Virtual Memory (Long's Lecture)

Paging and page table

- virtual addresses mapped to physical addresses
    - unit of mapping is called a page
    - all addresses in the same virtual page are in the same physical page
    - page table entry contains translation for a single page
- table translates virtual page number to physical page number
    - note all virtual memory has a physical page
    - not every physical page need be used
- Overlay: piece of software called overlay manager: to get something from memory
- Make everything the same size: so everything can fit anywhere
    - cut programs into the same size pieces
    - lie to the CPU, CPU thinks programs are next to each other -> MMU does this lie
        - programs are virtually contiguous to each other
        - programs are physically spread out from each other

What's in a page table entry?

- each entry in the page table contains
    - valid bit: set if this logical page number has a corresponding physical frame in memory
        - if not valid, remainder of PTE is irrelevant
- page frame number: page in physical memory -> the top half of addresses
- referenced but: set if data on the page has been accessed
- dirty (modified) but: set if the data on the page has been modified
- protection information

Mapping logical addresses to physical addresses:

- split the address from CPU into two pieces
    - page number (p)
    - page offset (d)
- page number
    - index into page table
    - page table contains base address pf page in physical memory
- page offset
    - added to base address to get actual physical memory address
- page size = 2^d bytes

Two-Level Page Tables

- problem: page tables can be too large
    - 2^32 in 4KB pages -> 1 million PTEs
    - worst for 64-bit addressing
- solution: use multi-level page tables
    - page size in first page table is large
    - PTE marked invalid in first page table needs no 2nd level page table
- 1st level page table has pointers to 2nd level page table
- 2nd level page table has actual physical page numbers in it
- goal: not to jump around: locality is key

How long do memory accesses take?

- assume the following times:
    - TLM lookup time = a (often zero - overlapped in CPU)
    - Memory access time = m
- Hit ration (h) is percentage of time that a logical page number is found in the TLB
    - larger TLB usually means higher h
    - TLB structure can affect h as well
- effective access time (an average) is calculated as:
    - EAT = (m+A)h + (m+m+a)(1-h)
    - EAT = a + (2-h)m
- ...

Inverted page table

- reduce page table size further: keep one entry for each frame memory
    - alternative: merge tables for pages in memory and on disk
- PTE contains
    - virtual address pointing to this frame
    - info about the process that owns this page
- Search page table by
    - hashing the virtual page number and process ID
    - Starting at the entry corresponding to the hash result
    - search until either the entry is found or a limit is reached
- Page frame number is index of PTE
- Improve performance by using more advanced hashing algorithms