

BROWN

# Computer System Design

## Lecture 3: Amdahl's Law

Prof. R. Iris Bahar  
EN164  
January 31, 2007

Reading: H&P Chap 1, section 1.6-1.10

BROWN

# "Iron Law" of Processor Performance

$$\text{Processor Performance} = \frac{\text{Wall-Clock Time}}{\text{Program}}$$

$$= \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Cycles}}{\text{Instruction}} \times \frac{\text{Time}}{\text{Cycle}}$$

(code size) (CPI) (cycle time)

Architecture → Implementation → Realization  
 Compiler Designer Processor Designer Chip Designer

EN164  
Lecture 3-3

BROWN

# 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 10 ns and a CPI of 2.0. Machine B has a clock cycle time of 20 ns and a CPI of 1.2.
  - Which machine is faster for this program, and by how much?
- Time per instruction:
  - for A:  $2.0 * 10 \text{ ns} = 20 \text{ ns}$ , for B:  $1.2 * 20 \text{ ns} = 24 \text{ ns}$
  - ➔ A is  $24/20 = 1.2$  times faster
- If two machines have the same ISA, which of our quantities (e.g., clock rate, CPI, execution time, # of instructions) will always be identical?

EN164  
Lecture 3-5

BROWN

# # of Instructions Example

- A compiler designer has 2 alternatives for a certain code sequence. There are 3 different classes of instructions: A, B, and C, that require 1, 2, and 3 cycles, respectively.
  - The 1<sup>st</sup> sequence has 5 instructions: 2 of A, 1 of B, and 2 of C.
  - The 2<sup>nd</sup> sequence has 6 instructions: 4 of A, 1 of B, and 1 of C.
  - Which sequence will be faster? What are the CPI values?
- Sequence 1:  $2*1+1*2+2*3 = 10$  cycles;  $\text{CPI}_1 = 10 / 5 = 2$
- Sequence 2:  $4*1+1*2+1*3 = 9$  cycles;  $\text{CPI}_2 = 9 / 6 = 1.5$
- Sequence 2 is faster.

EN164  
Lecture 3-7

BROWN

# MIPS

- Million Instructions Per Second
- $\text{MIPS} = \text{instruction count} / (\text{execution time} * 10^6)$
- Depends on
  - clock frequency
  - cycles/instruction (may vary even on a single machine)
- MIPS is easy to understand BUT...
  - does not take into account the capabilities of the instructions; the instruction counts of different instruction sets differ
  - varies between programs even on the same computer
  - can vary inversely with performance!

EN164  
Lecture 3-8

BROWN

# MIPS example

- Two compilers are being tested for a 2 GHz machine with three different classes of instructions: A, B, and C, that require one, two, and three cycles, respectively.
  - Compiler 1: Compiled code uses 5 million Class A, 1 million Class B, and 1 million Class C instructions.
  - Compiler 2: Compiled code uses 10 million Class A, 1 million Class B, and 1 million Class C instructions.
- Which sequence will be faster according to MIPS?
- Which sequence will be faster according to execution time?

EN164  
Lecture 3-9

BROWN

## Evolution of Computing

- Now we have a means of measuring performance, but we also need to consider the target applications.
- Today, there are three major microprocessor market segments: desktop, servers, embedded
  - Different applications
  - Different requirements
  - Different architectures
- Some machines may be better suited for certain applications, but not for others.

EN164  
Lecture 3-10

BROWN

## Characteristics of each Segment

- Desktop: Need high performance and low price
- Servers: Need high throughput, availability (revenue losses of \$14K-\$6.5M per hour of downtime), scalability
- Embedded: Need low power, low cost, low memory (performance is secondary, except real-time constraints)

Feature	Desktop	Server	Embedded
System price	\$500 - \$5,000	\$5,000 - \$5,000,000	\$10 - \$100,000
Processor price	\$50 - \$500	\$200 - \$10,000	\$0.01 - \$100
Design issues	Price vs. performance, graphics	Throughput, availability, scalability	Price, power, app-specific performance

EN164  
Lecture 3-11

BROWN

## 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
- A standard set of “real” programs and inputs allows for a more consistent means of comparison
  - valuable indicator of performance (and compiler technology)
  - can still be abused

EN164  
Lecture 3-12

BROWN

## Benchmarks for each Segment

- Standard Performance Evaluation Corporation (SPEC)
  - CPU2000 (integer and floating-point applications for desktops and servers) stresses CPU and memory, little I/O
  - SPECviewperf and SPECcapc for graphics applications
  - SPECsfs for file systems,
  - SPECweb for web servers (to stress disk and network I/O)
- Transaction Processing Council suites measure throughputs for database transactions
- Five EEMBC benchmark classes for embedded apps:
  - automotive/industrial, consumer, networking, office
  - automation, telecommunications

EN164  
Lecture 3-13

BROWN

## Amdahl's Law

- Architecture design is very bottleneck-driven
  - make the common case fast
  - do not waste resources on a component that has little impact on overall performance/power
- Amdahl's Law:
  - performance improvements through an enhancement is limited by the fraction of time the enhancement comes into play

Execution Time After Improvement =  $\frac{\text{Execution Time Unaffected} + \text{Execution Time Affected}}{\text{Amount of Improvement}}$

Speedup =  $\frac{\text{Performance after improvement}}{\text{Performance before improvement}}$

EN164  
Lecture 3-14

BROWN

## Amdahl's Law

- Before:  $n + a$
- After:  $n + \frac{a}{p}$
- Execution time:  $\left. \begin{array}{l} \text{before } n + a \\ \text{after } n + \frac{a}{p} \end{array} \right\} su = \frac{n+a}{n + \frac{a}{p}}$
- Principle: **Make the common case fast**

EN164  
Lecture 3-15

BROWN

## Amdahl's Law

- 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 the multiplication if we want the program to run 4 times faster?
- $100 / 4 = 80/n + 20$   
 $5 = 80/n$   
 $n = 16$  → the multiply needs to be 16 times faster!

EN164  
Lecture 3-17

BROWN

## Principle of Locality

- Most programs are predictable in terms of instructions executed and data accessed
- The 90-10 Rule: a program spends 90% of its execution time in only 10% of the code
  - Temporal locality: a program will shortly re-visit X
  - Spatial locality: a program will shortly visit X+1
- We will revisit predictability and locality later
  - Designing memory hierarchy
  - Speculating instruction outcome

EN164  
Lecture 3-20

BROWN

## Exploiting Parallelism

- Most operations do not depend on each other  
 → Execute them in parallel
- At the *circuit level*, simultaneously access multiple ways of a set-associative cache
- At the *organization level*, execute multiple instructions at the same time
- At the *system level*, execute a different program while one is waiting on I/O

EN164  
Lecture 3-21

BROWN

## Factors Determining Cost

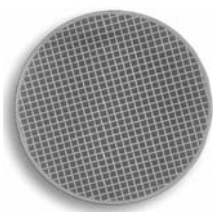
- Cost: amount spent by manufacturer to produce a finished good
- High volume leads to lower cost:
  - faster learning curve,
  - increased manufacturing efficiency,
  - lower R&D cost per produced item
- Commodities: identical products sold by many vendors in large volumes (keyboards, DRAMs)
  - low cost because of high volume and competition among suppliers

EN164  
Lecture 3-22

BROWN

## Wafers and Dies

An entire wafer is produced and chopped into dies that undergo testing and packaging



© 2003 Elsevier Science (USA). All rights reserved.


EN164  
Lecture 3-23

BROWN

## Integrated Circuit Cost

- Cost of an integrated circuit = (cost of die + cost of packaging and testing) / final test yield
- Cost of die = cost of wafer / (dies per wafer x die yield)
- Dies/wafer = wafer area / die area -  $2\pi$  wafer radius / die diagonal *dies along edge*
- Die yield = wafer yield x  $(1 + (\text{defect rate} \times \text{die area}) / \alpha)^{-\alpha}$
- Thus, die yield depends on die area and complexity arising from multiple manufacturing steps ( $\alpha \sim 4.0$ )

EN164  
Lecture 3-24


 BROWN

## Desktop Prices

Vendor	Model	Processor	Clock speed MHz	Price
Dell	Precision 380	Intel Pentium 4 Xeon	3.8GHz	\$3346
HP	ProLiant BL25p	AMD Opteron 252	2.6GHz	\$3099
HP	ProLiant ML350 G4	Intel Pentium 4 Xeon	3.4GHz	\$2907
HP	Integrity rx2620-2	Itanium 2	1.6GHz	\$5201
Sun	Java Workstation W1100z	AMD Opteron 150	2.4GHz	\$2145

- All systems have similar configurations
  - Price variations due to expandability, disks/memory/processor/OS configuration, and commoditization
- On average, the processor is less than 10% of the total cost of the system
- Around 70% of the cost is due to the disk storage

EN164  
Lecture 3-29

 BROWN

## Server Prices

System	CPUs	Price
IBM eSeries p5 595	64 IBM POWER 5	\$16.7 M
IBM eSeries p5 595	32 IBM POWER 5	\$8.4 M
HP Integrity rx5670 Cluster	64 Intel Itanium 2	\$6.5 M
Dell PowerEdge 2800	1 Intel Xeon	\$39 K
HP ProLiant ML350	1 Intel Xeon	\$28 K

- Generally cost much more than desktop systems
- Still, the processor accounts for less than 25% of the system, on average
- Most of the cost is still in the disk storage (~50%)

EN164  
Lecture 3-32