Amirkabir University of Technology
(Tehran Polytechnic)

Question 1: In the year 2000, Intel projected that CPU speeds could reach 10 GHz by 2005, a milestone that was never achieved. Address the following queries related to this topic:

- What challenges would a 10 GHz CPU face, and why were these challenges not significant when transitioning from the earliest ~500 KHz chips to the current ~5 GHz chips? Consider the concept of Dennard scaling in your answer.
- Despite the inability to significantly increase CPU clock speeds beyond a certain point, how have CPU designers continued to enhance CPU performance?

Question 2: Flynn's taxonomy categorizes parallel computer architectures into four distinct types. Please explain these four parallel architectures. Then, identify which parallel architecture(s) each of the following computer systems utilizes:

- GPU (Graphics Processing Unit)
- Multi-core pipelined x86-64 CPU with support for the AVX2 extension
- Single-core in-order 8-bit microprocessor

Question 3: We have a new Intel core i9 14900K which can roughly do 156 GFLOP/s and it's paired with a dual channel DDR5 RAM kit that has a half-duplex bandwidth of 96 GB/s.

a) Draw the roofline diagram for this processor.
b) Assume that this CPU has no cache but has a limited number of registers, analyze which component might limit the performance of the following code snippet, considering the operations inside the for loop body. Assume comparison and incrementation, together, are equivalent to a single floating point operation. Discuss where this limitation would appear on the roofline diagram provided in part (a).

```
1  int main()
2  {
3     const int rows = 256;
4     const int cols = 256;
5     fp16 mat[rows][cols]; // Set to some random values
6     fp16 acc[rows][cols] = {0};
7     fp16 mult;
8     for (int i = 0; i < rows; i++) {
9        for (int j = 0; j < rows; j++) {
10          mult = 1.0 / (abs(i - j) + 1.0);
```

```
11          for (int k = 0; k < cols; k++)
12              acc[i][k] += mult * mat[j][k] * mat[j][k];
13      }
14  }
15  return 0;
16 }
```

c) Now let's assume through some magical API we have allocated exactly 64KB of L1 data cache to this program. This cache block has an LRU eviction policy and negligible access time compared to main memory. Repeat part (b) with this assumption.

d) Compare power consumption of parts (b) and (c) to the point that there's no bottleneck in the system (without any cache usage).

Question 4: Consider two implementations of a single step in a naive weighted average calculation, detailed below. For each implementation, analyze the number of operations required under different processing architectures and conditions.

$\forall\, i \in [0,3]: p_i = P(\text{unfair\_rand() \% 4} == i)$

Implementation 1	Operations	Implementation 2	Operations
load v0, v1, v2, v3, acc, count	6	load v0, v1, v2, v3, acc, count	6
rnd = unfair_rand() % 4	2	rnd = unfair_rand() % 4	2
if (rnd == 0)	2	s0 = rnd == 0	1
acc += v0	1 + 1*	s1 = rnd == 1	1
else if (rnd == 1)	2	s2 = rnd == 2	1
acc += v1	1 + 1	s3 = rnd == 3	1
else if (rnd == 2)	2	acc += s0 * v0 + s1 * v1 +	
acc += v2	1 + 1	s2 * v2 + s3 * v3	8
else	0	count++	1
acc += v3	1	store acc, count	2
count++	1		
store acc, count	2		

* The additional operation in Implementation 1 is due to the control jump required to bypass 'else' conditions.

a) Calculate the average number of operations for N calls to this step for each implementation on an M_1 core SISD system.

b) Calculate the average number of operations for N calls to this step for each implementation in an M_2 core SIMD system.

c) Assuming that unfair_rand() is actually fair ($p_0 = p_1 = p_2 = p_3$); what is the chance of implementation 1 using less operations than implementation 2 in an SIMD system with 32 threads?

d) For a pipelined SISD system as described in question 3, which implementation would be faster? (Variables are all of type fp32)

Question 5: With an understanding of NUMA (Non-Uniform Memory Access) and UMA (Uniform Memory Access) architectures, delve into search on ccNUMA (cache-coherent Non-Uniform Memory Access) and explain its principles. Consider a computer CPU that incorporates at least three levels of cache, some of which are shared among different cores, potentially qualifying it as a ccNUMA system. Analyze the impact on program performance when a variable, shared between two cores and stored in the core-exclusive low-level cache of both, is continually modified by these cores.

Question 6: Explain Amdahl's Law and Gustafson's Law, and address the following questions:

- What limitation of Amdahl's Law is addressed by Gustafson's Law?
- Assuming the cost of the serial portion of the algorithms remains constant, compare the scalability of an O(n) algorithm to that of an O(n^3) algorithm.