CS 61 Lecture 8 - September 26, 2013

Question: O(1) or O(n) algorithms good enough?

- Answer: O(1) is best but O(n) will only dock minimal points.

Manyalloc.c

- Calls alloc and free multiple times
- Way that it calls alloc and free depends on two constants, nslots and noperations
- Just testing the allocator by bombarding it with allocs and frees
- Maximum 'n' slots + 1 are concurrently active allocations with this benchmark function
- Calls a function that prints all the active allocations
- Then frees the allocations and the array itself

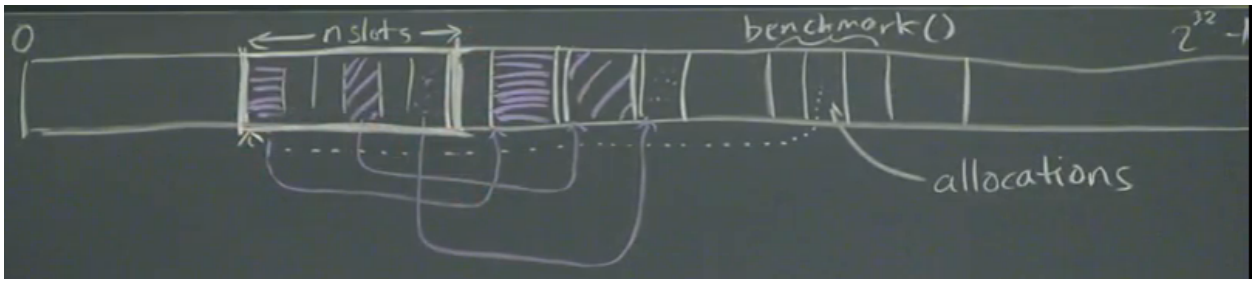

Figure 1: Review of manyallocs.c. Allocations points to an array that we allocate on the heap and nslots keeps track of the nslots+1 allocations.

Goal of Garbage Collector is to keep track of the missed frees using a pointer and keeping track of the size to clean memory automatically without calling free

Question: Why is it 1,002 blocks?

- Answer: Storing information about allocated memory regions in an array by sticking something at the end of the array. The remaining allocation comes from this metadata at the end, which is never freed.

Garbage Collector

- First we must find the garbage which is everything allocated that nobody is currently using
    - To find garbage, one way is to check the code text to see what's not being used
- Compiler manages the storage of local and global variables, not the heap.
- The garbage collector process:
    - Start with set of Roots
    - **Roots** - Accessible by definition
        - Includes stack and globals
    - Mark all accessible objects
        - Then by definition, the complement/unmarked objects are inaccessible

    - Free the unmarked objects
- Garbage collectors in C are “conservative” garbage collectors
    - **Conservative** - it will free some, which means it might accidentally preserve inaccessible memory, but that is okay.
    - **Precise** - means it will free all
- In completing the process there are various issues
    - How do you find accessible objects?
        - Look in a region of memory and look for pointers
    - How do we know if we’ve got a pointer?
        - It is inside the bounds of the heap, which we can estimate. We already have an array of all the addresses and the size of the allocations.
- Usually run garbage collector when memory runs out.

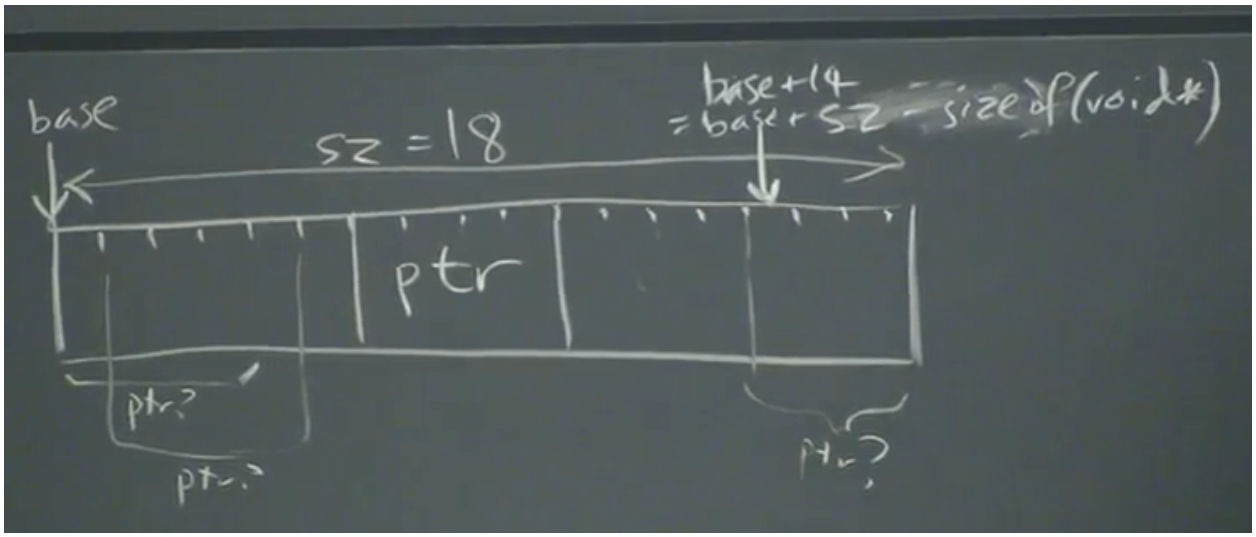

Figure 2: Iterating through one block of memory.

M61.c

- So what does the garbage collector do in this case?
    - find allocations from top of stack to bottom
    - if allocation is not marked, free it
    - recursion: if you free something, recheck that spot in the list
- in m61_find_allocations, How do we know where the top and bottom of the stack are?
    - stack top: somewhere in the heap
    - stack bottom: beginning of main (make this a global variable)
- Last chunk of memory checked?
    - Chunk 14 which is base + 14 = base + sz – sizeof(void *)
    - be careful of ‘off by 1 errors’!
- If there is a pointer, then we set ‘int mark’ equal to 1.
- Due to recursively going through the memory, the garbage collector is slow

    - To make it faster, check if the 'hypothetical_ptr' is equal to 0.
    - This is called **optimizing for the common case** because NULL || 0 is a large fraction of your memory.

Mark Sweep Garbage Collector

- First a mark phase where we find the accessible memory
- Then a sweep phase where we go through memory and free any unmarked accessible memory.

Macro Definitions

- '#define m61_free(x)' makes m61_free do nothing when you call it

Let's Talk Money

- Just as important as other types of costs (i.e. how fast? how much space?)
- One of the main things money influences is **where data gets stored**.
- (In class example) What is A/B?
    - A = $ for 1MB memory in 1955
    - B = $ for 1MB hard disk in 2012
    - A/B ends up being about 11 trillion
- Memory, flash, and hard disk storage for memory has become dramatically cheaper over the years.
- Why do people continue to use hard disk?
- **Latency** is the amount of time it takes to read/write a unit of data
    - Smaller is better
    - Register has the fastest latency
        - 0.5 nanoseconds
    - Hard disk has the slowest latency
        - Around 4-9 milliseconds
    - Hard disk is around 10 million times slower
- Throughput is the amount of data you can read/write per second.
    - Larger is better
    - Memory has throughput of over 1 GB/s
    - Hard disk (random access) has throughput of around 1MB/s
- Although hard disk is so much slower than flash, still being bought and used by companies like Google and Facebook.
    - Why?
        - Cost – flash is more than 20x more expensive than hard disk per byte
        - Hard disk also offers better durability
- **Durable storage** is storage that outlasts power failure

w01-syncbyte.c

- Program prints out 5.2 million '6's one character at a time.
- File being written into is written into disk

    - Durable storage
- Extremely slow performance
    - We can improve performance by **batching**
    - **Batching** is writing multiple characters at a time instead of one at a time