Performance Analysis

CS6133 - Computer Architecture I

Vikram Padman

Polytechnic Institute of New York University

vikram@poly.edu

Agenda

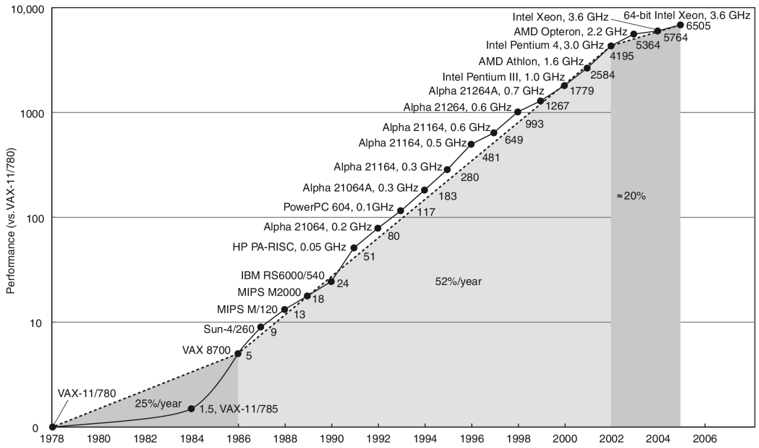
Performance

- Performance is a quantitative measure of how well a system works
- Why is performance important?
  - To compare and contrast computing system. Which system has the best performance for the price
  - For trade studies: Evaluate a system for a given application (its usefulness)
  - To understand a system's underlying architecture
  - To forecast future needs and estimate improvement cost

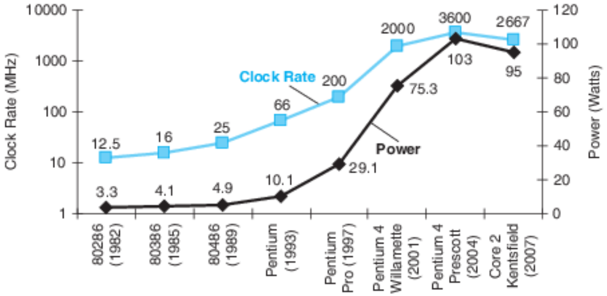
Performance

- A Car's Performance metrics:
    - Engine's horse power
    - 0-60Mph and/or 1/4 mile time
    - Miles per gallon
    - Braking distance
    - Seating and cargo capacity
- A computer's performance metrics:
    - Power
    - Clock speed
    - Battery life
    - Execution time / Responsiveness
    - Storage capacity

Performance since 1980's



The Power Wall



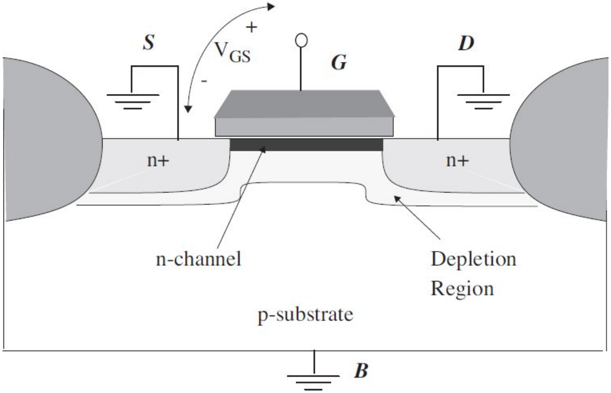
- Power dissipation is an issue in today's technology.
- Smaller transistors could switch faster and more could be packed in a given area
- Smaller transistors ⇒ smaller wires. Smaller wires increase resistance and capacitance.

The Power Wall

- **Dynamic Power**: Power consumed by a active transistor
- **Static Power**: Quiescent or leakage power consumed by an idle transistors
- $Power_{dynamic} = 1/2 * C * V^2 * T_r$
- $Power_{static} = I_{leakage} * V$
- C = Total (gate + wire) capacitance per transistor
- V = Rail voltage, T_r = Toggle rate, I = Current
- Total power for a chip: $Power_{total} = (Power_{dynamic} + Power_{static}) * N_t$
- N_t = Number of transistors

The Scaling Wall

A MOSFET Transistor



The Scaling Wall

- Reduction in transistor's size increases leakage current
- Increase in transistor count increases static power
- Reduction in rail voltage could introduce instability due to process variations
- Increase in frequency increases dynamic power

Performance From A User's Perspective

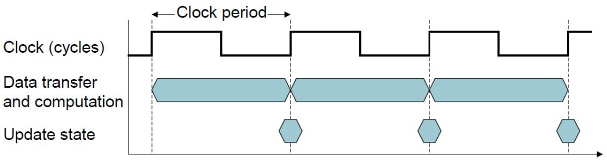
- **Execution Time** = Total run time of a program
  - Time could simply be "wall-clock time" or "elapsed time"
- **Performance** = 1 / (Execution time)
  - Clock speed, parallelism, I/O speed, and work load determines performance
- **Relative Performance** = Performance of one system compared to another
  - $R_p = Performance_A / Performance_B = Execution_B / Execution_A$
  - System A is R_p time faster than system B
- R_p accuracy depends on how execution time is measured in each system and its consistency

Cost-Performance Analysis

- Cost-Performance ratio:
  C_r = Cost * Execution time or Cost / Performance
- Let's consider system A and B running application App_1 and App_2
  - Execution time of App_1 in A = 8 and cost of A = 10, $C_r = 80$
  - Execution time of App_2 in A = 1 and cost of A = 10, $C_r = 10$
  - Execution time of App_1 in B = 2 and cost of A = 15, $C_r = 30$
  - Execution time of App_2 in B = 2 and cost of A = 15, $C_r = 30$

CPU's Performance Measurement

- CPU's performance is measured by the number of clock cycles it consumes to execute a given program
- Clock rate $=$ cycles per second, measured in Hz, Khz or Ghz
- Clock period $=$ time between rising edge of a clock signal



CPU's Performance Measurement

CPU time consumed by a given program =

- $CPU_{time} = I_{tot} * Cycles_i * Clockrate$
- I_{tot}: Total number of Instructions
  - Depends on CPU's architecture, ISA, compiler, coding style and application
- $Cycles_i$: Clock cycles per instruction
  - Depends on CPU's architecture and ISA
- $Clockrate$:
  - ASIC design, logic design, node size and silicon manufacturing technology

Clock Cycles Per Instruction (CPI)

- **CPI**: *Average* number of clock cycles per instruction: CPI $= C_{el}/I_{tot}$
    - I_{tot} = Number of instruction in a program. A program could have many types of instruction(s)
    - C_{el} = Number of elapsed clock cycle(s)
- Let consider a program AP_1
    - N = types of instruction in a program AP_1
    - I_t = number of instruction of type t in AP_1
      $\Rightarrow I_{tot} = \sum_{t=1}^{N} I_t$
    - CPI_t = clock cycles(s) to execute 1 instruction of type t
      $C_{el} = \sum_{t=1}^{N}(I_t * CPI_t)$
      $\Rightarrow CPI = C_{el}/I_{tot} = \sum_{t=1}^{N}(I_t/I_{tot} * CPI_t)$

Clock Cycles Per Instruction (CPI)

Continuing with AP_1

- Suppose AP_1 has 100 instructions (I_{tot}) and below is the distribution:

t	Instruction	I_t	CPI_t	$I_t/I_{tot} * CPI_t$
1	load	35	4	$35/100 * 4 = 1.4$
2	store	10	2	$10/100 * 2 = 0.2$
3	mul	15	35	$15/100 * 35 = 5.25$
4	jump	15	10	$15/100 * 10 = 1.5$
5	add	15	1	$20/100 * 1 = 0.2$
6	sub	10	1	$10/100 * 1 = 0.1$
$CPI = \sum_{t=1}^{6} (I_t/I_{tot}) * CPI_t$				8.65

- $C_{el} = CPU_{time} * ClockRate = I_{tot} * CPI$
- For example: AP_2 with $10 * 10^7$ instructions takes a second to complete on a $500Mhz$ CPU than CPI $= 1 * 500 * 10^6/(10 * 10^7) = 5$

MIPS - Million Instructions Per Second and DMIPS

- **MIPS** A theoretical peak instruction execution rate
- $MIPS = I_{tot}/(CPU_{time} * 10^6$
  $\Rightarrow MIPS = I_{tot}/(C_{el} * CycleTime * 10^6)$
  $\Rightarrow MIPS = (I_{tot} * ClockRate)/(I_{tot} * CPI * 10^6)$
  $\Rightarrow MIPS = ClockRate/(CPI * 10^6)$
- Using MIPS rating is tied to a particular program running on specific machine. MIPS rating is not a reliable measure that could be used to compare two different systems.
- **DMIPS** Dhrystone MIPS, on the other hand, is a useful measure
- Dhrystone is a benchmark program developed by Reinhold P. Weicker to measure a CPU's integer processing performance. Dhrystone could be obtained here `http://www.netlib.org/benchmark/dhry-c`

MFLOPS - Million Floating Point Operations Per Second

- **MFLOPS** A theoretical maximum floating point instruction execution rate.
- MFLOPS are measured using benchmark programs such as:
    - Whetstone : A floating point benchmark software
    - LINPACK : A software library for linear algebra written in fortran
    - LAPACK : Latest version of LINPACK

Speedup

- **Speedup** is defined by: $OriginalCPU_{time}/NewCPU_{time}$
- Overall improvement of a system's performance could be achieved by improving a subset of the system.
- Most of the performance increase come from improving the common case(s)

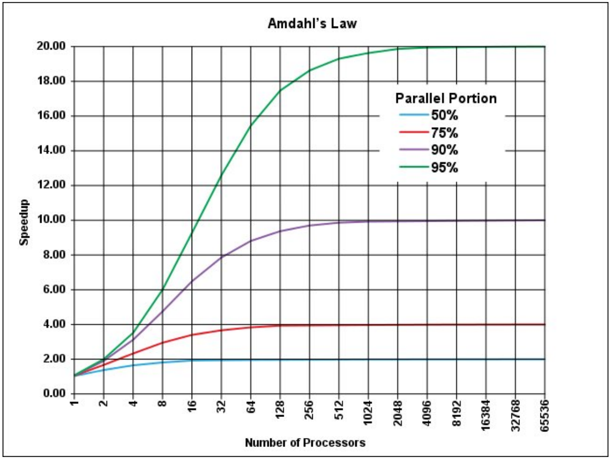
Amdahl's Law

- **Amdahl's Law** is used to calculate the overall performance improvement achieved by improving a subset of a system.
- Amdahl's Law states that performance improvement to be gained by using a faster mode is limited by the fraction of time the faster mode can be used.
- Speedup = Total performance with faster mode / Total performance without faster mode
- Speedup = Execution time without faster mode / Execution time with faster mode

Amdahl's Law

- $Speedup_{overall} = CPU_{Oldtime}/CPU_{Newtime}$
- $CPU_{Newtime} =$
  $UnenhancedCPU_{time} + EnhancedCPU_{time}$
  $\Rightarrow CPU_{Oldtime} * (1 - F_{enh})$
  $+(CPU_{Oldtime} * F_{enh})/Speedup_{enh}$
  $\Rightarrow CPU_{Oldtime} * ((1 - F_{enh}) + F_{enh}/Speedup_{enh})$
- $\Rightarrow Speedup_{overall} = 1/((1 - F_{enh}) + F_{enh}/Speedup_{enh})$

Implications of Amdahl's Law



Dependability

- **Dependability** Today, computers are integral part of our society. It is important to understand the reliability and availability of computers.
- Metrics for dependability:
  1. **MTTF**: Mean time to failure is a reliability measure
  2. **MTTR**: Mean time to repair is a service interruption measure
  3. **MTBF**: Mean time between failure is a commonly used term which is $= MTTR + MTTF$
  4. Module availability $= MTTF/(MTTF + MTTR)$

Week 11 Activity 1

Read Section 1.7-Dependability in the course text book and answer the following questions:

1. What is the difference between SLA, SLO, MTTF, MTBF and MTTR? Describe with examples how these metrics are used in real life. (3 pages max)
2. You are being asked to build a 15TB storage vault with SAS drives + spares to replace ageing SCSI storage vault in your organization. The storage vault you build/buy should last for at least 8 years, performance is crucial, downtime is very expensive and yearly maintenance cost should be minimum.
   1. Which metric(s) would you use to discriminate drives and drive models made by different manufacturers? and how?
   2. Pick a SAS drive that would be suitable for the storage vault and support your decision with analysis.
   3. Calculate the optimal number of spare drives?
   4. Which metric(s) would you use to estimate yearly maintenance cost?

Week 11 Activity 2

Describe and explain the purpose of the following benchmark standards and software:

1. BAPCo
2. SPEC
3. TPC
4. Coremark
5. Dhrystone
6. Whetstone

Week 11 Activity 3

Consider the following profiles of two application that are running on $CPU - ZX$:

APP_1 Profile

t	Instruction	I_t	CPI_t
1	load	35	4
2	store	15	2
3	mul	2	35
4	jump	25	10
5	add	35	5
6	sub	15	5

APP_2 Profile

t	Instruction	I_t	CPI_t
1	load	10	4
2	store	20	2
3	mul	4	35
4	jump	25	10
5	add	25	5
6	sub	10	5

Answer the following questions:

1. Calculate the CPI, performance and relative performance of APP_1 & APP_2?
2. Select three instruction in $CPU - ZX$'s ISA that when improved would provide the most performance increase. Support your answer with thorough analysis.
3. Assume that you were able to make CPI_t of the three instruction that you picked to 3. Calculate overall speedup of APP_1 and APP_2.