Virtual Memory Management System

G.Sai Ram pavan AM.EN.U4AIE20125
Department of Compter Science(Artificial Intelligence)
Amrita Viswa Vidyapeetam
kollam,kerala, India
amenu4aie20125@am.students.amrita.edu

JITHIN JOHN AM.EN.U4AIE20135
Department of Compter Science(Artificial Intelligence)
Amrita Viswa Vidyapeetam
kollam,kerala, India
amenu4aie20135@am.students.amrita.edu

K.Akash Varma AM.EN.U4AIE20141
Department of Compter Science(Artificial Intelligence)
Amrita Viswa Vidyapeetam)
kollam,kerala, India
amenu4aie20141@am.students.amrita.edu

N.Moneesh AM.EN.U4AIE20150
Department of Compter Science(Artificial Intelligence)
Amrita Viswa Vidyapeetam
kollam,kerala, India
amenu4aie20150@am.students.amrita.edu

***Abstract*—This study provides an overview of virtual storage, including its construction, functioning, and applications. Virtual memory is used in today's programs, as well as its user-friendliness is powerful and dynamic. The idea of demand pagination mechanism underpins the development of virtual memory in its current form. For the application's performance, the produced program needs a memory unit. Focus on ensuring that there is adequate memory for each process when we wish to operate an extra technique at the same time as the work equipment. This study contains detailed information regarding digital memory, its applications, when we need it, and how it is employed. All of the packets are saved in the inexhaustible memory.Whenever a program is created, it is first stored into essential memory, which is RAM (random access memory), which is expensive and why computers limit the amount of RAM available. Virtual memory is now employed in all current functions since it allows us to extend the most important memory without increasing the extremely expensive RAM measurement. Secondary memory is used to supplement basic memory. Virtual memory has two advantages. With the help of secondary memory, it first extends the physical remembrance. Moreover, because each virtual plating is translated at a body address, it provides memory protection.**

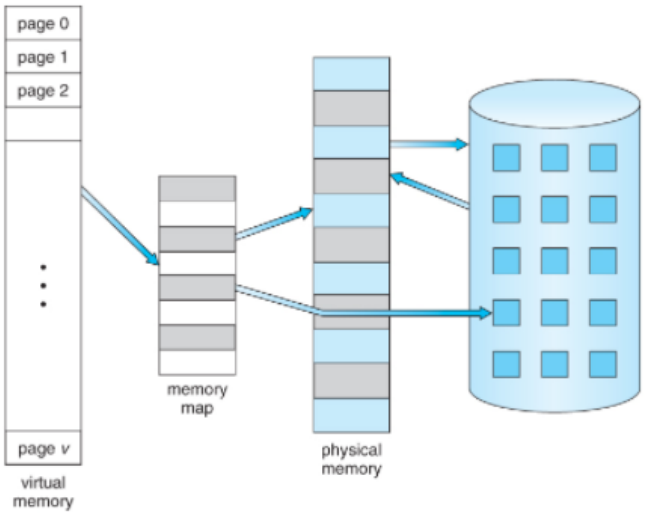
***Index Terms*—Virtual memory, virtual memory implementation, demand paging, swaps in and swap out, demand page,RAM.**

I. Introduction

VirtualMemory is a type of storage that gives the user the impression of having a large main memory. This is accomplished by reclassifying a portion of secondary memory as primary memory. This approach allows the user to load programs that are larger than accessible main memory by creating the appearance that more memory is available. Rather than loading a single large process into main memory, the Os loads the various elements of several processes into main memory. The degree of multi programming will be enhanced as a result, the CPU use will be increased as well.

A. *Working*

Virtual memory is becoming increasingly used in today's world. When certain pages need to be loaded into the main



memory for execution and there isn't enough memory, the OS looks for RAM areas that haven't been used in a while or aren't referenced and copies them into the secondary memory to free up space in the main memory for new pages. Because this technique is carried out automatically, the computer seems to have an endless amount of RAM. Demand paging can be used to implement virtual memory. Demand paging: When a process is switched in, the pager predicts which pages will be consumed before it is swapped out. The pager loads only the pages that are needed, rather than the complete procedure, into memory. So , it prevents reading data into memory pages that can never be utilised, shortening swap times and lowering physical memory needs.

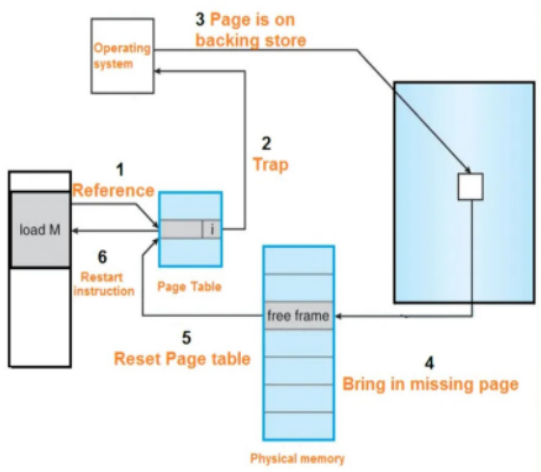
Below-mentioned steps are parts of this process:

- An interrupt will be created whenever the Central Processing Unit tries to access to the page which is currently absent in the main memory, representing a memory access fault.
- The interrupting process is placed in a blocking state

by the os. The Operating System must first load the necessary page into memory before the execution may begin.

- The os will search for the appropriate page in the logical address space.
- Every needed page would be transferred from its logical address to its physicaladdress. Page replacement algorithms are being used to determine whether a page should be swapped into the physical address space.
- This will be reflected in the page table.
- The signal will be delivered to the Central Processing Unit, which could permit program to run, the process to return to its ready state.

As a result, whenever a page fault happens, the operating system follows these steps and loads the appropriate page into memory.



B. Terminologies in demand paging

The valid-invalid bit is used to distinguish amongst pages in the main memory and pages on the disc.

This page is termed as ”valid” when the page is presented in memory, ”invalid” means when the page is not presented in the memory. When a process attempts to retrieve a page that isn't in main memory,then a pagefault will occur.

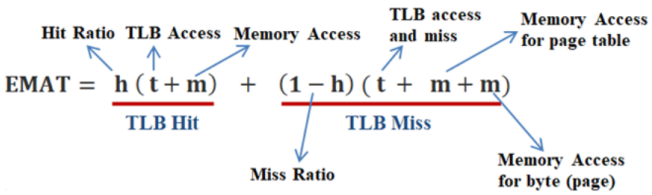
Page Hit - When the CPU attempts to obtain a page from main memory and that page already exists in main memory (RAM).

Page Miss - When a required page does not exist in the RAM

Page Fault Rate - The rate at which threads locate a page fault in memory. The Page Fault Rate is expressed in seconds.

Page Fault Time - The time it takes to collect a page from secondary memory plus the time it takes to recover the page from RAM after it has been loaded.

Performance of Demand Paging - It is evaluated by effective memory access time



C. PAGE REPLACEMENT ALGORITHMS

The Os employs page replacement algorithms to identify what all pages should be swapped in , swapped out in to main memory , whenever a page of main memory has to be assigned. When a page fault occurs paging occurs and a free page cannot be utilized for allocation purposes owing to a lack of available pages or a no of free pages that is fewer than the number of necessary pages. And, in general, FIFO and LRU are the most commonly employed algorithms.

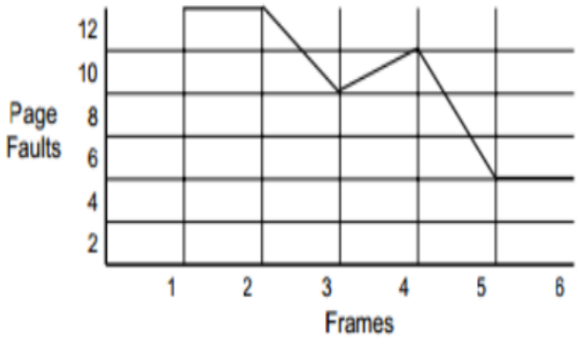
1) ***FIFO****:* Fifo stands for First In First Out.This is the most basic technique of replacing a page.The OS maintains a queue in this method, with the oldest pages at the front and the newest pages at the rear, that keeps track of every page in memory. Whenever a page has to be replaced, the FIFO algorithm unloads the page at the front of the queue, which has been in memory the longest. Consider the following scenario: A page link string with a frame size of 4 and a size of 12 and the string is 1, 2, 3, 4, 5, 1, 3, 1, 6, 3, 2, 3.

1	2	3	4	5	1	3	1	6	3	2	3
1	1	1	1	5	5	5	5	5	5	2	2
	2	2	2	2	1	1	1	1	1	1	1
		3	3	3	3	3	3	6	6	6	6
			4	4	4	4	4	4	3	3	3
M	M	M	M	M	M	H	H	M	M	M	H

M = Miss
H = Hit

The total no of page faults is nine. All four spaces are initially vacant, therefore when 1, 2, 3, and 4 arrive, all were assigned to the available ones in the order of their arrival. This is a page fault because the numbers 1, 2, 3, and 4 are not in memory. This is a page fault because the numbers 1, 2, 3, and 4 are not in memory. Because 5 is not in memory when it arrives, a page fault occurs, and it replaces the oldest page in memory, resulting in 1 being replaced by 5. Because 1 is not in memory when it arrives, a page fault occurs, and it replaces the oldest page in memory, which implies 2 is replaced with 1. When 3,1 arrives, it is already in memory, therefore Page Hit happens, and no page replacement is necessary. When the number 6 arrives, it is not found in memory, therefore a page fault occurs, and it changes the oldest page in memory, resulting in the number 3 being replaced by 6. When the number 3 arrives, it is not found in memory, thus a page fault happens, and it changes the oldest page in memory, resulting in the

number 4 being replaced by the number 3. When the number 2 arrives, it is not found in memory, therefore a page fault occurs, and it changes the oldest page in memory, resulting in the number 5 being replaced by 2. When the number 3 appears, it is already in the memory, therefore Page Hit happens, and no page replacement occurs. Page Fault Ratio = 9/12 The issue with FIFO is the belady anomaly. The belady anomaly occurs when we create a greater number of frames than the number of unsuccessful application pages despite higher performance.



2) Least Recently Used (LRU): The Least Recently Used page replacement algorithm maintains count of number of times a page has been visited recently. It is premised on assumption that pages that have widely utilised in the past would continue to be utilised in the future. When a page is replaced in LRU, the page that has not been utilised for the longest time is replaced.

1	2	3	4	5	1	3	1	6	3	2	3
1	1	1	1	5	5	5	5	5	5	2	2
	2	2	2	2	1	1	1	1	1	1	1
		3	3	3	3	3	3	3	3	3	3
			4	4	4	4	4	6	6	6	6
M	M	M	M	M	M	H	H	M	H	M	H

M = Miss
H = Hit

The total no of page faults is eight. All four spaces are initially vacant, therefore when 1, 2, 3, and 4 arrive, they are assigned to the empty spots in the order of their arrival. This is a page fault because the numbers 1, 2, 3, and 4 are not in memory. Because 5 is not in memory when it arrives, a page fault occurs, and it replaces 1 as the least recently utilised page. When 1 arrives, it is not in memory, therefore a page fault occurs, and it takes the place of 2. When 3,1 arrives, it is already in the memory, means Page Hit, therefore there is no need to update it. When 6 arrives, it is not found in memory, causing a page fault, and it takes the place of 4. When 3 arrives, it is already in the memory, means Page Hit, therefore there is no need to update it. When 2 arrives, it is not found in memory, causing a page fault, and it takes the place of 5. When 3 arrives, it is already in the memory, means Page Hit, therefore there is no need to update it. The ratio of page faults is 8/12.

D. Comparison between virtual memory, physical memory

PHYSICAL MEMORY	VIRTUAL MEMORY
Actual RAM and a form of computer data storage that stores currently executing programs	A memory management technique that creates an illusion to users of a larger physical memory
An actual memory	A physical memory
Faster	Slower
Ues the swapping technique	Uses paging
Limited to the size of the RAM chip	Limited by the size of the hard disk
Can directly access the CPU	Cannot directly access the CPU

II. LITERATURE REVIEW

RESEARCH PAPER ON VIRTUAL MEMORY by Sumit Sehgal which was published on February 3, 2003, in this paper they have discussed virtual memory history, concepts of virtual memory, how virtual memory is implemented, demand paging, demand paging process, page replacement algorithms like FIFO, LRU , virtual memory implementation in Windows 10 , how it works in windows and different pros, cons of virtual memory.

Virtual Memory: Structure and Operation by ALI SAADOON AHMEED ,which was published on december 15, 2018, in this paper they have discussed on structure and implementation of Virtual memory ,deployment of virtual memory, demand paged virtual memory, page replacement algorithms, advantages, disadvantages of Virtual Memory, comparison between virtual memory, physical memory.

Virtual and Cache Memory: Implications for Enhanced Performance of the Computer System by Ugah John Otozi, Ezeanyeji Peter C, Mbaocha Nnamdi Raymond which was published on October 2018, in this paper they have compared virtual, cache memories in aspect of implementation,

concepts,history, principles of operations, importance of them, their uses in computer for functioning.

Virtual memory - Operating system by Qasim Mohammed Hussein which was published on May 2015, in this paper they have discussed on concepts of virtual memory, implementations of virtual memory, demand paging process in detail,Advantages and disadvantages of demand paging , terminologies of demand paging, handling page faults, effective access time, page replacement algorithms(comparison between them), summary of virtual memory.

III. Methods

We are implementing logical address to physical address space conversion which is a part of virtual memory implementation , 2 page replacement algorithms like fifo, lru which are the part of demand paging.

A. *Conversion of logical Addresses to Physical Addresses*

First of all, we declare all the sizes of the pages table and TLB as 256 and 16 respectively, so that the page table can hold the page numbers from indexes 0 to 255 and TLB can hold the page numbers and frame numbers from indexes 0 to 15.

Then, using command-line arguments, we pass in the program name and the file names(paths) where the logical addresses are stored. Then we just check whether we got at least one argument for the file names other than the program name, i.e we check if the argc is less than 2 or not. If it is less than two, the program will stop the execution immediately. If it has 2 or more than 2 it opens the file and reads the content, and at that time if it finds out that there is no content inside the file, it will throw an error for file opening and stop the execution. Then we initialize variables to store the length, get the size of reading values, page numbers, offset, page table, total hits, fault, and pages. We also initialize a variable to maintain the queue position for the purpose of replacing TLB using FIFO. We then initialize the TLB and page table using the memset() function of C which gets the initial pointer as the starting address of TLB, and initializes the whole TLB with -1 and the same in the case of the page table. We start to read the address file which contains all the logical addresses using getline() and gets the page number and the offset from the logical addresses provided in the text file, we convert each value to an integer using atoi() function of the C standard library and mask out the rightmost 16 bits of each of the logical addresses and we obtain the first 8 bits as the page number and the next 8 bits as the page offset.

Then we check each of the page numbers first in TLB where we loop through the TLB comparing the values and we increase the total hits if we get a TLB hit and assign it to the frame and prints "TLB HIT" or if it was not found in the TLB we move on to check in the page table where we do the same looping to find if the page number exists there or not and if it exists, we increase the fault variable and after that, we just replace the TLB using FIFO. Then we print the logical addresses and move on to the calculation of the physical address where we multiply the frame with the page size and add the offset to it. We print the Physical address for every line corresponding to the address file inputted and at the ending, we display the Hit Rate and the Miss Rate of both the TLB and the Page Table by dividing the total hits or the total misses by the pages and multiplying with 100 to express it in percentage.

B. *Page Replacement Algorithms*

As mentioned above there are 2 commonly used page replacement algorithms. We are writing the code in C language The time complexity of this algorithm is O(n) because we are iterating through the array in search of ith element. If we consider it as worst case then it's position is n. So, the time complexity is O(n). The space complexity of the algorithm is O(Frames+size of Pagetable) First we'll import all required libraries. Then we'll now construct a new user defined data type using struct which has attributes corresponding to frame number and page is valid or not which are int and boolean respectively. Then we're writing a new function called to check whether the asked page is the in the page table or not. So this method will return the boolean value whether the frame is there or not. From the above method we'll get to know that the page is there or not in the page table. So, if the page is there then it is page hit condition. Else we're writing another method to replace the page or add the page in the page in the page table. Coming to main function we are now creating all the required variables. First we'll scan the total number of pages we've and then we'll scan the required order of pages and then we'll scan the no of frames in the page table.

Then we'll iterate through the given order and then we'll check whether the element is there in the page table or not. If there then it is page hit, else then we'll check whether the page table have any extra space or not. If there is no space then we'll Do the necessary algorithms. FIFO or LRU

*1) **FIFO**:* As we know the FIFO is done by replacing a frame which came at first among the frames which are there in the page table. So from above variables we are also counting the current variable which maps the first came frame. So, as we're tracking that when we need to replace that, first we'll update the current indexed element with new frame and change the current element to the next first added element.

*2) **LRU**:* Coming to this LRU we'll have another variable added in the Struct which is access time which is due to keep track of the access time of each frame in the page table. So, now we need to replace the appropriate page we'll calculate the access time of each page and we'll check which page has less access time then we'll replace that page with the current required page.

After doing the above algorithms we'll then print the page faults the page table page hit ratio and page fault ratio.

IV. Results

A. *Physical to Logical Address*

The output is each logical address present as integer value in the address.txt and the physical addresses that we obtain after

consulting TLB and pagetable in the program. The output will contain the Hit Rates and Miss Rates of both Page Table and TLB.

```
-❯ ./translator address.txt
VIRTUAL ADDRESS = 1234          PHYSICAL ADDRESS = 210
TLB HIT
VIRTUAL ADDRESS = 1256          PHYSICAL ADDRESS = 232
VIRTUAL ADDRESS = 6720          PHYSICAL ADDRESS = 320
TLB HIT
VIRTUAL ADDRESS = 1267          PHYSICAL ADDRESS = 243
TLB HIT
VIRTUAL ADDRESS = 6789          PHYSICAL ADDRESS = 389

TLB Hit Rate = 60.00 %
TLB Miss Rate = 40.00 %
Page Table Hit Rate = 40.00 %
```

B. FIFO

Let us see the sample output of the following input i.e. *7 0 1 2 0 3 0 4 2 3 0 3 3* and no.of pages are *12* and no.of frames in page table will be *3*

```
7 0 1
2 0 1
2 3 1
2 3 0
4 3 0
4 2 0
4 2 3
0 2 3

Total No. of Page Faults = 10

Page Fault ratio = 0.83

Page Hit Ratio = 0.17
```

C. LRU

Let us see the sample output of the following input i.e. *7 0 1 2 0 3 0 4 2 3 0 3 3* and no.of pages are *12* and no.of frames in page table will be *3*

```
7 -1 -1
7 0 -1
7 0 1
2 0 1
2 0 3
4 0 3
4 0 2
4 3 2
0 3 2

Total No. of Page Faults = 9

Page Fault ratio = 0.75

Page Hit Ratio = 0.25
```

V. Conclusion

A. Advantages of Virtual Memory

It is simple and inexpensive to use virtual memory. That's when the request comes in. It allows for environmentally responsible switching. It enables the facility to operate many goals at the same time. We can conduct mass targeting with significantly less RAM using virtual memory.

B. Disadvantages of Virtual Memory

When applications are executed from virtual memory, they are slower. Data must be translated between virtual and physical memory, which necessitates additional hardware support for address translations, causing a computer to slow down even further. The quantity of secondary storage as well as the computer system's addressing mechanism restrict the capacity of virtual storage. Switching between apps utilising virtual memory may take some time. It reduces the amount of hard disc space accessible.

The concept of virtual memory is appealing since it maximises memory use and provides us with the most cost-effective solution to run many apps simultaneously with significantly less RAM. However, the increased usage of virtual memory eventually degrades our laptop's overall performance since it takes longer to move data into secondary memory, such as RAM. It is useful when used in tiny amounts, but its use in large numbers of huge functions may degrade the overall performance of programs.

References

[1] Cicnavi (2010). "What is Virtual Memory and why do we need it?" Available fromwww.utilizewindow.com.

[2] "Virtual Memory." Studymode, https://www.studymode.com/essays/Virtual-Memory-1843367.html.

[3] John Papiewski (2018). "The Concept of Virtual Memory in Computer Architecture. Available From www.smallbuisness.chron.com/Concept-Virtual-memory.

[4] K. Elissa, "Title of paper if known," unpublished.

[5] Assistance Prof. Dr. Qasim Mohammed Hussein - Researchgate. https://www.researchgate.net/profile/Qasim-Hussein/publication/277405381-Virtual-memory—Operating-system/links/556a799608aefcb861d5f5a7/Virtual-memory-Operating-system.pdf.

[6] (PDF) Virtual and Cache Memory: Implications for Enhanced ... https://www.researchgate.net/publication/328357677-Virtual-and-Cache-Memory-Implications-for-Enhanced-Performance-of-the-Computer-System.