Operating Systems

211 - Peter Pietzuch - <prp@doc.ic.ac.uk>

Tutorial - Memory Management

1. Describe the difference between swapping and paging in the context of virtual memory management.

Answer: Swapping means moving entire address spaces between the disk and the memory. Paging means moving individual pages, so that part of an address space may be on disk while the other part is in main memory.

2. What is the advantage of a paged virtual memory system with a

    a. small page size?
    b. large page size?

Answer: Smaller page sizes imply less internal fragmentation and therefore more efficient memory use. Larger page sizes imply less overhead for address translation and therefore faster memory access.

3. What is an associative memory? How does it work and how is it implemented?

Answer: An associative memory is one that is accessed by content rather than by address. Often an "entry" in an associative memory will have two parts, a tag and a value. To access the associative memory, one supplies a tag. If that tag matches the tag of an entry in the associative memory, that entry is returned. In a hardware implementation (e.g., a TLB), the tags of all entries are checked in parallel.

4. Describe how a context switch affects the virtual memory system.

    What must the OS do to ensure that memory references made by the newly-running process will be properly translated?

    Answer: The operating system must locate the page table for the process that is to start running. It must set the base register in the MMU so that it points to the page table in memory, and it must set the length register if present. Finally, it must clear any now-invalid cached address translations from the TLB.

5. A system implements a paged virtual address space for each process using a one-level page table. The maximum size of an address space is 16 MB. The page table for the running process includes the following entries:

Page	Frame
0	4
1	8
2	16
3	17
4	9

The page size is 1024 bytes and the maximum physical memory size of the machine is 2 MB.

a. How many bits are required for each page table entry?
b. What is the maximum number of entries in a page table?
c. How many bits are there in a virtual address?

d. To which physical address will the virtual address 1524 translate to?
e. Which virtual address will translate to physical address 10020?

Answer: There can be at most $2^{21}/2^{10}$ frames in main memory, so a page table entry will require 11 bits for a frame number, plus protection and reference bits, etc.

A page table may have up to $2^{24}/2^{10} = 2^{14}$ entries.

A virtual address has 24 bits.

*Virtual address 1524 lies on page 1 which resides in frame 8 with an offset of 1524 modulo 1024, or 500. So, 1524 maps to physical address 8 * 1024 + 500 = 8192 + 500 = 8692.*

*Physical address 10020 is in frame 9, with an offset of 804. Virtual page 4 maps to frame 9, so virtual address 4 * 1024 + 804 = 4900 will map to 10020.*

6. A pure paging system uses a three level page table. Virtual addresses are decomposed into four fields *(a, b, c, d)* with *d* being the offset.

   In terms of these constants, what is the maximum number of pages in a virtual address space?

Answer: $2^{(a+b+c)}$, since there are $2^{(a+b+c+d)}$ total addresses in the address space, and there are 2^{d} on each page.

7. Calculate the access times for a four-level paging system assuming a TLB hit ratio of 80% and 98%. Assume that time for a memory access is 100 ns and for TLB access 20 ns.

   How does this compare to a single-level paging system?

Answer:

If the hit ratio is 80%, the time for translation is:

effective access time = 0.8 x 120 + 0.2 x 520 = 200ns, thus 100% slower than unpaged memory access (compared to 40% in the case of single-level paging)

If the hit ratio is 98%, the time for translation is:

effective access time = 0.98 x 120 + 0.02 x 520 = 128ns, thus 28% slower than unpaged memory access (compared to 22% in the case of single-level paging)

8. Suppose that pages in a virtual address space are referenced in the following order:

   1 2 1 3 2 1 4 3 1 1 2 4 1 5 6 2 1

   There are three empty frames available. Assume that paging decisions are made on demand, i.e., when page faults occur.

   Show the contents of the frames after each memory reference, assuming the LRU replacement

policy is used.

Repeat assuming that the clock policy (second chance) is used.

How many page faults occur in each case?

For LRU:

Frame	Reference																
	1	2	1	3	2	1	4	3	1	1	2	4	1	5	6	2	1
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	2	2
1		2	2	2	2	2	2	3	3	3	3	4	4	4	6	6	6
2				3	3	3	4	4	4	4	2	2	2	5	5	5	1
Fault?	Y	Y	N	Y	N	N	Y	Y	N	N	Y	Y	N	Y	Y	Y	Y

The total number of faults is 11.

For clock:

Frame	Reference																
	1	2	1	3	2	1	4	3	1	1	2	4	1	5	6	2	1
0	1	1	1	1	1	1	4	4	4	4	4	4	4	5	5	5	5
1		2	2	2	2	2	2	2	1	1	1	1	1	1	6	6	6
2				3	3	3	3	3	3	3	2	2	2	2	2	2	1
Fault?	Y	Y	N	Y	N	N	Y	N	Y	N	Y	N	N	Y	Y	N	Y

The total number of faults is 9.