Qualitative Computer Design

Design guided by measured performance.

Covered:

- Benchmarks. (1.5)
- Measures of performance. (1.5, 1.6)
- Principles and measured performance. (1.6)
- Example: Memory. (1.7)

(Numbers refer to book sections.)

Benchmarks

Benchmark:
program used to evaluate performance.

Uses

- Guide computer design.
- Guide purchasing decisions.
- Marketing tool.

Guiding Computer Design

Measure overall performance.

Determine characteristics of programs.
E.g., frequency of floating-point operations.

Determine effect of design options.

Choosing Benchmark Programs

Important: Choice of programs for evaluation.

Optimal but unrealistic:

The exact set of programs customer will run.

Problem: computers used for different applications.

Therefore, must model typical users' workload.

Options:

Real Programs
Programs chosen using surveys, for example.

\+ Measured performance improvements apply to customer.
– Large programs hard to run on simulator. (Before system built.)

Kernels
Use part of program responsible for most execution time.

\+ Easier to study.
– Not all program have small kernels.

Toy Benchmarks
Program performs simplified version of common task.

\+ Easier to study.
– May not be realistic.

Synthetic Benchmarks
Program "looks like" typical program, but does nothing useful.

\+ Easier to study.
– May not be realistic.

Commonly Used Option

Overall performance: real programs

Test specific features: synthetic benchmarks.

Benchmark Suites

Definition: a named set of programs used to evaluate a system.

Typically:

- Developed and managed by a publication or non-profit organization.
  E.g., Standard Performance Evaluation Corp., PC Magazine.
- Tests clearly delineated aspects of system.
  E.g., CPU, graphics, I/O, application.
- Specifies a set of programs and inputs for those programs.
- Specifies reporting requirements for results.

What Suites Might Measure

- Application Performance
  E.g., productivity (office) applications, database programs.
  Usually tests entire system.
- CPU and Memory Performance
  Ignores effect of I/O.
- Graphics Performance

Example, SPEC 95 Suites

Respected measure of CPU performance.

Managed by Standard Performance Evaluation Corporation,...

...a non-profit organization funded by computer companies.

Measures CPU and memory performance on integer and FP code.

Uses common Unix programs such as perl, gcc, compress.

Requires that results on each program be reported.

Programs compiled with publicly available compilers and libraries.

Programs compiled with and without expert tuning.

SPEC 95 Suites and Measures

CINT95 suite of integer programs run to determine:

- SPECint95, execution time of tuned code.
- SPECint_base95, execution time of untuned code.
- SPECint_rate95, throughput of tuned code.

CFP95 suite of floating programs run to determine:

- SPECfp95, execution time of tuned code.
- SPECfp_base95, execution time of untuned code.
- SPECfp_rate95, throughput of tuned code.

Other Examples

BAPCO Suites, measure productivity app. performance on Windows 95.

TPC, measure "transaction processing" system performance.

WinMARK, graphics performance.

Reporting Results

Options for Combining Performance of Suite Members
(This is harder than it sounds.)

Let n denote number of programs in suite.

Let t_i denote run time of program i.

Run times of suite members combined using:

- Arithmetic Mean of Execution Times

  $$\frac{1}{n}\sum_{i=1}^{n} t_i.$$

  Emphasizes programs with longest running time.

- Weighted Arithmetic Mean

  Let $w_i \in [0,1]$ be *weight* (importance) of program i,

  where $\sum_{i=1}^{n} w_i = 1$.

  The weighted arithmetic mean is given by:

  $$\sum_{i=1}^{n} w_i t_i.$$

  Emphasizes programs based on importance to users.

- Harmonic Mean of Execution Times

  $$\left(\frac{1}{n}\sum_{i=1}^{n}\frac{1}{t_i}\right)^{-1}.$$

  Emphasizes programs with shortest running time.

- Geometric Mean of Execution Times

  $$\sqrt[n]{\prod_{i=1}^{n} t_i}.$$

  Useful property: $\frac{\mathrm{GM}(X_i)}{GM(Y_i)} = GM\left(\frac{X_i}{Y_i}\right)$.

  Emphasizes programs with large change in performance.

- Normalized Execution Time

  For program i: t_i/t'_i,

  where t'_i is execution time on *reference* machine.

  Emphasizes performance relative to a common computer.

  SPEC 95 reference: Sun SPARCstation 10.

- Geometric Mean of Normalized Execution Times

  $$\sqrt[n]{\prod_{i=1}^{n} \frac{t_i}{t'_i}} = \frac{\sqrt[n]{\prod_{i=1}^{n} t_i}}{\sqrt[n]{\prod_{i=1}^{n} t'_i}}.$$

  Insensitive to relative performance of suite members on reference machine.

Measuring Performance

Bottom-Line Measures:

Execution Time: [of a particular program]

Time from program start to finish.

Throughput: [of a collection of programs]

Work per unit time.

Execution time important to users. (Obviously.)

Throughput important to accountants.

How Execution Time Reported By OS

Total Run Time Called

- *Elapsed Time*
- *Response Time*
- *Wall-Clock Time*

During execution CPU may be in:

- *User* mode: Possibly running "our" program.
- *System* mode: Possibly running OS for our program.
- In user or system mode running someone else's program.
- Idle.

Reported by Unix `time` Utility

For a particular program (process):

- User Time
- System Time
- Elapsed Time

Additional Performance Measures Using Above

Call a system running benchmark program only, *unloaded*.

System Performance:

Elapsed time on an unloaded system.

CPU Performance:

Sum of User and System time on an unloaded system.

Components of CPU Performance

CPU Performance Decomposed into Three Components:

- Clock Frequency (ϕ)
  Determined by technology and influenced by organization.
- Clocks per Instruction (CPI)
  Determined by organization.
- Instruction Count (IC)
  Determined by program and ISA.

Execution time $= \frac{1}{\phi} \times \mathrm{CPI} \times \mathrm{IC}$.

02-13

Principles of Computer Design

Principles computer designers apply widely.

- Make the common case fast.
  Obviously.
- Amdahl's Law: Don't make common case too fast.
  As speed of one part increases...
  ...impact on total performance drops.
- Locality of Reference.
  Temporal: It might happen again soon.
  Spatial: It might happen to your neighbors soon too.

EE 4720 Lecture Transparency. Formatted 14:43, 17 January 1997 from lsli02.
02-13