Module 2.5 Homework Exercises

Problem 1: Could simulating the motion of the planets in a solar system be performed well in a shared memory or a distributed memory approach using high-performance computing? For each solar system address the following items and explain your reason why.

1. Can it be executed on a shared memory environment?
2. Will the program run well in a shared memory environment, why or why not? Hint: Think about any bottlenecks that might be encountered.
3. Can it be executed on a distributed memory environment?
4. Will the program run well in a distributed memory environment, why or why not? Hint: Think about any bottlenecks that might be encountered.
5. Would you choose to run this in a shared memory environment or in a distributed memory environment? Explain why.

Solar system A: 9 planets

Solar system B: 1,000 planets

Solar system C: 100,000 planets

Solar system D: 100,000,000 planets

Solar system E: 100,000,000,000 planets

SOLUTION:
While all could be executed in both shared memory or distributed memory environments the size of the solar system is an important consideration. For small solar system sizes enough threads can be created on the physical node to simulate all of the planets. However, as the number of planets increases the number of threads will become too large to be supported on a single physical node efficiently and at that point it would be good to move to a distributed memory approach. The key is for the students to see that the size of the problem is important in determining whether shared or distributed memory should be used.

Problem 2: Review Problem 1 and identify how you would combine shared memory and distributed memory approaches to improve performance of the program? Assume that you have a function PLANET_MODEL(PLANET_NUMBER, PLANET INFORMATION) that takes the

planet number and planet information as input (i.e., this function is called once for each planet) and returns the position of the planet at the next timestep.

Assume that you have a supercomputer with 1,000,000 physical nodes and each node has one processing element. Explain how you would structure the program and what parts would be shared memory and what would be distributed memory. What language would you use for each part (shared memory and distributed memory)? Perform this for a single timestep.

SOLUTION: The key is to see that you can use a shared memory parallel programming language, such as OpenMP or OpenACC, on a single physical node, and then use a distributed memory parallel programming language, such as MPI, to distribute the problem among multiple physical nodes.

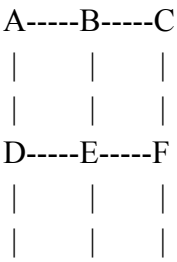
The problem difficulty could be increased if one moves beyond a single timestep because you need to handle synchronizing the system and passing needed data between processes to compute the next timestep.

Problem 3: A planar mesh is a 2D mesh of nodes with each node having a north, south, east, and west link. Traffic is routed from node to node from source to destination and can take any direction. What is the bisection bandwidth of a 3x3 square planar mesh (9 physical nodes)?

SOLUTION: The answer is 3 because the bisection bandwidth of a square planar mesh is the square root of the number of physical nodes in the network. The problem could be expanded by generalizing the size of the square planar mesh.

PROBLEM ON IDENTIFYING BOTTLENECKS

Problem 4: A 3x3 planar mesh network is shown below. Determine the number of hops for a message traveling from all nodes other than node F to node F. Use only the shortest path.



G-----H-----I

SOLUTION: See table below. Some have multiple paths, but the number of hops for the shortest path does not change. The key takeaway is that the network topology will impact the time (number of hops in this case) that a message takes to get from the source to the destination.

Source	Destination	Path	Number of Hops
A	F	1. A->B->C->F 2. A->B->E->F 3. A->D->E->F	3
B	F	1. B->C->F 2. B->E->F	2
C	F	C->F	1
D	F	D->E->F	2
E	F	E->F	1
F	F	N/A	N/A
G	F	1. G->H->I->F 2. G->H->E->F 3. G-> D->E->F	3
H	F	1. H->E->F 2. H->I->F	2
I	F	I->F	1

Problem 5: Look at the solutions to Problem 4 and the paths that are possible for each message. Identify the bottleneck nodes in the system and make a recommendation to reduce the number of messages traveling through each bottleneck node.

SOLUTION: TBD