University of Jaffna, Sri Lanka

Bachelor of Science Degree Examination in Computer Science, Level 3S - 2021

CSC309S3: High Performance Computing

- Answer **all** four questions. (This paper has 4 questions on 5 pages.)
- Time allowed: **Three Hours.**
- Start each question in a fresh page.
- Allocate your time wisely, the point value of each part shown in square brackets.
- At the bottom of the front page of your answer book, write the question numbers in the order you answered.

1. (a) In the execution of a particular computational problem on a system with a specific hardware configuration, the cache-hit ratio can be influenced by various factors.

Provide an explanation for two such factors, detailing their effects on the cache-hit ratio. [6%]

(b) Imagine a scenario where a computation involving the calculation of dot product of two vectors is underway on a computer system equipped with a 4GHz processor. In this system, the DRAM latency is measured at one hundred cycles, while the cache latency is one cycle. Additionally, the cache line size is set to accommodate eight words.

i. ***State*** the relationship between the speed of the processor and the duration of a single cycle. [1...

ii. ***Calculate*** the latencies to both DRAM and cache in seconds. [3%]

iii. Taking into account the cache line size, ***determine*** the anticipated cache-hit ratio for this specific dot product problem. [3%]

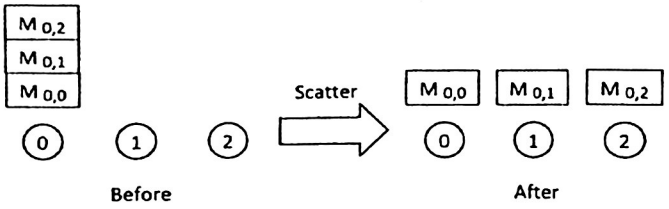
iv. Based on the cache-hit ratio you derived in the previous part, ***calculate*** the *average per word access time* by the processor. [3%]

v. ***Utilize*** the per-word access time to determine the overall computation rate. [3%]

(c) In particular computational scenarios, such as matrix-matrix multiplication, increasing the cache line size may not lead to improved performance, primarily due to the way data is organized in memory, known as spatial data locality. ***Explain*** your reasoning. [6%

[25% marks

(a) Scatter, also known as one-to-all personalized communication, distributes unique messages from a source node to all other nodes within the given domain.

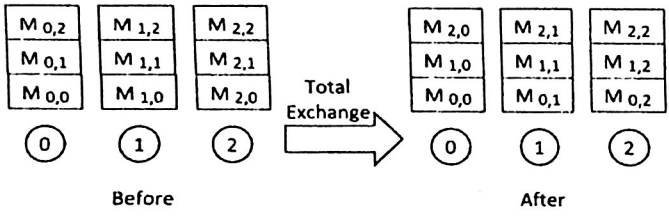


In this context, for a message $M_{i,j}$, i represents the source node, and j signifies the destination.

i. With the help of illustration, ***show*** how the recursive doubling method can be employed to perform the scatter operation in an eight-node hypercube network, with Node 0 as the source.

ii. ***Prove*** that the total time required for the scatter operation, you described in part (a), is equal to: $t_s \log_2 p + m\, t_w\, (p-1)$.

(b) Consider a scenario in which all nodes within a domain have unique messages to scatter to all other nodes, as illustrated below:



To efficiently accomplish this "total exchange" task on a p-node hypercube, the optimal approach involves each node directly transmitting its message to the intended recipient using e-cube routing.

i. ***Deduce*** the number of communication steps required to complete this task.

ii. ***Clarify*** how source nodes can identify the destinations during each communication step.

iii. With a suitable example, ***explain*** how the e-cube routing identifies the path from sender to receiver.

iv. ***Write*** a detailed pseudocode for executing the "total exchange" task as described above.

v. ***Calculate*** the overall time required to complete this task.

[25% ma

3. *Cost-optimality* and *scalability* are two intertwined considerations in the design and deployment of computing systems.

(a) i. ***Define*** what is *cost-optimality* in parallel systems, and ***deduce*** that a cost-optimal parallel method has an efficiency of $\Theta(1)$. [3%]

ii. Consider a problem with serial runtime Cn^2 and the parallel runtime $Dn+Cn^2/p$, where n is the problem size, p is the number of processing elements, whereas C and D are constants.

(α) ***Determine*** the speedup and efficiency. [4%]

(β) ***Find*** a relationship between n and p such that the system is cost-optimal. [4%]

(b) Suppose you are conducting an investigation on a complex computational problem. Initially, you execute it on a single processor and record an execution time of 1012 seconds. You then parallelize the code and run it with different numbers of processors, recording the corresponding execution times (T) as follows:

Number of Processors	Execution Time in seconds
1	1012
2	756
4	407
6	221
8	131

i. ***Calculate*** the speedup for each configuration, and ***plot*** a speedup vs. number of processors graph. [4%]

ii. ***Identify*** the abnormal behaviour of this system. [2%]

iii. ***Explain*** two potential factors that could contribute to this behaviour. [4%]

iv. ***Discuss*** whether the system demonstrates scalability and ***provide*** observations that support your argument. [4%]

[25% marks]

4. **Answer either question 4 (a) or question 4 (b).**

(a) OpenMP adapts to UMA and NUMA architectures, enabling efficient shared-memory parallelism on diverse multicore systems.

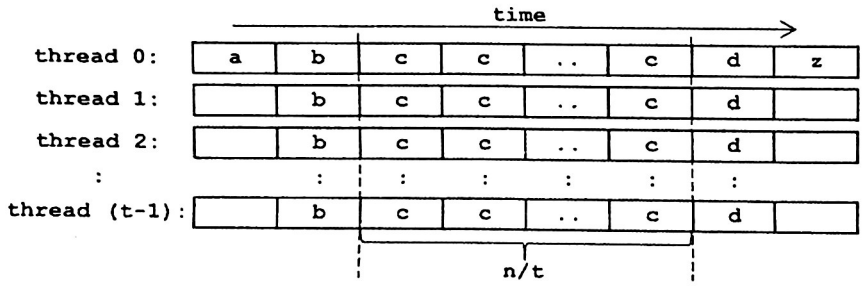
i. ***Provide*** illustrations and detailed descriptions for UMA and NUMA architectures.

ii. Consider the code given below with three function calls: `a()`, `c()`, and `z()`.

```
a();
for (int i = 0; i < n; i++) {
   c();
}
z();
```

(α) ***Modify*** the provided code using the "parallel for" OpenMP directive to enable parallel execution of the function `c()`.

(β) Consider a scenario where we need to implement thread-specific preprocessing using `b()` and post-processing with `d()`, as illustrated in the execution timeline below:



To accommodate this requirement, ***make*** necessary modifications to the code you wrote in part (a):ii.α.

(γ) Suppose the function `d()` involves a shared variable, and as a result, it is crucial to ensure that only one thread can execute the function `d()` at any given moment. To achieve this, ***modify*** the code from part (a).ii.β by using the "critical" directive of OpenMP.

(δ) You have been given a task to compute the product of even numbers in an array. Consider the following code and ***complete*** within `#pragma omp parallel` to ensure synchronized parallel processing and prevent race conditions:

```
int MultiplyEvenNumbers(int* array, int size) {
      int product = 1;
      #pragma omp parallel
      {
            // Your task is to complete this section
      }
      return product;
}
```

[25% ma

[Question 4 continues ...]

(b) Message passing via MPI is widely used with *SPMD architectures*.

i. With a help of illustration, clearly ***distinguish*** the following three parallel architectures: *SIMD*, *MIMD*, and *SPMD*. [5%]

ii. Clearly ***describe*** the following MPI functions: [6%]

(α) `int MPI_Bcast(void *buf, int count, MPI_Datatype datatype, int source, MPI_Comm comm)`

(β) `int MPI_Reduce(void *sendbuf, void *recvbuf, int count, MPI_Datatype datatype, MPI_Op op, int target, MPI_Comm comm)`

(γ) int MPI_Gather(void *sendbuf, int sendcount, MPI_Datatype sendtype, void *recvbuf, int recvcount, MPI_Datatype recvtype, int target, MPI_Comm comm)

iii. Suppose you have a large one dimensional integer array A of size n, where n is power of 2. The task is to calculate and print the **product** after adding each element with a value v as follows:

$$product = (A[0] + v) * (A[1] + v) * \cdots * (A[n-1] + v)$$

To achieve this, complete the following function to perform the operations described below the code:

```
void parallelProduct(int rank, int comsize, int* A, int v) {
      // Declare and initialise necessary variables

      // Send the value v from rank 0 process to all processes

      // Distribute equally-sized chunks of array A to all processes.

      // Perform the calculation locally within each process

      // Get the local products into rank 0 process and print
}
```

(α) ***Send*** the value v from rank 0 process to all other processes. [2%]

(β) ***Divide*** the array A into equal chunks and distribute to all processes. [5%]

(γ) ***Add*** the value v to each element of a local array, and calculate the product of all modified elements within each process using the formula shown above. [2%]

(δ) ***Get*** the local products into rank 0 process from all processes to calculate and print the final product of all modified elements. [5%]

[25% marks]

End