Spilling

Simone Campanoni
simone.campanoni@northwestern.edu
A graph-coloring register allocator structure

- **Liveness analysis**
- **Interferences analysis**

**Register allocator**
- **Code analysis**
- **Graph coloring**

Spill
- **spill(f, var, prefix)**

**IN, OUT**

- **f**
- **f**
- **f with var spilled**
- **f without variables and with registers**
Spilling

• Procedure used by a register allocator with the following inputs
  • A function $f$
  • A variable $v$ that needs to be allocated to the stack
    (as local in L1)
  • A string (see later)

• This procedure modifies $f$ to allocate $v$ on the stack
  • Make a new location on the stack
  • Replace all writes to $v$
    with stores to the new stack location
  • Replace all reads from $v$
    with reads from the new stack location
Spilling example

2 registers are needed

Only 1 register is now needed!

All L2 instructions can use variables, but only some L1 instructions can access a memory location!
Spilling example (2)

For every instruction that uses the spilled variable:

- Create a new variable that starts with %S and ends with a new number
- Replace the original instruction using the new variable
- Add loads/stores around the new instruction

In between L2 and L1:

```c
(@myF 0
 %a <- 42
 %a += %a
 return )
```

```c
spillForL1(@myF, %a, %S)
```

```c
(@myF 0 1
 %S0 <- 42
 mem rsp 0 <- %S0
 %S1 <- mem rsp 0
 %S1 += %S1
 mem rsp 0 <- %S1
 return )
```
Spilling example (2)

What if we have only 1 register?

```
(@myF
  0
  %a <- 42
  %b <- 40
  %b += 2
  -%a *= %a
  return
)
```

```
spillForL1(@myF, %a, %S)
```

```
(@myF
  0 1
  %S0 <- 42
  mem rsp 0 <- %S0
  %b <- 40
  %b += 2
  %S1 <- mem rsp 0
  %S1 *= %S1
  mem rsp 0 <- %S1
  return
)
```
Register allocator

Code analysis

Graph coloring

f without variables and with registers

f (in between L2 and L1) with var spilled

Spill

spill(f, var, prefix)

f

Liveness needs to handle it

• L2 does not have callee-save registers
• Spiller cannot generate callee-save registers
• So the language in between L2 and L1 is just L2 plus stack locals of L1
Testing your spiller for homework #2

• Under L2/tests/spill there are the tests you have to pass

• To test:
  • To check all tests: make test_spill
  • To check one test: ./spill tests/spill/test1.L2f

• Check out each input/output for each test if you have doubts
  • tests/spill/test1.L2f
  • tests/spill/test1.L2f.out
Always have faith in your ability

Success will come your way eventually

Best of luck!