High Performance Computer Architecture (CS60003)

Dept. of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Spring 2020

Measurement of Performance

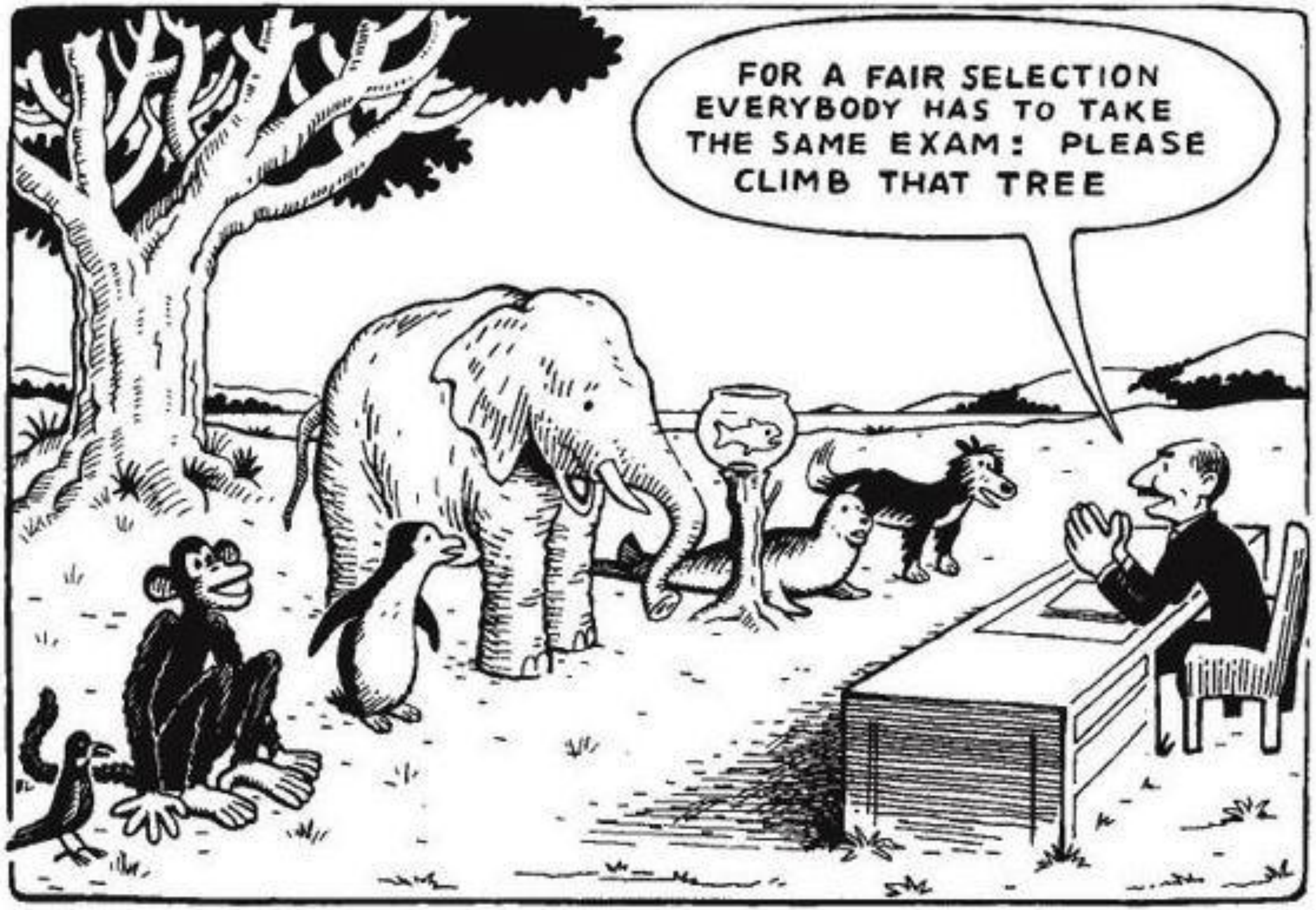
Measurement Techniques

- Modern computer systems are built from sophisticated microprocessors and extensive memory hierarchies.
- Advance in technology have resulted in exponential growth in speed (ie clock frequency) and amount of logic (number of transistors) that a chip can have.
- Computer Architects have exploited these factors to enhance performance using architectural techniques:
  - Main topic of our course.

Microprocessor Race

- Over 30 years old.
- Intel 4004 was introduced in 1971.
- The functionality of 4004 wrt. the mainframes of that period was modest.
- 30 years later, workstations powered by engines, like AMD Athlon, IBM PowerPC, Intel Pentium, Sun UltraSPARC can surpass remaining mainframes at a lower cost.
- Raw speed and number of transistors on a chip give a good feel for the progress of processors.
- However, to assess the performance of a computer system we need more precise metrics related to the execution of programs.

What to look for?



Performance Metrics

Hence, we need:

- Metrics that reflect the underlying architecture in a program independent fashion.
- A suite of programs, or benchmarks, that are representative of the intended load of a processor.

It is not easy! Lets start with some attempts.

Usual Performance Metrics

- Latency: delay from start to done for one output
- Throughput: no of outputs/second
- Can be confusing, Is Throughput=1/Latency? No!
- Consider, a car assembly.

Say, latency is 4 hours, and there are 20 steps.
The throughput is hence 5 cars/hour (if the pipeline is full!)

Does not represent the architecture.

SpeedUp

- Informally we say, “X is N times faster than Y”
- SpeedUp is $N = \frac{Speed(X)}{Speed(Y)}$
  - Using latency or throughput as a measure of speed.
  - $N = \frac{Latency(Y)}{Latency(X)}$
  - $N = \frac{Throughput(X)}{Throughput(Y)}$

Quiz 1

- A laptop takes 4 hours to compress a video.
- A new laptop can do it in 10 mnts.
- SpeedUp=4x60/10=24
- SpeedUp>1, is more intuitive and implies that there is a speedup of performance.

Quiz 2

- Consider the reverse situation.
- A laptop can compress a video in 10 mnts.
- It breaks, and so we now use the old one which compresses in 4 hours.
- SpeedUp=10/4x60=1/24=0.04
- SpeedUp<1, performance has deteriorated.

Average Execution Time

- How to compute the average SpeedUp?

	Processor++	Processor+	SpeedUp
AppA	9s	18s	2
AppB	10s	7s	0.7
AppC	5s	11s	2.2
Average	24/3=8s	36/3=12s	1.5
Geometric Mean			1.45

Average SpeedUp=1.63

Average using GM, SpeedUp=1.45

Average execution time is computed using Geometric Means.

Quiz 3

- Consider two laptops, which have speedups wrt. the following applications:
  - Music: SpeedUp of 2
  - Games: SpeedUp of 8
- Give an estimate of the overall SpeedUp?

Quiz 3

- Consider two laptops, which have speedups wrt. the following applications:
  - Music: SpeedUp of 2
  - Games: SpeedUp of 8
- Give an estimate of the overall SpeedUp?
- We shall compute the Geometric Mean, thus Overall SpeedUp= $\sqrt[2]{2\times 8} = 4.$
- Computing, the average speed as arithmetic means would not be correct, as SpeedUp is the ratio of two times.

Execution Time and Instructions Per Cycle (IPC)

- We look forward to metrics to evaluate the microarchitecture and its memory hierarchy.
- The execution time of a program whose code and data reside in the memory hierarchy is denoted by EX_{CPU}.
  - At this point we ignore the effect of I/O.
- EX_{CPU} depends on:
  - the number of instructions executed
  - the time to execute one instruction
- Time to execute one instruction depends on:
  - Cycle time (reciprocal of the clock frequency)
  - Cycles required to execute an instruction
    - CPI: Cycles Per Instruction (Program independent, clock-frequency independent)

Iron Law of Performance

- $EX_{CPU} = Number\ of\ Instructions \times CPI \times cycle\ time$
- These 3 components help to factor the contributions of important areas:
  - $Number\ of\ Instructions$:
    - affected by the algorithm and the compiler
    - affected by the instruction set architecture
  - CPI:
    - affected by instruction set architecture
    - affected by the processor design
  - $Clock\ cycle\ time$:
    - affected by processor design: reduced critical path
    - affected by circuit designer, transistor physics

First, approximation assuming that all the instructions take the same time to execute.

Role of the Computer Architect

- Can influence the:
  - Instruction Set: Choosing complex instructions will imply fewer instructions per program, but will take more cycles to execute each instruction.
  - Processor Design: Tradeoff between:
    - A processor that has very short clock cycle at the expense of spending more clock cycles per instruction, or,
    - A processor that has a larger clock cycle duration, but with reduced number of cycles per instruction.
- A Good design would try to balance these choices!

Quiz 4

- Program executes 3 billion instructions.
- Processor spends 2 cycles on each instruction.
- Processor clock is 3GHz.
- What is EX_{CPU}?

Quiz 4

- Program executes 3 billion instructions.
- Processor spends 2 cycles on each instruction.
- Processor clock is 3GHz.
- What is EX_{CPU}?
- $EX_{CPU} = 3\times 10^{9} \times 2 \times \frac{1}{3\times 10^{-9}} = 2\ sec.$

Iron Law of Performance for Unequal Instruction Times

- $EX_{CPU} = (\sum_i IC_i \times CPI_i) \times cycle\ time$
  - IC_i: No of instruction for the ith type of instruction
  - CPI_i: Cycles per Instruction for the ith type of instruction

Quiz 5

- Consider the following distribution of total 50 Billion instructions of an application:
  - 10 Billion instructions, of type Branch, CPI=4
  - 15 Billion instructions, of type Loads, CPI=2
  - 5 Billion instructions, of type Stores, CPI=3
  - Rest are integer Add, CPI=1
  - The clock is at 4 GHz.
  - What is the EX_{CPU}?

Quiz 5

- Consider the following distribution of instructions:
  - 10 Billion instructions, of type Branch, CPI=4
  - 15 Billion instructions, of type Loads, CPI=2
  - 5 Billion instructions, of type Stores, CPI=3
  - Rest are integer Add, CPI=1
  - The clock is at 4 GHz.
  - $EX_{CPU} = (10\times4 + 15\times2 + 5\times3 + 20\times1)/$ 4=105/4=26.25sec.

IPC (Instructions Per Cycle)

- IPC is a cosmetic change to CPI in that it is psychologically more appealing to strive for increases in IPC than decrease in CPI.
- Thus,

$$EX_{CPU} = (Number\ of\ Instructions \times cycle\ time)/IPC$$

- IPC is a metric which represents the throughput
- EX_{CPU} is a metric which represents the latency.

SpeedUp for Parallel Processor: Amdahl's law

- Let T_i denote the time to execute on i processors $i \in Z$;
- SpeedUp for n processors is:

$$SpeedUp_n = \frac{T_1}{T_n}$$

- Superficially for programs that exhibit lot of parallelism, one would expect $T_n = T_1/n$
- However, even for "embarrassingly parallel" programs, there is always a small amount of sequential execution and synchronization.
- Although, originally stated for parallel processors can be applied for any enhancement on a fraction of the total time.

Amdahl's Law

- $$SpeedUp = \frac{1}{(1 - Frac_{Enh}) + \frac{Frac_{Enh}}{SpeedUp_{Enh}}}$$
- $Frac_{Enh}$ is the fraction of **original execution time** that is affected by enhancement.
- **Note that the percentage is wrt. time and not say number of instructions!**

Quiz 6

- Consider a program with 50 Billion instructions.
- Clock frequency is 2GHz.
- Say, an architectural modification enhances the branch instruction so that its CPI goes from 4 to 2.

Instruction Type	% of all instructions in a program	CPI
Arithmetic	40	1
Branch	20	4
Load	30	2
Store	10	3

What is the overall SpeedUp?

Quiz 6

- Consider a program with 50 Billion instructions.
- Clock frequency is 2GHz.
- Say, an architectural modification enhances the branch instruction so that its CPI goes from 4 to 2.

Instruction Type	% of all instructions in a program	CPI
Arithmetic	40	1
Branch	20	4
Load	30	2
Store	10	3

$$\text{SpeedUp} = \frac{1}{0.8 + \frac{0.2}{2}} = \frac{1}{0.9} = 1.111$$

Quiz 6

- Consider a program with 50 Billion instructions.
- Clock frequency is 2GHz.
- Say, an architectural modification enhances the branch instruction so that its CPI goes from 4 to 2.

Instruction Type	% of all instructions in a program	CPI
Arithmetic	40	1
Branch	20	4
Load	30	2
Store	10	3

$$\text{SpeedUp}=\frac{1}{0.8+\frac{0.2}{2}}=\frac{1}{0.9}=1.111$$

Wrong!

Correct Solution

- Using Iron Law before enhancement,

$$EX_{CPU} = (0.4\times 1 + 0.2\times 4 + 0.3\times 2 + 0.1\times 3)\times 50\times \frac{1}{2}$$
$$= (0.4 + 0.8 + 0.6 + 0.3)25 = 2.1\times 25$$

- After enhancement,

$$EX_{CPU} = (0.4\times 1 + \color{red}{0.2\times 2} + 0.3\times 2 + 0.1\times 3)\times 50\times \frac{1}{2}$$
$$= (0.4 + 0.4 + 0.6 + 0.3)25 = 1.7\times 25$$

- Hence, SpeedUp=2.1/1.7=1.24

Implication of Amdahl's Law

- $SpeedUp = \frac{1}{(1-Frac_{Enh})+\frac{Frac_{Enh}}{SpeedUp_{Enh}}}$
- Enhancement 1: SpeedUp of 20 on component which takes 10% of time
  - $SpeedUp = \frac{1}{0.9+\frac{0.1}{20}} = 1.105$
- Enhancement 2: SpeedUp of 2 on component which takes 80% of time
  - $SpeedUp = \frac{1}{0.2+\frac{0.8}{2}} = 1.67$
- Even with an infinite SpeedUp on 10% of time, $SpeedUp = \frac{1}{0.9+0} = 1.111$

Implication of Amdahl's Law

- $SpeedUp = \frac{1}{(1-Frac_{Enh})+\frac{Frac_{Enh}}{SpeedUp_{Enh}}}$
- Enhancement 1: SpeedUp of 20 on component which takes 10% of time
  - $SpeedUp = \frac{1}{0.9+\frac{0.1}{20}} = 1.105$
- Enhancement 2: SpeedUp of 2 on component which takes 80% of time
  - $SpeedUp = \frac{1}{0.2+\frac{0.8}{2}} = 1.67$
- Even with an infinite SpeedUp on 10% of time, $SpeedUp = \frac{1}{0.9+0} = 1.111$

Make Common Case Fast!

Quiz 7

- Consider the following statistics:

Inst Type	% of Time	CPI
Arithmetic	40	1
Branch	20	4
Load	30	2
Store	10	3

Which is the best. Improvement among:

1. Branch: CPI $4 \rightarrow 3$
2. Increase Clock Frequency: $2 \rightarrow 2.3GHz$
3. Store: CPI $3 \rightarrow 2$?

SpeedUp1=$\frac{1}{0.8+\frac{0.2}{4/3}} = 1.05$

SpeedUp2=$\frac{1}{0+\frac{1}{1.15}} = 1.15$

SpeedUp3=$\frac{1}{0.9+\frac{0.1}{3/2}} = 1.034$

LHADMA's Law

- Cautions that in pursuit of optimizing the common case, the uncommon case shouldn't be slowed down too much.
- Example: Consider the following two options
  - Improvement of 2x on components which take 90% time
  - But slow down the rest by 10x

$$SpeedUp = \frac{1}{\frac{0.1}{0.1}+\frac{0.9}{2}} = \frac{1}{1.45} = 0.7$$

- Even with an infinite speedup on the 90% component: $SpeedUp = \frac{1}{\frac{0.1}{0.1}+\frac{0.9}{\infty}} = 1$

Law of Diminishing Returns

- Diminishing returns with respect to Amdahl's Law refers to the fact that continuing to optimize a specific part of the execution eventually provides less and less gains in speedup
  - This is so as the optimized portion grows smaller and smaller, accounting for less and less of execution time.
- Intuitively, it can be considered that the “low hanging fruit” of optimizations will be gone after a while, making it harder to optimize.
- The practical implication of this is that after optimizing a portion of the execution, one must re-evaluate which portion is now dominant.

Example

- Consider a program execution which has two parts in the execution time, p and q.
- Initially, p and q are both 50% of the time.
- Due to an optimization, say there is a SpeedUp on q by 50%.
  - Thus, $SpeedUp = \frac{1}{0.5+\frac{0.5}{2}} = 1.33$
- Say, the architect optimizes the 2nd part again with a further SpeedUp of 2.
- Note, now p takes 0.67 of the time, and q (due to the first optimization) takes 0.33 of the time.
  - Thus, $SpeedUp = \frac{1}{0.67+\frac{0.33}{2}} = 1.2$
- Thus, the speedup diminishes; eventually we will get almost no speedup, as we are speeding up less and less of the execution time!

Thank You!