**PONTIFICIA UNIVERSIDAD CATOLICA MADRE Y MAESTRA**



**Nombre:**

Félix Alejandro Guzmán 2014-0565

**Materia:**

Programación 2

**Profesor:**

JOSÉ L. ALONSO O.

**Reporte sobre:**

Mergesort Paralelo

#include <algorithm>

#include <vector>

#include "mpi.h"

#include <iostream>

using namespace std;

int main(int argc, char \*argv[])

{

int rank, size, tama;

vector<int> Original;

vector<int> \*Local;

MPI\_Init(&argc, &argv);

MPI\_Comm\_rank(MPI\_COMM\_WORLD,&rank);

MPI\_Comm\_size(MPI\_COMM\_WORLD,&size);

if( size % 2 != 0 )

{

MPI\_Abort(MPI\_COMM\_WORLD,0);

}

if(argc < 2){

if(rank == 0)

{

tama = 1000;

}

}else

{

tama = atoi(argv[1]);

}

if(rank == 0)

{

for(int i = 0; i < tama;++i)

{

Original.push\_back(rand()%1000);

}

}

Local = new vector<int>(tama/size);

MPI\_Scatter(&Original[0],tama/size,MPI\_INT,&((\*Local)[0]),tama/size,MPI\_INT,0,MPI\_COMM\_WORLD);

sort(Local->begin(),Local->end());

vector<int> \*ordenado;

MPI\_Status status;

int paso = 1;

while(paso<size)

{

if(rank%(2\*paso)==0)

{

if(rank+paso<size)

{

vector<int> localVecino(Local->size());

ordenado = new vector<int>(Local->size()\*2);

MPI\_Recv(&localVecino[0],localVecino.size(),MPI\_INT,rank+paso,0,MPI\_COMM\_WORLD,&status);

merge(Local->begin(),Local->end(),localVecino.begin(),localVecino.end(),ordenado->begin() );

delete Local;

Local = ordenado;

ordenado = NULL;

}

}

else

{

int vecino = rank-paso;

MPI\_Send(&((\*Local)[0]),Local->size(),MPI\_INT,vecino,0,MPI\_COMM\_WORLD);

break;

}

paso = paso\*2;

}

if(rank == 0){

cout<<endl<<"[";

for(unsigned int i = 0; i<Local->size();++i){

cout<< (\*Local)[i]<<" , ";

}

cout<<"]"<<endl;

}

MPI\_Finalize();

return 0;

}