

12-1) resolve-complex / rcd

collect  $\rightarrow$  expr / complex

let  $v = c$

allocate  $\rightarrow$  expr / complex

$\xrightarrow{\text{unit}}$

global  $\rightarrow$  arg

(let  $v = c$  in e)

unit  $\rightarrow$  arg

(segm c e)

vector-ref  $\rightarrow$  expr / complex

vector-set  $\rightarrow$  expr / complex

12-2/ econ

$C_1 \Rightarrow C_2$

arg := ... | (global str) | (unit) | (var<sup>ty</sup>: ty)

exp := ... | (allocate num ty) | (vector-ref capnum)

stmt := ... | (collect num) | (vector-set! any num arg)

econ (let x = (allocate num ty) in body) =

segm (set! x (allocate num ty) (econ body))

econ (let \_ = (collect num) in body) =

segm (collect num) (econ body)

12-3) uncover-locals

old ans: info about names of variables

New ans: variables and their types

12-9)   $X_1 \rightarrow X_2$

arg := ... | (global str) | (type ty)

instr := ... | leag arg, arg  
→

load effective address (like  $f$  in C)

emit (global str) =  $\downarrow^{n-''}$  str(%rip)

emit (type ty)

emit (type (Vector S64 Bool S64)) =  $\xrightarrow{\text{read-only region}} \xrightarrow{\text{rotate}} \text{section}$

type27(%rip)

type27:

.quad 3 — Vector

.quad 3 — 3 elements

.quad 1 — S64

.quad 2 — Bool

.quad 1 — S64

0 → Unit

1 → S64

2 → Bool

3 → Vector

12-5 (→ X (select))

selecta (global str) ← (global st) // str(%rip)  
selecta (unit) ← \$1

Selecte<sup>dst</sup> (allocate num ty) =

movq (global "free\_ptr"), dst

addq \$(8 × (1+num)), (global "free\_ptr")

movq dst, %rax

leaq ty, %r11 ~~leaq ty, dst(0)~~

movq r11, %rax(0)

12-6/

selecte dst (vector-ref arg num) =

movq (selecte arg) , %rktk

movq %rktk (8 x (1+num)) , dst

Selects (vector-set! va num na) =

movq (selecta va) , %rax

movq (selecta na) , %r11

movq %r11, %rax (8 x (1+num))

12-7/

Select<sub>g</sub> (collect num) =

movq ROOT-STACK-REG, %rdi // 1st arg

movq \$num, %rsi // (2nd arg)

callq \_collect

12-8) live x  
conflicts x  
assign  
patch  
main

live / leaq is just many  
conflicts / leaq just like many  
update ralig

odd:

$\forall v \in L_A^k, \forall r \in \text{CALLER-SAVED}$ ,

add  $(r, v)$  to I

new, old v

$\forall v \in L_A^k$ , where  $M(v) = \text{Vector} \dots$

$\forall r \in \underline{\text{CALLEE-SAVED}}$

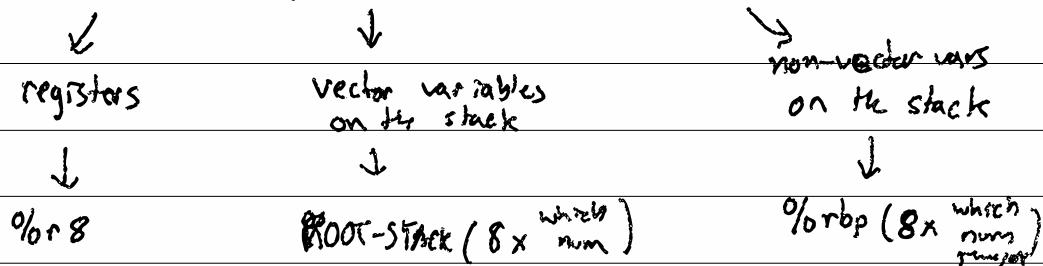
add  $(r, v)$  to I

12-9)

assign /

dd: vars either go to a register or the stack  
 $\leq 13$        $\nwarrow$        $\downarrow$   
                         $%rbp (8 \times (c-13))$

New: split arguments into three sets:



patch, leap is like mang

12-10

main/ BEGIN:

old: push all of the used callee-saved register

movq %rsp, %rbp

subq ss, %rsp

jmp BODY

callq -initilize

movq -root-stack\_start, root-reg

subq root-ss, root-reg

12-11

main : END:

old : movq %rax, %rdi  
callq \_printf\_int or \_printf\_bool

new : leaq ans-type, %rdi  
movq %rax, %rsi;  
callq \_printf\_val