Computer Performance Overview

CIT 595
Spring 2010

Motivation
- Why certain piece of software performs the way it does?
- Why one instruction set can be implemented to perform better than another?

Definition
- Is the amount of useful work accomplished by a computer system compared to the time and resources used

What defines Performance?
- Running a program on 2 different workstations
  - Then faster one is the one that gets the job done first
  - As a user you are interested reducing the response time
    - i.e. time from start to completion of task (a.k.a. execution time)
- Running a computer center with 2 timeshared computers running jobs submitted by many users
  - Then faster computer is the one that completed most jobs during the day
  - As system administrator you are interested increasing throughput
    - i.e. the total amount of work done in a given time

Measuring Response Time
- Response Time is one of the measure of computer performance

- So how do we measure the time?
  - Just time from start of the program till the time it was done executing (i.e. elapsed or response time)
- However modern computers are timeshared and may simultaneously work on several programs
  - Distinguish between the time the processor is doing work vs. the time spent waiting
Execution Time or Latency

- Response Time = CPU Execution Time + Wait Time

- CPU (processor) execution time i.e. time spent computing for a particular program
  - Not running other programs or waiting on I/O
  - Also called as Latency

CPU Execution Time or Latency

- CPU execution time of program depends on:
  - Dependent on the total number of instructions executed
  - Each instruction performance is dependent on the number of clock cycles it takes to complete the instruction cycle (Fetch thru Store) called as cycles per instruction (CPI)
  - One clock cycle is time interval with clock ticks i.e. clock cycle time

CPU Time = \[
\frac{\text{time program}}{\text{cycle program}} = \frac{\text{time instruction}}{\text{CPI}} \times \frac{\text{cycles instruction}}{\text{Instruction Count}}
\]

CPU Execution Time (contd..)

- Clock cycle time = 1/Clock Rate or 1/Frequency
- Clock rate is given in Hertz (Hz) = cycles/sec

CPU Time = \[
\frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}
\]

Cycles Per Instruction (CPI)

- CPI is an average of all the instructions executed in the program

- Depends on
  - Instruction Set Architecture (ISA)
  - Design details (micro-architecture)
    - CPI varies among different implementations of the same ISA
  - CPI is obtained by performing a detailed simulation of an implementation
Example: Calculating CPI

- Given:

<table>
<thead>
<tr>
<th>Instruction</th>
<th># cycles</th>
<th>% Instr. Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load</td>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>Store</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>ALU</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>Branch/Jump</td>
<td>3</td>
<td>15</td>
</tr>
<tr>
<td>Others</td>
<td>6</td>
<td>5</td>
</tr>
</tbody>
</table>

The average CPI is
\[ 0.25 \times 5 + 0.15 \times 5 + 0.40 \times 5 + 0.15 \times 3 + 0.05 \times 5 \]
\[ = 4.75 \text{ cycles} \]

Number of Instructions & Clock Cycle Time

- Number of Instructions depends on:
  - Algorithm
  - Compiler
  - ISA

- Clock Cycle Time depends on:
  - Hardware Technology (transistor technology)
  - Design (Micro-architecture)
  - Simulate the longest propagation path of your circuit

Maximizing Performance in relation to Execution Time

- One, way to maximize performance, we want to minimize execution time

- Thus we can relate performance and execution time for a machine X as:
  \[ \text{Performance}_X = \frac{1}{\text{Execution Time}_X} \]

- We want to minimize all 3 factors – CPI, # instructions and Clock cycle time
  - This difficult as trying to decrease one often leads to increasing the other parameter
  - E.g. Two ISA Philosophies
    - RISC – low CPI/clock period, high inst count
    - CISC – low insn count, high CPI/clock period

Example for RISC vs. CISC

- Consider the the program fragments:

<table>
<thead>
<tr>
<th>CISC</th>
<th>RISC</th>
</tr>
</thead>
<tbody>
<tr>
<td>mov ax, 10</td>
<td>mov bx, 10</td>
</tr>
<tr>
<td>mul bx, ax</td>
<td>add ax, bx</td>
</tr>
</tbody>
</table>

  - The total clock cycles for the CISC version might be:
    \[ (2 \text{ movs} \times 1 \text{ cycle}) + (1 \text{ mul} \times 30 \text{ cycles}) = 32 \text{ cycles} \]

  - While the clock cycles for the RISC version is:
    \[ (3 \text{ movs} \times 1 \text{ cycle}) + (5 \text{ adds} \times 1 \text{ cycle}) + (5 \text{ loops} \times 1 \text{ cycle}) = 13 \text{ cycles} \]
Performance Ratio (Comparing Performance)

To claim that Machine X performs better than Machine Y:

\[ \text{Performance}_X > \text{Performance}_Y \quad \text{i.e.} \quad \frac{1}{\text{Execution Time}_X} > \frac{1}{\text{Execution Time}_Y} \]

Performance Ratio = \[ \frac{\text{Performance}_X}{\text{Performance}_Y} = \frac{\text{Execution Time}_Y}{\text{Execution Time}_X} \]

Relative Performance Example

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Clock Cycle Time (ns)</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>CPI</td>
<td>2.0</td>
<td>1.2</td>
</tr>
</tbody>
</table>

2 implementations of the same ISA

Which machine (A or B) is faster and by how much?

Let "I" be the number instructions per program

\[ \text{Ex Time}_A = 1 \times 2.0 \times I = 2 \times I \text{ ns} \]
\[ \text{Ex Time}_B = 2 \times 1.2 \times I = 2.4 \times I \text{ ns} \]

\[ \text{Performance Ratio} = \frac{\text{Time}_B}{\text{Time}_A} = \frac{2.4 \times I \text{ ns}}{2.0 \times I \text{ ns}} = 1.2 \]

Machine A is 1.2 times faster than B

Misleading Metrics

Clock
- Which is faster?
  - Clock\(_A\) = 5 GHz, Clock\(_B\) = 3 GHz
- Probably A, but make this assumption only if the same ISA & compiler
  - If stated that CPI\(_A\) = 2 and CPI\(_B\) = 1, then which is faster?

MIPS – Millions of instructions per second
- MIPS = \(1/(\text{cycles/instr}) \times (\text{seconds/cycle}) \times 10^{-6}\)
  - Higher the MIPS, faster the machine
- Different ISAs vary in CPI, difficult to compare MIPS with different ISA

Compilers

Requires knowing the underlying architecture
- Number of Registers, Pipelining, RISC vs. CISC, Number of functional units

Task
- Translating HLL to machine code
- Scheduling operations
- Optimizing Code
  - Loop
  - Local vs. Inter-procedural
Maximizing Performance in relation to Throughput

- Throughput is amount of work that can be done in a given time
  - Measured in tasks per time unit e.g. x tasks per hour
- So higher the throughput, better the performance

- Also, Throughput and Response Time are inversely related
  - If a system carries out a task in k seconds then its throughput is 1/k tasks per second

- Throughput can be increased through parallelism:
  - Instruction level
    - Pipelining and Superscalar techniques
  - Concurrent Programming
    - Multiprogramming
    - Multiprocessors
    - Shared Memory, Distributed, Chip Multiprocessors
  - Improving I/O performance