CS433 Programming Assignment 5

Analysis of Page Replacement Algorithms

For this assignment, you are to write a program to analyze different memory page replacement algorithms, assuming a single running process and a fixed size physical memory.

Overview

Your simulator should allow the page size and physical memory size (# of address bits) as command-line arguments. The simulation should support at least three page-replacement policies: Random, FIFO, and LRU. The page size must be a number of power of 2 between 256 (2^8) and 8192 (2^13) bytes inclusively. The physical memory size is also a power of 2 between 4 MB (2^22) and 64 MB (2^26).

Assume that when the simulation starts, the physical memory is empty, i.e. all page table entries are invalid. As they are brought in, the simulator should make sure that the total number of pages in memory should never exceed the physical memory size.

The simulator should read a sequence of logical memory references from a text file. This file contains a list of virtual (logical) memory byte addresses. In this simulation, the maximum virtual memory address is 128 MB (2^{27}bytes). Remember

$$page\ number = \frac{logical\ memory\ address}{Page\ size}$$

, while page offset is irrelevant for this simulation. For example, given logical address 2000 and page size 1024,

$$\frac{2000}{1024} = \frac{2000}{2^{10}} = 1\ R\ 976,$$

it has page number = 1 and page offset = 976. Since page sizes are always a power of 2, the page number can be quickly calculated using the shift operator.

Your program should keep track of pages in the memory and free frames. Therefore, you need to maintain a page-table data structure, which can be easily implemented as an array of page-table entries. The number of pages is dependent on the page size. In the page-table entry data structure, you need additional fields, e.g. valid and dirty bits, besides the mapped frame number of the page. For each memory reference, find out its page number and whether this page is in the main memory according to the page table. If this page is not in the main memory, a page fault is generated and this page is loaded into the main memory and the page table is updated. However, if the main memory is full, i.e. no more free frame, a victim page must be selected and evicted according to a page replacement algorithm. Your simulation is to compare different page replacement algorithms. Notice you will need to keep track of additional information such as last page access time for the LRU (least recently used) algorithm.

Required Output

- Your program should always first print your names and a short description when it starts. You must submit your source code and Makefile so that we can compile your program with the make command.
- In the first test, your program should read and run the simulation for a small list of logical addresses ("small_refs.txt"). Assuming an initial empty physical memory, for each logical address in the list, print out its logical page #, physical frame #, and whether it caused a page fault or not.
- In the second test, assuming an initial empty physical memory, your program should read and run the simulation for a large list of logical addresses ("large_refs.txt"), and collect and print the following statistics for different algorithms:
  - The total number of memory references.
  - The total number of page faults.
  - The total number of page replacements.
  - The total time it took the simulator to produce the results.
- In addition to a program printout, your report should present and analyze the results of different replacement algorithms. You should thoroughly test your program with several different page sizes (256 - 8192 bytes) and physical memory sizes (e.g. 4 MB, 16 MB, 32 MB, 64 MB), analyze the data and summarize results and conclusions in the report.

In the report, you should use tables or graphs to compare the number of page faults and run time of simulated page replacement algorithms.

Extra Credits

(10 points) The group that implements the page replacement algorithms correctly and has the fastest time for the LRU replacement algorithm will get 10 points extra credits. We will use page

size 4096 bytes and physical memory size 32 MB to compare the performance of the submitted programs. Please state the run time for this configuration in your report.