Assignment 3

CPSC 416

Due March 21, 2014

Use MPI to write an efficient leader election algorithm for a bidirectional ring.

In class we looked at the Chang-Roberts algorithm for leader election on a unidirectional ring with N processes. We noticed that in the worst case, it could require $O(n^2)$ messages. It is possible to achieve better performance for a bidirectional ring.

The goal of this assignment is to use similar assumptions as the Chang-Roberts example discussed in class to write a distributed leader election algorithm for N processes.

Your program should run with the following command:

```
mpiexec -n NUM ./electleader PNUM
```

Please make sure your executable is called "electleader" and include a Makefile to make the executable. Your program should use the following rule,

```
(myRank + 1) * PNUM mod NUM
```

to assign a unique identifier for each of the [NUM] nodes as long as [PNUM] is relatively prime to [NUM].

Once a leader is elected, the program should have each node print its rank, identifier value, leader status, number of messages received, and number of messages sent according to the specification:

```
rank=%d, id=%d, leader=0, mrcvd=%d, msent=%d
```

Once every process has printed the information described above, there should be one more messaging round to send the total messages sent and total messages received by every node to the leader. The leader, finally, prints the following.

```
rank=%d, id=%d, trcvd=%d, tsent=%d
```

Handin instructions: Please try to send no more than two files, namely, a Makefile and an electleader.c file. Please put your name and student ID at the top of your .c file and also add mention any special details about your implementation.