11-1

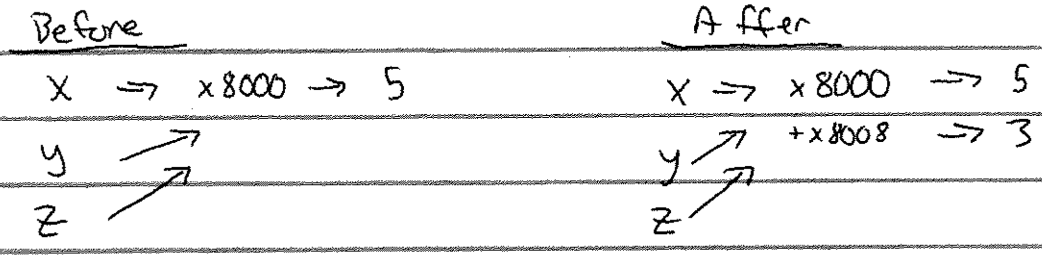
Manual

- Sound? NO.
- complete? NO. ???%

- memory overhead — fragmentation +copies
- cost — Malloc ~ lgn free - lgn
- latency — uniform w/ spikes

Reference Counting / Smart Pointers

Every object has a count of the number of aliases

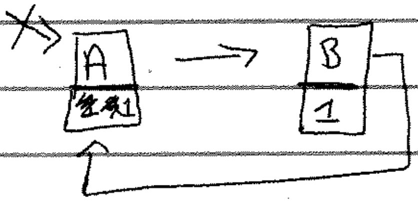


when you create a reference, increment count (retain)

when you destroy a reference, decrement (release)

If the count == 0, free

- Sound? NO — you might forget to call retain/release
- completeness?



cyclic references allways live forever

- memory overhead — fragmentation, no copies, cycles spend, counters take space

word counters ⇒ 25%, 50% ↑

byte counter ⇒ small, but significant 5%

```
retain(*p)                release(*p)
  if count < 255            if count < 255
    count++;                  count--;
```

- time — malloc + free same, still have spikes

updating counts isn't free → uniform tax to work

↳ w. case for caches

11-2/ Sound MM = "Garbage Collection" From 1960, John McCarth

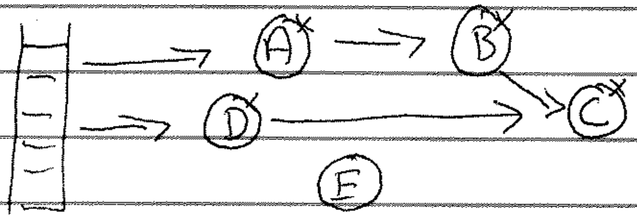
Root Set := the objects pointed by global vars, program text, and the stack

Live := the objects that are reachable

True Live/used := ~~the objects that if freed will cause the program to crash~~

reachable := a path exists from obj in root set to the object

A, B, C, D are live
and E is not



constraints: 1. pointers may not be manufactured

2. MM must know obj layout

GC - its job - figure out live

may free others → soundness

Mark and Sweep

- Figure out live: do a DFS over root set / object graph and record what you saw
- Free others: walk across mem from 0 to N

  if marked → unmark

  o.w → free

-overhead: memory - fragmentation, marks are free

live + dead

cpu/time - mark = O(live) sweep = O(mem)

malloc = same free = cheaper

latency - high, BUT tri-color real-time marking