Paleti Krishnasai
CED18I039

# HPC LAB - DA2

**Name**: Paleti Krishnasai
**Roll No**: CED18I039
**Programming Environment**: MPI
**Problem**: DA2
**Date**: 21-10-2021

**Hardware Configuration:**
PU NAME: Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz
Number of Sockets: 1
Cores per Socket: 4
Threads per core: 2
L1d cache: 128 KiB
L1i cache: 128 KiB
L2 cache: 1 MiB
L3 cache: 8 MiB

```
paleti@paleti-Lenovo-ideapad-330-15ICH:~$ likwid-topology
--------------------------------------------------------------------------------
CPU name:       Intel(R) Core(TM) i5-8300H CPU @ 2.30GHz
CPU type:       Intel Coffeelake processor
CPU stepping:   10
********************************************************************************
Hardware Thread Topology
********************************************************************************
Sockets:                1
Cores per socket:       4
Threads per core:       2
--------------------------------------------------------------------------------
HWThread        Thread          Core            Socket          Available
0               0               0               0               *
1               0               1               0               *
2               0               2               0               *
3               0               3               0               *
4               1               0               0               *
5               1               1               0               *
6               1               2               0               *
7               1               3               0               *
--------------------------------------------------------------------------------
Socket 0:               ( 0 4 1 5 2 6 3 7 )
--------------------------------------------------------------------------------
********************************************************************************
Cache Topology
********************************************************************************
Level:                  1
Size:                   32 kB
Cache groups:           ( 0 4 ) ( 1 5 ) ( 2 6 ) ( 3 7 )
--------------------------------------------------------------------------------
Level:                  2
Size:                   256 kB
Cache groups:           ( 0 4 ) ( 1 5 ) ( 2 6 ) ( 3 7 )
--------------------------------------------------------------------------------
Level:                  3
Size:                   8 MB
Cache groups:           ( 0 4 1 5 2 6 3 7 )
--------------------------------------------------------------------------------
********************************************************************************
NUMA Topology
********************************************************************************
NUMA domains:           1
--------------------------------------------------------------------------------
Domain:                 0
Processors:             ( 0 1 2 3 4 5 6 7 )
Distances:              10
Free memory:            3546.2 MB
Total memory:           7831.84 MB
--------------------------------------------------------------------------------
```

```
**************************************************
Graphical Topology
**************************************************
Socket 0:
+-------------------------------------------------+
| +---------+ +---------+ +---------+ +---------+ |
| |  0 4    | |  1 5    | |  2 6    | |  3 7    | |
| +---------+ +---------+ +---------+ +---------+ |
| +---------+ +---------+ +---------+ +---------+ |
| |  32 kB  | |  32 kB  | |  32 kB  | |  32 kB  | |
| +---------+ +---------+ +---------+ +---------+ |
| +---------+ +---------+ +---------+ +---------+ |
| | 256 kB  | | 256 kB  | | 256 kB  | | 256 kB  | |
| +---------+ +---------+ +---------+ +---------+ |
| +-------------------------------------------+ |
| |                   8 MB                    | |
| +-------------------------------------------+ |
+-------------------------------------------------+
```

No of nodes: 12 ( 4 for each as written in the machine file ).

Paleti Krishnasai
CED18I039

**Serial Code:**

```
#include <mpi.h>
#include <bits/stdc++.h>
#include<fstream>
#include<iostream>
#include <string.h>
#include <string>
#include<algorithm>
    int image[514][514],new_image[512][512];
using namespace std;

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

 MPI_Init (&argc, &argv);
double start = MPI_Wtime();
    ifstream file("lena.txt");
    string s,x;

    int i=1,j=1;
    while(getline(file,s))
    {
        j=1;
        stringstream ss(s);
        while(getline(ss,x,','))
        {
            image[i][j] = std::stoi(x);
            j++;
        }
        i++;
    }


   for(int j=0;j<514;j++)
   {
      image[0][j]=0;
      image[j][0]=0;
      image[513][j]=0;
      image[j][513]=0;
   }
```

```
 for(int i=1;i<513;i++)
 {
   for(int j=1;j<513;j++)
   {
     int m1,m2;
               m1=INT_MIN;
               m2=INT_MAX;
               vector<int> pq;
               pq.push_back(image[i][j-1]);
               pq.push_back(image[i][j+1]);
               pq.push_back(image[i][j]);
               pq.push_back(image[i-1][j-1]);
               pq.push_back(image[i-1][j]);
               pq.push_back(image[i-1][j+1]);
               pq.push_back(image[i+1][j-1]);
               pq.push_back(image[i+1][j]);
               pq.push_back(image[i+1][j+1]);
               for (int k=0;k<pq.size();k++)
               {
                   m1=max(m1,pq[k]);
                   m2=min(m2,pq[k]);
               }
     new_image[i-1][j-1]=(m1+m2)/2;


   }

 }

  ofstream myfile;
myfile.open ("lena_out.txt");

 for(int i=0; i<512; i++)
 {
   for(int j=0;j<512;j++)
   {
     myfile<<new_image[i][j];
     myfile<<",";
```

```
      }
      myfile<<"\n";
    }
    myfile.close();
double end = MPI_Wtime();
printf("execution time = %lf\n",end - start);

    return 0;
}
```

Paleti Krishnasai
CED18I039

**MPI code:**

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

using namespace std;
int Lena[514][514];
int out_Lena[512][512];

#define M_PI 3.14159265358979323846
int  n=512;

#define MASTER 0
#define FROM_MASTER 1
#define FROM_WORKER 2
void printfile();
void process_Lena(int argc, char *argv[]);



void process_Lena(int argc, char *argv[])
{
    int no_tasks, taskid, no_workers, source, workers, mtype,
no_elements_iter,no_elements_it2, no_elements, no_elements_left, index, j,
k,rc, ack;

    double s1,s2;

    MPI_Status status;
    MPI_Request request;
    MPI_Init(&argc, &argv);

    MPI_Comm_rank(MPI_COMM_WORLD, &taskid);
    MPI_Comm_size(MPI_COMM_WORLD, &no_tasks);

    if (no_tasks < 2)
    {
        printf("Available Processors =%d\n",no_tasks);
```

```
        printf("Program is Terminated since there are less than 2
threads\n");
        MPI_Abort(MPI_COMM_WORLD, rc);
        exit(1);
    }

    char pro_name[MPI_MAX_PROCESSOR_NAME];
    int length;
    MPI_Get_processor_name(pro_name,&length);

    no_workers = no_tasks - 1;
    s1=MPI_Wtime();

        ////////////////////////////////////////////////////////////
MASTER ////////////////////////////////////////////////////////////////

    if(taskid==0)
    {
        no_elements = n/no_workers;
        no_elements_left = n%no_workers;
        index = 0;

        mtype = FROM_MASTER;
        int count=0;
        int i,i1=0,i2,start,end;
        int a[no_workers];
        for(i=0;i<no_workers-1;i++){
            a[i]=no_elements;

        }
        a[i]=no_elements+no_elements_left;

        for (workers=1; workers<=no_workers; workers++)
        {
            no_elements_iter=a[workers-1];
            int temp = a[workers-1];
            i2=i1+temp-1;
            start = i1; // for main and out
            end = i2+3-1; // for main
```

```
            no_elements_it2=end-start+1;
            MPI_Send(&start, 1, MPI_INT, workers, 1, MPI_COMM_WORLD);
            MPI_Send(&no_elements_iter, 1, MPI_INT, workers, 2,
MPI_COMM_WORLD);
            MPI_Send(&no_elements_it2, 1, MPI_INT, workers, 3,
MPI_COMM_WORLD);

            MPI_Send(&Lena[start][0],no_elements_it2* 514, MPI_INT,
workers, 4,MPI_COMM_WORLD);
            i1=i2+1;

        }

        for (i=1; i<=no_workers; i++)
        {
            source = i;
            MPI_Recv(&index, 1, MPI_INT, source, 5, MPI_COMM_WORLD,
&status);

            MPI_Recv(&no_elements_iter, 1, MPI_INT, source, 6,
MPI_COMM_WORLD, &status);

            MPI_Recv(&out_Lena[index][0], no_elements_iter*512, MPI_INT,
source, 7,MPI_COMM_WORLD, &status);
            //cout<< "TESTUNG 3";

        }

        s2=MPI_Wtime()-s1;

        printf("Execution Time :%f\n",s2);
                printfile();
    }
    ////////////////////////////////////////////////////////////////// SLAVE
//////////////////////////////////////////////////////////////////
    if(taskid>0)
    {
        mtype = FROM_MASTER;

        MPI_Recv(&index, 1, MPI_INT, MASTER, 1, MPI_COMM_WORLD, &status);
```

```
        MPI_Recv(&no_elements_iter, 1, MPI_INT, MASTER, 2, MPI_COMM_WORLD,
&status);
        MPI_Recv(&no_elements_it2, 1, MPI_INT, MASTER, 3,MPI_COMM_WORLD,
&status);

        MPI_Recv(&Lena[index][0], no_elements_it2*514, MPI_INT, MASTER,
4,MPI_COMM_WORLD, &status);

        int start =index+1;

        for(int i=start;i<index+no_elements_it2-1;i++)
        {
            for(int j=1;j<513;j++)
            {

                int m1=0,m2=0;
                m1=INT_MIN;
                m2=INT_MAX;
                vector<int> pq;
                pq.push_back(Lena[i][j-1]);
                pq.push_back(Lena[i][j+1]);
                pq.push_back(Lena[i][j]);

                pq.push_back(Lena[i-1][j-1]);
                pq.push_back(Lena[i-1][j]);
                pq.push_back(Lena[i-1][j+1]);

                pq.push_back(Lena[i+1][j-1]);
                pq.push_back(Lena[i+1][j]);
                pq.push_back(Lena[i+1][j+1]);

                for (int k=0;k<pq.size();k++)
                {
                    if(i==2 && j==2)
                    {
                        //cout<<pq[k]<<" ";
                        // cout<<"TESTING 5";
                    }
                    m1=max(m1,pq[k]);
                    m2=min(m2,pq[k]);
```

```
                }
                out_Lena[i-1][j-1]=(m1+m2)/2;

            }

        }

        MPI_Send(&index, 1, MPI_INT, MASTER, 5, MPI_COMM_WORLD);
        MPI_Send(&no_elements_iter, 1, MPI_INT, MASTER, 6,
MPI_COMM_WORLD);

        MPI_Send(&out_Lena[index][0], no_elements_iter*512, MPI_INT,
MASTER, 7, MPI_COMM_WORLD);

    }
    MPI_Finalize();
}

void printfile()
{
        FILE *fp;
        fp = fopen("lena_out.txt", "w");
           for(int i=0;i<512;i++)
           {
               for(int j=0;j<512;j++)
               {
                   if(j!=511)
                   {
                       fprintf(fp,"%d,",out_Lena[i][j]);
                   }
                   else{
                       fprintf(fp,"%d\n",out_Lena[i][j]);
                   }
               }
           }

        fclose(fp);
}
int main(int argc, char *argv[]){
    memset(Lena,0,sizeof(Lena));
```

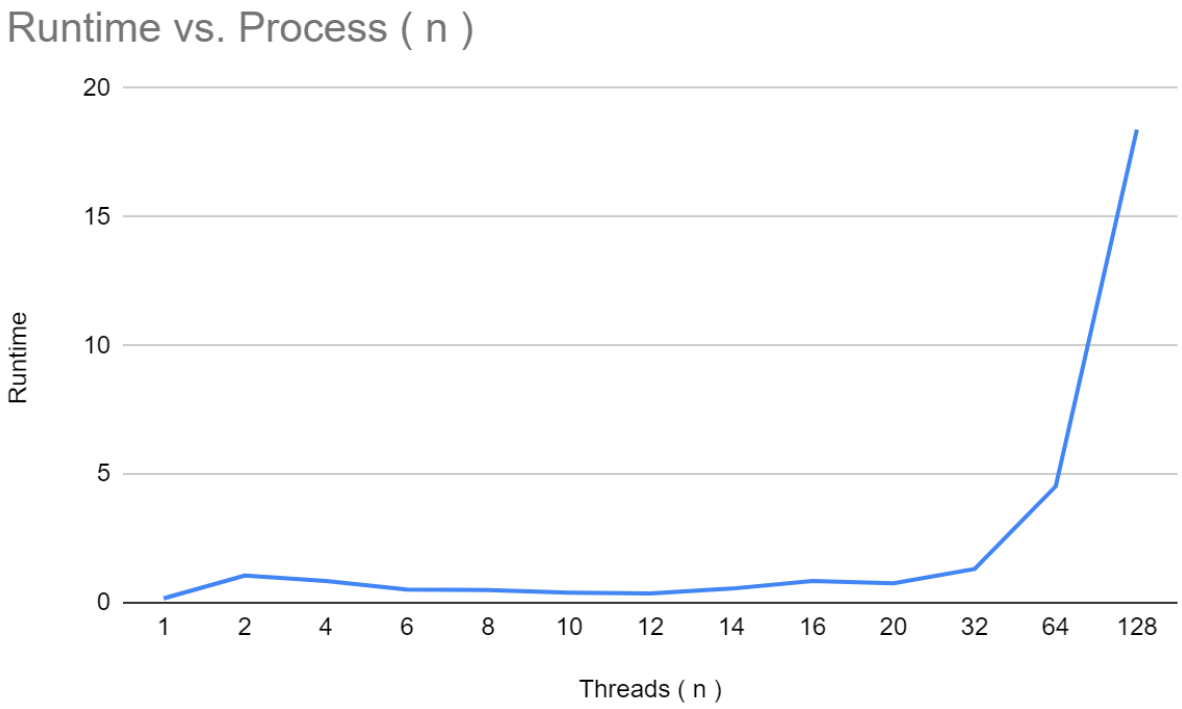
```
    ifstream file("lena.txt");
    string s,x;
    int i=1,j=1;
// string to int
    while(getline(file,s))
    {
        j=1;
        stringstream ss(s);
        while(getline(ss,x,','))
        {
            Lena[i][j] = std::stoi(x);
            j++;
        }
        i++;
    }

//padding
   for(int j=0;j<514;j++)
   {
      Lena[0][j]=0;
      Lena[j][0]=0;
      Lena[513][j]=0;
      Lena[j][513]=0;
   }

    file.close();

    process_Lena(argc,argv);
    return 0;
}
```

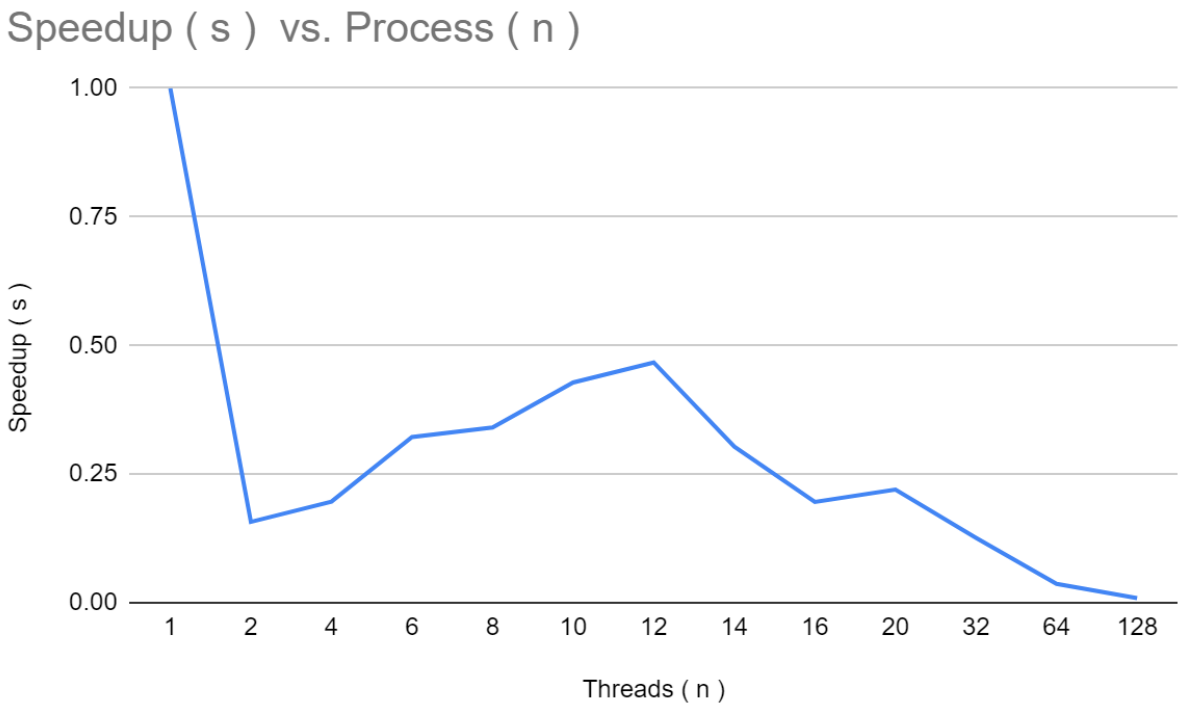
Paleti Krishnasai
CED18I039

**Observations:**

| Processes ( n ) | Runtime | Speedup ( s ) | Parallelization Fraction |
|---|---|---|---|
| 1 | 0.165801 | 1 | |
| 2 | 1.054621 | 0.1572138237 | -10.72152761 |
| 4 | 0.843701 | 0.1965163014 | -5.45151517 |
| 6 | 0.515028 | 0.3219261865 | -2.52756256 |
| 8 | 0.486628 | 0.3407140567 | -2.211442805 |
| 10 | 0.387321 | 0.4280712897 | -1.484510548 |
| 12 | 0.355016 | 0.4670240215 | -1.244964528 |
| 14 | 0.546584 | 0.3033403832 | -2.47329027 |
| 16 | 0.846557 | 0.1958533212 | -4.37958597 |
| 20 | 0.754608 | 0.2197180523 | -3.738197249 |
| 32 | 1.314623 | 0.1261205684 | -7.152434389 |
| 64 | 4.528085 | 0.03661614126 | -26.72798477 |
| 128 | 18.390312 | 0.00901567086 | -110.7834759 |

Paleti Krishnasai
CED18I039

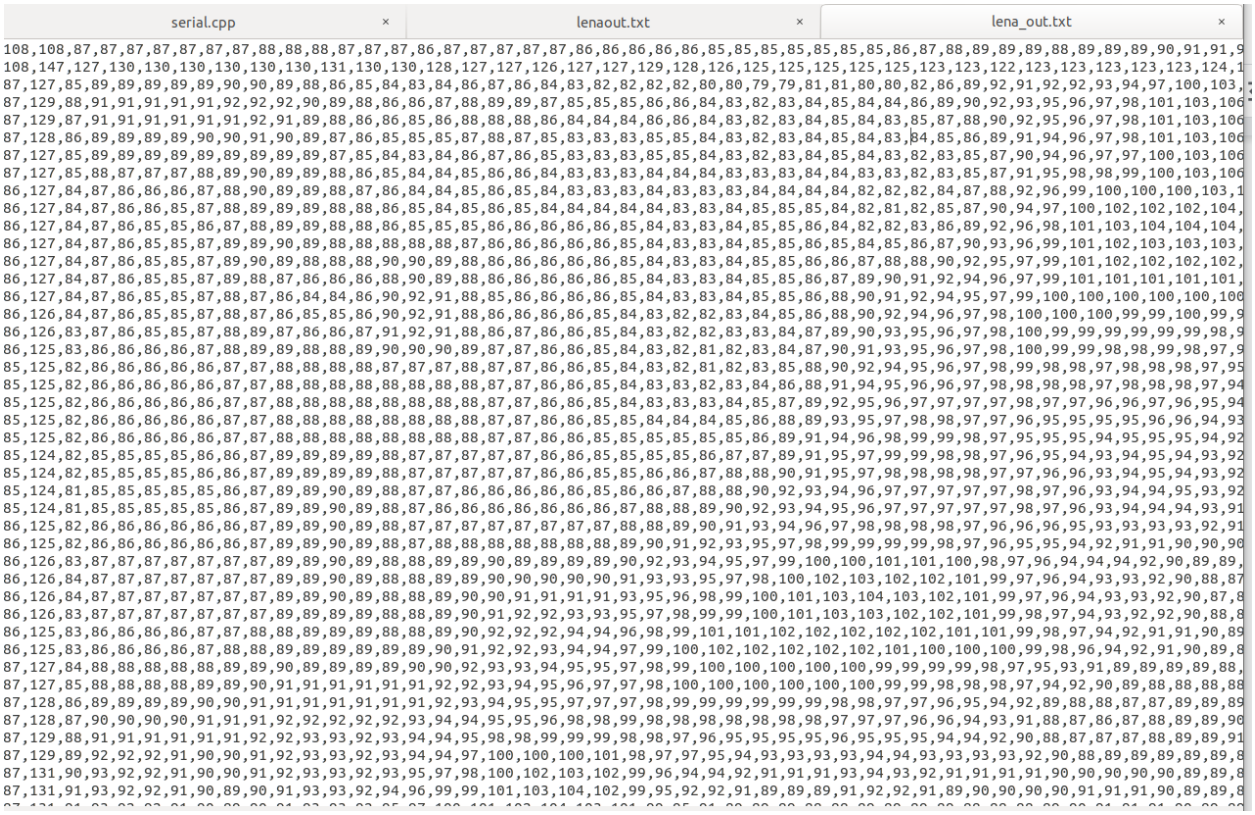


Paleti Krishnasai
CED18I039

**Output File Serial Code :**

**lena_out.txt**



Paleti Krishnasai
CED18I039

**Output File MPI Code :**

**lenaout.txt**


**Instructions to Execute:**

mpic++ serial.cpp -o serial

ubuntuhpc@c01:~/mirror$ mpirun -n 1 -f machinefile ./serial
execution time = 0.165801

mpic++ main.cpp -o main

ubuntuhpc@c01:~/mirror$ mpirun -n 4 -f machinefile ./main
Execution Time :0.853682