CSE 240a Fall 2013 Homework 1, Due, Thursday, October 10

Remember to type your hw solutions.

Homework should be typed. Do not forget to put your name.

(from the book) H&P A.7a A.8a, (below) P1, P2, P3, P4, P5, P6, P7

P1. Program A runs 20 billion instructions on a 2 GHz processor, and achieves a CPI of 1.5. Introduction of a new instruction to the ISA (and recompiling the code) would allow a reduction in the instruction count to 19 billion instructions, resulting in a speedup of 1.2. What is the CPI for the new code on the improved processor?

P2. Machine B runs at 2 GHz and has a CPI of 1.3 for a particular program. Machine C runs at 5 GHz and has a CPI of 2.4 for that program, while executing 20% more instructions. Which machine is faster? What is the speedup over the slower machine?

P3. You have an optimization that speeds up floating point operations by a factor of 2, but does not help other instructions. You have another optimization that only speeds up non-FP instructions by 10% (that is, it speeds up those instructions by a factor of 1.1), and you want to decide which to use. Suppose your favorite program has 10% of its instructions FP operations. Further, assume FP operations have a CPI of 3.0, while non-FP instructions have a CPI of 1.0.

P4. (Amdahl's law backwards) You improve your memory subsystem so that memory latencies are sped up by a factor of 2.4. After applying the optimization, you observe that you now spend half your time on waiting for memory. What percentage of the original execution (before the optimization) was spent waiting for memory?

P5. Give me the code for $D = (A + B) / C$ in an accumulator, stack, GPR, and load-store ISA. For each give me the total size (in bits) of the code, assuming (a) fixed-length instructions, and then (b) variable-length instructions. Assume opcodes are 6 bits, memory addresses are 32 bits, and register specifiers are 4 bits. You'll probably make other assumptions -- just tell me what they are.

P6. Suppose we wanted to add the auto-increment addressing mode to the MIPS ISA -- e.g., lw R1, 1000(R2++). This saves an instruction every time we observe a load followed by an increment of the address register. Is this a good idea? Consider only performance, and assume that we have to increase cycle time by 5% to accommodate the new instruction, that 20% of our instructions are loads, that we can apply this change to 40% of all loads, and that the CPI doesn't change.

P7: Assume we are executing MIPS code where 15% of instructions are conditional branches, 20% are load instructions, 5% are stores, and the rest are arithmetic. For this code, (a) what percentage of all memory accesses are for data? (b) what percentage of all data accesses are reads? (c) what percentage of all memory accesses are reads? [minor hint -- what do we access memory for besides loads and stores to data memory?]