CS5500 HW2

Chase Mortensen A01535275

September 2019

1 Description

Write an MPI program that performs a bionic integer sort using partial list of numbers. You may assume power of 2 processors.

My implementation is based off of the globalsum.cpp program written in class. It takes a slightly different approach using the pow() method instead of bit shifting and nested `for` loops instead of a while loop. Values are sent back and forth using MPI_Send and MPI_Recv methods based on the process rank, the process value, and whether the segment is ascending or descending. Each process is initialized to `size - rank - 1`; a descending list. The comments in the code provide greater insight. The sorted list is printed by iterating through a vector created on process zero and having the other processes send their values to process zero.

2 Program

```
#include <iostream>
#include <mpi.h>
#include <bits/stdc++.h>
#include <cmath>

#define MCW MPI_COMM_WORLD

using namespace std;

int main(int argc, char **argv){

    int rank, size;
    int data;
    int powOfTwo = 0;
    int powSize = 1;
    int mask;
int dest;
int power;

MPI_Init(&argc, &argv);
MPI_Comm_rank(MCW, &rank);
MPI_Comm_size(MCW, &size);

bitset<32> bitRank = rank;
int myVal = size - rank - 1;
int destVal = 0;

while(powSize<size){
    powSize<<=1;
    powOfTwo++;
}

mask = powSize>>1;

for (int i = 0; i < powOfTwo; i++){ // Convert list to bitonic list
    for (int j = i; j >= 0 ; j--){
        power = pow(2, j);
        dest = rank ^ power;

        MPI_Send(&myVal,1,MPI_INT,dest,0,MCW);
        MPI_Recv(&destVal,1,MPI_INT,dest,0,MCW,MPI_STATUS_IGNORE);


        if (bitRank[i+1] == 0){ // ascending
            if (rank > dest) { // rank gets bigger number
                if (myVal < destVal){
                    MPI_Send(&myVal,1,MPI_INT,dest,0,MCW);
                }
                else MPI_Send(&destVal,1,MPI_INT,dest,0,MCW);
            }
            else{ // rank gets smaller number
                if (myVal > destVal){
                    MPI_Send(&myVal,1,MPI_INT,dest,0,MCW);
                }
                else MPI_Send(&destVal,1,MPI_INT,dest,0,MCW);
            }
        }else{ // descending
            if (rank > dest) { // rank gets smaller number
                if (myVal > destVal){
                    MPI_Send(&myVal,1,MPI_INT,dest,0,MCW);
                }
                else MPI_Send(&destVal,1,MPI_INT,dest,0,MCW);
                }
                else { // rank gets bigger number
                    if (myVal < destVal){
                        MPI_Send(&myVal,1,MPI_INT,dest,0,MCW);
                    }
                    else MPI_Send(&destVal,1,MPI_INT,dest,0,MCW);
                }
            }

            MPI_Recv(&myVal,1,MPI_INT,dest,0,MCW,MPI_STATUS_IGNORE);
        }
    }

    if (rank) MPI_Send(&myVal,1,MPI_INT,0,0,MCW);
        if (!rank){
            vector<int> vec;
            vec.push_back(myVal);
            int tmp = 0;
            for (int i = 1; i < size; i++){
                MPI_Recv(&tmp,1,MPI_INT,i,0,MCW,MPI_STATUS_IGNORE);
                vec.push_back(tmp);
            }
            for (int i = 0; i < size; i++){
                cout << vec[i] << " ";
            }
            cout << endl;
        }

        MPI_Finalize();

        return 0;
    }
}
```

3 Output

Example 1:

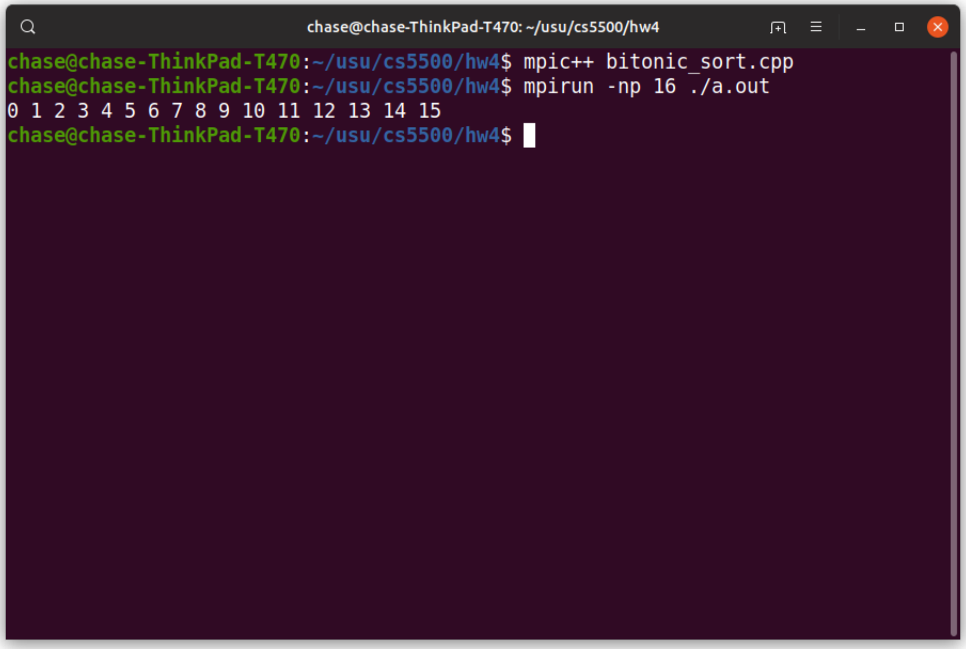
```
$ mpic++ bitonic_sort.cpp
$ mpirun -np 8 ./a.out

0 1 2 3 4 5 6 7
```

Example 2:

```
$ mpic++ bitonic_sort.cpp
$ mpirun -np 16 ./a.out

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
```



```
chase@chase-ThinkPad-T470:~/usu/cs5500/hw4$ mpic++ bitonic_sort.cpp
chase@chase-ThinkPad-T470:~/usu/cs5500/hw4$ mpirun -np 16 ./a.out
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
chase@chase-ThinkPad-T470:~/usu/cs5500/hw4$
```