Chapter 2

Adaptado do material de aula de Patterson & Hennessy

Performance

- **Measure, Report, and Summarize**
- **Make intelligent choices**
- **See through the marketing hype**
- **Key to understanding underlying organizational motivation**

Why is some hardware better than others for different programs?

What factors of system performance are hardware related?
(e.g., Do we need a new machine, or a new operating system?)
(utilização efetiva do conj. de instruções pela aplicação, implementação do hardware-arquitetura, desempenho do sistema de memória e do sistema de I/O…)

How does the machine's instruction set affect performance?

- Tendo somente o manual do conjunto de instruções e uma aplicação/sistema a ser executado, é impossível de determinar a performance deste sistema.

- Diferentes tipos de métricas de desempenho podem ser aplicadas para diferentes tipos de aplicações, assim como diferentes aspectos podem ser avaliados para determinar o desempenho

Performance - Example

- **Suponha a execução de um programa em 2 workstations diferentes**
  - ***A estação mais rápida é aquela que termina a execução primeiro***
- **Suponha a execução de programas em 2 workstations compartilhadas**
  - ***A estação mais rápida é aquela que completa a execução de mais programas durante um período de tempo.***
- **Usuário tradicional: reduzir o tempo de resposta**
  - ***Tempo entre o início e o fim de uma tarefa, também chamado de tempo de execução.***
- **Usuário tradicional: aumentar o throughput**
  - ***Quantidade de trabalho realizado em um dado intervalo de tempo***

Which of these airplanes has the best performance?

Airplane	Passengers	Range (mi)	Speed (mph)
Boeing 737-100	101	630	598
Boeing 747	470	4150	610
BAC/Sud Concorde	132	4000	1350
Douglas DC-8-50	146	8720	544

- **How much faster is the Concorde compared to the 747?**
- **How much bigger is the 747 than the Douglas DC-8?**

Computer Performance: TIME, TIME, TIME

- **Response Time (latency)**
  - **— How long does it take for my job to run?**
  - **— How long does it take to execute a job?**
  - **— How long must I 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?**
- **Tempo de Resposta x Throughput:**
  - ***If we upgrade a machine with a new processor what do we increase (substituir o processador por outro mais rápido)?***
  - ***If we add a new machine to the lab what do we increase (colocar processadores adicionais permitindo a utilização de múltiplos processadores para tarefas separadas)?***

Execution Time

- Elapsed 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

Book's Definition of Performance

- For some program running on machine X,

  $\text{Desempenho}_X = 1 / \text{Tempo de Execução}_X$

- "X is n times faster than Y"

  $\text{Desempenho}_X / \text{Desempenho}_Y = n$ ou

- “Y is n times slower than X"

  $\text{Tempo de Execução}_y / \text{Tempo de Execução}_x = n$

- Comparação entre 2 máquinas, X e Y:

  $\text{Desempenho}_X > \text{Desempenho}_Y$ ou

  $1 / \text{Tempo de Execução}_X > 1 / \text{Tempo de Execução}_y$ ou

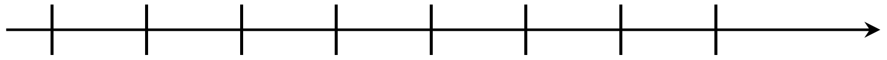
  $\text{Tempo de Execução}_X < \text{Tempo de Execução}_y$

Clock Cycles

- **Instead of reporting execution time in seconds, we often use cycles**

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

- **Clock "ticks" indicate when to start activities (one abstraction):**



- **cycle time = time between ticks = seconds per cycle**
- **clock rate (frequency) = cycles per second (1 Hz. = 1 cycle/sec)**

A 200 Mhz. clock has a $\frac{1}{200 \times 10^6} \times 10^9 = 5 \text{ nanoseconds}$ **cycle time**

How to Improve Performance

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

So, to improve performance (everything else being equal) you can either

________ the # of required cycles for a program, or

________ the clock cycle time or, said another way,

________ the clock rate.

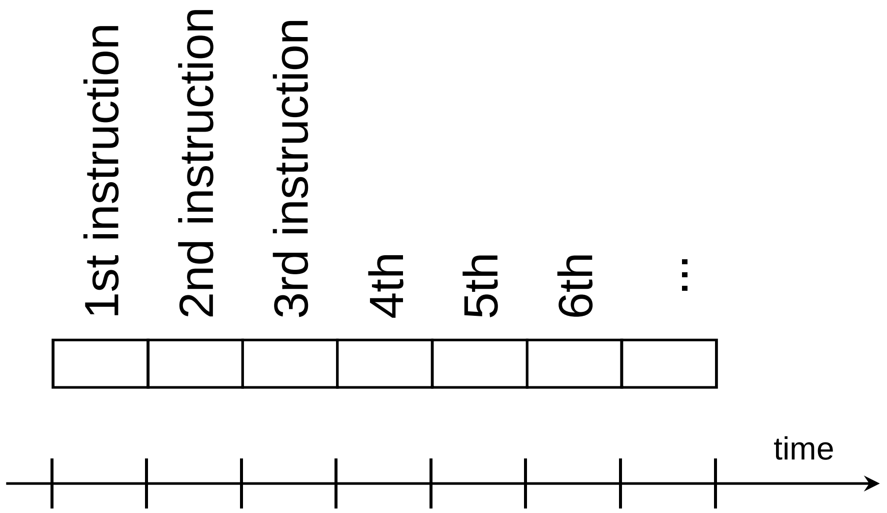
Desta forma, tem-se:

Tempo de execução = nro. ciclos de clock * período do clock ou

nro. ciclos de clock / freqüência do clock

How many cycles are required for a program?

- **Could assume that # of cycles = # of instructions**

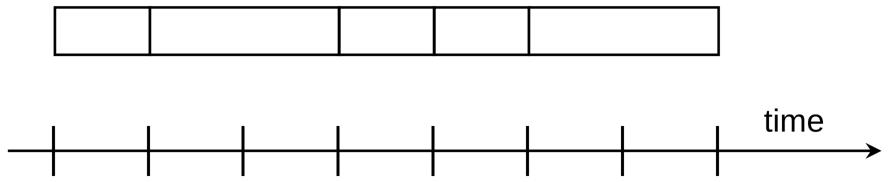


This assumption is incorrect,

different instructions take different amounts of time on different machines.

Why? hint: remember that these are machine instructions, not lines of C code

Different numbers of cycles for different instructions



- Multiplication takes more time than addition
- Floating point operations take longer than integer ones
- Accessing memory takes more time than accessing registers

- ***Important point: changing the cycle time often changes the number of cycles required for various instructions (more later)***

Example

- Our favorite program runs in 10 seconds on computer A, which has a 400 Mhz. clock. We are trying to help a computer designer build a new machine B, that will run this program in 6 seconds. The designer can use new (or perhaps more expensive) technology to substantially increase the clock rate, but has informed us that this increase will affect the rest of the CPU design, causing machine B to require 1.2 times as many clock cycles as machine A for the same program.   What clock rate should we tell the designer to target?"

- **Don't Panic, can easily work this out from basic principles**

Now that we understand cycles

- A given program will require
  - some number of instructions (machine instructions)
  - some number of cycles
  - some number of seconds
- We have a vocubulary that relates these quantities:
  - cycle time (seconds per cycle)
  - clock rate (cycles per second)
  - CPI (cycles per instruction)

    a floating point intensive application might have a higher CPI
  - MIPS (millions of instructions per second)

    this would be higher for a program using simple instructions

CPI

- Considerando o nro. de instruções no calculo de desempenho:
  - Nro. de ciclos de clock = nro. de instruções * CPI
  - Como diferentes instruções executam em número de ciclos de clock diferentes, <u>a CPI é uma média ponderada</u> de todas as instruções executadas do programa

Performance

- Performance is determined by execution time
- Do any of the other variables equal performance?
  - # of cycles to execute program?
  - # of instructions in program?
  - # of cycles per second?
  - average # of cycles per instruction?
  - average # of instructions per second?

- Common pitfall: thinking one of the variables is indicative of performance when it really isn’t.

CPI Example

- **Suppose we have two implementations of the same instruction set architecture (ISA).**

  For some program,

  Machine A has a clock cycle time of 1 ns. and a CPI of 2.0
  Machine B has a clock cycle time of 2 ns. and a CPI of 1.2

  What machine is faster for this program, and by how much?

- ***If two machines have the same ISA which of our quantities (e.g., clock rate, CPI, execution time, # of instructions, MIPS) will always be identical?***

Equação Básica de Desempenho

- **A partir do exemplo anterior, podemos escrever a equação básica de desempenho:**

  – ***Tempo de execução CPU = nro. de instruções * CPI * período de clock ou***

  (nro. de instruções * CPI) / freqüência

# of Instructions Example

- A compiler designer is trying to decide between two code sequences for a particular machine. Based on the hardware implementation, there are three different classes of instructions: Class A, Class B, and Class C, and they require one, two, and three cycles (respectively).

  The first code sequence has 5 instructions: 2 of A, 1 of B, and 2 of C
  The second sequence has 6 instructions: 4 of A, 1 of B, and 1 of C.

Classe de Instrução	CPI por Classe
A	1
B	2
C	3

Seq. de Código	Nro. Inst. por Classe		
	A	B	C
1	2	1	2
2	4	1	1

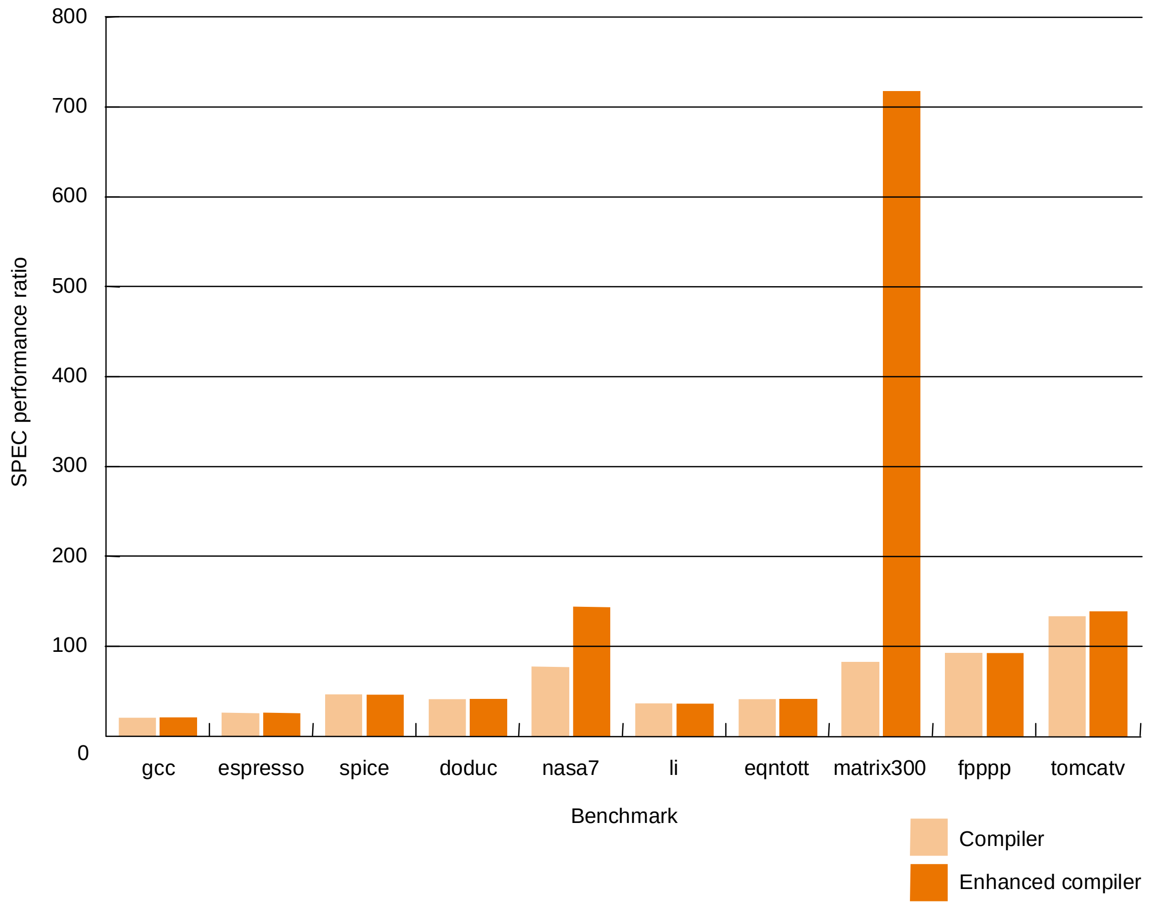
- Qual seq. executa mais instruções? Qual será mais rápida? Qual o CPI para cada seq.?

Benchmarks

- Performance best determined by running a real application
  - Use programs typical of expected workload
  - Or, typical of expected class of applications
    e.g., compilers/editors, scientific applications, graphics, etc.
- Small benchmarks
  - nice for architects and designers
  - easy to standardize
  - can be abused
- SPEC (System Performance Evaluation Cooperative)
  - companies have agreed on a set of real program and inputs
  - can still be abused (Intel’s “other” bug)
  - valuable indicator of performance (and compiler technology)

SPEC ‘89

- **Compiler “enhancements” and performance**



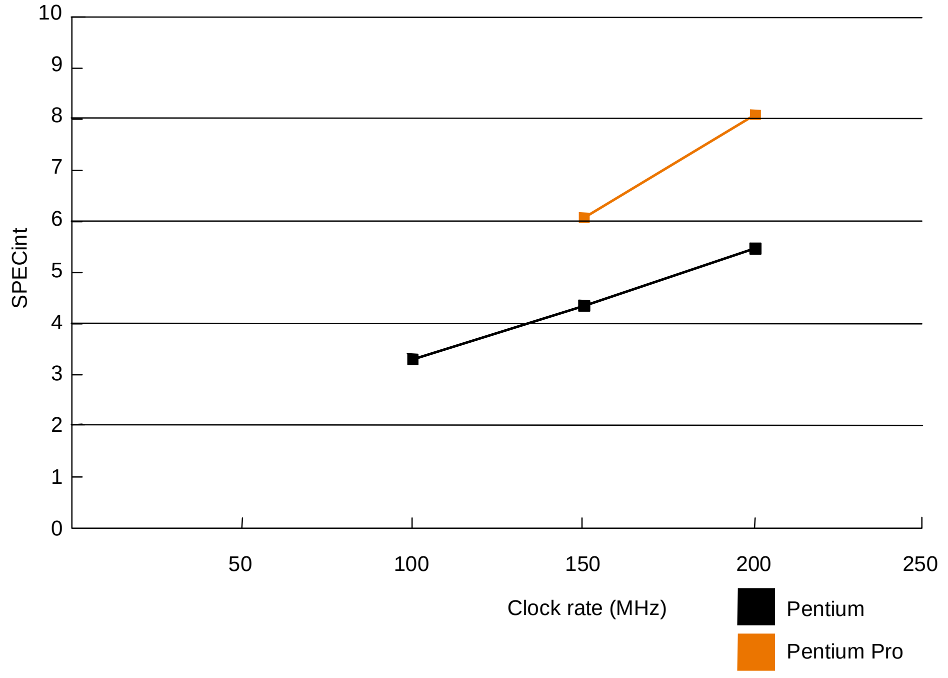
SPEC ‘95

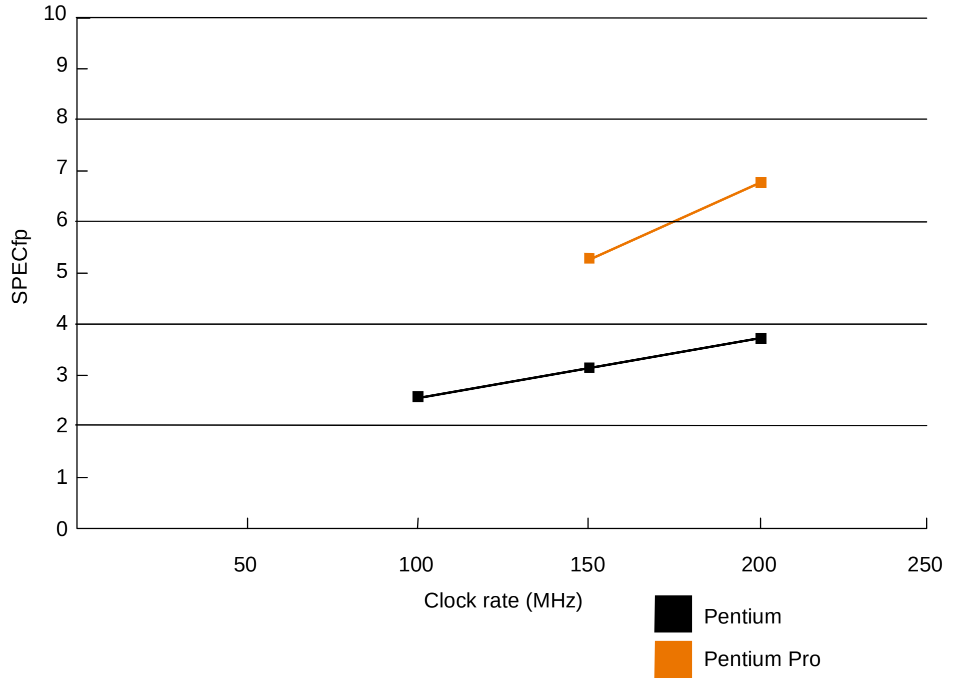
Benchmark	Description
go	Artificial intelligence; plays the game of Go
m88ksim	Motorola 88k chip simulator; runs test program
gcc	The Gnu C compiler generating SPARC code
compress	Compresses and decompresses file in memory
li	Lisp interpreter
ijpeg	Graphic compression and decompression
perl	Manipulates strings and prime numbers in the special-purpose programming language Perl
vortex	A database program
tomcatv	A mesh generation program
swim	Shallow water model with 513 x 513 grid
su2cor	quantum physics; Monte Carlo simulation
hydro2d	Astrophysics; Hydrodynamic Naiver Stokes equations
mgrid	Multigrid solver in 3-D potential field
applu	Parabolic/elliptic partial differential equations
trub3d	Simulates isotropic, homogeneous turbulence in a cube
apsi	Solves problems regarding temperature, wind velocity, and distribution of pollutant
fpppp	Quantum chemistry
wave5	Plasma physics; electromagnetic particle simulation

SPEC '95

Does doubling the clock rate double the performance?

Can a machine with a slower clock rate have better performance?





Amdahl's Law

Execution Time After Improvement =

Execution Time Unaffected +(Execution Time Affected / Amount of Improvement)

- **Example:**

  "Suppose a program runs in 100 seconds on a machine, with multiply responsible for 80 seconds of this time. How much do we have to improve the speed of multiplication if we want the program to run 4 times faster?"

  How about making it 5 times faster?

- ***Principle: Make the common case fast***

Remember

- Performance is specific to a particular program/s
  - Total execution time is a consistent summary of performance

- For a given architecture performance increases come from:
  - increases in clock rate (without adverse CPI affects)
  - improvements in processor organization that lower CPI
  - compiler enhancements that lower CPI and/or instruction count

- Pitfall: expecting improvement in one aspect of a machine's

  performance to affect the total performance

- You should not always believe everything you read! Read carefully!