EE/CS 451 Midterm 1

Fall 2019

Instructor: Viktor Prasanna

Friday, 10/11/2019

3:30 – 5:30pm, GFS 118

Problem #	Topic	Points	Score
1	Definitions	20	
2	Memory System Performance	15	
3	Shared Memory Programming	10	
4	Message Passing and Shared Memory Programming	20	
5	Interconnection Networks	10	
6	Program Mapping	15	
7	Analytical Modeling	10	
Total		100	

Student Name:

Student USC-ID:

Name Initials: _____

Problem 1 ($10 \times 2 = 20$ points)

Define/explain the following terms:

1. Superscalar processor

2. Omega network

3. PRAM

4. Diameter of a network

5. Store and forward routing

6. Race condition

7. Instruction-level parallelism

8. Hypercube network

9. Shared-address-space programming model

10. Cut through routing

Name Initials: _____

Problem 2 (15 points)
Memory System Performance:
Consider a memory system which has 100-cycle latency to DRAM, no on-chip cache and is connected to a processor that operates at 1 GHz. Assume the memory can stream data in each processor cycle. The processor has two floating point multiply-add units. Each multiply-add unit is capable of executing 1 multiplication and 1 addition per processor cycle. Thus, the processor can execute four floating point operations in each processor cycle.

1. What is the peak floating point performance of the processor? (3 points)

2. Consider the problem of executing the following code on such a platform.

```
result = 0;
for (i = 0; i < dim; i++)
  result += A[i] × B[i];
```

(a). Suppose the processor-memory bus can support one word (one element of A or B) in a cycle. What is the best case achievable performance (in FLOPS)? State any assumptions you may make. (3 points)

(b). Suppose the processor-memory bus can support **four** words (four elements of A or B or any combination) in a cycle. What is the best case achievable performance (in FLOPS)? State any assumptions you may make. (3 points)

3. Redo 2(a) and 2(b) if only one floating point multiply-add unit is available. (4 points)

4. For the scenario in 2(a), what is the worst case achievable performance? (2 points)

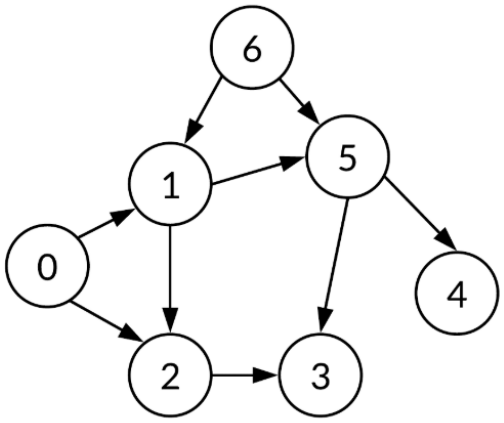
Name Initials: _____

Problem 3 (10 points)

Shared Memory Programming:

Topological sort of a directed acyclic graph is a linear ordering of its vertices such that for every directed edge (u, v) from vertex u to v, u comes before v in the ordering. For example, a possible topological order of the following graph is 0 6 1 2 5 3 4. We can find the topological order by maintaining an in-degree array in_deg. The serial pseudo-code is shown as below.

```
Input: Directed acyclic graph G(V, E)
Output: A topological order of G in a buffer b
1   empty b
2   compute in_deg for all nodes v
3   while size of b ≠ |V|
4       for node i in G with in_deg (i) = 0
5           push i into b
6       for node i in G with in_deg (i) = 0
7           in_deg(i) = −1
8           for all edges (i, j)
9               in_deg(j) −= 1
```



Suppose we want to parallelize this algorithm with p processors using a shared memory programming model. Each processor is responsible for n/p (assume n is a multiple of p) nodes and their in-degrees. The edges are accessible to all the processors. The array in_deg is a shared variable among all the processors. Assume buffer b is a thread-safe buffer so that it can handle multiple writes concurrently. Your parallel algorithm should only make use of barriers (**NO locks**). Write a pseudo-code for each processor. (Hint: consider three steps in each iteration. Step 1: push nodes with zero in-degree in the n/p nodes assigned to me to the buffer. Step 2: update the in-degrees of the n/p nodes assigned to me based on the removed nodes from the n/p nodes assigned to me. Step 3: carefully update the in-degrees of the other nodes based on the removed nodes from the n/p nodes assigned to me.) (10 points)

Name Initials: _____

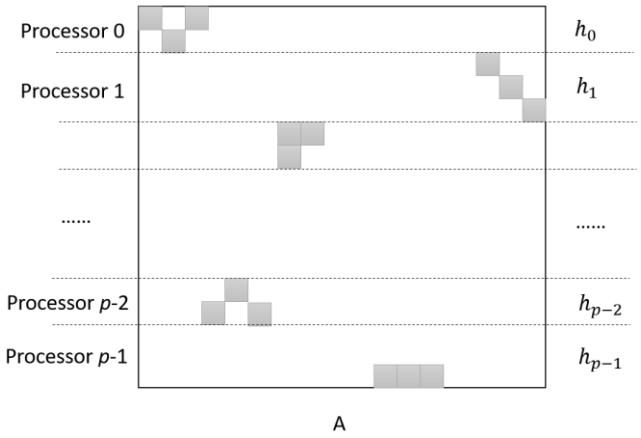
Problem 4 (20 points)

Message Passing and Shared Memory Programming:

Consider iterative matrix-vector multiplication $x = \boldsymbol{A}x$ where x is a vector of length n and $\boldsymbol{A}$ is a sparse matrix of size $n \times n$ with nnz number of non-zero elements. The matrix-vector multiplication is to be computed on p processors in a **1-D mesh with wraparound connection** (using the network model). x is stored in each processor and updated in each iteration. For load balancing, $\boldsymbol{A}$ is partitioned row-wise to p portions such that each portion has exactly nnz/p number of non-zero elements. Let h_k denotes the rows assigned to processor k, $0 \leq k < p$. In each iteration, processor k first computes the matrix-vector product of its corresponding rows and then passes the result to the next processor. The pseudo-code is shown below (assume all variables are private).

Function of processor k in each iteration:

```
1   set x' to all zeros
2   for nonzero elements A(i,j) in my h_k portion:
3       x'(i) += A(i,j) × x(j)
    // compute the result for my h_k rows
4   Barrier
5   x = x' for my h_k rows
    // update my h_k rows in x
6   for i from 0 to p − 2
7       send the h_{k−i mod p} rows of x to
        processor k + 1 mod p
        // update the (received) h_{k−i mod p} rows
8       barrier
```



A

1. Derive an expression for the total parallel execution time (computation and communication) per iteration of the algorithm using the network model. Assume all overheads can be ignored. (8 points)

2. Suppose we want to do the same iterative sparse matrix vector multiplication in the shared memory programming model with p processors. The rows of sparse matrix $\boldsymbol{A}$ is assigned to the processors in the same way. In each iteration, processor k updates the h_k rows in the shared vector x'. Write the pseudo-code for processor k in each iteration, you may use barriers and/or locks to make sure there is no data race. (4 points)

3. Suppose we want to do the same iterative sparse matrix vector multiplication in the shared memory programming model with p processors. However, the sparse matrix A is assigned **column-wise** such that each processor is responsible for n/p (assume n is a multiple of p) columns. In each iteration, processor k updates the rows of vector x' based on the non-zero elements in its columns. Write the pseudo-code for processor k in each iteration, you may use barriers and/or locks to make sure there is no data race. You need to ensure the number of locks used is minimal. (8 points)

Name Initials: _____

Problem 5 (10 points)
Interconnection networks:

1. For the 2-stage permutation network in *Figure 1*, show a permutation that can **NOT** be realized. (2 points)

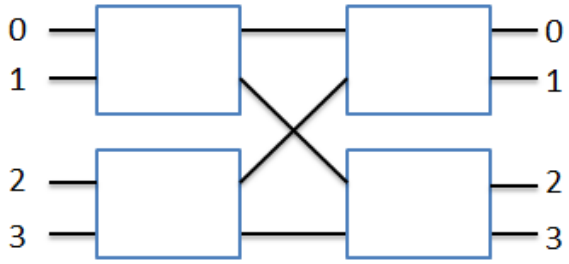


Figure 1: 2-stage permutation network

2. For the n-dimensional hypercube network, show a routing algorithm to go from $x = x_{n-1} \dots x_0$ to destination $y = y_{n-1} \dots y_0$. (3 points)

3. Draw the shuffle-exchange network for $n = 8$. (2 points)

4. For $n = 8$ shuffle exchange network, show a permutation such that the conflict occurs in the first iteration of the routing algorithm discussed in the class. Show the conflict on the shuffle exchange network. (3 points)

Name Initials: _____

Problem 6 (15 points)
Program Mapping:

Embedding $n = 2^k$ hypercube (k is a multiple of 2) on $\sqrt{n} \times \sqrt{n}$ 2-D mesh without wraparound links. A simple embedding is to map PE(i), $i = (i_{k-1} \dots i_0)_2$ in the hypercube to PE($i_{k-1} \dots i_{\frac{k}{2}}$, $i_{\frac{k}{2}-1} \dots i_0$,) in the 2-D mesh. For example, $i = 6 = (0110)_2$ is mapped to PE(1, 2) in the 2-D mesh. Assume that the shortest path is used to embed the edges of the hypercube in the 2-D mesh.

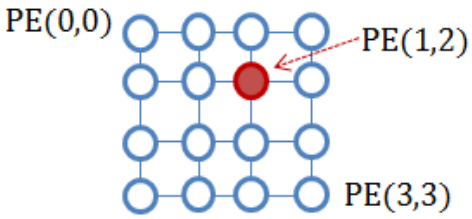
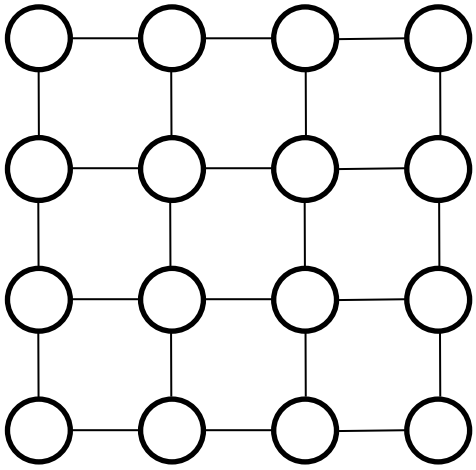


Figure 2. 4×4 *2-D mesh*

1. Show the node embedding for $n = 16$. (2 points)
   Show the mapping of edges (0,1), (0,2), (0,4) and (0, 8) in the 2-D mesh. (2 points)



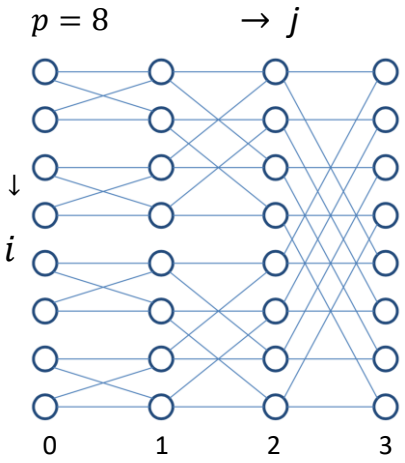
2. For $n = 2^k$, derive the exact expressions for expansion, dilation and congestion (in terms of k) and explain your answer. (5 points)

3. Suppose the 2-D mesh is of size $\frac{\sqrt{n}}{2} \times \frac{\sqrt{n}}{2}$, define your mapping function to embed the $n = 2^k$ hypercube (k is a multiple of 2) and repeat Part 2. (Hint: Partition $\sqrt{n} \times \sqrt{n}$ mesh into 4 $\frac{\sqrt{n}}{2} \times \frac{\sqrt{n}}{2}$ meshes and fold it along X and Y axes) (6 points)

Name Initials: _____

Problem 7 (10 points)
Analytical Modeling:
Adding n numbers on a **butterfly** network with each stage having p processing elements using the network model. Assume $p = n$. Initially, each processing element in stage 0 of the butterfly network is assigned one of the numbers to be added. At the end of the computation, one of the processing elements of the last stage of the butterfly network stores the sum of all the numbers.

1. Write a detailed algorithm by extending the PRAM algorithm discussed in the class. (4 points)



2. What is the parallel time? Explain your answer. (2 points)

3. If $p < n$, initially, each processing element of stage 0 of the butterfly network is assigned n/p numbers to be added (assume n is a multiple of p). Show an approach to compute the sum of the n numbers using the butterfly network. What is the parallel time? (2 points)

4. In this problem, we will pipeline the addition to be performed on a series of input vectors. Assume $p = n$ and in each cycle the processing elements in stage 0 receive one data item (i.e., vector of size n as input in stage 0). Write a pseudo-code using the network model such that the algorithm adds the elements in each input vector with a throughput of 1 vector/cycle. You should show the code to be executed in each processor. (2 points)

Blank Sheet

Blank Sheet