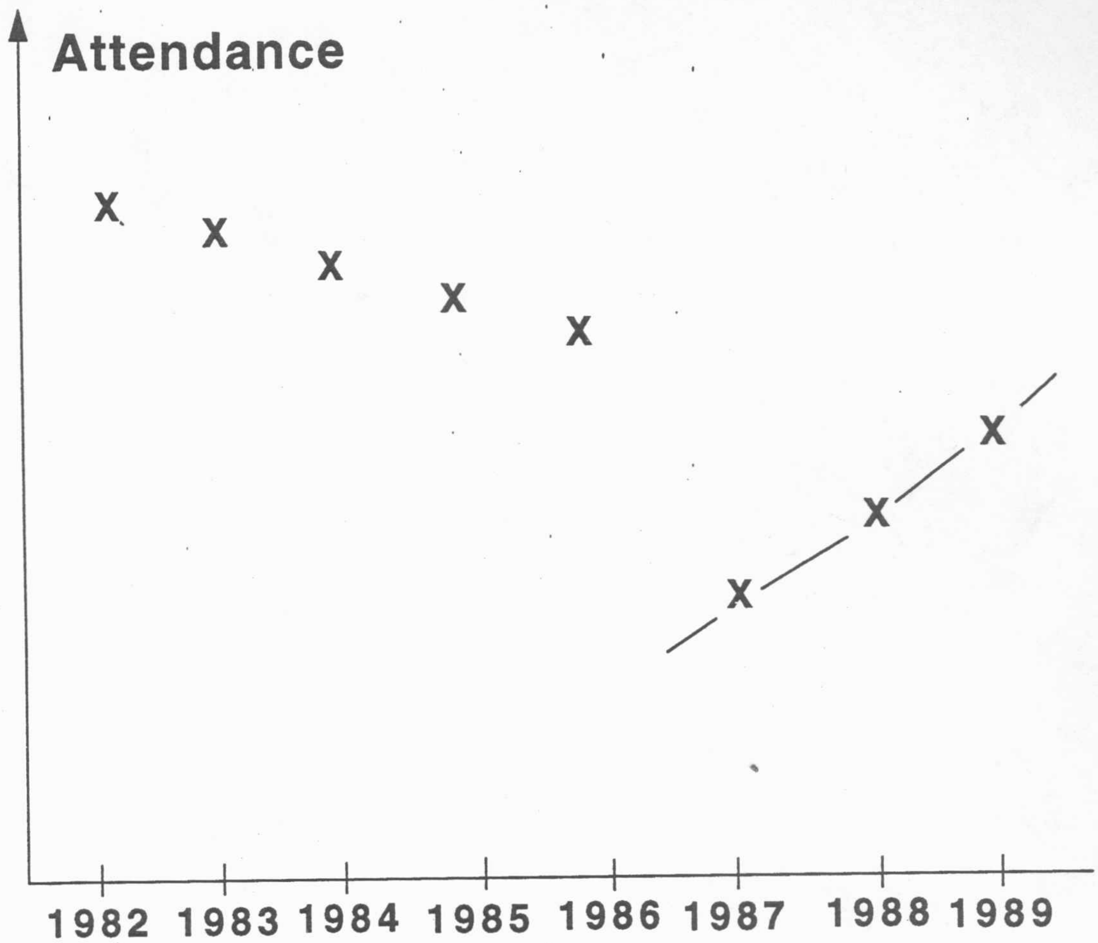
Introduction to Measurement Methodology

Outline

* Introduction
  - Misuse of the data
  - The Basic Equation (how long did it take)
  - The Mean
* How do we Measure
  - Real Hardware, Simulator, Analytical Model
  - Hardware Instrument, µcode, Software Monitor
* What do we Measure (Benchmarks)
  - Synthetic code
  - Kernels
  - Toy Benchmarks
  - SPEC
  - The Perfect Club
  - Your Relevant Workload
* Serious Abuses

From a Welcoming Address At A Well-Known Conference



Why Measure

* Before the fact
  - So we know what to build

* After the fact
  - So we know what to do next time

The Standard Performance Equation

$$T = \ell * CPI * t_c$$

Path Length

- Algorithm
- Language
- Compiler
- ISA

Clock

- Technology
- Organization

Cycles Per Instruction

- ISA
- Organization
  - Pipelining
  - Issue Rate
  - Branch Handling

Means

* **Arithmetic Mean**

$$A = \frac{1}{n} \sum_{i=1}^{n} P_i$$

* **Geometric Mean**

$$G = \sqrt[n]{\prod_{i=1}^{n} P_i}$$

* **Harmonic Mean**

$$H = \frac{1}{\frac{1}{n} \sum_{i=1}^{n} \frac{1}{P_i}}$$

How Do We Measure

<u>Degree of Santizing</u>

Real Hardware — Simulation — Analytic Model

Real Hardware

- "Gotchas" Have a chance to get in the way
- Least Flexible
- Fast for doing thorough job

Simulation

- Some effects are missing
- Most Flexible
- Slowest

Analytic Model

- Good for gross effects
- <u>Must</u> be validated

How Do We Measure (Continued)

Invasiveness

Hardware Instrumentation — Microcode Instrumentation — Software Monitoring

Hardware Instrumentation

- Most Expensive
- Non-Invasive
- Least Flexible

Microcoded Instrumentation

- Best of Both Worlds
- SPAM

Software Monitoring

- Cheap
- Very Invasive
- Most Flexible

SPEC 2006

CINT2006 (12)

9 IN C
3 IN C++

perlbench
bzip2
gcc
mcf
gobmk
hmmer
sjeng
libquantum
h264ref

---

omnetpp
astar
xalancbmk

CFP 2006 (~~16~~ 17)

6 IN FORTRAN
3 IN C
4 IN C, FORTRAN
4 IN C++

bwaves (F)
gamess (F)
milc (C)
zeusmp (F)
gromacs (C, F)
cactusADM (C, F)
leslie3d (F)
namd (C++)
dealII (C++)
soplex (C++)
povray (C++)
calculix (C, F)
GEMSFDTD (F)
tonto (F)
lbm (C)
wrf (C, F)
sphinx3 (C)

Benchmarks

Rationale: Find a set of programs or program fragments representative of the workload you will be requiring of the machine

Types:

1. The ADD instruction - very old
2. Instruction MIX - Old (Gibson MIX, 1959)
3. Kernels
   - e.g., Livermore Loops
4. Synthetic Benchmarks
   - Parameterized
   - Careful: RRW is not RWR
5. Toy Benchmarks
   - Easy to hand-compile
   - Pretty much in disrepute today e.g., Towers of Hanoi
6. SPEC Suite (Systems Performance Evaluation Co-operative)
   - At least common agreement, I Guess!!
7. Real Workload

Bad Ways to Measure Performance (... and each has been used and reported in the Open Literature)

* **Apples & Oranges**
  - A Lightly Loaded VAX vs. Counting Simulated Cycles

* **Who Gets the Credit**
  - The Architecture or the Compiler
  - Example: Berkeley Pascal vs VMS Pascal
  - Algorithm Optimizations
  - Instruction set or register windows (Colwell)

* **Choice on Benchmarks**
  - Selective
    * Overstates significance of one feature
      e.g. Regularity (Fl. Pt.)
      e.g. Procedure Call Intensive
      e.g. No Floating Point
  - Small
    * 100% Cache, TB Hits
    * No I/O, Context Switch

* Play with Statistics

	Program A	Program B
Machine 1:	1 unit	2 units
Machine 2:	2 units	1 unit

Machine 1 is $\frac{2}{1}$ on A, $\frac{1}{2}$ on B

Speed Up is $\frac{1}{2}\left(2 + \frac{1}{2}\right) = 1.25$

* Too Focused on Frequency

	Frequency	Execution Time
Calls	2.5%	21.6%
MOVL	12.4%	6.8%