Timothy Colaneri
No Partner
CSC411 Profiling README
Spring 2020

Acknowledgements:

-I did not seek any outside help on this one. I did however take influence from the Professor's lecture when he went over his own implementation.

Most expensive routine: interp_prog()

This is the main interpretation loop in my universal machine implementation. Essentially, its instructions read like this…

- Get the next instruction in the program
- Unpack the instruction word and determine what kind of instruction it is
- Interpret the instruction

This function is somewhat large at 140 lines due to in-lining of its function calls. After thoroughly going over this assembly, I think I may have arrived at a very efficient implementation of it. The extraction of the instructions at the beginning of the function is very efficient. A single shift to get the opcode followed by a single comparison handles what type of instruction we are getting, followed by a jump to a body with the appropriate bit shifts. The bit shifting is done by constant values that do not need to be loaded from memory. Following this, we only take a few move operations before we jump into the interpretation switch. The actual interpreting is in the switch which takes place at the end of this function(lines ~60-140). Each of the switch instructions are very concise, consisting of only a few lines. The assembly code simply moves the needed values into place, executes the instruction, and then jumps back to the beginning. I do not see an obvious way to improve this code.

Time Spent:

Analysis: ~2-3 hours
Solving: ~20 hours