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

Department of Computer Science, Virginia Atheist

Distributed Systems
File Systems and I/O Storage

Remainings:

Memory Management

Next:

Synchronization & Deadlock
CPU Scheduling
Process & Threads

Discussed:

Where we are in the course

Department of Computer Science, Virginia Atheist
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
0S is protected from process by checking address used by process.

Maximum address = OS 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.

0S gets a fixed part of memory (highest memory in DOS).

Where do addresses come from?

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

not move in memory.

Determines the process’ 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?
Sharing:

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:

Assigned to.

Process should not care what physical portion of memory they are

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 the address is less than the limit, the process can access the memory.

- If the address is greater than or equal to the limit, a trap is generated.
- The OS is notified, and the process is handled accordingly.

Dynamic Relocation:

- Move to the new location.
- Once the process is assigned a place in memory and starts executing, the OS cannot relocate the process.
- At load time, the OS adjusts the addresses in a process to reflect its position in memory.

Static Relocation:

- The first (smallest) physical address of the process is the base address and the largest address the process can access is the limit address.
- Load a process by allocating a contiguous segment of memory in which the process fits.
- Memory size = OS size
- Put the OS in the highest memory.
- If the compile/link time the process starts at 0 with a maximum address.
be moved which is very slow.

Efficiency: Memory checks and virtual to physical address translation are

Safety: Each memory reference is checked.

Transparency: Processes are largely unaware of sharing.

Relocation: Properties

---

Disadvantages:
- Complicated 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 add on every memory reference.

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

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 search the entire list or store the list sorted by size.

**BEST-FIT:** Allocate the smallest hole that is big enough to hold the process. The OS must search the entire list or store the list sorted by size.

**FIRST-FIT:** Allocate the first one in the list in which the process fits. The search can start with the first hole, or where the previous first-fit search ended.

---

**Memory Allocation Policies**

- Given a new process, the OS must decide which hole to use for the process.
- **Holes:** pieces of free memory (shaded above in figure).
- **Terminates:** pieces of free memory. When a process is terminated, the OS must keep track of which memory is available and utilized.

---

**Memory Management:** Memory Allocation
Any other choice?

How big a block is created?

How much memory is moved?

Compaction

Internal Fragmentation:
- We want an allocation policy that minimizes wasted space.
- We use allocation that minimizes wasted space.
- 50 percent used: Simulations show that for every 10 allocated blocks, 5 blocks are
- More memory, but there is space not continuous
- Fragmentation exists when there is enough memory to fit a process in
- Frequent loading and unloading programs cause free space to be broken into little

External Fragmentation

Fragmentation
Time

Swapping allows the total memory being used by all processes to exceed
the amount of physical memory available, but increases context switch

If expensive for processes to grow over time.

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 swapping interact with CPU scheduling?

If swapping 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 read it in memory.

Roll out a process to disk, releasing all the memory it holds.