Virtual Memory
Page Replacement Strategies

COMP755

Implementing Algorithms

- The PageFault method is called whenever the simulation system determines that the simulated program requires a page whose resident bit is clear
- The PageFault method needs to make a decision based on the information available
- The simulated OS does not change

Page Table Contents

- **Resident bit** – true if the page is in RAM. If false, a page fault will occur.
- **Reference bit** – set by the simulator when page is used.
- **Dirty bit** – set by the simulator when page is changed
- **RamAllocated** – Count of the pages with the resident bit set

Extra Information

- The frameAddress is an integer field that can be used to keep additional information about each page

Simulated Time

- The Student.java class can create a simulated time with an integer counter as a class variable
- The counter should be incremented at the beginning of PageFault
- The time counter is not real time, but it always increases and orders events like real time

Replacement Policies

Read chapter 9.4 of the text

- **FIFO** - denies the concept of locality
- **Clock** - used by Mac OS
- **LRU** – local or global
- **LFU** – Least Frequently Used
- **Working Set** – Sets working set for programs
- **Page Fault Frequency** – Based on rate for each program

Least Recently Used

- Assume the frameAddress field contains the simulated time of last reference
- To find the least recently used page

```
for all programs /* if global LRU */
   for all pages
      if (frameAddress[page] < oldest)
         oldest = frameAddress[page]
         save oldest page and program numbers
```

- You can then replace the oldest page

How can the program put the simulated time of last reference in the frameAddress variable?

- In small groups, determine how the program might set the frameAddress variable
- The PageFault method is called when there is a page fault

Saving the Time

- To set the frameAddress field to the simulated time of last reference, the PageFault method should do the following for every call

```
for all pages
  if (isRef(page))
    frameAddress[page] = simtime;
  setRef(page,false);
```

Least Frequently Used

- You can change LRU to LFU by incrementing a counter for each page instead of saving the simulated time
- LFU replaces the page that has been used the least

Page Fault Frequency Algorithm

- If a program is generating more page faults than some limit, give the program more pages in RAM.
- If the OS needs to take pages from a program, remove them from a program that is generating less than some minimum number of page faults.

Working Set Strategy

- For each program, the OS keeps resident all pages that have been accessed in the last Δ time units.
- Increasing the parameter Δ increases the number of pages.
- Usually implemented so that at a page fault the system keeps those pages that have been accesses in Δ time units.
- The page table reference bits can be used to determine which pages have been accessed.

Clock Algorithm

- Enhancement to FIFO
- Considers if a page has been referenced and if it has been modified.
- Unmodified pages are replaced before modified pages to avoid having to write the old page back to disk.
- Pages are kept in a circular queue with a pointer to the page after last replaced page.

Clock Algorithm

- Scan 1 – Beginning with the pointer, the pages are scanned for one that is **!referenced and !modified**. The first page found is replaced.
- Scan 2 – Starting from the pointer, look for a page that is **!referenced and modified**. As the pages are checked, clear the referenced bit.

Clock Algorithm

- Scan 3 - Starting from the pointer, the pages are scanned for one that is **referenced and !modified**.
- Scan 4 - Starting from the pointer, look for a page that is **referenced and modified**.