CS654 Advanced Computer Architecture

Lec 4 - Introduction

Peter Kemper

Adapted from the slides of EECS 252 by Prof. David Patterson
Electrical Engineering and Computer Sciences
University of California, Berkeley

Technology Trends

- **Moore's Law: 2X transistors / "year"**
  - **# on transistors / cost-effective integrated circuit double every N months ($12 \le N \le 24$)**
  - **Note: N varies over time**
- **Bandwidth Rule:**
  - **For disk, LAN, memory, and microprocessor, bandwidth improves by square of latency improvement**
  - **In the time that bandwidth doubles, latency improves by no more than 1.2X to 1.4X**

Outline

- Review
- Technology Trends: Culture of tracking, anticipating and exploiting advances in technology
- **Careful, quantitative comparisons:**

1. **Define and quantify power**
2. **Define and quantify dependability**
3. **Define, quantity, and summarize relative performance**
4. **Define and quantify relative cost**

Define and quantify power (1 / 2)

- **For CMOS chips, traditional dominant energy consumption has been in switching transistors, called *dynamic power***

$$Power_{dynamic} = 1/2 \times CapacitiveLoad \times Voltage^2 \times FrequencySwitched$$

- **For mobile devices, energy better metric**

$$Energy_{dynamic} = CapacitiveLoad \times Voltage^2$$

- **For a fixed task, slowing clock rate (frequency switched) reduces power, but not energy**
- **Capacitive load a function of number of transistors connected to output and technology, which determines capacitance of wires and transistors**
- **Dropping voltage helps both, so went from 5V to 1V**
- **To save energy & dynamic power, most CPUs now turn off clock of inactive modules (e.g. Fl. Pt. Unit)**

Example of quantifying power

- **Suppose 15% reduction in voltage results in a 15% reduction in frequency. What is impact on dynamic power?**

$$Power_{dynamic} = 1/2 \times CapacitiveLoad \times Voltage^2 \times FrequencySwitched$$

$$= 1/2 \times .85 \times CapacitiveLoad \times (.85 \times Voltage)^2 \times FrequencySwitched$$

$$= (.85)^3 \times OldPower_{dynamic}$$

$$\approx 0.6 \times OldPower_{dynamic}$$

- **Trends:**
  - **First microprocessors uses 1/10 of a Watt**
  - **3.2 GHz Pentium 4 Extreme Edition uses 135 Watt**

  ⇒ **Challenge for power distribution and power supply,**

  ⇒ **Challenge for cooling (air cooling has limits …)**

Define and quantify power (2 / 2)

- Because leakage current flows even when a transistor is off, now *static power* important too

$$Power_{static} = Current_{static} \times Voltage$$

- Leakage current increases in processors with smaller transistor sizes
- Increasing the number of transistors increases power even if they are turned off
- In 2006, goal for leakage is 25% of total power consumption; high performance designs at 40%
- Very low power systems even gate voltage to inactive modules to control loss due to leakage

Outline

- Review
- Technology Trends: Culture of tracking, anticipating and exploiting advances in technology
- Careful, quantitative comparisons:

1. Define and quantify power
2. Define and quantify dependability
3. Define, quantify, and summarize relative performance
4. Define and quantify relative cost

Define and quantify dependability (1/3)

- How decide when a system is operating properly?
- Infrastructure providers now offer Service Level Agreements (SLA) to guarantee that their networking or power service would be dependable
- Systems alternate between 2 states of service with respect to an SLA:

1. Service accomplishment, where the service is delivered as specified in SLA
2. Service interruption, where the delivered service is different from the SLA

- Failure = transition from state 1 to state 2
- Restoration = transition from state 2 to state 1

Define and quantify dependability (2/3)

- ***Module reliability*** **= measure of continuous service accomplishment (or time to failure).**
  2 metrics

1. ***Mean Time To Failure*** **(*MTTF*) measures Reliability**
2. ***Failures In Time*** **(*FIT*) = 1/MTTF, the rate of failures**
   - **Traditionally reported as failures per billion hours of operation**

- ***Mean Time To Repair*** **(*MTTR*) measures Service Interruption**
  – ***Mean Time Between Failures*** **(*MTBF*) = MTTF+MTTR**
- ***Module availability*** **measures service as alternate between the 2 states of accomplishment and interruption (number between 0 and 1, e.g. 0.9)**
- ***Module availability = MTTF / (MTTF + MTTR)***

Example calculating reliability

- If modules have *exponentially distributed lifetimes* (age of module does not affect probability of failure), overall failure rate is the sum of failure rates of the modules
- Calculate FIT and MTTF for 10 disks (1M hour MTTF per disk), 1 disk controller (0.5M hour MTTF), and 1 power supply (0.2M hour MTTF):

$$FailureRate =$$

$$MTTF =$$

Example calculating reliability

- **If modules have *exponentially distributed lifetimes* (age of module does not affect probability of failure), overall failure rate is the sum of failure rates of the modules**
- **Calculate FIT and MTTF for 10 disks (1M hour MTTF per disk), 1 disk controller (0.5M hour MTTF), and 1 power supply (0.2M hour MTTF):**

$$
\begin{aligned}
FailureRate &= 10 \times (1/1{,}000{,}000) + 1/500{,}000 + 1/200{,}000 \\
&= (10 + 2 + 5)/1{,}000{,}000 \\
&= 17/1{,}000{,}000 \\
&= 17{,}000\,FIT \\
MTTF &= 1{,}000{,}000{,}000/17{,}000 \\
&\approx 59{,}000\,hours
\end{aligned}
$$

Outline

- Review
- Technology Trends: Culture of tracking, anticipating and exploiting advances in technology
- **Careful, quantitative comparisons:**

1. Define and quantify power
2. Define and quantify dependability
3. **Define, quantify, and summarize relative performance**
4. **Define and quantify relative cost**

Definition: Performance

- Performance is in units of things per sec
  - bigger is better
- If we are primarily concerned with response time

$$\text{performance}(x) = \frac{1}{\text{execution\_time}(x)}$$

" X is n times faster than Y" means

$$n = \frac{\text{Performance}(X)}{\text{Performance}(Y)} = \frac{\text{Execution\_time}(Y)}{\text{Execution\_time}(X)}$$

Performance: What to measure

- Usually rely on benchmarks vs. real workloads
- To increase predictability, collections of benchmark applications, called *benchmark suites*, are popular
- SPECCPU: popular desktop benchmark suite
  - CPU only, split between integer and floating point programs
  - SPECint2000 has 12 integer, SPECfp2000 has 14 integer pgms
  - SPECCPU2006 to be announced Spring 2006
  - SPECSFS (NFS file server) and SPECWeb (WebServer) added as server benchmarks
- Transaction Processing Council measures server performance and cost-performance for databases
  - TPC-C Complex query for Online Transaction Processing
  - TPC-H models ad hoc decision support
  - TPC-W a transactional web benchmark
  - TPC-App application server and web services benchmark

How Summarize Suite Performance (1/5)

- Arithmetic average of execution time of all pgms?
  - But they vary by 4X in speed, so some would be more important than others in arithmetic average
- Could add a weight per program, but how pick weight?
  - Different companies want different weights for their products
- SPECRatio: Normalize execution times to reference computer, yielding a ratio proportional to performance =

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

How Summarize Suite Performance (2/5)

- If program SPECRatio on Computer A is 1.25 times bigger than Computer B, then

$$1.25 = \frac{SPECRatio_A}{SPECRatio_B} = \frac{\dfrac{ExecutionTime_{reference}}{ExecutionTime_A}}{\dfrac{ExecutionTime_{reference}}{ExecutionTime_B}}$$

$$= \frac{ExecutionTime_B}{ExecutionTime_A} = \frac{Performance_A}{Performance_B}$$

- **Note that when comparing 2 computers as a ratio, execution times on the reference computer drop out, so choice of reference computer is irrelevant**

How Summarize Suite Performance (3/5)

- **Since ratios, proper mean is geometric mean (SPECRatio unitless, so arithmetic mean meaningless)**

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

1. **Geometric mean of the ratios is the same as the ratio of the geometric means**
2. **Ratio of geometric means**
   = Geometric mean of performance ratios
   ⇒ choice of reference computer is irrelevant!

- **These two points make geometric mean of ratios attractive to summarize performance**

How Summarize Suite Performance (4/5)

- Does a single mean well summarize performance of programs in benchmark suite?
- Can decide if mean a good predictor by characterizing variability of distribution using standard deviation
- Like geometric mean, geometric standard deviation is multiplicative rather than arithmetic
- Can simply take the logarithm of SPECRatios, compute the standard mean and standard deviation, and then take the exponent to convert back:

$$GeometricMean = \exp\left(\frac{1}{n} \times \sum_{i=1}^{n} \ln\left(SPECRatio_i\right)\right)$$

$$GeometricStDev = \exp\left(StDev\left(\ln\left(SPECRatio_i\right)\right)\right)$$

How Summarize Suite Performance (5/5)

- **Standard deviation is more informative if know distribution has a standard form**
  - ***bell-shaped normal distribution*, whose data are symmetric around mean**
  - ***lognormal distribution*, where logarithms of data--not data itself--are normally distributed (symmetric) on a logarithmic scale**
- **For a lognormal distribution, we expect that**

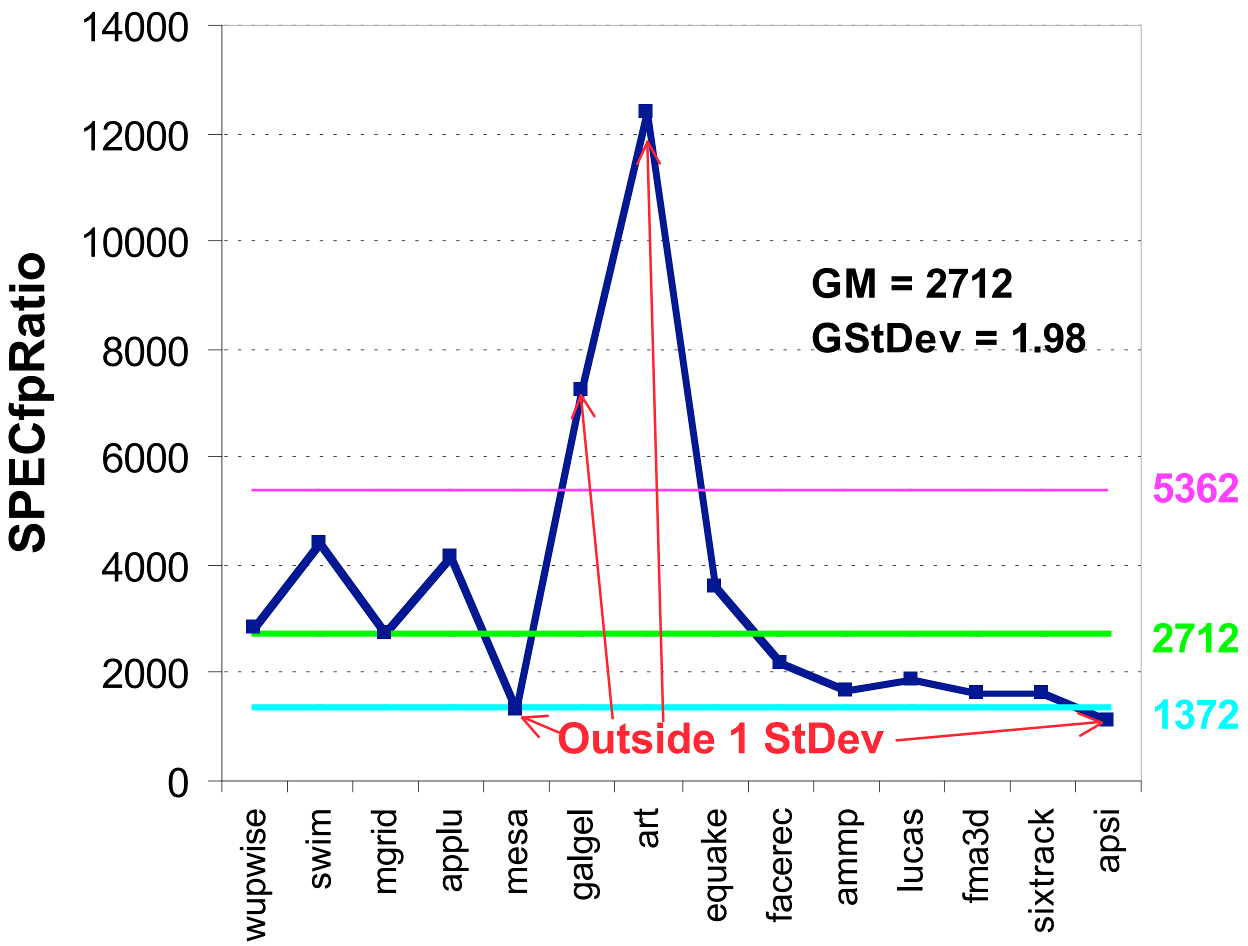
68% of samples fall in range $\left[mean / gstdev, mean \times gstdev\right]$

95% of samples fall in range $\left[mean / gstdev^2, mean \times gstdev^2\right]$

- **Note: Excel provides functions EXP(), LN(), and STDEV() that make calculating geometric mean and multiplicative standard deviation easy**

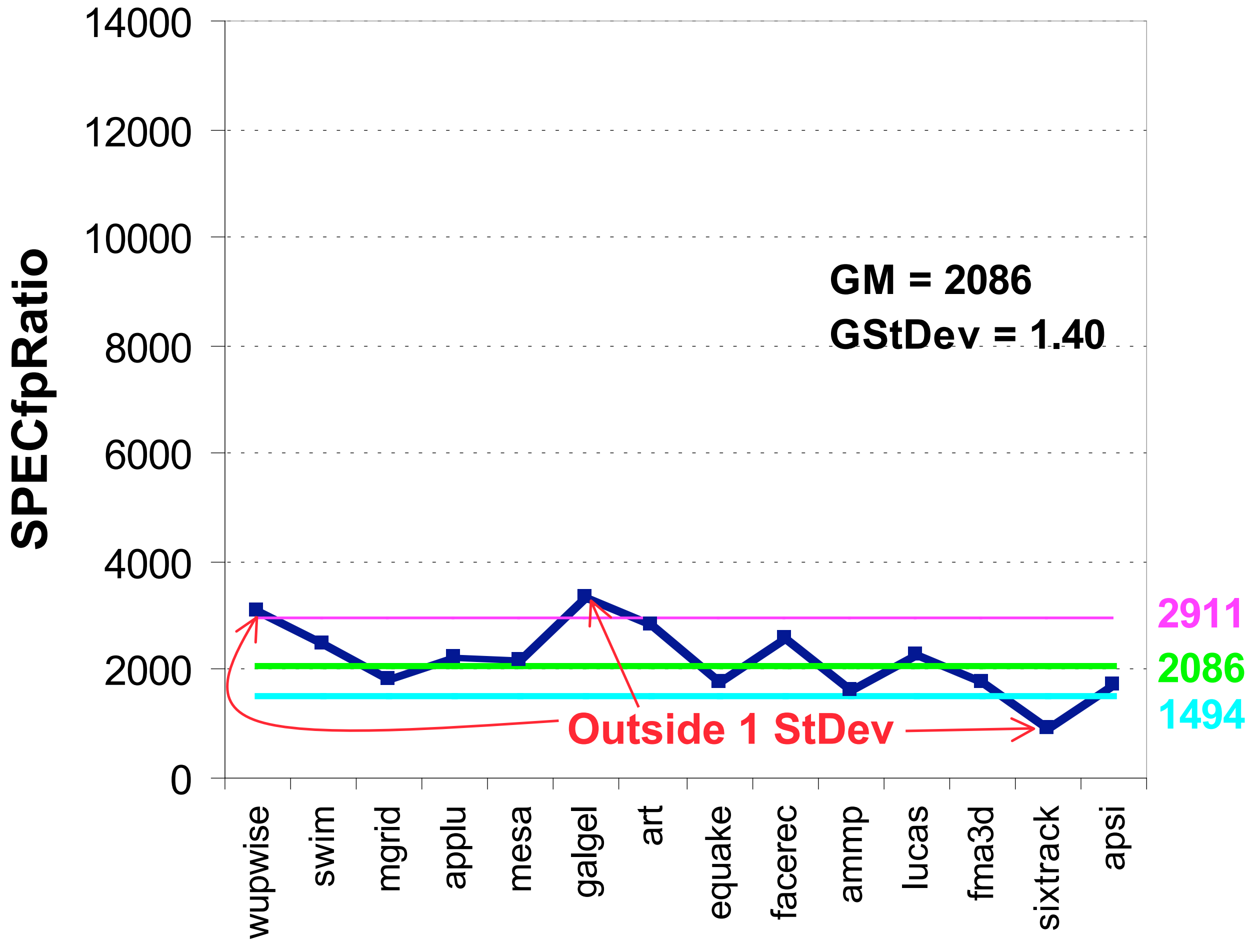
Example Standard Deviation (1/2)

- GM and multiplicative StDev of SPECfp2000 for Itanium 2



Example Standard Deviation (2/2)

- **GM and multiplicative StDev of SPECfp2000 for AMD Athlon**



Comments on Itanium 2 and Athlon

- Standard deviation of 1.98 for Itanium 2 is much higher-- vs. 1.40--so results will differ more widely from the mean, and therefore are likely less predictable
- Falling within one standard deviation:
  - 10 of 14 benchmarks (71%) for Itanium 2
  - 11 of 14 benchmarks (78%) for Athlon
- Thus, the results are quite compatible with a lognormal distribution (expect 68%)

And in conclusion …

- Tracking and extrapolating technology part of architect's responsibility
- Expect Bandwidth in disks, DRAM, network, and processors to improve by at least as much as the square of the improvement in Latency
- Quantify dynamic and static power
  - Capacitance x Voltage^2 x frequency, Energy vs. power
- Quantify dependability
  - Reliability (MTTF, FIT), Availability (99.9…)
- Quantify and summarize performance
  - Ratios, Geometric Mean, Multiplicative Standard Deviation
- Read Chapter 1, read Appendix A!