16 Performance (1)
CSC 230

Department of Computer Science
University of Victoria

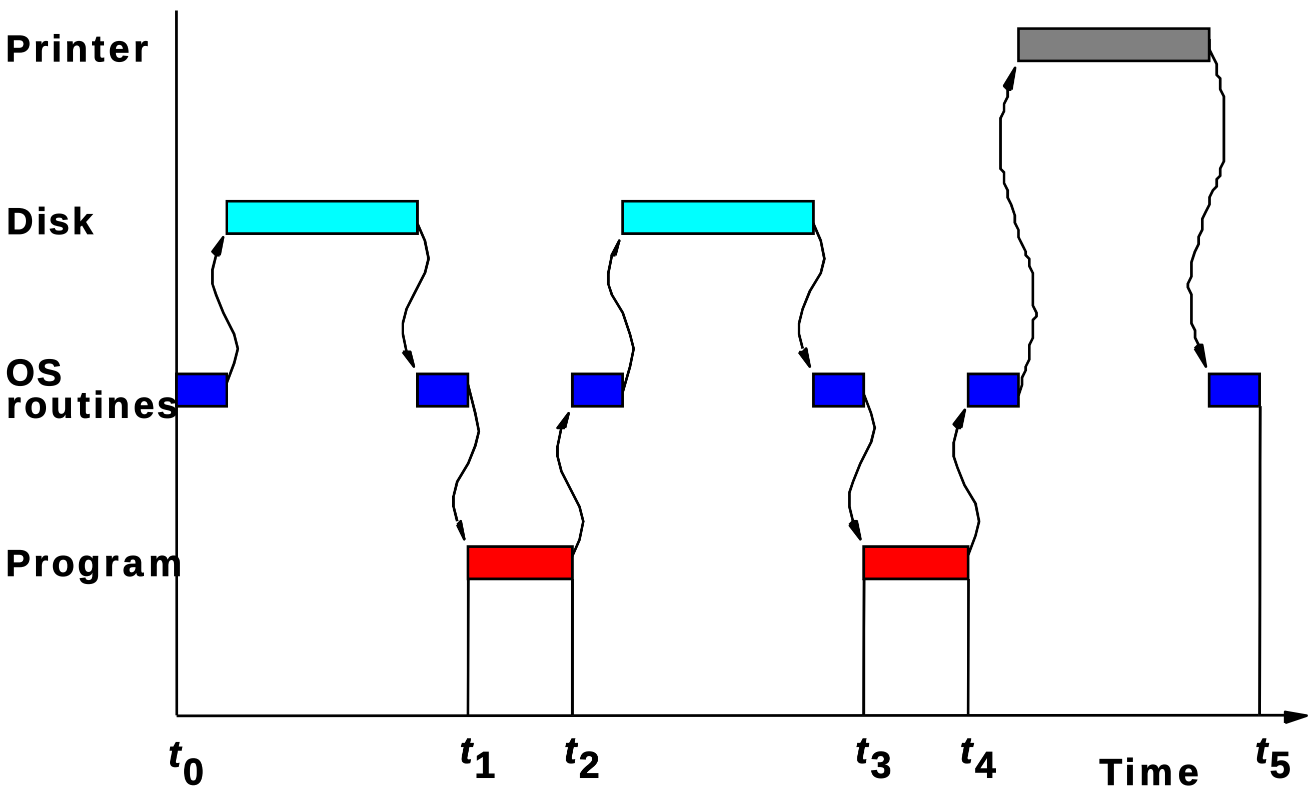
M&H: 6.5, 6.6

Stallings: 2.6 (skip Amdhal's law till later)

Performance issues: an introduction

how quickly can a program be executed?

compiler   instruction set   hardware

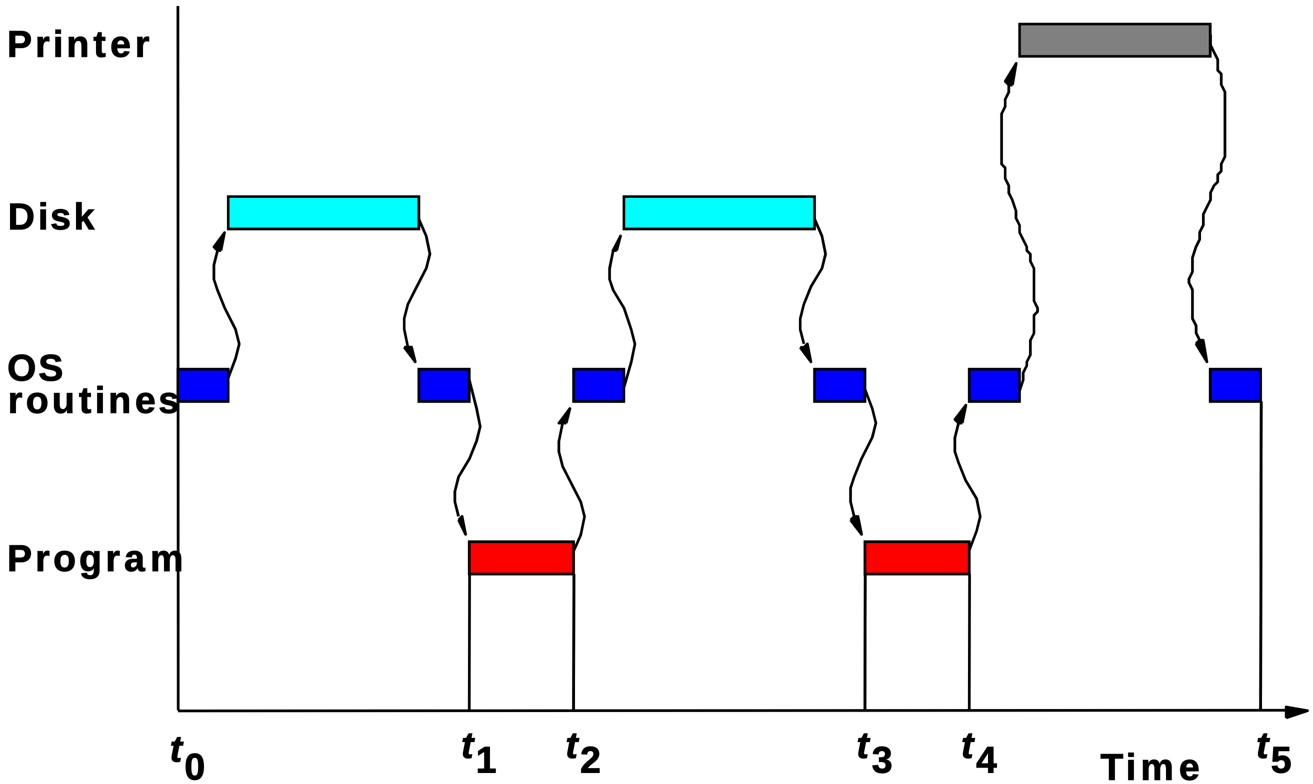


Review

Execution = OS time + program time

Unix "time" command

how long does a program take?



Answer is in 3 numbers:

- real
- user
- system

real	=	**(t0 to t5)**
user	=	**(t1 to t2) + (t3 to t4)**
system	=	**(t0 to t1) + (t2 to t3) + (t4 to t5)**

Execution Time

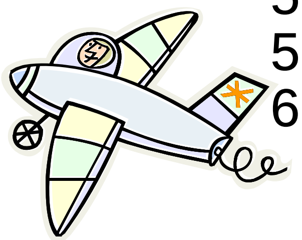
- **Elapsed Time (or Real Time)**
  - counts everything *(disk and memory accesses, I/O, etc.)*
  - a useful number, but often not good for comparison purposes

- **CPU time**
  - doesn't count I/O or time spent running other programs
  - can be broken up into *system* time and *user* time

- **Our focus: user CPU time**
  - time spent executing the lines of code that are "in" our program

Which of these airplanes has the best performance?

Airplane	Passengers	Range (mi)	Speed (mph)
Boeing 737-100	118	1860	485
Boeing 747-400	416	7284	584
Boeing 787-8	210	8200	561
BAC/Sud Concorde	128	4500	1350
Douglas DC-8-50	146	8720	544
Airbus A320-200	148	3000	594
Airbus A340-200	261	8000	557
Airbus A380-800	525 to 853	8200	647



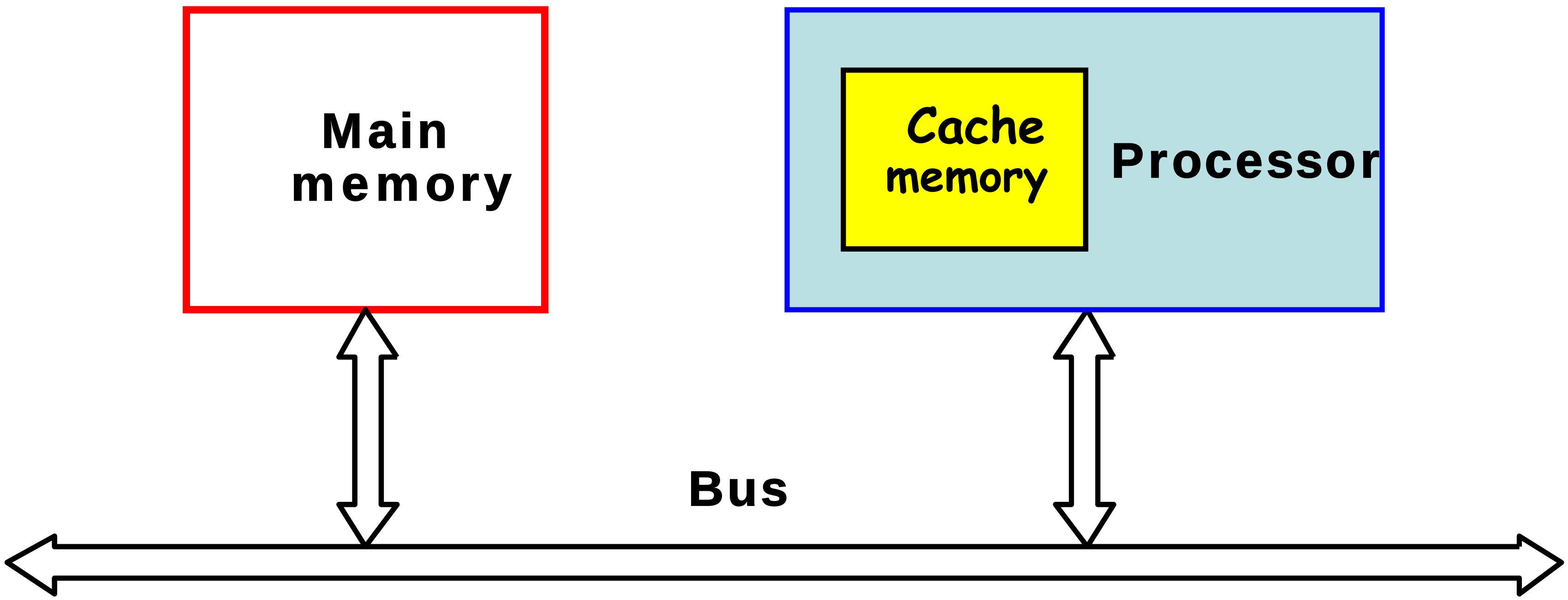
- ✓ **Which plane delivers the greatest capacity on a route? (As measured in passenger-miles per hour)**
- ✓ **What factors does an airline consider when purchasing a fleet of planes?**

Computer Performance: TIME, TIME, TIME

- **Response Time (latency)**
  - **How long does it take for a job to run?**
  - **How long does it take to execute a job?**
  - **How long must one wait for the database query?**
- **Throughput**
  - **How many jobs can the machine run at once?**
  - **What is the average execution rate?**
  - **How much work is getting done?**
  - **How many processes are executed in a unit of time?**

✓ ***If we upgrade a machine with a new processor what do we increase?***

✓ ***If we add a new machine to the lab what do we increase?***

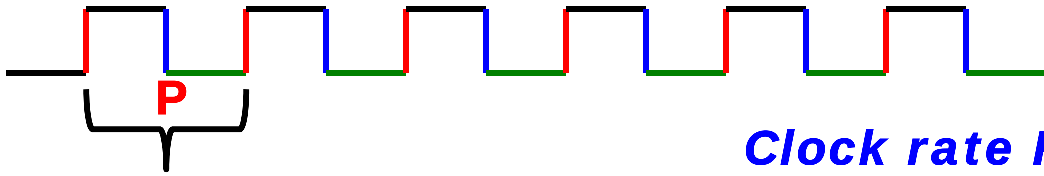


Fetch-Decode-Execute Cycle

Processor speed versus memory access speed

Processor Clock

- **Clock is a timing signal**
- **It defines intervals called clock cycles, each of length P**



Clock rate R = 1 / P

- **A machine instruction is divided into small steps**

  ➔ 1 clock cycle per step

- **"Cycles per second" is a measure known as *Hertz (Hz)***

500 million cycles per second = 500 megahertz or 500 MHz

➔ 1 cycle every 2 ns

Is the clock speed the most important factor?

Two small examples

If P = 25 ns/cycle,

Then $R = \dfrac{1}{25} \dfrac{cycles}{ns}$

= 0.04 cycles/ns

Translate to seconds → multiply by 10^9

→ 40,000,000 cycles/sec

→ 40 MHz is the clock rate R

Reminders

1 ns = 10^{-9} sec

1 ms = 10^{-3} sec

1 sec = 10^9 ns = 10^3 ms

1 MHz = 10^6 hertz

1 GHz = 10^9 hertz

R = 1/P where R is clock rate, P time for 1 clock cycle

If R = 1GHz (clock rate) then

1GHz = 10^9 (hertz) cycles/sec = 1,000,000,000 cycles/sec

→ Divide by 10^9 to get cycles/ns

→ (1,000,000,000/ 10^9) = 1 cycles/ns = R

Since R = 1/P

→ invert to get ns/cycles

$$R = \frac{1}{1} \frac{cycles}{ns} \Rightarrow P = \frac{1}{1} \frac{ns}{cycles}$$

Basic Performance Equation

$$T = \frac{N \times S}{R}$$

T = performance (time) for a program

N = actual number of executed instructions (not just number of instructions, i.e. loops are multiplied by number of iterations)

S = number of basic 1-clock steps needed for one instruction

R = clock rate (cycles per second)

GOAL: reduce T → reduce (N or S or both) AND (increase R)

NOTE 1: these parameters are NOT independent

NOTE 2: architectural issues affect them

Performance Helpers

- **Cache (done here)**
- **Pipelining (will do here an introduction)**
- **Superscalar architectures (will do here an introduction)**
- **Optimizing compilers (a bit)**
- **CISC (Complex Instruction Set Computers)**
- **RISC (Reduced Instruction Set Computers)**

Increase in performance is proportional to the square root of the complexity
- 2 simpler components better than 1 large complex one?

SPEC Ratings

- SPEC (System Performance Evaluation Corporation)
- SPEC Benchmarks (SPEC CPU2000 benchmarks use a 300-MHz Sun Ultra5_10 as the reference machine)

$$\text{SPEC rating} = \frac{\text{running time on reference computer}}{\text{running time on computer under test}}$$

For example, the Intel D925XECV2 motherboard (3.80 GHz, Pentium 4 processor 570J) has a SPECint2000 rating of 1671.

See http://www.spec.org/cpu2006/results/

All these concepts will be reviewed and re-evaluated after the description of more architectural issues

Higher Performance ⟺ Lower Execution Time

$$\text{Performance} = \frac{1}{\text{ExecutionTime}}$$

X is *n* times faster than Y ➔

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

If X is *n* times faster than Y, then the execution time on Y is n times longer than on X

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

CPU Performance and clock

$$\text{CPU Execution time for a program} = \text{\# CPU clock cycles for a program} \times \text{clock cycle time}$$

$$\text{CPU Execution time for a program} = \frac{\text{\# CPU clock cycles for a program}}{\text{Clock rate}}$$

Independent of the number of instructions in a program

CPI: clock cycles per instruction

Think of execution time as:

$$\text{CPU clock cycles} = \begin{array}{c}\text{Instruction count}\\ \text{for a program}\end{array} \times \begin{array}{c}\text{Average clock cycles}\\ \text{per instruction (CPI)}\end{array}$$

OR

$$\text{CPI} = \frac{\text{CPU clock cycles}}{\text{Instruction count}}$$

Small Example: same program on 2 machines

CPU A:

CPI = 5 *(i.e. 5 clock cycles per instruction)*

Clock period P = 100 ns

CPU B:

CPI = 3.5 *(i.e. 3.5 clock cycles per instruction)*

Clock period P = 120 ns

Test yourself: what is the clock rate R for each CPU?

Small Example: same program on 2 machines

CPU A:

CPI = 5 *(i.e. 5 clock cycles per instruction)*

Clock period P = 100 ns

CPU B:

CPI = 3.5 *(i.e. 3.5 clock cycles per instruction)*

Clock period P = 120 ns

Clock A = 25% faster

Test yourself: what is the clock rate R for each CPU?

CPU A:

Clock period = 100 ns/cycle

$$\Rightarrow R = \frac{1}{100} \text{ cycles / ns}$$

= 0.01 cycles/ns

Multiply by 10^9 to get seconds

= 10,000,000

= 10 MHz clock rate R

CPU B:

Clock period = 120 ns/cycle

$$\Rightarrow R = \frac{1}{120} \text{ cycles / ns}$$

= 0.008 cycles/ns

Multiply by 10^9 to get seconds

= 8,000,000

= 8 MHz clock rate R

Small Example: same program on 2 machines

CPU A:

CPI = 5 *(i.e. 5 clock cycles per instruction)*

Clock period P = 100 ns

CPU B:

CPI = 3.5 *(i.e. 3.5 clock cycles per instruction)*

Clock period P = 120 ns

Let I = number of instructions

CPU clock cycles [A] = I x 5

CPU clock cycles [B] = I x 3.5

CPU time = CPU clock cycles x clock cycle time

CPU time[A] = I x 5 x 100 ns = I x 500 ns

CPU time[B] = I x 3.5 x 120 ns = I x 420 ns

How much faster is B?

$$\frac{\text{Performance}_B}{\text{Performance}_A} = \frac{\text{Execution time}_A}{\text{Execution time}_B} = \frac{I x 500 ns}{I x 420 ns} = 1.19$$

➔ **B is 19% faster than A**

Classic CPU Performance Equation

CPU time = Instruction count x CPI x Clock cycle time

OR

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

Comparing Code Segments

A compiler designer needs to decide between two code sequences for a system. Here are the data from hardware for each class of instructions which can be used:

CPI for each instruction class

	A	B	C
CPI	1	2	3

Here is the data for each code sequence:

Code Sequence	**Instruction count for each instruction class**		
	A	**B**	**C**
1	2	1	2
2	4	1	1

Which Code Segment executes the most instructions?

Sequence 1 = 2+1+2 = 5 instructions

Sequence 2 = 4+1+1 = 6 instructions

Code Sequence	**Instruction count for each instruction class**		
	A	**B**	**C**
1	2	1	2
2	4	1	1

CPI for each instruction class

	A	**B**	**C**
CPI	1	2	3

Which Code Segment is faster?

Need to compute total CPU clock cycles, using CPI and instruction count

CPU clock cycles = for each class (Instruction count x CPI)

CPU clock cycles[1] = (2x1) + (1x2) + (2x3) = 10 cycles

CPU clock cycles[2] = (4x1) + (1x2) + (1x3) = 9 cycles

Sequence 2 is faster even if it executes more instructions

Code Sequence	Instruction count for each instruction class		
	A	B	C
1	2	1	2
2	4	1	1

CPI for each instruction class

	A	B	C
CPI	1	2	3

What is the CPI for each sequence?

Since Sequence 2 is faster even if it has more instructions, it should have a lower CPI overall

$$\text{CPI} = \frac{\text{CPU clock cycles}}{\text{Instruction count}}$$

$$\text{CPI}_1 = \frac{\text{CPU clock cycles}}{\text{Instruction count}} = \frac{10}{5} = 2.0$$

$$\text{CPI}_2 = \frac{\text{CPU clock cycles}}{\text{Instruction count}} = \frac{9}{6} = 1.5$$

Code Sequence	Instruction count for each instruction class: A	B	C
1	2	1	2
2	4	1	1

CPI for each instruction class

	A	B	C
CPI	1	2	3

Instruction Frequency

Frequency of occurrence of instruction types for a variety of languages. The percentages do not sum to 100 due to roundoff. (Adapted from Knuth, D. E., *An Empirical Study of FORTRAN Programs, Software—Practice and Experience*, 1, 105-133, 1971.)

Statement	Average Percent of Time
Assignment	47
If	23
Call	15
Loop	6
Goto	3
Other	7

Complexity of Assignments

Percentages showing complexity of assignments and procedure calls. (Adapted from Tanenbaum, A., *Structured Computer Organization*, 4/e, Prentice Hall, Upper Saddle River, New Jersey, 1999.)

	Percentage of Number of Terms in Assignments	Percentage of Number of Locals in Procedures	Percentage of Number of Parameters in Procedure Calls
0	–	22	41
1	80	17	19
2	15	20	15
3	3	14	9
4	2	8	7
≥ 5	0	20	8

Speedup: another view

Speedup *S* is the ratio of the time needed to execute a program without improvement (T_{wo}) to the time required with an improvement (T_w)

$$S = \frac{T_{wo}}{T_w} \qquad S = \frac{T_{wo} - T_w}{T_w} \times 100$$

Time T is computed as: (the instruction count IC) X (the number of cycles per instruction CPI) X (the cycle time t)

$$T = IC \times CPI \times t$$

Substituting T into the speedup percentage calculation above yields:

$$S = \frac{(IC_{wo} \times CPI_{wo} \times t_{wo}) - (IC_w \times CPI_w \times t_w)}{(IC_w \times CPI_w \times t_w)} \times 100$$

Speedup: another view

Speedup *S* is the ratio of the time needed to execute a program without improvement (T_{wo}) to the time required with an improvement (T_w)

$$S = \frac{T_{wo}}{T_w} \qquad S = \frac{T_{wo} - T_w}{T_w} \times 100$$

Time T is computed as: (the instruction count IC) X (the number of cycles per instruction CPI) X (the cycle time t)

$$T = IC \times CPI \times \tau$$

Substituting T into the speedup percentage calculation above yields:

$$S = \frac{IC_{wo} \times CPI_{wo} \times \tau_{wo} - IC_w \times CPI_w \times \tau_w}{IC_w \times CPI_w \times \tau_w} \times 100$$