To answer the first 3 questions you will need to use some network related UNIX or Windows commands. If you are not familiar with the commands, please read the UNIX man pages (type `man command-name`) or the Windows help.

1. (5 marks) Use the `host` (/usr/sbin/host or /usr/bin/host, depending on where you are logged on) command in any UNIX server where you have an account or the `nslookup` Windows command, to determine the IP address of at least one of the computers in the UWO’s computer network. Report both, the symbolic name and IP address of the machine that you have selected. Assuming the classful IP addressing scheme, which class of IP addresses (A, B, C, D, or E) are assigned to the computers in the University’s network? Which is the prefix shared by all IP addresses in this network? Up to how many different computers can belong to this network?

2. (15 marks) Use the `ping` command (/usr/sbin/ping) to estimate the number of actual computers in the UWO network. To do this, generate a list $L$ of random IP addresses of the type and network number that you determined in the previous question. Issue a `ping` command for each one of these IP addresses and determine the fraction of $L$ that corresponds to actual computers. Use this fraction to estimate the size of the University’s network.

Show the list $L$ that you used, the fraction of addresses that correspond to actual machines, and your estimate for the size of the network.

3. (10 marks) Use the `traceroute` (/usr/sbin/traceroute) UNIX command or the `tracert` Windows command to find the number of hops between your computer and remote destinations (e.g. to Web sites). The maximum number of hops between two computers is called the diameter of the Internet. Report on the maximum number of hops that you can find. Include the list of servers contacted by `traceroute` and their possible geographical locations (if you can find them). To try to find the geographical location of an IP address you can use one of several available Web services, like `IP2Location` (www.ip2location.com/ip-lookup), or `IP Address Location` (www.ipaddresslocation.org).

4. Let a set of $n$ processors be connected in a ring. In class we studied two algorithms for the leader election problem. The algorithms assume that each processor knows which of its 2 neighbours appears to its right and which one appears to its left in a clockwise ordering of the ring (hence, in that algorithm we could use the instruction “send $mssg$ to the right neighbour”). Suppose that now a processor $p_i$ only knows that it has two neighbours $p_j$ and $p_k$, but it does not know which one appears to its right and which one appears to its left in a clockwise ordering of the ring.

Modify the first leader election algorithm given in class, so that it works in this new situation. You might assume that when a processor receives a message it can determine whether the message arrived from the first neighbour or from the second one. You might also assume that communication links are full duplex, which means that the link connecting processors $p_i$ and $p_j$ can simultaneously carry a message from $p_i$ to $p_j$ and a message from $p_j$ to $p_i$.

- (17 marks) Give in pseudocode (similar to the examples that we have done in class) a synchronous distributed algorithm for solving this problem. You must also explain in English how your algorithm works.
• (8 marks) Prove that your algorithm correctly solves the leader election problem.
• (10 marks) Compute the time complexity and communication complexity of your algorithm in the worst case.

5. Consider a network formed by a set of processors connected in two dimensional grid, as shown in the following figure

As always, assume that each processor $p_i$ has a unique identifier $id_i$. The only information that a processor $p_i$ knows about the network is its neighbours. A processor has either two, three, or four neighbours. Assume that each processor knows where its neighbours are located (to its right, left, above, or below). So a processor can, for example, send a message to the neighbor above it.

Design a synchronous algorithm for finding the largest processor identifier in the network and for sharing this information with each processor in the system; so at the end of the execution of the algorithm each processor must know the largest identifier in the network.

As above, when a processor receives a message you might assume that the processor knows which of its neighbours sent it. You might also assume that communication links are full duplex.

• (17 marks) Give in pseudocode (similar to the examples that we have done in class) a synchronous distributed algorithm for solving this problem. You must also explain in English how your algorithm works.
• (8 marks) Prove that your algorithm correctly solves the problem.
• (10 marks) Compute the time complexity and communication complexity of your algorithm in the worst case. Assume that the grid has width $W$ and length $L$ so the number of processors in the network is $n = W \times L$. 