18-1 / Memory Management

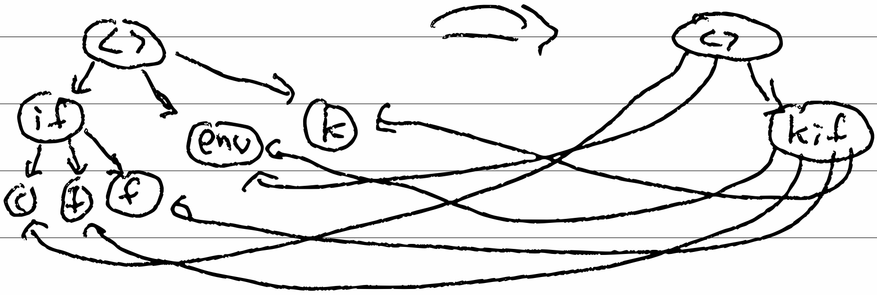
alloc: 4

5

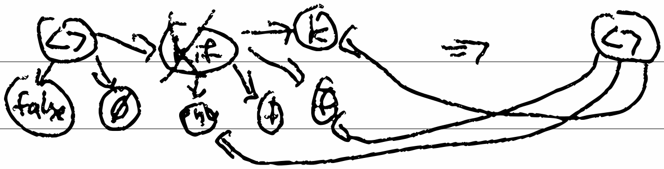
dealloc: 4

$\langle$ if c t f , env, k $\rangle$

$\mapsto$ $\langle$ c, env, kif (env, t, f, k) $\rangle$



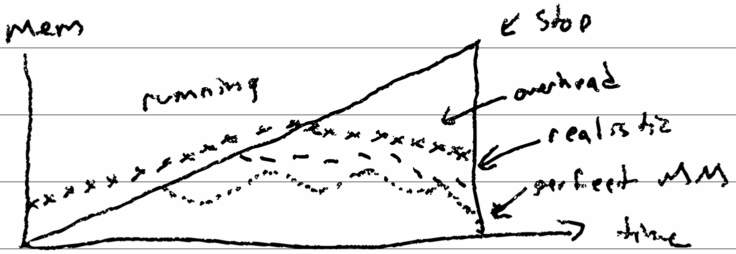
$\langle$ false, $\emptyset$, kif(env, t, f, k) $\rangle$ $\mapsto$ $\langle$ f, env, k $\rangle$



18-2/ What should a MM do?

- when to call free()

Calls free only at ~~the~~ the end of program



soundness . . . $f(x) = a$ if any MM, returned b

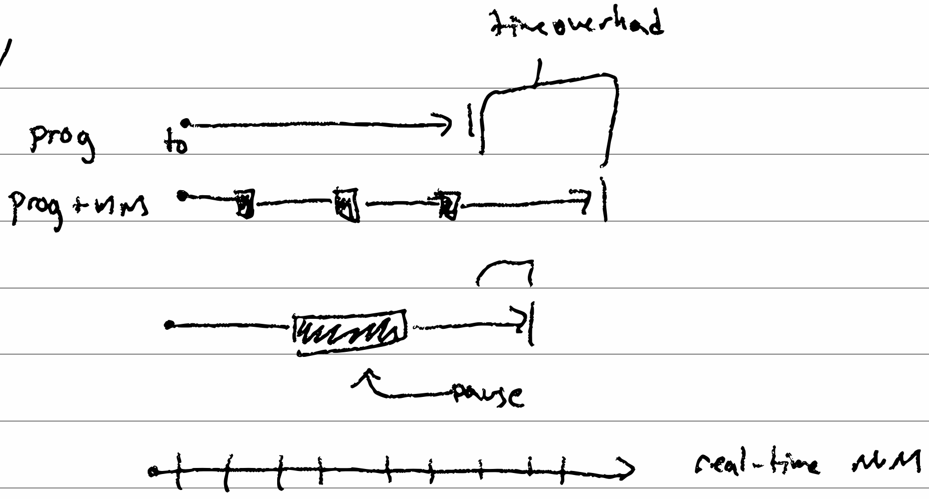
. . then it's wrong

we can call free() if any object won't be looked at

18-3/

```
        {
          obj  x = new()
20 <----  ....  use x
          ....
          if ( f () == true ) {
40 <----       use x; }
          ---------------------> safe Gon free *
          return  g() ; }
```

18-4



18-5) Manual insertion of free()

Is it sound? $\longrightarrow$ definitely <u>not</u>

alias $\longrightarrow$ but, two references to it

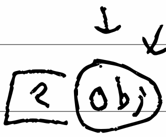
↓ one obj in memory (the pointer)

```
F() {                              global c
  char * c = malloc ....           g ( char * c) {
       .... g(c)  . . . . .          globalc = c
↗return; }
free(c)
```

- always assume you cannot free

track ownership - always make a copy rather than an alias

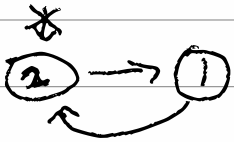
18-6 / Reference Counting / Smart pointer



```
retain(p) = p.count++
release(p) =
    if (--p.count == 0)
        free(p)
```

counts aren't free

64-bit . . . 8-bit



ref counting fails to release cycles