Compiler Project, Stage 5: Instruction Selection!

Computer Science 371
Amherst College
Spring 2014

This assignment is due on **Friday, May 9**.

1 Getting Ready

Go into your `cs371` directory and issue the following commands:

```
cp -r hw4 hw5
cp -r ~lamcgeoch/cs371/hw5/* hw5
cd hw5
rm Makefile
ln -s Makefile5 Makefile
```

In addition, issue the command `chmod -R a+w hw5` if you are working in a group directory. The effect of these commands is to give you a new directory, `hw5`. You have many new directories and files.

Try running `make`. Assuming that your `hw4` files compile correctly, it should run without errors.

2 New files in this Release

Makefile5: You should save your old Makefile and then rename this one to replace it.

arch: Contains object code required for library calls (for printing, string manipulation, etc.) on different architectures.

tests5: (Note the name!) Contains test programs and scripts for running them. You should strive to have your compiler work correctly on all the .java files in this directory.

minijava/Canonical.java: A main program for converting .icode1 files into linearized .icode2 files.

minijava/Interp2.java: A main program for interpreting .icode2 files.

minijava/CodeGen.java: A main program for converting .icode2 files into assembly code.

minijava/Canon: Code for canonicalization.

minijava/BackEnd: Lots of code for various back-end activities.

minijava/BackEnd/Arch/Linux64/CodeGen.java: **This is the file that you should fill in to do code generation.**

3 Your Task

Your task is to do instruction selection, working in the file listed above. We'll talk lots about this task in class, and I may have further written instructions and/or advice.

One way to print the result of your instruction selection is to uncomment lines that you will find in **minijava/BackEnd/ICode/Method.java**.

4 Running Various Scripts

To compile, link, and execute a .java program, you can type:

```
./doit Hanoi.java
```

Once you have an executable, you can simply run:

```
./Hanoi
```

You can run separate programs for various phases of compilation. To produce intermediate code for the Linux64 architecture, do:

```
./compile Hanoi.java
```

To linearize the resulting intermediate code, do:

```
./canonical Hanoi.icode1
```

(You can run an interpreter on .icode2 files, but you probably won't need to do this. The proper sequence of commands would be:

```
./compile -target simple Hanoi.java
./canonical Hanoi.icode1
./interp2 Hanoi.icode2
```

In other words, the interpreter only works when you are targetting the simple architecture.)

To create assembly code from .icode2 files, do:

```
./codegen Hanoi.icode2
```

To assemble and link the result assembly file, do:

```
./link Hanoi.s
```

5 Submitting Your Work

To do an electronic submission in the usual way.