CS3204-Spring 2001
Dr. Sallie Henry
Programming Assignment #3

Due April 9, 2001 midnight
Demo April 10, 2001
Design is due March 13, 2001 (class)

You are to modify programming assignment #2, to add a simulation of demand paged virtual memory management system. Your program will actually simulate the contents of main and secondary memory, and copy pages between main and secondary memory just as an actual virtual page memory manager would.

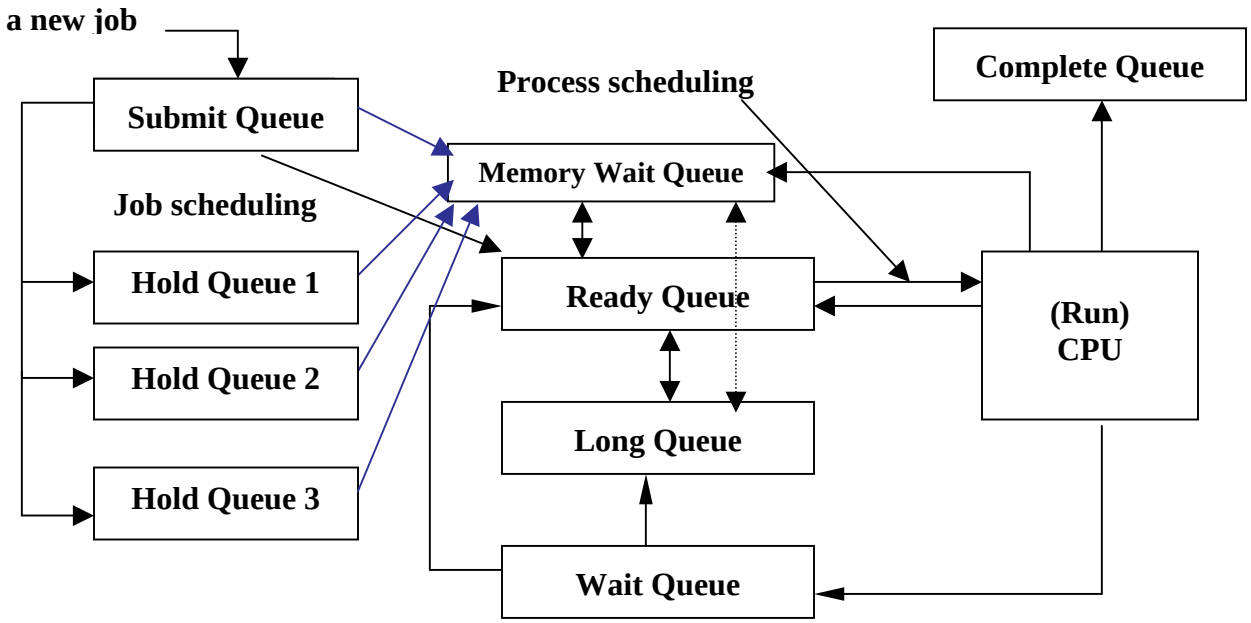
Memory Design

The memory system you are to simulate obeys the following constraints:

- Assume that a page and block are size 4k.
- The secondary storage area used for secondary memory is 300k = 75 blocks.
- The main memory is word addressable. You are to represent each word in the virtual address (VA) space by an integer. (This assumption is made to minimize the memory required by your simulation program itself). That integer is INITIALY the job ID.
- The page replacement algorithm is LRU Approximation, as specified by the lecture note transparencies containing LRU Approximation algorithm. This will require you to keep the reference bits for each block of Main Memory and you must also keep the Replacement Pointer (RP).
- A main memory block and a job page are both size 4k (This implies that relative addresses 0-3 are on the first page of a job, and 4-7 are on the second page, and so on. We will therefore refer to relative address (RA) 10 as VA [P2, O2], for a page 2 / offset 2.)
- The first four blocks (as in the first 16 words), physical address space 0 – 15, of main memory are reserved for the operating system.
- Before a process enters the Ready Queue for the first time, page 0 of the process must be loaded into main memory, this implies that the job must enter the Memory Wait Queue prior to entering the Ready Queue. <u>This is also considered a page fault</u>.
- A job going to and from the Memory Wait Queue is an internal event. When a job on the CPU (the only time a job can reference memory) references a page that is NOT in main memory that is a page fault and an interrupt.
- Page faults occur whenever a page that is **not** in main memory gets accessed.
- When LRU Approximation selects a block from MM to "hold" the incoming page, that block is marked "in transit", or whatever you want to use, so that the same block is not selected while the process is in the MWQ.
- **The disk processor can service only one process at a time.**
- Copying a page to/from MM takes .2 (time).
- When calculating "next internal event time", you will have to add "time the next process comes off of the MWQ" to next completion time, and next end of quantum.

- When a process comes off of the MWQ, it returns to the back of the ready or long queue. (FIFO)
- When a process completes, copy all of its changed pages back to secondary memory, this implies that the process will have to enter the MWQ for (0.2*number of changed pages) until it finishes. Record the finish as the time it comes off of the MWQ. (I know that this doesn't make much sense but it is the only way to 'record' the changed pages).
- When two events occur "at the same time", process the internal event first.
- When two internal events occur at the same time, process them in the following order:
    - Completion
    - End of quantum
    - Memory movement complete

A graphic view of the simulator



Input Data File

The input to your program is a modification of the input to project 1. The modifications are described below.

- The Q and L commands will not be modified.
- The C commands:

  C M=16 L=1.0 D=7 Q=0.3

The M=16 field means that the total number of blocks in Main Memory is 16.

- A job arrives:

  A 10 J=2 M=50 D=5 R=5 P=1 W=3

The M=50 field means that the Main Memory space size that job 2 requires is 50k (which requires the ceiling of (50 / 4) = 13 blocks). **W is the working set for the job. This will be discussed in class.** Load the job into secondary memory (Initialize secondary memory (SM) to the Job ID). In this example, 13 blocks would be used.

- A job references memory:

  R 10 J=2 RA=10

Job 2 references relative address 10. If the page containing the relative address 10 is not main memory, a page fault occurs, the appropriate bits are set in the (PMT) and in main memory (MM), and the page is loaded from SM. (Page faults are described below). If the RA = 1000 (where the RA could be between 1000-1039, this is assumed to be a shared memory location.

- A job modifies the contents of memory:

  M 10 J=3 RA =12 T0 = 18

Job 3 changes the contents of relative address 12 to the integer value 18. If the page containing relative address 12 is not in the main memory, a page fault occurs and the same reference bits are set, etc. as in the reference example. In addition, the change bit of MM is also set.

- Display Command: Add the PMT, FMT, MBT, main memory contents, secondary memory contents, SMBT, and MWQ contents along with the output displayed from project #2.

MWQ is different from HOLDQ used in assignment #1. A job is placed in the MWQ when a reference by the job results in a page fault or when the job first enters the ready queue.

SHARED PAGES – these are addresses as 1000-1039 (in other words 10 shared blocks). Many processes may share the same-shared page, BUT only one copy of that shared page is in memory at a time.

Semantic Errors in Input file

The input file may contain semantic errors, but it will not contain syntax errors. You must at least check for the errors listed below.

- A Job arrives whose memory requirement exceeds the size of the SM.
- An "R" input command refers to an illegal relative address (e.g., not in the range ∅-L-1, where L is (the current memory size-1), accounting for all requests and releases of memory. However, a job may reference its own internal fragmentation.
- An "M" input command refers to an illegal relative address (e.g., not in the range ∅-L-1.)
- An "R" or "M" command that refers to a job not currently on the CPU. Tell the GTA if this occurs.

A command containing an error should generate an informative error message and be ignored. Do not remove the offending job from the system.

Output

- Output from all displayed
- T, W, PFR

Simulating Page Faults

A page fault is simulated by:

- Writing the page to be replaced back to disk (causing the faulting process to wait for time .2), only if the page was modified, and
- Copying the page containing the missing relative address to main memory (causing the faulting process to wait either for time .2 or until the page is in memory due to an earlier reference).

Note that the Job Table, PMT, FMT, MBT, main memory, and secondary memory in your simulation must all be updated.

Additional Specifications

- When a job releases memory: you must copy pages corresponding to the process' released memory to secondary memory, if the page(s) was (were) changed.
- Secondary memory must be reused. Keep a pool of free secondary memory. Initially all secondary memory is in the pool. When a job arrives, allocate secondary memory from the pool. When a job completes, release its secondary memory to the pool. Be careful of what data structure you use to represent free pool, A stack would probably be the easiest.
- Free up the MM used by that job, the SM used, the PMT and FMT, update the MBT and SMBT

PMT-Page Map Table- For each job
FMT-File Map Table- For each job

The FMT and PMT may be combined if you desire.
SM-Secondary Memory/Disk
MM-Main Memory
MBT- Memory Block Table-ID of processes contained in block.
SMBT- Secondary MBT

- Block Size = Page Size.
- Memory wait queue is FIFO
- When a job completes (comes off the MWQ after copying its changed pages),
    - Release
    - Devices
    - Main memory
    - Secondary memory
    - Working set

Check queues in the following order:

- Device wait
- Hold
- Long

What to turn in

This will remain the same as program 1, if there are any changes they will be posted on the web.