

## Assignment 1 - Part II

### Parallel Programming

---

Objective: The objective of this part of the assignment is to learn a very simple MPI point-to-point communication in an embarrassingly parallel example.

You must complete part I of this assignment before this assignment. This exercise is based on the book example that we discussed in class. Write a program that takes from the line command an integer number  $n$ , and two floating-point numbers  $a$  and  $b$ . Then, your program should compute the definite integral of a given function **func** on the interval  $[a, b]$ . The function **func** should be provided in a separate file named `func.cpp`. The prototype of the function must be

```
float func(float *x, int n) {  
}
```

Follow the following steps in completing the assignment:

- 1) Write a sequential program `integrate.cpp` that computes the integration. The program should exit with usage message if the user didn't pass the correct number of parameters.
- 2) Write an MPI parallel program `mpi_integrate_v1.cpp` that does the integration by dividing the work  $a$  among  $p$  processes. For example, if  $n$  is 100 and  $p$  is 10, then each process computes its local integration on part of the interval using the trapezoid rule as discussed in class. Then have each process send its local integration value to the master process (process 0) using point-to-point communication. Your MPI program should accept parameters from the line command similar to the serial version, for example

```
$ mpirun -np 4 ./mpi_integrate 1 3.5 100
```

- 3) Modify your `mpi_integration_v1` program, so that the master process receives from any source instead of receiving from process 1, 2, ..., etc. Name this version of the program `mpi_integrate_v2.cpp`
- 4) Modify your program so that you use collective communication instead of point-to-point communication. Call this version of the program `mpi_integrate_v3.cpp`

## What to turn in

Submit by the due date a tar ball that contains the following files. The files names must be as specified below. If your file naming is different you will get zero points. Also, the interface of your program must be as specified.

- 1) `integrate.cpp` - the serial version of your program (5 points)
- 2) `func.cpp` - contains the implementation of the function  $f(x) = x^2 + 3x + 10$
- 3) `mpi_integrate_v1.cpp` - mpi program version 1 (5 points)
- 4) `mpi_integrate_v2.cpp` - mpi program version 2 (5 points)
- 5) `mpi_integrate_v3.cpp` - mpi program version 3 (5 points)
- 6) `integrate.txt` – a text file that contain a short description of what you did in each part (5 points)