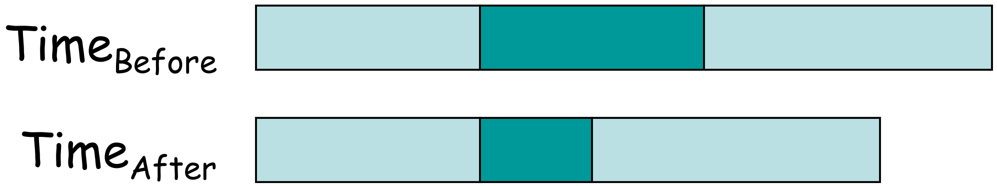
Amdahl's Law and Evaluating and Summarizing Performance

Based on slides created by Mark Hill and others

Slides adapted by Dr Sparsh Mittal

Compute Speedup – Amdahl's Law

Speedup is due to enhancement(E):

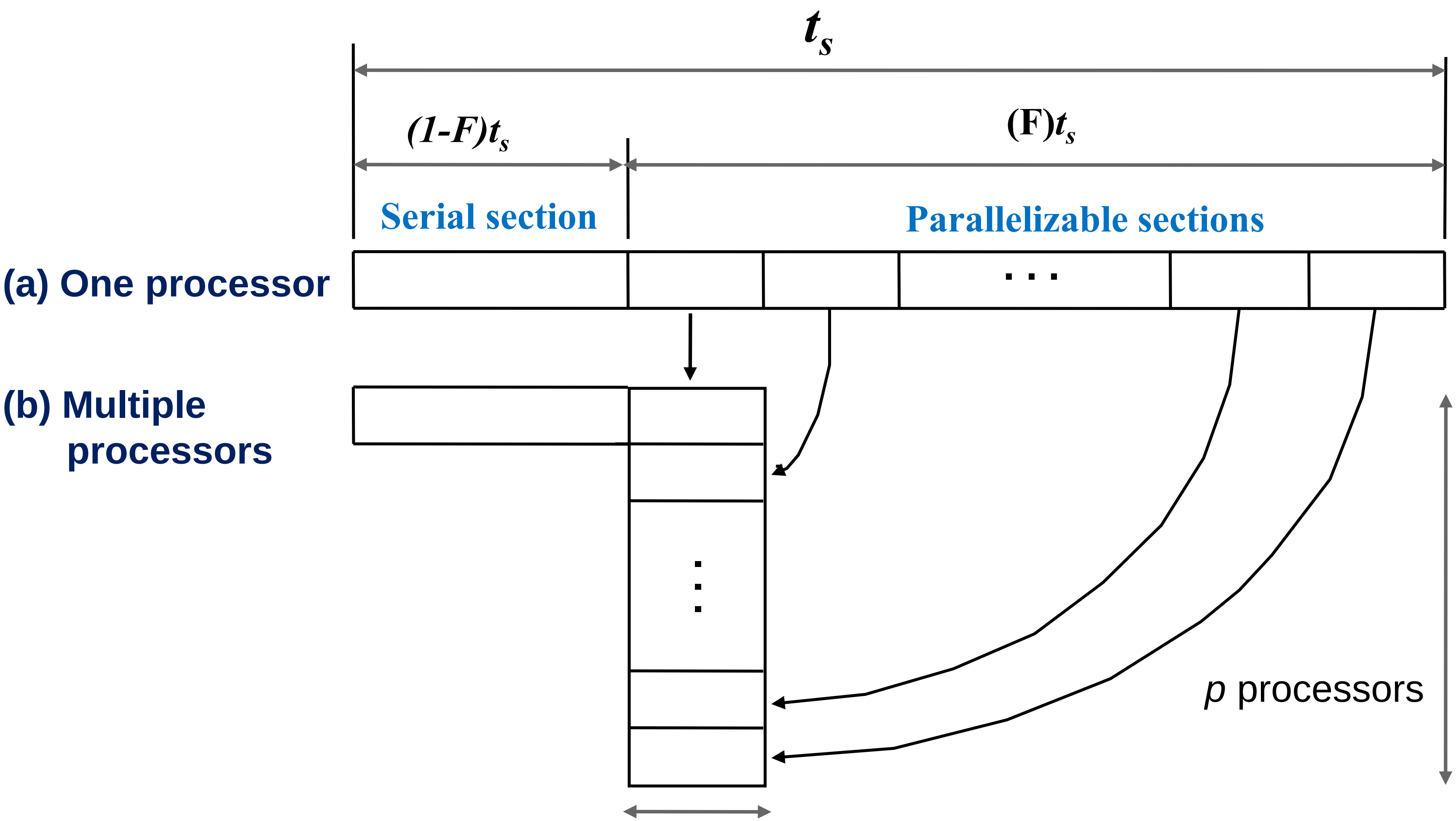


Let F be the fraction where enhancement is applied => Also, called parallel fraction and (1-F) as the serial fraction

$$\textit{Execution time}_{after} = ExTime_{before} \times [(1-F) + \frac{F}{S}]$$

$$\textit{Speedup(E)} = \frac{ExTime_{before}}{ExTime_{after}} = \frac{1}{[(1-F) + \frac{F}{S}]}$$

Even with infinite number of processors, maximum speedup is limited to 1/(1-F).



t_s
$(1-F)t_s$
$(F)t_s$
Serial section
Parallelizable sections
(a) One processor
(b) Multiple processors
p processors

Performance

- Which computer is fastest?
- Not so simple

Response Time vs. Throughput

- Is throughput = 1/av. response time?

  - Only if NO overlap
  - Otherwise, throughput > 1/av. response time

Principles of Computer Design

$$\text{CPU time} = \text{CPU clock cycles for a program} \times \text{Clock cycle time}$$

$$\text{CPU time} = \frac{\text{CPU clock cycles for a program}}{\text{Clock rate}}$$

$$\text{CPI} = \frac{\text{CPU clock cycles for a program}}{\text{Instruction count}}$$

$$\text{CPU time} = \text{Instruction count} \times \text{Cycles per instruction} \times \text{Clock cycle time}$$

$$\frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}} = \frac{\text{Seconds}}{\text{Program}} = \text{CPU time}$$

Principles of Computer Design

- Different instruction types having different CPIs

$$\text{CPU clock cycles} = \sum_{i=1}^{n} \text{IC}_i \times \text{CPI}_i$$

$$\text{CPU time} = \left(\sum_{i=1}^{n} \text{IC}_i \times \text{CPI}_i \right) \times \text{Clock cycle time}$$

Measure of Performance

$$\textbf{Processor Performance} = \frac{\textbf{Time}}{\textbf{Program}}$$

$$= \frac{\textbf{Instructions}}{\textbf{Program}} \times \frac{\textbf{Cycles}}{\textbf{Instruction}} \times \frac{\textbf{Time}}{\textbf{Cycle}}$$

(code size) **(CPI)** **(cycle time)**

Architecture --> Implementation --> Realization

Compiler Designer **Processor Designer** **Chip Designer**

Our Goal

- Minimize time which is the product, NOT isolated terms
- Common error to miss terms while devising optimizations
  - E.g. ISA change to decrease instruction count
  - BUT leads to CPU organization which makes clock slower
- Bottom line: terms are inter-related

ISA = instruction set architecture

Other Metrics

- MIPS and MFLOPS
- MIPS = instruction count/(execution time x 10^6)

  = clock rate/(CPI x 10^6)
- But MIPS has serious shortcomings

Problems with MIPS

- E.g. without FP hardware, an FP op may take 50 single-cycle instructions
- With FP hardware, only one 2-cycle instruction

- Thus, adding FP hardware:
  - Total execution time decreases
- BUT, MIPS gets worse!

Problems with MIPS

- It ignores program
- When is MIPS ok?
  - Same compiler, same ISA
  - E.g. same binary running on Pentium-III, IV
  - Why? Instr/program is constant and can be ignored

Other Metrics

- MFLOPS = FP ops in program/(execution time x 10^6)
- Assuming FP ops independent of compiler and ISA
  - Often safe for numeric codes: matrix size determines # of FP ops/program
  - However, not always safe:
    - Missing instructions (e.g. FP divide, sqrt/sin/cos)
    - Optimizing compilers
- Relative MIPS and normalized MFLOPS
  - Normalized to some common baseline machine
    - E.g. VAX MIPS in the 1980s

Rules

- Use ONLY Time
- Beware of ***Peak***
  - Guaranteed not to exceed

Example

- Machine A: clock 1ns, CPI 2.0, for program x
- Machine B: clock 2ns, CPI 1.2, for program x
- Which is faster and how much?

  Time/Program = instr/program x cycles/instr x sec/cycle

  Time(A) = N x 2.0 x 1 = 2N

  Time(B) = N x 1.2 x 2 = 2.4N

  Compare: Time(B)/Time(A) = 2.4N/2N = 1.2

- So, Machine A is 20% faster than Machine B for this program

SUMMARIZING PERFORMANCE

Summarizing Performance

- Indices of central tendency
  - Sample mean
  - Median
  - Mode
- Other means
  - Arithmetic
  - Harmonic
  - Geometric
- Quantifying variability

Why mean values?

- Desire to reduce performance to a single number
  - Makes comparisons easy
  - People like a measure of “typical” performance

The Problem

- Performance is multidimensional
  - CPU time
  - I/O time
  - Network time
  - Interactions of various components
  - etc, etc

The Problem

- Systems are often specialized
  - Performs great on application type X
  - Performs lousy on anything else
- Potentially a wide range of execution times on one system using different benchmark programs

The Problem

- Nevertheless, people still want a single number answer!
- *How to (correctly) summarize a wide range of measurements with a single value?*

Index of Central Tendency

- Tries to capture “center” of a distribution of values
- Use this “center” to summarize overall behavior
- Not recommended for real information, but …
  - You will be pressured to provide mean values
    - Understand how to choose the best type for the circumstance
    - Be able to detect bad results from others

Indices of Central Tendency

- Sample mean
  - Common “average”
- Sample median
  - ½ of the values are above, ½ below
- Mode
  - Most common value

Sample mean

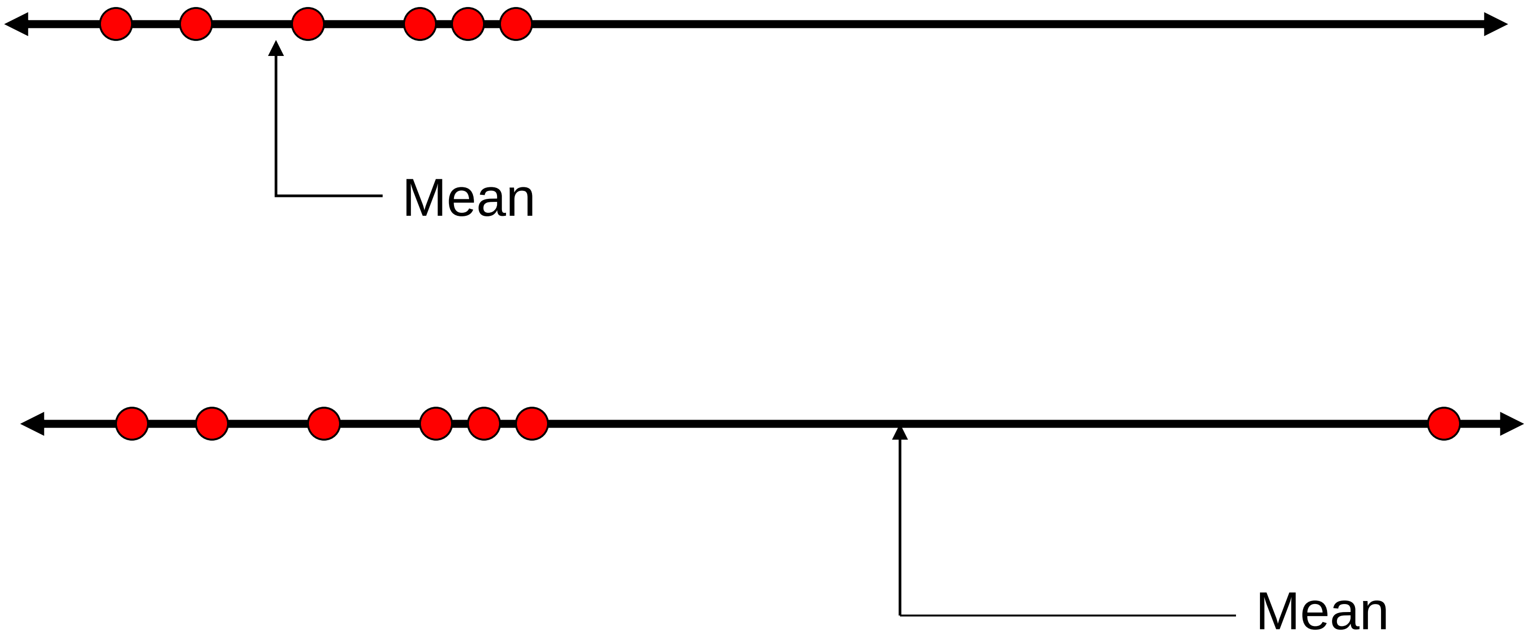
- assume
  - n = number of measurements
- ***Arithmetic mean***
  - Common “average”

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Potential Problem with Means

- Sample mean gives equal weight to all measurements
- *Outliers* can have a large influence on the computed mean value
- Distorts our intuition about the *central tendency* of the measured values

Potential Problem with Means



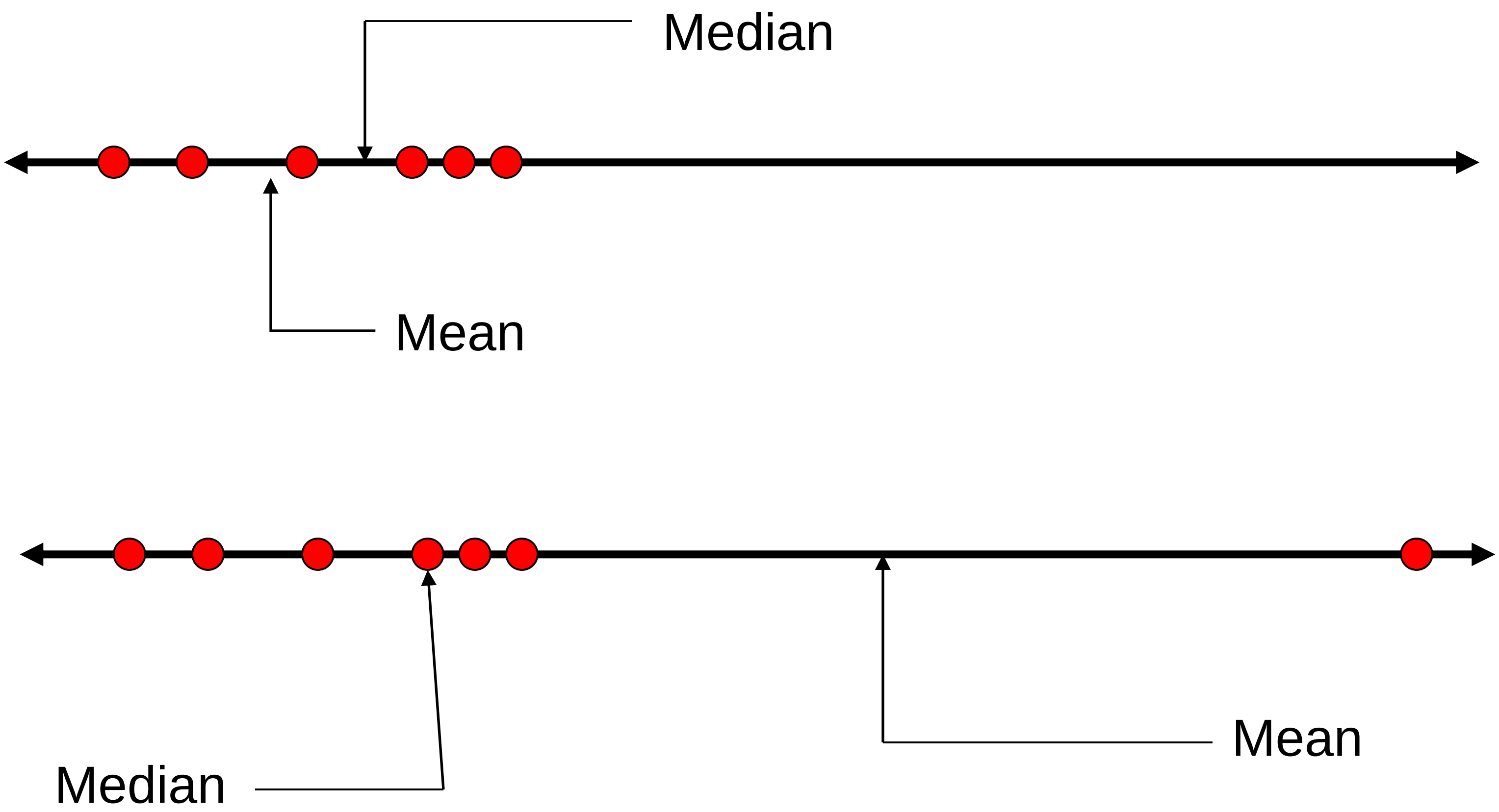
Median

- Index of central tendency with
  - ½ of the values larger, ½ smaller
- Sort *n* measurements
- If *n* is odd
  - Median = middle value
  - Else, median = mean of two middle values
- *Reduces skewing effect of outliers* on the value of the index

Example

- Measured values: 10, 20, 15, 18, 16
  - Mean = 15.8
  - Median = 16
- Obtain one more measurement: 200
  - Mean = 46.5
  - Median = ½ (16 + 18) = 17
- Median give more intuitive sense of central tendency

Potential Problem with Means



Mode

- Value that occurs most often
- May not exist
- May not be unique
  – E.g. “bi-modal” distribution
    - Two values occur with same frequency

Mean, Median, or Mode?

- Mean
  - If the sum of all values is meaningful
  - Incorporates all available information
- Median
  - Intuitive sense of central tendency with outliers
  - What is “typical” of a set of values?
- Mode
  - When data can be grouped into distinct types, categories (*categorical data*)

Arithmetic mean

$$\overline{x_A} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Harmonic mean

$$\overline{x_H} = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$$

Geometric mean

$$\overline{x_G} = \sqrt[n]{x_1 x_2 \cdots x_i \cdots x_n}$$

$$= \left(\prod_{i=1}^{n} x_i \right)^{1/n}$$

Weighted means

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

$$\bar{x}_A = \sum_{i=1}^{n} w_i x_i$$

$$\bar{x}_H = \frac{1}{\sum_{i=1}^{n} \frac{w_i}{x_i}}$$

- Standard definition of mean assumes all measurements are equally important
- Instead, choose weights to represent relative importance of measurement *i*

Which is the right mean?

- Arithmetic (AM)?
- Harmonic (HM)?
- Geometric (GM)?
- WAM, WHM, WGM?
- Which one should be used when?

Which mean to use?

- Mean value must still conform to characteristics of a *good* performance metric
  - Linear
  - Reliable
  - Repeatable
  - Easy to use
  - Consistent
  - Independent
- Best measure of performance still is *execution time*

What makes a good mean?

- *Time*–based mean (e.g. seconds)
  - Should be *directly proportional* to total weighted time
  - If time doubles, mean value should double
- *Rate*–based mean (e.g. operations/sec)
  - Should be *inversely proportional* to total weighted time
  - If time doubles, mean value should reduce by half
- Which means satisfy these criteria?

Assumptions

- Measured execution times of *n* benchmark programs
  - T_i, i = 1, 2, …, *n*
- Total work performed by each benchmark is constant
  - F = # operations performed
  - Relax this assumption later
- Execution rate = $M_i = F / T_i$

Arithmetic mean for times

- Produces a mean value that is *directly proportional to total time*

$\rightarrow$ Correct mean to summarize *execution time*

$$\overline{T_A} = \frac{1}{n}\sum_{i=1}^{n} T_i$$

Arithmetic mean for rates

- Produces a mean value that is proportional to *sum of inverse of times*
- But we want *inversely proportional to sum of times*

$$\overline{M_A} = \frac{1}{n}\sum_{i=1}^{n} M_i$$

$$= \sum_{i=1}^{n} \frac{F / T_i}{n}$$

$$= \frac{F}{n}\sum_{i=1}^{n} \frac{1}{T_i}$$

Arithmetic mean for rates

- Produces a mean value that is proportional to *sum of inverse of times*
- But we want *inversely proportional to sum of times*

→ Arithmetic mean is ***NOT*** appropriate for summarizing rates

$$\overline{M_A} = \frac{1}{n}\sum_{i=1}^{n} M_i = \sum_{i=1}^{n} \frac{F/T_i}{n} = \frac{F}{n}\sum_{i=1}^{n}\frac{1}{T_i}$$

Harmonic mean for times

- Not directly proportional to *sum of times*

$$\overline{T_H} = \frac{n}{\sum_{i=1}^{n} \frac{1}{T_i}}$$

Other Averages

- E.g., drive 30 mph for first 10 miles, then 90 mph for next 10 miles, what is average speed?
- Average speed = (30+90)/2 **WRONG**
- Average speed = total distance / total time

  = (20 / (10/30 + 10/90))

  = 45 mph
- When dealing with *rates* (mph) do not use arithmetic mean

Harmonic mean for times

- Not directly proportional to *sum of times*

→ Harmonic mean is ***not*** appropriate for summarizing times

$$\overline{T}_H = \frac{n}{\sum_{i=1}^{n} \frac{1}{T_i}}$$

Harmonic mean for rates

- Produces

  (total number of ops)

  ÷ (sum execution times)

- Inversely proportional to total execution time

→ Harmonic mean is appropriate to summarize rates

$$\overline{M_H} = \frac{n}{\sum_{i=1}^{n} \frac{1}{M_i}}$$

$$= \frac{n}{\sum_{i=1}^{n} \frac{T_i}{F}}$$

$$= \frac{Fn}{\sum_{i=1}^{n} T_i}$$

Harmonic mean for rates

Sec	10^9 FLOPs	MFLOPS
321	130	405(=130000/321)
436	160	367
284	115	405
601	252	419
482	187	388

$$\overline{M_H} = \frac{5}{\left(\frac{1}{405}+\frac{1}{367}+\frac{1}{405}+\frac{1}{419}+\frac{1}{388}\right)}$$

$$= 396$$

$$\overline{M_H} = \frac{844 \times 10^9}{2124} = 396$$

130+160+115+252+187 = 844
321+436+284+601+482 = 2124

Geometric Mean

- Use geometric mean for ratios
- Geometric mean of ratios = $\sqrt[n]{\prod_{i=1}^{n} ratio(i)}$

- Independent of reference machine

Geometric mean

- Correct mean for averaging normalized values
- Good when averaging measurements with wide range of values
- Maintains consistent relationships when comparing normalized values
  - Independent of basis used to normalize

Geometric mean with times

	System 1	System 2	System 3
	417	244	134
	83	70	70
	66	153	135
	39,449	33,527	66,000
	772	368	369
Product	6.95E+13	3.22E+13	3.08E+13
Geo mean	587	503	499
Rank	3	2	1

Geometric mean normalized to System 1

	System 1	System 2	System 3
	1.0	0.59	0.32
	1.0	0.84	0.85
	1.0	2.32	2.05
	1.0	0.85	1.67
	1.0	0.48	0.45
Geo mean	1.0	0.86	0.84
Rank	3	2	1

Geometric mean normalized to System 2

	System 1	System 2	System 3
	1.71	1.0	0.55
	1.19	1.0	1.0
	0.43	1.0	0.88
	1.18	1.0	1.97
	2.10	1.0	1.0
Geo mean	1.17	1.0	0.99
Rank	3	2	1

Sum of execution times

	System 1	System 2	System 3
	417	244	134
	83	70	70
	66	153	135
	39,449	33,527	66,000
	772	368	369
Sum	40,787	34,362	66,798
Arith mean	8157	6872	13,342
Rank	2	1	3

What's going on here?!

	System 1	System 2	System 3
Geo mean wrt 1	1.0	0.86	0.84
Rank	3	2	1
Geo mean wrt 2	1.17	1.0	0.99
Rank	3	2	1
Arith mean	8157	6872	13,342
Rank	2	1	3

Geometric mean for times

- Not directly proportional to *sum of times*

$$\overline{T_G} = \left(\prod_{i=1}^{n} T_i \right)^{1/n}$$

Geometric mean for times

- Not directly proportional to *sum of times*

→ Geometric mean is ***not*** appropriate for summarizing times

$$\overline{T_G} = \left(\prod_{i=1}^{n} T_i \right)^{1/n}$$

Geometric mean for rates

- Not inversely proportional to *sum of times*

$$\overline{T_G} = \left(\prod_{i=1}^{n} M_i \right)^{1/n}$$

$$= \left(\prod_{i=1}^{n} \frac{F}{T_i} \right)^{1/n}$$

Geometric mean for rates

- Not inversely proportional to *sum of times*

→ Geometric mean is ***not*** appropriate for summarizing rates

$$\overline{T_G} = \left(\prod_{i=1}^{n} M_i \right)^{1/n} = \left(\prod_{i=1}^{n} \frac{F}{T_i} \right)^{1/n}$$

Summary of Means

- Avoid means if possible
  - Loses information
- Arithmetic
  - When sum of raw values has physical meaning
  - Use for summarizing **times** (not rates)
- Harmonic
  - Use for summarizing **rates** (not times)
- Geometric mean
  - Not useful when *time* is best measure of perf

Geometric mean

- Does provide consistent rankings
  - Independent of basis for normalization
- But can be consistently wrong!
- Value can be computed
  - But has no physical meaning

AM? GM? HM? WAM? WHM? WGM? What are the Weights??????

Measure	Valid central tendency for summarized measure over the suite	
IPC		
CPI		
Speedup		
MIPS		
MFLOPS		
Cache hit rate		
Cache misses per instruction		
Branch misprediction rate per branch		
Normalized execution time		
Transactions per minute		
A/B		

Conditions under which unweighted arithmetic and harmonic means are valid indicators of overall performance

	To summarize measure over the suite	
Measure	When is AM valid?	When is H.M. valid?
A/B	If B's are equal	If A's are equal
IPC	If equal cycles in each benchmark	If equal work (I-count) in each benchmark
CPI	If equal I-count in each benchmark	If equal cycles in each benchmark
Speedup	If equal execution times in each benchmark in the improved system	If equal execution times in each benchmark in the baseline system
MIPS	If equal times in each benchmark	If equal I-count in each benchmark
MFLOPS	If equal times in each benchmark	If equal FLOPS in each benchmark
Cache hit rate	If equal number of references to cache for each benchmark	If equal number of cache hits in each benchmark
Cache misses per instruction	If equal I-count in each benchmark	If equal number of misses in each benchmark
Branch misprediction rate per branch	If equal number of branches in each benchmark	If equal number of mispredictions in each benchmark
Normalized execution time	If equal execution times in each benchmark in the system considered as base	If equal execution times in each benchmark in the system being evaluated
Transactions per minute	If equal times in each benchmark	If equal number of transactions in each benchmark

The mean to be used to find aggregate measure over a benchmark suite from measures corresponding to individual benchmarks in the suite

Measure	Valid central tendency for summarized measure over the suite	
IPC	W.A.M. weighted with cycles	W.H.M. weighted with I-count
CPI	W.A.M. weighted with I-count	W.H.M. weighted with cycles
Speedup	W.A.M. weighted with execution time ratios in improved system	W.H.M. weighted with execution time ratios in the baseline system
MIPS	W.A.M. weighted with time	W.H.M. weighted with I-count
MFLOPS	W.A.M. weighted with time	W.H.M. weighted with FLOP count
Cache hit rate	W.A.M. weighted with number of references to cache	W.H.M. weighted with number of hits
Cache misses per instruction	W.A.M. weighted with I-count	W.H.M weighted with number of misses
Branch misprediction rate per branch	W.A.M. weighted with branch counts	W.H.M. weighted with number of mispredictions
Normalized execution time	W.A.M. weighted with execution times in system considered as base	W.H.M. weighted with execution times in the system being evaluated
Transactions per minute	W.A.M. weighted with exec times	W.H.M. weighted with proportion of transactions for each benchmark
A/B	W.A.M. weighted with B's	W.H.M. weighted with A's