HPCA Metrics and Evaluation

This lesson shows methods for measuring improvements to a system. These methods are important to the computer architect during the decision making process because they show which designs will yield the best performance.

Performance usually refers to the speed of the processor. Speed can be broken down into two aspects of speed:

        Latency: how much time does it take to go from the start to the finish

        Throughput: how many tasks can be completed in one time unit.

Throughput is NOT always equal to 1/Latency

Comparing Performance

Performance can be compared using speedup.

Speedup is defined as "X is N times faster than Y".

Speedup: $N = Speed(X) / Speed(Y) = Throughput(X) / Throughput(Y)$

             $= Latency(Y) / Latency(X)$

*Note the difference when using Latency

Speedup < 1 : The performance has gotten worse with the newer version.

            The degradation is because of worse performance.

Speedup > 1: The performance has gotten better with the newer version.

            The improvement is through shorter execution time

            or with Higher Throughput.

Performance is proportional to 1/latency

Measuring Performance

Performance ~ 1/Execution time

When comparing performances of different processors, a standard task should be used. This will lead to a standard execution time for the given task for the processor.

Benchmarks

Benchmarks are standard tasks for measuring processor performance. A benchmark is a suite of programs that represent common tasks.

Types of Benchmarks

Real Applications - most realistic, most difficult to set up - used for real machine comparisons.

Kernels - most time consuming part of an application, is still difficult to set up - used for prototypes.

<u>Synthetic</u> - similar to kernels, but simpler to compile, used for design studies
Peak performance - used for marketing.

Benchmark Standards
There are benchmark organizations that create the standard benchmarks.

Summarizing Performance
Summarize performance using the average execution time.
If Speedup is compared to Speedup; the Geometric mean, NOT the average, must be used.

<u>Geometric mean</u> = (Product of the terms)$^{1/\text{Number of terms}}$

Iron Law of Performance
The <u>Iron Law of Performance</u> is :
<u>CPU Time</u> = # of Instructions in the Program * Cycles per Instructions * Clock cycle Time

 All three aspects are important in decision making in computer architecture

- # of Instructions in the Program: is effected by the algorithm, the compiler used, and/or the instruction set used.
- Cycles per Instructions: is affected by the instruction set and/or the processor design
- Clock cycle Time: is affected by processor design, circuit design, and/or transistor physics
- Computer architects influence the instruction set and the processor design.

The <u>Iron Law for Unequal Instruction Times</u>
CPU Time = [Sum of(Inst/Program * cycles/Inst)] * Time/cycle

<u>Amdahl's Law</u>
Used for measuring speedup when only a fraction of the system is improved.
Speedup = 1/((1-Frac of Enhancement) + (Frac Enhancement/Speedup Enhancement))

Frac of Enhancement = % of original execution time that is affected by enhancement.

Amdahl's Law Implications
Use Amdahl's law to determine which design decisions will yield the best system.
Make the Common Case Fast : small improvements for a large percentage of the system are better than large improvements on a small percentage of the system.

Lhadma's Law
While trying to make the common case fast, do not make the uncommon case worse.

Diminishing Returns

Once the easy changes for improvement have been made, further improvements will not yield great returns. More and more work must be done to obtain small improvements and these improvements will result in small to nearly zero speedups.