CSE 312 Operating Systems Part 1 Report

Gökçe Nur Erer - 171044079

Page Table Entry Structure

In this project, page table entries contain these variables to implement NRU, FIFO, Second-Chance, LRU, WSClock algorithms:

1. Virtual Frame Number: The virtual frame number corresponding to the page table entry.
2. Referenced Bit: The variable which denotes if the page is referenced or not.
3. Modified Bit: The variable which denotes if the page is modified or not.
4. Access Time: The variable which holds the last accessed time value. This variable is used in LRU algorithm.
5. Second Chance Bit: The variable which denotes if the page will be given a second chance at replacement or not.

Page table entry structure is implemented in "PageTableEntry.h/PageTableEntry.cpp" files.

Page Table Structure

In this project, page table is defined with a class which has the following items:

1. Entries: This is an array of page table entries which has the size of physical page number.
2. isEmpty: This variable denotes if the page table is empty or not.
3. isFull: This variable denotes if the page table is full or not. This is used to determine if page replacement is necessary or not.
4. entryToReplace: This variable holds the index of the page table entry which will be replaced. This value is determined by the page replacement algorithms.
5. size : This variable is the total size of the page table.

6. currentSize: This variable denotes the entry count in the page table.
7. oldestPageNum: This variable denotes the oldest page number.
8. SCPointer: This variable is used in Second-Chance algorithm as the pointer.

Page table structure is implemented in "PageTable.h/PageTable.cpp" files.

Implementation of the Page Replacement Algorithms

Not Recently Used (NRU)

In this algorithm, the referenced variable of the page table entries in the page table are checked. The first entry encountered whose referenced variable is false, becomes the page to be replaced. To make this work a time counter in MemoryManager class used and gets incremented when get/set functions are used. And at the time values which are multiples of 5, the referenced variables of all entries get setted to false.

FIFO

In this algorithm, the oldestPageNum variable in the page table class is used. When doing the page replacement, the oldest page number which is defined with that variable becomes the page to be replaced. And then the oldest page number is updated to be the next oldest page.

Second Chance (SC)

In this algorithm, the SCPointer variable in the page table and the second chance variable in page table entries are used. First when a page is referenced the second chance variable of the entry becomes true. When a page replacement is needed, with the help of the SCPointer the page table is scanned and the first entry whose second chance variable is false is searched for. While doing the scanning if the encountered entry's second chance bit is 1 it is setted as 0 and the pointer is moved to the next location.

Least Recently Used (LRU)

In this algorithm, the accessTime variable in page table entry class is used. When an entry is added to the page table or when an entry is referenced which was already in the page table the current time value which is held in the MemoryManager class is setted to accessTime value of the entry. So, this algorithm scans and finds the lowest accessTime value and sets that to be the number of the page which will be replaced.

Get Function

This function is implemented in MemoryManager.h/MemoryManager.cpp files.

It first increments the timer value then checks if the timer value is a multiple of 5. Depending on the result it decides if the referenced variables of the page table entries to be resetted or not(for the NRU algorithm).

Then the function calculates which virtual page number the index belongs to and checks if the requested virtual page is in the physical memory or not by using a function called existInPhysicalMemory, which checks the page table entries' virtual page numbers.

If the value exists in the physical memory, referenced, secondChance and accessTime variables of the existing entry gets updated and the requested value is returned from physical memory. (which is an integer array)

If the value doesn't exist in the physical memory, the disk file is read and requested frame is held in an integer array. If the page table is full and a replacement has to be done, depending on the command line parameters the requested page replacement algorithm is run.

The page replacement algorithms will change the entryToReplace parameter in page table. Using that value it is checked if the entry which will be replaced is modified or not. If it is modified it is written back to disk before the replacement happens.
Then the frame read from the disk file is written into physical memory, and then the requested value is returned from physical memory.

Set Function

This function is implemented in MemoryManager.h/MemoryManager.cpp files.

It first increments the timer value then checks if the timer value is a multiple of 5. Depending on the result it decides if the referenced variables of the page table entries to be resetted or not(for the NRU algorithm).

Then the function calculates which virtual page number the index belongs to and checks if the requested virtual page is in the physical memory or not by using a function called existInPhysicalMemory, which checks the page table entries' virtual page numbers.

If the value exists in the physical memory, referenced, secondChance and accessTime variables of the existing entry gets updated and the requested element is updated with the given value in the physical memory. (which is an integer

array)

If the value doesn't exist in the physical memory, the disk file is read and requested frame is held in an integer array. If the page table is full and a replacement has to be done, depending on the command line parameters the requested page replacement algorithm is run.

The page replacement algorithms will change the entryToReplace parameter in page table. Using that value it is checked if the entry which will be replaced is modified or not. If it is modified it is written back to disk before the replacement happens.
Then the frame read from the disk file is written into physical memory, and then the requested element is updated with the given value in the physical memory.

The changes are only written back to disk when a page replacement occurs. Otherwise, the changed values just stay in the physical memory.