CSE Qualifying Exam, Fall 2025

CSCE 513-Computer Architecture

Question 1. Instruction Dependencies and Superscalar Processor Performance

1. Given the below latencies for the dependant Floating-Point (FP) and Integer operations in a processor:

Instruction Producing Results	Instruction Using Results	Latency in clock cycles
FP Multiply	FP Add	5
FP Add	FP Multiply	2
FP Add	FP Store	2
Load FP	FP Multiply	3
Load FP	FP Add	1
Load integer	Integer ALU	1
Integer ALU	Branch	0

(a) Compute the average CPI of a scalar pipeline executing this loop without optimizations (no scheduling or unrolling).

```
Loop: l.d    $f2,0($s0)      #FP Load
      mul.d  $f4,$f2,$f6     #FP multiply
      add.d  $f8,$f4,$f2     #FP add
      s.d    $f8,0($s0)      #FP Store
      addi   $s0,s0,-8       #integer add
      bne    $s0,$zero,Loop  #Branch
```

(b) For a **4-way superscalar processor** with register renaming and scheduling allowed, what is the minimum unrolling factor needed to achieve an **IPC greater than or equal to 2 (IPC >= 2)**?

Question 2. Memory Hierarchy

A processor runs at **3 GHz** with a base CPI of **1.0**. The instruction mix has **30% loads/stores**. The memory hierarchy is:

- **L1 cache:** hit time = 1 cycle, miss rate = 3%
- **L2 cache:** hit time = 6 cycles, miss rate = 20%
- **Main memory:** 120 cycles access

An architect proposes two possible improvements:

1. **Add an L3 cache** with hit time = 14 cycles, miss rate = 5%
2. **Increase frequency** to 3.6 GHz, but base CPI rises to **1.33**

(a) Compute the effective CPI of the baseline (no L3).

(b) Compute the effective CPI and performance speedup with the L3 cache.

(c) Compare the L3 option with the higher-frequency option. Which provides better performance? Justify with calculations.

Question 3. Performance

A scientific application executes the following kernel:

```
for (int i = 0; i < n; i++) {
    energy[i] += coeff[i] * temp[i];
    for (int j = 0; j < n; j++)
        energy[i] = alpha * (temp[j] / (dist[i][j] + beta));
}
```

System characteristics:

- Main memory bandwidth: **16 GB/s**
- Peak computational throughput: **8 GFlops/s**
- No cache

Arrays `energy`, `coeff`, `temp` are single-precision floats; `dist` is an `n×n` float matrix. Constants `alpha` and `beta` are scalars.

(a) For **n = 4**, determine whether the system is compute-bound or memory-bound. Show your calculations.

(b) Engineers propose optimizations:

- Increase memory bandwidth to **22 GB/s**
- Add a cache large enough to store the entire **`temp`** array

What speedup is achieved? Does the bottleneck shift from memory-bound to compute-bound (or vice versa)? Explain.

Fall 2025 Qualifying Exam—Algorithms (750)

Question 1 (A recurrence). Find tight asymptotic bounds on any positive function $T(n)$ satisfying the following recurrence for all sufficiently large n:

$$T(n) = \sqrt{n}\,T(\sqrt{n}) + n\ .$$

You may assume that n is a power of 2 and that any implicit floors and ceilings are of no consequence.

Prove your upper bound by the substitution method. (You do not need to prove the matching lower bound, but if your upper bound is not tight, you will not receive credit even for a correct substitution proof.)

Question 2 (The SUBSET-SUM problem). In the SUBSET-SUM problem, you are given a list $L := \langle a_1, \ldots, a_n \rangle$ of n positive integers, where $n \geq 0$, and an integer $t \geq 0$ (the *target*), and your task is to determine whether some sublist of L sums to t. That is, whether there exists $S \subseteq \{1, \ldots, n\}$ such that

$$\sum_{i \in S} a_i = t\ .$$

If such an S exists, the answer is **yes**, and otherwise the answer is **no**.

For example, if $L = \langle 2, 5, 13 \rangle$, then the answer is **yes** if $t \in \{0, 2, 5, 7, 13, 15, 18, 20\}$ and **no** otherwise. (The empty sum is 0 by convention.)

Describe an algorithm that takes as input a list $L := \langle a_1, \ldots, a_n \rangle$ and a target t as above, and returns the correct answer (**yes** or **no**) of the SUBSET-SUM problem for this instance. Your description should include enough detail so that someone who did well in CSCE 750 can implement it without specific knowledge of the problem. The output only needs to be **yes** or **no**; you are not required to output an actual sublist.

For full credit, your algorithm must run in time $O(tn)$. You may assume that all arithmetic and comparison operations on integers take $O(1)$ time each.

[Hint: Use dynamic programming with a table $T[0 \ldots n, 0 \ldots t]$ of Booleans. What should $T[i, j]$ contain for each i and j?]

Question 3 (MST on a special graph). You are given a connected (undirected) graph G with weight function $w : G.E \to \mathbb{R}$ such that $w(e)$ is either 1 or 2 for all $e \in G.E$.

Explain how to find a minimum spanning tree for G in *linear time*. Linear time means time $O(|G.V| + |G.E|)$ with the usual adjacency-list representation of G. Your explanation can be high-level, but should be clear and precise enough so that someone who did well in CSCE 750 can implement it.

[Hint: The priority queue operations used by Prim's algorithm for *general* weighted graphs make the algorithm run too long—longer than linear time.]