CIS501: Performance & Benchmarking

Shreyas S. Shivakumar

December 2, 2019

1 Questions

1. **What is the difference between *throughput* and *latency*?**

   Latency is the time required to finish a given task, whereas **Throughput** is the number of tasks that can be finished per unit time.

2. **Which is easier to improve and how?**

   Throughput is easier to improve, and this can be done by introducing **parallelism**.

3. **What is CPI?**

   Cycles per instruction. For *example*, in the single cycle LC4 implementation, the **CPI** is **1**.

4. **Which is a faster processor - Processor A (5 GHz, 2 CPI) or Processor B (3 GHz, 1 CPI)?**

   Processor B is actually faster. B will have a lower latency than A. Most marketing material will ignore these dynamic instruction counts and report clock rate only.

5. **What are the basic CPU performance equations?**

$$Latency = \frac{seconds}{insn} = \frac{cycles}{insn} \times \frac{seconds}{cycle} \quad (1)$$

$$Throughput = \frac{insn}{second} = \frac{insn}{cycle} \times \frac{cycle}{second} \quad (2)$$

6. **What is another way of looking at Latency?**

$$Latency = \frac{seconds}{program} = \frac{insn}{program} \times \frac{cycle}{insn} \times \frac{seconds}{cycle} \quad (3)$$

7. **How is throughput usually measured?**

   Throughput is usually measured as **MIPS** - millions of instructions per second. For *example*, if you have CPI = 2 and a clock speed of 500MHz, you can use the above equation for throughput to solve the following.

   $Throughput = \frac{1}{2} \times 500 = 250$ MIPS

8. **What is the CPI of a program that executes an equal number of Integer, Floating Point and Memory operations, where the CPI of each type is 1, 2 and 3 respectively?**

   $CPI = 0.33 * 1 + 0.33 * 2 + 0.33 * 3 = 2$ CPI

9. **What is the CPI of a program that has Integer ALU (50%, 1 cycle), Load (20%, 5 cycle), Store (10%, 1 cycle) and Branch (20%, 2 cycle)?**

   $CPI_{base} = 0.5 * 1 + 0.2 * 5 + 0.1 * 1 + 0.2 * 2 = 2$ CPI

10. **In the above question, which would improve performance more? (A) Branch prediction to reduce branch cost to 1 cycle or (B) faster data memory to reduce load cost to 3 cycles?**

    Option (B) will result in better performance because:

    $CPI_A = 0.5 * 1 + 0.2 * 5 + 0.1 * 1 + 0.2 * 1 = 1.8$ CPI

$CPI_B = 0.5 * 1 + 0.2 * 3 + 0.1 * 1 + 0.2 * 2 = 1.6$ CPI

11. If a program C is 1x faster than A, how many cycles does C run for if A runs for 200 cycles?

The same number of cycles as A. This is just to illustrate some of the nuances of understanding speedups (1x, 2x, 2.3x) versus percentages (0% increase, 100% increase, etc.)

12. What if program C is 1.5x faster than A?

C will then run for $\frac{200}{1.5} = 133$ cycles

13. How do you average Latency and Throughput?

You can find mean *Latency* by using a simple **Arithmetic Mean**. But since *Throughput* is inversely proportional to time, you must use a **Harmonic Mean** instead.

Example: the average of 1 mile at 30 mph and 1 mile at 90 mph is not 60 mph. Instead, you can do the following:

For the first mile: $\frac{1}{30} = 0.0333$ hours per mile

For the second mile: $\frac{1}{90} = 0.0111$ hours per mile

These can then be averaged to get $\frac{0.0333+0.0111}{2} = 0.0222$ hours per mile, which can then be converted to *mph* to get 45 miles per hour.

14. What are systematic errors?

This is measurement bias. You must make sure you are measuring the system when it is in a steady state. Vary experimental setup and identify appropriate statistics.

15. What are random errors?

Perform the experiment many times and measure the statistics. Distributions of errors matter more than individual absolute values.

16. What is Amdahl's Law?

It helps analyze, in advance, how much of a performance boost you may receive by optimizing a specific part of your algorithm.

$$Speedup = \frac{1}{(1-P)+\frac{P}{S}} \tag{4}$$

P = proportion of running time affected by the optimization

S = speedup

17. What if 25% of a program is speedup by 2x?

$Speedup = \frac{1}{(1-0.25)+\frac{0.25}{2}} = \frac{1}{0.75+0.125} = \frac{1}{0.825} = 1.14\text{x}$

18. What if 25% of a program is speedup by ∞?

$Speedup = \frac{1}{(1-0.25)+\frac{0.25}{\infty}} = \frac{1}{0.75} = 1.33\text{x}$

19. What is Amdahl's Law for Parallelization?

It helps analyze, in advance, how much of a performance boost you may receive by adding parallelism to a specific part of your algorithm.

$$Speedup = \frac{1}{(1-P)+\frac{P}{N}} \tag{5}$$

P = proportion of parallel code

S = number of threads

20. What is the maximum speedup for a program that is 10% serial?

$Speedup = \frac{1}{(1-0.9)+\frac{0.9}{X}} = \frac{1}{0.1+\frac{0.9}{X}}$

If X is made very large such that $\frac{0.9}{X}$ is approximately 0, then:

$Speedup = \frac{1}{(1-0.9)+\frac{0.9}{X}} = \frac{1}{0.1+\frac{0.9}{X}} = \frac{1}{0.1} = 10\text{x}$

21. What is the maximum speedup for a program that is 1% serial?

$Speedup = \frac{1}{(1-0.99)+\frac{0.99}{X}} = \frac{1}{0.01+\frac{0.99}{X}}$

If X is made very large such that $\frac{0.99}{X}$ is approximately 0, then:

$Speedup = \frac{1}{(1-0.99)+\frac{0.99}{X}} = \frac{1}{0.01+\frac{0.99}{X}} = \frac{1}{0.01} = 100\text{x}$

22. What is Little's Law?

Little's Law helps make decisions in queuing systems, with the assumption that *arrival rate = departure rate.*

$$L = \lambda \times W \tag{6}$$

L = items in the system

λ = average arrival rate

W = average wait time

23. What is the intuition behind Little's Law for throughput?

To get high throughput (λ), we need to either reduce latency per request (W) or service more requests in parallel, i.e increase L.

2 Textbook Notes

1. **Throughput & Response Time (Latency):**

   By replacing the processor in a computer with a faster version, you will most likely decrease response time and almost always improve throughput. By adding more processors to a system that is using multiple processors for different/separate tasks, you will most likely not improve response time (latency), but improve throughput only. But if the demand for processing was as large as the throughput, this could improve both latency and throughput.

2. **Performance & Latency:**

$$Performance_X = \frac{1}{Execution\ Time_X} \tag{7}$$

   "X is n times faster than Y" is equivalent to:

$$\frac{Performance_X}{Performance_Y} = n \tag{8}$$

3. **Elapsed Time vs CPU Time:**

   A processor may work on several programs simultaneously. Here, a system might try to optimize throughput rather than attempt to minimize elapsed time for one specific program. CPU time recognizes this distinction and it is the time the CPU spends computing for a specific task and does not include time spent waiting for I/O or running other programs. CPU time is further divided into *user CPU time* and *system CPU time*.

4. **CPU Performance Factors:**

$$\text{CPU execution time} = \text{CPU clock cycles} \times \text{clock cycle time} \tag{9}$$

   And since *clock cycle time* $= \frac{1}{clock\ rate}$

$$\text{CPU execution time} = \text{CPU clock cycles} \times \frac{1}{\text{clock rate}} \tag{10}$$

5. **Problem 1:** Our favorite program runs in 10 seconds on computer A, which has a 2 GHz clock. We are trying to help a computer designer build a computer, B, which will run this program in 6 seconds. The designer has determined that a substantial increase in the clock rate is possible, but this increases will affect the rest of the CPU design, causing computer B to require 1.2 times as many clock cycles as computer A for this program. What clock rate should we tell the designer to target?

$$CPU\ Time_A = \frac{CPU\ clock\ cycles_A}{Clock\ rate_A}$$

$$CPU\ clock\ cycles_A = Clock\ rate_A \times CPU\ Time_A$$

$$CPU\ clock\ cycles_A = 2\ GHz \times 10\ seconds = 2 \times 10^9 \times 10 = 20 \times 10^9$$

And from the problem, we are told that computer B will require 1.2 times as many clock cycles as computer A. We know that $CPU\ Time_B = 6$ seconds.

$$Clock\ rate_B = \frac{CPU\ clock\ cycles_B}{CPU\ Time_B}$$

$$Clock\ rate_B = \frac{1.2 \times CPU\ clock\ cycles_A}{CPU\ Time_B} = \frac{1.2 \times 20 \times 10^9}{6}$$

$$Clock\ rate_B = 4 \times 10^9\ cycles\ per\ second = 4\ GHz$$

6. **Instruction Performance:**

$$CPU\ clock\ cycles = \#Instructions \times Avg\ CPI$$

7. **Problem 2:** Suppose we have two implementations of the same instruction set architecture. Computer A has a clock cycle time of 250ps and a CPI of 2.0 for some program, and computer B has a clock cycle time of 500ps and a CPI of 1.2 for the same program. Which computer is faster for this program and by how much?

Let us assume that this program has n instructions.

$$CPU\ clock\ cycles_A = n \times 2.0$$
$$CPU\ clock\ cycles_B = n \times 1.2$$

The CPU Time can then be calculated as follows:

$$CPU\ time_A = CPU\ clock\ cycles_A \times Clock\ cycle\ time_A = n \times 2.0 \times 250\ ps$$

$$CPU\ time_B = CPU\ clock\ cycles_B \times Clock\ cycle\ time_B = n \times 1.2 \times 500\ ps$$

The speedup is therefore:

$$Speedup = \frac{CPU\ time_B}{CPU\ time_A} = \frac{n \times 1.2 \times 500\ ps}{n \times 2.0 \times 250\ ps} = 1.2\times$$

8. **Classic CPU Performance Equation**

$$CPU\ Time = Insn\ Count \times CPI \times Clock\ cycle\ time \tag{11}$$

$$CPU\ Time = \frac{Insn\ Count \times CPI}{Clock\ cycle\ rate} \tag{12}$$

3 Midterm Exams

3.1 Spring 2015 : CIS501

1. **Assume a typical program has the following instruction type breakdown: 30% loads, 15% stores, 50% adds, 5% multiplies. Assume the current generation processors have the following instruction latencies: 2 cycles, 4 cycles, 1 cycle and 15 cycles respectively. If for the next generation design you could pick one type of instruction to make twice as fast (half the latency), which type would you pick?**

$$CPI_{current} = 0.30 * 2 + 0.15 * 4 + 0.5 * 1 + 0.05 * 15$$
$$CPI_{current} = 0.60 + 0.60 + 0.5 + 0.75 = 2.45$$

   Each of these halved would be 0.30, 0.30, 0.25 and 0.375 respectively. Therefore it would be best to pick the **multiply** instructions for this optimization. The resulting CPI would then be:

$$CPI_{updated} = 0.60 + 0.60 + 0.5 + 0.375 = 2.075$$

2. **You are deciding between two optimizations for a piece of software. Optimization A will affect half of the program's execution and accelerate that by 10%. Optimization B will affect one third of the program's execution and accelerate that by 40%. Which optimization will improve the program's running time more?**

$$Speedup_A = \frac{1}{(1-0.5)+\frac{0.5}{1.1}} = \frac{1}{0.5+0.4545} = \frac{1}{0.9545}$$
$$Speedup_B = \frac{1}{(1-0.33)+\frac{0.33}{1.4}} = \frac{1}{0.66+0.2354} = \frac{1}{0.8954}$$

   Therefore, $Speedup_B > Speedup_A$

3.2 Fall 2015 : CIS501

1. **Assume a typical program has the following instruction type breakdown: 25% loads, 10% stores, 60% adds, 5% multiplies. Assume the current generation processors have the following instruction latencies: 3 cycles, 5 cycles, 1 cycle and 12 cycles respectively. If for the next generation design you could pick one type of instruction to make twice as fast (half the latency), which type would you pick?**

$$CPI_{current} = 0.25 * 3 + 0.1 * 5 + 0.6 * 1 + 0.05 * 12$$

$$CPI_{current} = 0.75 + 0.5 + 0.6 + 0.6 = 2.45$$

   Each of these halved would be 0.375, 0.25, 0.3 and 0.3 respectively. Therefore it would be best to pick the **load** instructions for this optimization. The resulting CPI would then be:

$$CPI_{updated} = 0.375 + 0.5 + 0.6 + 0.6 = 2.075$$

2. **Profiling of a simulator reveals that it spends $\frac{1}{3}$ of it's time parsing the trace file. With an infinitely-fast parser, how much of a speedup would this yield for the overall simulation?**

   We can use *Amdahl's Law* to find out how much of a speedup this will yield.

$$Speedup = \frac{1}{(1-P) + \frac{P}{S}}$$

$$Speedup = \frac{1}{(1-0.33) + \frac{0.33}{\infty}} = \frac{1}{0.66} = 1.515\times$$

3.3 Spring 2016: CIS501

1. **What is easier to improve: bandwidth or latency?**

   Bandwidth is easier to improve.

2. **What performance law ($L = \lambda W$) describes the behaviour of a queuing system?**

   Little's Law

3.4 Spring 2016 : CIS371

1. **Assume a typical program has the following instruction type breakdown: 10% loads, 15% stores, 15% branches, 55% adds, 5% divides. Assume the current generation processors have the following instruction latencies: 7 cycles, 10 cycles, 4 cycle, 4 cycle and 5 cycles respectively.**

$$CPI_{current} = 0.10 * 7 + 0.15 * 10 + 0.15 * 4 + 0.55 * 4 + 0.05 * 5$$
$$CPI_{current} = 0.70 + 1.5 + 0.6 + 2.2 + 0.25 = 5.25$$

(1) Will tripling the load latency and halving the store latency improve this program?

Answer: No.

$$CPI_{CASE-1} = 0.10 * 21 + 0.15 * 5 + 0.15 * 4 + 0.55 * 4 + 0.05 * 5 = 5.9$$

(2) Leave the load latency unchanged but add 50% to the store latency and cut divide latency to just 2 cycles. Will this improve the program?

Answer: No.

$$CPI_{CASE-2} = 0.10 * 7 + 0.15 * 15 + 0.15 * 4 + 0.55 * 4 + 0.05 * 2 = 5.85$$

(3) Or should it just be left as it is?

Answer: Yes.

(4) What is the speedup of the fastest method compared to the other two methods?

Answer: The original method is $\frac{5.9}{5.25} = 1.12\times$ faster than CPI_{CASE-1}. The original method is $\frac{5.85}{5.25} = 1.11\times$ faster than CPI_{CASE-2}

3.5 Spring 2017 : CIS371

1. **Virtually all smart phones sold in the US contain an ARM CPU designed by either Apple or Qualcomm. (These are not the only games in town, but Qualcomm dominates in non-Apple phones, especially in the US, through heavy-handed patent licensing agreements on cellular radio components.) In a constant effort to one-up each other's performance, Qualcomm has come across a dumpster-full of internal Apple documents, including specs for Apple's upcoming A371 processor for the next iPhone. The specs include the breakdown of instructions Apple typically sees in iPhone programs, and the A371's latencies for each instruction:**

   Load - 5% - 3 cycles

   Add - 10% - 5 cycles

   Divide - 10% - 8 cycles

   Branch - 50% - 2 cycles

   Shift Left - 15% - 5 cycles

   Shift Right - 10% - 1 cycle

   Along with this table, the internal specs include some notes on possible improvements that weren't implemented. Qualcomm decides to analyze these and quickly design a processor that mirrors Apple's but incorporates the best of these alternatives.

   (a) Calculate the CPI of Apple's A371 processor:

$$CPI_{A371} = 0.05 * 3 + 0.1 * 5 + 0.1 * 8 + 0.5 * 2 + 0.15 * 5 + 0.1 * 1$$

$$CPI_{A371} = 0.15 + 0.5 + 0.8 + 1 + 0.75 + 0.1 = 3.30$$

   (b) Variant 1: Apple's notes propose a variant in which all add instructions are replaced with corresponding subtract instructions that take the same amount of time. The idea is apparently to dazzle the press with a "less is more" approach. Calculate the CPI of variant 1:

   This will result in the same CPI as above.

$$CPI_{A371-B} = 0.05 * 3 + 0.1 * 5 + 0.1 * 8 + 0.5 * 2 + 0.15 * 5 + 0.1 * 1$$

$$CPI_{A371-B} = 0.15 + 0.5 + 0.8 + 1 + 0.75 + 0.1 = 3.30$$

(c) Variant 2: Alternatively, remove the highest-latency instruction, and replace all those instructions with (on average), three of the lowest latency instructions in the mix. Calculate the CPI of variant 2 (relative to the original instruction count):

$$CPI_{A371-C} = (CPI_{A371} - 0.8) + 3 * 0.1 = 2.80$$
$$CPI_{A371-C} = (3.30 - 0.8) + 3 * 0.1 = 2.80$$

(d) Variant 3: Cut the latency for divides by a factor of four, but increase the latencies of branches by 50%. Calculate the CPI of variant 3:

$$CPI_{A371-B} = 0.05 * 3 + 0.1 * 5 + \frac{0.1 * 8}{4} + (0.5 * 2) * 1.5 + 0.15 * 5 + 0.1 * 1$$

$$CPI_{A371-B} = 3.20$$

3.6 Fall 2017 : CIS501

1. **Assume a typical program has the following instruction type breakdown:**

   25% loads — 20% stores — 50% adds — 5% multiplies

   Assume the current-generation processor has the following instruction latencies:

   loads: 2 cycles — stores: 5 cycles — adds: 1 cycle — multiplies: 25 cycles

   (a) What is the CPI of the current-generation processor on the typical program?

$$CPI_{typical} = 0.25 * 2 + 0.20 * 5 + 0.50 * 1 + 0.05 * 25$$
$$CPI_{typical} = 0.50 + 1.00 + 0.50 + 1.25$$

(b) If for the next-generation processor you could pick one type of instruction to make twice as fast (half the latency), which instruction type would you pick?

Answer: Multiply operations (1.25), this will then reduce the overall CPI to $3.25 - 0.625 = 2.625$

2. **A software developer is trying to optimize the performance of her code. She finds that it spends 33% of its time in a function foo(). After further experimentation, she estimates that this function can be accelerated by 4x with the use of vector insns. What is the overall application speedup that can be achieved if foo() is optimized?**

$$Speedup = \frac{1}{(1-P)+\frac{P}{N}}$$

$$Speedup = \frac{1}{(1-0.33)+\frac{0.33}{4}} = \frac{1}{0.66+0.0825} = 1.346\times$$

3. **A processor architect is trying to measure the average occupancy of a 5-stage pipeline (how many insns are in the pipeline, on average). He measures the average arrival rate of insns to be 667 MHz, and the average time spent in the pipeline to be 6 ns. What is the average occupancy of the pipeline?**

$$L = \lambda \times W$$

$$L = 667MHz \times 6ns = 667 * 10^6 * 6 * 10^{-9}$$

$$L = 4002 * 10^{-3} = 4 \ instructions$$

3.7 Spring 2018 : CIS371

1. **Assume a typical program has the following instruction type breakdown:**

   20% loads — 20% stores — 50% adds — 10% multiplies

   Assume the current-generation processor has the following instruction latencies:

   loads: 2 cycles — stores: 9 cycles — adds: 1 cycle — multiplies: 15 cycles

(a) What is the CPI of the current-generation processor on the typical program?

$$CPI_{typical} = 0.20 * 2 + 0.20 * 9 + 0.50 * 1 + 0.10 * 15$$
$$CPI_{typical} = 0.40 + 1.80 + 0.50 + 1.50 = 4.20$$

(b) If for the next-generation processor you could pick one type of instruction to make twice as fast (half the latency), which instruction type would you pick?

Answer: Stores

2. **Tara Hertz, PhD is trying to optimize the performance of her code. She finds that it spends 25% of its time in a function foo(). After further experimentation, she estimates that this function can be accelerated by 5x with the use of vector insns. What is the overall application speedup that can be achieved if foo() is optimized?**

$$Speedup = \frac{1}{(1-P) + \frac{P}{N}}$$
$$Speedup = \frac{1}{(1-0.25) + \frac{0.25}{5}} = \frac{1}{0.75 + 0.05} = 1.25\times$$

3.8 Spring 2019 : CIS501

1. **What is the maximum IPC obtainable by our single-cycle design (Lab 3)?**

   Answer: 1

2. **Tara Hertz, PhD is trying to optimize the performance of her code. She finds that it spends 30% of its time in a function foo(). After further experimentation, she estimates that this function can be accelerated by 3x with the use of an FPGA. What is the overall application speedup that can be achieved if foo() is optimized?**

$$Speedup = \frac{1}{(1-P) + \frac{P}{N}}$$

$$Speedup = \frac{1}{(1-0.30)+\frac{0.30}{3}} = \frac{1}{0.7+0.10} = 1.25\times$$

3. A web server receives and processes 100 requests per second, and is processing 50 requests at any given time. Assume the system is in a steady state.

$$L = \lambda \times W$$
$$W = \frac{L}{\lambda} = \frac{50\ requests}{100\ requests\ per\ second}$$
$$W = 0.5\ seconds$$

(b) The IT department wants to cut client waiting time in half. It is considering buying more RAM for the web server so that it can hold 100 requests at any given time. Will this meet their performance objective?

Answer: No, this will not help as the system is already in steady state. It is not limited by storage space or insufficient buffer size. The only way to improve this will be to increases the rate at which requests are processed, i.e increase λ.