Chapter

Performance

2 Performance

What do you mean by performance?

The importance metric is how does your application perform?
How does your mix of applications perform?

Speed .... is 0.1 seconds different from 0.5 seconds
1% chance of response exceeding 2 seconds
total throughput or individual latency.

Acceptable?

Cost .... do we need to train staff, or hire extra staff

Time ... can it be installed in 6 weeks.

Speed: rendering images.
A system of 100 cores which renders images such that each core takes 5 seconds to render one image. Average throughput is 0.05 seconds.
Good for rendering a movie, useless for a real time computer game.

User wants response.
conflicts
Provider throughput

Acceptable?

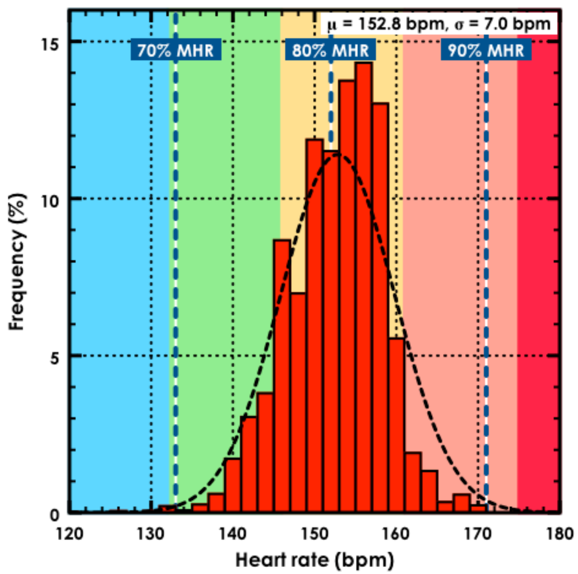
3 Analysing*

What do you do with the measures?

Much of statistical analysis assumes Gaussian.
But computer responses may not be Gaussian.
Careful interpretation of the data

A good assumption.

2008 Terrapin Mountain Marathon



Gaussian measurement

Mean is a sensible measure of behaviour.
Sigma gives a good measure of “width”.

Enough data to draw robust conclusions.
Are your results repeatable?
Even here some asymmetry. Significant?

Likely range

Extrapolation
Remember uncertainties must also be propagated.

New effects may occur.
Things are not always linear.

4 MTTF

What does this mean?

Mean Time To Failure	MTTF
Mean Time To Repair	MTTR

MTTF	1,200,000 hours
Disk lifetime	43,000 hours

Measured by taking a large number of disks – say 10,000 and running say 2400 hours (4 months) and count the failures.

$$\text{MTTF} = \frac{\text{\# of hours run}}{\text{\# of failed disks}} = \frac{10{,}000*2400}{20} = 1{,}200{,}000$$

So 1 disk running for 1 year has a $\frac{43{,}800}{1{,}200{,}000} = 0.9\%$

Chance of failing – or around 4½% over its lifetime.

Failures are correlated – manufacturing fault.

or environmental insult …

All disks in lifetime

Backup …

Last year at RAL disk failures every dew days

Brunel Grid node motherboards

Performance

5 Metrics

Terms

Performance

CPU time

Wall time

Clock period

Clock rate MHz/GHz

CPI

Measurements of performance

$$\text{Performance} = \frac{1}{\text{Time}}$$

Time: *Elapsed time/response time/Wall clock time*
Time from submission to retrieval.

CPU (execution) time system
user

Clock cycles (ticks) = *1/clock period.*

CPU Time for a programme = CPU clock cycles X Clock period for programme

There is a design trade off – powerful instruction sets take fewer instructions per programme, but more time per instruction.

The average number of cycles per instruction is referred to as **clock cycles per instruction (CPI)**

Time = Instructions/program * CPI * Clock period
Alternative expression for execution time

Do you include network transfer? Queuing?

Constraints

Hard real time constraint.
A fly by wire system must respond in a maximum time.

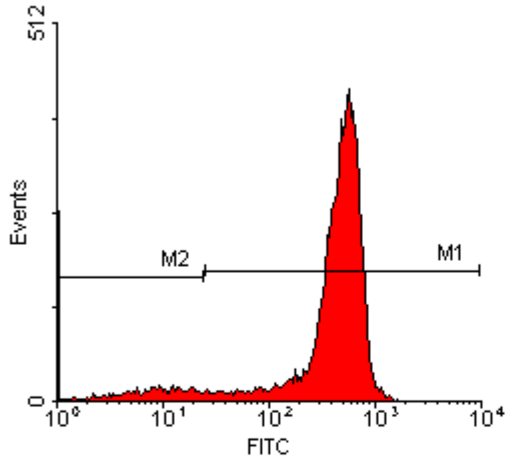
Soft real time constraint.
iPod playback must return the stream within a maximum time, most of the time.

Ignores latency from any cause

6 Non Gaussian

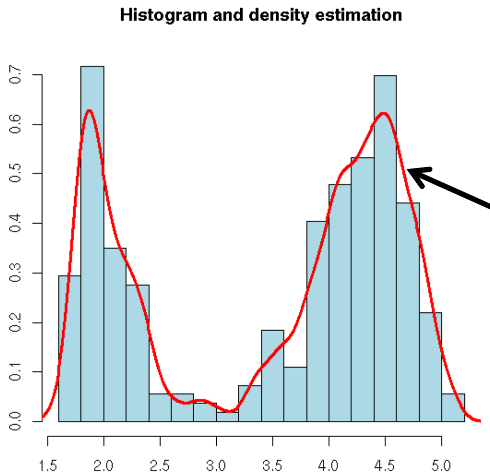
How do you deal with non Gaussian?

Display the full results. This may be the only way.



Give the full range : minimum to maximum
Give the 90% range – about some suitable point …. mean, mode, median, from smallest, from largest, from 5%-95%

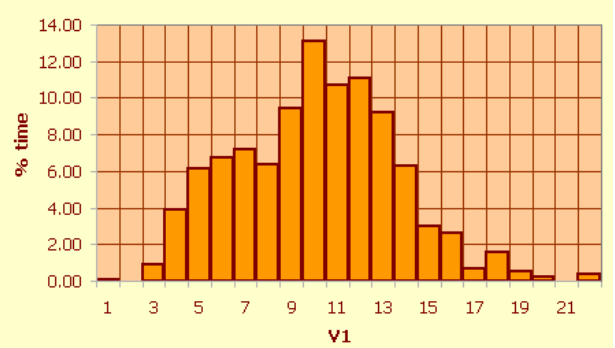
Compare with a model and give the model parameters (**plus errors**).



Two Gaussians.

What are the results for …..

Don't just give the mean!



Performance

Standard benchmarks Spec

Industry standard set of benchmarks.
Measures amount of time to finish a task.
New version produced every few years.
Spec CPU92, CPU95, CPU2000, CPU2006.
1.Because the performance increases and if we didn't the times for some tasks would become so small as to be meaningless.
2.Nature of a suitable set of tasks changes
3.Manufacturers tune their machines and compilers to perform well on benchmarks.
Review to ensure they continue to provide a **real** measure of performance

Set of tasks, meant to reflect the real world "typical" mix of tasks.
Weighting also meant to reflect real world weighting.
.

It will be misleading

A single number

.

Execution time on a number of different programs.

What to use?
Arithmetic average of execution time of all programs?

They vary in speed … implicit weighting.
Explicit weight … but the mix is supposed to be representative.

Weighting would encourage companies to reweight.

SPECRatio: Normalize execution times to reference computer

$$\text{Ratio} = \frac{\text{time on reference computer}}{\text{time on computer being rated}}$$

Note ratio

machines A and B. SpecRatio(A) = 1.25*SpecRatio(B)

$$1.25 = \frac{\text{SpecRatio(A)}}{\text{SpecRatio(B)}} = \frac{\text{Time on Ref}}{\text{Time on A}} \Big/ \frac{\text{Time on Ref}}{\text{Time on B}}$$

$$= \frac{\text{Time on B}}{\text{Time on A}}$$

Actual ref machine is unimportant

Summary

How to aggregate the ratios of the different programs?

$$GeometricMean = \sqrt[n]{\prod_{i=1}^{n} SPECRatio_i}$$

Geometric mean of the ratios is the same as the ratio of the geometric means.
Again choice of computer is irrelevant

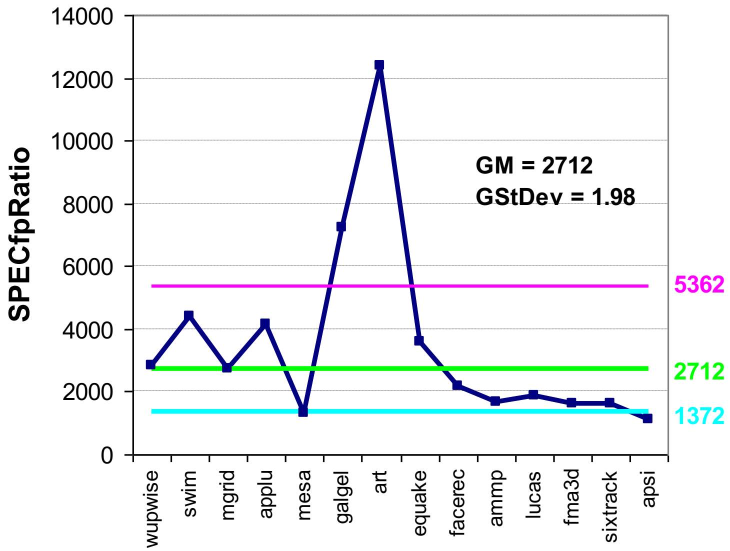
A	B		A/B	
1.3	1.2		1.083333	
2.2	2.1		1.047619	
0.9	1		0.9	
1.7	1.75		0.971429	
0.8	0.75		1.066667	
1.1	1.1		1	
0.85	0.8		1.0625	
1.184583	1.164885	1.01691	1.01691	Geom
1.166369	1.159134	1.006242	1.018793	Arith

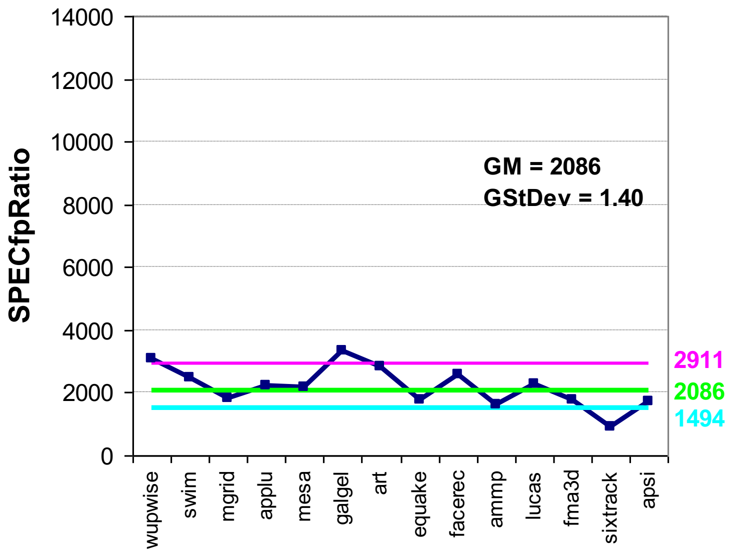
SpecRatio for different programs

Performance ratio for different programs

Means of the ratios

Ratios of the means





Equal means are not (always) equally useful
.

Two distributions, both with similar means.

Top distribution is less useful

Bottom distribution … which ever benchmark most resembles your job, the mean is a good measure.

Top distribution, if your job looks like *art or galgel* significantly under estimated. Like the others overestimated.

Beware Manufacturers can tune to the benchmark.
Special compiler switches.
70% of SPEC programs were dropped from the next release as no longer useful.

List of benchmarks

.

gzip	compression	wupwise	Quantum Chromodynamics	
vpr	FPGA circuit placement	swim	Shallow water model	
gcc	GNU C compiler	mgrid	3D potential field	
mcf	Combinatorial opitmisation	applu	Elliptic PDE solver	
crafty	Chess program	mesa	3d Graphics	
parser	Word Processor	galgel	CFD	
eon	Visualisation	art	Image recognition	
perlbmk	perl application	equake	Seismic wave propagation	
gap	Group theory	facerec	Face recognition	
vortex	OO database	ammp	Computational chemistry	
bzip2	Compression	lucas	Primality testing	
twolf	Place and rote simulator	fma3d	Crash simulation	
		sixtrack	HEP accelerator design	
		apsi	Meteorology	

A number are easy to scale up
gcc – bigger programme
simulations – increase size or increase mesh density:
sixtrack, wupwise, swim, mgrid, equake.

Calculating reliability

If modules have exponentially distributed lifetimes. (actually look more U shape). Age of module does not affect failure probability.
1 power supply with a MMTF of 100,000.
Dual power supply – expected time to first failure?
50,000 hours.

System failure is of course longer. But replacement is more frequent. More costly, more time consuming.

Failure time for a system of 10 disks each with a MMTF of 1 million hours. A disk controller with a MMTF of ½ million hours and a power supply with a MMTF of $^1/_5$ million hours.
Power supply is 1/200,000,
Controller is 1/500,000
Disk is 1/1,000,000 – but ten of them
Total 10*1/1,000,000 + 1/500,000 + 1/200,000
=17/1,000,000
MMTF = 1,000,000/17 = 58,800 hours

Failure rate is sum of individual failure rates

Calculating reliability

MTTR – mean time to repair.
Asking about reliability it is also important to ask how long does it take to fix a problem.

Very unlikely but long break
v. likely but minimal break.

So probable time loss is probability of break*time to repair.
Sum over all such incidents to get estimate of down time.

Raid works because although MTTF is shorter than for high spec disks. MTTR can be zero.

Subtle problems

Assume you want to run two jobs – with "equal" computing requirements
Each takes 6 hours on core A and 12 hours on core B

Compare a chip with 1 A core, with two B cores.
Performance time is the same.

Systems are equivalent?

Memory requirement doubles.
Number of I/O channels – to files and database channels doubles.
I/O rate fixed – but channel overhead
Number of jobs simultaneously handled by scheduler.

Beware

Correspond to current paradigm.