CSCI 2321 (Computer Design), Spring 2019

Homework 1

Credit: 40 points.

1 Reading

Be sure you have read, or at least skimmed, the assigned readings from Chapter 1.

2 Problems

Answer the following questions. You may write out your answers by hand or using a word processor or other program, but please submit hard copy, either in class or in one of my mailboxes (outside my office or in the ASO).

Tips:

- If the assignment asks you to do one or more problems from the textbook, *be sure* you get them from the edition specified in the syllabus; these sets of problems change from edition to edition.
- If a question requires you to do calculations, please show enough work to help me understand how you got the answer you did, so if you make a mistake I can give partial credit for anything you did get right. *Also,* it's a good idea to apply to any result a "reasonableness test" — for example, if a processor's clock rate is in the GHz range and it's executing a program with billions of instructions, execution time is likely to be in the range of a few seconds, and if it's different by a couple of orders of magnitude you probably made a mistake!

1. (10 points) Below are two lists, one of terms from Chapter 1 and one of short definitions/descriptions. For each definition say which term matches it best.

   Terms:

   - desktop computer
   - server
   - low-end server
   - supercomputer
   - embedded computer
   - multicore processor
   - RAM
   - CPU
   - operating system
   - compiler
   - bit
   - instruction
   - assembly language
   - machine language
   - assembler
   - high-level language
   - system software
   - application software

   Definitions/descriptions:

   (a) Computer used to run large problems and usually accessed via a network.

(b) Computer composed of hundreds or even thousands of processor and terabytes of memory and having the highest performance and cost.

(c) Random access memory.

(d) Central processing unit.

(e) Microprocessor containing several processors in the same chip.

(f) Desktop-class computer without a screen or keyboard and usually accessed via a network.

(g) Computer used for running one predetermined application or collection of applications, often found as part of another piece of hardware.

(h) Personal computer delivering good performance to a single user at low cost.

(i) Program that translates statements in high-level language to assembly language.

(j) Program that translates symbolic instructions to binary instructions

(k) Binary language that a processor can understand.

(l) Command that a processor understands.

(m) Symbolic representation of machine instructions.

(n) Interface between user's program and hardware providing a variety of services and supervision functions.

(o) Software/programs developed by a user.

(p) Binary digit (value 0 or 1).

(q) Software layer between the application software and the hardware that includes the operating system and compilers.

(r) Portable language usually composed of words and algebraic expressions that must be translated into assembly language before being run on a computer.

2. (10 points) Suppose two processors implement a given instruction set architecture, in which the instructions can be grouped into four classes A, B, C, and D. Processor P1 has a clock rate of 2.5GHz, and the CPI (cycles per instruction) for the four classes are 1, 2, 3, and 3 respectively. Processor P2 has a clock rate of 3GHz and a CPI of 2 for all classes. For a program that at runtime executes 1E9 (10^9) instructions, of which 10% are class A, 20% class B, 50% class C, and 20% class D, answer the following:

   (a) How many seconds does this program take on P1? on P2?

   (b) If "global CPI" is an average CPI over all four classes, calculate global CPI for this program on both P1 and P2. (Note here that this average is probably best computed as total cycles divided by total instructions, to allow for the different frequencies of the different classes of instructions.)

3. (10 points) Review the scenario described in problem 1.9 in the textbook (a program being executed on a particular multiprocessor system) and answer the following questions:

   (a) Assuming there is what you might call an original sequential program that just executes all the instructions mentioned in the description as well as a parallel program for which the number of instructions per processor is as described, calculate execution time for the sequential program and for the parallel program running on 1, 2, 4, and 8 processors, and compute speedups *relative to the sequential program* (i.e., sequential time divided by time on n processors).

Suggestion: The calculations here are kind of tedious. If you like programming you might consider writing a short throw-away program to help you, in whatever language appeals to you. If you do, include a copy of your source code; that will be the equivalent of "showing your work". (Note that while normally I prefer to get source code by e-mail, since I don't plan to test anything you write for this problem it will be simpler just to turn in hardcopy.)

(b) What would happen if the clock rate were changed to 2.5GHz but arithmetic instructions now had a CPI of 2? (Recalculate all the times in the first part of the problem and also the speedups.)

(c) For up to 2 extra-credit points, plot the calculated speedups and a line showing what linear speedup would look like. (If you don't already have a favorite program for making plots, ask me about `gnuplot`.)

4. (10 points) Do problems 1.12.1, 1.12.2, and 1.12.3 from the textbook. (Note that you will need information from the paragraph numbered 1.12.)

3 Honor Code Statement

Include the Honor Code pledge or just the word "pledged", plus *at least one of the following* about collaboration and help (as many as apply).[1] Text *in italics* is explanatory or something for you to fill in. For programming assignments, this should go in the body of the e-mail or in a plain-text file `honor-code.txt` (no word-processor files please).

- This assignment is entirely my own work. *(Here, "entirely my own work" means that it's your own work except for anything you got from the assignment itself — some programming assignments include "starter code", for example — or from the course Web site. In particular, for programming assignments you can copy freely from anything on the "sample programs page".)*
- I worked with *names of other students* on this assignment.
- I got help with this assignment from *source of help — ACM tutoring, another student in the course, the instructor, etc. (Here, "help" means significant help, beyond a little assistance with tools or compiler errors.)*
- I got help from *outside source — a book other than the textbook (give title and author), a Web site (give its URL), etc.. (Here too, you only need to mention significant help — you don't need to tell me that you looked up an error message on the Web, but if you found an algorithm or a code sketch, tell me about that.)*
- I provided help to *names of students* on this assignment. *(And here too, you only need to tell me about significant help.)*

4 Essay

Include a brief essay (a sentence or two is fine, though you can write as much as you like) telling me what about the assignment you found interesting, difficult, or otherwise noteworthy. For pro-

[1] Credit where credit is due: I based the wording of this list on a posting to a SIGCSE mailing list. SIGCSE is the ACM's Special Interest Group on CS Education.

gramming assignments, it should go in the body of the e-mail or in a plain-text file `essay.txt` (no word-processor files please).