Performance Metrics for Computer Systems

CASS 2018

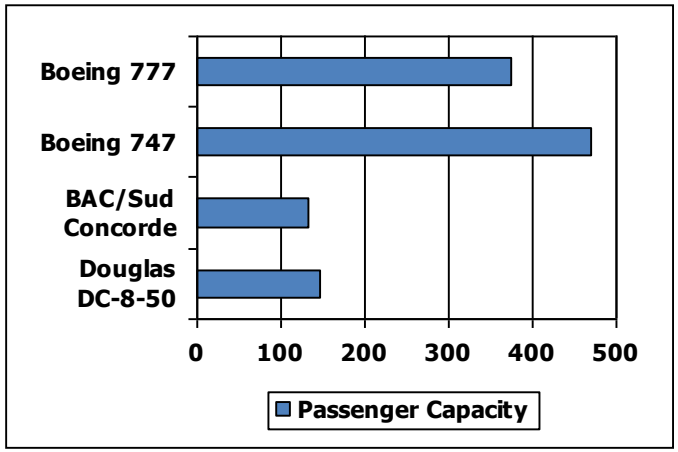
Lavanya Ramapantulu

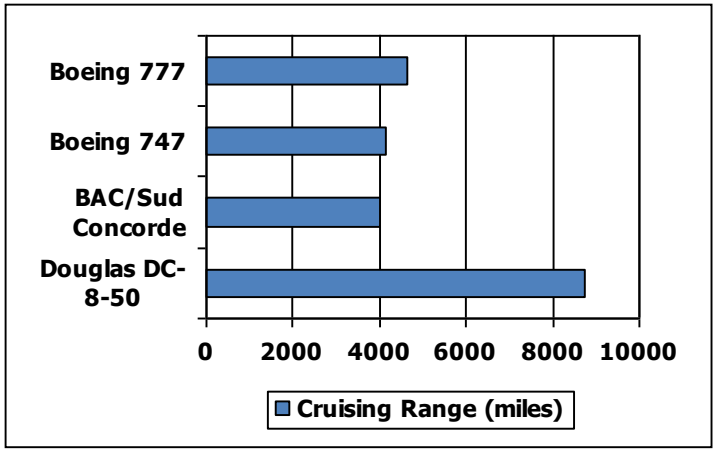
Eight Great Ideas in Computer Architecture

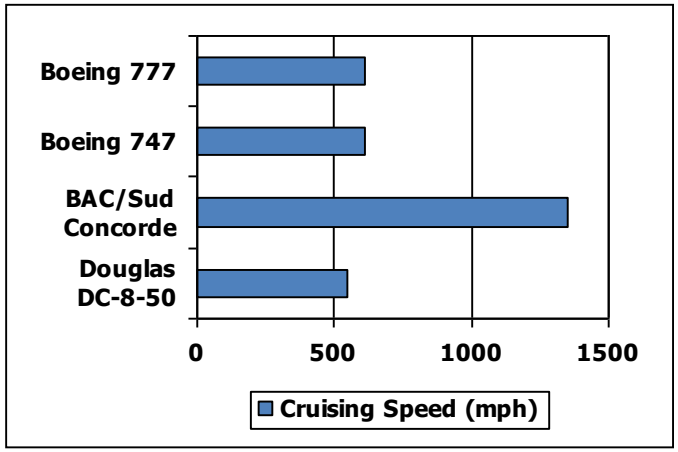
- Design for ***Moore's Law***
- Use ***abstraction*** to simplify design
- Make the ***common case fast***
- Performance *via* ***parallelism***
- Performance *via* ***pipelining***
- Performance *via* ***prediction***
- ***Hierarchy*** of memories
- ***Dependability*** *via* redundancy

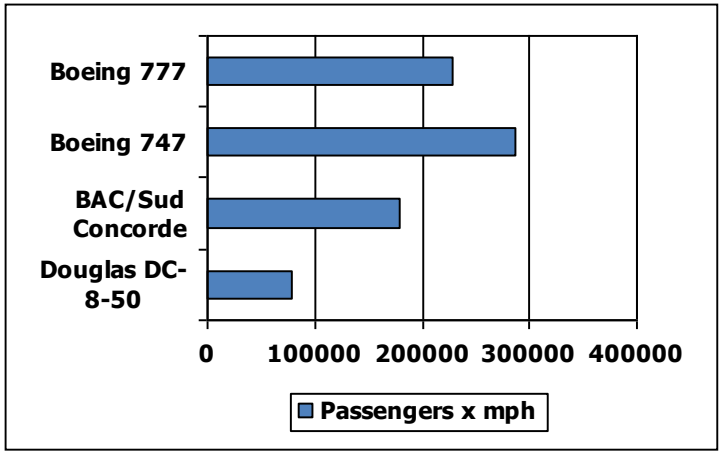
Defining Performance

- Which airplane has the best performance?









Response Time and Throughput

- Response time
  - How long it takes to do a task
- Throughput
  - Total work done per unit time
    - e.g., tasks/transactions/… per hour
- How are response time and throughput affected by
  - Replacing the processor with a faster version?
  - Adding more processors?
- We'll focus on response time for now…

Relative Performance

- Define Performance = 1/Execution Time
- "X is n time faster than Y"

$$\text{Performance}_X / \text{Performance}_Y$$
$$= \text{Execution time}_Y / \text{Execution time}_X = n$$

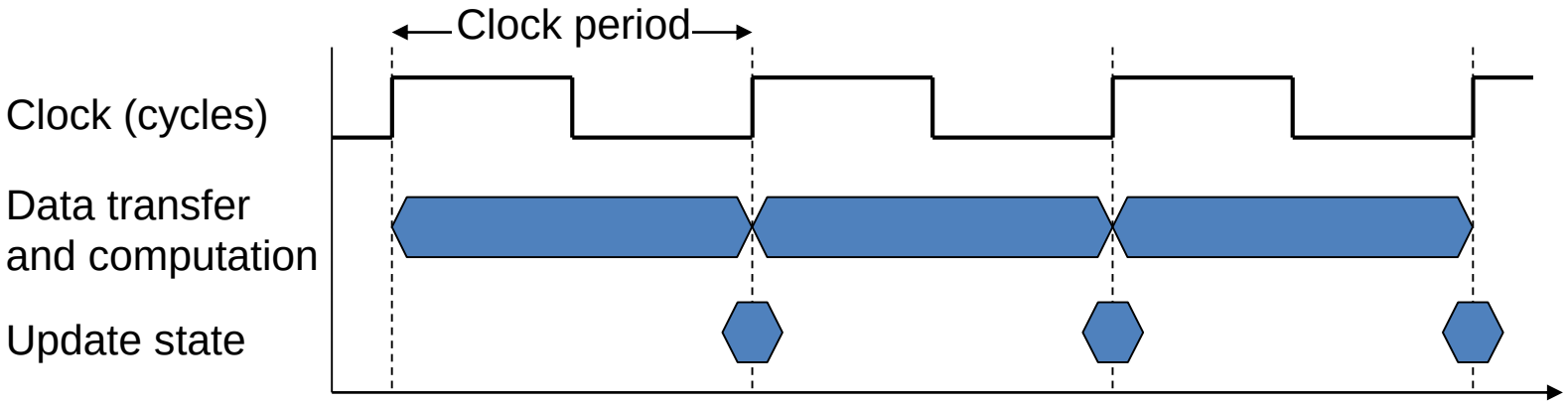
- Example: time taken to run a program
  - 10s on A, 15s on B
  - Execution Time_B / Execution Time_A
    = 15s / 10s = 1.5
  - So A is 1.5 times faster than B

Measuring Execution Time

- Elapsed time
  - Total response time, including all aspects
    - Processing, I/O, OS overhead, idle time
  - Determines system performance
- CPU time
  - Time spent processing a given job
    - Discounts I/O time, other jobs' shares
  - Comprises user CPU time and system CPU time
  - Different programs are affected differently by CPU and system performance

CPU Clocking

- Operation of digital hardware governed by a constant-rate clock



  - Clock period: duration of a clock cycle
    - e.g., 250ps = 0.25ns = 250×10^{-12}s
  - Clock frequency (rate): cycles per second
    - e.g., 4.0GHz = 4000MHz = 4.0×10^{9}Hz

CPU Time

$$\text{CPU Time} = \text{CPU Clock Cycles} \times \text{Clock Cycle Time}$$

$$= \frac{\text{CPU Clock Cycles}}{\text{Clock Rate}}$$

- Performance improved by
  - Reducing number of clock cycles
  - Increasing clock rate
  - Hardware designer must often trade off clock rate against cycle count

CPU Time Example

- Computer A: 2GHz clock, 10s CPU time
- Designing Computer B
  - Aim for 6s CPU time
  - Can do faster clock, but causes 1.2 × clock cycles
- How fast must Computer B clock be?

$$\text{Clock Rate}_B = \frac{\text{Clock Cycles}_B}{\text{CPU Time}_B} = \frac{1.2 \times \text{Clock Cycles}_A}{6\text{s}}$$

$$\text{Clock Cycles}_A = \text{CPU Time}_A \times \text{Clock Rate}_A$$

$$= 10\text{s} \times 2\text{GHz} = 20 \times 10^9$$

$$\text{Clock Rate}_B = \frac{1.2 \times 20 \times 10^9}{6\text{s}} = \frac{24 \times 10^9}{6\text{s}} = 4\text{GHz}$$

Instruction Count and CPI

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{Cycles per Instruction}$$

$$\text{CPU Time} = \text{Instruction Count} \times \text{CPI} \times \text{Clock Cycle Time}$$

$$= \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Rate}}$$

- Instruction Count for a program
  - Determined by program, ISA and compiler
- Average cycles per instruction
  - Determined by CPU hardware
  - If different instructions have different CPI
    - Average CPI affected by instruction mix

CPI Example

- Computer A: Cycle Time = 250ps, CPI = 2.0
- Computer B: Cycle Time = 500ps, CPI = 1.2
- Same ISA
- Which is faster, and by how much?

$$\text{CPU Time}_A = \text{Instruction Count} \times \text{CPI}_A \times \text{Cycle Time}_A$$

$$= I \times 2.0 \times 250\text{ps} = I \times 500\text{ps}$$

A is faster…

$$\text{CPU Time}_B = \text{Instruction Count} \times \text{CPI}_B \times \text{Cycle Time}_B$$

$$= I \times 1.2 \times 500\text{ps} = I \times 600\text{ps}$$

$$\frac{\text{CPU Time}_B}{\text{CPU Time}_A} = \frac{I \times 600\text{ps}}{I \times 500\text{ps}} = 1.2$$

…by this much

CPI in More Detail

- If different instruction classes take different numbers of cycles

$$\text{Clock Cycles} = \sum_{i=1}^{n} (\text{CPI}_i \times \text{Instruction Count}_i)$$

  - Weighted average CPI

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}} = \sum_{i=1}^{n} \left(\text{CPI}_i \times \underbrace{\frac{\text{Instruction Count}_i}{\text{Instruction Count}}}_{\text{Relative frequency}} \right)$$

CPI Example

- Alternative compiled code sequences using instructions in classes A, B, C

Class	A	B	C
CPI for class	1	2	3
IC in sequence 1	2	1	2
IC in sequence 2	4	1	1

- Sequence 1: IC = 5
  - Clock Cycles
    = 2×1 + 1×2 + 2×3
    = 10
  - Avg. CPI = 10/5 = 2.0
- Sequence 2: IC = 6
  - Clock Cycles
    = 4×1 + 1×2 + 1×3
    = 9
  - Avg. CPI = 9/6 = 1.5

Performance Summary

The BIG Picture

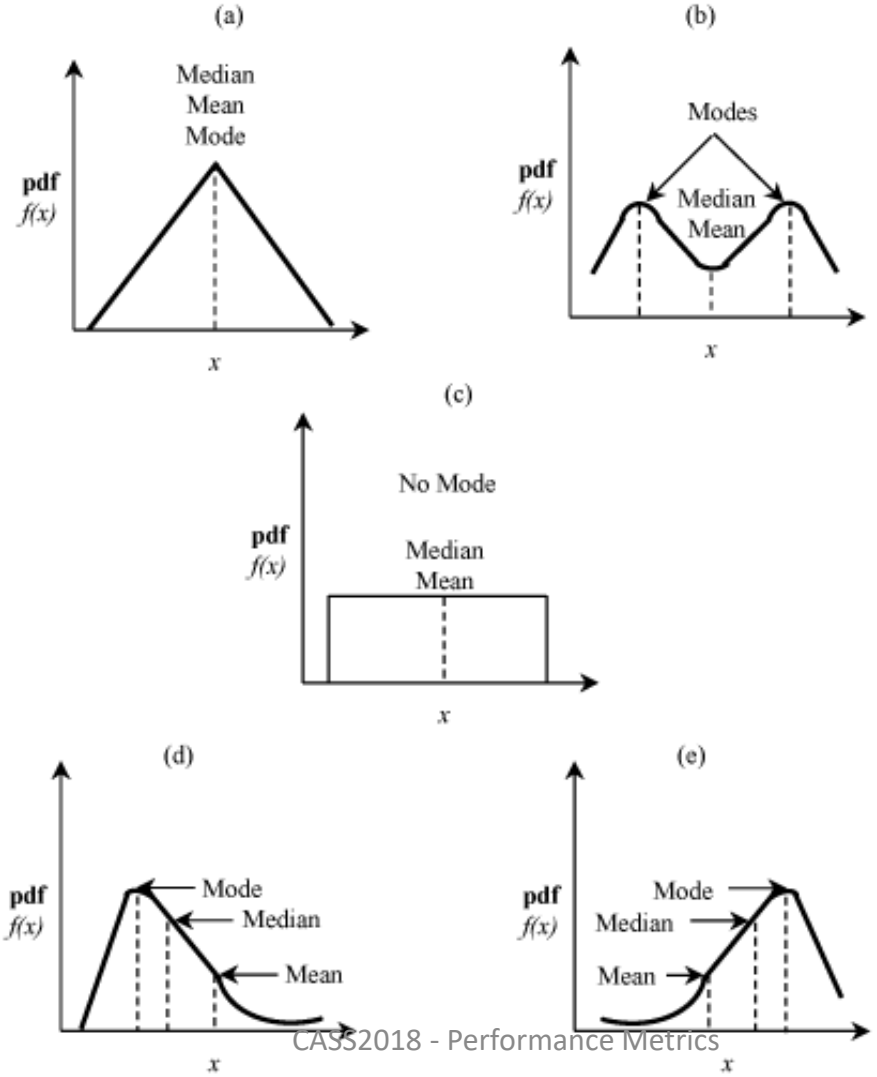
$$\text{CPU Time} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instruction}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

- Performance depends on
  - Algorithm: affects IC, possibly CPI
  - Programming language: affects IC, CPI
  - Compiler: affects IC, CPI
  - Instruction set architecture: affects IC, CPI, T_c

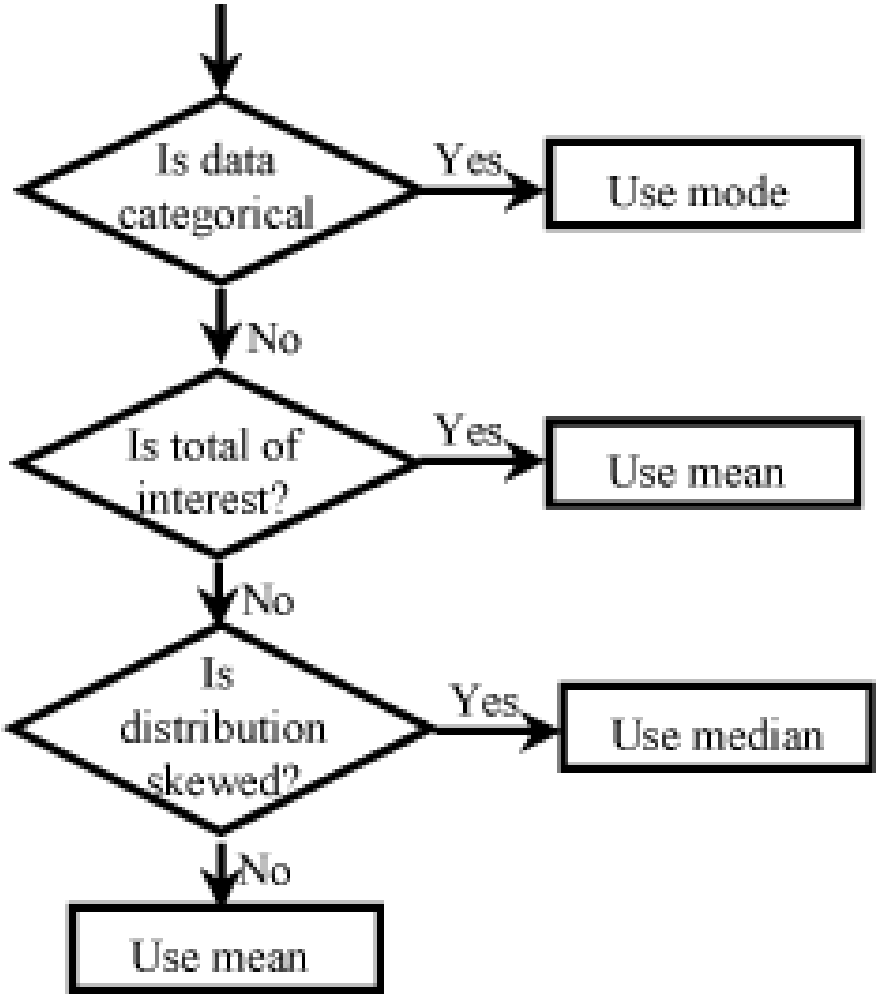
Summarizing Data by a Single Number

- **Indices of central tendencies**: Mean, Median, Mode
- **Sample Mean** is obtained by taking the sum of all observations and dividing this sum by the number of observations in the sample.
- **Sample Median** is obtained by sorting the observations in an increasing order and taking the observation that is in the middle of the series. If the number of observations is even, the mean of the middle two values is used as a median.
- **Sample Mode** is obtained by plotting a histogram and specifying the midpoint of the bucket where the histogram peaks. For categorical variables, mode is given by the category that occurs most frequently.
- Mean and median always exist and are unique. Mode, on the other hand, may not exist.

Mean, Median, and Mode: Relationships



Selecting Mean, Median, and Mode



Indices of Central Tendencies: Examples

- Most used resource in a system: Resources are categorical and hence mode must be used.
- Interarrival time: Total time is of interest and so mean is the proper choice.
- Load on a Computer: Median is preferable due to a highly skewed distribution.
- Average Configuration: Medians of number devices, memory sizes, number of processors are generally used to specify the configuration due to the skewness of the distribution.

Common Misuses of Means

- ***Using mean of significantly different values***:
  (10+1000)/2 = 505
- ***Using mean without regard to the skewness of distribution***.

System A	System B
10	5
9	5
11	5
10	4
10	31
Sum=50	Sum=50
Mean=10	Mean=10
Typical=10	Typical=5

Misuses of Means (cont)

- ***Multiplying means to get the mean of a product***

$$E(xy) \neq E(x)E(y)$$

- Example: On a timesharing system,
  Average number of users is 23
  Average number of sub-processes per user is 2
  What is the average number of sub-processes?
  Is it 46? No!
  The number of sub-processes a user spawns depends upon how much load there is on the system.
- ***Taking a mean of a ratio with different bases***.

Geometric Mean

$$\dot{x} = \left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}}$$

- Geometric mean is used if the product of the observations is a quantity of interest.

Geometric Mean: Example

- The performance improvements in 7 layers:

Protocol Layer	Performance Improvement
7	18%
6	13%
5	11%
4	8%
3	10%
2	28%
1	5%

$$\text{Average improvement per layer}$$
$$= \{(1.18)(1.13)(1.11)(1.08)(1.10)(1.28)(1.05)\}^{\frac{1}{7}} - 1$$
$$= 0.13$$

Examples of Multiplicative Metrics

- Cache hit ratios over several levels of caches
- Cache miss ratios
- Percentage performance improvement between successive versions
- Average error rate per hop on a multi-hop path in a network.

Geometric Mean of Ratios

$$gm\left(\frac{x_1}{y_1}, \frac{x_2}{y_2}, \ldots, \frac{x_n}{y_n}\right) = \frac{gm(x_1, x_2, \cdots, x_n)}{gm(y_1, y_2, \ldots, y_n)} = \frac{1}{gm\left(\frac{y_1}{x_1}, \frac{y_2}{x_2}, \ldots, \frac{y_n}{x_n}\right)}$$

- The geometric mean of a ratio is the ratio of the geometric means of the numerator and denominator
  => the choice of the base does not change the conclusion.
- It is because of this property that sometimes geometric mean is recommended for ratios.
- However, if the geometric mean of the numerator or denominator do not have any physical meaning, the geometric mean of their ratio is meaningless as well.

Harmonic Mean

$$\ddot{x} = \frac{n}{\frac{1}{x_1} + \frac{1}{x_2} + \cdots + \frac{1}{x_n}}$$

- Used whenever an arithmetic mean can be justified for $1/x_i$
  E.g., Elapsed time of a benchmark on a processor
- In the i^{th} repetition, the benchmark takes t_i seconds. Now suppose the benchmark has m million instructions, MIPS x_i computed from the i^{th} repetition is: $x_i = \frac{m}{t_i}$

- t_i's should be summarized using arithmetic mean since the sum of t_i has a physical meaning
  => x_i's should be summarized using harmonic mean since the sum of $1/x_i$'s has a physical meaning.

Harmonic Mean (Cont)

- The average MIPS rate for the processor is:

$$\ddot{x} = \frac{n}{\frac{1}{m/t_1} + \frac{1}{m/t_2} + \cdots + \frac{1}{m/t_n}}$$

$$= \frac{m}{\frac{1}{n}(t_1 + t_2 + \cdots + t_n)}$$

- However, if x_i's represent the MIPS rate for n different benchmarks so that i^{th} benchmark has m_i million instructions, then harmonic mean of n ratios m_i/t_i cannot be used since the sum of the t_i/m_i does not have any physical meaning.
- Instead, as shown later, the quantity $\sum m_i / \sum t_i$ is a preferred average MIPS rate.

Weighted Harmonic Mean

- The weighted harmonic mean is defined as follows:

$$\ddot{x} = \frac{1}{\frac{w_1}{x_1} + \frac{w_2}{x_2} + \cdots + \frac{w_n}{x_n}}$$

where, w_i's are weights which add up to one:

$$w_1 + w_2 + \cdots + w_n = 1$$

- All weights equal => Harmonic, I.e., $w_i = 1/n$.
- In case of MIPS rate, if the weights are proportional to the size of the benchmark: $w_i = \frac{m_i}{m_1 + m_2 + \cdots + m_n}$

- Weighted harmonic mean would be:

$$\ddot{x} = \frac{m_1 + m_2 + \cdots + m_n}{t_1 + t_2 + \cdots + t_n}$$

Mean of A Ratio

1. ***If the sum of numerators and the sum of denominators, both have a physical meaning, the average of the ratio is the ratio of the averages***.
   For example, if $x_i=a_i/b_i$, the average ratio is given by:

$$\begin{aligned}\text{Average}\left(\frac{a_1}{b_1}, \frac{a_2}{b_2}, \cdots, \frac{a_n}{b_n}\right) &= \frac{a_1 + a_2 + \cdots + a_n}{b_1 + b_2 + \cdots + b_n} \\ &= \frac{\sum_{i=1}^{n} a_i}{\sum_{i=1}^{n} b_i} \\ &= \frac{\frac{1}{n}\sum_{i=1}^{n} a_i}{\frac{1}{n}\sum_{i=1}^{n} b_i} = \frac{\bar{a}}{\bar{b}}\end{aligned}$$

Mean of a Ratio: Example

- CPU utilization

Measurement Duration	CPU Busy
1	45%
1	45%
1	45%
1	45%
100	20%
Sum	200
Mean	$\neq 200/5$ or 40%

Example (Cont)

$$\text{Mean CPU utilization} = \frac{\text{Sum of CPU busy times}}{\text{Sum of measurement durations}}$$

$$= \frac{0.45 + 0.45 + 0.45 + 0.45 + 20}{1 + 1 + 1 + 1 + 100}$$

$$= 21\%$$

- Ratios cannot always be summarized by a geometric mean.
- A geometric mean of utilizations is useless.

Mean of a Ratio: Special Cases

a. ***If the denominator is a constant and the sum of numerator has a physical meaning, the arithmetic mean of the ratios can be used***.

That is,

$$\begin{aligned}&\text{Average}\left(\frac{a_1}{b}, \frac{a_2}{b}, \cdots, \frac{a_n}{b}\right)\\ &= \frac{1}{n}\left(\frac{a_1}{b} + \frac{a_2}{b} + \cdots + \frac{a_n}{b}\right)\\ &= \frac{\sum_{i=1}^{n} a_i}{nb}\end{aligned}$$

- Example: mean resource utilization.

Mean of Ratio (Cont)

b. If the sum of the denominators has a physical meaning and the numerators are constant then a harmonic mean of the ratio should be used to summarize them.

That is, if $a_i=a$ for all i's, then:

$$\text{Average}\left(\frac{a}{b_1}, \frac{a}{b_2}, \cdots, \frac{a}{b_n}\right) = \frac{n}{\frac{b_1}{a}+\frac{b_2}{a}+\cdots+\frac{b_n}{a}} = \frac{na}{\sum_{i=1}^{n} b_i}$$

Example: MIPS using the same benchmark

Mean of Ratios (Cont)

2. ***If the numerator and the denominator are expected to follow a multiplicative property such that $a_i = c\, b_i$, where c is approximately a constant that is being estimated, then c can be estimated by the geometric mean of a_i/b_i.***

- Example: Program Optimizer: $a_i = cb_i$
- Where, b_i and a_i are the sizes before and after the program optimization and c is the effect of the optimization which is expected to be independent of the code size.

$$\log a_i = \log c + \log b_i$$

- $\log c$ = arithmetic mean of $(\log b_i - \log a_i)$
  => c geometric mean of b_i/a_i

Program Optimizer Static Size Data

Program	Code Size		Ratio
	Before	After	
BubbleP	119	89	0.75
IntmmP	158	134	0.85
PermP	142	121	0.85
PuzzleP	8612	7579	0.88
QueenP	7133	7062	0.99
QuickP	184	112	0.61
SieveP	2908	2879	0.99
TowersP	433	307	0.71
Geometric Mean			0.79

References

- Computer Organization and Design RISC-V Edition, 1st Edition, The Hardware Software Interface by David Patterson John Hennessy - Chapter 1
- The Art of Computer System Performance Analysis , Techniques for Experimental Design, Measurement, Simulation and Modeling by Raj Jain – Chapter 12