chance, its reference bit is cleared and its arrival time is reset to the current time.

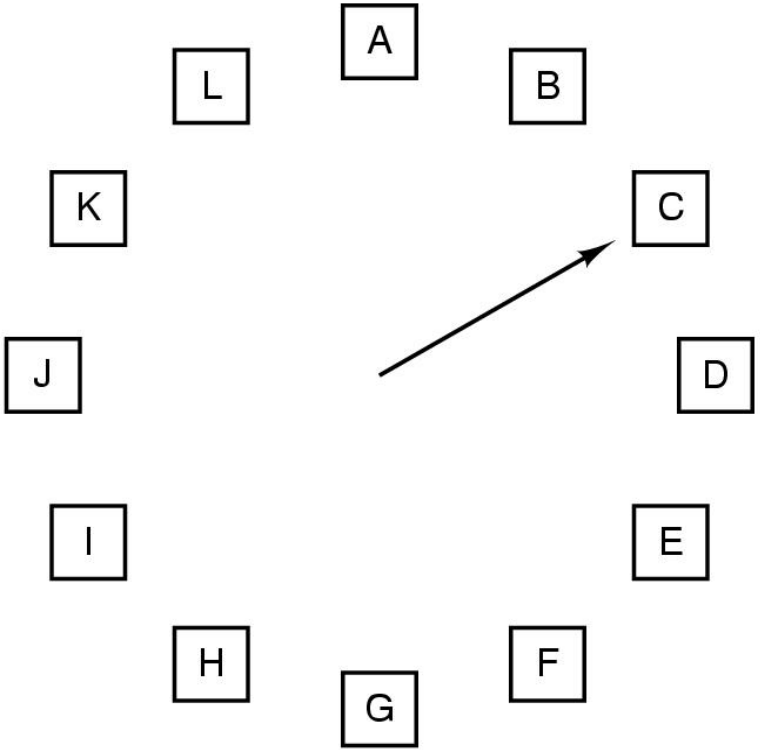
Advantages:

Big improvement over FIFO.

Problems:

If all the pages have been referenced, second chance degenerates into pure FIFO.

Clock Page Replacement



When a page fault occurs, the page the hand is pointing to is inspected. The action taken depends on the R bit:

R = 0: Evict the page

R = 1: Clear R and advance hand

Arrange the pages in a circular list instead of a linear list. Differ from second chance only in implementation.

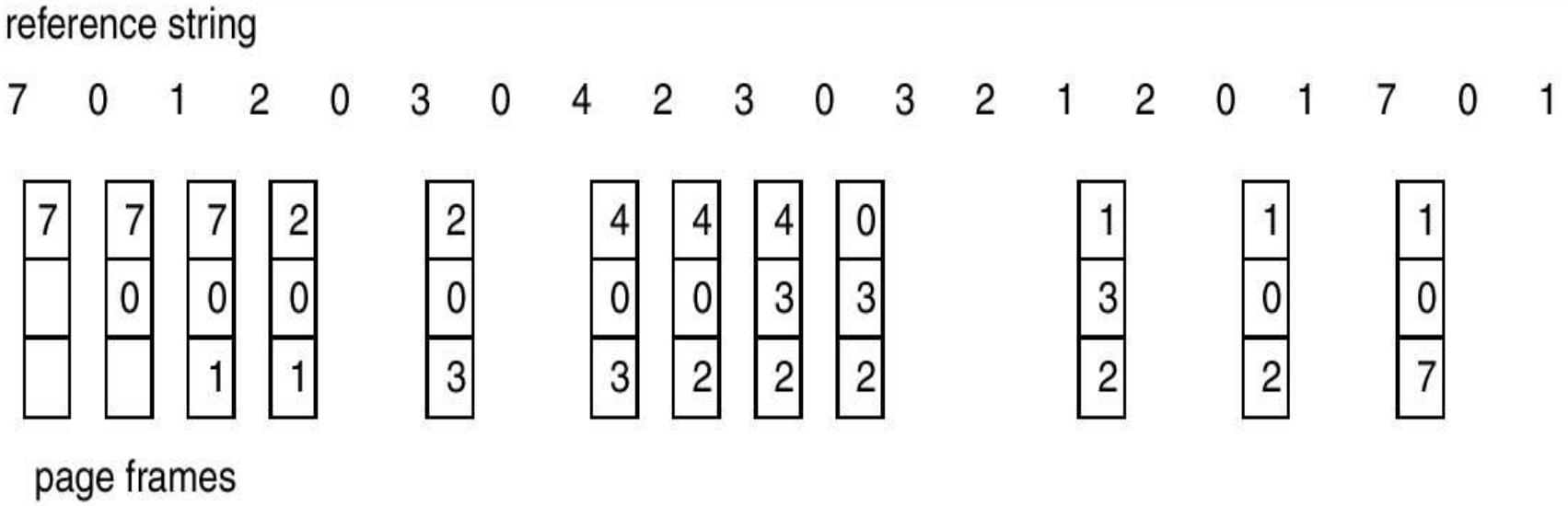
Advantages: More efficient than second chance.

Least Recently Used (LRU)

Recent past is a good indicator of the near future.

When a page fault occurs, throw out the page that has been unused for longest time.

It maintain a linked list of all pages in memory with the most recently used page at the front and least recently used page at the rare. The list must be updated on every memory reference.



Least Recently Used (LRU)

Advantages:

Excellent, efficient is close to the optimal algorithm.

Problems:

Difficult to implement exactly.

How to find good heuristic?

If it is implemented as linked list, updating list in every reference is not a way making system fast!

The Alternate implementation is by hardware primitivs, it requires a time-of-use field in page table and a logical clock or counter in the CPU.

Least Frequently Used (LFU)

One approximation to LRU, software implementation.

LFU requires that the page with the smallest count be replaced.
The reason for this selection is that an actively used page should have a large reference count.
It requires a software counter associated with each page. When page fault occure the page with lowest counter is chosen for replacement.

Problem: likely to replace highly active pages.
This algorithm suffers from the situation in which a page is used heavily during the initial phase of a process, but then is never used again.

Not Recently Used (NRU)

LRU approximation by enhancing second chance.

Pages not recently used are not likely to be used in near future and they must be replaced with incoming pages.

To keep useful statistics about which pages are being used and which pages are not, most computers have two status bits associated with each page – referenced and modified.
These bits must be updated on every memory reference.
When a page fault occurs, the OS inspects all the pages and divides them into four categories based on the current values of their referenced and modified bits.

Not Recently Used (NRU)

Class 0: not referenced, not modified.
Class 1: not referenced, modified.
Class 2: referenced, not modified.
Class 3: referenced, modified.

Pages in the lowest numbered class should be replaced first, and those in the highest numbered groups should be replaced last. Pages within the same class are randomly selected.

Advantage: Easy to understand and efficient to implement.
Problem: Class 1 unrealistic.

Working Set (WS) Page Replacement

In multiprogramming, processes are frequently move to disk to let other process have a turn at the CPU.

What to do when a process just swapped out and another process has to load?

The set of pages that a process is currently using is called its working set.

If the entire working set is in memory, the process will run without causing many faults until it moves into another execution. Otherwise, excessive page fault might occur called *thrashing*.

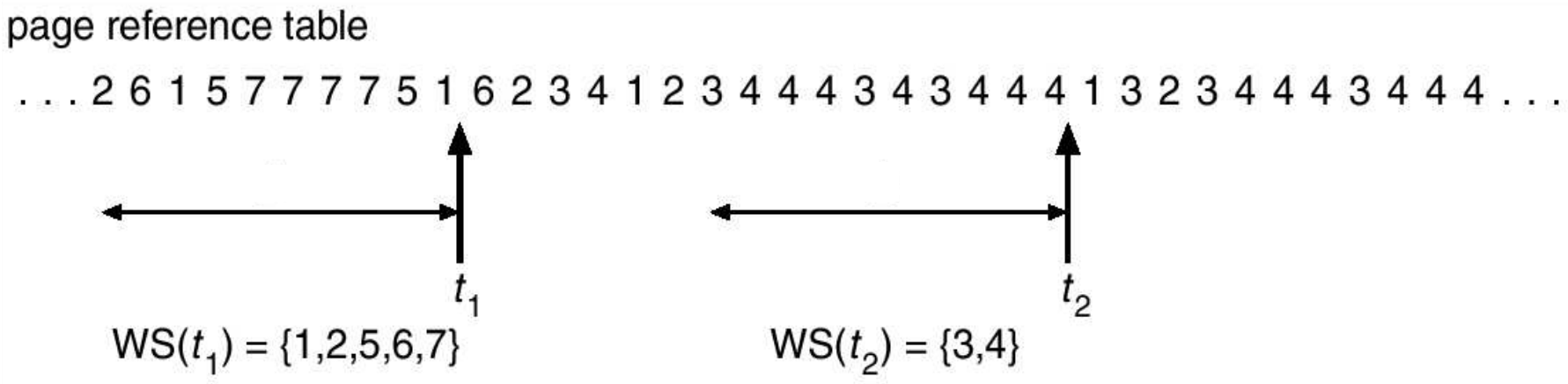
Many paging systems try to keep track of each process' working set and make sure that it in memory before the process run--*working set model or prepaging*.

Working Set (WS) Page Replacement

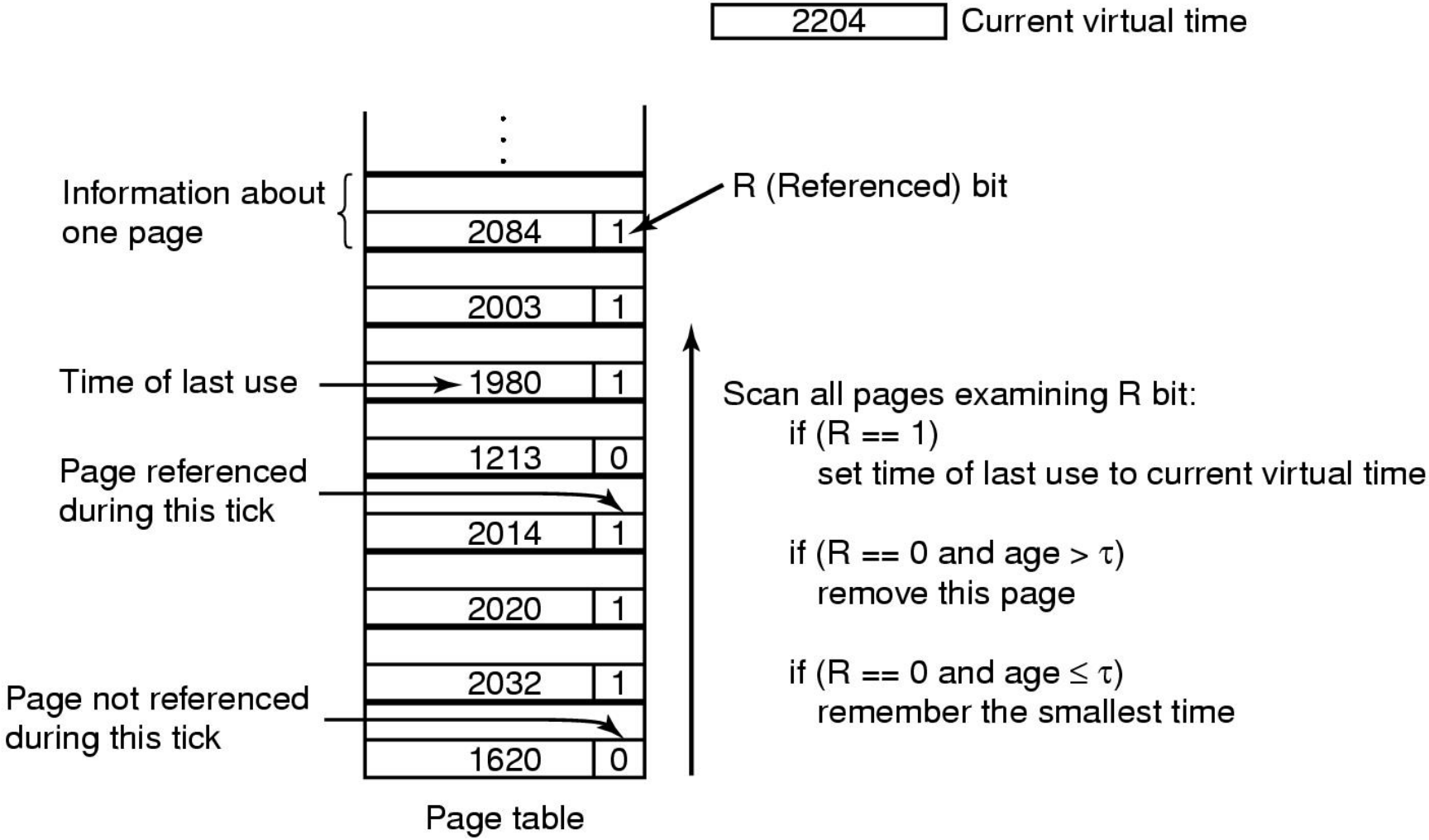
The working set of pages of process, ws(t,) at time t, is the set of pages referenced by the processes in time interval t-k to t.

The variable is called working-set-window, the size of is central issue in this model.

Ex: Working-set-model with = 10.



Working Set (WS) Page Replacement



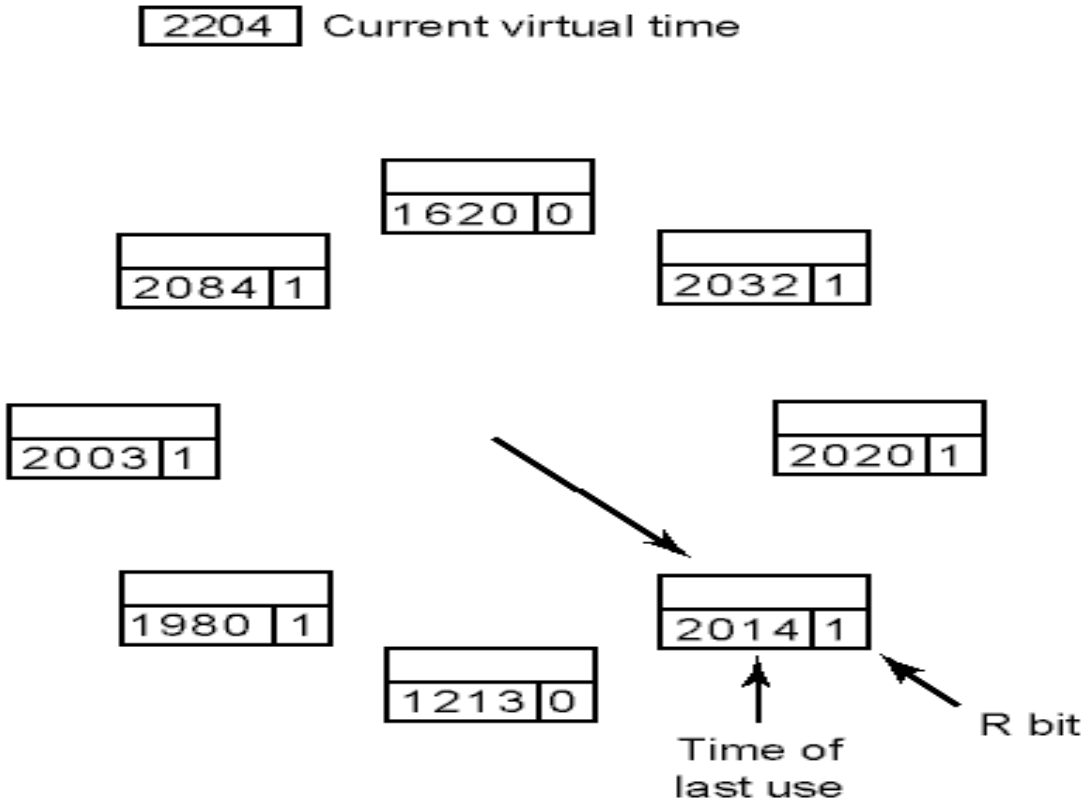
Page replacement with working set model

Efficient but expensive to implement.

WSClock Page Replacement

Improved WS. Implement as clock with working set.

Each entry contain the time of last use field and reference bit.



Good efficient and widely used algorithm.

Home Works

HW#9:

1. 23, 24, 25, 26, 27 & 29 from your Textbook(Tanenbaum) Ch. 4.
2. Under what circumstances do page fault occur? Describe the action taken by the OS when a page fault occurs.
3. Given references to the following pages by a program,
   1,2,3,4,2,1,5,6,2,1,2,3,7,6,3,2,1,2,3,6.
   How many page faults will occur if the program has three page frames available to it and uses:
   a) FIFO replacement?
   b) LRU replacement?
   c) Optimal replacement?
   (Remember that all frames are initially empty)

Read: Segmentation, Section 4.8 (Tanenbaum).