Register Allocation

Lecture 11
Coloring a Graph

Interference graph (nodes are temps)

three colors
three registers

adjacent nodes should have different colors
Graph Coloring

If we can color these, we can color the rest.

nodes of significant degree (degree $\geq 3$)
Graph Coloring

a second phase of removing nodes of low degree (insignificant nodes)
remaining nodes all insignificant and can therefore be colored
Graph Coloring

add 2nd phase insignificants and color them
Graph Coloring

add 1st phase insignificants and color them
Need for Spills

No insignificant nodes after first phase!
Choose this node as a candidate for spilling and remove it.
Spill Candidate Becomes Spill

After coloring remainder, try to color spill candidate. If not possible, then spill it to memory.
Spilling

If we need to spill temp $t$:

1. rewrite the code to incorporate the spilling of $t$:
   
i. at each node defining $t$, replace $t$ with a new temp $t'$ and follow that node with a store: $n(r1) := t'$ (where $n$ is a new frame slot allocated using allocSpill)

   ii. at each node using $t$, replace $t$ with a new temp $t''$ and precede that node with a load: $t'' := n(r1)$

   Note that $t'$ and $t''$ will be live-out for only one instruction.

2. redo liveness analysis, construction of interference graph, and coloring using the rewritten code
Spilling Situation

4 temps
3 registers

Interference
Lifetimes
Effect of Spilling

Lifetimes

Interference

$a_1$, $a_2$, $b$, $c$, $d$
Coloring After Spilling

\[ \text{a}_1 \quad \text{a}_2 \quad \text{b} \quad \text{c} \quad \text{d} \]

Interference

Lifetimes

\[ \text{a}_2 \quad \text{a}_1 \]

\[ \text{b} \quad \text{c} \]

\[ \text{d} \]
Which Temp to Spill?

- **Spill the least used temp**
  - statically least used (fewest occurrences in the code)
  - dynamically least used (weight occurrences in loops higher)
  - this minimizes runtime cost of spills (number of loads and stores)

- **Spill the temp with the most interferences (largest number of adjacent nodes in interference graph)**
  - this removes the most edges, decreasing likelihood of further spills

✔ **Spill a temp that hasn’t been spilled before (?)**
  - but rewriting replaces spilled temps with new ones!?
Coallescing Moves

**live-in:** k j

\[
g := \text{mem}[j+12] \\
h := k - 1 \\
f := g * h \\
e := \text{mem}[j+8] \\
m := \text{mem}[j+16] \\
b := \text{mem}[f] \\
c := e + 8 \\
d := c \\
k := m + 4 \\
j := b
\]

**Assume 4 registers**
(i.e. 4 colors)

**live-out:** d k j
Simplification

Removal of insignificant nodes
Coalescing Moves

1. Coalescing $c$ and $d$
2. Coalescing $b$ and $j$
Further Simplification