Oscar Galindo

Week 3 Homework

Dr. Shirley Moore

Open MPI exercises

1. Parallel implementation of the code:

Text

Description automatically generated

Comment: To test the code the values for a, b, and n were 0, 1000, and 100,000,000 respectively. These values were hard coded into the program to have a more reliable reading of time using the “time” command.

8 threads:

Text

Description automatically generated

4 threads, took 1.981 secs:

Text

Description automatically generated

2 threads, took 3.925 secs:

Text

Description automatically generated

1 thread, took 7.760 secs:

Text

Description automatically generated

Graph:

Chart, line chart

Description automatically generated

* 1. In the case of the even-odd transposition algorithm it is possible since the changes are localized in pairs of entries of the same data-structure, in other words, you can change/access indices of the same array without affecting the consistency of the data stored in the array when every thread executes. Nevertheless, the implementation of bubble sort provided as code cannot be implemented as a parallel program since the entries of the data structure are overlapped, this causes a race condition in the data and such race condition could cause the loss of data or simply a disorder on how elements are stored in the array.
  2. Even-odd transposition implementation:

Text

Description automatically generated

8 threads:

Text

Description automatically generated

4 threads:

Text

Description automatically generated

2 threads:

Text

Description automatically generated

1 thread:

Text

Description automatically generated with medium confidence

Graph:

Chart, line chart

Description automatically generated

1. Solution:

Text

Description automatically generated

For this code I basically introduced the strtok\_r function which is a reentrant version of the function (which means it allows the threads to be preemptied and then return to the place where they were printing), and I added a synchronization mechanism so that the threads had a busy waiting before printing their respective line. This implementation is not efficient, it does work, but it requires the synchronization of the threads. I believe there can be a version that does indeed allow for multiple threads to print, but I didn’t achieve such implementation.