COMPUTER ARCHITECTURE
CS2010

Faculty of Computer Science and Engineering
Department of Computer Engineering

Nam Ho

Chapter 1

Computer Abstraction and Technology - Exercises

Adapted from *Computer Organization and Design*, *4th Edition*, Patterson & Hennessy, © 2008

Pitfall: MIPS as a Performance Metric

- MIPS: Millions of Instructions Per Second
  - Doesn't account for
    - Differences in ISAs between computers
    - Differences in complexity between instructions

$$\text{MIPS} = \frac{\text{Instruction count}}{\text{Execution time} \times 10^6}$$

$$= \frac{\text{Instruction count}}{\frac{\text{Instruction count} \times \text{CPI}}{\text{Clock rate}} \times 10^6} = \frac{\text{Clock rate}}{\text{CPI} \times 10^6}$$

- CPI varies between programs on a given CPU

Using MIPS

- There are three problems with MIPS:
  - MIPS specifies the instruction execution rate but not the capabilities of the instructions
  - MIPS varies between programs on the same computer
  - MIPS can vary inversely with performance (see next example)

Exercise 1

- *Consider the machine with the following three instruction classes and CPI:*

	CPI for this instruction class		
	A	B	C
CPI	1	2	3

- *Now suppose we measure the code for the same program from two different compilers and obtain the following data:*

Code from	Instruction count in (billons) for each instruction class		
	A	B	C
Compiler 1	5	1	1
Compiler 2	10	1	1

- *Assume that the machine's clock rate is 500 MHz. Which code sequence will execute faster according to execution time ? According to MIPS?*

Exercise 1

- Using the formula: CPU clock cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

Execution time = CPU clock cycles/clock rate

- Compiler 1:
  - CPU clock cycles = (5 x 1 + 1 x 2 + 1 x 3) x 10^9 = 10 x 10^9 cycles
  - Execution time = (10 x 10^9)/500 x 10^6 = 20 seconds
- Compiler 2:
  - CPU clock cycles = (10 x 1 + 1 x 2 + 1 x 3) x 10^9 = 15 x 10^9 cycles
  - Execution time = (15 x 10^9)/(500 x 10^6) = 30 seconds
- Therefore compiler 1 generates a faster program

Exercise 1

$$MIPS = \frac{Instruction\ count}{Execution\ Time \times 10^6}$$

- Compiler 1:

$$MIPS = \frac{(5+1+1)\times 10^9}{20\times 10^6} = 350$$

- Compiler 2:

$$MIPS = \frac{(10+1+1)\times 10^9}{30\times 10^6} = 400$$

- Although compiler 2 has a higher MIPS rating, the code from generated by compiler 1 runs faster

Exercise 2

- PowerPC G4 *can* execute 4 instruction/cycle with a clock rate 867 MHz. What is the MIPS rate ?
- Answer:
  - CPI = 1/4
  - MIPS = 1/(CPI x Cycle Time x 10^6)
  - MIPS = Clock Rate/CPI x 10^6
  - MIPS = 4 x 867 = 3468

Speedup - Amdahl's Law

- $Speedup = \frac{Execution\ Time_{Old}}{Execution\ Time_{New}} = \frac{T_{before\ improved}}{T_{improved}}$
- $Speedup = \frac{1}{1 - f_{enhanced} + f_{enhanced}/n}$, *n is the improvement factor*
- $f_{enhanced} = \frac{T_{affected}}{T_{before\ improved}}$, $T_{before\ improved} = T_{affected} + T_{unaffected}$

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

Exercise 3

- Consider the implementation of a ISA. The clock rate is 2Ghz. What is the execution time of the following program ?

	Arith	Store	Load	Branch
Instructions	500	50	100	50
CPI	1	5	5	2

- Execution Time = (500 × 1 + 50 × 5 + 100 × 5 + 50 × 2) × 0.5 × 10^{-9} = 675 ns
- If the number of load instructions can be reduced by one-half, what is the speedup and the CPI?
- Execution Time = (500 × 1 + 50 × 5 + 50 × 5 + 50 × 2) × 0.5 × 10^{-9} = 550 ns
- Speedup = 675/550 = 1.22
- CPI = Execution Time x Clock rate/ Instruction Count
- CPI = 550 x 10^{-9} x 2 x 10^{9}/700 = 1.57

Exercise 4

- $Speedup = \frac{Execution\ Time_{Old}}{Execution\ Time_{New}} = \frac{T_{before\ improved}}{T_{improved}}$

- *Prove that* $Speedup = \frac{1}{1 - f_{enhanced} + f_{enhanced}/n}$
- *n is the improvement factor*
- $f_{enhanced} = \frac{T_{affected}}{T_{before\ improved}}$
- $T_{before\ improved} = T_{affected} + T_{unaffected}$

$$T_{improved} = \frac{T_{affected}}{improvement\ factor} + T_{unaffected}$$

Exercise 5

- 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. Compare these two design alternatives.

Exercise 5

- $\text{Speedup}_{\text{FPSQR}} = \dfrac{1}{(1-0.2)+\dfrac{0.2}{10}} = \dfrac{1}{0.82} = 1.22$

- $\text{Speedup}_{\text{FP}} = \dfrac{1}{(1-0.5)+\dfrac{0.5}{1.6}} = \dfrac{1}{0.8125} = 1.23$

Geometric Mean

$$GM = \sqrt[n]{\prod_{i=1}^{n} \text{Execution time ratio}_i}$$

- The advantage of the geometric mean is that
  - It is independent of the running times of the individual programs
  - And it doesn't matter which computer is used for normalization.
- The drawback to using geometric means of execution times is that
  - They violate our fundamental principle of performance measurement, they do not predict execution time

Geometric Mean

	A	B	C
Program P1 (secs)	1.00	10.00	20.00
Program P2 (secs)	1000.00	100.00	20.00

	Normalized to A			Normalized to B			Normalized to C		
	A	B	C	A	B	C	A	B	C
Program P1	1.0	10.0	20.0	0.1	1.0	2.0	0.05	0.5	1.0
Program P2	1.0	0.1	0.02	10.0	1.0	0.2	50.0	5.0	1.0
Arithmetic mean	1.0	5.05	10.01	5.05	1.0	1.1	25.03	2.75	1.0
Geometric mean	1.0	1.0	0.63	1.0	1.0	0.63	1.58	1.58	1.0
Total time	1.0	0.11	0.04	9.1	1.0	0.36	25.03	2.75	1.0

- A and B have the same performance.
- Performance of C is 0.63 of A or B.
- Unfortunately, the total execution time of A is 10 times longer than that of B, and B in turn is about 3 times longer than C.

Exercise 6

- Show that the Geometric Mean doesn't matter which computer is used for normalization.

$$\frac{\text{Geometric mean}_A}{\text{Geometric mean}_B} = \frac{\sqrt[n]{\prod_{i=1}^{n} \text{SPECRatio A}_i}}{\sqrt[n]{\prod_{i=1}^{n} \text{SPECRatio B}_i}} = \sqrt[n]{\prod_{i=1}^{n} \frac{\text{SPECRatio A}_i}{\text{SPECRatio B}_i}}$$

$$= \sqrt[n]{\prod_{i=1}^{n} \frac{\dfrac{\text{Execution time}_{\text{reference}_i}}{\text{Execution time}_{A_i}}}{\dfrac{\text{Execution time}_{\text{reference}_i}}{\text{Execution time}_{B_i}}}} = \sqrt[n]{\prod_{i=1}^{n} \frac{\text{Execution time}_{B_i}}{\text{Execution time}_{A_i}}} = \sqrt[n]{\prod_{i=1}^{n} \frac{\text{Performance}_{A_i}}{\text{Performance}_{B_i}}}$$

Fallacy: MFLOPS

- MFLOPS: *Million Floating-point Operations Per Second*
- Fallacy: MFLOPS is a consistent and useful measure of performance.
- For example, the Cray C90 has no divide instruction, while the Intel Pentium has divide, square root, sine, and cosine

$$\text{MFLOPS} = \frac{\text{Number of floating-point operations in a program}}{\text{Execution time} \times 10^6}$$

Exercise 7

- Consider the programs in the following table, running on a processor with clock rate = 3GHz. Find the MFLOPS for the program a, b.

	Instr. count	L/S instr.	FP instr.	Branch Instr.	CPI(L/S)	CPI(FP)	CPI(Branch)
a.	10^6	50%	40%	10%	0.75	1	1.5
b.	3×10^6	40%	40%	20%	1.25	0.70	1.25

- **Program a:**
  - $FP_{op} = 10^6 \times 0.4 = 4 \times 10^5$, $clockcyles_{fp}$ = CPI × No. FP instr. = 4×10^5
  - $T_{fp} = 4 \times 10^5 \times 0.33 \times 10^{-9} = 1.32 \times 10^{-4}$ then MFLOPS = 3.03×10^3
- **Program b:**
  - $FP_{op} = 3 \times 10^6 \times 0.4 = 1.2 \times 10^6$, $clockcyles_{fp}$ = CPI × No. FP instr. = $0.70 \times 1.2 \times 10^6$
  - $T_{fp} = 0.84 \times 10^6 \times 0.33 \times 10^{-9} = 2.77 \times 10^{-4}$ then MFLOPS = 4.33×10^3