CSE240A HW 1

Xinhao Luo

TOTAL POINTS

29 / 30

QUESTION 1

CPI 20 pts

1.1 2 / 2

✓ **- 0 pts** Correct

1.2 6 / 6

✓ **- 0 pts** Correct

1.3 3 / 3

✓ **- 0 pts** Correct

1.4 3 / 3

✓ **- 0 pts** Correct

1.5 3 / 3

✓ **- 0 pts** Correct

1.6 3 / 3

✓ **- 0 pts** Correct

QUESTION 2

Amdahl's Law 10 pts

2.1 2 / 3

✓ **- 1 pts** Calculation is wrong. (The P in the Amdahl's Law should be calculated based on the execution time instead of the number of instructions)

2.2 2 / 2

✓ **- 0 pts** Correct

2.3 2 / 2

✓ **- 0 pts** Correct

2.4 3 / 3

✓ **- 0 pts** Correct

CSE 240A Homework 1

Performance Metrics and ISA

Xinhao Luo

Due October 27, 2022 11:59 PM

Instructions

- Submit typed answers to the following questions as a PDF via Gradescope.
- Due: October 25, 2022 at 11:59 PM.
- Homework is to be done individually.
- There are two questions for a total of 30 points (7.5% of your total grade).
- It would be great if you can finish the HW by typing rather than handwriting.

1 CPI (20 points)

Assume a processor with instruction frequencies in percentage and costs
Integer ALU: 50%, 1 cycle
Load: 20%, 2 cycles
Store: 10%, 2 cycles
Branch: 20%, 3 cycles
The clock frequency is 500MHz.

1. Please compute the CPI. (2 points)

   $CPI = 1 * 0.5 + 2 * 0.2 + 2 * 0.1 + 3 * 0.2 = 1.7$

2. Please identify which case achieves more performance improvement, and explain why (note that the total number of instructions in the program does not change):

   (a) Increase Integer ALU frequency by 20%, and increase Load cycle to 4 cycles.

   Integer ALU has frequency overall $0.5 * 1.2 = 0.6$, then the rest redistribute the 0.4
   Load & Branch: $0.4/5 * 2 = 0.16$, Store: $0.4/5 = 0.08$
   $CPI = 1 * 0.6 + 4 * 0.16 + 2 * 0.08 + 3 * 0.16 = 1.88$

   (b) Reduce Integer ALU frequency by 20%, and increase Branch frequency by 50%.

   Integer ALU has frequency overall $0.5 * 0.8 = 0.4$, Branch frequency is $0.2 * 1.5 = 0.3$ then the rest redistribute the 0.3
   Load: $0.3/3 * 2 = 0.2$, Store: $0.3/3 = 0.1$
   $CPI = 1 * 0.4 + 2 * 0.2 + 2 * 0.1 + 3 * 0.3 = 1.9$

1.1 **2 / 2**

✓ **- 0 pts** Correct

CSE 240A Homework 1

Performance Metrics and ISA

Xinhao Luo

Due October 27, 2022 11:59 PM

Instructions

- Submit typed answers to the following questions as a PDF via Gradescope.
- Due: October 25, 2022 at 11:59 PM.
- Homework is to be done individually.
- There are two questions for a total of 30 points (7.5% of your total grade).
- It would be great if you can finish the HW by typing rather than handwriting.

1 CPI (20 points)

Assume a processor with instruction frequencies in percentage and costs
Integer ALU: 50%, 1 cycle
Load: 20%, 2 cycles
Store: 10%, 2 cycles
Branch: 20%, 3 cycles
The clock frequency is 500MHz.

1. Please compute the CPI. (2 points)

   $CPI = 1 * 0.5 + 2 * 0.2 + 2 * 0.1 + 3 * 0.2 = 1.7$

2. Please identify which case achieves more performance improvement, and explain why (note that the total number of instructions in the program does not change):

   (a) Increase Integer ALU frequency by 20%, and increase Load cycle to 4 cycles.

   Integer ALU has frequency overall $0.5 * 1.2 = 0.6$, then the rest redistribute the 0.4
   Load & Branch: $0.4/5 * 2 = 0.16$, Store: $0.4/5 = 0.08$
   $CPI = 1 * 0.6 + 4 * 0.16 + 2 * 0.08 + 3 * 0.16 = 1.88$

   (b) Reduce Integer ALU frequency by 20%, and increase Branch frequency by 50%.

   Integer ALU has frequency overall $0.5 * 0.8 = 0.4$, Branch frequency is $0.2 * 1.5 = 0.3$ then the rest redistribute the 0.3
   Load: $0.3/3 * 2 = 0.2$, Store: $0.3/3 = 0.1$
   $CPI = 1 * 0.4 + 2 * 0.2 + 2 * 0.1 + 3 * 0.3 = 1.9$

(c) Reduce Branch cycle to 2 cycles, and reduce ALU frequency by 40%. (6 points)

Integer ALU has frequency overall $0.5 * 0.6 = 0.3$, then the rest redistribute the 0.7
Load & Branch: $0.7/5 * 2 = 0.28$, Store: $0.7/5 = 0.14$
$CPI = 1 * 0.3 + 2 * 0.28 + 2 * 0.14 + 2 * 0.28 = 1.7$

There is no improvement since all CPI calculated are either higher or equal to the original CPI in question 1.

3. Please compute the MIPS (Million instructions per second) for the three cases in Problem 1.2. (3 points)

500Mhz means $5 * 10^8$ cycle per second

(a) $MIPS = \frac{1}{1.88*10^6/(5*10^8)} \approx 266$

(b) $MIPS = \frac{1}{1.9*10^6/(5*10^8)} \approx 263$

(c) $MIPS = \frac{1}{1.7*10^6/(5*10^8)} \approx 294$

4. We observe that 30% of Integer ALU operations are paired with Load. If we were to replace these Integer ALU ops and their Loads with a single instruction that executes in 1 clock cycle, the number of clocks cycles taken for the branch operation to perform increases to 5 cycles. Compute the CPI for this new version. (3 points)

We have a new type Instruction and its frequency is $0.5 * 0.3 = 0.15$, and this also remove part of the instruction from load and ALU, so they have updated frequency of $0.5-0.15 = 0.35$ and $0.2-0.15 = 0.05$
$CPI = (0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1)/0.85 = 2.12$

5. If a program includes 500 million integer ALU operations, 80 million load operations, 200 million store operations, and 100 million branch operations. How long does it take if we execute it on this processor (in the original statement)? Please provide the answer in seconds. (3 points)

$Cycles = 5 * 10^8 * 1 + 8 * 10^7 * 2 + 2 * 10^8 * 2 + 1 * 10^8 * 3 = 1.36 * 10^9$
$1.36 * 10^9/(5 * 10^8) = 2.72s$
So it will take 2.72s to finish.

6. If you hope to optimize this processor (in the original statement), which instruction will you plan to optimize, and how? Explain why. (3 points)

It would be optimize the branch instruction. Since the most operation, ALU, has already reach 1 cycle, and it is pretty hard to further optimize down this part. Among the rest of the instruction, branch and load takes the second highest number of operation, and branch takes 3 cycles, the most among others. This also brings some gap for improvement.

To improve branch, we may make use of branch prediction to bring instruction to execution stage earlier. By removing this stall, the overall cycle branch instruction take could be lowered. Although under some cases it has to be reverted and cause larger delay, this situation could be avoided with higher prediction accuracy.

2 Amdahl's Law (10 points)

We use the same processor in Problem 1.

1.2 6 / 6

✓ - **0 pts** Correct

(c) Reduce Branch cycle to 2 cycles, and reduce ALU frequency by 40%. (6 points)

Integer ALU has frequency overall $0.5 * 0.6 = 0.3$, then the rest redistribute the 0.7
Load & Branch: $0.7/5 * 2 = 0.28$, Store: $0.7/5 = 0.14$
$CPI = 1 * 0.3 + 2 * 0.28 + 2 * 0.14 + 2 * 0.28 = 1.7$

There is no improvement since all CPI calculated are either higher or equal to the original CPI in question 1.

3. Please compute the MIPS (Million instructions per second) for the three cases in Problem 1.2. (3 points)

500Mhz means $5 * 10^8$ cycle per second

(a) $MIPS = \frac{1}{1.88*10^6/(5*10^8)} \approx 266$

(b) $MIPS = \frac{1}{1.9*10^6/(5*10^8)} \approx 263$

(c) $MIPS = \frac{1}{1.7*10^6/(5*10^8)} \approx 294$

4. We observe that 30% of Integer ALU operations are paired with Load. If we were to replace these Integer ALU ops and their Loads with a single instruction that executes in 1 clock cycle, the number of clocks cycles taken for the branch operation to perform increases to 5 cycles. Compute the CPI for this new version. (3 points)

We have a new type Instruction and its frequency is $0.5 * 0.3 = 0.15$, and this also remove part of the instruction from load and ALU, so they have updated frequency of $0.5-0.15 = 0.35$ and $0.2-0.15 = 0.05$
$CPI = (0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1)/0.85 = 2.12$

5. If a program includes 500 million integer ALU operations, 80 million load operations, 200 million store operations, and 100 million branch operations. How long does it take if we execute it on this processor (in the original statement)? Please provide the answer in seconds. (3 points)

$Cycles = 5 * 10^8 * 1 + 8 * 10^7 * 2 + 2 * 10^8 * 2 + 1 * 10^8 * 3 = 1.36 * 10^9$
$1.36 * 10^9/(5 * 10^8) = 2.72s$
So it will take 2.72s to finish.

6. If you hope to optimize this processor (in the original statement), which instruction will you plan to optimize, and how? Explain why. (3 points)

It would be optimize the branch instruction. Since the most operation, ALU, has already reach 1 cycle, and it is pretty hard to further optimize down this part. Among the rest of the instruction, branch and load takes the second highest number of operation, and branch takes 3 cycles, the most among others. This also brings some gap for improvement.

To improve branch, we may make use of branch prediction to bring instruction to execution stage earlier. By removing this stall, the overall cycle branch instruction take could be lowered. Although under some cases it has to be reverted and cause larger delay, this situation could be avoided with higher prediction accuracy.

2 Amdahl's Law (10 points)

We use the same processor in Problem 1.

1.3 3 / 3

✓ - **0 pts** Correct

(c) Reduce Branch cycle to 2 cycles, and reduce ALU frequency by 40%. (6 points)

Integer ALU has frequency overall $0.5 * 0.6 = 0.3$, then the rest redistribute the 0.7
Load & Branch: $0.7/5 * 2 = 0.28$, Store: $0.7/5 = 0.14$
$CPI = 1 * 0.3 + 2 * 0.28 + 2 * 0.14 + 2 * 0.28 = 1.7$

There is no improvement since all CPI calculated are either higher or equal to the original CPI in question 1.

3. Please compute the MIPS (Million instructions per second) for the three cases in Problem 1.2. (3 points)

500Mhz means $5 * 10^8$ cycle per second

(a) $MIPS = \frac{1}{1.88*10^6/(5*10^8)} \approx 266$

(b) $MIPS = \frac{1}{1.9*10^6/(5*10^8)} \approx 263$

(c) $MIPS = \frac{1}{1.7*10^6/(5*10^8)} \approx 294$

4. We observe that 30% of Integer ALU operations are paired with Load. If we were to replace these Integer ALU ops and their Loads with a single instruction that executes in 1 clock cycle, the number of clocks cycles taken for the branch operation to perform increases to 5 cycles. Compute the CPI for this new version. (3 points)

We have a new type Instruction and its frequency is $0.5 * 0.3 = 0.15$, and this also remove part of the instruction from load and ALU, so they have updated frequency of $0.5-0.15 = 0.35$ and $0.2-0.15 = 0.05$
$CPI = (0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1)/0.85 = 2.12$

5. If a program includes 500 million integer ALU operations, 80 million load operations, 200 million store operations, and 100 million branch operations. How long does it take if we execute it on this processor (in the original statement)? Please provide the answer in seconds. (3 points)

$Cycles = 5 * 10^8 * 1 + 8 * 10^7 * 2 + 2 * 10^8 * 2 + 1 * 10^8 * 3 = 1.36 * 10^9$
$1.36 * 10^9/(5 * 10^8) = 2.72s$
So it will take 2.72s to finish.

6. If you hope to optimize this processor (in the original statement), which instruction will you plan to optimize, and how? Explain why. (3 points)

It would be optimize the branch instruction. Since the most operation, ALU, has already reach 1 cycle, and it is pretty hard to further optimize down this part. Among the rest of the instruction, branch and load takes the second highest number of operation, and branch takes 3 cycles, the most among others. This also brings some gap for improvement.

To improve branch, we may make use of branch prediction to bring instruction to execution stage earlier. By removing this stall, the overall cycle branch instruction take could be lowered. Although under some cases it has to be reverted and cause larger delay, this situation could be avoided with higher prediction accuracy.

2 Amdahl's Law (10 points)

We use the same processor in Problem 1.

1.4 3 / 3

✓ - **0 pts** Correct

(c) Reduce Branch cycle to 2 cycles, and reduce ALU frequency by 40%. (6 points)

Integer ALU has frequency overall $0.5 * 0.6 = 0.3$, then the rest redistribute the 0.7
Load & Branch: $0.7/5 * 2 = 0.28$, Store: $0.7/5 = 0.14$
$CPI = 1 * 0.3 + 2 * 0.28 + 2 * 0.14 + 2 * 0.28 = 1.7$

There is no improvement since all CPI calculated are either higher or equal to the original CPI in question 1.

3. Please compute the MIPS (Million instructions per second) for the three cases in Problem 1.2. (3 points)

500Mhz means $5 * 10^8$ cycle per second

(a) $MIPS = \frac{1}{1.88*10^6/(5*10^8)} \approx 266$

(b) $MIPS = \frac{1}{1.9*10^6/(5*10^8)} \approx 263$

(c) $MIPS = \frac{1}{1.7*10^6/(5*10^8)} \approx 294$

4. We observe that 30% of Integer ALU operations are paired with Load. If we were to replace these Integer ALU ops and their Loads with a single instruction that executes in 1 clock cycle, the number of clocks cycles taken for the branch operation to perform increases to 5 cycles. Compute the CPI for this new version. (3 points)

We have a new type Instruction and its frequency is $0.5 * 0.3 = 0.15$, and this also remove part of the instruction from load and ALU, so they have updated frequency of $0.5-0.15 = 0.35$ and $0.2-0.15 = 0.05$
$CPI = (0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1)/0.85 = 2.12$

5. If a program includes 500 million integer ALU operations, 80 million load operations, 200 million store operations, and 100 million branch operations. How long does it take if we execute it on this processor (in the original statement)? Please provide the answer in seconds. (3 points)

$Cycles = 5 * 10^8 * 1 + 8 * 10^7 * 2 + 2 * 10^8 * 2 + 1 * 10^8 * 3 = 1.36 * 10^9$
$1.36 * 10^9/(5 * 10^8) = 2.72s$
So it will take 2.72s to finish.

6. If you hope to optimize this processor (in the original statement), which instruction will you plan to optimize, and how? Explain why. (3 points)

It would be optimize the branch instruction. Since the most operation, ALU, has already reach 1 cycle, and it is pretty hard to further optimize down this part. Among the rest of the instruction, branch and load takes the second highest number of operation, and branch takes 3 cycles, the most among others. This also brings some gap for improvement.

To improve branch, we may make use of branch prediction to bring instruction to execution stage earlier. By removing this stall, the overall cycle branch instruction take could be lowered. Although under some cases it has to be reverted and cause larger delay, this situation could be avoided with higher prediction accuracy.

2 Amdahl's Law (10 points)

We use the same processor in Problem 1.

1.5 3 / 3

✓ - 0 pts Correct

(c) Reduce Branch cycle to 2 cycles, and reduce ALU frequency by 40%. (6 points)

Integer ALU has frequency overall $0.5 * 0.6 = 0.3$, then the rest redistribute the 0.7
Load & Branch: $0.7/5 * 2 = 0.28$, Store: $0.7/5 = 0.14$
$CPI = 1 * 0.3 + 2 * 0.28 + 2 * 0.14 + 2 * 0.28 = 1.7$

There is no improvement since all CPI calculated are either higher or equal to the original CPI in question 1.

3. Please compute the MIPS (Million instructions per second) for the three cases in Problem 1.2. (3 points)

500Mhz means $5 * 10^8$ cycle per second

(a) $MIPS = \frac{1}{1.88*10^6/(5*10^8)} \approx 266$

(b) $MIPS = \frac{1}{1.9*10^6/(5*10^8)} \approx 263$

(c) $MIPS = \frac{1}{1.7*10^6/(5*10^8)} \approx 294$

4. We observe that 30% of Integer ALU operations are paired with Load. If we were to replace these Integer ALU ops and their Loads with a single instruction that executes in 1 clock cycle, the number of clocks cycles taken for the branch operation to perform increases to 5 cycles. Compute the CPI for this new version. (3 points)

We have a new type Instruction and its frequency is $0.5 * 0.3 = 0.15$, and this also remove part of the instruction from load and ALU, so they have updated frequency of $0.5-0.15 = 0.35$ and $0.2-0.15 = 0.05$
$CPI = (0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1)/0.85 = 2.12$

5. If a program includes 500 million integer ALU operations, 80 million load operations, 200 million store operations, and 100 million branch operations. How long does it take if we execute it on this processor (in the original statement)? Please provide the answer in seconds. (3 points)

$Cycles = 5 * 10^8 * 1 + 8 * 10^7 * 2 + 2 * 10^8 * 2 + 1 * 10^8 * 3 = 1.36 * 10^9$
$1.36 * 10^9/(5 * 10^8) = 2.72s$
So it will take 2.72s to finish.

6. If you hope to optimize this processor (in the original statement), which instruction will you plan to optimize, and how? Explain why. (3 points)

It would be optimize the branch instruction. Since the most operation, ALU, has already reach 1 cycle, and it is pretty hard to further optimize down this part. Among the rest of the instruction, branch and load takes the second highest number of operation, and branch takes 3 cycles, the most among others. This also brings some gap for improvement.

To improve branch, we may make use of branch prediction to bring instruction to execution stage earlier. By removing this stall, the overall cycle branch instruction take could be lowered. Although under some cases it has to be reverted and cause larger delay, this situation could be avoided with higher prediction accuracy.

2 Amdahl's Law (10 points)

We use the same processor in Problem 1.

1.6 3 / 3

✓ - 0 pts Correct

1. Using the processor in problem 1's original statement, if I want to achieve at least a 1.15X speedup by accelerating one operation, how can I get there? (Hint: there is no space to speedup ALU, as it only occupies 1 cycle. Every operation at least consumes 1 cycle, and the cycle should be integer.) (3 points)
   We have the Amdahl's Law:
   $$S_{total} = \frac{1}{(1-P) + \frac{P}{S}} \tag{1}$$
   We know have the $S_{total} = 1.15$, and we would like to get the S for each instruction.
   $$S = \frac{P}{\frac{1}{S_{total}} - (1-P)} \tag{2}$$
   Load $S = \frac{0.2}{\frac{1}{1.15} - (1-0.2)} = 2.875$, so we need to have a 2.875X speed up on our load instruction, which means load instruction needs to be done in $2/2.875 \approx 0.7$ cycles
   Store $S = \frac{0.1}{\frac{1}{1.15} - (1-0.1)} = -3.28$, this negative value means no matter how we speed up this part, we can't reach the goal
   Branch $S = \frac{0.2}{\frac{1}{1.15} - (1-0.2)} = 2.875$, so we need to have a 2.875X speed up on our branch instruction, which means branch instruction needs to be done in $3/2.875 \approx 1$ cycles.

2. Assume this processor supports parallelization, and if I want to run a program on it: this program contains the same operations with Problem 1.4 . The processor can run at most 1000 threads parallely. Each thread is only used for executing ALU operations (including ALU operations and combined ALU-Load). How much performance improvement can I achieve? (2 points)

   $$P_{ALU} = \frac{0.35 * 1}{0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1} \approx 0.194 \tag{3}$$
   $$P_{ALU-LOAD} = \frac{0.15 * 1}{0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1} \approx 0.083 \tag{4}$$
   Together we have $P = 0.194 + 0.083 = 0.277$
   From
   $$S_{total-parallel} = \frac{1}{(1-P) + \frac{P}{N}} \tag{5}$$
   Then we have
   $$S_{total} = \frac{1}{(1-0.277) + \frac{0.277}{1000}} \approx 1.383 \tag{6}$$
   Thus, we can achieve 1.383X performance improvement

3. Unfortunately, my processor is I/O bound, as it consumes 80% of the total time to wait for data. What is the highest speedup of such a processor? (2 points)

   Our instructions under such situation will only take $1 - 0.8 = 0.2$ of the runtime. Assume we have infinite speed up over this portion.
   $$S_{total} = \frac{1}{(1-0.2) + \frac{0.2}{\infty}} = 1.25 \tag{7}$$
   Our highest speedup at this time would be 1.25X

2.1 2 / 3

✓ **- 1 pts** Calculation is wrong. (The P in the Amdahl's Law should be calculated based on the execution time instead of the number of instructions)

1. Using the processor in problem 1's original statement, if I want to achieve at least a 1.15X speedup by accelerating one operation, how can I get there? (Hint: there is no space to speedup ALU, as it only occupies 1 cycle. Every operation at least consumes 1 cycle, and the cycle should be integer.) (3 points)
   We have the Amdahl's Law:
   $$S_{total} = \frac{1}{(1-P) + \frac{P}{S}} \tag{1}$$
   We know have the $S_{total} = 1.15$, and we would like to get the S for each instruction.
   $$S = \frac{P}{\frac{1}{S_{total}} - (1-P)} \tag{2}$$
   Load $S = \frac{0.2}{\frac{1}{1.15} - (1-0.2)} = 2.875$, so we need to have a 2.875X speed up on our load instruction, which means load instruction needs to be done in $2/2.875 \approx 0.7$ cycles
   Store $S = \frac{0.1}{\frac{1}{1.15} - (1-0.1)} = -3.28$, this negative value means no matter how we speed up this part, we can't reach the goal
   Branch $S = \frac{0.2}{\frac{1}{1.15} - (1-0.2)} = 2.875$, so we need to have a 2.875X speed up on our branch instruction, which means branch instruction needs to be done in $3/2.875 \approx 1$ cycles.

2. Assume this processor supports parallelization, and if I want to run a program on it: this program contains the same operations with Problem 1.4 . The processor can run at most 1000 threads parallely. Each thread is only used for executing ALU operations (including ALU operations and combined ALU-Load). How much performance improvement can I achieve? (2 points)

   $$P_{ALU} = \frac{0.35 * 1}{0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1} \approx 0.194 \tag{3}$$
   $$P_{ALU-LOAD} = \frac{0.15 * 1}{0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1} \approx 0.083 \tag{4}$$
   Together we have $P = 0.194 + 0.083 = 0.277$
   From
   $$S_{total-parallel} = \frac{1}{(1-P) + \frac{P}{N}} \tag{5}$$
   Then we have
   $$S_{total} = \frac{1}{(1-0.277) + \frac{0.277}{1000}} \approx 1.383 \tag{6}$$
   Thus, we can achieve 1.383X performance improvement

3. Unfortunately, my processor is I/O bound, as it consumes 80% of the total time to wait for data. What is the highest speedup of such a processor? (2 points)

   Our instructions under such situation will only take $1 - 0.8 = 0.2$ of the runtime. Assume we have infinite speed up over this portion.
   $$S_{total} = \frac{1}{(1-0.2) + \frac{0.2}{\infty}} = 1.25 \tag{7}$$
   Our highest speedup at this time would be 1.25X

2.2 2 / 2

✓ - 0 pts Correct

1. Using the processor in problem 1's original statement, if I want to achieve at least a 1.15X speedup by accelerating one operation, how can I get there? (Hint: there is no space to speedup ALU, as it only occupies 1 cycle. Every operation at least consumes 1 cycle, and the cycle should be integer.) (3 points)
   We have the Amdahl's Law:
   $$S_{total} = \frac{1}{(1-P) + \frac{P}{S}} \tag{1}$$
   We know have the $S_{total} = 1.15$, and we would like to get the S for each instruction.
   $$S = \frac{P}{\frac{1}{S_{total}} - (1-P)} \tag{2}$$
   Load $S = \frac{0.2}{\frac{1}{1.15} - (1-0.2)} = 2.875$, so we need to have a 2.875X speed up on our load instruction, which means load instruction needs to be done in $2/2.875 \approx 0.7$ cycles
   Store $S = \frac{0.1}{\frac{1}{1.15} - (1-0.1)} = -3.28$, this negative value means no matter how we speed up this part, we can't reach the goal
   Branch $S = \frac{0.2}{\frac{1}{1.15} - (1-0.2)} = 2.875$, so we need to have a 2.875X speed up on our branch instruction, which means branch instruction needs to be done in $3/2.875 \approx 1$ cycles.

2. Assume this processor supports parallelization, and if I want to run a program on it: this program contains the same operations with Problem 1.4 . The processor can run at most 1000 threads parallely. Each thread is only used for executing ALU operations (including ALU operations and combined ALU-Load). How much performance improvement can I achieve? (2 points)

   $$P_{ALU} = \frac{0.35 * 1}{0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1} \approx 0.194 \tag{3}$$
   $$P_{ALU-LOAD} = \frac{0.15 * 1}{0.35 * 1 + 0.05 * 2 + 0.1 * 2 + 0.2 * 5 + 0.15 * 1} \approx 0.083 \tag{4}$$
   Together we have $P = 0.194 + 0.083 = 0.277$
   From
   $$S_{total-parallel} = \frac{1}{(1-P) + \frac{P}{N}} \tag{5}$$
   Then we have
   $$S_{total} = \frac{1}{(1-0.277) + \frac{0.277}{1000}} \approx 1.383 \tag{6}$$
   Thus, we can achieve 1.383X performance improvement

3. Unfortunately, my processor is I/O bound, as it consumes 80% of the total time to wait for data. What is the highest speedup of such a processor? (2 points)

   Our instructions under such situation will only take $1 - 0.8 = 0.2$ of the runtime. Assume we have infinite speed up over this portion.
   $$S_{total} = \frac{1}{(1-0.2) + \frac{0.2}{\infty}} = 1.25 \tag{7}$$
   Our highest speedup at this time would be 1.25X

2.3 2 / 2

✓ - 0 pts Correct

4. What are the different ways of measuring CPI or execution time? Explain briefly. What are the potential benefits of measuring CPI/execution time? (3 points)

   Instead of calculating based on the parameters, we can do benchmark to measure execution time. We can run a program and measure actual time a program uses, thus determine the CPI accordingly.

   Running practical program and measure it on different machines can helps determine the performance difference between them. Different workload also perform differently on different machines so it helps people to pick the right machine, saving time for their specific tasks. Also, micro-benchmark that target specific but common patterns helps chip developer to identify which direction they could focus on to reduce the general execution time or reduce energy footprint on some common cases.

2.4 3 / 3

✓ **- 0 pts** Correct