1. Suppose that we want to enhance the processor used for Web serving. The new processor is 10 times faster on computation in the Web serving application than the original processor. Assuming that the original processor is busy with computation 40% of the time and is waiting for I/O 60% of the time, what is the overall speedup gained by incorporating the enhancement?

2. A common transformation required in graphics processors is square root. Implementations of floating-point (FP) square root vary significantly in performance, especially among processors designed for graphics. Suppose FP square root (FPSQR) is responsible for 20% of the execution time of a critical graphics benchmark. One proposal is to enhance the FPSQR hardware and speed up this operation by a factor of 10. The other alternative is just to try to make all FP instructions in the graphics processor run faster by a factor of 1.6; FP instructions are responsible for half of the execution time for the application. The design team believes that they can make all FP instructions run 1.6 times faster with the same effort as required for the fast square root. Compare these two design alternatives.

3. Suppose we have made the following measurements:

   Frequency of FP operations = 25%

   Average CPI of FP operations = 4.0

   Average CPI of other instructions = 1.33

   Frequency of FPSQR = 2%

   CPI of FPSQR = 20

   Assume that the two design alternatives are to decrease the CPI of FPSQR to 2 or to decrease the average CPI of all FP operations to 2.5. Compare these two design alternatives using the processor performance equation.

4. You are designing a system for a real-time application in which specific deadlines must be met. Finishing the computation faster gains nothing. You find that your system can execute the necessary code, in the worst case, twice as fast as necessary

   a. How much energy do you save if you execute at the current speed and turn off the system when the computation is complete?

   b. How much energy do you save if you set the voltage and frequency to be half as much?

5. In this exercise, assume that we are considering enhancing a machine by adding vector hardware to it. When a computation is run in vector mode on the vector hardware, it is 10 times faster than the normal mode of execution. We call the

percentage of time that could be spent using vector mode the percentage of vectorization. Vectors are discussed in Chapter 4, but you don't need to know anything about how they work to answer this question!

    a. Draw a graph that plots the speedup as a percentage of the computation performed in vector mode. Label the y-axis "Net speedup" and label the x-axis "Percent vectorization."

    b. What percentage of vectorization is needed to achieve a speedup of 2?

    c. What percentage of the computation run time is spent in vector mode if a speedup of 2 is achieved?

    d. What percentage of vectorization is needed to achieve one-half the maximum speedup attainable from using vector mode? e. [15] <1.9> Suppose you have measured the percentage of vectorization of the program to be 70%. The hardware design group estimates it can speed up the vector hardware even more with significant additional investment. You wonder whether the compiler crew could increase the percentage of vectorization, instead. What percentage of vectorization would the compiler team need to achieve in order to equal an addition 2× speedup in the vector unit (beyond the initial 10×)?

6. Assume that we make an enhancement to a computer that improves some mode of execution by a factor of 10. Enhanced mode is used 50% of the time, measured as a percentage of the execution time when the enhanced mode is in use. Recall that Amdahl's law depends on the fraction of the original, unenhanced execution time that could make use of enhanced mode. Thus, we cannot directly use this 50% measurement to compute speedup with Amdahl's law.

    a. What is the speedup we have obtained from fast mode?

    b. What percentage of the original execution time has been converted to fast mode?

7. When making changes to optimize part of a processor, it is often the case that speeding up one type of instruction comes at the cost of slowing down something else. For example, if we put in a complicated fast floating point unit, that takes space, and something might have to be moved farther away from the middle to accommodate it, adding an extra cycle in delay to reach that unit. The basic Amdahl's law equation does not take into account this trade-off.

    a. If the new fast floating-point unit speeds up floating-point operations by, on average, 2×, and floating-point operations take 20% of the original program's execution time, what is the overall speedup (ignoring the penalty to any other instructions)?

    b. Now assume that speeding up the floating-point unit slowed down data cache accesses, resulting in a 1.5× slowdown (or 2/3 speedup). Data cache accesses consume 10% of the execution time. What is the overall speedup now?

    c. After implementing the new floating-point operations, what percentage of execution time is spent on floating-point operations? What percentage is spent on data cache accesses?

8. Your company has just bought a new Intel Core i5 dual-core processor, and you have been tasked with optimizing your software for this processor. You will run two applications on this dual core, but the resource requirements are not equal. The first application requires 80% of the resources, and the other only 20% of the resources. Assume that when you parallelize a portion of the program, the speedup for that portion is 2.

    a. Given that 40% of the first application is parallelizable, how much speedup would you achieve with that application if run in isolation?

    b. Given that 99% of the second application is parallelizable, how much speedup would this application observe if run in isolation?

    c. Given that 40% of the first application is parallelizable, how much overall system speedup would you observe if you parallelized it?

    d. Given that 99% of the second application is parallelizable, how much overall system speedup would you observe if you parallelized it?

9. When parallelizing an application, the ideal speedup is speeding up by the number of processors. This is limited by two things: percentage of the application that can be parallelized and the cost of communication. Amdahl's law takes into account the former but not the latter.

    a. What is the speedup with N processors if 80% of the application is parallelizable, ignoring the cost of communication?

    b. What is the speedup with 8 processors if, for every processor added, the communication overhead is 0.5% of the original execution time.

    c. What is the speedup with 8 processors if, for every time the number of processors is doubled, the communication overhead is increased by 0.5% of the original execution time?

    d. What is the speedup with N processors if, for every time the number of processors is doubled, the communication overhead is increased by 0.5% of the original execution time?

    e. Write the general equation that solves this question: What is the number of processors with the highest speedup in an application in which P% of the original execution time is parallelizable, and, for every time the number of processors is doubled, the communication is increased by 0.5% of the original execution time?