Chapter 1

The Role of Performance Measurement

Performance

What does it means?

- Purchasing perspective
  - given a collection of machines, which has the
    - best performance ?
    - least cost ?
    - best performance / cost ?
- Design perspective
  - faced with design options, which has the
    - best performance improvement ?
    - least cost ?
    - best performance / cost ?
- Both require
  - basis for comparison
  - metric for evaluation
- Our goal is to understand cost & performance implications of architectural choices

Two Notions of performance

Airplane	Passenger Capacity	Cruising range	Cruising speed	Passenger Throughput
Boeing 777	375	4630	610	228750
Boeing 747	470	4150	610	286700
BAC/Sud Concorde	132	4000	1350	178200
Douglas DC-8-50	146	8720	544	79424

- Which has higher Performance?
- Response Time
  - Time to do a task
    - execution time, response time, latency
- Throughput
  - Task per time
    - throughput, bandwidth
- Response Time and Throughput are often in opposition

The winner?

Airplane	Passenger Capacity	Cruising range	Cruising speed	Passenger Throughput
Boeing 777	375	4630	610	228750
Boeing 747	470	4150	610	286700
BAC/Sud Concorde	132	4000	1350	178200
Douglas DC-8-50	146	8720	544	79424

- If we define performance by speed, we have two possibilities:
  - Highest cruising speed -> Concorde wins
  - Taking a single passenger with the least time -> 747 wins
- Performance is defined by many parameters
- The same with computers
  - Reduce response time
  - Increase thoughput

Example

- Do the following changes to a computer system increase throughput, decrease response time, or both?
  - Replacing with faster processor
  - Adding an additional processor

- Case 1: reducing reponse time will increase throughput
  - -> Both
- Case 2: adding throughput reducing waiting time (response time)
  - -> Both

Definition

- Performance is in units of things-per-second
  - bigger is better
- If we are primarily concerned with response time

$$\text{Performance}_X = \frac{1}{\text{Execution time}_X}$$

- How to read:
  - Performace of Machine X
  - Execution time of Machine X

Performance Comparison

- Greater Than or Less Than

$$\text{Performance}_X < \text{Performance}_Y$$

$$\frac{1}{\text{Execution time}_X} < \frac{1}{\text{Execution time}_Y}$$

$$\text{Execution time}_X > \text{Execution time}_Y$$

Example

Airplane	Passenger Capacity	Cruising range	Cruising speed	Passenger Throughput
Boeing 777	375	4630	610	228750
Boeing 747	470	4150	610	286700
BAC/Sud Concorde	132	4000	1350	178200
Douglas DC-8-50	146	8720	544	79424

- Time of Concorde vs. Boeing 747?
  - Concord is 1350 mph / 610 mph = 2.2 times faster
- Throughput of Concorde vs. Boeing 747 ?
  - Concord is 178,200 pmph / 286,700 pmph = 0.62 “times faster”
  - Boeing is 286,700 pmph / 178,200 pmph = 1.6 “times faster”
- Boeing is 1.6 times (“60%”)faster in terms of throughput
- Concord is 2.2 times (“120%”) faster in terms of flying time
- We will focus primarily on execution time for a single job

Performance Relation

- Machine X is n times faster than Machine Y

$$\frac{\text{Performance}_X}{\text{Performance}_Y} = n$$

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

Example

- Machine P runs a program in 20 seconds and Machine Q runs the same program in 15 seconds
  - How much faster is machine Q than machine P?

- We know Q is n times faster than P

$$\frac{\text{Performance}_Q}{\text{Performance}_P} = n$$

$$\frac{\text{Execution time}_P}{\text{Execution time}_Q} = n$$

- Thus the performance ratio is 20/15 = 1.33..
  - And Q is 1.33.. Times faster than P

Measuring Performance

- **Time is the measure of computer performance**

- The computer that perform the same amount of work in the least time is the fastest
- Program *execution time* is seconds per program
- The most straightforward is
  - Wall clock time
  - Response time
  - Elapsed time

What is execution time or elapsed time?

- **Problem: Computer are often time shared**

- Distinguish between elapsed time and CPU time.
  - CPU time is the time the processor is working on our program (does not include time spent on I/O or other program)
  - CPU time can be divided into
    - User CPU time
    - System CPU time
  - Difficult to measure
- Performance
  - CPU performance
  - System performance

Example

- Unix time for a task or program
  - 90.7u 12.9s 2:39 65%
  - User CPU time is 90.7 seconds
  - System CPU time is 12.9 seconds
  - Elapsed time is 2 minutes 39 seconds (159 seconds)
  - The percentage of the elapsed time that is the CPU time is 65%

$$\frac{90.7 + 12.9}{159} = 0.65$$

- 35 % is spent on I/O and other programs

Clock cycle

Almost all computer runs at a constant rate clock

- Other name for clock cyles : ticks, clock ticks, clock periods, clock, cycles.
- Clock period is the inverse of clock cycle
  - Ex: 2 ns clock period is 500 MHz clock cycle

Relating the metric

$$\text{CPU execution time} = \text{CPU clock cycle} \times \text{clock cycle time}$$

$$\text{CPU execution time} = \frac{\text{CPU clock cycle}}{\text{clock rate}}$$

or

$$\frac{\text{seconds}}{\text{program}} = \frac{\text{cycles}}{\text{program}} \times \frac{\text{seconds}}{\text{cycles}}$$

- Hardware designer can improve performance by reducing
  - the length of the clock cycle or
  - the number of clock cycle per program

Example 1

Improving performance

- Machine A which has 500 MHz clock runs a program in 5 seconds
  - What is the CPU cyle of machine A?
  - We improve machine A with a new machine B which has 750 MHz clock. Assuming the same clock cyle, how long does the same program runs on B?
  - We improve machine A with a new machine C whic has 1000 MHz clock but the number of cycle is 1.3 times the number of cyle of machine A. How long does the same program runs on C?

Answer

$$CPU\,time_A = \frac{CPU\,clock\,cycle_A}{Clock\,rate_A}$$

$$5 = \frac{CPU\,clock\,cycle_A}{500 \times 10^6}$$

$$CPU\,clock\,cycle_A = 5 \times 500 \times 10^6 = 2500 \times 10^6\,cycle$$

Answer

- CPU time for machine B:

$$CPU\,time_B = \frac{CPU\,clock\,cycle_A}{Clock\,rate_B}$$

$$CPU\,time_B = \frac{2500 \times 10^6}{750 \times 10^6} = 3.3333\ seconds$$

- CPU time for machine C:

$$CPU\,time_C = \frac{1.3 \times CPU\,clock\,cycle_A}{Clock\,rate_C}$$

$$CPU\,time_C = \frac{1.3 \times 2500 \times 10^6}{1000 \times 10^6} = 3.25\ seconds$$

Example 2

- Machine A which has 500 MHz clock runs a program in 10 seconds.
  - We want to build a machine that will run the same program in 6 seconds. What is the clock rate of a new machine D if the clock cycle is increased by 1.2 times?

Example 2

- Machine A which has 500 MHz clock runs a program in 10 seconds.
  - We want to build a machine that will run the same program in 6 seconds. What is the clock rate of a new machine D if the clock cycle is increased by 1.2 times?

$$CPU\,time_A = \frac{CPU\,clock\,cycle_A}{Clock\,rate_A}$$

$$10 = \frac{CPU\,clock\,cycle_A}{500 \times 10^6}$$

$$CPU\,clock\,cycle_A = 5000 \times 10^6\,cycles$$

$$CPU\,time_D = \frac{1.2 \times CPU\,clock\,cycle_A}{Clock\,rate_D}$$

$$6 = \frac{1.2 \times 5000 \times 10^6}{Clock\,rate_D}$$

$$Clock\,rate_D = 1000 \times 10^6 = 1\,GHz$$

Hardware Software Interface

- **Execution must depends on the number of instruction per program**
- Compiler generated the instructions to be execute and the machine had to execute the instructions to run the program

$$CPU\ clock\ cycle = Instructions\ for\ a\ program \times Average\ clock\ cycle\ per\ instruction$$

- The average number of cycles per instruction is abbreviated as CPI - clock cycles per instruction

Example

- **Suppose we have two machine with the same ISA**
  - Machine A: clock cycle 1.5 ns and CPI 2
  - Machine B: clock cycle 2ns and CPI 1.75
  - Which one is faster and by how much?

Answer

- CPU cycles

$$CPU\,clock\,cycle_A = I \times 2$$
$$CPU\,clock\,cycle_B = I \times 1.75$$

- CPU time

$$CPU\,time_A = CPU\,clock\,cyle_A \times Clock\,cycle\,time_A$$
$$CPU\,time_A = 2 \times I \times 1.5ns = 3.0 \times I\,ns$$
$$CPU\,time_B = 1.75 \times I \times 2ns = 3.5 \times I\,ns$$

Answer

- Comparison

$$\frac{CPU\,performance_A}{CPU\,performance_B} = \frac{Execution\,time_B}{Execution\,time_A}$$

$$\frac{Execution\,time_B}{Execution\,time_A} = \frac{3.5 \times I\,ns}{3.0 \times I\,ns} = 1.167$$

- Machine A is 1.167 faster than machine B for this program

Performance equation

$$CPU\,time = Instruction\,count \times CPI \times Clock\,cycle\,time$$

$$CPU\,time = \frac{Instruction\,count \times CPI}{Clock\,rate}$$

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

Aspect of CPU performance

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

	Instruction Count	CPI	Clock rate
Program	x		
Compiler	x	x	
ISA	x	x	
Organization		x	x
Technology			x

How do we obtain these numbers?

- We can measure CPU execution time
- We can get clock cycle time
- Instruction count and CPI are very difficult to obtain
- Instruction count:
  - Profiler
  - Trace
  - Simulator
- CPI
  - Detail simulation
  - Hand count clock cycle for each instruction

CPI

- **Several different classes of instructions**
- n many instruction classes
- C_i is the count of the number of instructions of class i executed
- CPI_i is the average number of cycles per instruction in class i
- CPU clock cycles

$$CPU\,clock\,cycle = \sum_{i=1}^{n} CPI_i \times C_i$$

$$= CPI_1 \times C_1 + CPI_2 \times C_2 + CPI_3 \times C_3 + \ldots + CPI_n \times C_n$$

Example

- Machine facts

Instruction Class	CPI for this class
A	1
B	2
C	3

- A compiler generates two code sequence

Instruction Count for instruction class			
Code	A	B	C
1	2	1	2
2	4	1	1

- Which code sequence has the most instructions?
- Which one is faster?
- What is the CPI?

Answer

- Code Sequence
  - Sequence 1 : 2 + 1 + 2 = 5 instructions
  - Sequence 2 : 4 + 1 + 1 = 6 instructions
  - Sequence 2 has more instructions
- CPU clock cycles

$$CPU\,clock\,cycle = \sum_{i=1}^{3} CPI_i \times C_i$$

$$= CPI_1 \times C_1 + CPI_2 \times C_2 + CPI_3 \times C_3$$

  - Sequence 1 : (2x1)+(1x2)+(2x3)=10 cycles
  - Sequence 2 : (4x1)+(1x2)+(1x3)= 9 cycles
  - Sequence 2 is faster

Answer

- CPI

$$CPI = \frac{CPU\,clock\,cycles}{Instruction\,Count}$$

$$CPI_1 = \frac{CPU\,clock\,cycles_1}{Instruction\,Count_1} = \frac{10}{5} = 2$$

$$CPI_2 = \frac{CPU\,clock\,cycles_2}{Instruction\,Count_2} = \frac{9}{6} = 1.5$$

A Simple Example

- C_i = Frequency

Operation	Freq	CPI_i	Freq x CPI_i
ALU	50%	1	0.5
Load	20%	5	1
Store	10%	3	0.3
Branch	20%	2	0.4
		$\sum$	2.2

A Simple Example

- Machine A:
  - How much faster would the machine be if a better data cache reduced the average load time to 2 cycles?
- Machine B:
  - How does this compare with using branch prediction to shave a cycle off the branch time?

	Original Machine			Machine A		Machine B	
Operation	Freq	CPI_i	Freq x CPI_i	CPI_i	Freq x CPI_i	CPI_i	Freq x CPI_i
ALU	50%	1	0.5				
Load	20%	5	1				
Store	10%	3	0.3				
Branch	20%	2	0.4				
		∑ =	2.2				

A Simple Example

- Machine A:
  - How much faster would the machine be if a better data cache reduced the average load time to 2 cycles?
- Machine B:
  - How does this compare with using branch prediction to shave a cycle off the branch time?

	Original Machine			Machine A		Machine B	
Operation	Freq	CPI_i	Freq x CPI_i	CPI_i	Freq x CPI_i	CPI_i	Freq x CPI_i
ALU	50%	1	0.5	1	0.5	1	0.5
Load	20%	5	1	2	0.4	5	1
Store	10%	3	0.3	3	0.3	3	0.3
Branch	20%	2	0.4	2	0.4	1	0.2
		∑ =	2.2	∑ =	1.6	∑ =	2

Choosing Programs to Evaluate Performance

- Workload is a set of application programs that the machine runs to measure performance
- Benchmark is a set of programs specifically chosen for measuring performance

	PROs	CONs
Actual Target Workload	representative	very specific non-portable difficult to run, or measure hard to identify cause
Full Benchmarks	Portable Widely used Improvements useful in reality	Less representative
Small “Kernel” Benchmarks	Easy to run early in design cycle	Easy to fool
Micro benchmarks	Identify peak capability and potential bottleneck	Peak may be a long way from application performance

SPEC Benchmarks www.spec.org

Integer benchmarks		FP benchmarks	
gzip	compression	wupwise	Quantum chromodynamics
vpr	FPGA place & route	swim	Shallow water model
gcc	GNU C compiler	mgrid	Multigrid solver in 3D fields
mcf	Combinatorial optimization	applu	Parabolic/elliptic pde
crafty	Chess program	mesa	3D graphics library
parser	Word processing program	galgel	Computational fluid dynamics
eon	Computer visualization	art	Image recognition (NN)
perlbmk	perl application	equake	Seismic wave propagation simulation
gap	Group theory interpreter	facerec	Facial image recognition
vortex	Object oriented database	ammp	Computational chemistry
bzip2	compression	lucas	Primality testing
twolf	Circuit place & route	fma3d	Crash simulation fem
		sixtrack	Nuclear physics accel
		apsi	Pollutant distribution

Metrics

- **Levels of Abstraction**
- Applications : Useful operations per seconds
- Programming Language
- Compiler
- Instruction Set Architecture
  - MIPS : Million Instruction per Seconds
  - MFLOPS : Million Floating Point Operation per Seconds
- Datapath/Control : Megabytes per seconds
- Functional Units
- Transistors : Cycles per Seconds (clock rate)

Comparing and Summarizing

- A is 10 times faster than B for program 1
- B is 10 times faster than A for program 2

	Computer A	Computer B
Program 1	1	10
Program 2	1000	100
Total Time	1001	110

- Total Execution time

$$\frac{Performance_B}{Performance_A} = \frac{Executiontime_A}{Executiontime_B} = \frac{1001}{110} = 9.1$$

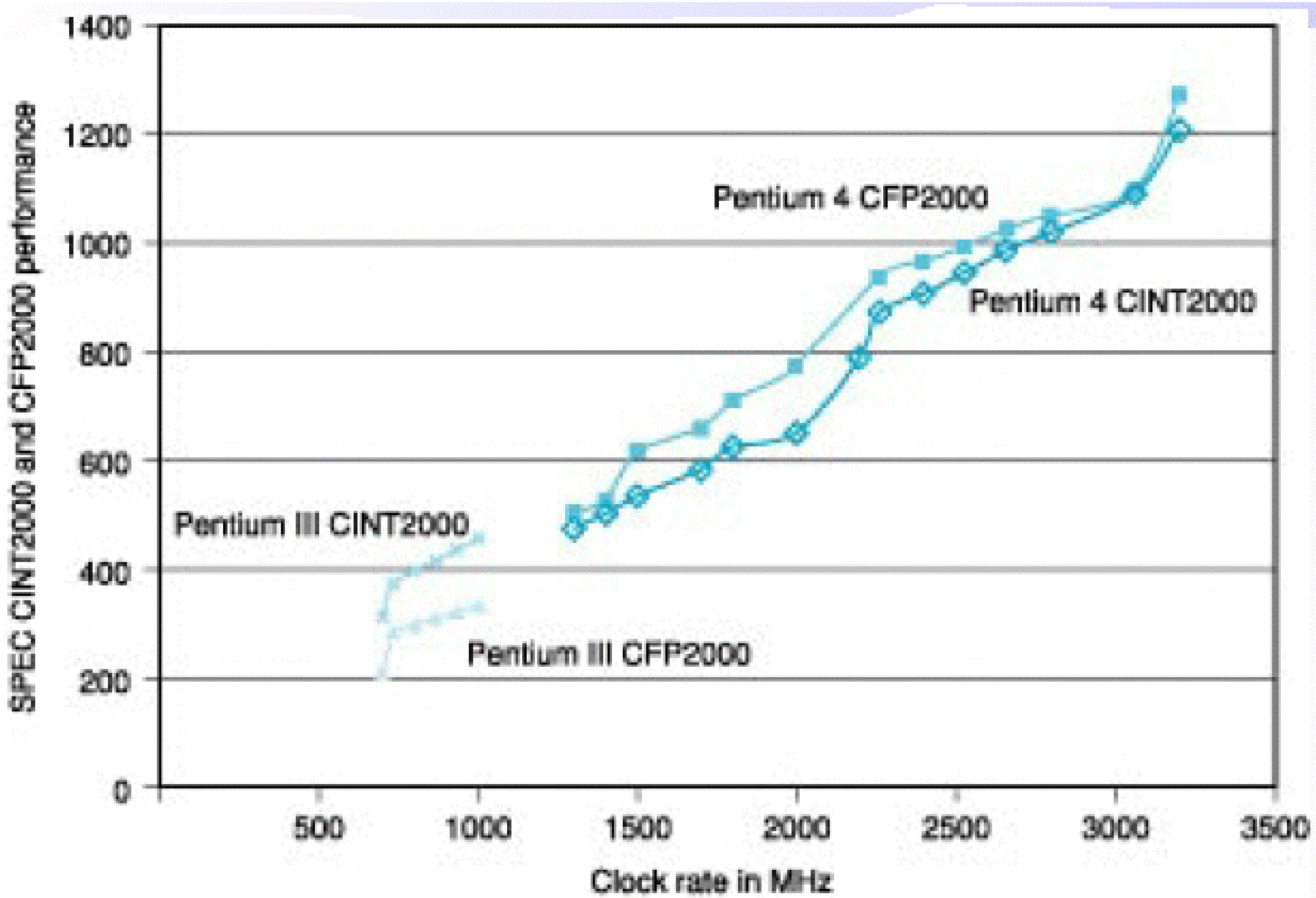
Average Execution Time

- Running multiple programs in a workload

- Average execution time that is directly proportional to total execution time is the arithmetic mean (AM)

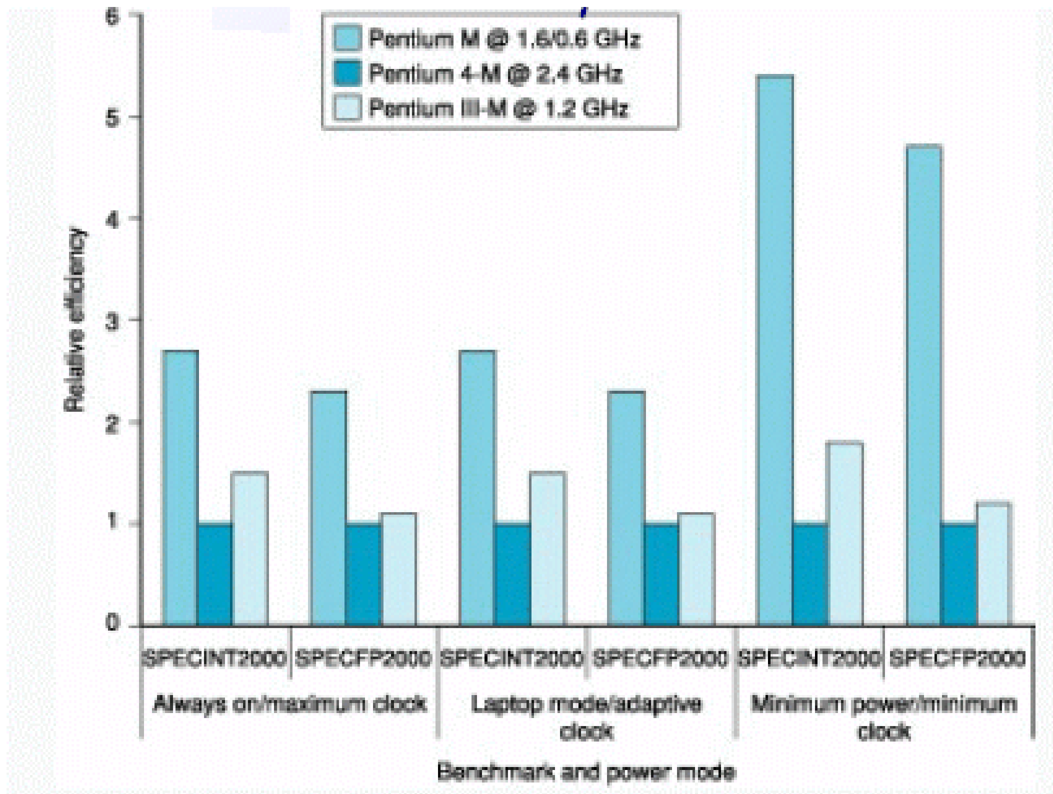
$$AM = \frac{1}{N}\sum_{i=1}^{N} Time_i$$

Example SPEC Rating



Other Performance Metrics

- Power consumption – especially in the embedded market where battery life is important (and passive cooling)
  - For power-limited applications, the most important metric is energy efficiency



Amdahl's Law

- Speedup : how a machine performs after enhancement
- Law of diminishing returns

$$Speedup(E) = \frac{Performance\ with\ E}{Performance\ without\ E} = \frac{Execution\ time\ without\ E}{Execution\ time\ with\ E}$$

$$Execution\ time(E) = Execution\ time\ unaffected + \frac{Execution\ time\ with\ E}{Amount\ of\ Improvement}$$

Example 1

- A program runs on a machine for 10 seconds. 50 % of the time is doing multiplications. If we improve the multiplication unit so that it runs twice as fast, how big is the speedup?

Example 1

- A program runs on a machine for 10 seconds. 50 % of the time is doing multiplications. If we improve the multiplication unit so that it runs twice as fast, how big is the speedup?

$$Extime(E) = \frac{Affected\,extime}{improvement} + unaffected\,extime$$

$$Extime(E) = \frac{5s}{2} + 5s = 7.5s$$

$$Speedup(E) = \frac{10s}{7.5s} = 1.3333$$

- Not two times faster

Example 2

- A program runs for 10 seconds. 70% of the time is doing additions. How much improvement on the additions if we want to reduce the running time to 3 seconds?

Example 2

- A program runs for 10 seconds. 70% of the time is doing additions. How much improvement on the additions if we want to reduce the running time to 3 seconds?

$$Ex\,time(E) = \frac{Affected\,ex\,time}{improvement} + unaffected\,ex\,time$$

$$3s = \frac{7\,s}{n} + (10 - 7)\,s$$

$$3s = \frac{7\,s}{n} + 3\,s$$

$$0 = \frac{7\,s}{n}$$

- No amount of improvement can reduce the running time to 3 seconds.

MIPS

- Instruction Rate

$$MIPS = \frac{Instruction\,Count}{Execution\,time \times 10^6}$$

- Faster machine have higher MIPS rating (?)

Example

Instruction Class	CPI for this class
A	1
B	2
C	3

	Instruction count (billions)		
Code from	A	B	C
Compiler 1	5	1	1
Compiler 2	10	1	1

- Assume the machine is running at 500 Mhz.
  - Which one is faster according to execution time?
  - Which one is faster according to MIPS?

Answer

- **Execution Time**

$$execution\ time = \frac{CPU\ clock\ cycle}{clock\ rate}$$

$$CPU\ clock\ cycle = \sum_{i=1}^{n} CPI_i \times C_i$$

- CPU clock $cyle_1$ = (5x1)+(1x2)+(1x3)x10^9 = 10x10^9
- CPU clock $cyle_2$ = (10x1)+(1x2)+(1x3)x10^9 = 15x10^9
- Execution $time_1$ = (10x10^9)/(500x10^6) = 20 s
- Execution $time_2$ = (15x10^9)/(500x10^6) = 30 s
- Compiler 1 produces a faster program

Answer

- MIPS

$$MIPS = \frac{Instruction\,Count}{Execution\,time \times 10^6}$$

$$MIPS_1 = \frac{(5+1+1)\times 10^9}{20\times 10^6} = 350$$

$$MIPS_1 = \frac{(10+1+1)\times 10^9}{30\times 10^6} = 400$$

- Compiler 2 is faster -> MIPS fails