GROUP ONE AND THREE

GROUP MEMBERS

AHIMBISIBWE SAM 2019/BSE/019/PS

SEBAYIGA ABRAHAM 2019/BSE/030/PS

MUSANJE ANTHONY LAWRENCE 2019/BSE/014/PS

KUYESO ROGERS 2019/BSE/031/PS

LUMULI TONNY 2019/BSE/011/PS

KASAJJA GAMOIZA EMMANUEL 2019/BSE/009/PS

AGABA EDWIN 2019/BSE/001/PS

SWAPPING

This is mechanism in which a process can be swapped temporarily out of the main memory to

secondary storage and make that memory available for other process. At some later time, the system swaps back the process from secondary storage to main memory. Though performance in usually affected by swapping process but it helps in running multiple and

big process in parallel and that's the reason swapping is also known as a technique for memory

compaction . memory compaction means free space is collected in a large memory chunk to make some space

available for processes. ADVANTAGES

 The process helps the CPU to manage multiple processes within the same main memory.  The method helps to create and use virtual memory.  The method is economical.  It makes CPU perform several tasks simultaneously.

**Paging.**

In computer operating systems, paging is a memory management scheme by which a computer stores and retrieves data from secondary storage for use in main memory. In this scheme, the operating system retrieves data from secondary storage in same-size blocks called pages. Paging is an important part of virtual memory implementations in modern operating systems, using secondary storage to let programs exceed the size of available physical memory.

For simplicity, main memory is called "RAM" (an acronym of "random-access memory") and secondary storage is called "disk" (a shorthand for "hard disk drive"), but the concepts do not depend on whether these terms apply literally to a specific computer system.

**Page Faults.**

When a process tries to reference a page not currently present in RAM, the processor treats this invalid memory reference as a page fault and transfers control from the program to the operating system. The operating system must:

1. Determine the location of the data on disk.

2. Obtain an empty page frame in RAM to use as a container for the data.

3. Load the requested data into the available page frame.

4. Update the page table to refer to the new page frame.

5. Return control to the program, transparently retrying the instruction that caused the page fault.

When all page frames are in use, the operating system must select a page frame to reuse for the page the program now needs. If the evicted page frame was dynamically allocated by a program to hold data, or if a program modified it since it was read into RAM (in other words, if it has become "dirty"), it must be written out to disk before being freed. If a program later references the evicted page, another page fault occurs and the page must be read back into RAM.

The method the operating system uses to select the page frame to reuse, which is its page replacement algorithm, is important to efficiency. The operating system predicts the page frame least likely to be needed soon, often through the least recently used (LRU) algorithm or an algorithm based on the program's working set. To further increase responsiveness, paging systems may predict which pages will be needed soon, preemptively loading them into RAM before a program references them.

**Page replacement techniques.**

**1. Demand paging**

When pure demand paging is used, pages are loaded only when they are referenced. A program from a memory mapped file begins execution with none of its pages in RAM. As the program commits page faults, the operating system copies the needed pages from a file, e.g., memory-mapped file, paging file, or a swap partition containing the page data into RAM.

**2. Anticipatory paging**

This technique, sometimes also called swap prefetch, predicts which pages will be referenced soon, to minimize future page faults. For example, after reading a page to service a page fault, the operating system may also read the next few pages even though they are not yet needed (a prediction using locality of reference). If a program ends, the operating system may delay freeing its pages, in case the user runs the same program again.

**3. Free page queue, stealing, and reclamation**

The free page queue is a list of page frames that are available for assignment. Preventing this queue from being empty minimizes the computing necessary to service a page fault. Some operating systems periodically look for pages that have not been recently referenced and then free the page frame and add it to the free page queue, a process known as "page stealing". Some operating systems[b] support page reclamation; if a program commits a page fault by referencing a page that was stolen, the operating system detects this and restores the page frame without having to read the contents back into RAM.

**4. Pre-cleaning**

The operating system may periodically pre-clean dirty pages: write modified pages back to disk even though they might be further modified. This minimizes the amount of cleaning needed to obtain new page frames at the moment a new program starts or a new data file is opened, and improves responsiveness. (Unix operating systems periodically use sync to pre-clean all dirty pages; Windows operating systems use "modified page writer" threads.)

**Thrashing.**

After completing initialization, most programs operate on a small number of code and data pages compared to the total memory the program requires. The pages most frequently accessed are called the working set.

Working set is a concept in computer science which defines the amount of memory that a process requires in a given time interval.

**Sharing.**

In multi-programming or in a multi-user environment, many users may execute the same program, written so that its code and data are in separate pages. To minimize RAM use, all users share a single copy of the program. Each process's page table is set up so that the pages that address code point to the single shared copy, while the pages that address data point to different physical pages for each process.

**ONTIGUOUS MEMORY ALLOCATION**

* Contiguous memory allocation is a classic memory allocation model that assigns a process consecutive memory blocks( memory blocks having consecutive addresses.

It is one of the oldest memory allocation schemes.

* When the process needs to execute, memory is requested by the process. The size of the process is compared with the amount of contiguous main memory available to execute the process.
* If found,the process is allocated memory to start its execution.

if not, it is added to the queue of waiting processes until sufficient free contiguous memory is available

* Contiguous memory allocation can be implemented by the operating with the help of two registers known as base and limit registers.

When a process is executing in main memory, its base registers contains the starting address of the memory location where the process is executing, while the amount of bytes consumed by the process is stored in the limit register.

* A process does not directly refer to the actual address for the corresponding memory location. Instead, it uses relative address with respect to its base register.
* All addresses referred by a program are referred to as virtual addresses. The CPU generates logical and virtual address, which is converted into actual address with the help of the memory management unit(MMU).
* The base address register is used for address translation by the MMU
* Therefore physical address = Base register address + Logical address/ virtual address
* The address of any memory location referenced by a process is checked to ensure it does not refer to address of neighbouring process. This processing security is handled by the OS.

Advantages

1. It is simple to implement.
2. We will get excellent read performance
3. Supports random access into files

Disadvantages

1. The disk will become fragmented
2. It may be difficult to have a file grow
3. The degree of multi programming is reduced due to processes waiting for free memory

**Memory segmentation**

Memory segmentation is a computer (primary) memory management technique of division of a computer's primary memory into segments or sections.

Different segments may be created for different program modules, or for different classes of memory usage such as code and data segments.

Segmentation was originally invented as a method by which system software could isolate different software processes (tasks) and data they are using. It was intended to increase reliability of the

systems running multiple processes simultaneously.

In a system using segmentation, computer memory addresses consist of a segment id and an offset within the segment

A hardware memory management unit (MMU) is responsible for translating the segment and offset into a physical address, and for performing checks to make sure the translation can be done and Hardware implementation that the reference to that segment and offset is permitted.

Each segment has a length and set of permissions (for example, read, write, execute) associated with it.

Segmentation is one method of implementing memory protection. Segmentation has been implemented in Segmentation without paging.

Associated with each segment is information that indicates where the segment is located in memory— the segment base.

SEGMENTATION WITH PAGING.

1. **sedimentation with paging**:

These are non-contiguous memory allocation techniques.

Paging divides the process into equal sized portions called pages.

Sedimentation divides the process into unequal portions called sediments.Details about each segment are stored in a table called segment table.

1. **.Segmentation** supports the user’s view of the memory.The process is divided int variable size segments and loaded to the logical memory address space.

The address specified by the user contain two quantities I.e. the segment name and the offset (segment length).

Segment number is used as the index in the segment table and the offset value decides the length of the segment.

1. **. Paging** allows a process to be stored in a memory in a non-contiguous manner.The size of the process is measured by the number of pages .

The main memory is divided into small fixed sized blocks of the memory called frames and the size of the blocks is kept the same as the page to have optimum utilization of the main memory and avoid external fragmentation.

***NB***. 1.External fragmentation comes in when the process does not the entire block space(it can be solved by storing the process in a non-contiguous manner).

2.**Non-contiguous memory allocation** the separate memory blocks at a different location in memory space in a non-consecutive manner to a process requiring for memory while **contiguous memory allocation** assigns the consecutive blocks of memory to a process requesting for memory.

**SEGMENTED PAGING.**

Segmented paging is a memory management scheme that implements the combination of segmentation and paging.

In segmented paging ,the process is first divided into segments and then each segment is divided into pages.

These pages are then stored in the frame of main memory.

A page table exists for each segment that keeps track of the frames storing the pages of that segment.

Each page table occupies one frame in the main memory.

**Difference between paging and segmentation**

|  |  |
| --- | --- |
| **paging** | **segmentation** |
| 1. **A page is a fixed block size.** 2. **They lead to internal fragmentation.** 3. **The hard ware decides the page size.** 4. **Paging involves a page table that contains base address of each page.** | **1. A segment is of variable size.**  **2. They lead to external fragmentation.**  **3. Size is specified by the user.**  **4. Segmentation involves a segment table that contains segment number and offset.** |

Advantages and disadvantages of segmented paging

|  |  |
| --- | --- |
| advantages | disadvantages |
| 1.It reduces memory usage.  2.Page table size is limited by the segment size.  3.Segment table has only entry corresponding one actual segment.  4.External fragmentation is no there.  5.It simplifies memory allocation. | 1.May lead to internal fragmentation.  2.The complexity level will be much higher as compared to paging.  3.Page tables need to be contiguously stored in memory |