

# Parallel Programming HW5 Write Up

## Due Monday February 22, 2021, 11:59 pm

Connor Osborne  
A01880782

Monday February 22, 2021

## 1 Description

For homework 5, I wrote an MPI program that performs a bionic integer sort using a list of numbers, assuming power of 2 processes and list size.

## 2 Implementation

The implementation for this program consists of three main parts. The first section initializes the variables needed for MPI processes such as rank, size, and mystatus. Then it seeds the random number generator for each process based off rank to ensure that most numbers will be different from each other. Then an array of random integers, toSort, is generated to be sorted as well as, recvData, an array for storing the sorted list once it is ready. Next, rank 0 prints out the array that will be sorted. Finally, MPI\_Scatter is used to send each process an element of the toSort array and storing them as data.

---

```
int rank, size;

MPI_Status mystatus;
int data;
MPI_Init(&argc, &argv);
MPI_Comm_rank(MCW, &rank);
MPI_Comm_size(MCW, &size);
srand(time(NULL) + rank);

int toSort[size];
for(int i = 0; i < size; ++i){
    toSort[i] = rand() % (size * 3) + 1;
}
int recvData[size];

if(rank==0){
    cout << "Original List: " << endl;
    for(int i=0;i<size;++i){
        cout<<toSort[i]<<endl;
    }
}
```

```
MPI_Scatter(toSort,1,MPI_INT,&data,1,MPI_INT,0,MCW);
```

---

The second section of the program is where the sorting takes place. First a variable, pwr, is made to establish what power of 2 is the number of processes being used. Then a nested for loop is used so that for i ranging from 0 to pwr, the hyper cube swap will be used j times where j ranges from i to 0. The hyper cube swap uses j as it's swapping bit each time. After the swap an if statement checking the ith bit of the process' rank checks to see if an ascending list or a descending list is being formed, if the ithbit() function returns 0 then it is descending if it returns 1 it is ascending. Then nested in that if-else statement is another if-else statement to check the jth bit of the process' rank to determine if it wants the lower number or the higher number of the swap if ithbit() returns a 0 then the process wants the smaller number if it returns a 1 then it wants the higher number. Then one more if statement is used to decide whether to swap the stored value with the received value or keep it.

---

```
int currentData = data;

MPI_Barrier(MCW);
for(int i = 0; i < pwr; ++i){
    for(int j = i; j >= 0; j--){
        MPI_Barrier(MCW);
        unsigned int map = 1;
        int dest = rank^(map<<j);

        MPI_Send(&currentData,1,MPI_INT,dest,0,MCW);
        MPI_Recv(&data,1,MPI_INT,MPI_ANY_SOURCE,0,MCW,&mystatus);
        if(!ithbit(rank,i+1)){
            if(ithbit(rank,j)){
                if (data >= currentData){
                    currentData = data;
                }
            }
            else{
                if (data <= currentData){
                    currentData = data;
                }
            }
        }
        else{
            if(ithbit(rank,j)){
                if (data <= currentData){
                    currentData = data;
                }
            }
            else{
                if (data >= currentData){
                    currentData = data;
                }
            }
        }
    }
}

MPI_Barrier(MCW);
```

---

The `ithbit()` function used in the second section of code was implemented by passing in the rank number and what bit position to check. Then the return value is the result of an and operation between rank at the `ith` bit and 1.

---

```
int ithbit(int r,int i){
return (r & (1<<i));
}
```

---

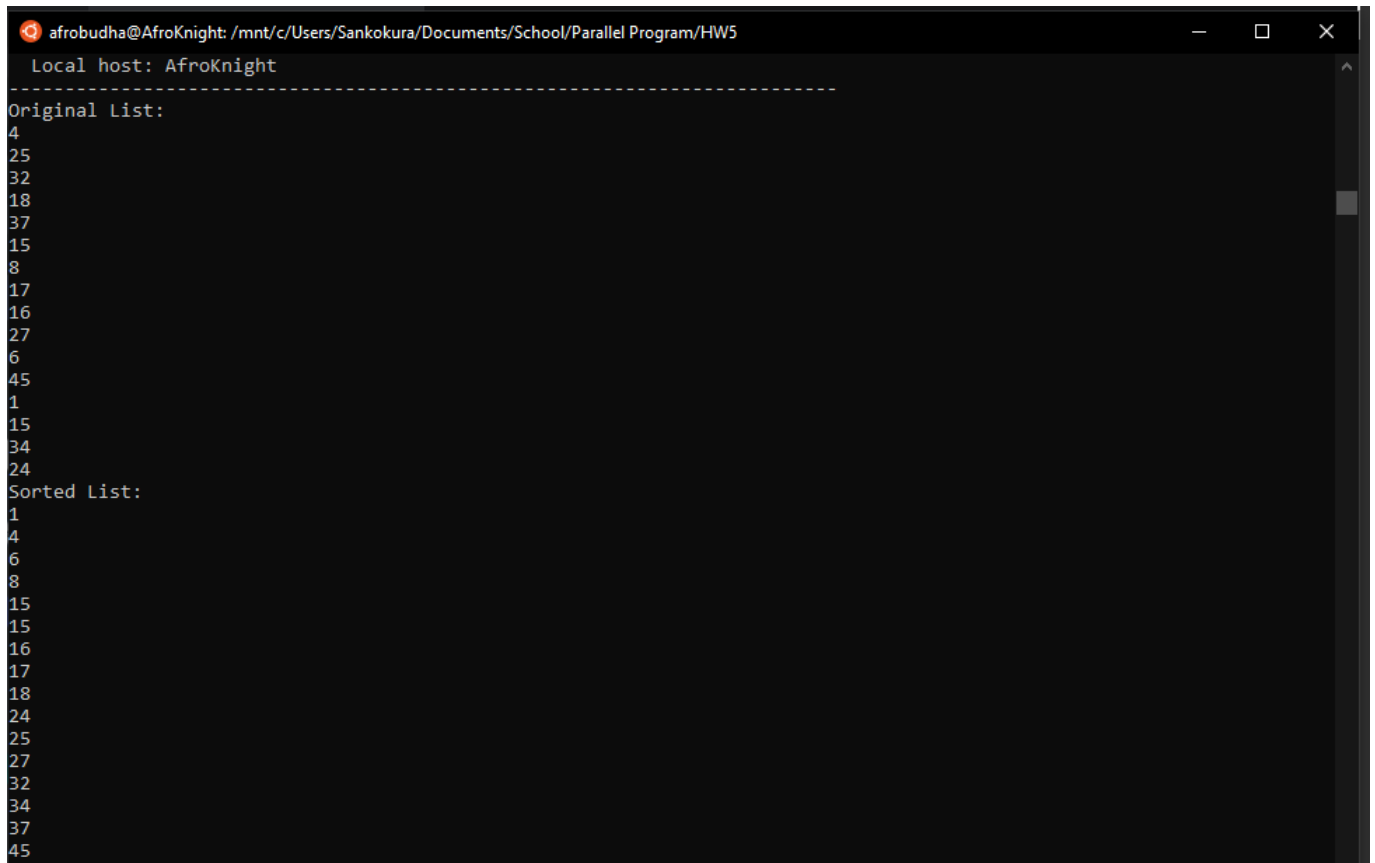
The final section of code takes all of the currently stored values of each process and stores them all in the `recvData` array by using `MPI_Allgather`. Then rank zero prints out the sorted array.

---

```
MPI_Allgather(&currentData,1,MPI_INT,recvData,1,MPI_INT,MCW);
sleep(1);
if(rank==0){
    cout << "Sorted List: " << endl;
    for(int i=0;i<size;++i){
        cout<<recvData[i]<<endl;
    }
}
```

---

### 3 Input and Output with 16 processes



```
afrobudha@AfroKnight: /mnt/c/Users/Sankokura/Documents/School/Parallel Program/HW5
Local host: AfroKnight
-----
Original List:
4
25
32
18
37
15
8
17
16
27
6
45
1
15
34
24
Sorted List:
1
4
6
8
15
15
16
17
18
24
25
27
32
34
37
45
```