Overview

- Defining performance
  - Response time
  - Throughput
  - Relative performance
- Measuring performance
- CPU performance and its factors
  - Improving performance
  - Performance equation
  - Program performance
Performance

- Measure, Report, and Summarize
- Make intelligent choices; see through the marketing hype
- Key to understanding underlying organizational motivation
  - Why is some hardware better than others for different programs?
  - What factors of system performance are hardware related? (e.g., Do we need a new machine, or a new operating system?)
  - How does the machine's instruction set affect performance?

Defining Performance
Airplane Example

<table>
<thead>
<tr>
<th>Airplane</th>
<th>Passengers</th>
<th>Range (mi)</th>
<th>Speed (mph)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Boeing 777</td>
<td>375</td>
<td>4630</td>
<td>610</td>
</tr>
<tr>
<td>Boeing 747</td>
<td>470</td>
<td>4150</td>
<td>610</td>
</tr>
<tr>
<td>BAC/Sud Concorde</td>
<td>132</td>
<td>4000</td>
<td>1350</td>
</tr>
<tr>
<td></td>
<td>146</td>
<td>8720</td>
<td>544</td>
</tr>
</tbody>
</table>

Which of these airplanes has the best performance? Which of these airplanes has the best passenger throughput?
Computer Performance

- **Response Time (or execution time)**
  - “the time between the start and the completion of a task”
  - How long does it take for a job to run?
  - How long must I wait for the database query?

- **Throughput**
  - “the total amount of work done in a given time”
  - How many jobs can the machine run at once?
  - How much work is getting done?

**Q1. If we upgrade a machine with a new processor, what do we improve?**

**Q2. If we add an additional processor to a system, what do we improve?**

---

Performance - Execution Time

- **Elapsed Time (response or execution time)**
  - Total time to complete a task → counts everything (other processes running, disk and memory accesses, I/O, CPU time, operating system overhead...)

- **CPU time**
  - doesn’t count waiting for I/O or time spent running other programs
  - can be broken up into system time and user time

- **User CPU time**
  - time spent in a user program

- **System CPU time**
  - time spent in operating system performing tasks on behalf of the program.
CPU Performance

We will use “n times faster”, which means both increased performance and decreased execution time.

- For some program running on machine X, \( \text{Performance}_X = 1 / \text{Execution time}_X \)
- "X is n times faster than Y"  
  \( \text{Performance}_X / \text{Performance}_Y = n \)

CPU Performance – problem 1

Machine A runs a program in 20 seconds and machine B runs the same program in 25 seconds. How much faster is A than B?
CPU Performance – problem 2

Computer C’s performance is 4 times better than the performance of computer B, which runs a given application in 28 seconds.

How long will computer C take to run that application?

Clock Cycles

- Instead of reporting execution time in seconds, we often use cycles: \( \frac{\text{seconds}}{\text{program}} \times \frac{\text{cycles}}{\text{program}} = \frac{\text{cycles}}{\text{cycle}} \)

- Clock ticks indicate when to start activities:

<table>
<thead>
<tr>
<th>Program execution order (in instructions)</th>
<th>Time (in clock cycles)</th>
</tr>
</thead>
<tbody>
<tr>
<td>lw $10, $20($1)</td>
<td></td>
</tr>
<tr>
<td>sub $11, $2, $3</td>
<td></td>
</tr>
</tbody>
</table>

  instruction fetch | instruction decode | execution | data access | write back |

  instruction fetch | instruction decode | execution | data access | write back |

  cycle time = \( \frac{\text{seconds}}{\text{cycles per cycle}} \)

- clock rate (frequency) = cycles per second \( (1 \text{ Hz} = 1 \text{ cycle/sec}) \)
  A 200 MHz. clock has a ______ cycle time.
How to improve performance

\[
\text{seconds} = \frac{\text{cycles}}{\text{program}} \times \frac{\text{seconds}}{\text{cycle}} = \frac{\text{cycles/program}}{\text{clock rate}}
\]

- So, to improve performance (everything else being equal) one can either
  
  ________ the # of required cycles for a program, or
  
  ________ the clock cycle time or, said another way,
  
  ________ the clock rate.

Performance - Word problem!

- Our favorite program runs in 10 seconds on computer A, which has a 4 GHz clock. We are trying to help a computer designer build a new machine B, that will run this program in 6 seconds. The designer can use new (or perhaps more expensive) technology to substantially increase the clock rate, but has informed us that this increase will affect the rest of the CPU design, causing machine B to require 1.2 times as many clock cycles as machine A for the same program.

- What clock rate should we tell the designer to target?
How many cycles are required for a program?

- Could we assume that # of cycles = # of instructions?

This assumption is incorrect; different instructions take different amounts of time depending on what they do.


**Review - Different instructions take different number of cycles**

Figure 6.3 (modified)

Program execution order (in instructions)

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Reg</th>
<th>ALU</th>
<th>Reg</th>
</tr>
</thead>
<tbody>
<tr>
<td>beq $1, $2, 40</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>lw $3, 300(30)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>add $4, $5, 36</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Note that “lw” is longer than “add” and “beq”, why?
- lw~800 ps, R-type~600 ps, beq~500ps [Fig. 6.2 in book]
- What is the worst-case clock cycle in this pipelined execution?

**CPI - Cycles per Instruction**

- Number of clock cycles to execute an instruction
- Average CPI for a program can be computed:

\[
\text{CPI}_{\text{ave}} = \frac{\text{CPU clock cycles}}{\text{instruction count}}
\]

- Average CPI is useful for comparing two implementations of the same ISA since the instruction count for a program will be the same.
Performance
Review some ideas and terms...

- A given program will require
  - some number of instructions (machine instructions)
  - some number of cycles
  - some number of seconds

- The vocabulary that relates these quantities:
  - cycle time (seconds per cycle)
  - clock rate (cycles per second)
  - CPI (cycles per instruction)
  - MIPS (million of instructions per second)
    - MIPS (Microprocessor without Interlocked Piped Stages)
    - SPIM (Software that runs MIPS code)

Performance Question...

- **Performance is determined by execution time only.**

- Do any of the other variables equal performance? If the answer is yes, which one(s)?
  - # of cycles to execute program?
  - # of instructions in program?
  - # of cycles per second?
  - average # of cycles per instruction?
  - average # of instructions per second?

- Common pitfall: thinking only one of the above variables is indicative of performance.
Performance

\[
\text{CPU time} = \frac{\text{cycles}}{\text{program}} \times \frac{\text{seconds}}{\text{cycle}} = \frac{\text{cycles/program}}{\text{clock rate}}
\]

- **In terms of instructions:**

\[
\text{CPU time} = \text{instruction count} \times \frac{\text{clock cycles}}{\text{instruction}} \times \frac{\text{seconds}}{\text{clock cycles}}
\]

\[
\text{CPU time} = \frac{\text{instruction count} \times \text{CPI}}{\text{clock rate}}
\]

\[
\text{CPU time} = \frac{\text{clock cycles}}{\text{clock rate}}
\]

MIPS as a Performance Measure

- MIPS is an instruction execution rate
- MIPS is Million Instructions Per Second
- MIPS is easy to understand

\[
\text{MIPS} = \frac{\text{instruction count}}{\text{CPU time} \times 10^6}
\]

- **Problems with using MIPS as a measure for comparing computers:**
  - Cannot use MIPS to compare two machines with different ISA’s!
  - MIPS varies between programs on same computer
  - MIPS can vary inversely with performance!
Performance

CPI example 1

- Suppose we have two implementations of the same instruction set architecture (ISA). For some program, Machine A has a clock rate of 4 GHz and a CPI of 2.0. Machine B has a clock rate of 2 GHz and a CPI of 1.2.

- What machine is faster for this program, and by how much?
Performance
CPI example 2

Two computers, C1 and C2, have the following metrics when running the same program. Which computer is faster? Which has higher MIPS?

<table>
<thead>
<tr>
<th></th>
<th>Computer C1</th>
<th>Computer C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Instructions #</td>
<td>10 billion</td>
<td>8 billion</td>
</tr>
<tr>
<td>Clock Rate</td>
<td>4 GHz</td>
<td>4 GHz</td>
</tr>
<tr>
<td>CPI</td>
<td>1.0</td>
<td>1.1</td>
</tr>
</tbody>
</table>

Performance
CPI example 2 - solution
Performance
In terms of instructions...

- CPU time = Instruction count x CPI x Clock cycle time
  or
- CPU time = Instruction count x CPI / Clock rate

- What about instructions with different cycle count?

Performance
In terms of instruction count...

CPU clock cycles = \( \sum_{i=1}^{n} (CPI_i \times C_i) \)

Where “C” is the count of the number of instructions of class “i” executed and “n” is the number of different instruction classes.
Performance example
In terms of instruction count...

A compiler designer is trying to decide between two code sequences for a particular machine. Based on the hardware implementation, there are three different classes of instructions: Class A, Class B, and Class C, and they require one, two, and three CPI (respectively).

The first code sequence has 5 instructions:
- 2 of A, 1 of B, and 2 of C

The second sequence has 6 instructions:
- 4 of A, 1 of B, and 1 of C.

Which sequence will be faster? How much? What is the CPI for each sequence?

Performance in terms of instruction count... Solution

\[ \text{CPU clock cycles} = \sum_{i=1}^{n} (CPI_i \times C_i) \]

Where “C” is the count of the number of instructions of class “i” executed and “n” is the number of different instruction classes.

\[
\begin{align*}
\text{CPU clock cycles}_1 &= (1 \times 2) + (2 \times 1) + (2 \times 3) = 10 \text{ cycles} \\
\text{CPU clock cycles}_2 &= (1 \times 4) + (2 \times 1) + (3 \times 1) = 9 \text{ cycles}
\end{align*}
\]

Sequence 2 is faster.

\[
\begin{align*}
\text{CPI}_1 &= \frac{\text{CPU clock cycles}_1}{\text{Instruction count}_1} = \frac{10}{5} = 2 \\
\text{CPI}_2 &= \frac{\text{CPU clock cycles}_2}{\text{Instruction count}_2} = \frac{9}{6} = 1.5
\end{align*}
\]
Performance
Amdahl's Law

Pitfall: Expecting the improvement of one aspect of a computer to increase performance by an amount proportional to the size of the improvement!

\[
\text{Execution Time After Improvement} = \\
\text{Execution Time Unaffected} + \left( \frac{\text{Execution Time Affected}}{\text{Amount of Improvement}} \right)
\]

Example (pg. 267):
“Suppose a program runs in 100 seconds on a machine, with multiply operation responsible for 80 seconds of this time. How much do we have to improve the speed of multiplication if we want the program to run five times faster?”

Performance
Remember...

- Performance is specific to a particular program(s)
  - Total execution time is a consistent summary of performance

- For a given architecture performance increases come from:
  - increases in clock rate (without adverse CPI effects)
  - improvements in processor organizations that lower CPI
  - compiler enhancements that lower CPI and/or instruction count
Performance
Aspects that affect it...

<table>
<thead>
<tr>
<th>CPU time = Seconds</th>
<th>= Instructions x Cycles x Seconds</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program</td>
<td>Program</td>
</tr>
<tr>
<td></td>
<td>Instruction</td>
</tr>
<tr>
<td></td>
<td>Instruction/Program</td>
</tr>
<tr>
<td>instr count</td>
<td>CPI</td>
</tr>
<tr>
<td>clock rate</td>
<td></td>
</tr>
<tr>
<td>Program</td>
<td>X</td>
</tr>
<tr>
<td>Compiler</td>
<td>X</td>
</tr>
<tr>
<td>Instr. Set</td>
<td>X</td>
</tr>
<tr>
<td>Organization</td>
<td>X</td>
</tr>
<tr>
<td>Technology</td>
<td>X</td>
</tr>
</tbody>
</table>

The performance of software systems is dramatically affected by how well software designers understand the basic hardware technologies at work in a system. Similarly, hardware designers must understand the far-reaching effects their design decisions have on software applications.

They need to work together...
Hypothetical news release

Question 4-51 pg. 277

“The company will unveil the industry’s first 5 GHz version of the chip, which offers a 25% performance boost over the company’s former speed champ, which runs at 4 GHz...”

Comment on the definition of performance that you believe the company used. Do you think the news release is misleading?

---

Review

Performance Equations

- For some program running on machine X,
  \[ \text{Performance}_X = \frac{1}{\text{Execution time}_X} \]

- “X is n times faster than Y” (performance ratio)
  \[ \frac{\text{Performance}_X}{\text{Performance}_Y} = n \]

- Execution time for a program using clock rate:
  \[ \text{CPU Time} = \frac{\text{Clock Cycles}}{\text{Clock Rate}} \]

- Execution time for a program using cycle time:
  \[ \text{CPU Time} = \text{Clock Cycles} \times \text{Cycle Time} \]
Review
More Performance Equations

- Cycles per instruction (CPI) average for a program
  \[ \text{CPI}_{\text{ave}} = \frac{\text{CPU clock cycles}}{\text{instruction count}} \]

- Execution time in terms of instruction count & CPI
  \[ \text{CPU Time}_{\text{prog}} = \text{Instr. Count}_{\text{prog}} \times \text{CPI} \times \text{Cycle Time} \]
  \[ \text{CPU Time}_{\text{prog}} = \frac{\text{Instr. Count}_{\text{prog}} \times \text{CPI}}{\text{Clock Rate}} \]

- Number of clock cycles for variable length instructions
  \[ \text{CPU clock cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times C_i) \]
  Where “C” is the count of the number of instructions of class “i” executed and “n” is the number of different instruction

Review
More Performance Equations

- Amdahl’s Law for “improvements”
  - Execution Time After Improvement = Execution Time Unaffected + ( Execution Time Affected / Amount of Improvement)

- Performance in terms of MIPS
  - MIPS - Million Instructions Per Second
  - MIPS = \frac{\text{Instr. Count}}{(\text{CPU time} \times 10^6)}
  - Specifies performance inversely to execution time.
Problems with MIPS as a measure of comparing computers

- MIPS specifies the execution rate but doesn’t take into the account the capabilities of the instructions.
- MIPS varies between programs on the same computer.
- MIPS can vary inversely with performance.

Performance
Aspects that affect it...

<table>
<thead>
<tr>
<th>CPU time = Seconds Program = Instructions Program x Cycles Instruction x Seconds Cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>instr count</td>
</tr>
<tr>
<td>Program</td>
</tr>
<tr>
<td>Compiler</td>
</tr>
<tr>
<td>Instr. Set</td>
</tr>
<tr>
<td>Organization</td>
</tr>
<tr>
<td>Technology</td>
</tr>
</tbody>
</table>

Two major factors that affect CPI are performance of pipeline and performance of memory system (cache). Is performance affected adversely or favorably? Why?
Pipelining

Performance best determined by running a real application
- Use programs typical of expected workload
- Or, typical of expected class of applications
e.g., compilers/editors, scientific applications, graphics, etc.

Small benchmarks
- nice for architects and designers
- easy to standardize
- can be abused

SPEC (System Performance Evaluation Cooperative)
- companies have agreed on a set of real program and inputs
- valuable indicator of performance (and compiler technology)
- can still be abused
Benchmark Games

- An embarrassed Intel Corp. acknowledged Friday that a bug in a software program known as a compiler had led the company to overstate the speed of its microprocessor chips on an industry benchmark by 10 percent. However, industry analysts said the coding error...was a sad commentary on a common industry practice of “cheating” on standardized performance tests... The error was pointed out to Intel two days ago by a competitor, Motorola... came in a test known as SPECint92... Intel acknowledged that it had “optimized” its compiler to improve its test scores. The company had also said that it did not like the practice but felt compelled to make the optimizations because its competitors were doing the same thing... At the heart of Intel's problem is the practice of “tuning” compiler programs to recognize certain computing problems in the test and then substituting special handwritten pieces of code... New York Times, Saturday, January 6, 1996

Benchmarks

SPEC

- SPEC = Standard Performance Evaluation Corporation
- SPEC 95 CPU Benchmark
  - Used to be industry PC benchmark
  - Consisted of 8 integer (C) & 10 floating point (Fortran)
  - Retired and replaced by SPEC2000
  - SPEC ratio normalized against Sun SPARCstation 10/40
- SPEC 2000 CPU Benchmark
  - Consists of 12 integer (C,C++) & 14 floating point (C, Fortran)
- SPEC now has “suite” of other benchmarks
SPEC CPU2000

<table>
<thead>
<tr>
<th>Integer benchmarks</th>
<th>FP benchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>gzip</td>
<td>perl</td>
</tr>
<tr>
<td>gzip</td>
<td>mcf</td>
</tr>
<tr>
<td>bzip2</td>
<td>soplex</td>
</tr>
<tr>
<td>bzip2</td>
<td>perl</td>
</tr>
<tr>
<td>Erica</td>
<td>mcf</td>
</tr>
<tr>
<td>Parboil</td>
<td>soplex</td>
</tr>
<tr>
<td>Shogun</td>
<td>perl</td>
</tr>
<tr>
<td>Parboil</td>
<td>mcf</td>
</tr>
<tr>
<td>SuperMongoDB</td>
<td>soplex</td>
</tr>
<tr>
<td>SuperMongoDB</td>
<td>perl</td>
</tr>
<tr>
<td>Shogun</td>
<td>mcf</td>
</tr>
<tr>
<td>SPEC CPU2000</td>
<td>SPEC CPU2000</td>
</tr>
</tbody>
</table>

**FIGURE 4.5** The SPEC CPU2000 benchmarks. The U&L integer benchmarks on the left half of the table are written in C and C++, while the floating-point benchmarks in the right half are written in Fortran (F77 or F90) and C. For more information on SPEC and on the SPEC benchmarks, see www.spec.org. The SPEC CPU2000 benchmarks are described in more detail in the Conference on Advanced Research in Computing Science and Engineering (ARCS) proceedings.

**SPEC 2000**

- **important metric**: benchmark performance/clock rate

![Graph showing performance ratios of different benchmarks](image)

- Pentium III is better than Pentium IV on INT Benchmark
- Pentium IV is better than Pentium III on FP Benchmark
Benchmarks Summary

- Performance is specific to a particular program(s)
  - Total execution time is a consistent summary of performance
  - See The Big Picture on page 250 & paragraph on page 271

- For a given architecture performance increases come from:
  - increases in clock rate (without adverse CPI affects)
  - improvements in processor organization that lower CPI
  - compiler enhancements that lower CPI and/or instruction count.

- Pitfall: Expecting improvement in one aspect of a machine's performance to affect the total performance.

- Don't always believe every ad that you read!

Next class...

- Chapter 2
  Instructions: Language of the computer

Next HW:
Exercises 4.1-2, 4.7-8, 4.17