Performance Assessment
Electrical and Computer Engineering
Florida International University
Fall 2009

Performance

• What we care most about the computer performance
  - How fast it can run a program
  - Response time or throughput
    - Response time: time to finish one single program
    - Throughput: total amount of work done in unit time

CPU Performance Equation

• CPU Time
  
  \[ \text{CPU time} = \frac{\text{clock cycles for a program}}{\text{cycle time}} \]

• Can be known
  - Total instruction counts (IC)
  - Cycles per instruction (CPI)
  - Cycle times (the inverse of the clock rate)

\[ \text{CPU time} = \frac{\text{IC} \times \text{CPI}}{\text{clock rate}} = \text{IC} \times \text{CPI} \times \text{cycle time} \]
What if different instructions have different CPIs

- CPU time
  \[ \text{CPU time} = (\sum_{i} IC_{i} \times CPI_{i}) \times \text{cycle time} \]
- Overall CPI
  \[ \text{CPI} = \frac{\sum_{i} IC_{i} \times CPI_{i}}{\sum_{i} IC_{i} \times CPI_{i}} = \sum_{i} IC_{i} \times CPI_{i} \]

Improve CPU time

- Instruction count
  - ISA and compiler technology
- CPI
  - Organization and ISA
- Clock cycle time
  - Hardware technology and organization

Speed measurement

\[ \text{Speedup} \ n = \ \frac{\text{execution time of } Y}{\text{execution time of } X} \times \frac{\text{performance of } Y}{\text{performance of } X} \]

X performs n times better than Y
Example
- 400-MHz processor
- 2 million instructions
- CPU Time?

<table>
<thead>
<tr>
<th>Instruction Type</th>
<th>CPI</th>
<th>Instruction Mix</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALU</td>
<td>1</td>
<td>60%</td>
</tr>
<tr>
<td>Load/Store with cache hit</td>
<td>2</td>
<td>18%</td>
</tr>
<tr>
<td>Load/store with cache miss</td>
<td>8</td>
<td>10%</td>
</tr>
<tr>
<td>Branch</td>
<td>4</td>
<td>12%</td>
</tr>
</tbody>
</table>

MIPS and MFLOPS
- MIPS (million instructions per second)
  - MIPS = IC/(CPU Time x 10^6)
  - Problems?
    - High MIPS ≠ shorter CPU time
- MFLOPS (million floating point operations per second)
  - MFLOPS = floating point operations in a program / (CPU Time x 10^6)
  - Problems?

True/False
- Two processors with same ISA can be judged by clock rate or with single benchmark suite
Performance Comparison

- A is 10x faster than B for Prog P1
- B is 10x faster than C for Prog P2
- A is 20x faster than C for Prog P1
- C is 50x faster than A for Prog P2

Which one is faster?

Using total execution time

<table>
<thead>
<tr>
<th>Program</th>
<th>Computer 1</th>
<th>Computer 2</th>
<th>Computer 3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1100</td>
<td>1101</td>
<td>1102</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>1001</td>
<td>1002</td>
</tr>
<tr>
<td></td>
<td>10001</td>
<td>10001</td>
<td>10001</td>
</tr>
<tr>
<td></td>
<td>1001</td>
<td>1001</td>
<td>1001</td>
</tr>
<tr>
<td>Total</td>
<td>1100</td>
<td>1101</td>
<td>1102</td>
</tr>
</tbody>
</table>

Both program A and B run equal number of times.

Arithmetic Mean

\[ \bar{\text{time}} = \frac{1}{n} \sum_{i=1}^{n} \text{time}_i \]

What if Program A and B run different times?
Weighted arithmetic mean example

Arithmetic mean (weighted)

\[ \sum_{i=1}^{n} \text{weight} \times \text{time} \]

<table>
<thead>
<tr>
<th>Program</th>
<th>Comp A</th>
<th>Comp B</th>
<th>Comp C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prog 1</td>
<td>500</td>
<td>91.909</td>
<td>1.999</td>
</tr>
<tr>
<td>Prog 2</td>
<td>55</td>
<td>18.10</td>
<td>10.09</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Computer</th>
<th>Weighted Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computer A</td>
<td>500.5</td>
</tr>
<tr>
<td>Computer B</td>
<td>91.909</td>
</tr>
<tr>
<td>Computer C</td>
<td>1.999</td>
</tr>
</tbody>
</table>

Normalized Execution Time

- Normalize to a particular machine by dividing all execution times by chosen machine’s time
- Example
  - Program P1 has the following execution times:
    - On machine A: 10 secs
    - On machine B: 100 secs
    - On machine C: 150 secs
  - Normalized to A: A=1, B=10, C=15
  - Normalized to B: A=1, B=1, C=1.5

Normalized Mean

Taking the average of the normalized times
Normalized Geometric Mean Example

- Two programs and three machines

<table>
<thead>
<tr>
<th>ETR P1</th>
<th>ETR P2</th>
</tr>
</thead>
<tbody>
<tr>
<td>11 0.1</td>
<td>10 0.2</td>
</tr>
<tr>
<td>0.5 11</td>
<td>50 11</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Normalized to A</th>
<th>Normalized to B</th>
<th>Normalized to C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>Normalized NGM</td>
<td>Normalized NGM</td>
<td>Normalized NGM</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
</tr>
</tbody>
</table>

NGM

<table>
<thead>
<tr>
<th>Comp A</th>
<th>Comp B</th>
<th>Comp C</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>63</td>
<td>1</td>
<td>63</td>
</tr>
<tr>
<td>1.58</td>
<td>1.58</td>
<td>1</td>
</tr>
</tbody>
</table>

Amdahl’s Law

Improvement by the faster mode is limited by the fraction of time the faster mode can be used

Execution time of any code has two portions

- \( C_{\text{old}} = C_{\text{p1}} + C_{\text{p2}} \)
- \( C_{\text{p1}} = (1 - \alpha) \cdot C_{\text{old}} \): not affected by enhancement
- \( C_{\text{p2}} = \alpha \cdot C_{\text{old}} \): affected by enhancement

Let \( n \) be the speedup factor for \( C_{\text{p2}} \), then

\[
C_{\text{new}} = C_{\text{p1}} + \frac{C_{\text{p2}}}{n} = ((1 - \alpha) + \frac{\alpha}{n}) \cdot C_{\text{old}}
\]

As \( n \to \infty \), \( C_{\text{new}} \to (1 - \alpha) \cdot C_{\text{old}} \)

Example: alpha = 80%

Speedup = \( \frac{1}{1 - \alpha} \cdot \frac{\alpha}{n} \)
Example

- Enhancement: Vector mode
- Portions of code containing computations run 20x faster in vector mode.
- What % of original code must be vectorizable to achieve speedup\(_{overall} = 2\)?

\[
\frac{\text{Speedup}}{\text{execution time}} = \frac{1}{(1-\alpha) + \frac{\alpha}{20}} \quad \alpha = .5263
\]

Example

- FP operations = 25%
- FP operation AVG CPI = 4.0
- AVG CPI for others = 1.33
- FP operations for FPSQR = 2%
- CPI of FPSQR = 20
- Design 1: decrease the CPI of FPSQR to 2
- Design 2: decrease average CPI of all FP to 2.5.
- Which one is better?
Summary

- Measure performance
  - Execution time/throughput
  - CPU time
- Fair comparison of performance
  - Weighted mean,
  - Normalization, geometric mean
- Principles in architecture design
  - Amdahl's law