

Computer Performance



Z. Jerry Shi

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 \(zhijieshi.github.io\)](https://zhijieshi.github.io)

(It is a link!)

Metrics to compare computer performance

- **Response Time** (one Task)
 - How long it takes to complete a task (RISC-V INSTR Flow)
- **Throughput** (Arith, logic, mem) (avg Total Tasks)
 - Total work done per unit time
 - e.g., tasks/transactions/... per hour
- **Rates**
 - MFLOPS (million floating-point operations per second)
 - MIPS (million instructions per second)
 - Not the same MIPS in MIPS ISA

Measuring Time

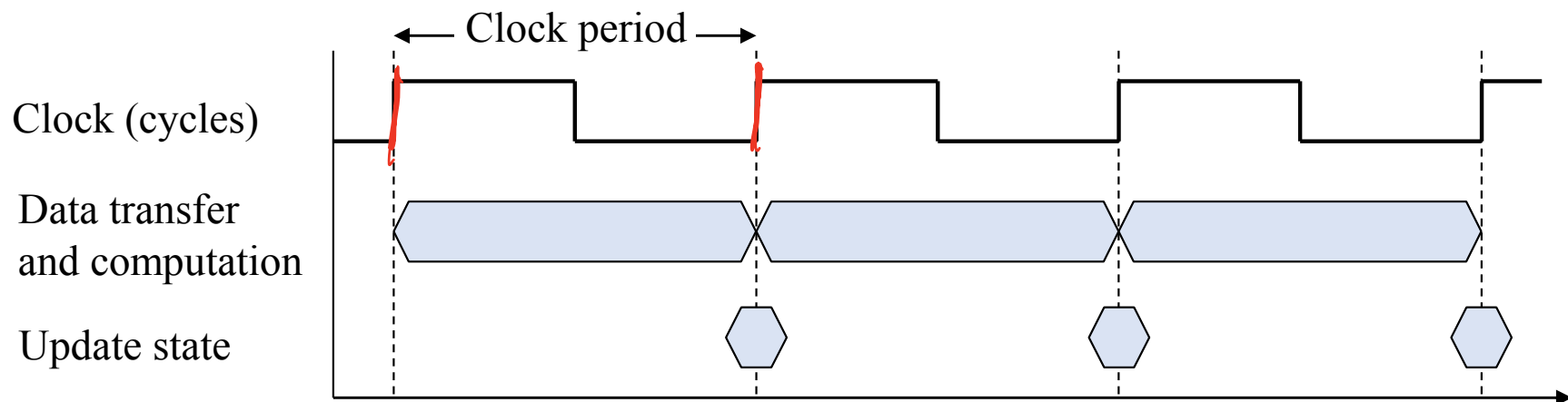
- Elapsed time
 - Total response time, including all aspects
 - Processing, I/O, OS overhead, idle time
 - Determines system performance
- → CPU Execution time
 - Time the processor needs to process a given job
 - Excluding I/O time and other jobs' shares
 - Comprises user CPU time and system CPU time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

CPU Clocking

Processor is sequential. It needs a clock to run.

Assumption in this course: **the clock rate of a processor is constant**

We can calculate the CPU time of a program by counting the number of clock cycles it needs.

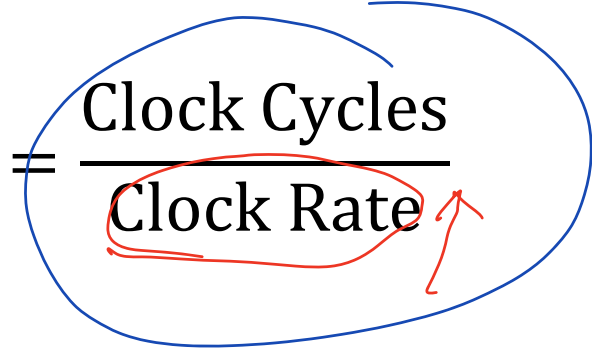


CPU Time


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

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
 - Reducing the clock cycle time
 - Hardware designer must often trade off clock rate against clock cycle

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

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

We will use the execution time, to avoid confusion.

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

Slower

Example

Time taken to run a program is 10s on Processor A and 15s on B.
Intuitively, A is faster than B.

"... than B"
B's time is the top

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

So A is 1.5 times faster **than B**.

Question - Performance

Processor A's clock rate is 2 GHz

Processor B's clock rate is 3 GHz

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

How much faster is the application on Processor B?

$$n = \frac{E_{\text{ex}}(B)}{E_{\text{ex}}(A)} = \frac{1.3 \cancel{C_A} \cdot \cancel{t_B} / \cancel{C_B} 3}{\cancel{C_A} / \cancel{C_B} 2}$$

Solutions

CC: Clock cycles

$$\text{ExecutionTime}_A = CC_A \times 0.5 \text{ ns} = 0.5 \times CC_A \text{ ns}$$

B is faster than A

$$\text{ExecutionTime}_B = (1.30 \times CC_A) \times 0.333 \text{ ns} = 0.433 \times CC_A \text{ ns}$$

$$\frac{\text{ExecutionTime}_A}{\text{ExecutionTime}_B} = \frac{0.5 \times CC_A \text{ ns}}{0.433 \times CC_A \text{ ns}} = 1.15$$

Processor B is 1.15 times faster than process A.

...by this much

Calculating the number of cycles

Very often, we know the
number of executed
instructions.

*Inst Count
(IC)*

How many cycles do they
take?

Method 1 RISC-V code

```
addi    s0, x0, 0
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?

Inst

CPI (clock Cycles Per Instruction)

$$\text{CPI} = \frac{\text{Clock Cycles}^{(200)}}{\text{Instruction Count}^{(10,000)} \text{ (IC)}}$$

or

$$= \frac{2}{100}$$

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

Inst 0
Inst 1

⋮

Inst 9999

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

Computing average CPI

- Method 1: if we know clock cycles and instruction count:

$$\text{CPI} = \frac{\text{Clock Cycles (Total)}}{\text{Instruction Count (Total)}} \quad \text{By definition}$$

- * Method 2: Compute weighted average (No Total info!!)

$$\text{CPI} = \sum_{i=1}^n \left(\text{CPI}_i \times \frac{\text{Instruction Count}_i}{\text{Instruction Count}} \right) = \frac{\text{CPI}_1 \times \text{IC}_1 + \text{CPI}_2 \times \text{IC}_2}{\text{IC}_{\text{Total}}}$$

Clock cycles that
class i instructions take

frequency (or weight) of
class i instructions

Example

- 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 <i>2</i>	B <i>1</i>	C <i>3</i>
CPI of each class	<i>1</i>	<i>2</i>	<i>3</i>
IC in Sequence 1	<i>2</i>	<i>1</i>	<i>2</i>
IC in Sequence 2	4	1	1

(5)
(6)

What is the CPI for Sequence 1?

$$\frac{10}{5} = \textcircled{2}$$

$$C C_{(Total)} = 2 \times 1 + 1 \times 2 + 2 \times 3 = 10$$

Example: Sequence 1

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

Sequence 1

$$\text{IC} = 2 + 1 + 2 = 5 \text{ billion}$$

$$\text{Clock Cycles} = 2 \times 1 + 1 \times 2 + 2 \times 3 = 10 \text{ billion}$$

$$\text{Average CPI} = 10 / 5 = 2$$

Example: Another way - Sequence 1

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

Sequence 1

Find the frequency of each class of instructions

Total number of instructions in Sequence 1 is $2 + 1 + 2 = 5$

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

And then compute weighted average

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

What is the CPI for Sequence 2?

Answer for Sequence 2

Sequence 2

$$IC = 4 + 1 + 1 = 6$$

$$\text{Clock Cycles} = 4 \times 1 + 1 \times 2 + 1 \times 3 = 9$$

$$\text{Average CPI} = 9 / 6 = 1.5$$

Questions

- Find average CPI for each program.

Class	A	B	C
CPI of each class	1	2	3
Program 1	<u>50%</u>	<u>10%</u>	<u>40%</u>
Program 2	40%	35%	25%
Program 3	30%	30%	40%

= /
= /
= /

Solutions

Program 1

$$\text{Avg. CPI} = 0.5 \times 1 + 0.1 \times 2 + 0.4 \times 3 = 1.9$$

Program 2

$$\text{Avg. CPI} = 0.4 \times 1 + 0.35 \times 2 + 0.25 \times 3 = 1.85$$

Compute clock cycles

- Method 1: Compute average CPI (of all instructions) first, and then compute clock cycles

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

- Method 2: add up clock cycles of each class of instructions

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

These two methods are the same.

The frequency of each class is hidden in CPI-Average

Combining two equations

We have two equations:

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

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

Substitute Eq. 2 into Eq. 1 :

$$\text{CPU Time} = \underbrace{\text{Instruction Count} \times \text{CPI}}_{\text{clock cycles}} \times \text{Clock Cycle Time} \quad \text{Eq. 3}$$

Handwritten annotations: "IL" above "Instruction Count", "x CPI x CCT" above the multiplication, and "clock cycles" below the bracket.

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?

$$\frac{CPU_A}{CPU_B} = \frac{? \cancel{IC_A} \times \overset{2}{CPI_A} \times \overset{250}{CCT_A}}{? \cancel{IC_B} \times \underset{1.2}{CPI_B} \times \underset{500}{CCT_B}} = \cancel{1.2} \times \frac{1}{1.2}$$

Same program
Same ISA \rightarrow Same IC

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?

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

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

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

A is faster
than B

...by this much

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

Speedup

no A & B

old A \rightarrow new A

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

$$\text{Speedup} = \frac{\text{CPU Time}_{\text{before_enhancement}}}{\text{CPU Time}_{\text{after_enhancement}}}$$

original
new

Example:

$$= \frac{40\% + 60\% \text{ (Time)}}{40\% + 60\% \text{ (Time)}} = \frac{1}{\frac{0.4}{10} + 0.6}$$

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

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

Handwritten red annotations: A red circle around 0.6, a red circle around the fraction $\frac{0.4}{10}$, and a red circle around 1.56. A red arrow points from the denominator of the fraction to the handwritten text below.

1M↑
1B↑

$$0.6 + 0$$

Handwritten red annotations: A red circle around the expression, a red arrow pointing from the denominator of the fraction in the equation above to this expression, and a red arrow pointing from the denominator of the fraction in the equation above to the handwritten text below.

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

$$T_{\text{improved}} = \frac{T_{\text{affected}}}{\text{improvement factor}} + T_{\text{unaffected}}$$

Example:

Multiplication accounts for 80% of the execution time.

How much improvement in multiplication performance is needed to get $5\times$ overall speedup?

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

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

$x \approx NA$

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
 - Which section should we optimize? Can we achieve what we want?



Understand and explain the equations.
Do NOT simply memorize them.

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

Question

$$CPU_{Time} = CC / \text{Clock Rate}$$

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

Designing Processor B, using the same ISA

Processor B can run at a faster clock rate, but needs $1.2 \times$ 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.

$$\begin{aligned} CC_A &= \frac{\text{CPU Time}_A}{\text{Clock Cycle Time}_A} = \text{CPU Time}_A \times \text{Clock Rate}_A \\ &= 10\text{s} \times 2\text{GHz} = 20 \times 10^9 \end{aligned}$$

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

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

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

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.

$$CPU_T = IC \times CPI \times CC_T$$

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.

$$\frac{\text{CPU Time}_A}{\text{CPU Time}_B} = \frac{\text{IC} \times 1.5 \times \frac{1}{2 \times 10^9}}{\text{IC} \times 1.7 \times \frac{1}{2.5 \times 10^9}} = \frac{1.5 \times 2.5}{1.7 \times 2} = 1.10$$

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

A's time is at the top.

Defining Performance

- Which airplane has the best performance?

