CIS 452 02 – Assignment 12 – Gabe Baksa

1. List 2 advantages of using a virtual address space as opposed to the physical address space.

Shared System Libraries: Virtual memory allows multiple processes to access system libraries as if they are part of each process's individual memory. This reduces redundancy by loading the libraries only once in physical memory, even though each process accesses them independently.

Shared Memory Regions for Inter-process Communication: Virtual memory enables shared memory regions, which allow processes to communicate and share data efficiently without needing to copy data between processes. This is particularly beneficial for applications that require frequent data exchange, improving overall system performance.

2. What is meant by the term 'lazy swapper'? List some advantages of using this technique for memory management.

A lazy swapper (or lazy pager) loads pages into memory only when they're needed, not the entire process at once.

Advantages:

- Efficient Memory Usage: Only required pages are loaded, allowing more processes to run simultaneously.
- Reduced I/O Operations: Fewer unnecessary I/O operations improve performance and responsiveness by reducing disk access.

3. What is the valid/invalid bit used for in the page table?

The **valid/invalid bit** in a page table indicates whether a page is currently in physical memory:

- **Valid**: The page is in physical memory and can be accessed directly.
- **Invalid**: The page is not in physical memory (it's either on disk or unallocated).

This bit helps the memory manager determine if a memory reference is valid. If the bit is invalid and the address is valid (but not loaded), a page fault is triggered, prompting the system to load the page into memory.

4. What is meant by the term ‘copy-on-write’? How is this approach advantageous to memory management?

Copy-on-Write (COW) is a memory management technique where a page initially shared between two processes (e.g., after a `fork()` call) is only duplicated when one of the processes attempts to modify it. Until a write occurs, both processes can read from the same physical page, conserving memory.

Advantages:

- **Memory Efficiency**: Reduces memory usage by avoiding unnecessary copies of pages that are not modified.
- **Improved Performance**: Reduces the overhead of duplicating pages during process creation, as only modified pages need to be copied later.

5. Given the reference string used in the class lecture and textbook, compare the performance (page faults) of each of the following page replacement algorithms assuming the process has 3 pages available to it:

FIFO: 15 faults

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

2	3	0
2	3	0
3	0	1
0	1	2
0	1	2
0	1	2
1	2	7
2	7	0
7	0	1

LRU: 12 faults

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

1	0	2
1	0	2
1	0	7
1	0	7
1	0	7

Second Chance: 10 faults

7		
7	0	
7	0	1
0	1	2
0 Ⓡ	1	2
1	2	0
2	0	3
2	0 Ⓡ	3
0 Ⓡ	3	4
3	4	0
4	0	2
0	2	3
0 Ⓡ	2	3
0 Ⓡ	2	3 Ⓡ
0 Ⓡ	2 Ⓡ	3 Ⓡ
2 Ⓡ	3 Ⓡ	0
3 Ⓡ	0	2
0	2	3
2	3	1

2 ®	3	1
3	1	2
1	2	0
1 ®	2	0
2	0	1
0	1	7
0 ®	1	7
0 ®	1 ®	7

Optimal Page Replacement: 9 faults

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

2	0	1
7	0	1
7	0	1
7	0	1

Reference String:

7, 0, 1, 2, 0, 3, 0, 4, 2, 3, 0, 3, 2, 1, 2, 0, 1, 7, 0, 1

6. **(optional) If time permits in class, create your own algorithm for page replacement. Describe it below and check it's performance using the above reference string (if able).**

Highest Page Number: 10 faults

My algorithm that I came up with is just replacing the page number that has the highest value. I noticed that the reference string repeated a lot of 0s and 1s so I knew this algorithm would do well.

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

2	0	1
2	0	1
2	0	1
2	0	1
7	0	1
	0	
		1