Seongwoo Choi
Week 6
CMPS 111
Winter 2017

Note 6

Basic Memory Management

Swapping
Virtual memory
Page replacement algorithms
Modeling page replacement algorithms
Design issues for paging systems
Implementation issues
Segmentation

In the world we are living, we want to have memory that very large capacity, very fast , non-volatile. However, in a reality, this cannot happen since we can have only two options among those: very large, very fast, affordable. We can pick any two.
The main goal of the memory management to make the real world as much like the ideal world as possible.

Memory hierarchy is when different levels of memory, some are small & fast, others are large & slow exist.
Usually these levels are included:
Cache: small amount of fast, expensive memory
L1 (level1) cache is usually on the CPU
L2 is may be on of off chip
And L3 cache is off-chip, made of SRAM
Main memory: medium-speed, medium price memory
Disk: many gigabytes of slow, cheap, non-volatile memory

Memory manager handles the memory hierarchy.

Components include:
Operating system (perhaps with device drivers)
Single process

Goal: lay these out in memory
Memory protection may not be an issue.
Flexibility may still be useful

No swapping or paging
Fixed partitions: multiple programs
Fixed memory partitions
Divide memory into fixed spaces
Assign a process to a space when it’s free

Mechanism
Separate input queues for each partition
Single input queue: better ability to optimize CPU usage.

Throughput
Response time that we do not want

Modeling Multiprogramming
More I/O wait means less processor utilization

Memory and multiprogramming
Executable files
Memory needs two things for multiprogramming
• Relocation
• Protection

❖ The OS cannot be certain where a program will be loaded in memory
• Variables and procedures can't use absolute locations in memory
• Several ways to guarantee this

❖ The OS must keep processes' memory separate
- Protect a process from other processes reading or modifying its own
- memory
- Protect a process from modifying its own memory in undesirable ways (such as writing to program code)

Base and Limit registers
Special CPU registers: base & limit
- Access to the registers limited to system mode
- Registers contain

• Base: start of the process's memory partition
• Limit: length of the process's memory partition
Address generation
Physical address: location in actual
memory
Logical address: location from the process's point of view
Physical address = base + logical address
Logical address larger than limit ➙ error
Policy vs. Mechanism

Swapping: leaving room to grow
Need to allow for programs to grow
-> allocate more memory for data
-> larger stack
Handled by allocating more space than is necessary at the start
-> Insufficient: wastes memory that is not currently in use
-> What if the process requests too much memory/

Memory compaction:

Tracking memory usage to bitmaps
Operating system needs to track allocation state of memory

Regions that are available to hand out
Regions that are in use
Possibly what they're being used for

Multiple approaches to doing this
Bitmap
Linked list
Buddy allocation
Slab allocation

Techniques also used for tracking free / allocated storage space

Keep track of free/allocated memory regions

Allocating memory
Search through region list to find a large enough space
Suppose there are several choices: which one to use?
First fit: the first suitable hole on the list
Next fit: the first suitable after the previously allocated
Best fit: the smallest hole that is larger than the desired region
Worst fit: the largest available hole (leaves largest fragment)

Freeing memory
Allocation structures must be updated when memory is freed.
Easy with bitmaps: just set the appropriate bits in the map
Linked lists: modify adjacent elements as needed
Merge adjacent free regions into a single region
May involve merging two regions with the just-freed area.

Buddy allocation
Allocate memory in powers of two
Good for objects of varied sizes
Split larger chunks to create two smaller chunks
When chunk is freed, see if it can be combined with its buddy to rebuild a larger chiunk
Different from continuous
Contiguous is the word we want to use.