6 Performance Evaluation [70 points]

Some fellow students are working on a project called *AwesomeMEM*, where their goal is to optimize the memory hierarchy (caches and DRAM) to enhance the performance of a multi-core system.

They evaluate two system configurations. First, the *Baseline* configuration constitutes a system with two processors, a last-level cache (LLC), and DRAM as main memory. Second, the *AwesomeMEM* configuration builds on top of the *Baseline* configuration by employing optimizations to the memory hierarchy.

The students evaluate the performance benefits of *AwesomeMEM* in simulation as follows:

1. First, they collect performance metrics of four single-threaded applications (App_1, App_2, App_3, App_4) running in isolation in the *Baseline* configuration.
2. Second, they create two-application mixes to perform a multi-program simulation, where two applications run concurrently in the *Baseline* configuration, each in a dedicated processor. They evaluate two application mixes: Mix_1 (consisting of App_1 and App_2); and Mix_2 (consisting of App_3 and App_4).
3. Third, they use the same two-application mixes as in the second step to perform a multi-program simulation, where two applications run *concurrently* in the *AwesomeMEM* configuration, each in a dedicated processor.

Table 1 summarizes the performance metrics the students collected for each step.

Table 1: Performance metrics the students collected.

Execution Mode	Application Mix	Configuration	Application	Executed Instructions	Executed Cycles	LLC Miss Rate (%)	Branch Misprediction Rate (%)	DRAM Bank Conflict Rate (%)
Single-threaded	N/A	Baseline	App_1	100,000	40,000	26%	1%	42%
			App_2	100,000	800,000	99%	1%	94%
			App_3	100,000	500,000	52%	1%	89%
			App_4	100,000	20,000	10%	1%	14%
Multi-programmed	Mix_1	Baseline	App_1	100,000	200,000	99%	1%	97%
			App_2	100,000	900,000			
		AwesomeMEM	App_1	80,000	100,000	65%	1%	55%
			App_2	80,000	400,000			
	Mix_2	Baseline	App_3	100,000	600,000	60%	1%	90%
			App_4	100,000	20,000			
		AwesomeMEM	App_3	80,000	400,000	50%	1%	45%
			App_4	100,000	20,000			

Answer the following questions based on the performance metrics the students collected.

(a) [20 points] What is the Instructions Per Cycle (IPC) of each of the four applications when the application is executed *in isolation* in the *Baseline* configuration? Show your work.

App_1:

$$IPC = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{40{,}000} = \mathbf{2.5}$$

App_2:

$$IPC = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{800{,}000} = \mathbf{0.125}$$

App_3:

$$IPC = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{500{,}000} = \mathbf{0.2}$$

App_4:

$$IPC = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{20{,}000} = \mathbf{5}$$

To measure the system throughput of a multi-core system, the students use the *weighted speedup* metric, which sums the Instructions Per Cycle (IPC) slowdown experienced by each application compared to when it is run alone (IPC_i^{alone}) for the same number of instructions as it executed in the multi-programmed application mix (IPC_i^{shared}):

$$\text{System Throughput} = \text{Weighted Speedup} = \sum_i \frac{IPC_i^{shared}}{IPC_i^{alone}}$$

(b) [20 points] What is the IPC_i^{shared}, $i \in \{App_1, App_2, App_3, App_4\}$, of each of the four applications when they are executed *concurrently* in accordance with their multi-programmed application mix in the *Baseline* and *AwesomeMEM* configurations? Show your work.

App_1:

Baseline:

For the *Baseline*: $IPC_{App_1}^{shared} = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{200{,}000} = \mathbf{0.5}$

AwesomeMEM:

For *AwesomeMEM*: $IPC_{App_1}^{shared} = \frac{\#instructions}{\#cycles} = \frac{80{,}000}{100{,}000} = \mathbf{0.8}$

App_2:

Baseline:

For the *Baseline*: $IPC_{App_2}^{shared} = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{900{,}000} = \mathbf{0.11}$

AwesomeMEM:

For *AwesomeMEM*: $IPC_{App_2}^{shared} = \frac{\#instructions}{\#cycles} = \frac{80{,}000}{400{,}000} = \mathbf{0.2}$

App_3:

Baseline:

For the *Baseline*: $IPC_{App_3}^{shared} = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{600{,}000} = \mathbf{0.16}$

AwesomeMEM:

For *AwesomeMEM*: $IPC_{App_3}^{shared} = \frac{\#instructions}{\#cycles} = \frac{80{,}000}{400{,}000} = \mathbf{0.2}$

App_4:

Baseline:

For the *Baseline*: $IPC_{App_4}^{shared} = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{20{,}000} = \mathbf{5}$

AwesomeMEM:

For *AwesomeMEM*: $IPC_{App_4}^{shared} = \frac{\#instructions}{\#cycles} = \frac{100{,}000}{20{,}000} = \mathbf{5}$

(c) [10 points] What is the *weighted speedup* of each of the two application mixes when it is executed in the *Baseline* configuration? Show your work.

Mix_1:

$$\text{Weighted Speedup} = \sum_i \frac{IPC_i^{shared}}{IPC_i^{alone}} = \frac{IPC_{App_1}^{shared}}{IPC_{App_1}^{alone}} + \frac{IPC_{App_2}^{shared}}{IPC_{App_2}^{alone}}$$

$$\text{Weighted Speedup} = \frac{0.5}{2.5} + \frac{0.11}{0.125} = \mathbf{1.08}$$

Mix_2:

$$\text{Weighted Speedup} = \sum_i \frac{IPC_i^{shared}}{IPC_i^{alone}} = \frac{IPC_{App3}^{shared}}{IPC_{App3}^{alone}} + \frac{IPC_{App4}^{shared}}{IPC_{App4}^{alone}}$$

$$\text{Weighted Speedup} = \frac{0.16}{0.2} + \frac{5}{5} = \mathbf{1.8}$$

(d) [10 points] What is the *weighted speedup* of each of the two application mixes when it is executed in the *AwesomeMEM* configuration? Show your work.

Mix_1:

$$\text{Weighted Speedup} = \sum_i \frac{IPC_i^{shared}}{IPC_i^{alone}} = \frac{IPC_{App1}^{shared}}{IPC_{App1}^{alone}} + \frac{IPC_{App2}^{shared}}{IPC_{App2}^{alone}}$$

$$\text{Weighted Speedup} = \frac{0.8}{2.5} + \frac{0.2}{0.125} = \mathbf{1.92}$$

Mix_2:

$$\text{Weighted Speedup} = \sum_i \frac{IPC_i^{shared}}{IPC_i^{alone}} = \frac{IPC_{App3}^{shared}}{IPC_{App3}^{alone}} + \frac{IPC_{App4}^{shared}}{IPC_{App4}^{alone}}$$

$$\text{Weighted Speedup} = \frac{0.2}{0.2} + \frac{5}{5} = \mathbf{2}$$

The students do *not* want to reveal the primary technique behind *AwesomeMEM*. When asked, they provided the following list of architectural techniques and told you that some of them could be the reason behind *AwesomeMEM*'s system throughput improvement:

(i) *AwesomeMEM* increases the LLC capacity by 2× that of the *Baseline*.

(ii) *AwesomeMEM* randomizes main memory requests to reduce DRAM bank conflicts.

(iii) *AwesomeMEM* employs a perfect branch predictor that always predicts a branch's direction correctly.

(iv) *AwesomeMEM* employs an efficient hardware prefetcher.

(e) [10 points] Which of the above explanations **cannot** possible be a reason for *AwesomeMEM*'s higher performance over the *Baseline*? Explain your reasoning based on the data in Table 1.

Option (iii) cannot possible be a reason for *AwesomeMEM*'s performance improvement compared to the *Baseline*.

Explanation:
(iii) is *not* possible since the branch misprediction rate n the *AwesomeMEM* configuration is the same for Mix_1 and Mix_2 compared to the *Baseline* configuration.

(i) is possible since LLC miss rate in the *AwesomeMEM* configuration drops for Mix_1 and Mix_2 compared to the *Baseline* configuration.

(ii) is possible since bank conflicts in the *AwesomeMEM* configuration drops for Mix_1 and Mix_2 compared to the *Baseline* configuration.

(iv) is possible since LLC miss rate in the *AwesomeMEM* configuration drops for Mix_1 and Mix_2 compared to the *Baseline* configuration.