Gebze Technical University
Department of Computer Engineering
CSE 312
Operating Systems Summer 2021

Homework # 1
Due Date: Aug 19th 2024

Virtual Memory Systems and Paging

In this project, you will design and implement a simulated and simplified virtual memory management system and a number of page replacement algorithms in C/C++.

Since this is a simulation, we will use an integer C array for your physical memory as in Fig 3-9 in your textbook. There will be two C threads that will act as two separate processes. Each process will have its linear virtual address space but in fact they are sharing the same C array. As expected, your virtual memory may be much larger as shown in the same figure. You will keep your data that does not fit in your physical memory on a disk file.

Your simplified system will do simple integer array sorting with merge sort and a search algorithm, binary search. The main steps of your program will be as follows

1. Each thread will fill its own entire virtual memory with random integers using C/C++ function **`rand`**. Before you start filling, call **`srand(1000)`** for thread 1 and call **`srand(2000)`** for thread 2 to use the same random numbers for all runs of your program.

2. Each thread will sort its array. When done sorting, these threads will make search operations on the sorted array. Search 5 numbers 2 of which are not in the array.

The above steps will access their virtual integer memory using only the following C/C++ functions

```
void set(unsigned int threadNum, unsigned int index, int value);

int get(unsigned int threadNum, unsigned int index);
```

where **`index`** is the virtual address of the integer, **`value`** is the value of the integer to put at that index, **`threadNum`** is number of the caller thread (1 or 2) . The above functions will be implemented by you to get/set the desired data. Note that each thread accesses a different part of the shared C array.

If the data is available in the physical memory, it is handled immediately. If the data is not available, then the page replacement algorithm has to be executed. The page replacement statistics can be easily calculated in these functions. We are going to use a global page replacement policy.

Part 1

Design a page table structure that makes it possible to implement Clock (CL) and Least-Recently-Used (LRU) algorithms. Note that since we are simulating the hardware, we can add any features as we like to the page table including updating the table at every array reference for LRU.

In your project report, draw figures and explain each field of the page table entries like Fig 3-11. Your report should include references to your implementation code (line numbers, C files, etc.) in your report.

Part 2

Write a C/C++ program that follows above steps. Your program will be named **sortArrays** and it will have the following command line structure

sortArrays frameSize numPhysical numVirtual pageReplacement pageTablePrintInt diskFileName.dat

frameSize is the size of the page frames, **numPhysical** is the total number of page frames in the physical memory, **numVirtual** is the total number of frames in the virtual address space, **pageReplacement** is the page replacement algorithm (CL, LRU), **pageTablePrintInt** is the interval of memory accesses after which the page table is printed on screen, **diskFileName.dat** is the name of the file that keeps the out of memory frames. For example,

sortArrays 8 5 10 LRU 10000 diskFileName.dat

defines a frame size of 2^8 (256) integers, 2^5 (32) physical frames, 2^7 (128) virtual frames for each thread, uses LRU as page replacement algorithm, inverted page table, and **diskFileName.dat** as the disk file name. In other words, this system has a physical memory that can hold 256*32 integers and has a virtual memory that can hold 256*128 integers. This command prints the page table on the screen at every 10000 memory accesses.

At the end of the program run, your statistics will include the following for each thread of the program

- Number of reads
- Number of writes
- Number of page misses
- Number of page replacements
- Number of disk page writes
- Number of disk page reads
- Number of physical frames in memory

In this part, you will implement all necessary functions including get/set, page replacement, and all program phases. Note that you will use the virtual memory only for array storage. For other temporary sorting data (such as indexes), you will use C variables.

Write your report for this part that discusses how you implemented the get/set methods, your page replacement algorithms, and allocation policy issues. Do not forget to include a discussion about your backing store (Section 3.6.5 of the textbook)

Notes

1. Always be careful about the errors, such as inconsistent memory configurations
2. Run experiments that tests end cases such as same size virtual and physical memory
3. Do not use any code from any other source even a single line! If you use ChatGPT or similar sources, just indicate where you used them in your code and report.

General Homework Guidelines

1. No cheating, No copying, No peaking to other people homework
2. Follow the instructions very carefully.
3. Send required files only. Do not share your whole file system with us.
4. If you fail to implement one of the requirements, leave it be. Do not send an empty file
5. Respect the file names! Our HW grading is case-sensitive.
6. Failing to comply any of the warnings above will result in getting a **0** for your current homework.

Homework Instructions

1. Download and Install Vmware Player from Official site.
2. Download and install our virtual machine the teams page link