September 23, 2014

Garbage Collection

By Brian Arroyo, Victoria Gu, Kristen Faulkner, and Nicholas Larus-Stone

- What are some examples of dynamic memory errors in C?
    - Wild reads and wild writes
    - Double frees (could free a 2nd allocation that you didn't mean to free)
    - Memory Leaks (forgetting to free)
    - Invalid free (free something that was never allocated)
    - Dangling reference (attempting to use a pointer after it's memory has been freed)
- Why does garbage collection exist?
    - The goal of garbage collection is to take away our power to manage memory allocation. We lose power and flexibility, but in exchange we get easier allocation and fewer memory errors
        - Eliminates some dynamic memory errors
            - Double frees
            - Invalid frees
            - Dangling Reference
        - No segmentation faults
        - We can still get memory leaks
        - Does not eliminate wild reads/writes
    - Garbage collection results in memory-safe languages
- Memory unsafe-languages
    - C, C++, assembly
    - fast & dangerous & powerful!
- How do we make C a memory-safe language?
    - Make a garbage collector
        - However, in order to make C a little bit safer, we must give up some of the power that comes with it
        - This helps with double frees, invalid frees, and dangling references
    - Conservative collector (errs on the side of keeping memory rather than removing it when it is unsure if the memory is still needed)
    - "Mark-Sweep" garbage collector
        - Marks every piece of memory that is still allocated
        - Frees everything that hasn't been marked
        - Should touch entire heap
- Why does testgc-lecture work with a small number of allocations, but fail with 1 million?
    - We're calling the garbage collector at the end of the program
- Where should we call the garbage collector?
    - When malloc() fails, call the collector, and then try calling malloc() again
        - This works, but is slow because it keeps calling find_allocations
        - Instead, let's do a simple check and prempt the find_allocations if the pointer is null. Otherwise, run find_allocations
- What does a profiler do?

    - It tells you how much time different parts of your code use
    - Gives insight as to where code should be optimized

Memory

- Primary Memory Storage
    - Pros : Large, fast access
    - Cons: requires power to stay alive, costs 124x more per byte than hard disk memory
- Durable Memory Storage
    - slow access
    - survives power loss
    - Ex) Spinning hard drives, flash memory, tapes
- Cost comparison:
    - Hard disk costs about 3.8 x 10^-5 $/MB
    - RAM costs about 4.7 x 10^-3 $/MB
    - In 1955, RAM cost ~68750 times more than disk.
    - Comparing 1955 RAM to 2013 disk, the RAM is 11 trillion times more expensive
    - This explains why people program the way they do (maximize disk to minimize cost)
- Speed comparison:
    - ~10 ns to access RAM and ~10 ms to access hard disk
    - RAM is roughly 1 million times faster
- Storage hierarchy from fastest to slowest:
    - Registers, Primary Memory (RAM), flash/hard disk, Backup (tape)
    - As you go up, your memory becomes more expensive, smaller, and faster
    - I/O tries to utilize the advantages of the bottom of the pyramid while making it seem fast

File I/O

- By changing how we write to memory, we can change how fast our program runs
- per call cost: time it takes to call a write (C)
- per-unit cost: time it takes to write a byte (U)
- Total time: C + n*U (n= number of bytes to write)
- We can minimize the time it takes to write to memory by batching (writing lots of bytes at once)
    - This eliminates the per-unit cost of calling the write function
- We can reduce our per unit cost by deciding not to wait for the hard disk to confirm that a byte has been written (this means we're not syncing)
- We also have two write functions:
    - write, which is a system call provided by the OS
    - fwrite, which is a C library function that uses batching to write to hard disk faster than write can.
- Can use strace to profile a program