455303 Parallel Programming 2014
Exercise 1. Communication in a ring of processes

Student name : Vu Nguyen
Student number: 71141

Description of the implementation of the program

The implementation of the program is based on the following algorithm which is written in pseudocode. All MPI functions (except MPI_Wtime) are handled with error return code before proceeding. Blocking mode is chosen for sending and receiving messages in the program.

/* For all processes in the ring */
- Initialize MPI
- Get the number of active processors
- Get the id number of the process which is running the task
- Check that we run on at least two processors
- Find out the id number of the previous processor and of the next processor

/* For process 0 */
(*) Get the size of the message from the user
- Check the size of the message
- If the message size > maximum size or message size < 0
  Display a message to stdout and give its status
  Assign the message size to 0
- If the size of the message is 0
  Send this size to the next process and wait for it from the last process
  Terminate process 0
- If the size of the message is valid (0 < size <= maximum size)
  Allocate memory dynamically for the output buffer and for the input buffer with the specified size
- Generate a message of specified size to the output buffer
- Start the timer
- Send the message to the next process
- Wait for the message from the last process
- Stop the timer
- Calculate message transfer time and display it to stdout
- Deallocate memory from the heap for both the input buffer and the output buffer
- Repeat step (*)

/* For other processes (process 1 to process p-1) in the ring except process 0 where p is the number of active processes */
(**)Receive the size of the message from the previous process and forward it to the next process
- Check the size of the message
- If the size of the message is 0
    Terminate the current running process
- If the size of the message is valid (not equal to 0, since we already know that processes other than process 0 will receive the size of the message which is either 0 or in the range of (0, maximum message size].
- Dynamically allocate memory for the input buffer with the specified size
- Receive the message of specified size from the previous process and forward it to the next process
- Deallocate memory from the heap for the input buffer
Repeat step (**) if the size of the message is not equal to 0

Result of measurements

Nr. of processors	Message size (bytes)	Times (s)
2	1	0.000001
2	1000	0.000003
2	1000000	0.000531
2	1000000000	0.716478
12	1	0.000011
12	1000	0.000020
12	1000000	0.003611
12	1000000000	4.750158
13	1	0.000021
13	1000	0.000032
13	1000000	0.003914
13	1000000000	5.592618
24	1	0.000035
24	1000	0.000061
24	1000000	0.007147
24	1000000000	10.713358
48	1	0.000075
48	1000	0.000123
48	1000000	0.014649
48	1000000000	21.352584