[50 marks total]

Due Wednesday, 5th October, midnight.

Question 1 [5 marks]

Suppose we have a message passing system composed of two nodes: node 5 and node 10. Both nodes have their own local copies of the variable "x".

As this is a message passing system, changes to "x" at node 5 does not automatically cause the value of variable "x" held at node 10 to change (unlike say if these local memories represented a cache).

Assume that we want to implement an algorithm that requires node 5 to update the value of variable "x" held at node 10.

Write two pseudocode programs (one for node 5 and one for node 10) that jointly achieve this. These programs will need to use the send and receive primitives introduced in Lecture 2.

Question 2 [15 marks]

Imagine that we have a message passing system that supports a send and receive operation. These can be used to implement distributed programs as well as parallel ones. We shall use the term ***coordinator*** instead of *manager* and ***sensor*** instead of *worker* because there is no notion of computational work being handed out.

Assume that we have implemented a distributed system that monitors the movement of people on campus. Movement is detected when people break an infrared beam across the entrance/exit to a room. Rooms on a campus map are briefly lit as people enter or leave rooms across campus.

Node 0 is the coordinator, and it executes an infinite loop waiting for notifications that cause the appropriate room on the map to flash when a notification is received from a sensor node.

Sensor nodes (1-199) are distributed across campus with each node assigned to monitor one room. For simplicity, assume that the node IDs map to the room IDs on the campus map display.

Write some pseudocode for the coordinator and sensor nodes that implements this system. You can assume that within a node that you do not need to worry about race conditions, etc.

Assume that we also have following primitives:

movement_detected() – polls the movement detector (allowing busy wait), returns true when the beam is broken

flash(room_id) – causes the specified room to briefly flash on the campus map

Question 3 [10 marks]

Speedup compares the "best serial implementation of an application measured running on a single node or processor" with the parallel implementation running on N nodes. Briefly discuss why we need to write a different serial implementation rather than just comparing our parallel program running on 1 node versus N nodes?

Question 4 [20 marks]

Consider parallelising the computation of the average of a very large list of numbers of size ***S*** across ***N*** worker nodes. Outline how this could be achieved in pseudocode (with explanations) using a scatter/gather architecture and explain why this problem fits the definition of an embarrassingly parallel problem?