CHAPTER VIII VIRTUAL MEMORY REVIEW QUESTIONS

Jehan-François Pâris
jfparis@uh.edu

Chapter overview

- Virtual Memory
  - Address translation
  - On-demand fetch
- Page table organization
- Page replacement policies
  - Performance issues

Problem

- A computer has 32 bit addresses and a virtual memory with a page size of 8 kilobytes.
    - How many bits are used by the *byte offset*?
    - What is the size of a *page table*?

First part

- A computer has 32 bit addresses and a virtual memory with a page size of 8 kilobytes.
  - How many bits are used by the *byte offset*?

Solution

- A computer has 32 bit addresses and a virtual memory with a page size of 8 kilobytes.
  - How many bits are used by the *byte offset*?
    - 8 kilobytes = 2^{13} bytes
    - The byte offset uses 13 bits

Second part

- A computer has 32 bit addresses and a virtual memory with a page size of 8 kilobytes.
  - What is the size of a *page table*?

Solution

- A computer has 32 bit addresses and a virtual memory with a page size of 8 kilobytes.
  - What is the size of a *page table*?
    - Since the byte offset uses 13 bits, the page number will use 32 – 13 = 19 bits
    - Page tables will have 2^{19} = 512K entries

Problem

- A computer system has 32-bit addresses and a page size of 4 kilobytes.
  - What is the maximum number of pages a process can have?
  - How many bits of the virtual address will remain *unchanged* during the address translation process?

First part

- A computer system has 32-bit addresses and a page size of 4 kilobytes.
  - What is the maximum number of pages a process can have?

Solution

- A computer system has 32-bit addresses and a page size of 4 kilobytes.
  - What is the maximum number of pages a program can have?
    - We divide the size of the virtual address space by the page size:

$$2^{32}\text{ B} / 4\text{ KB} = 2^{32} / 2^{12} = 2^{20} = 1\text{ M}$$

Second part

- A computer system has 32-bit addresses and a page size of 4 kilobytes.
  - How many bits of the virtual address will remain ***unchanged*** during the address translation process?

Solution

- A computer system has 32-bit addresses and a page size of 4 kilobytes.
  - How many bits of the virtual address will remain ***unchanged*** during the address translation process?
    - Since the page size is 4 KB = 2^{12} B,
      the 12 least significant bits of the virtual address will remain unchanged.

Problem

- An old virtual memory system has 512 MB of main memory, a virtual address space of 4 GB and a page size of 2 KB. Each page table entry occupies 4 bytes.
  - How many bits of the virtual address will remain ***unchanged*** by the address translation process?
  - What is the size of a page table?
  - How many page frames are there in main memory?

First part

- An old virtual memory system has 512 MB of main memory, a virtual address space of 4 GB and a page size of 2 KB. Each page table entry occupies 4 bytes.
  - How many bits of the virtual address will remain ***unchanged*** by the address translation process?

Solution

- An old virtual memory system has 512 MB of main memory, a virtual address space of 4 GB and a page size of 2 KB. Each page table entry occupies 4 bytes.
  - How many bits of the virtual address will remain ***unchanged*** by the address translation process?
    - Since the page size is 2KB = 2^{11} B,
      the 11 least significant bits of the virtual address will remain unchanged

Second part

- An old virtual memory system has 512 MB of main memory, a virtual address space of 4 GB and a page size of 2 KB. Each page table entry occupies 4 bytes.
  - What is the size of a page table?

Solution

- An old virtual memory system has 512 MB of main memory, a virtual address space of 4 GB and a page size of 2 KB. Each page table entry occupies 4 bytes.
  - What is the size of a page table?
  - We divide the size of the virtual address space by the page size:

    4GB/2KB= $2^{32}/2^{11}= 2^{21}$ entries or

    2^{21}x4 B = 2^{23} B = 8 MB

Third part

- A virtual memory system has 512 MB of main memory, a virtual address space of 4 GB and a page size of 2KB.
  Each page table entry occupies 4 bytes.
  - How many page frames are there in main memory?

Solution

- An old virtual memory system has 512 MB of main memory, a virtual address space of 4 GB and a page size of 2 KB. Each page table entry occupies 4 bytes.
  - How many page frames are there in main memory?
    - We divide the size of the main memory by the page size:
      512 MB/2 KB= $2^{29}/2^{11}$ = 2^{18} = **256 K** page frames.

Problem

- Given the following page reference string

  `0 1 1 0 1 1 0 0 2  0`

  and a very small memory that can only accommodate two pages, how many page faults will occur if the memory is managed

  A. By a FIFO policy

  B. By an LRU policy

Answer (I)

- Given the following page reference string

  `0 1 1 0 1 1 0 0 2 0`

  and a very small memory that can only accommodate two pages, the FIFO policy will cause four page faults
  - Fetch page 0
  - Fetch page 1
  - Fetch page 2 and expel page 0
  - Fetch again page 0 and expel page 1

Answer (II)

- Given the following page reference string

  0 1 1 0 1 1 0 0 2 0

  and a very small memory that can only accommodate two pages, the LRU policy will cause ***three*** page faults
  - Fetch page 0
  - Fetch page 1
  - Fetch page 2 and expel page 1

More review questions

True or false

- A computer will never have a page referenced bit **and** a missing bit
- The dirty bit indicates whether a page has been recently accessed
- A page fault rate of one page fault per one thousand references is a good page fault rate
- A TLB miss rate of one miss per one thousand references is a good miss rate

Solution (I)

- A computer will never have a page referenced bit **and** a missing bit **FALSE**
- The dirty bit indicates whether a page has been recently accessed
- A page fault rate of one page fault per one thousand references is a good page fault rate
- A TLB miss rate of one miss per one thousand references is a good miss rate

Solution (II)

- A computer will never have a page referenced bit **and** a missing bit **FALSE**
- The dirty bit indicates whether a page has been recently accessed **FALSE**
- A page fault rate of one page fault per one thousand references is a good page fault rate
- A TLB miss rate of one miss per one thousand references is a good miss rate

Solution (III)

- A computer will never have a page referenced bit **and** a missing bit **FALSE**
- The dirty bit indicates whether a page has been recently accessed **FALSE**
- A page fault rate of one page fault per one thousand references is a good page fault rate **FALSE**
- A TLB miss rate of one miss per one thousand references is a good miss rate

Solution (IV)

- A computer will never have a page referenced bit **and** a missing bit **FALSE**
- The dirty bit indicates whether a page has been recently accessed **FALSE**
- A page fault rate of one page fault per one thousand references is a good page fault rate **FALSE**
- A TLB miss rate of one miss per one thousand references is a good miss rate **TRUE**

Page table organization

- Which page table organization allows entire page tables to reside in main memory?
- How is it possible?

Answer

- Which page table organization allows entire page tables to reside in main memory?
  - Inverted page tables
- How is it possible?
  - Inverted page tables only keep track of the pages that are present in main memory.

Page Replacement Policies

- Among the five following page replacement policies:

  Local LRU, ***Global LRU***, ***Berkeley Clock***, ***Mach*** and ***Windows***

  Which one(s)

  - Support ***real-time processes*** ?
  - Simulate a ***page-referenced bit*** ?
  - Are partially based on the ***FIFO*** policy ?

First part

- Among the five following page replacement policies**:**

  Local LRU, ***Global LRU***, ***Berkeley Clock***, ***Mach*** and ***Windows***

  - Which one(s) support real-time processes?

Solution

- Among the five following page replacement policies:

  Local LRU, ***Global LRU***, ***Berkeley Clock***, ***Mach*** and ***Windows***

  - Which one(s) support real-time processes?
    - Windows because each process has a fixed-size minimum resident set.

Second part

- Among the five following page replacement policies:

  Local LRU, ***Global LRU***, ***Berkeley Clock***, ***Mach*** and ***Windows***

  - Which one(s) simulate a ***page-referenced bit*** ?

Solution

- Among the five following page replacement policies:

  Local LRU, ***Global LRU***, ***Berkeley Clock***, ***Mach*** and ***Windows***

  - Which one(s) simulate a ***page-referenced bit***?
    - Berkeley UNIX is the only one.

Third part

- Among the five following page replacement policies**:**
- ***Local LRU***, ***Global LRU***, ***Berkeley Clock***, ***Mach*** and ***Windows***
  - Which one(s) are partially based on the ***FIFO*** ?

Solution

- Among the five following page replacement policies:
- ***Local LRU***, ***Global LRU***, ***Berkeley Clock***, ***Mach*** and ***Windows***
  - Which one(s) are partially based on the ***FIFO*** ?
    - Mach and Windows.

Page Replacement Policies

- Give examples of
  - Very bad page replacement policies?
  - Policies that are too costly to implement?
  - Good policies that do not require any hardware support?

Solution (I)

- Give examples of
  - Very bad page replacement policies?

    Local FIFO, Global FIFO
  - Policies that are too costly to implement?
  - Good policies that do not require any hardware support?

Solution (II)

- Give examples of
  - Very bad page replacement policies?

    Local FIFO, Global FIFO
  - Policies that are too costly to implement?

    Local LRU, Global LRU, Working Set
  - Good policies that do not require any hardware support?

Solution (III)

- Give examples of
  - Very bad page replacement policies?

    Local FIFO, Global FIFO
  - Policies that are too costly to implement?

    Local LRU, Global LRU, Working Set
  - Good policies that do not require any hardware support?

    Mach, Berkeley Clock, Windows