Project 2 - Group 5

Team Members

Group Member	R#
Michael Beebe	R11772231
Diego Salas Noain	R11794236
Bandar Alkhalil	R11836831
Yongjian Zhao	R11915830
Denish Otieno	R11743138
Shiva Kumar Neekishetty	R11842757

Required Software

- MPI implementation (we are using Open MPI)
- C Compiler (such as gcc or clang)
- Make
- Bash

Instructions

Compile

To change the MPI wrapper to something other than `mpicc` (such as `mpich`), edit line 1 of the Makefile.

```
make
```

Run

```
./run.sh <desired number of processes>
```

The default number of processes is 4. You can change this by passing a command line argument when executing `run.sh`.

If you get an error saying "permission denied", run

```
chmod +x run.sh
```

then rerun `./run.sh`

Clean Build

```
make clean
```

Code Breakdown

mybarrier():

- Synchronizes all participating processors.
- The synchronization is achieved in two main phases, each running for `log2(process_count)` iterations, where `process_count` is the total number of processors:
- Reducing Phase:
  - Starts with the highest power of 2 less than `process_count` and decrements in each iteration.
  - In each iteration, processors send notification messages to other processors based on their rank and the current power of 2.
- Scattering Phase:
  - Starts from the lowest power of 2 and increments in each iteration.
  - Similar to the reducing phase, processors send and receive notification messages based on their rank and the current power of 2.
- The use of `log2(process_count)` iterations ensures efficient synchronization, as each processor communicates directly with another processor in each step, reducing the total number of steps needed for all processors to synchronize.

main():

- Initializes the MPI environment.
- Retrieves and prints information about the current processor.
- Calls the `mybarrier()` function to synchronize all processors.
- Finalizes the MPI environment.