PAGE REPLACEMENT ALGORITHEM

A

Operating System Term work

Submitted by

JAYMIN VALAKI D (MA075), 22MAP0G014

Under the guidance of

Dr. Narayan Joshi

MCA Department,

Faculty of Management & Information Sciences

Dharmsinh Desai University,

Nadiad-387001.

MCA (Master of Computer Applications) Department

Faculty of Management & Information Sciences,

Dharmsinh Desai University,

(ISO 9001:2015 Certified University),

College Road, Nadiad-387 001.

MCA Department
Faculty of Management & Information Sciences,
Dharmsinh Desai University,
College Road, Nadiad-387001

Certificate

This is to certify that **JAYMIN VALAKI D (MA075), 22MAP0G014**
of MCA Semester-II of Dharmsinh Desai University, Nadiad, has worked on the Operating system term work topic **"PROCESS SCHEDULING ALGORTHEM"** under my guidance.

Guided By	**Dr. Narayan Joshi**
Prof./Dr. Narayan Joshi	Professor & Head, MCA Department
MCA Department	D.D.U University
Date:	Date:

NAME: JAYMIN VALAKI D
ROLL NO: MA075

❖ VIRTUAL MEMORY

A computer system has a limited amount of memory. Adding more memory physically is very costly. Therefore, most modern computers use a combination of both hardware and software to allow the computer to address more memory than the amount physically present on the system. This extra memory is actually called Virtual Memory.

Virtual Memory is a storage allocation scheme used by the Memory Management Unit (MMU) to compensate for the shortage of physical memory by transferring data from RAM to disk storage. It addresses secondary memory as though it is a part of the main memory. Virtual Memory makes the memory appear larger than actually present which helps in the execution of programs that are larger than the physical memory.

Virtual Memory can be implemented using two methods:

I. Paging

II. Segmentation

Paging

- Paging is a process of reading data from, and writing data to, the secondary storage. It is a memory management scheme that is used to retrieve processes from the secondary memory in the form of pages and store them in the primary memory. The main objective of paging is to divide each process in the form of pages of fixed size. These pages are stored in the main memory in frames. Pages of a process are only brought from the secondary memory to the main memory when they are needed.
- When an executing process refers to a page, it is first searched in the main memory. If it is not present in the main memory, a page fault occurs.
- In such a case, the OS has to bring the page from the secondary storage into the main memory. This may cause some pages in the main memory to be replaced due to limited storage. A Page Replacement Algorithm is required to decide which page needs to be replaced.

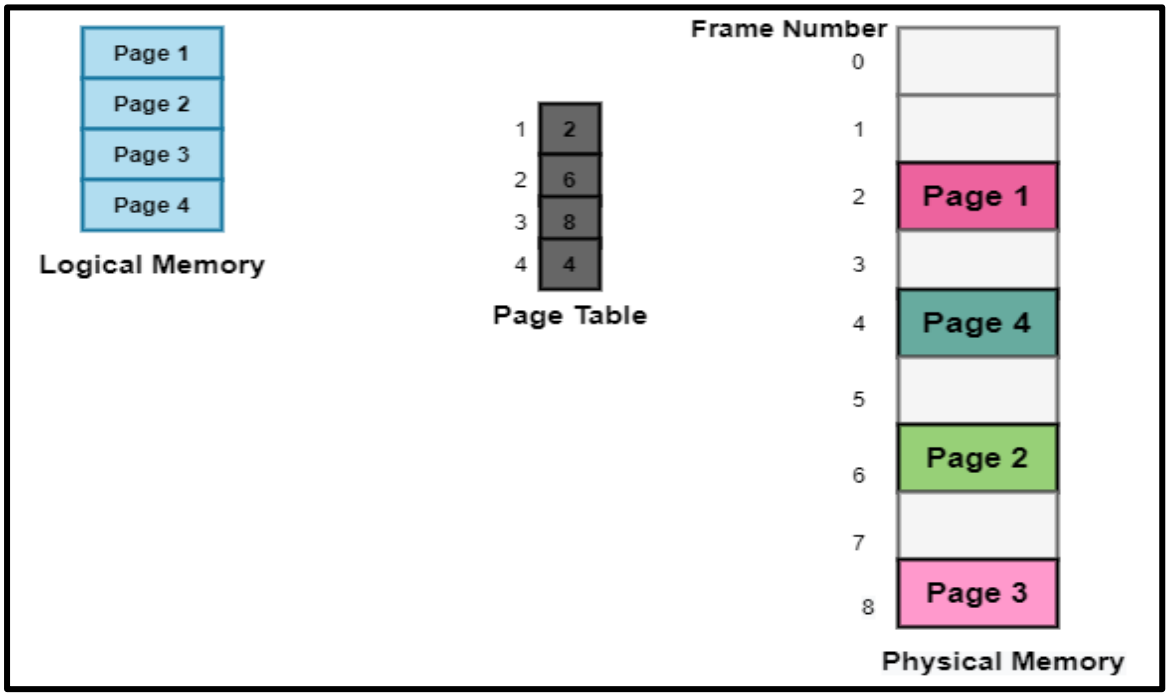
What is Page Replacement algorithm?

Page replacement is a memory management technique used by operating systems to manage the allocation of memory to different processes. It involves the replacement of pages in the physical memory with pages from the virtual memory when there is a shortage of physical memory.

- In a virtual memory system, processes are allocated a limited amount of physical memory while their entire memory space is mapped to the virtual memory.
- When a process needs to access a page that is not in the physical memory, it triggers a page fault, causing the operating system to fetch the required page from the virtual memory and allocate a frame in the physical memory to store it.
- However, if there is not enough free physical memory available, the operating system must choose a page to replace with the required page.
- This process of selecting a page to replace is known as page replacement, and there are several algorithms used for this purpose, such as Least Recently Used (LRU), First-In-First-Out (FIFO), and Clock.
- The algorithm chosen by the operating system determines which page will be replaced, based on criteria such as the time since the page was last accessed or the number of times it has been accessed.
- The selected page is then evicted from the physical memory and replaced with the new page fetched from the virtual memory.
- This process of page replacement continues as long as there is a shortage of physical memory and processes require additional pages to be allocated.
- Proper page replacement algorithms are essential for ensuring efficient memory utilization and preventing memory thrashing, where the operating system spends more time swapping pages in and out of memory than actually executing processes.

❖ PAGE AND FRAME

- A page is a fixed-size block of data that is transferred between the physical memory and virtual memory. It is a unit of memory management used by the operating system to divide a process's memory into small, manageable chunks. Pages are typically 4KB in size, but their size can vary depending on the architecture of the system.
- On the other hand, a frame is a fixed-size block of physical memory that is used to store a page. A frame is typically the same size as a page, but it can be larger or smaller depending on the architecture of the system. The relationship between pages and frames is essential for efficient memory management. The operating system maps pages to frames using a page table, which is a data structure that keeps track of the mapping between the logical addresses used by the process and the physical addresses used by the system.
- When a process requests a page that is not currently in physical memory, the operating system triggers a page fault. The page fault handler retrieves the required page from virtual memory and allocates a free frame in physical memory to store it. The page table is updated to reflect the new mapping between the logical address.



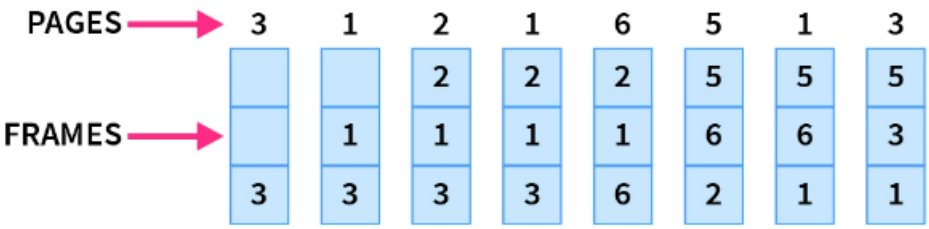
NAME: JAYMIN VALAKI D
ROLL NO: MA075

❖Types Page Replacement Algorithms:

1. First In First Out (FIFO)
2. Optimal Page Replacement
3. Least Recently Used (LRU)
4. Last In first out (LIFO)

1. First In First Out (FIFO)

- This is the simplest page replacement algorithm. In this algorithm, the OS maintains a queue that keeps track of all the pages in memory, with the oldest page at the front and the most recent page at the back.
- When there is a need for page replacement, the FIFO algorithm, swaps out the page at the front of the queue, that is the page which has been in the memory for the longest time.

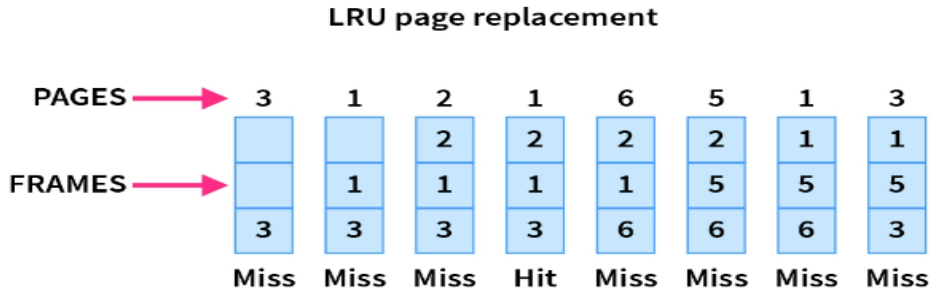


2) Optimal Page Replacement

- Optimal Page Replacement algorithm is the best page replacement algorithm as it gives the least number of page faults. In this algorithm, pages are replaced which would not be used for the longest duration of time in the future, i.e., the pages in the memory which are going to be referred farthest in the future are replaced,
- This algorithm was introduced long back and is difficult to implement because it requires future knowledge of the program behavior. However, it is possible to implement optimal page replacement on the second run by using the page reference information collected on the first run.

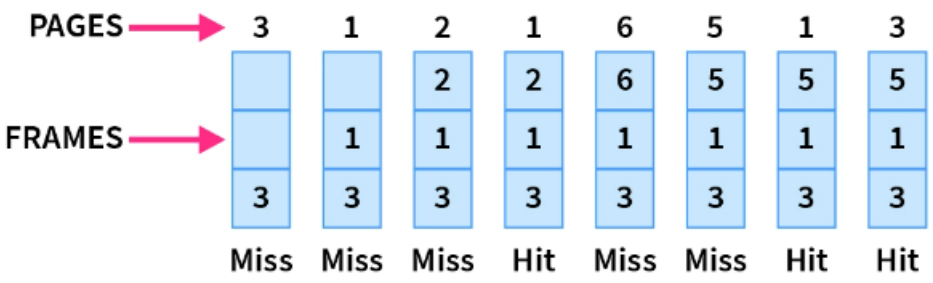
3)Least Recently Used

- Least Recently Used page replacement algorithm keeps track of page usage over a short period of time. It works on the idea that the pages that have been most heavily used in the past are most likely to be used heavily in the future too.
- In LRU, whenever page replacement happens, the page which has not been used for the longest amount of time is replaced.



4)Last in First Out

- This is the Last in First Out algorithm and works on LIFO principles. In this algorithm, the newest page is replaced by the requested page. Usually, this is done through a stack, where we maintain a stack of pages currently in the memory with the newest page being at the top. Whenever a page fault occurs, the page at the top of the stack is replaced.



❖ CODE FOR VISUALIZATION OF A FIFO

➢ Index.html

```
 <!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>PAGE REPLACEMENT ALGORITHM</title>
    <link
href="https://fonts.googleapis.com/css2?family=Concert+One&display=swap"
rel="stylesheet">
    <link rel="stylesheet" href="style.css" type="text/css">

    <script src="https://kit.fontawesome.com/a6d27a9f7c.js"
crossorigin="anonymous"></script>
</head>
<body>

    <h2 class="line anim-typewriter">FIFO - PAGE REPLACEMENT</h2>
    <div class="sub">
        <div class="content">
            <label for="text">Enter No of frames: </label>
            <input type="number" id="cap" min="0"><br><br>

            <label for="text">Enter The page string one by one:</label>
            <input type="text" name="element"id="text" class="form-control">
            <div class="buttons">
                <button type="button" class="btn btn-success" id="add"
onclick="pageFaults()">ADD</button>
            </div>
        <div id="fifo">Memory pages in the memory slot
            <ul id="list"></ul><br>
        </div>
        <div id="page_faults">Total Page faults are:
            <h3 id="faults">0</h3>
        </div>
        <div>
            <div id="page_replace">Total Page replace are:
                <h3 id="replace">0</h3>
        </div>
        <div id="page_hits">Total Page Hits are:
            <h3 id="hits">0</h3>
        </div>
```

NAME: JAYMIN VALAKI D
ROLL NO: MA075

```
        </div>
    </div>

    <br>
    <script src="script.js"></script>
    <!-- jQuery library -->
    <script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></scrip
t>
    <!-- Latest compiled and minified JavaScript -->
    <script
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"
            integrity="sha384-
Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa"
            crossorigin="anonymous"></script>
            <script type="module">
                // Import the functions you need from the SDKs you need
                import { initializeApp } from
"https://www.gstatic.com/firebasejs/9.10.0/firebase-app.js";

 </script>
 </body>
 </html>
```

Style.css

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@100;200;300;500&dis
play=swap');
*{
    font-family: 'Poppins';
    margin:0; padding:0;
    box-sizing: border-box;
    outline: none; border:none;
    text-decoration: none;
    text-transform: capitalize;
    transition:all .2s linear;
}
html {
    margin: 5%;
    background: linear-gradient(#e66465, #9198e5);
}
line anim-typewriter{
    font-family: 'Poppins';
}
.sub{
    background-color: #ffffff;
    padding-top: 20px;
    border-radius: 20px;  }
```

NAME: JAYMIN VALAKI D
ROLL NO: MA075

```
ul li {
    list-style-type: none;
    border: 3px solid black;
    padding: 10px;
    padding-left: 35px;
    width: 7%;
    margin: 1%;
    align-items: center;
    border-radius: 25%;
    font-size: large;
    font-weight: bolder;    }
li:nth-child(1) {
    background-color: #f9ed69;
}
li:nth-child(2) {
    background-color: #f07b3f;
}
li:nth-child(3) {
    background-color: #03f5e1;
}
li:nth-child(4) {
    background-color: #17b978;
}
li:nth-child(5) {
    background-color: #f10751;
}
li:nth-child(6) {
    background-color: #0075f6;
}
li:nth-child(7) {
    background-color: #e4f169;
}
li:nth-child(8) {
    background-color: #a2ef44;
}

#text{
    width: 20%;
    margin-bottom: 1%;
}
input[type=text] {
    height: 30%;
    border: 4px solid #aaa;
    border-radius: 5px;
    outline: none;
    padding: 15px;
    font-size: medium;
    box-sizing: border-box;
    transition: .3s;
}
```

NAME: JAYMIN VALAKI D
ROLL NO: MA075

```
input[type=number] {
    height: 30%;
    border: 4px solid #aaa;
    border-radius: 5px;
    outline: none;
    padding: 15px;
    font-size: medium;
    box-sizing: border-box;
    transition: .3s;
}
input[type=text]:focus {
    border-color: dodgerBlue;
    box-shadow: 0 0 12px 0 dodgerBlue;
}
input[type=number]:focus {
    border-color: dodgerBlue;
    box-shadow: 0 0 12px 0 dodgerBlue;
}
.content {
    margin: 3%;
}
h1 {
    margin-bottom: 5%;
    font-family: 'Anton', sans-serif;
    font-weight: 900;
}
.line {
    position: relative;
    top: 50%;
    width: 34em;
    margin-top: 1%;
    margin-left: 12%;
    border-right: 2px solid rgba(255, 255, 255, 0.75);
    font-size: 4em;
    text-align: center;
    white-space: nowrap;
    overflow: hidden;
    transform: translateY(-50%);
}
.anim-typewriter {
    animation: typewriter 6s steps(40) 1s 1 normal both, blinkTextCursor
1000ms steps(40) infinite normal;
}

@keyframes typewriter {
    from {
        width: 0;
    }to {
        width: 16em;
    } }
```

NAME: JAYMIN VALAKI D
ROLL NO: MA075

```
@keyframes blinkTextCursor {
    from {
        border-right-color: rgba(255, 255, 255, 0.75);
    } to {
        border-right-color: transparent;
    }  }
#fifo {
    font-size: xx-large;
    font-weight: bold;
    font-family: 'Concert One', cursive;
}
#page_faults,#page_hits,#page_replace{
    font-size: xx-large;
    font-weight: bold;
    font-family: 'Concert One', cursive;
}
label {
    font-size: xx-large;
    font-family: 'Concert One', cursive;  }
#add {
    font-weight: 900;
    font-size: larger;
    letter-spacing: 2px;
    padding: 4px;
    transition: 0.8s;
    position: relative;
    width: 10%;
    overflow: hidden;  }
#add {
    margin-right: 3.5%;
}
#add:hover {
    background: rgb(5, 245, 5);
    width: 11%;
}
@media screen and (min-width: 480px) {
    h2 {
        font-size: small;
    }
    li {
        width: 100px;
        padding-left: 20px;
    }
    #add {
        width: 100px;
    } }
@media screen and (min-width: 768px) {
    li {
        width: auto;
    }
```

NAME: JAYMIN VALAKI D
ROLL NO: MA075

➢ Script.js

```
var cap = document.getElementById('cap');
var list = document.getElementById('list');
var text = document.getElementById('text');
var add = document.getElementById('add');
var fifo = document.getElementById('fifo');
var faults = document.getElementById('faults');
var replace = document.getElementById('replace');
var hits = document.getElementById('hits');
var hit = 0;
let page_faults = 0;
let page_replace = 0;
var indexes = [];
let s = new Set();
var i = 0;
class Fifo extends Array {
    constructor(...elem) {
        super(...elem);
    }
    display_elements(s2) {
        for (let i = 0; i < s2.length; i++) {
            var x = document.createElement('li');
            var t = document.createTextNode(s2[i]);
            x.appendChild(t);
            list.appendChild(x);
        }
    }
}
function insertAt(array, index, ...elementsArray) {
    array.splice(index, 0, ...elementsArray);
}
let removeElement = (array, n) => {
    let newArray = [];

    for (let i = 0; i < array.length; i++) {
        if (array[i] !== n) {
            newArray.push(array[i]);
        }
    }
    return newArray;
};
const s1 = new Fifo();
function pageFaults() {
    var pageValue = text.value;
    var capacity = cap.value;
    $('ul').empty();
    text.value = '';
    if (s.size < capacity) {
        if (!s.has(pageValue)) {
```

```
            s.add(pageValue);
            page_faults++;
            indexes.push(pageValue);
        }
        else {
            hit++;
        }
    }
    else {
        if (i == capacity) {
            i = 0;
        }
        if (!s.has(pageValue)) {
            let val = indexes[i];
            indexes = removeElement(indexes, val);
            s.delete(val);
            s.add(pageValue);
            insertAt(indexes, i++, pageValue);
            page_faults++;
        }
        else {
            hit++;
        }
    }
       page_replace = `${page_faults-capacity}`;
    for (let j = indexes.length - 1; j >= 0; j--) {
        s1.display_elements(indexes[j]);
    }
    faults.innerHTML = page_faults;
    hits.innerHTML = hit;
    replace.innerHTML = page_replace;
}
```

NAME: JAYMIN VALAKI D
ROLL NO: MA075

❖ Summary

The page replacement algorithm project focuses on implementing a First-In-First-Out (FIFO) page replacement algorithm. This algorithm is used by operating systems to select pages to evict from memory when there is a shortage of physical memory.

The project involves designing and implementing a program that simulates the behavior of the FIFO page replacement algorithm. The program must read in a series of memory access requests and maintain a list of pages currently in physical memory. When a page fault occurs, the program must select the oldest page in memory for eviction and replace it with the new page.

The program must be tested thoroughly to ensure that it correctly implements the algorithm and efficiently manages memory.

Overall, the page replacement algorithm project is an important exercise in understanding memory management in operating systems and provides practical experience in designing and implementing algorithms. The focus on the FIFO algorithm helps to develop a strong foundation in memory management and lays the groundwork for more complex algorithms in the future.

NAME: JAYMIN VALAKI D
ROLL NO: MA075