ECE 598 – Advanced Operating Systems
Lecture 10

Vince Weaver

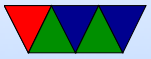
http://www.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

17 February 2015

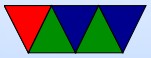
Announcements

- Homework #1 and #2 grades, HW#3 Coming soon



Various Types of Memory Management

- Application
- Operating System
- Hardware

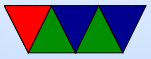


Application Memory Allocation on Linux

- compiler can (and will) optimize away memory accesses when possible and optimization enabled. As long as you don't use a pointer to the variable (the register keyword makes this explicit).

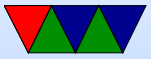
- Local variables on the stack
  Stack auto-grows down

- Global and static variables that are initialized in the data segment, loaded directly from disk.

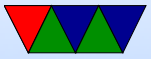


- Global and static variables initialized to zero in the bss segment.

- Dynamic variables allocated on the heap or via mmap
  - `malloc()` is not a syscall, but a library call.
  - Kernel interface is the `brk()` system call which moves the end of the data segment (essentially making the heap bigger)
  - `mmap()` initially was map file into memory so can be accessed with memory allocations rather than read/write



anonymous mmap will allocate memory range with no backing file.

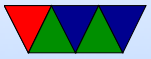


How Heap/Malloc Works

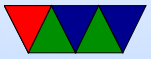
- Memory allocation libraries can vary underneath. Basically a big chunk of RAM is grabbed from the OS, and then split into parts in a custom way.

- The biggest problem is fragmentation, which happens when memory is freed in non-contiguous areas.

- dlmalloc – Doug Lea – glibc uses ptmalloc based on it. Memory allocated in chunks, with 8 or 16-byte header bins of same sized objects, doubly linked list

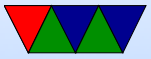


Small allocations (256kB?) closest power of two used
Larger, mmap used, multiple of page size.

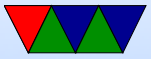


Manual vs Automatic

- With C you can manually allocate and free memory. Prone to errors:
  - Use-after-free errors
  - Buffer overflows
  - Memory Leaks
- High-level languages such as Java will automatically allocate memory for objects. The user never sees memory pointers. Unused memory areas are periodically freed via

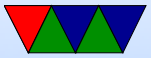


"garbage-collection". At the same time the memory can be compacted, avoiding fragmentation. Problem? Slow, not real-time, can be complex detangling complex memory dependency chains.

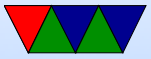


Memory Allocation on uclinux

- sbrk() doesnt work
- mmap
- overhead

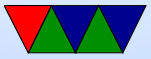


Operating System Setup



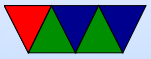
Mono-Programming

- Simple mono-programming: just OS and one program in memory at once (like DOS)

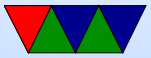


Fixed Multi-Programming

- Multiprogramming: let you run multiple tasks at once.
- Fixed Partitions of memory available. Jobs queued. When spot frees up job can run. Can have complex scheduling rules out which size and priority to give to jobs. Older mainframes (OS/MFT) used this.
- Relocations a problem
- Memory protection. Permissions on pages.

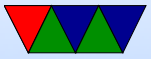


- Solution to both protection and permission in segments (with base offset and range that are valid to access)



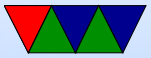
Swapping

- Timesharing systems. All jobs not fit in RAM?
- Swapping: bring in each program in entirety, run it a while, then when done writing all back out to disk.
- Paging: virtual memory.



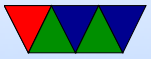
Fragmentation

- Enough memory available, but split up. How can fix?
- Memory compaction. Swap everything out, bring it back in (Relocating)



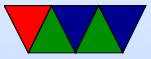
Tracking Memory

- What granularity should be used?
- Bitmap – chunks of memory, a bitmap representing it all.
  Have to search bitmap and find N consecutive empty areas for each allocation
- Free-lists, linked list of memory areas

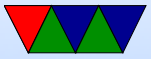


Fit Algorithms

- Scans memory, returns first block big enough to meet request. Fastest.
- Next fit. Picks up where the last first fit case left off (optimization)
- Best fit – search entire map and find hole that fits it best. Actually causes more fragmentation, end up with lots of tiny holes
- Worst Fit – always biggest hole. Not so great either.

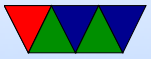


- Quick Fit – separate lists for more common sizes



Fixed Sized Allocation

- Memory Pool – fast – blocks of pre-allocated memory in power of two sizes that can be handed out
  fast to allocate/free
  fragmentation



Buddy Allocator

- Used by Linux
- Pre-allocated areas of various sizes
- Separate free lists for each area
- Rounds up request and tries to get for that size
  If none available, try next one up.