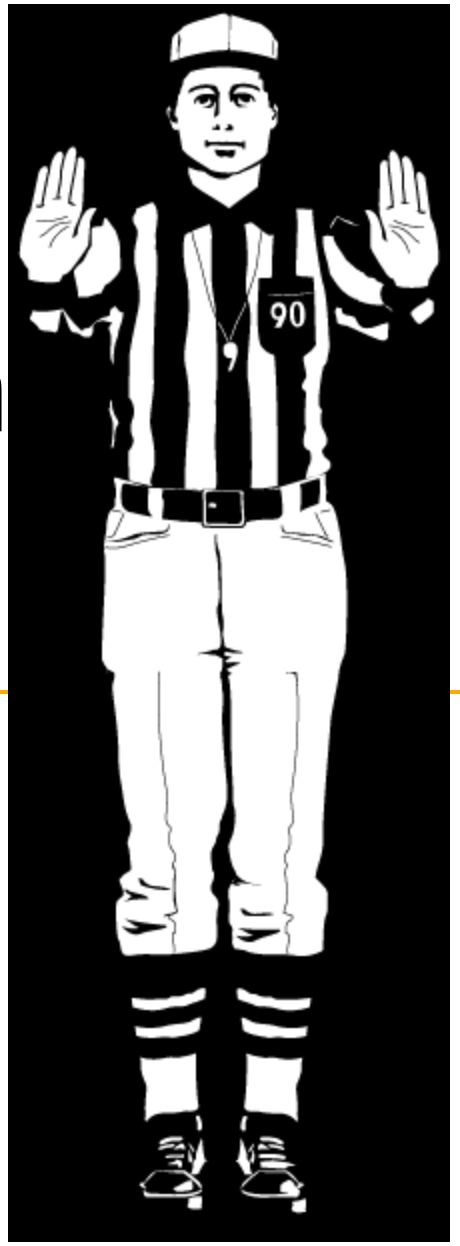
Register Allocation
Interference

March 23, 2011



CS132

Review: Liveness

When is a variable at a program point live?

Optimal answer?

Computable approximation?

Review: Liveness Analysis

For each instruction s:

Define $def(s)$ to be variables potentially modified by s.

Define $use(s)$ to be the variables accessed by s.

Solve for $livein(s)$ and $liveout(s)$ (e.g., by iteration)

$$livein(s) = use(s) \cup (liveout(s) \setminus def(s))$$

$$liveout(s) = \bigcup_{p \in succ(s)} livein(p)$$

$$liveout(terminal\ node) = \varnothing$$

Register Allocation

Maps temporaries to machine locations

Input: Assembly code using arbitrary number of temporaries.

Output: Equivalent code using only machine registers + memory

Note: I will refer to machine registers as temporaries as well

These are assigned to themselves by the allocator

Very rarely people distinguish between

register allocation (which variables will be in registers)

register assignment (which registers these will be in)

Interference Graph

Temporaries *interfere* if they cannot be stored in the same machine location.

Thus, we can define an (undirected) *interference graph*

Nodes =

Edges =

Graph Coloring

A *coloring* of a graph is an assignment of "colors" to nodes with no adjacent nodes having the same color.

A k*-coloring* of a graph is a coloring that uses at most k colors.

This leads to the *graph coloring problem*.

[Simplest form: yes/no.]

Graph Coloring Register Allocation

Elegant observation of Chaitin:

The result of a correct register allocation would be an assignment of registers to temporaries such that no interfering temporaries are assigned the same register.

Graph coloring where

the graph =

the colors =

Algorithm for GCRA

Step 1: Build the interference graph.

Step 2: Find a k-coloring, where k is the number of available machine registers.

Any questions before I ask you to implement this?

Solving the first step

Suppose we have the live-in and live-out sets for every instruction. How do we construct an interference graph?

Example 1

Which temporaries interfere?

```
a ← x+1
b ← a+x
return b
```

Example 2a

Which temporaries interfere?

```
a ← x+1
b ← a+x
return a
```

Example 2b

Which temporaries interfere?

```
a ← x+y
b ← f(x,3)
c ← a+b
return c
```

Example 2c

Which temporaries interfere?

```
a ← x+1
a ← a+1
return a
```

A Correct Algorithm?

Example 3a

Which temporaries interfere?

```
w ← 3
a ← x+y
b ← a
c ← a+x
d ← b+c
return d+w
```

Example 3a

Which temporaries interfere?

```
w ← 3
a ← x+y
b ← a
c ← a+x
a ← b+1
d ← b+c
return d+w
```