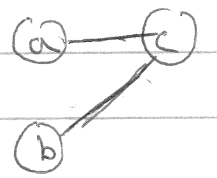
4-2

u interferes with v = "live at once"

$(\forall u, v. \exists k. \{u, v\} \subset Lafter(k)) = \times$ wrong

Graph $\mathcal{I} = (V, E)$ V = variables

$(u, v) \in E$ iff u inter. v



1. a goes in rax
...
5. a go in rbx

---

for (a, b, c) → put c on stack

for (a, b, e) → get c from stack

read c

For I_k in I_1 to I_n ...

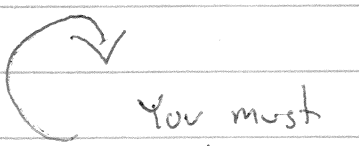
If I_k is (movq s d), then for $v \in Lafter(k)$

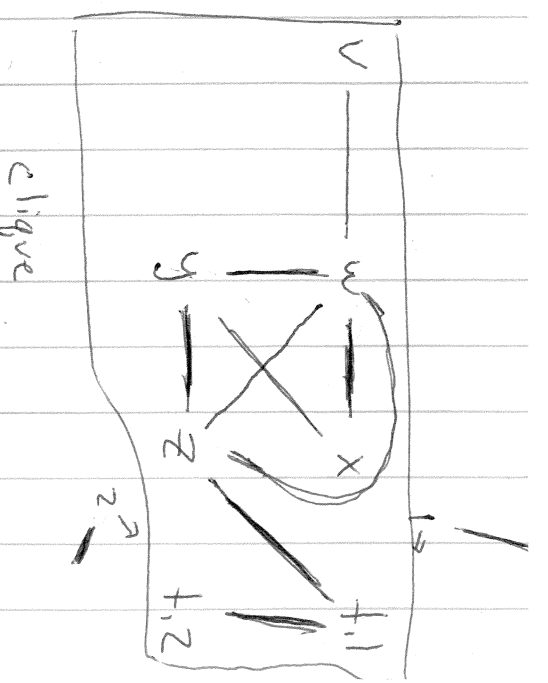
add (d, v) to E unless v = d or v = s

If I_k is like (addq s d), then for $v \in Lafter(k)$

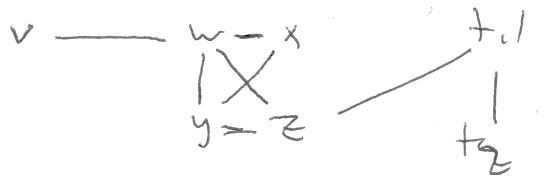
add (d, v) to E unless v = d

{} {3} {1, 2, 3} {1, 3} {1, 2} {1, 3} {2, 3} {2, 3} {3, 4} {3, 4} {4, 5} {4, 5} {5}

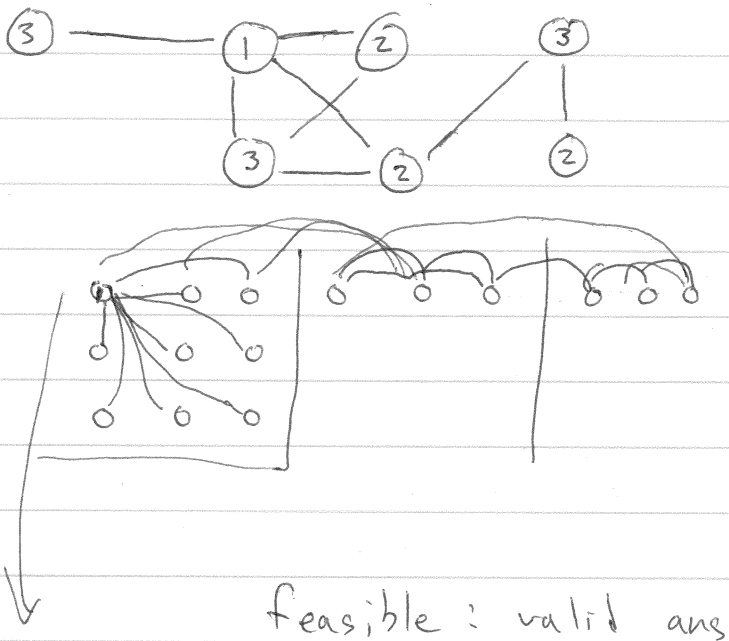


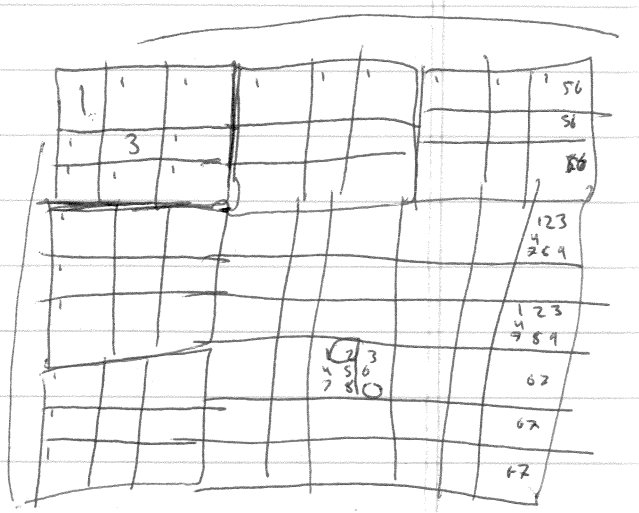


5-1



Goal: assign var to reg s.t. no two vars that are adj share the same register





Saturation
"Pencil Marks"

feasible: valid ans
optimal: ans w/ least registers

16 possible 5 caller-saves 11 registers

✱ ○ take 2^n (PTAS)

(1-a) ○ take $n^{f(a)}$ — polynomial time approx. scheme

a = .000001 f(a) = ~~3~~ 3-a

---

saturation(u) = { c | ∃v. v ∈ adj(v) and color(v) = c }

options(u) = REGs - saturation(u)

(v1, ..., vn) ↝ (∅, ..., ∅)

1. Pick var v from unassigned
2. Look at options(v), call that O
3. Pick one c, color(v) = c
4. Update saturation of neighbors

what's a good order?
- highest degree
- program order
- saturation order

O = ∅?
1. Backtrack
2. Spill (stack)

Clique order first

5-2

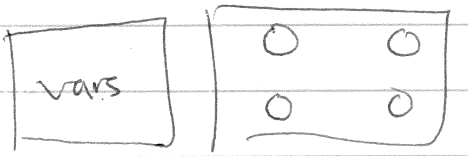
Input : Graph G
Output : Map color from V → REG

W ← V ; ← 0
while W ≠ ∅ do
  let v be a member of W with smallest
    options(v), largest degree(v), random o.w.
  select c s.t. c ∈ options(v) and c is minimal
  if options(v) empty, then c = stack(i++)
  o.w. color(v) = c
  W ← W - {v}
  → update adj neighbors of v for new saturations
    resort of queue

n · (n + lg n) O(n²)

---

return value is always rax (movq (var x) rax)
mult result is always rcx, rdx (multq (var y) (var z))



callq label → interfers w/ caller-saves registers
rax, rdx, rcx, rsi, rdi, r8, r9, r10, r11

7. push y callee: sp, bp, b, 12-15
(x=10, y=17, z=12)
8. (call label) y ∈ caller, x, z ∈ callee
(x=10, y=???, z=12)
9. pop y

5-3 move-biasing

(movq (var x) (var y))

↓

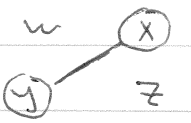
(movq (reg a) (reg b))

color(x) = a color(y) = b

color(x) = color(y) (assuming they don't interfere)

G:

Move-RELATED



1, 1

1, 2