Performance Benchmarking (Lying with numbers)

CS 347

Feb 1, 1996

Topics

- Lying with peak measures
- Lying with benchmark measures

Characterizing Computer Performance

Computer buyers want a single number that predicts performance of real applications.

Computer makers have resisted measures that would allow meaninful direct comparisons.

- lack of operating system and language standards
- fear of losing

1970's and 1980's:

- golden age of meaningless rates

1980's:

- dawn of meaningless benchmarks

1990's:

- semi-realistic benchmarks

Meaningless rate #1: MHz

MHz = millions of clock cycles/sec

MHz doesn't predict running time:

- **T secs = I inst x (C ticks/I inst) x 1/(MHz x 10^6) cycles/sec**

CPU	MHz	System	SPECfp95 time (secs)
Pentium Pro	180	Alder	6,440
POWER2	77	RS/6000 591	3,263

Meaningless rate #2: peak MIPS

MIPS = millions of instructions / second

Peak MIPS = MIPS for some optimal instuction stream.

Peak MIPS doesn't predict running time:

- **number of instructions executed don't predict running time.**
- **optimal instruction stream can be meaningless**

Example:

If the instruction stream is a sequence of NOPS, then a 100 MHz Pentium is a 3,200 MIPS machine!

Instruction decoder looks at 32 bytes at a time. NOP is a one-byte instruction. Decoder discards NOP's.

Meaningless rate #3: peak MFLOPS

MFLOPS = millions of floating operations /sec

peak MFLOPS = MFLOPS for some optimal instruction stream.

MFLOPS doesn't predict execution time:

- **floating point operations do not predict running time**
- **even if the did, the ideal instruction stream is usually unrealistic**

Measured MFLOPS on Intel i860 (peak MFLOPS = 80):

Program	1d fft	sasum	saxpy	sdot	sgemm	sgemv	spvma
MFLOPS	8.5	3.2	6.1	10.3	6.2	15.0	8.1
%peak	11%	4%	7%	13%	8%	19%	10%

Benchmarking

Goal: Measure a set of programs (benchmarks) that represent the workload of real applications and predict the running time of the applications.

Steps in the benchmarking process:

(1) Choose representative benchmark programs.
- difficult to find realistic AND portable programs.

(2) Choose an individual performance measure (for each benchmark)
- time, normalized time, rate

(3) Choose an aggregate performance measure (for all benchmarks)
- sum, normalized sum, mean, normalized mean

Benchmarks

Synthetic benchmarks

- artificial code designed to approximate real workloads
- examples: Whetstone, Dhrystone
- big in the 80's but less important now

Kernel benchmarks

- small to medium code fragments from real applications
- examples: Livermore Loops, Linpack
- big in the 80's but less important now.

Semi-real applications

- larger programs based on real applications
- example: SPEC benchmarks (System Perf Evaluation Corp)
- most significant modern benchmark suite.

Real applications

- the ultimate benchmarks, but not always feasible.

SPEC Benchmarks

http://www.specbench.org/osg/cpu95/news/cpu95desc.html

Three incarnations: SPEC89, SPEC92, SPEC95.

SPEC95 Benchmark Suite

Integer benchmarks (C)	Floating point benchmarks (Fortran)
go player	mesh generation
processor simulation	ocean model
gcc	particle simulation
lzw compression	galactic jet model
lisp interpreter	potential field model
jpeg compression	linear system solver
perl interpreter	turbulence model
database	weather model
	chemistry model
	electromagnetic model

SPEC performance measures

SPECfp = $(T_1 \times T_2 \times \ldots \times T_n)^{1/n}$

Each T_k is a normalized time:

- **(reference time for benchmark k) / (measured time for benchmark k)**
- **reference times are measured on a Sparcstation 10/40**

Problem: SPEC performance measures don't predict execution time!!!

system	total time	SPECfp95
166 MHz Pentium Pro	6470	5.47
180 MHz Pentium Pro	6440	5.40

Lying with means and ratios

	frames	sys A	sys B	sys C
prog 1	320	20 secs	10 secs	40 secs
prog 2	320	40 secs	80 secs	20 secs
total		60 secs	90 secs	60 secs

Total running time predicts performance.

Lying with means and ratios

	frames	seconds sys A	sys B	sys C
prog 1	320	20	10	40
prog 2	320	40	80	20
total		60	90	60
normalized to A		1	1.5	1
normalized to B		.67	1	.67
normalized to C		1	1.5	1

Normalized total running time is OK too. It tracks with total running time.

Lying with means and ratios

Arithmetic mean (AM) = (T1 + T2 + ... + Tn)/n

	frames	seconds sys A	sys B	sys C
prog 1	320	20	10	40
prog 2	320	40	80	20
total		60	90	60
AM		30	45	30
normalized to A		1	1.5	1
normalized to B		.67	1	.67
normalized to C		1	1.5	1

Normalized and unormalized arithmetic means predict running time.

Lying with means and ratios

		seconds (normalized to A)(normalized to B)		
	frames	sys A	sys B	sys C
prog 1	320	20 (1.0) (2.0)	10 (0.5) (1.0)	40 (2.0) (4.0)
prog 2	320	40 (1.0) (0.5)	80 (2.0) (1.0)	20 (0.5) (0.25)
total		60 (2.0) (2.5)	90 (2.5) (2.0)	60 (2.5) (4.25)
AM		30 (1.0) (1.25)	45 (1.25) (1.0)	30 (1.25) (2.13)

The arithmetic mean of the total time predicts running time.

The sum of normalized times and the AM of normalized times do NOT predict running time.

Lying with means and ratios

Geometric mean (GM) = $(T1 \times T2 \times \ldots \times Tn)^{1/n}$

		seconds (normalized to A)(normalized to B)		
	frames	sys A	sys B	sys C
prog 1	320	20 (1.0) (2.0)	10 (0.5) (1.0)	40 (2.0) (4.0)
prog 2	320	40 (1.0) (0.5)	80 (2.0) (1.0)	20 (0.5) (0.25)
total		60 (2.0) (2.5)	90 (2.5) (2.0)	60 (2.5) (4.25)
GM		28.3 (1) (1)	28.3 (1) (1)	28.3 (1) (1)

The geometric means of the total, and the geometric means of the normalized total are consistent (i.e. independent of the system they are normalized to), but they are consistently wrong!

This explains the inconsistency in the SPECfp95 numbers.

Lying with means and ratios

The harmonic mean (HM) is a measure for rates (and ratios in general) that predicts running time:

$HM = n / sum_{k=1..n}(1/rate_k)$

where $rate_k = operations/time_k$

$HM = n / sum_{k=1..n}(time_k/operations)$

$= (operations \times n) / sum_{k=1..n}(time_k)$

Lying with means and ratios

	frames	frames/sec sys A	sys B	sys C
prog 1	320	16	32	8
prog 2	320	8	4	16
total frames/sec		24	36	24
AM		12	18	12
GM		11.3	11.3	11.3
HM		10.7	7.1	10.7
(total time)		(60)	(90)	(60)

HM is the only measure for rates (and ratios in general) that predicts running time.

Lying with rates and means (Summary)

1. Total running is the true performance measure.
2. AM can be used to summarize performance expressed as an unnormalized time.
3. AM should NOT be used to summarize performance expressed as a ratio (i.e. a rate or normalized time)
4. GM should not be used for summarizing performance expressed as a time or a rate.
5. HM should be used for summarizing performance expressed as a ratio.
6. If you want to normalize, compute the aggregate measure first, then normalize.