Yakshita B Rakholiya
yr92282n@pace.edu
Student ID: U01875270
Course: CS-610-22756
Project-5

- Developing an efficient MPI-based parallel program for efficient spatial filtering of a visual image on a 7-dimensional hypercube.

C:\parallel\cstar.exe

```
/*
Pace University CS610
Yakshita Rakholiya
Project-8 @Dr.Lixin Tao @Kai Wang
*/

ARCHITECTURE HYPERCUBE(7);
#include <mpi.h>
#include<stdlib.h>
#define image_dim 640;
#define rows_per_partition 5
int inrows[rows_per_partition+2][image_dim+2], outrows[rows_per_partition][image_dim];
int filter[3][3] = {1,1,1,1,1,1,1,1,1};
MPI_Status status;
int totalproc, myrank, blocksize,i,j,k,destination,messageSource, messageTag;
void input_image( ) {
int image[image_dim+2][image_dim+2];

    for (i = 0; i <image_dim; i++)
    for (j = 0; j <image_dim; j++) {
      image[i][j] = (rand() % 10);
    }

for (k = 0; k < totalproc; k++) {
destination = k; messageTag = 1;
MPI_Send(&image[k*rows_per_partition][0], blocksize, MPI_INT,destination, messageTag, MPI_COMM_WORLD);

}
}
main( ) {

MPI_Init( );
MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
MPI_Comm_size(MPI_COMM_WORLD, &totalproc);
blocksize = (rows_per_partition+2)*(image_dim+2);
if (myrank == 0) input_image( );
messageSource = 0; messageTag = 1;
MPI_Recv(&inrows[0][0], blocksize, MPI_INT, messageSource,
messageTag, MPI_COMM_WORLD, &status);

for (i = 1; i <= rows_per_partition; i++)
for (j = 1; j <= image_dim; j++)
outrows[i-1,j-1] =
```

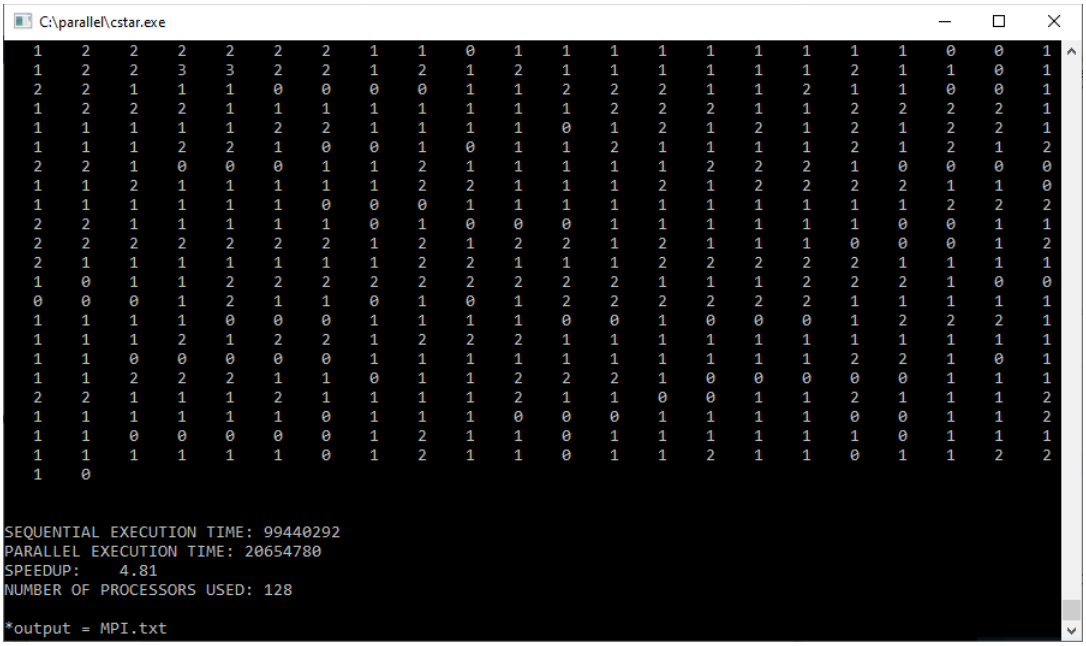
C:\parallel\cstar.exe

```
}
main( ) {

MPI_Init( );
MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
MPI_Comm_size(MPI_COMM_WORLD, &totalproc);
blocksize = (rows_per_partition+2)*(image_dim+2);
if (myrank == 0) input_image( );
messageSource = 0; messageTag = 1;
MPI_Recv(&inrows[0][0], blocksize, MPI_INT, messageSource,
messageTag, MPI_COMM_WORLD, &status);

for (i = 1; i <= rows_per_partition; i++)
for (j = 1; j <= image_dim; j++)
outrows[i-1,j-1] =
(int) ((filter[0,0]*inrows[i-1,j-1]
+ filter[0,1]*inrows[i-1,j]
+ filter[0,2]*inrows[i-1,j+1]
+ filter[1,0]*inrows[i,j-1]
+ filter[1,1]*inrows[i,j]
+ filter[1,2]*inrows[i,j+1]
+ filter[2,0]*inrows[i+1,j-1]
+ filter[2,1]*inrows[i+1,j]
+ filter[2,2]*inrows[i+1,j+1]) / 9);
blocksize = rows_per_partition*image_dim;
destination = 0; messageTag = 2;
MPI_Send(&outrows[0][0], blocksize, MPI_INT, destination,
messageTag, MPI_COMM_WORLD);
if (myrank == 0) {
for (k = 0; k < totalproc; k++) {
messageSource = k; messageTag = 2;
MPI_Recv(&outrows[0][0], blocksize, MPI_INT,
messageSource, messageTag, MPI_COMM_WORLD, &status);
for (i = 0; i < rows_per_partition; i++) {
for (j = 0; j < image_dim; j++)
cout << outrows[i][j] << " ";
    cout << "\n";
}
}
}
MPI_Finalize( );
}

```


C:\parallel\cstar.exe
1 2 2 2 2 2 2 1 1 0 1 1 1 1 1 1 1 1 1 0 0 1
1 2 2 3 3 2 2 1 2 1 2 1 1 1 1 1 1 2 1 1 0 1
2 2 1 1 1 0 0 0 0 1 1 2 2 2 1 1 2 1 1 0 0 1
1 2 2 2 1 1 1 1 1 1 1 1 2 2 2 1 1 2 2 2 2 1
1 1 1 1 1 2 2 1 1 1 1 0 1 2 1 2 1 2 1 2 2 1
1 1 1 2 2 1 0 0 1 0 1 1 2 1 1 1 1 2 1 2 1 2
2 2 1 0 0 0 1 1 2 1 1 1 1 1 2 2 2 1 0 0 0 0
1 1 2 1 1 1 1 1 2 2 1 1 1 2 1 2 2 2 2 1 1 0
1 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 2
2 2 1 1 1 1 1 0 1 0 0 0 1 1 1 1 1 1 0 0 1 1
2 2 2 2 2 2 2 1 2 1 2 2 1 2 1 1 1 0 0 0 1 2
2 1 1 1 1 1 1 1 2 2 1 1 1 2 2 2 2 2 1 1 1 1
1 0 1 1 2 2 2 2 2 2 2 2 2 1 1 1 2 2 2 1 0 0
0 0 0 1 2 1 1 0 1 0 1 2 2 2 2 2 2 1 1 1 1 1
1 1 1 1 0 0 0 1 1 1 1 0 0 1 0 0 0 1 2 2 2 1
1 1 1 2 1 2 2 1 2 2 2 1 1 1 1 1 1 1 1 1 1 1
1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 2 2 1 0 1
1 1 2 2 2 1 1 0 1 1 2 2 2 1 0 0 0 0 0 1 1 1
2 2 1 1 1 2 1 1 1 1 2 1 1 0 0 1 1 2 1 1 1 2
1 1 1 1 1 1 0 1 1 1 0 0 0 1 1 1 1 1 0 0 1 1 2
1 1 0 0 0 0 0 1 2 1 1 0 1 1 1 1 1 1 0 1 1 1
1 1 1 1 1 1 0 1 2 1 1 0 1 1 2 1 1 0 1 1 2 2
1 0
SEQUENTIAL EXECUTION TIME: 99440292
PARALLEL EXECUTION TIME: 20654780
SPEEDUP: 4.81
NUMBER OF PROCESSORS USED: 128
*output = MPI.txt