CENG 3420
Computer Organization & Design
Lecture 11: Performance

Textbook: Chapter 1.6, 1.9

Zhengrong Wang

CSE Department, CUHK

zhengrongwang@cuhk.edu.hk

Performance Metrics

Performance Metrics

- Purchasing perspective: Given a collection of machines, which has the
  - Best performance?
  - Lowest cost?
  - Best performance/cost?
- Design perspective: Given some design choices, which has the
  - Best performance?
  - Lowest cost?
  - Best performance/cost?
- Both requires performance metrics for comparison.

- Goal: Understand what factors in the architecture contribute to overall system performance and the relative importance (and cost) of these factors.

Throughput vs. Response Time

- Response time (execution time, latency)
  - The time between the start and the completion of a task.
  - Important to individual users.
- Throughput (bandwidth)
  - The total amount of work done in a given time.
  - Important to data center managers.
- Example: Airplanes

Airplane	Passenger capacity	Cruising range (miles)	Cruising speed (m.p.h.)	Passenger throughput (passengers × m.p.h.)
Boeing 737	240	3000	564	135,360
BAC/Sud Concorde	132	4000	1350	178,200
Boeing 777-200LR	301	9395	554	166,761
Airbus A380-800	853	8477	587	500,711

Defining Performance

- To maximize performance, need to minimize execution time.

$$performance_X = \frac{1}{execution\_time_X}$$

- If X is n times faster than Y, then

$$\frac{performance_X}{performance_Y} = \frac{execution\_time_Y}{execution\_time_X} = n$$

- Decreasing response time almost always improves throughput.

- Ex: If computer A runs a program in 10 seconds and computer B runs the same program in 15 seconds, how much faster is A than B?

Performance Factors

- CPU execution time (CPU time): time the CPU spends working on a task.
- Does not include time waiting for I/O or running other programs.

$$CPU\ execution\ time = CPU\ clock\ cycles \times clock\ cycle\ time$$
$$= \frac{CPU\ clock\ cycles}{CPU\ clock\ rate}$$

  - Clock rate is the inverse of clock cycle time, e.g., 1ns clock cycle time is 1GHz.

- Can improve performance by reducing.
  - Length of the clock cycle.
  - Number of clock cycles required for a program

Example of Improving Performance

A program runs on computer A with a 2 GHz clock in 10 seconds. What clock rate must a computer B must run this program in 6 seconds? Unfortunately, to accomplish this, computer B will require 1.2 times as many clock cycles as computer A to run the program.

- To finish in 6s, $10s \times 2GHz \div 6s = 3.33GHz$
- Considering 1.2 times more cycles, $10s \times 2GHz \times 1.2 \div 6s = 4GHz$

Cycles per Instruction (CPI)

- Not all instructions take the same amount of time to execute.
- But we can take the average:

$$CPU\ Clock\ Cycles = \#instruction \times clock\ cycle\ per\ instruction$$

- Clock cycles per instruction (CPI):
  - The average number of clock cycles each instruction takes to execute.
  - A way to compare two different implementations of the same ISA

Average CPI

- We can average across different class of instruction to refine the estimation.
  - E.g., arithmetic, branch, memory access.
  - Different class usually takes different number of cycles (recall our pipeline design).

$$CPI = \sum_{i=1}^{n} CPI_i \times IC_i$$

  - Across n class of instructions, IC_i is the percentage of instructions within ith class.

- Computing the overall effective CPI is done by looking at the different types of instructions and their individual cycle counts and averaging

Basic Performance Equation

$$CPU\ time = Instruction\ count \times CPI \times clock\ cycle$$

$$CPU\ time = \frac{Instruction\ count \times CPI}{clock\ rate}$$

- Discussions about the three key factors
  - instruction count: can be measured by using profilers/ simulators without knowing all of the implementation details.
  - CPI: varies by instruction type and ISA implementation for which we must know the implementation details.
  - clock rate: is usually given.

Practice with Performance Equation

- Computers A and B implement the same ISA. Computer A has a clock cycle time of 250 ps and an effective CPI of 2.0 for some program and computer B has a clock cycle time of 500 ps and an effective CPI of 1.2 for the same program. Which computer is faster and by how much?
  - A: $2.0 \times 250ps = 500ps$
  - B: $1.2 \times 500ps = 600ps$
  - A is faster by $600ps \div 500ps = 1.2 \times$

Determines the Performance

$$CPU\ time = Instruction\ count \times CPI \times clock\ cycle$$

	Instruction count	CPI	Clock cycle
Algorithm			
Programming Language			
Compiler			
ISA			
Core organization			
Technology			

Determines the Performance

$$CPU\ time = Instruction\ count \times CPI \times clock\ cycle$$

	Instruction count	CPI	Clock cycle
Algorithm	X	X	
Programming Language	X	X	
Compiler	X	X	
ISA	X	X	X
Core organization		X	X
Technology			X

Practice

Op	Freq	CPI	Freq x CPI
ALU	50%	1	
Load	20%	5	
Store	10%	3	
Branch	20%	2	
			CPI=

- How much faster would the machine be if a better data cache reduced the average load time to 2 cycles?
- How does this compare with using branch prediction to shave a cycle off the branch time?
- What if two ALU instructions could be executed at once?

Workloads and Benchmarks

- Benchmark: A set of programs that form a "workload" specifically chosen to measure performance.
  - SPEC (System Performance Evaluation Cooperative) is an effort funded and supported by several computer vendors to create standard sets of benchmarks for modern computer systems. First edition created in 1989.
  - Latest 2017 edition contains 10 integer workloads and 13 floating-point workloads.

- There are many different benchmarks for different scenarios:
  - MLPerf for AI workloads.
  - SPEC Cloud IaaS 2018 for cloud service.

SPECspeed 2017 Integer on Intel Xeon

Description	Name	Instruction Count x 10^9	CPI	Clock cycle time (seconds x 10^{-9})	Execution Time (seconds)	Reference Time (seconds)	SPECratio
Perl interpreter	perlbench	2684	0.42	0.556	627	1774	2.83
GNU C compiler	gcc	2322	0.67	0.556	863	3976	4.61
Route planning	mcf	1786	1.22	0.556	1215	4721	3.89
Discrete Event simulation - computer network	omnetpp	1107	0.82	0.556	507	1630	3.21
XML to HTML conversion via XSLT	xalancbmk	1314	0.75	0.556	549	1417	2.58
Video compression	x264	4488	0.32	0.556	813	1763	2.17
Artificial Intelligence: alpha-beta tree search (Chess)	deepsjeng	2216	0.57	0.556	698	1432	2.05
Artificial Intelligence: Monte Carlo tree search (Go)	leela	2236	0.79	0.556	987	1703	1.73
Artificial Intelligence: recursive solution generator (Sudoku)	exchange2	6683	0.46	0.556	1718	2939	1.71
General data compression	xz	8533	1.32	0.556	6290	6182	0.98
Geometric mean	–	–	–	–	–	–	2.36

FIGURE 1.18 SPECspeed 2017 Integer benchmarks running on a 1.8 GHz Intel Xeon E5-2650L. As the equation on page 35 explains, execution time is the product of the three factors in this table: instruction count in billions, clocks per instruction (CPI), and clock cycle time in nanoseconds. SPECratio is simply the reference time, which is supplied by SPEC, divided by the measured execution time. The single number quoted as SPECspeed 2017 Integer is the geometric mean of the SPECratios. SPECspeed 2017 has multiple input files for perlbench, gcc, x264, and xz. For this figure, execution time and total clock cycles are the sum running times of these programs for all inputs.

Comparing and Summarizing Performance

- How to summarize performance with a single number?
  - First the execution times are normalized given the "SPEC ratio" (bigger is faster, i.e., SPEC ratio is the inverse of execution time).
  - SPEC ratios are "averaged" using the geometric mean (GM).

$$GM = \sqrt[n]{\prod_{i=1}^{n} SPEC\ ratio_i}$$

- Guiding principle – reproducibility.
  - List everything another experimenter would need to duplicate the experiment: version of the operating system, compiler settings, input set used, specific computer configuration (clock rate, cache sizes and speed, memory size and speed, etc.)