Compiler Design

Lecture-25

The right shift problem

Topics Covered

- The right shift problem
- Addition chains for multiplication

Replace Multiply by Shift

- `A := A * 4;`
  - Can be replaced by 2-bit left shift (signed/unsigned)
  - But must worry about overflow if language does
- `A := A / 4;`
  - If unsigned, can replace with shift right
  - But shift right arithmetic is a well-known problem
  - Language may allow it anyway (traditional C)

The right shift problem

- Arithmetic Right shift:
  - shift right and use sign bit to fill most significant bits

    -5 111111…1111111011

    SAR 111111…1111111101

    which is -3, not -2
  - in most languages -5/2 = -2

Addition chains for multiplication

- If multiply is very slow (or on a machine with no multiply instruction like the original SPARC), decomposing a constant operand into sum of powers of two can be effective:

```
X * 125    =    x * 128 - x*4 + x
```

  - two shifts, one subtract and one add, which may be faster than one multiply
  - Note similarity with efficient exponentiation method

Folding Jumps to Jumps

- A jump to an unconditional jump can copy the target address

```
        JNE lab1
        ...
lab1: JMP lab2
```

Can be replaced by:

```
        JNE lab2
```

As a result, lab1 may become dead (unreferenced)

Jump to Return

- A jump to a return can be replaced by a return

```
        JMP lab1
         ...
lab1:   RET
```

  - Can be replaced by

```
RET
```

    lab1 may become dead code

Usage of Machine idioms

- Use machine specific hardware instruction which may be less costly.

i := i + 1

ADD i, #1 INC i