Lecture 10: Performance

(CPEG323: Intro. to Computer System Engineering)

How do we evaluate computer architectures?

- Think of 5 characteristics that differentiate computers

How do we evaluate computer architectures?

- **Think of 5 characteristics that differentiate computers**
  - Memory size
  - Disk size
  - Cost
  - Power consumption
  - Heat dissipation
  - Battery
  - Look
  - **Performance**
  - …

What is Performance?

- ***Latency***
  - Time to complete one task
- ***Throughput***
  - Number of tasks completed per unit time

Why Latency Matters?

Server Delay (ms)	Increased time to next click (ms)	Queries/ user	Any clicks/ user	User satisfac-tion	Revenue/ User
50	--	--	--	--	--
200	500	--	-0.3%	-0.4%	--
500	1200	--	-1.0%	-0.9%	-1.2%
1000	1900	-0.7%	-1.9%	-1.6%	-2.8%
2000	3100	-1.8%	-4.4%	-3.8%	-4.3%

Figure 6.10 Negative impact of delays at Bing search server on user behavior [Brutlag and Schurman 2009].

Longer the delay
→the fewer the user clicks
→the lower the revenue per user

Two Notions of "Performance"

Plane	DC to Paris	Top Speed	Passen-gers	Throughput (pmph)
Boeing 747	6.5 hours	610 mph	470	286,700
BAD/Sud Concorde	3 hours	1350 mph	132	178,200

- Which has higher performance?
- From a passenger's viewpoint: latency (time to do the task)
  - Hours per flight, execution time, response time
- From an airline's viewpoint: throughput (tasks per unit time)
  - Passengers per hour, bandwidth
- Latency and throughput are often in opposition

Some Definitions

- Relative performance: "x is N times faster than y"

$$\frac{\text{Performance(x)}}{\text{Performance(y)}} = N$$

- If we are primarily concerned with latency,
  Performance(x) = $\dfrac{1}{\text{Latency(x)}}$

- If we are primarily concerned with throughput,
  Performance(x) = throughput(x)

7

CPU Performance

- The obvious metric: how long does it take to run a test program? This depends upon three factors:

1. The number of instructions N in the program
   - Obviously, time increases as N increases
2. The kind of instructions in the program
   - Some instructions take more CPU cycles than others
   - Let c be the *average* number of cycles per instruction (CPI)
3. The time t per CPU clock cycle (clock-cycle time)

CPU time = Instructions executed × CPI × Clock cycle time

$$\frac{\text{Seconds}}{\text{Program}} = \frac{\text{Instructions}}{\text{Program}} \times \frac{\text{Clock cycles}}{\text{Instructions}} \times \frac{\text{Seconds}}{\text{Clock cycle}}$$

8

The three components of CPU performance

CPU $\text{time}_{X,P}$ = Instruction count_P * $\text{CPI}_{X,P}$ * Clock cycle time_X

- **Instructions executed:**
  - the dynamic instruction count (#instructions actually executed)
  - not the (static) number of lines of code
- **Average Cycles per instruction:**
  - function of the machine and program
    - CPI(floating-point operations) > CPI(integer operations)
    - Improved processor may execute same instructions in fewer cycles
  - Single-cycle machine: each instruction takes 1 cycle (CPI = 1)
    - CPI can be > 1 due to memory stalls and slow instructions
    - CPI can be < 1 on superscalar machines
- **Clock cycle time:** 1 cycle = minimum time it takes the CPU to do any work
  - clock cycle time = 1/ clock frequency
  - 500MHz processor has a cycle time of 2ns (nanoseconds)
  - 2GHz (2000MHz) CPU has a cycle time of just 0.5ns
  - higher frequency is usually better

Question

- Computer A clock cycle time 250 ps, CPI_A = 2
- Computer B clock cycle time 500 ps, CPI_B = 1.2
- Assume A and B have same instruction set
- Which statement is true?

A. Computer A is ~1.2 times faster than B
B. Computer A is ~4.0 times faster than B
C. Computer B is ~1.7 times faster than A
D. Computer B is ~3.4 times faster than A
E. None of the above

10

Answer

- Computer A clock cycle time 250 ps, CPI_A = 2
- Computer B clock cycle time 500 ps, CPI_B = 1.2
- Assume A and B have same instruction set
- Which statement is true?

A. Computer A is ~1.2 times faster than B
B. Computer A is ~4.0 times faster than B
C. Computer B is ~1.7 times faster than A
D. Computer B is ~3.4 times faster than A
E. None of the above

11

Example: ISA-compatible processors

- Let's compare the performances two x86-based processors
  - An 800MHz AMD Duron, with a CPI of 1.2 for an MP3 compressor
  - A 1GHz Pentium III with a CPI of 1.5 for the same program
- Compatible processors implement identical instruction sets and will use the same executable files, with the same number of instructions
- But they implement the ISA differently, which leads to different CPIs

CPU $\text{time}_{AMD,P}$ = $\text{Instructions}_P \times \text{CPI}_{AMD,P} \times \text{Cycle time}_{AMD}$
=

CPU $\text{time}_{P3,P}$ = $\text{Instructions}_P \times \text{CPI}_{P3,P} \times \text{Cycle time}_{P3}$
=

13

Another Example: Comparing across ISAs

- Intel's Itanium (IA-64) ISA is designed facilitate executing multiple instructions per cycle. If it achieves an average of 3 instructions per cycle, how much faster is it than a Pentium4 (which uses the x86 ISA) with an average CPI of 1?

  a) Itanium is three times faster
  b) Itanium is one third as fast
  c) Not enough information

14

I. Clock cycle time

- One "cycle" is the minimum time it takes the CPU to do any work.
  - The clock cycle time or clock period is just the length of a cycle.
  - The clock rate, or frequency, is the reciprocal of the cycle time.
- Generally, a higher frequency is better.
- Some examples illustrate some typical frequencies.
  - A 500MHz processor has a cycle time of 2ns.
  - A 2GHz (2000MHz) CPU has a cycle time of just 0.5ns (500ps).

II. Instruction Count

- Instructions count:
  - We are not interested in the static instruction count, or how many lines of code are in a program.
  - Instead we care about the dynamic instruction count, or how many instructions are actually executed when the program runs.
- There are three lines of code below, but the number of instructions executed would be 2001.

```
         li   $a0, 1000
Ostrich: sub  $a0, $a0, 1
         bne  $a0, $0, Ostrich
```

Loop Unrolling

- Discussion section notes

17

III. CPI

- The average number of clock cycles per instruction, or CPI, is a function of the machine <u>and</u> program.
  - The CPI depends on the actual instructions appearing in the program—a floating-point intensive application might have a higher CPI than an integer-based program.
  - It also depends on the CPU implementation. For example, a Pentium can execute the same instructions as an older 80486, but faster.

Improving CPI

- Some processor design techniques improve CPI
  - Often they only improve CPI for certain types of instructions

$$CPI = \sum_{i=1}^{n} CPI_i \times F_i$$

where F_i = fraction of instructions of type i

19

Example: CPI improvements

- Base Machine:

Op Type	Freq (F_i)	CPI_i	contribution to CPI
ALU	50%	3	
Load	20%	6	
Store	20%	3	
Branch	10%	2	

- How much faster would the machine be if:
  - we added a cache to reduce average load time to 3 cycles?
  - we added a branch predictor to reduce branch time by 1 cycle?
  - we could do two ALU operations in parallel?

20

Example: CPI improvements

- Base Machine:

Op Type	Freq (F_i)	CPI_i	contribution to CPI
ALU	50%	3	0.5*3
Load	20%	6	0.2*6
Store	20%	3	0.2*3
Branch	10%	2	0.1*2

- How much faster would the machine be if:
  - we added a cache to reduce average load time to 3 cycles?
  - we added a branch predictor to reduce branch time by 1 cycle?
  - we could do two ALU operations in parallel?

21

Amdahl's Law

- Amdahl's Law states that optimizations are limited in their effectiveness

$$\text{Execution time after improvement} = \frac{\text{Time affected by improvement}}{\text{Amount of improvement}} + \text{Time unaffected by improvement}$$

- *Example*: Suppose we double the speed of floating-point operations
  - If only 10% of the program execution time T involves floating-point code, then the overall performance improves by just 5%

$$\text{Execution time after improvement} = \frac{0.10\ T}{2} + 0.90\ T = 0.95\ T$$

Amdahl's Law (Cont.)

- Suppose that the enhancement E accelerates a fraction F(F<1) of the task by a factor S (S>1) and the remainder of the task is unaffected.

$$\text{speedup} = \frac{1}{(1-F) + \frac{F}{S}}$$

23

Amdahl's Law

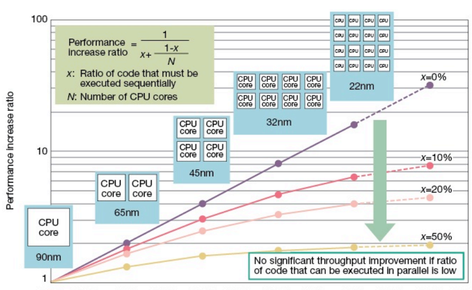

Fig 3 Amdahl's Law an Obstacle to Improved Performance Performance will not rise in the same proportion as the increase in CPU cores. Performance gains are limited by the ratio of software processing that must be executed sequentially. Amdahl's Law is a major obstacle in boosting multicore microprocessor performance. Diagram assumes no overhead in parallel processing. Years shown for design rules based on Intel planned and actual technology. Core count assumed to double for each rule generation.

24

Reading

- 5th Edition: 1.4

25