COSC1112/1114: Operating Systems Principles

Tutorial 07 (week 08)

1. **Explain the difference between internal and external fragmentation.**

Answer:

- Internal fragmentation is the area in a region or a page that is not used by the job occupying that region or page. This space is unavailable for use by the system until that job is finished and the page or region is released.
- External fragmentation is unused space between allocated regions of memory. Typically external fragmentation results in memory regions that are too small to satisfy a memory request, but if we were to combine all the regions of external fragmentation, we would have enough memory to satisfy a memory request.

2. **Consider a logical address space of 256 pages with a 4-KB page size, mapped onto a physical memory of 64 frames.**
   a) **How many bits are required in the logical address?**
   b) **How many bits are required in the physical address?**

Answer:

a) 12 + 8 = 20 bits.
b) 12 + 6 = 18 bits.

3. **Compare the memory organization schemes of contiguous memory allocation, and pure paging with respect to the following issues:**
   a) **External fragmentation**
   b) **Internal fragmentation**
   c) **Ability to share code across processes**

Answer:

The contiguous memory allocation scheme suffers from external fragmentation as address spaces are allocated contiguously and holes develop as old processes die and new processes are initiated. It also does not allow processes to share code, since a process's virtual memory segment is not broken into non-contiguous fine grained segments.

Pure segmentation also suffers from external fragmentation as a segment of a process is laid out contiguously in physical memory and fragmentation would occur as segments of dead processes are replaced by segments of new processes. Segmentation, however, enables processes to share code; for instance, two different processes could share a code segment but have distinct data segments.

Pure paging does not suffer from external fragmentation, but instead suffers from internal fragmentation. Processes are allocated in page granularity and if a page is not completely utilized, it results in internal fragmentation and a corresponding wastage of space. Paging also enables processes to share code at the granularity of pages.

4. **Assuming a 1-KB page size, what are the page numbers and offsets for the following address references (provided as decimal numbers):**
   a) **3085**
   b) **42095**
   c) **215201**
   d) **650000**
   e) **2000001**

Answer:

a) page = 3; offset = 13
b) page = 41; offset = 111
c) page = 210; offset = 161
d) page = 634; offset = 784
e) page = 1953; offset = 129

5. **What is the purpose of paging the page tables?**

Answer:

In certain situations the page tables could become large enough that by paging the page tables, one could simplify the memory allocation problem (by ensuring that everything is allocated as fixed-size pages as opposed to variable-sized chunks) and also enable the swapping of portions of page table that are not currently used.

6. **Consider a system that uses pure demand paging:**
   a. **When a process first starts execution, how would you characterize the page fault rate?**
   b. **Once the working set for a process is loaded into memory, how would you characterize the page fault rate?**

Answer:

a. Initially quite high as needed pages are not yet loaded into memory.
b. It should be quite low as all necessary pages are loaded into memory.

7. **A certain computer provides its users with a virtual-memory space of 2^{32} bytes. The computer has 2^{18} bytes of physical memory. The virtual memory is implemented by paging, and the page size is 4096 bytes. A user process generates the virtual address 11123456. Explain how the system establishes the corresponding physical location. Distinguish between software and hardware operations.**

Answer:

The virtual address in binary form is 0001 0001 0001 0010 0011 0100 0101 0110. Since the page size is 2^{12}, the page table size is 2^{20}. Therefore the low order 12 bits "0100 0101 0110" are used as the displacement into the page, while the remaining 20 bits "0001 0001 0001 0010 0011" are used as the displacement in the page table.