What is an address and how is one interpreted?

How do we allow multiple processes to use main memory simultaneously?

Where is the executing process?

Memory Management

- Distributed Systems
- File Systems and I/O Storage

Remainings:
- Memory Management

Next:
- Synchronization & Deadlock
- CPU Scheduling
- Processes &Threads

Discussed:

Where we are in the course
Virtual Address: an address relative to the start of a process's address space.

Physical Address: a real address in memory

Segment: a chunk of memory assigned to a process.

Memory Management Terminology:

CPU fetches instructions and data from memory while executing the program.

The OS loads the program into memory.

Program executable starts out on disk.

Background: Computer Architecture
• OS is protected from process by checking addresses used by process.
• Maximum address = Memory size - OS size
• Compiler can generate physical addresses.
• Process executes in a contiguous section of memory.
• Process is always loaded starting at address 0.
• One process executes at a time.
• OS gets a fixed part of memory (highest memory in DOS).

Uniprogramming

Where it wants in memory:

Execution time: Compiler generates an address, and OS can place it any

not move in memory.

determines the process's starting position. Once the process loads, it does
load time: Compiler generates an address, but at load time the OS
nothing.

memory starting from some fixed starting position. The OS does
Compile time: The compiler generates the exact physical location in

How do Programs Generate Instruction and data addresses?
Memory

Performance of CPU and memory should not be degraded badly due to

Efficiency:

- Process must not be able to corrupt the OS.
- Process must not be able to corrupt each other.

Safety:

- Process should not care what physical position of memory they are assigned to.
- No process should be aware that memory is shared.
- We want multiple processes to coexist in memory.

Transparency:

Multiple Programs Share Memory

Simple, but does not allow for overlap of I/O and computation.
If a literal address is the process's takes an address and ignores the physical address.

- Hardware compares address with limit register (address must be less than base).
- Hardware and relocation register (base) to virtual address to get a physical address.

**Dynamic Relocation:**

- **move it:** (why?)
- Once a process is assigned a place in memory and starts executing, the OS cannot
  - at load time, the OS adjusts the addresses in a process to reflect its position in

**Static Relocation:**

- **Relocation**

The first (smallest) physical address of the process is the base address and the largest

- Load a process by allocating a contiguous segment of memory in which the process fits.
- Memory size - OS size = Assume at compile time that the process starts at 0 with a maximum address
- Put the OS in the highest memory.

**Relocation**
be moved which is very slow. But if a process grows, it may have to have reference to physical to address translation and memory checks and virtual memory. Each memory reference is checked. Processes are largely unaware of sharing.

Properties: Relocation

- Compiles memory management: must fit in memory
- Degree of multiprogramming is very limited since all memory of all active processes
- Process is still limited to physical memory size
- Can't share memory (such as program text) between processes
- Slow down hardware due to the load on every memory reference

Disadvantages:
- Simple, fast hardware: two special registers, an add, and a compare.
- OS can grow a process over time.
- OS can easily move a process during execution.

Advantages: Relocation
utilization than worst-fit. First-fit is generally faster than best-fit.

Simulations show first-fit and best-fit usually yield better storage

search the entire list or keep the list sorted.

WORST-FIT: Allocate the largest hole to the process. Again the OS must

hole list. The OS must search the entire list or store the list sorted by size

process. The OS must search the entire list or store the list sorted by size

BEST-FIT: Allocate the smallest hole that is big enough to hold the

end.

search can start with the first hole, or where the previous first-fit search

FIRST-FIT: Allocate the first one in the list in which the process fits. The

Memory Allocation Policies

Memory Management: Memory Allocation

Given a new process, the OS must decide which hole to use for the process

Holes: pieces of free memory (shaded above in figure)

of which memory is available and utilized.

As processes enter the system, grow, and terminate, the OS must keep track

of which memory is available and utilized.
Any other choice?

How big a block is created?

How much memory is moved?

**Compaction**

- Alternatives:
  - Alternative 1:
  - Alternative 2:

**Fragmentation**

Internal Fragmentation:
- We want an allocation policy that minimizes wasted space.

---

External Fragmentation:
- Frequent loading and unloading programs cause free space to be broken into little
Swappping allows the total memory being used by all processes to exceed
the amount of physical memory available, but increases context switch
•
It expensive for processes to grow over time.
Swappping allows the total memory to exceed
•
Segmentation allows multiple processes to share main memory, but makes
physical addresses just before accessing memory
•
Processes generally use virtual addresses which are translated into
•
Processes must reside in memory in order to execute.

Summary

How could or should swappining interact with CPU scheduling?
•
If swappining is part of the system, compaction is easy to add.
•
Updates the relocation and limit registers.
- With dynamic relocation, the OS finds a new position in memory for the process and
- With static relocation, the process must be put in the same position.
•
When process becomes active again, the OS must reload it in memory.
•
Roll out a process to disk, releasing all the memory it holds.
•