Computer Performance

Caiwen Ding
Department of Computer Science and Engineering
University of Connecticut

CSE3666: Introduction to Computer Architecture

Outline

- Metrics to compare computer performance
- CPU time (a.k.a. CPU Execution time or Execution time)
- CPI
- Speedup
- Amdahl's law

Reading: Sections 1.6 and 1.10

Math background | UConn CSE-3666

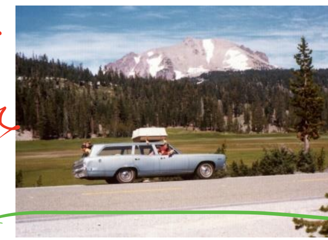
Metrics to compare computer performance

perspective of System = Throughput
individual = Latency

- Response Time / Latency / Delay
  - How long it takes to complete a task

- Throughput
  - Total work done per unit time
    - e.g., tasks/transactions/… per hour

$\frac{100 \text{ Tasks}}{24 \text{ hours}}$



A B

Throughput vs. Latency (delay)

- Rates
  - MFLOPS (million floating-point operations per second)
  - MIPS (million instructions per second)
    - Not the same MIPS in MIPS ISA

Measuring Time

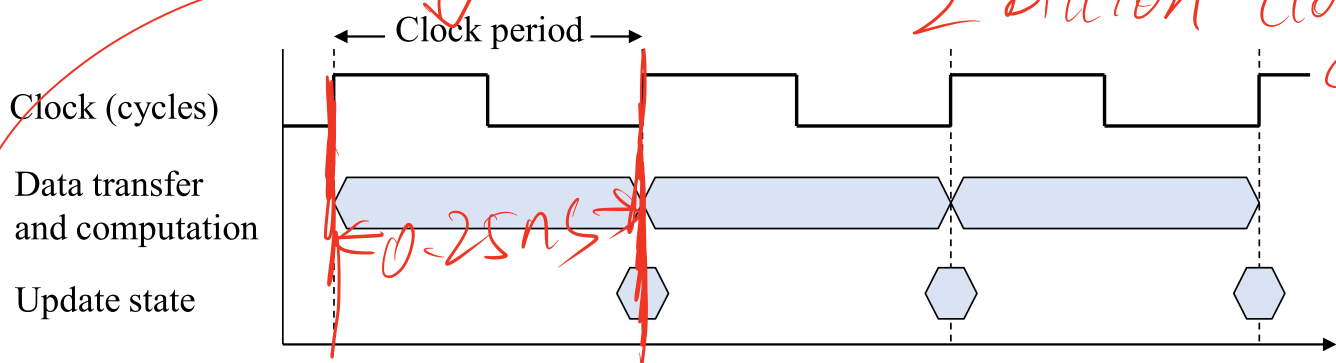
- Elapsed time
  - Total response time, including all aspects
    - Processing, I/O, OS overhead, idle time
  - Determines system performance
- CPU Execution time = In this class
  - Time the processor needs to process a given job
    - Excluding I/O time and other jobs' shares
  - Comprises user CPU time and system CPU time
  - Also called CPU time, or execution time
    - We do not consider I/O, OS, etc. in this course

CPU Clocking

clock tree

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

2 billion clock cycles



- Clock period (cycle time): 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

(ns)

CPU time: 0.25ns × 2billion (s)

CPU Time

$= 0.25 \times 10^{9} \times 2 \times 10^{9} = 0.5$ (ms) (us)

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

2 billion

0.25ns

Number of clock cycles

or

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

?

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

Performance

We can define performance as

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

It is CPU execution time

As CPU execution time decreases, performance increases

Comparing performance

"X is *n* time faster than Y"

"… than Y"
Y's time is the top number

$$n = \frac{\text{Performance}_x}{\text{Performance}_y} = \frac{\dfrac{1}{\text{ExecutionTime}_x}}{\dfrac{1}{\text{ExecutionTime}_y}} = \frac{\text{ExecutionTime}_y}{\text{ExecutionTime}_x}$$

X : 5s

Y : 10s

We will use the execution time, to avoid confusion.

n can be less than 1. What does that mean?

Example

Time taken to run a program is 10s on A and 15s on B.

Intuitively, A is faster than B.

$$\frac{Perf_A}{Perf_B} = n = \frac{\text{ExecutionTime}_B}{\text{ExecutionTime}_A} = \frac{15}{10} = 1.5$$

So A is 1.5 times faster than B.

B's time is at the top.

Question - Performance

Processor A's clock rate is 2 GHz → CCT = 0.5ns

Processor B's clock rate is 3 GHz → CCT = 0.333ns

An application needs 30% more clock cycles on Processor B.

How much faster is the application on Processor B? than Proc A

$$n = \frac{CPU_A}{CPU_B} = \frac{CC_A / CR_A}{CC_B / CR_B} = \frac{CC_A / 2}{1.3\,CC_A / 3}$$

Solutions

CC: Clock cycles

B is faster…

$$\text{ExecutionTime}_\text{A} = \text{CC}_\text{A} \times 0.5\text{ ns} = 0.5 \times \text{CC}_\text{A}\text{ ns}$$

$$\text{ExecutionTime}_\text{B} = (1.30 \times \text{CC}_\text{A}) \times 0.333\text{ ns} = 0.433 \times \text{CC}_\text{A}\text{ ns}$$

$$\frac{\text{ExecutionTime}_\text{A}}{\text{ExecutionTime}_\text{B}} = \frac{0.5 \times \text{CC}_\text{A}\text{ ns}}{0.433 \times \text{CC}_\text{A}\text{ ns}} = 1.15$$

…by this much

Processor B is 1.15 times faster than process A.

Question

$T_{CPU} = CC \times CCT$

Processor A: 2GHz clock, 10s CPU time for an application

or CC / CR

Designing Processor B, using the same ISA

Processor B can run at a faster clock rate, but needs 1.2 × clock cycles.

If we want to achieve 6s CPU time on process B, how fast must processor B's clock be?

Solutions

Compute the number of clock cycles (CC_A) for A first.

Compute the number of clock cycles for B: $CC_B = 1.2 \times CC_A$

Compute the targeted clock rate.

$$CC_A = \frac{\text{CPU Time}_A}{\text{Clock Cycle Time}_A} \;\text{CCT} = \text{CPU Time}_A \times \overset{\text{CR}}{\text{Clock Rate}_A}$$

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

$$CC_B = 1.2 \times CC_A = 1.2 \times 20 \times 10^9 = 24 \times 10^9$$

$$\text{Clock Cycle Time}_B = \frac{\text{CPU Time}_B}{CC_B} = \frac{6\text{s}}{24 \times 10^9} = 0.25 \text{ ns}$$

$$\text{Clock Rate}_B = \frac{1}{\text{Clock Cycle Time}_B} = \frac{1}{0.25 \text{ ns}} = 4\text{GHz}$$

Calculating the number of cycles

CPU = Inst Count (IC), Cycle per Inst (CPI)

How many cycles do these instructions take?

IC × CPI

```
# Method 1 RISC-V code
PC →    addi    s0, x0, 0
PC →    addi    s1, x0, 0
        addi    s2, x0, 100
loop:   bge     s0, s2, exit
        add     s1, s1, s0
        addi    s0, s0, 1
        beq     x0, x0, loop
exit:
```

How many instructions are executed?

CPI (clock Cycles Per Instruction)

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}}$$

Seperate for { Arith, Memory ...

or

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

- Instructions take different number of cycles to execute
  - Example: One cycle for add, 10 cycles for mul, 20 cycles for lw
- Very often we need to compute average CPI with mixed types of instructions

Computing average CPI

- If we know the clock cycles and the instruction count:

$$\text{CPI} = \frac{\text{Clock Cycles}}{\text{Instruction Count}}$$

- Compute weighted average

$$\text{CPI} = \sum_{i=1}^{n} \left(\text{CPI}_i \times \frac{\text{Instruction Count}_i}{\text{Instruction Count}} \right)$$

types of Inst

Clock cycles that class i instructions take

frequency (or weight) of class i instructions

Example

types of Inst, e.g. add mult, lw

- A processor has three classes of instructions: A, B, and C
- A program can be compiled differently and has two instruction (execution) sequences.
- Find average CPI for each sequence.

Instruction counts (ICs) are in billions.

Class	A	B	C
CPI of each class	1	2	3
IC in Sequence 1	2 +	1 +	2 = 5
IC in Sequence 2	4	1	1

What is the CPI for Sequence 1?

$$CC_n = CC_A + CC_B + CC_C$$

$$= 1 \times 2 + 2 \times 1 + 3 \times 2 = 10$$

Example: Sequence 1

$$\text{Avg CPI} = \frac{CC_n}{IC_{Total}} = \frac{10}{5} = 2$$

Class	A	B	C	
CPI of each class	1	2	3	= 9
IC in Sequence 1 Seq	2 /5	1 /5	2 /5	
IC in Sequence 2 Seq	4 /6 +	1 /6 +	1 /6	= 6

Sequence 1

IC = 2 + 1 + 2 = 5 billion

Clock Cycles = 2×1 + 1×2 + 2×3 = 10 billion

Average CPI = 10 / 5 = 2

What is the CPI for Sequence 2?

IC = 4+1+1 = 6

CC = 4×1+1×2+1×3 = 9

$$\text{Avg CPI} = \frac{CC}{IC} = 1.5$$

Answer for Sequence 2

Sequence 2

IC = 4 + 1 + 1 = 6

Clock Cycles = 4×1 + 1×2 + 1×3 = 9

Average CPI = 9 / 6 = 1.5

Questions

- Find average CPI for sequences 1 and 2.

Class	A	B	C	
CPI of each class	1	2	3	
Sequence 1	50%	10%	40%	= 100%
Sequence 2	40%	35%	25%	= 100%

Solutions

Sequence 1

Avg. CPI $= 0.5\times1 + 0.1\times2 + 0.4\times3 = 1.9$

Sequence 2

Avg. CPI $= 0.4\times1 + 0.35\times2 + 0.25\times3 = 1.85$

Earlier Example: Sequence 1

Class	A	B	C
CPI of each class	1	2	3
IC in Sequence 1	2 40%	1 20%	2 40%
IC in Sequence 2	4	1	1

Sequence 1

IC = 2 + 1 + 2 = 5 billion

Clock Cycles = 2×1 + 1×2 + 2×3 = 10 billion

Average CPI = 10 / 5 = 2

Another method:

Find the frequency of each class of instructions and compute weighted average

Class A: 2/5 = 0.4   Class B: 1/5 = 0.2   Class C: 2/5 = 0.4

Average CPI = 1 * 0.4 + 2 * 0.2 + 3 * 0.4 = 2

Compute clock cycles

- Compute average CPI (of all instructions) first

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{CPI}_{\text{average}}$$

- Add up clock cycles of each class of instructions

$$\text{Clock Cycles} = \sum_{i=1}^{n} \underbrace{(\text{CPI}_i \times \text{Instruction Count}_i)}_{\text{clock cycles for class i}}$$

Combining two equations

We have two equations:

clock

$$\text{CPU Time} = \text{Clock Cycles} \times \text{Clock Cycle Time} \qquad \text{Eq. 1}$$

RISC-V

$$\text{Clock Cycles} = \text{Instruction Count} \times \text{CPI} \qquad \text{Eq. 2}$$

Substitute Eq. 2 into Eq. 1 :

$$\text{CPU Time} = \underbrace{\text{Instruction Count} \times \text{CPI}}_{\text{clock cycles}} \times \cancel{\text{Clock Cycle Time}}$$

$$\frac{1}{\text{Clock Rate}}$$

Classic equation

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

- To reduce CPU Time, reduce any of the three factors

Hardware/software	Affects what?
Algorithm	Instruction count, CPI
Programming language	Instruction count, CPI
Compiler	Instruction count, CPI
Instruction set architecture	Instruction count, CPI, clock cycle time
Microarchitecture	CPI, clock cycle time
Circuit design	CPI, clock cycle time

Example

Run the same program on two processors that have the same ISA.

Computer A: Cycle Time = 250ps, CPI = 2.0

Computer B: Cycle Time = 500ps, CPI = 1.2

Which computer is faster, and by how much?

Solutions

Run the same program on two processors that have the same ISA.

Computer A: Cycle Time = 250ps, CPI = 2.0

Computer B: Cycle Time = 500ps, CPI = 1.2

Which computer is faster, and by how much?

$$\begin{aligned}\text{CPU Time}_A &= \text{IC} \times \text{CPI}_A \times \text{Cycle Time}_A \\ &= \text{IC} \times 2.0 \times 250\text{ps} = \text{IC} \times 500 \text{ ps}\end{aligned}$$

A is faster…

$$\begin{aligned}\text{CPU Time}_B &= \text{IC} \times \text{CPI}_B \times \text{Cycle Time}_B \\ &= \text{IC} \times 1.2 \times 500\text{ps} = \text{IC} \times 600 \text{ ps}\end{aligned}$$

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

…by this much

A is 1.2 times faster than B. B's time is at the top.

Speedup

Compare the performance of design options.

The new design is _________ times faster than the original.

" .... than the original."
The original time is at the top

$$\text{Speedup} = \frac{\text{CPU Time}_{\text{before\_enhancement}}}{\text{CPU Time}_{\text{after\_enhancement}}}$$

Example:

Using a new method, 40% of an application can execute 10 times faster.
What is the overall speedup of the new method on the application?

Solution

$$\text{Speedup} = \frac{1}{0.6 + \frac{0.4}{10}} = \frac{1}{0.64} = 1.56$$

Amdahl's law

- The performance improvement to be gained from using some faster mode of execution is limited by the fraction of the time the faster mode can be used.

  Make common case fast!

The best speedup you can achieve by optimizing the 40% code is

$$\text{BestSpeedup} = \frac{1}{0.6} \approx 1.6$$

Pitfall: Amdahl's Law

- Improving an aspect of a computer and expecting a proportional improvement in overall performance

$$\mathrm{T}_{\mathrm{improved}} = \frac{\mathrm{T}_{\mathrm{affected}}}{\text{improvement factor}} + \mathrm{T}_{\mathrm{unaffected}}$$

Example:

Multiplication accounts for 80% of the execution time.

How much improvement in multiplication performance is needed to get 5× overall speedup?

$$\frac{1}{\frac{0.8}{x} + 0.2} = 5$$

Performance Summary

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

- Performance depends on all three factors
- Optimization can be done at many levels
  - Algorithm, programming language, compiler, instruction set architecture, microarchitecture, and hardware implementation

- Amdahl’s Law

Question

Processors A and B have the same ISA.

Processor A runs at 2 GHZ and Processor B runs at 2.5 GHZ.

For an application, the CPI is 1.5 on Processor A, and 1.7 on Processor B.

If you switch from Processor A to Processor B, what is the speedup?

Truncate to the first digit after the decimal point, e.g., 4.15 to 4.1

Answer

Instruction count (IC) is the same.

$$\frac{\text{CPU Time}_\text{A}}{\text{CPU Time}_\text{B}} = \frac{\text{IC} \times 1.5 \times \dfrac{1}{2 \times 10^9}}{\text{IC} \times 1.7 \times \dfrac{1}{2.5 \times 10^9}} = \frac{1.5 \times 2.5}{1.7 \times 2} = 1.10$$

B (the new system) is 1.10 times faster than A.

A's time is at the top.

To Improve Performance

- Algorithm
  - Determines number of operations executed
- Programming language, compiler, architecture
  - Determine number of machine instructions executed per operation
- Processor and memory system
  - Determine how fast instructions are executed
- I/O system (including OS)
  - Determines how fast I/O operations are executed

Defining Performance

- Which airplane has the best performance?