CS536 Lecture 24

Thursday 30 April 2015

Last class:

- Code Generation (finish)

Today:

- Code Optimization

One last thing on control flow code generation

Numeric approach:

Control-flow approach:

Code Optimization

Goal:

Tradeoff: Performance vs. safety

Difficulties:

Soundness, completeness, efficiency

Peephole Optimization

Pattern-match small obvious problems **after** code generation (essentially, counter-acting silliness of naïve code generator)

Correctness over efficiency

Examples:

```
sw  $t0  0($sp)
subu$sp  $sp(4)
lw  $t0  4($sp)
addu$sp  $sp 4
```

Can we optimize this? How? Is it sound?

CFG review:

Program as a flowchart

Nodes are “basic blocks”

Edges are control transfers (fallthrough, jump)

Peephole optimization: Limit optimizations to intra-block analysis.

Examples of peephole optimizations

Push followed by pop

Same register:

Different register:

Pop followed by push

Branch to next instruction

Examples of peephole optimizations (cont'd)

Jump to a jump

Jump around a jump

Useless operations

Faster operations

Passes

Copy Propagation