

Plan for today

Finish instruction selection for x86

Register allocation

- spill all
- possible improvements over spill all

CS453 Lecture

x86 code gen and register allocation

1

Instruction selection for x86

Registers

- EAX, the accumulator
- EBX, the base register
- ECX, the counter register
- EDX, the data register
- ESI, the source register
- EDI, the destination register
- ESP, the stack pointer register
- EBP, the frame pointer register

Representations

- Constants prefixed with '\$', for example \$3, \$4, \$-5, etc
- Registers prefixed with '%', for example %eax, %esp, etc.

Some Instructions

- `movl -12(%ebp), %eax` // `M[%ebp-12] --> %eax`
- `imull -4(%ebp), %eax` // `M[%ebp-4] * %eax --> %eax`
- `cmpl -4(%ebp), %eax`
- `jge .L2` // `if (M[%ebp-4] >= %eax) goto .L2`

CS453 Lecture

x86 code gen and register allocation

2

Temps as destinations and sources

Destination (or Temp defines)

- in Translate to IR Trees
 - pointer to class instance created by NewExp
 - pointer to start of allocated array
 - array length is defined
 - result of function call
 - holds 0 or 1 to indicate result of a less than
- in CodeGen
 - a Temp holds the result of each expression evaluation

Source (or Temp uses)

- in Translate, a Temp is used in Tree.Stms generated for same AST node where Temp is defined
- in Translate, false body must come after StmCJUMP, but exit doesn't necessarily have to come after either (POSSIBLE PROBLEM)
- in CodeGen, when generating code for a parent node, Temps defined in the children nodes are often used

CS453 Lecture

x86 code gen and register allocation

3

Possible approaches for improving over spill all

Linear passes over the Assem.Instrs

- after spill all, remove loads from sw-lw pairs where the temp and frame location are the same
 - eg. `sw $t2, -16($fp)`
`lw $t2, -16($fp)`
- before spill all, assign each Temp to a callee-saved register until run out
 - eg. `t34 ==> frame.RA`
`t35 ==> frame.S0`
...
`calleeSaves = { RA, S0, S1, S2, S3, S4, S5, S6, S7}`
- after spill all, assign each frame location to a callee-saved register until run out
 - eg. `$fp-12 ==> frame.RA`
`$fp-16 ==> frame.S0`
...
`calleeSaves = { RA, S0, S1, S2, S3, S4, S5, S6, S7}`

CS453 Lecture

x86 code gen and register allocation

4

Example

```
.text
LS_Start:
LS_Start_framesize=100
LS_Start_paramsNregsaves=16
```

- the above function has $(100-16)/4$ locals and temps, 21 locals and temps

```
.text
LS_Search:
LS_Search_framesize=224
LS_Search_paramsNregsaves=16
```

- the above function has $(224-16)/4$ locals and temps, 52 locals and temps