UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
PROGRAMA DE PÓS GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
Curso: Doutorado Interinstitucional
Disciplina: Arquitetura de Computadores
1o Semestre de 2011
Página do Curso: http://www.verlab.dcc.ufmg.br/cursos/arquitetura/2011-1/index
Professor: Mario Fernando Montenegro Campos (mario@dcc.ufmg.br)

Lista de Exercícios 2

Os exercícios são do livro: John Hennessy and David Patterson. *Computer Architecture – A Quantitative Approach*, 3rd. Edition.

Leia detalhadamente as seções Fallacies *and Pitfalls, Concluding Remarks* e *Historical Perspective and References*.

1.1 [20/10/10/15] <1.6> In this exercise, assume that we are considering enhancing a machine by adding a vector mode to it. When a computation is run in vector mode it is 20 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 Appendix B, but you don't need to know anything about how they work to answer this question!
a. [20] <1.6> 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. [10] <1.6> What percentage of vectorization is needed to achieve a speedup of 2?

c. [10] <1.6> What percentage of vectorization is needed to achieve one-half the maximum speedup attainable from using vector mode?

d. [15] <1.6> Suppose you have measured the percentage of vectorization for programs to be 70%. The hardware design group says they can double the speed of the vector rate with a significant additional engineering investment. You wonder whether the compiler crew could increase the use of vector mode as another approach to increasing performance. How much of an increase in the percentage of vectorization (relative to current usage) would you need to obtain the same performance gain? Which investment would you recommend?

1.2 [15/10] <1.6> Assume—as in the Amdahl's Law Example on page 41—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. [15] <1.6> What is the speedup we have obtained from fast mode?

b. [10] <1.6> What percentage of the original execution time has been converted to fast mode?

1.5 [15] <1.6> Suppose we are considering a change to an instruction set. The base machine initially has only loads and stores to memory, and all operations work on the registers. Such machines are called *load-store* machines (see Chapter 2). Measurements of the loadstore machine showing the *instruction mix* and clock cycle counts per instruction are given in Figure 1.32 on page 69.
Let's assume that 25% of the *arithmetic logic unit* (ALU) operations directly use a loaded operand that is not used again.

We propose adding ALU instructions that have one source operand in memory. These new *register-memory instructions* have a clock cycle count of 2. Suppose that the extended instruction set increases the clock cycle count for branches by 1, but it does not affect the clock cycle time. (Chapter 3, on pipelining, explains why adding register-memory instructions might slow down branches.) Would this change improve CPU performance?

1.6 [15] <1.7> Assume that we have a machine that with a perfect cache behaves as given in Figure 1.32.
With a cache, we have measured that instructions have a miss rate of 5%, data references have a miss rate of 10%, and the miss penalty is 40 cycles. Find the CPI for each instruction type with cache misses and determine how much faster the machine is with no cache misses versus with cache misses.

1.7 [20] <1.6> After graduating, you are asked to become the lead computer designer at Hyper Computers, Inc. Your study of usage of high-level language constructs suggests that procedure calls are one of the most expensive operations. You have invented a scheme that reduces the loads and stores normally associated with procedure calls and returns. The first thing you do is run some experiments with and without this optimization. Your experiments use the same state-of-the-art optimizing compiler that will be used with either version of the computer. These experiments reveal the following information:

- The clock rate of the unoptimized version is 5% higher.
- Thirty percent of the instructions in the unoptimized version are loads or stores.
- The optimized version executes two-thirds as many loads and stores as the unoptimized version. For all other instructions the dynamic execution counts are unchanged.
- All instructions (including load and store) take one clock cycle.
-

Which is faster? Justify your decision quantitatively.

1.13 [15/15/15] <1.6,1.9> Three enhancements with the following speedups are proposed for a new architecture:

$\text{Speedup}_1 = 30$
$\text{Speedup}_2 = 20$
$\text{Speedup}_3 = 10$

Only one enhancement is usable at a time.
a. [15] <1.6> If enhancements 1 and 2 are each usable for 30% of the time, what fraction of the time must enhancement 3 be used to achieve an overall speedup of 10?

b. [15] <1.6,1.9> Assume the distribution of enhancement usage is 30%, 30%, and 20% for enhancements 1, 2, and 3, respectively. Assuming all three enhancements are in use, for what fraction of the reduced execution time is no enhancement in use?

c. [15] <1.6> Assume for some benchmark, the fraction of use is 15% for each of enhancements 1 and 2 and 70% for enhancement 3. We want to maximize performance.
If only one enhancement can be implemented, which should it be? If two enhancements can be implemented, which should be chosen?