1) (2 Pts): State Amdahl's Law in words

Answer: Amdahl's Law states that the overall speedup of a system is limited by the portion of the system that cannot be improved. Specifically, the maximum speedup achievable is determined by the fraction of the execution time that is affected by the enhancement. The law shows that focusing on improving the most time-consuming parts of a system yields the greatest overall performance gains.

Mathematical form: Speedup = 1 / [(1 - f) + f/s] where f = fraction of execution time affected by enhancement, s = speedup of the enhanced portion

2) (4 Pts): I/O System Speedup

Given:

- 65% of time spent on I/O
- Disk upgrade provides 50% greater throughput (1.5x speedup)

Solution: Using Amdahl's Law:

- f = 0.65 (fraction using I/O)
- s = 1.5 (I/O speedup factor)

Speedup = 1 / [(1 - 0.65) + 0.65/1.5] Speedup = 1 / [0.35 + 0.433] Speedup = 1 / 0.783 Speedup = 1.28x (28% improvement)

3) (8 Pts): 30% Faster Servers

Given:

- Need 30% overall speedup (1.3x)
- 70% CPU time, 30% I/O time

Part a: CPU Speedup Required

Using Amdahl's Law for CPU enhancement: 1.3 = 1 / [(1 - 0.7) + 0.7/s_cpu] 1.3 = 1 / [0.3 + 0.7/s_cpu] 0.3 + 0.7/s_cpu = 1/1.3 = 0.769 0.7/s_cpu = 0.769 - 0.3 = 0.469 s_cpu = 0.7/0.469 = 1.49

CPU needs to be 1.49x faster (49% improvement)

Part b: Disk Speedup Required

Using Amdahl's Law for I/O enhancement: 1.3 = 1 / [(1 - 0.3) + 0.3/s_disk] 1.3 = 1 / [0.7 + 0.3/s_disk] 0.7 + 0.3/s_disk = 0.769 0.3/s_disk = 0.069 s_disk = 0.3/0.069 = 4.35

Disk needs to be 4.35x faster (335% improvement)

4) (4 Pts): Musical Instrument Processor

Given:

- Need 12% performance boost (1.12x speedup)
- Processor accounts for 25% of workload

Solution: Using Amdahl's Law: 1.12 = 1 / [(1 - 0.25) + 0.25/s_proc] 1.12 = 1 / [0.75 + 0.25/s_proc] 0.75 + 0.25/s_proc = 1/1.12 = 0.893 0.25/s_proc = 0.893 - 0.75 = 0.143 s_proc = 0.25/0.143 = 1.75

New processor needs to be 1.75x faster (75% improvement)

5) (16 Pts): Cost-Performance Analysis

Given:

- 60% CPU, 40% disk activity
- Disk upgrade: 2.5x speedup for $8,000
- CPU upgrade: 1.4x speedup for $5,000

Part a: Best Performance per Dollar

Disk upgrade speedup: Speedup = 1 / [0.6 + 0.4/2.5] = 1 / [0.6 + 0.16] = 1.32x Cost per 1% improvement = $8,000 / 32% = $250 per 1%

CPU upgrade speedup: Speedup = 1 / [0.4 + 0.6/1.4] = 1 / [0.4 + 0.429] = 1.21x Cost per 1% improvement = $5,000 / 21% = $238 per 1%

Answer: CPU upgrade provides better performance per dollar

Part b: Best Performance (Money No Object)

Disk upgrade provides 1.32x speedup vs CPU's 1.21x speedup. Answer: Choose disk upgrade for maximum performance

Part c: Break-even Point

For equal cost per 1% improvement: Disk: $8,000 / 32% = $250 per 1% To match this, CPU price should be: 21% × $250 = $5,250

Answer: CPU should cost $5,250 for break-even

Part d: Changed Activity (55% CPU, 45% disk)

Disk upgrade speedup: Speedup = 1 / [0.55 + 0.45/2.5] = 1 / [0.55 + 0.18] = 1.37x

CPU upgrade speedup: Speedup = 1 / [0.45 + 0.55/1.4] = 1 / [0.45 + 0.393] = 1.19x

Answer: Disk upgrade still provides better absolute performance, and CPU still provides better cost-performance ratio

6) (8 Pts): Disk Drive Analysis #1

Given:

- 6 surfaces, 16,383 tracks/surface, 63 sectors/track, 512 bytes/sector
- 8.5ms seek time, 7,200 rpm

Part a: Capacity

Capacity = 6 × 16,383 × 63 × 512 bytes Capacity = 3,145,521,216 bytes Capacity = 3.15 GB

Part b: Access Time

Rotational latency = (60 sec / 7,200 rpm) / 2 = 4.17 ms Access time = Seek time + Rotational latency Access time = 8.5 + 4.17 = 12.67 ms

7) (12 Pts): Disk Drive Analysis #2

Given:

- 6 surfaces, 953 tracks/surface, 256 sectors/track, 512 bytes/sector
- 6.5ms seek time, 5,400 rpm

Part a: Capacity

Capacity = 6 × 953 × 256 × 512 bytes = 754,974,720 bytes = 755 MB

Part b: Access Time

Rotational latency = (60 sec / 5,400 rpm) / 2 = 5.56 ms Access time = 6.5 + 5.56 = 12.06 ms

Part c: Speed Comparison

6 disk: 12.67 ms access time 7 disk: 12.06 ms access time 7 disk is slightly faster (by 0.61 ms) but has much smaller capacity

8) (4 Pts): Instruction Encoding

Given:

- 32-bit instructions, 12-bit addresses
- 250 two-address instructions

Solution: Two-address instruction format: [opcode][addr1][addr2] Bits needed: opcode + 12 + 12 = opcode + 24 Available for opcode: 32 - 24 = 8 bits Maximum opcodes with 8 bits: 2^8 = 256 Used for two-address: 250 Remaining: 256 - 250 = 6

One-address instruction format: [opcode][addr1] For remaining opcodes, we can use longer opcode fields. Since we have 6 remaining opcode values, and each can represent multiple one-address instructions:

Maximum one-address instructions = 6 × 2^(32-12-8) = 6 × 2^12 = 24,576

9) (4 Pts): Addressing Modes

Given: Load 100, R1 = 0x300, Memory values as shown

Mode Value loaded into AC

Mode	Value loaded into AC
Immediate	100 (0x64)
Direct	0x600 (value at address 100)
Indirect	0x300 (value at address pointed to by 100)
Indexed	0x500 (value at address 100 + R1 = 400)

10) (6 Pts): Expanding Opcodes

Given: 16-bit instructions, 32 registers (5 bits each)

- 60 instructions with two register operands
- 30 instructions with one register operand
- 3 instructions with one 10-bit address
- 26 instructions with zero operands

Analysis: Total instructions needed: 60 + 30 + 3 + 26 = 119

Format 1 (2 registers): [6-bit opcode][5-bit reg][5-bit reg] = 16 bits Format 2 (1 register): [11-bit opcode][5-bit reg] = 16 bits
Format 3 (10-bit addr): [6-bit opcode][10-bit addr] = 16 bits Format 4 (0 operands): [16-bit opcode] = 16 bits

Encoding scheme:

- Opcodes 000000-111011 (60 codes): Two register format
- Opcodes 111100 xxxxx (32 codes): One register format (uses 30)
- Opcodes 111101 (1 code): 10-bit address format (uses 3 sub-codes)
- Opcodes 111110-111111: Zero operand format (can encode 26)

Answer: Yes, this encoding is possible

11) (8 Pts): Pipeline Analysis #1

Given:

- Non-pipelined: 200ns per task
- Pipeline: 5 segments, 40ns clock cycle

Part a: Speedup for 200 tasks

Non-pipelined time: 200 × 200ns = 40,000ns Pipeline time: (5-1) × 40ns + 200 × 40ns = 160ns + 8,000ns = 8,160ns Speedup = 40,000 / 8,160 = 4.9x

Part b: Theoretical Maximum Speedup

Maximum speedup = 200ns / 40ns = 5x

12) (8 Pts): Pipeline Analysis #2

Given:

- Non-pipelined: 100ns per task
- Pipeline: 5 stages, 20ns clock cycle

Part a: Speedup for 100 tasks

Non-pipelined time: 100 × 100ns = 10,000ns Pipeline time: (5-1) × 20ns + 100 × 20ns = 80ns + 2,000ns = 2,080ns Speedup = 10,000 / 2,080 = 4.8x

Part b: Theoretical Maximum Speedup

Maximum speedup = 100ns / 20ns = 5x

13) (8 Pts): Memory and Instruction Format

Given:

- 24 bits per word
- 150 different operations
- One address per instruction

Part a: Opcode Bits

Bits needed = $\lceil \log_2(150) \rceil$ = 8 bits

Part b: Address Bits

Address bits = 24 - 8 = 16 bits

Part c: Maximum Memory Size

Maximum memory = 2^16 = 65,536 words

Part d: Largest Unsigned Binary Number

Largest number = 2^24 - 1 = 16,777,215

14) (8 Pts): Computer Instruction Format

Given:

- 256K words of 32 bits each
- 7 addressing modes, 60 registers
- 32-bit instructions

Part a: Mode Field Size

Mode field = $\lceil \log_2(7) \rceil$ = 3 bits

Part b: Register Field Size

Register field = $\lceil \log_2(60) \rceil$ = 6 bits

Part c: Address Field Size

Address field = $\lceil \log_2(256K) \rceil$ = 18 bits

Part d: Opcode Field Size

Opcode field = 32 - 3 - 6 - 18 = 5 bits