Measuring and Discussing Computer System Performance

or

“My computer is faster than your computer”

The bottom line: Performance

<table>
<thead>
<tr>
<th>Car</th>
<th>Time to Bay Area</th>
<th>Speed</th>
<th>Passengers</th>
<th>Throughput (pmph)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ferrari</td>
<td>3.1 hours</td>
<td>160 mph</td>
<td>2</td>
<td>320</td>
</tr>
<tr>
<td>Greyhound</td>
<td>7.7 hours</td>
<td>65 mph</td>
<td>60</td>
<td>3900</td>
</tr>
</tbody>
</table>

° Time to do the task
  – execution time, response time, latency
° Tasks per day, hour, week, sec, ns. ..
  – throughput, bandwidth

How to measure Execution Time?

% time program
... program results ...
90.7u 12.9s 2:39 65%
%

• Wall-clock time?
• user CPU time?
• user + kernel CPU time?
• Answer:

Our definition of Performance

\[
\text{Performance}_X = \frac{1}{\text{Execution Time}_X}, \text{ for program X}
\]

• only has meaning in the context of a...
• Not very intuitive as an absolute measure, but most of the time we’re more interested in...
Relative Performance

- can be confusing
  - A runs in 12 seconds
  - B runs in 20 seconds
  - A/B = 0.6, so A is 40% faster, or 1.4X faster, or B is 40% slower
  - B/A = 1.67, so A is 67% faster, or 1.67X faster, or B is 67% slower

- needs a precise definition

Examples

- Machine A runs program C in 9 seconds, Machine B runs the same program in 6 seconds. What is the speedup we see if we move to Machine B from Machine A?

- Machine B gets a new compiler, and can now run the program in 3 seconds. Speedup?

Relative Performance (Speedup), the Definition

\[
\text{Speedup } (X/Y) = \frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution Time}_Y}{\text{Execution Time}_X} = n
\]

What is Time?

CPU Execution Time = CPU clock cycles * Clock cycle time

- Every conventional processor has a clock with an associated clock cycle time or clock rate
- Every program runs in an integral number of clock cycles

Cycle Time

MHz = millions of cycles/second, GHz = billions of cycles/second

- X MHz = 1000/X nanoseconds cycle time
- Y GHz = 1/Y nanoseconds cycle time
How many clock cycles?

Number of CPU cycles = Instructions executed * Average Clock Cycles per Instruction (CPI)

Computer A runs program C in 3.6 billion cycles. Program C consists of 2 billion dynamic instructions. What is the CPI?

All Together Now

CPU Execution Time = Instruction Count x CPI x Clock Cycle Time

• IC = 1 billion, 500 MHz processor, execution time of 3 seconds. What is the CPI for this program?

• Suppose we reduce CPI to 1.2 (through an architectural improvement). What is the new execution time?

Who Affects Performance?

• programmer
• compiler
• instruction-set architect
• machine architect
• hardware designer
• materials scientist/physicist/silicon engineer
Performance Variation

CPU Execution Time = Instruction Count × CPI × Clock Cycle Time

<table>
<thead>
<tr>
<th>Number of instructions</th>
<th>CPI</th>
<th>Clock Cycle Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same machine different programs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>same programs, different machines, same ISA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Same programs, different machines</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

MIPS

MIPS = Millions of Instructions Per Second

= Instruction Count / Execution Time × 10^6

= Clock rate / CPI × 10^6

- program-independent
- deceptive

Other Performance Metrics

- MIPS
- MFLOPS

Which Programs?

- peak throughput measures (simple programs)?
- synthetic benchmarks (whetstone, dhrystone,...)?
- Real applications
- SPEC (best of both worlds, but with problems of their own)
  - System Performance Evaluation Cooperative
    - Provides a common set of real applications along with strict guidelines for how to run them.
    - provides a relatively unbiased means to compare machines.
Danger in Benchmark-Specific Performance Measures

- measures compiler as much as architecture!

Amdahl’s Law

- The impact of a performance is limited by the percent of affected by the improvement

\[
\text{Execution time after improvement} = \frac{\text{Execution Time Affected}}{\text{Amount of Improvement}} + \text{Execution Time Unaffected}
\]

- Make the fast!!

SPEC Performance on Pentium III and Pentium 4

SPEC CPU*2000
SPECint_base*2000
SPECfp_base*2000

Key Points

- Be careful how you specify performance
- Execution time = instructions * CPI * cycle time
- Use real applications
- Use standards, if possible
- Make the common case fast

CSE 141  Dean Tullsen
CSE 141  Dean Tullsen