Calling Convention; Simple Register Allocation

IR: CALL(NAME(f), e_1, ..., e_n)

x86: call f ≈ sub esp, 4
mov [esp], eip
jmp f

Calling Convention

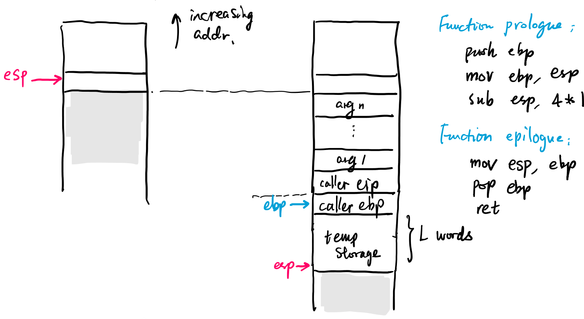
cdecl — args are pushed onto stack in reverse order
results returned in eax

T[[CALL(f, e_1, ..., e_n) ; MOVE(t, RET)]] =

T[[e_n]] t_n
push t_n
⋮
T[[e_1]] t_1
push t_1
call f
mov t, eax
add esp, 4*n

push t = sub esp, 4
mov [esp] t

pop t = mov t, [esp]
add esp, 4



Function prologue:
push ebp
mov ebp, esp
sub esp, 4*L

Function epilogue:
mov esp, ebp
pop ebp
ret

Constant-size frame opt.

ebp = esp + 4*L ⇒ avoid saving/restoring of ebp.

[ebp + k] = [esp + 4L + k]

prologue: sub esp, 4L
epilogue: add esp, 4L
ret

in C.

int a[f()];

call f
shl eax, 2
sub esp, eax
mov t_a, esp

Caller saved vs. callee saved.

eax ecx edx — ebp esp edi esi ebx — cdecl

Simplistic reg. allocation.

Idea: put all temps on stack at [ebp - k]

Any given x86 instr. uses ≤ 3 registers

eax ecx edx

push t ⇝ mov ecx, [ebp - k_t]
push t

mov a, [b+8] ⇝ mov edx, [ebp - k_b]
mov ecx, [edx+8]
mov [ebp - k_a], ecx

add x, y ⇝ mov ecx, [ebp - k_x]
mov edx, [ebp - k_y]
add ecx, edx
mov [ebp - k_x], ecx

Linear Scan Reg Allocation.

1. For each temp, find its live interval.
2. Scan instructions in seq.
   - assign real register to temp as temp becomes in use.
   - deassign as temp becomes dead.
   - no available real reg ⇒ spill to stack