The CPU Perfomance Equation

 Jyotiprakash Mishra

Jul 18, 2024 · 19 min read

Understanding the Performance Equation for Processors

The performance of a processor is a crucial aspect that determines how effectively a computer can execute tasks. A common way to define performance is through the inverse of execution time:

$$\text{Performance} = \frac{1}{\text{Execution Time}}$$

To delve deeper into what impacts execution time, we can break it down into three key factors: the number of instructions per program, the number of cycles per instruction, and the clock cycle time. Incorporating CPU frequency, which is the number of clock cycles per second, we get a more comprehensive view of processor performance.

The Performance Equation

The detailed performance equation is expressed as:

$$\text{Execution Time} = \left(\frac{\text{Instructions}}{\text{Program}}\right) \times \left(\frac{\text{Cycles}}{\text{Instruction}}\right) \times \left(\frac{\text{Seconds}}{\text{Cycle}}\right)$$

Since the clock cycle time is the inverse of CPU frequency ((f)), we can rewrite this as:

$$\text{Seconds per Cycle} = \frac{1}{f}$$

Thus, the execution time equation becomes:

$$\text{Execution Time} = \frac{\text{Instruction Count} \times \text{CPI}}{f}$$

To express performance, which is the inverse of execution time, we have:

$$\text{Performance} = \frac{f}{\text{Instruction Count} \times \text{CPI}}$$

Key Components of the Performance Equation

1. Instruction Count (Instructions per Program)

- **Definition**: This term refers to the total number of instructions executed by the program.
- **Impact**: A lower instruction count typically means faster execution, assuming other factors remain constant. Optimization involves efficient coding practices, good algorithm design, and effective compiler optimizations.
- **Reduction Methods**: To reduce instruction count, programmers and compilers work together to optimize code, streamline algorithms, and eliminate unnecessary operations.

2. Cycles per Instruction (CPI)

- **Definition**: CPI is the average number of clock cycles each instruction takes to execute.
- **Impact**: Lower CPI indicates higher performance since fewer cycles are needed per instruction. CPI can be influenced by various factors, including the complexity of instructions, the efficiency of the processor's pipeline, and mechanisms like branch prediction and memory access.
- **Reduction Methods**: Architectural enhancements such as pipelining, superscalar execution, out-of-order execution, and improved branch prediction can reduce CPI. However, these improvements can increase design complexity and power consumption.

3. CPU Frequency (Clock Cycles per Second, (f))

- **Definition**: CPU frequency is the number of clock cycles the CPU completes in one second, measured in hertz (Hz).
- **Impact**: Higher CPU frequency generally means faster execution as more cycles occur per second. However, increasing frequency can lead to higher power consumption and thermal challenges, requiring robust cooling solutions and energy-efficient design.
- **Increase Methods**: Advancements in semiconductor technology, architectural optimizations, and efficient cooling solutions allow for higher clock rates. However, there are trade-offs as higher clock rates can result in greater power consumption and heat generation.

Balancing the Factors

Improving processor performance requires a balanced approach to optimizing all three factors. Here's a summary of how each factor interplays:

- **Instruction Count**: Optimizing algorithms and code to reduce the number of instructions can significantly improve performance, but this must be balanced with maintaining program correctness and functionality.
- **Cycles per Instruction (CPI)**: Enhancing processor architecture to execute instructions more efficiently can lower CPI. Techniques such as pipelining, branch prediction, and out-of-order execution are crucial here, although they can increase the complexity and power requirements of the processor.
- **CPU Frequency (f)**: Increasing the clock speed can improve performance, but it introduces challenges related to power consumption and heat dissipation. Efficient cooling and power management are essential to manage these effects.

Problems

1. Processor A executes a program with 1 billion instructions with a CPI of 2 and a clock frequency of 2 GHz. Processor B executes the same program with a CPI of 1.5 and a clock frequency of 1.5 GHz. Compare the execution time and performance of both processors.
2. Processor P has a clock speed of 2.5 GHz and a CPI of 3 for a given program with 200 million instructions. If improvements in the architecture reduce the CPI to 2.5 without changing the clock speed, calculate the reduction in execution time and the percentage improvement in performance.
3. A processor with a CPI of 1.6 executes a program in 0.5 seconds. If the processor runs at 3.6 GHz, calculate the number of instructions executed. Additionally, determine how the execution time would change if the CPI were improved to 1.4 while maintaining the same clock speed.
4. Processor G has a clock speed of 2.2 GHz and executes a program with 600 million instructions in 0.75 seconds. Calculate the CPI of

Processor G. If the instruction count is reduced by 10% without changing the clock speed or CPI, what will be the new execution time?

5. Processor J has an initial CPI of 3 and a clock speed of 2.5 GHz. After optimization, the CPI is reduced to 2.4, but the clock speed is also reduced to 2.2 GHz. If the program executed has 1.3 billion instructions, calculate the initial and optimized execution times and the percentage change in performance.

Why Multi-Core CPUs Do Not Achieve Linear Speedup

Imagine a multi-lane highway with (n) lanes that allows cars to travel in parallel, much like a multi-core CPU allows tasks to be executed in parallel. However, if at certain points on the highway the lanes merge into a single lane, these bottlenecks significantly reduce the overall speed of travel. Similarly, in a multi-core CPU, not all parts of a program can be parallelized. Some portions of the code must still be executed sequentially, creating bottlenecks that prevent achieving a linear speedup proportional to the number of cores.

Amdahl's Law

Amdahl's Law provides a formula to estimate the maximum possible speedup of a program based on the portion that can be parallelized. If (S) is the fraction of the program that must be executed sequentially, and (P) is the fraction that can be parallelized, then:

[S + P = 1]

If we have (n) cores, the parallel part of the program can be executed (n) times faster, but the sequential part remains unchanged. Therefore, the speedup (\text{Speedup}(n)) is given by:

[\text{Speedup}(n) = \frac{1}{S + \frac{P}{n}}]

Derivation of Amdahl's Law

Let:

- T(1) be the execution time with 1 core.
- T(n) be the execution time with n cores.
- Ts be the time spent on the sequential part.
- Tp be the time spent on the parallel part.

With 1 core:

$$T(1) = T_s + T_p$$

With n cores, the parallel part Tp is divided among n cores:

$$T(n) = T_s + \frac{T_p}{n}$$

The speedup is then:

$$\text{Speedup}(n) = \frac{T(1)}{T(n)} = \frac{T_s + T_p}{T_s + \frac{T_p}{n}}$$

Let

$$S = \frac{T_s}{T_s + T_p}$$

(the fraction of the program that is sequential) and

$$P = \frac{T_p}{T_s + T_p}$$

(the fraction that can be parallelized). Since (S + P = 1), we have:

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

Incorporating Amdahl's Law into the Performance Equation

The performance equation can be enhanced by incorporating Amdahl's Law to reflect the limits of parallelization. Let:

- Instruction Count be the total number of instructions.
- CPI be the average cycles per instruction.
- f be the clock frequency in Hz.
- n be the number of cores.
- S be the fraction of the program that is sequential.
- P be the fraction of the program that can be parallelized.

Execution time T(n) for n cores:

$$T(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left(S + \frac{P}{n}\right)$$

Performance for n cores:

$$\text{Performance}(n) = \frac{f}{\text{Instruction Count} \times \text{CPI}} \times \frac{1}{S + \frac{P}{n}}$$

Problems:

6. Processor A has 4 cores with a clock speed of 2.5 GHz and a CPI of 2. Processor B has 1 core with a clock speed of 2 GHz and a CPI of 1.8. For a program with 1 billion instructions where 60% can be parallelized, calculate the execution time and performance for both processors.

7. A program has 2 billion instructions with 70% parallelizable (P) and 30% sequential (S). Processor X has 6 cores with a clock speed of 3 GHz and a CPI of 2.2. Processor Y has 4 cores with a clock speed of 2.8 GHz and a CPI of 1.9. Calculate the execution time and performance for both processors.
8. Processor A has 4 cores with a clock speed of 3 GHz and a CPI of 2. Processor B has 8 cores with a clock speed of 2.5 GHz and a CPI of 2.5. A program consists of 1.5 billion instructions. How much of the program must be parallelized for Processor A to match the performance of Processor B?
9. Processor M has 4 cores with a clock speed of 3.2 GHz and a CPI of 2.3. Processor N has 32 cores with a clock speed of 4 GHz and a CPI of 2.0. If a program is 60% parallelizable, calculate how much the parallelizable portion needs to be increased for Processor M to match Processor N's performance.

Understanding Different Classes of Instructions and Their Impact on Performance

A program typically consists of various classes of instructions such as arithmetic operations, branches, memory accesses, and others. Each class of instructions can have a different CPI (Cycles Per Instruction), which affects the overall execution time and performance of the program.

Instruction Classes and CPIs

Let's consider a program with the following classes of instructions:

- **Arithmetic Instructions**: Perform basic operations like addition, subtraction, multiplication, and division.
- **Branch Instructions**: Control the flow of the program by making decisions (e.g., if-else statements, loops).

- **Memory Access Instructions**: Involve loading data from memory or storing data to memory.

Each of these instruction classes can have a different CPI. For example:

- Arithmetic Instructions: CPI = 1.5
- Branch Instructions: CPI = 2.0
- Memory Access Instructions: CPI = 2.5

The overall execution time and performance of the program depend on the proportion of each class of instructions and their respective CPIs.

Execution Time Formula

To calculate the execution time for a program with different classes of instructions, we use the following formula:

$$T = \sum_i \left(\frac{\text{Instruction Count}_i \times \text{CPI}_i}{f} \right)$$

where i is the instruction class

Performance Formula

The performance of the program can be calculated using the inverse of the execution time:

$$\text{Performance} = \frac{1}{T} = \frac{f}{\sum_i (\text{Instruction Count}_i \times \text{CPI}_i)}$$

Example Calculation

Let's consider a program with the following distribution of instructions:

- Arithmetic Instructions: 500 million

- Branch Instructions: 200 million
- Memory Access Instructions: 300 million

Assume the clock frequency f is 3 GHz.

Execution Time Calculation

For Arithmetic Instructions:

$$T_{\text{arith}} = \frac{500 \times 10^6 \times 1.5}{3 \times 10^9} = 0.25 \, \text{seconds}$$

For Branch Instructions:

$$T_{\text{branch}} = \frac{200 \times 10^6 \times 2.0}{3 \times 10^9} = 0.133 \, \text{seconds}$$

For Memory Access Instructions:

$$T_{\text{mem}} = \frac{300 \times 10^6 \times 2.5}{3 \times 10^9} = 0.25 \, \text{seconds}$$

Total Execution Time:

$$T = T_{\text{arith}} + T_{\text{branch}} + T_{\text{mem}} = 0.25 + 0.133 + 0.25 = 0.633 \, \text{seconds}$$

Performance Calculation

$$\text{Performance} = \frac{1}{T} = \frac{1}{0.633} \approx 1.58 \, \text{programs/second}$$

Problems

10. A processor has a clock speed of 3.5 GHz and a program with 3 billion instructions distributed among three classes: arithmetic, branch, and memory access. The distribution and CPIs for each class are as follows: Arithmetic Instructions: 1.2 billion instructions, CPI = 1.8; Branch Instructions: 0.8 billion instructions, CPI = 2.5; Memory Access Instructions: 1 billion instructions, CPI = 3.0.

Calculate the total execution time for the program and the overall performance in programs per second.

11. Two processors, Processor A and Processor B, have the following specifications: - **Processor A**: 4 cores, 3.5 GHz clock speed
    - **Processor B**: 8 cores, 4.0 GHz clock speed
    The program consists of the following instruction distribution and CPIs for each class:
    - Arithmetic Instructions: 40%, CPI = 1.5
    - Memory Access Instructions: 30%, CPI = 2.0
    - Branch Instructions: 30%, CPI = 2.5
    The program has a parallelizable portion (P) of 70% and a sequential portion (S) of 30%. Calculate how much the percentage of branch instructions needs to be reduced for Processor A to match the performance of Processor B, assuming that decreasing the percentage of branch instructions does not change the values of S and P. Assume that decreasing branches also reduces the total number of instructions in the program.

12. Two processors, Processor A and Processor B, have the following specifications: - **Processor A**: 4 cores, 3.5 GHz clock speed
    - **Processor B**: 8 cores, 4.0 GHz clock speed
    The program consists of the following instruction distribution and CPIs for each class:
    - Arithmetic Instructions: 40%, CPI = 1.5
    - Memory Access Instructions: 30%, CPI = 2.0
    - Branch Instructions: 30%, CPI = 2.5
    The program has a parallelizable portion (P) of 70% and a sequential portion (S) of 30%. Calculate how much more efficient (in terms of CPI) the branch instructions need to be for Processor A to match the performance of Processor B without changing the percentages of instruction counts.

Solutions:

Problem 1

Processor A:

- Instructions: (1 \times 10^9)
- CPI: 2
- Clock frequency: 2 GHz

Processor B:

- Instructions: (1 \times 10^9)
- CPI: 1.5
- Clock frequency: 1.5 GHz

Execution Time Calculation

For Processor A:

$$\text{Execution Time}_A = \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Frequency}} = \frac{1 \times 10^9 \times 2}{2 \times 10^9} = 1 \,\text{second}$$

For Processor B:

$$\text{Execution Time}_B = \frac{\text{Instruction Count} \times \text{CPI}}{\text{Clock Frequency}} = \frac{1 \times 10^9 \times 1.5}{1.5 \times 10^9} = 1 \,\text{second}$$

Performance Calculation

Performance is the inverse of execution time.

For Processor A:

$$\text{Performance}_A = \frac{1}{\text{Execution Time}_A} = 1 \,\text{programs/second}$$

For Processor B:

$$\text{Performance}_B = \frac{1}{\text{Execution Time}_B} = 1\,\text{programs/second}$$

Problem 2

Processor P:

- Clock speed: 2.5 GHz
- Initial CPI: 3
- Improved CPI: 2.5
- Instructions: 200 million

Initial Execution Time

$$\text{Execution Time}_{\text{initial}} = \frac{200 \times 10^6 \times 3}{2.5 \times 10^9} = 0.24\,\text{seconds}$$

Improved Execution Time

$$\text{Execution Time}_{\text{improved}} = \frac{200 \times 10^6 \times 2.5}{2.5 \times 10^9} = 0.2\,\text{seconds}$$

Performance Improvement

Initial Performance:

$$\text{Performance}_{\text{initial}} = \frac{1}{0.24} = 4.17\,\text{programs/second}$$

Improved Performance:

$$\text{Performance}_{\text{improved}} = \frac{1}{0.2} = 5\,\text{programs/second}$$

Percentage Improvement:

$$\text{Percentage Improvement} = \left(\frac{5 - 4.17}{4.17}\right) \times 100 = 19.9\%$$

Problem 3

Initial Conditions:

- CPI: 1.6
- Execution time: 0.5 seconds
- Clock speed: 3.6 GHz

Number of Instructions Executed

$$\text{Instructions} = \frac{0.5 \times 3.6 \times 10^9}{1.6} = 1.125 \times 10^9$$

New Execution Time with Improved CPI

Improved CPI: 1.4

$$\text{Execution Time}_{\text{new}} = \frac{1.125 \times 10^9 \times 1.4}{3.6 \times 10^9} = 0.4375 \,\text{seconds}$$

Problem 4

Processor G:

- Clock speed: 2.2 GHz
- Instructions: 600 million
- Execution time: 0.75 seconds

CPI Calculation

$$\text{CPI} = \frac{0.75 \times 2.2 \times 10^9}{600 \times 10^6} = 2.75$$

New Execution Time with Reduced Instructions

Reduced Instructions by 10%:

$$\text{New Instructions} = 600 \times 10^6 \times 0.9 = 540 \times 10^6$$

New Execution Time:

$$\text{Execution Time}_{\text{new}} = \frac{540 \times 10^6 \times 2.75}{2.2 \times 10^9} = 0.675 \,\text{seconds}$$

Problem 5

Initial Conditions:

- Initial CPI: 3
- Initial clock speed: 2.5 GHz
- Instructions: 1.3 billion

Initial Execution Time

$$\text{Execution Time}_{\text{initial}} = \frac{1.3 \times 10^9 \times 3}{2.5 \times 10^9} = 1.56 \,\text{seconds}$$

Optimized Conditions:

- Optimized CPI: 2.4
- Optimized clock speed: 2.2 GHz

Optimized Execution Time

$$\text{Execution Time}_{\text{optimized}} = \frac{1.3 \times 10^9 \times 2.4}{2.2 \times 10^9} \approx 1.418 \,\text{seconds}$$

Performance Calculation

Initial Performance:

$$\text{Performance}_{\text{initial}} = \frac{1}{1.56} \approx 0.641 \,\text{programs/second}$$

Optimized Performance:

$$\text{Performance}_{\text{optimized}} = \frac{1}{1.418} \approx 0.705 \, \text{programs/second}$$

Percentage Change in Performance

$$\text{Percentage Change in Performance} = \left(\frac{0.705 - 0.641}{0.641} \right) \times 100 \approx 10\%$$

Problem 6

Given:

- **Processor A:**
  - Number of cores (n): 4
  - Clock speed (f): 2.5 GHz
  - CPI: 2
- **Processor B:**
  - Number of cores (n): 1
  - Clock speed (f): 2 GHz
  - CPI: 1.8
- **Program:**
  - Instruction count: 1 billion (1e9)
  - Parallelizable portion (P): 60% (0.6)
  - Sequential portion (S): 40% (0.4)

Execution Time Calculation

For Processor A:

$$T_A(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left(S + \frac{P}{n} \right)$$

$$T_A(4) = \frac{1 \times 10^9 \times 2}{2.5 \times 10^9} \times \left(0.4 + \frac{0.6}{4}\right)$$

$$T_A(4) = \frac{2 \times 10^9}{2.5 \times 10^9} \times (0.4 + 0.15)$$

$$T_A(4) = 0.8 \times 0.55 = 0.44 \, \text{seconds}$$

For Processor B:

$$T_B(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left(S + \frac{P}{n}\right)$$

$$T_B(1) = \frac{1 \times 10^9 \times 1.8}{2 \times 10^9} \times \left(0.4 + \frac{0.6}{1}\right)$$

$$T_B(1) = \frac{1.8 \times 10^9}{2 \times 10^9} \times (0.4 + 0.6)$$

$$T_B(1) = 0.9 \times 1 = 0.9 \, \text{seconds}$$

Performance Calculation

For Processor A:

$$\text{Performance}_A(n) = \frac{f}{\text{Instruction Count} \times \text{CPI}} \times \frac{1}{S + \frac{P}{n}}$$

$$\text{Performance}_A(4) = \frac{2.5 \times 10^9}{1 \times 10^9 \times 2} \times \frac{1}{0.4 + \frac{0.6}{4}}$$

$$\text{Performance}_A(4) = \frac{2.5}{2} \times \frac{1}{0.55}$$

$$\text{Performance}_A(4) = 1.25 \times 1.818 = 2.273 \, \text{programs/second}$$

For Processor B:

$$\text{Performance}_B(n) = \frac{f}{\text{Instruction Count} \times \text{CPI}} \times \frac{1}{S + \frac{P}{n}}$$

$$\text{Performance}_B(1) = \frac{2 \times 10^9}{1 \times 10^9 \times 1.8} \times \frac{1}{0.4 + \frac{0.6}{1}}$$

$$\text{Performance}_B(1) = \frac{2}{1.8} \times \frac{1}{1}$$

$$\text{Performance}_B(1) = 1.111 \times 1 = 1.111 \, \text{programs/second}$$

Problem 7

Given:

- **Program:**
  - Instruction count: 2 billion (2e9)
  - Parallelizable portion (P): 70% (0.7)
  - Sequential portion (S): 30% (0.3)
- **Processor X:**
  - Number of cores (n): 6
  - Clock speed (f): 3 GHz
  - CPI: 2.2
- **Processor Y:**
  - Number of cores (n): 4
  - Clock speed (f): 2.8 GHz
  - CPI: 1.9

Execution Time Calculation

For Processor X:

$$T_X(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left(S + \frac{P}{n} \right)$$

$$T_X(6) = \frac{2 \times 10^9 \times 2.2}{3 \times 10^9} \times \left(0.3 + \frac{0.7}{6}\right)$$

$$T_X(6) = \frac{4.4 \times 10^9}{3 \times 10^9} \times (0.3 + 0.1167)$$

$$T_X(6) = 1.4667 \times 0.4167 = 0.611 \, \text{seconds}$$

For Processor Y:

$$T_Y(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left(S + \frac{P}{n}\right)$$

$$T_Y(4) = \frac{2 \times 10^9 \times 1.9}{2.8 \times 10^9} \times \left(0.3 + \frac{0.7}{4}\right)$$

$$T_Y(4) = \frac{3.8 \times 10^9}{2.8 \times 10^9} \times (0.3 + 0.175)$$

$$T_Y(4) = 1.3571 \times 0.475 = 0.6456 \, \text{seconds}$$

Performance Calculation

For Processor X:

$$\text{Performance}_X(n) = \frac{f}{\text{Instruction Count} \times \text{CPI}} \times \frac{1}{S + \frac{P}{n}}$$

$$\text{Performance}_X(6) = \frac{3 \times 10^9}{2 \times 10^9 \times 2.2} \times \frac{1}{0.3 + \frac{0.7}{6}}$$

$$\text{Performance}_X(6) = \frac{3}{4.4} \times \frac{1}{0.4167}$$

$$\text{Performance}_X(6) = 0.6818 \times 2.4 = 1.637 \, \text{programs/second}$$

For Processor Y:

$$\text{Performance}_Y(n) = \frac{f}{\text{Instruction Count} \times \text{CPI}} \times \frac{1}{S + \frac{P}{n}}$$

$$\text{Performance}_Y(4) = \frac{2.8 \times 10^9}{2 \times 10^9 \times 1.9} \times \frac{1}{0.3 + \frac{0.7}{4}}$$

$$\text{Performance}_Y(4) = \frac{2.8}{3.8} \times \frac{1}{0.475}$$

$$\text{Performance}_Y(4) = 0.7368 \times 2.105 = 1.551 \,\text{programs/second}$$

Problem 8

Given:

- **Processor A:**
  - Number of cores (n): 4
  - Clock speed (f): 3 GHz
  - CPI: 2
- **Processor B:**
  - Number of cores (n): 8
  - Clock speed (f): 2.5 GHz
  - CPI: 2.5
- **Program:**
  - Instruction count: 1.5 billion (1.5e9)

Execution Time Calculation for Processor B

For Processor B (assuming S = 1 and P = 0):

$$T_B(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left(S + \frac{P}{n}\right)$$

$$T_B(8) = \frac{1.5 \times 10^9 \times 2.5}{2.5 \times 10^9} \times 1$$

$$T_B(8) = \frac{3.75 \times 10^9}{2.5 \times 10^9} \times 1$$

$$T_B(8) = 1.5 \, \text{seconds}$$

Execution Time Calculation for Processor A

Assume that for Processor A, (S = x) and (P = 1 - x):

$$T_A(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left(S + \frac{P}{n} \right)$$

$$T_A(4) = \frac{1.5 \times 10^9 \times 2}{3 \times 10^9} \times \left(x + \frac{1 - x}{4} \right)$$

$$T_A(4) = \frac{3 \times 10^9}{3 \times 10^9} \times \left(x + \frac{1 - x}{4} \right)$$

$$T_A(4) = 1 \times \left(x + \frac{1 - x}{4} \right)$$

$$T_A(4) = x + \frac{1 - x}{4}$$

Set the execution times equal to each other:

$$T_A(4) = T_B(8)$$

$$x + \frac{1 - x}{4} = 1.5$$

Multiply through by 4 to clear the fraction:

$$4x + 1 - x = 6$$

$$3x + 1 = 6$$

Subtract 1 from both sides:

$$3x = 5$$

Solve for (x):

$$x = \frac{5}{3}$$

Thus, the sequential portion (x) is:

$$x = \frac{5}{12}$$

And the parallelizable portion (1 - x) is:

$$1 - x = 1 - \frac{5}{12} = \frac{7}{12}$$

For Processor A to match the performance of Processor B, approximately 7/12 or 58.33% of the program must be parallelized.

Problem 9

Given:

- **Processor M:**
  - Number of cores (n): 4
  - Clock speed (f): 3.2 GHz
  - CPI: 2.3
- **Processor N:**
  - Number of cores (n): 32
  - Clock speed (f): 4 GHz
  - CPI: 2.0
- **Program:**
  - Instruction count: (x) instructions
  - Initial parallelizable portion (P): 60% (0.6)
  - Sequential portion (S): 40% (0.4)

Execution Time Calculation for Processor N

For Processor N:

$$T_N(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left(S + \frac{P}{n} \right)$$

$$T_N(32) = \frac{x \times 2.0}{4 \times 10^9} \times \left(0.4 + \frac{0.6}{32} \right)$$

$$T_N(32) = \frac{2x}{4 \times 10^9} \times (0.4 + 0.01875)$$

$$T_N(32) = \frac{x}{2 \times 10^9} \times 0.41875$$

$$T_N(32) = \frac{0.41875x}{2 \times 10^9} = 0.209375 \times \frac{x}{10^9} \text{ seconds}$$

Execution Time Calculation for Processor M

Assume the new parallelizable portion is (P) and the new sequential portion is (1 - P):

$$T_M(n) = \frac{\text{Instruction Count} \times \text{CPI}}{f} \times \left((1 - P) + \frac{P}{n} \right)$$

To match the execution time of Processor N, set (T_M) equal to (T_N):

$$T_M(4) = T_N(32)$$

$$\frac{x \times 2.3}{3.2 \times 10^9} \times \left((1 - P) + \frac{P}{4} \right) = 0.209375 \times \frac{x}{10^9}$$

Cancel out (x) from both sides:

$$\frac{2.3}{3.2} \times \left((1 - P) + \frac{P}{4} \right) = 0.209375$$

$$0.71875 \times \left((1 - P) + \frac{P}{4} \right) = 0.209375$$

$$(1 - P) + \frac{P}{4} = \frac{0.209375}{0.71875}$$

$$(1 - P) + \frac{P}{4} = 0.2916667$$

Multiply through by 4 to clear the fraction:

$$4(1 - P) + P = 4 \times 0.2916667$$

$$4 - 4P + P = 1.1666668$$

$$4 - 3P = 1.1666668$$

Subtract 1.1666668 from both sides:

$$4 - 1.1666668 = 3P$$

$$2.8333332 = 3P$$

Solve for (P):

$$P = \frac{2.8333332}{3} = 0.9444444$$

The new parallelizable portion (P) is approximately 0.944 or 94.44%. Therefore, to match the performance of Processor N, the parallelizable portion of the program must be increased to approximately 94.44% (increased by 94.44% - 60% = 34.44%).

Problem 10

1. **Calculate the total number of clock cycles for each class of instructions**:

$$\text{Total Clock Cycles} = \sum (\text{Number of Instructions} \times \text{CPI})$$

- For Arithmetic Instructions:

$$1.2 \times 10^9 \times 1.8 = 2.16 \times 10^9 \text{ cycles}$$

- For Branch Instructions:

$$0.8 \times 10^9 \times 2.5 = 2.0 \times 10^9 \text{ cycles}$$

- For Memory Access Instructions:

$$1.0 \times 10^9 \times 3.0 = 3.0 \times 10^9 \text{ cycles}$$

2. **Calculate the total number of clock cycles for the entire program**:

$$\text{Total Clock Cycles} = 2.16 \times 10^9 + 2.0 \times 10^9 + 3.0 \times 10^9 = 7.16 \times 10^9 \text{ cycles}$$

3. **Convert clock cycles to execution time**:

Given the clock speed of the processor is 3.5 GHz (which is 3.5×10^9 cycles per second), the execution time T can be calculated as:

$$T = \frac{\text{Total Clock Cycles}}{\text{Clock Speed}}$$

$$T = \frac{7.16 \times 10^9}{3.5 \times 10^9}$$

$$T \approx 2.046 \text{ seconds}$$

4. **Calculate the overall performance in programs per second**:

Performance P in programs per second is the inverse of the execution time:

$$P = \frac{1}{T}$$

$$P \approx \frac{1}{2.046} \approx 0.489 \text{ programs per second}$$

Final Answers

- **Total Execution Time**:

$$T \approx 2.046 \text{ seconds}$$

- **Overall Performance**:

$$P \approx 0.489 \text{ programs per second}$$

Hints for Problem 11

1. Calculate the initial average CPI using the given instruction percentages and CPIs.
2. Use Amdahl's Law to set up the execution time formula for both processors.
3. Assume 100 total instructions; express the execution times in terms of these instructions.
4. To match performance, reduce branch instructions by (x); branches become (30 - x), and the total instructions become (100 - x).
5. Write equations for execution times with the new distribution for Processor A and original for Processor B.
6. Set the execution times equal and solve for (x).
7. Verify the reduction in branch percentage makes Processor A's execution time match Processor B's.

Hints for Problem 12

1. Calculate the initial average CPI using the given percentages and CPIs for each instruction type.
2. Use Amdahl's Law to set up the execution time formula for both processors considering the parallel (P) and sequential (S) portions.
3. Write down the execution time equations for Processor A and Processor B using their specifications.

4. Assume the new CPI for branch instructions is (x); recalculate the average CPI for Processor A with this new CPI.
5. Substitute the new average CPI in Processor A's execution time equation.
6. Set Processor A's execution time equal to Processor B's execution time to find (x).
7. Solve for (x) to determine the required CPI reduction for branch instructions on Processor A.
8. Verify that the new CPI value allows Processor A to match the performance of Processor B.

amdahl computer architecture performance

Written by

Jyotiprakash Mishra

I am Jyotiprakash, a deeply driven computer systems engineer, software developer, teacher, and philosopher. With a decade of professional experience, I have contributed to various cutting-edge software products in

network security, mobile apps, and healthcare software at renowned companies like Oracle, Yahoo, and Epic. My academic journey has taken me to prestigious institutions such as the University of Wisconsin-Madison and BITS Pilani in India, where I consistently ranked among the top of my class.

At my core, I am a computer enthusiast with a profound interest in understanding the intricacies of computer programming. My skills are not limited to application programming in Java; I have also delved deeply into computer hardware, learning about various architectures, low-level assembly programming, Linux kernel implementation, and writing device drivers. The contributions of Linus Torvalds, Ken Thompson, and Dennis Ritchie—who revolutionized the computer industry—inspire me. I believe that real contributions to computer science are made by mastering all levels of abstraction and understanding systems inside out.

In addition to my professional pursuits, I am passionate about teaching and sharing knowledge. I have spent two years as a teaching assistant at UW Madison, where I taught complex concepts in operating systems, computer graphics, and data structures to both graduate and undergraduate students. Currently, I am an assistant professor at KIIT, Bhubaneswar, where I continue to teach computer science to undergraduate and graduate students. I am also working on writing a few free books on systems programming, as I believe in freely sharing knowledge to empower others.

Follow

MORE ARTICLES

Jyotiprakash Mishra

The Ultimate Guide to Smart Investment Planning and Wealth Withdrawal Using Python

Investing can be one of the most powerful ways to secure your financial future, but it comes with a ...

Jyotiprakash Mishra

Data-Level Parallelism

Data-Level Parallelism (DLP) refers to the parallel execution of identical operations on different e...

Jyotiprakash Mishra

Protection: Virtual Everything

In the world of computing, ensuring security, privacy, and efficient management of resources are som...