6CS005 Learning Journal - Semester 1 2019

Dahal Nancy 1917506

# Table of Contents

[Table of Contents 1](#_Toc8076797)

[1 POSIX Threads 2](#_Toc8076798)

[1.1 Password Cracking 2](#_Toc8076799)

[1.2 Image Processing 20](#_Toc8076800)

[1.3 Linear Regression 53](#_Toc8076801)

[2 CUDA 80](#_Toc8076802)

[2.1 Password Cracking 80](#_Toc8076803)

[2.2 Image Processing 90](#_Toc8076804)

[2.3 Linear Regression 120](#_Toc8076805)

[3 MPI 154](#_Toc8076806)

[3.1 Password Cracking 154](#_Toc8076807)

[3.2 Image Processing 166](#_Toc8076808)

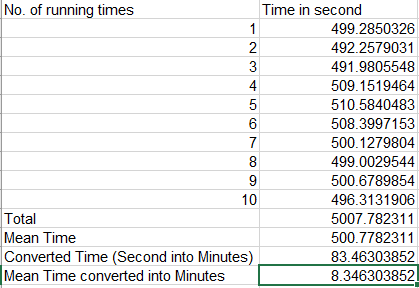
[3.3 Linear Regression 204](#_Toc8076809)

[4 Verbose Repository Log 232](#_Toc8076810)

# POSIX Threads

## Password Cracking

1. Run the program 10 times and calculate the mean running time.



The above table shows the mean running time of the original program which has two initials two digit password and its mean running time is found to be 8.346303852 minutes.

1. In your learning journal make an estimate of how long it would take to run on the same computer if the number of initials were increased to 3. Insert your working in your answer.

Answer:

As mentioned in the above table the mean running time for the normal program is 500.7782311s which is 8.346303852 minutes. But when we increase the initial to three we have “CCC99” which is a three initial five digits password that means it automatically increases 26 characters from A to Z. So, the added loop will run for 26 times because there are 26 characters.

Time for three initial = Mean running time \* 26

= 8.346303852 \* 26

= 217.003900152 minutes

= 3.6167316692 hours

Therefore, the estimates time to run the two initial three-digit password is 3.6167316692 hours.

1. Modify the program to crack the three-initials-two-digits password given in the three\_initials variable. An example password is KAB99.

Answer:

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <crypt.h>

#include <time.h>

#include <math.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Demonstrates how to crack an encrypted password using a simple

"brute force" algorithm. Works on passwords that consist only of 2 uppercase

letters and a 3 digit integer. Your personalised data set is included in the

code.

Compile with:

cc -o CrackCCC99-With-Data 1\_C\_ThreeInitial.c -lcrypt

If you want to analyse the results then use the redirection operator to send

output to a file that you can view using an editor or the less utility:

./CrackCCC99-With-Data > results.txt

The password used is CCC99

Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int n\_passwords = 1;

// encrypted password hash function in array

char \*encrypted\_passwords[] = {

"$6$KB$HHtmO1Kn/YG/835DhBsg8i8.R5z6QiToKTUA5UOZ4Q3BeW8EGYALj1reFInGCdDwEQyx5v8mIrEuvM1S0xTHy."

};

void substr(char \*dest, char \*src, int start, int length){

memcpy(dest, src + start, length);

\*(dest + length) = '\0';

}

/\*\*

This function can crack the kind of password explained above. All

combinations

that are tried are displayed and when the password is found, #, is put

at the

start of the line. Note that one of the most time consuming operations

that

it performs is the output of intermediate results, so performance

experiments

for this kind of program should not include this. i.e. comment out the

printfs.

\*/

void crack(char \*salt\_and\_encrypted){

int a1, b2, c3, digit; // Loop counters

char salt[7]; // String used in hashing the password. Need space

char plain[7]; // The combination of letters currently being checked

char \*enc; // Pointer to the encrypted password

int count = 0; // The number of combinations explored so far

substr(salt, salt\_and\_encrypted, 0, 6);

for(a1='A'; a1<='Z'; a1++){ // loop for first initial

for(b2='A'; b2<='Z'; b2++){// loop for second initial

for(c3='A'; c3<='Z'; c3++){ // loop for third initial

for(digit=0; digit<=99; digit++){ //loop for last 2 digits 00 to 99

sprintf(plain, "%c%c%c%02d", a1, b2, c3, digit);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(salt\_and\_encrypted, enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

printf(" %-8d%s %s\n", count, plain, enc);

}

}

}

}

}

printf("%d solutions explored\n", count);

}

//time calculation fucnction

int time\_difference(struct timespec \*start,

struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

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

int i;

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

for(i=0;i<n\_passwords;i<i++) {

crack(encrypted\_passwords[i]);

}

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed,

(time\_elapsed/1.0e9));

return 0;

}

1. Write a short paragraph to compare the running time of your three\_initials program with your earlier estimate. If your estimate was wrong explain why you think that is.

Answer: In the earlier question I had estimated the average time taken by the 3 initial 2 digit program was 3.6167316692 hours. Because the running time for the program was nearly increased by 26 times and the actual time it took was 3.5857627783 hours which is nearly the same time taken.

1. Modify the original version of the program to run on 2 threads.

Answer:

/\*\*

run with

gcc -o Posixtwothread 1\_e\_TwoThread.c -lcrypt -pthread

./Posixtwothread > PosixTwoThreadresult.txt

\*\*/

#include <stdlib.h>

#include <stdio.h>

#include <ctype.h>

#include <errno.h>

#include <sys/stat.h>

#include <string.h>

#include <time.h>

#include <math.h>

#include <crypt.h>

#include <pthread.h>

int n\_passwords = 4;

pthread\_t thread\_1, thread\_2; //thread variables

//arrayy of hashed passowrd with salt

char \*encrypted\_passwords[] = {

"$6$KB$0G24VuNaA9ApVG4z8LkI/OOr9a54nBfzgQjbebhqBZxMHNg0HiYYf1Lx/HcGg6q1nnOSArPtZYbGy7yc5V.wP/",

"$6$KB$S8w.zpH4EcQ7.HXep3qUhZMfS6ODZAMrf.UC6BmnWzBWkYKniOzIRDRO4XjWqQY0xbQiAFzFwO2Xx7ZGjoKD90",

"$6$KB$dL/UJksvXIob3FSHXkSPN.gJ.racNc7al3B5oIoCeH0jwbCGFxJQ4qkLhjGBvtDF4TVnOczq4pz8ZW/WmBpX50",

"$6$KB$BU6OM3QHROtDFEX259Z7GM6zCH9USl3aVVwsbmGQX2qfprJ/sE8iwPc3/.lUA0nzAObmL7tKREfHDBlDIdrK5/"

};

void substr(char \*dest, char \*src, int start, int length){

memcpy(dest, src + start, length);

\*(dest + length) = '\0';

}

//kernal function 1 that iterates overs password that starts from A to M

void \*kernel\_function\_1(void \*salt\_and\_encrypted){

int a1, b2, c3; // Loop counters

char salt[7]; // String used in hashing the password. Need space

char plain[7]; // The combination of letters currently being checked

char \*enc; // Pointer to the encrypted password

int count = 0; // The number of combinations explored so far

substr(salt, salt\_and\_encrypted, 0, 6);

for(a1='A'; a1<='M'; a1++){ //iiterates overs password that starts from A to M

for(b2='A'; b2<='Z'; b2++){

for(c3=0; c3<=99; c3++){

sprintf(plain, "%c%c%02d", a1, b2, c3);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(salt\_and\_encrypted, enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

//printf(" %-8d%s %s\n", count, plain, enc);

}

}

}

}

printf("%d solutions explored\n", count);

}

//kernal function 2 that iterates overs password that starts from N to Z

void \*kernel\_function\_2(void \*salt\_and\_encrypted){

int a1, b2, c3; // Loop counters

char salt[7]; // String used in hashing the password. Need space

char plain[7]; // The combination of letters currently being checked

char \*enc; // Pointer to the encrypted password

int count = 0; // The number of combinations explored so far

substr(salt, salt\_and\_encrypted, 0, 6);

for(a1='N'; a1<='Z'; a1++){ //iterates overs password that starts from N to Z

for(b2='A'; b2<='Z'; b2++){

for(c3=0; c3<=99; c3++){

sprintf(plain, "%c%c%02d", a1, b2, c3);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(salt\_and\_encrypted, enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

// printf(" %-8d%s %s\n", count, plain, enc);

}

}

}

}

printf("%d solutions explored\n", count);

}

///time difference code

int time\_difference(struct timespec \*start,

struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

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

int i;

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

//pthread create and pass kernal function to each thread

for(i=0;i<n\_passwords;i<i++) {

pthread\_create(&thread\_1, NULL, kernel\_function\_1, encrypted\_passwords[i]);

pthread\_create(&thread\_2, NULL, kernel\_function\_2, encrypted\_passwords[i]);

pthread\_join(thread\_1, NULL);

pthread\_join(thread\_2, NULL);

}

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed,

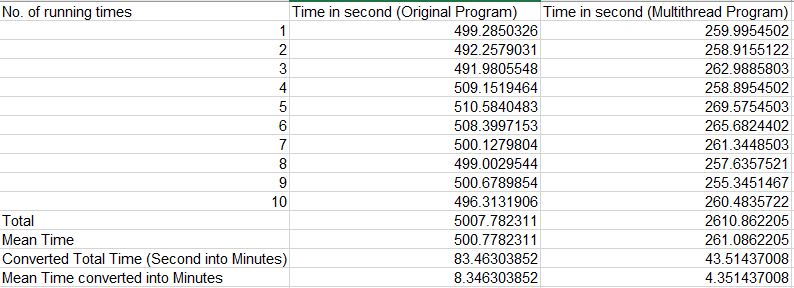
(time\_elapsed/1.0e9));

return 0;

}

1. Compare the results of the mean running time of the original program with the mean running time of the multithread version.

Answer:



The mean time taken by two initial original program was 8.346303852 minutes, similarly on the other side the multi-thread version took 4.351437008 minutes which is half of the time from the original program. Because the two initial original program cracks one password at one time whereas the multi-thread program cracks password in parallel way, so it take less time comparing with the original program.

## Image Processing

1. Run the program and capture the resulting image to put in your learning journal.

Answer:



1. Code for multithread image detector

Answer:

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <GL/glut.h>

#include <GL/gl.h>

#include <malloc.h>

#include <signal.h>

#include <pthread.h>

#include <ctype.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Displays two grey scale images. On the left is an image that has come from an

image processing pipeline, just after colour thresholding. On the right is

the result of applying an edge detection convolution operator to the left

image. This program performs that convolution.

Things to note:

- A single unsigned char stores a pixel intensity value. 0 is black, 256 is

white.

- The colour mode used is GL\_LUMINANCE. This uses a single number to

represent a pixel's intensity. In this case we want 256 shades of grey,

which is best stored in eight bits, so GL\_UNSIGNED\_BYTE is specified as

the pixel data type.

To compile adapt the code below wo match your filenames:

gcc -o Multithread\_ip\_coursework\_074 2c\_Multithread.c -lglut -lGL -lm -pthread

./Multithread\_ip\_coursework\_074

Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define width 100

#define height 72

unsigned char image[], results[width \* height];

typedef struct arguments {

unsigned char \*input;

unsigned char \*output;

int start;

int stride;

}

arguments\_t;

void edges(unsigned char \*image,unsigned char \*results) {

pthread\_t t1, t2, t3, t4;

arguments\_t t1\_arguments;

t1\_arguments.start = 0;

t1\_arguments.stride = 4;

t1\_arguments.input = image;

t1\_arguments.output = results;

arguments\_t t2\_arguments;

t2\_arguments.start = 1;

t2\_arguments.stride = 4;

t2\_arguments.input = image;

t2\_arguments.output = results;

arguments\_t t3\_arguments;

t3\_arguments.start = 2;

t3\_arguments.stride = 4;

t3\_arguments.input = image;

t3\_arguments.output = results;

arguments\_t t4\_arguments;

t4\_arguments.start = 3;

t4\_arguments.stride = 4;

t4\_arguments.input = image;

t4\_arguments.output = results;

void \*detect\_edges();

pthread\_create(&t1, NULL, detect\_edges, &t1\_arguments);

pthread\_create(&t2, NULL, detect\_edges, &t2\_arguments);

pthread\_create(&t3, NULL, detect\_edges, &t3\_arguments);

pthread\_create(&t4, NULL, detect\_edges, &t4\_arguments);

pthread\_join(t1, NULL);

pthread\_join(t2, NULL);

pthread\_join(t3, NULL);

pthread\_join(t4, NULL);

}

void \*detect\_edges(arguments\_t \*args) {

int i;

unsigned char \*in = args->input;

unsigned char \*out =args->output;

int n\_pixels = width \* height;

for(i=args->start;i<n\_pixels;i+=args->stride) {

int x, y; // the pixel of interest

int b, d, f, h; // the pixels adjacent to x,y used for the calculation

int r; // the result of calculate

y = i / width;

x = i - (width \* y);

if (x == 0 || y == 0 || x == width - 1 || y == height - 1) {

results[i] = 0;

} else {

b = i + width;

d = i - 1;

f = i + 1;

h = i - width;

r = (in[i] \* 4) + (in[b] \* -1) + (in[d] \* -1) + (in[f] \* -1)

+ (in[h] \* -1);

if (r > 0) { // if the result is positive this is an edge pixel

out[i] = 255;

} else {

out[i] = 0;

}

}

}

}

int time\_difference(struct timespec \*start, struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

void tidy\_and\_exit() {

exit(0);

}

void sigint\_callback(int signal\_number){

printf("\nInterrupt from keyboard\n");

tidy\_and\_exit();

}

static void display() {

glClear(GL\_COLOR\_BUFFER\_BIT);

glRasterPos4i(-1, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image);

glRasterPos4i(0, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results);

glFlush();

}

static void key\_pressed(unsigned char key, int x, int y) {

switch(key){

case 27: // escape

tidy\_and\_exit();

break;

default:

printf("\nPress escape to exit\n");

break;

}

}

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

signal(SIGINT, sigint\_callback);

printf("image dimensions %dx%d\n", width, height);

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

edges(image, results);

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed,

(time\_elapsed/1.0e9));

glutInit(&argc, argv);

glutInitWindowSize(width \* 2,height);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_LUMINANCE);

glutCreateWindow("6CS005 Image Progessing Courework");

glutDisplayFunc(display);

glutKeyboardFunc(key\_pressed);

glClearColor(0.0, 1.0, 0.0, 1.0);

glutMainLoop();

tidy\_and\_exit();

return 0;

}

unsigned char image[] = {255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,0,255,255,0,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,0,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,

0,0,0,0,0,0,0,255,255,255,0,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,0,0,0,0,0,0,0,255,255,255,0,0,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,0,255,0,0,0,0,0,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,

0,0,0,255,255,255,0,0,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,0,0,0,255,255,0,0,0,0,0,0,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,0,0,0,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,0,0,0,0,255,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,0,0,

0,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,255,

255,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,0,0,0,0,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,255,255,255,0,0,0,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,0,0,

0,0,0,0,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,255,255,255,0,0,0,0,0,255,255,255,255,0,0,0,255,255,255,

255,0,0,0,255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,

255,255,255,0,0,0,0,255,255,0,0,0,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,

255,255,0,0,0,0,255,255,255,255,255,0,0,0,255,255,255,255,0,

0,0,0,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,255,255,255,255,255,0,0,0,255,255,0,0,0,255,255,

255,255,255,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,255,

255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,

0,255,255,255,255,255,255,0,0,0,0,255,255,255,255,255,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,

0,0,0,0,255,255,0,0,0,0,255,255,255,255,0,0,0,0,255,

255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,255,255,255,0,0,0,0,255,255,255,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,255,255,255,255,0,0,0,0,0,255,

255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,255,255,255,255,

0,0,0,0,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,0,

0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,255,255,0,0,

0,0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,0,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,255,0,0,0,

255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,0,0,0,

0,255,255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,255,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,255,255,0,0,0,255,255,255,255,255,255,0,0,

255,255,255,0,0,0,255,255,255,255,0,0,0,0,0,255,255,255,255,

255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,255,0,0,0,

0,255,255,255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,255,0,

0,0,255,255,255,255,255,255,0,0,0,255,255,0,0,0,255,255,255,

255,255,0,0,0,255,255,255,0,0,0,255,255,255,255,0,0,0,0,

0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,0,255,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,0,0,

0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,255,

0,0,0,255,255,255,255,0,0,0,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,

0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,255,0,0,0,

0,0,0,0,0,255,255,0,0,0,255,255,255,255,255,0,0,0,255,

255,255,255,0,0,0,0,255,255,255,0,0,0,0,255,255,255,0,0,

0,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,0,

0,0,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,0,

0,0,0,255,255,0,0,0,0,255,255,0,0,0,0,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,255,255,255,255,0,0,255,255,255,255,255,0,0,0,0,0,255,255,

255,255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,0,0,0,

0,0,0,0,0,255,255,0,0,0,0,255,0,0,0,0,255,255,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,255,255,

255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,255,255,255,255,255,0,0,255,255,255,255,255,0,0,

0,0,0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,

255,255,0,0,0,0,0,0,0,0,255,255,0,0,0,0,255,255,0,

0,0,255,255,255,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,255,255,255,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,255,

255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,

255,0,0,0,0,0,255,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,

0,0,0,0,0,255,255,0,0,0,0,255,255,255,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,255,255,255,255,255,255,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255,255,

0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,

0,0,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,255,0,0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255,

255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,0,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,0,255,255,255,255,255,255,255,255,

255,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,

0,0,0,255,255,0,255,255,0,255,255,255,255,255,255,255,255,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

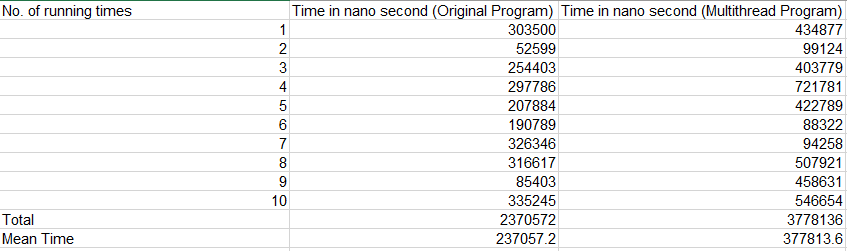
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0

};

1. Compare the relative running tomes of the original edge detection program, with the multithread one.

Answer:

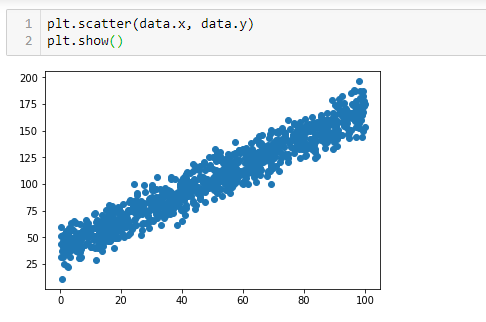


According to the above table the mean time of the original program is 237057.2ns whereas the multithread program mean time is 377813.6ns which shows that it takes more time to run, because the multithread program has four pixels in parallel which means it processes each thread. So, the running time is more than the original program.

## Linear Regression

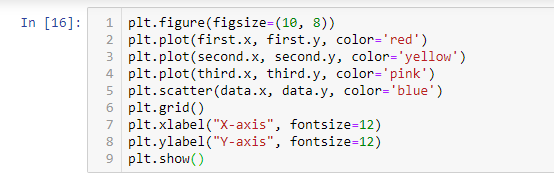
1. Do a scatter plot of your dataset.

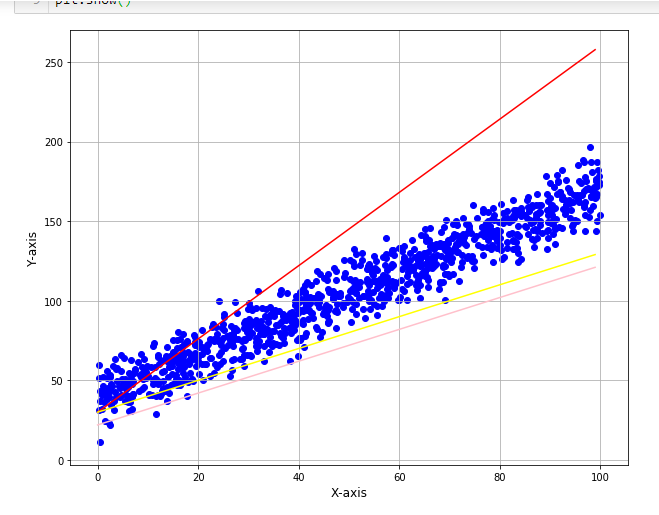
Answer:



1. Have three guesses at the optimum values for m and c and plot them.

Answer:



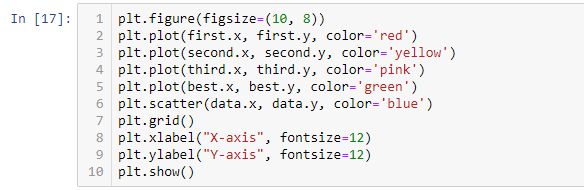


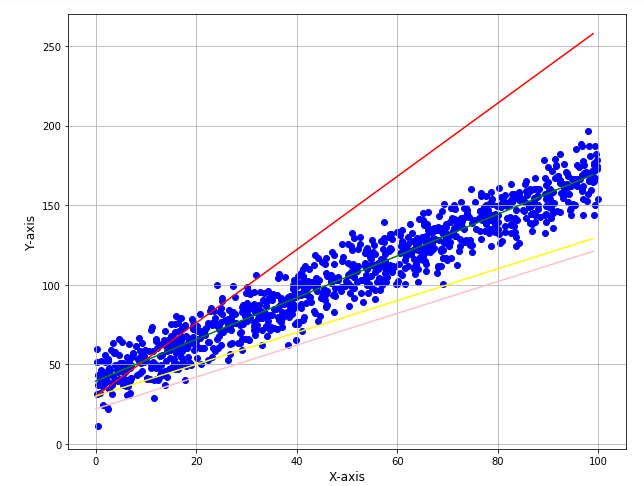
In the above graph

* Red color represents first guess which m = 2.3 and c = 30
* Yellow color represent second guess with m = 1 and c = 30
* Pink color represent third guess with m = 1 and c = 22

1. Run the program to see what solution it finds. Overlay the line that was found by the program on to a dataset scatter plot and comment on the solution.

Answer:



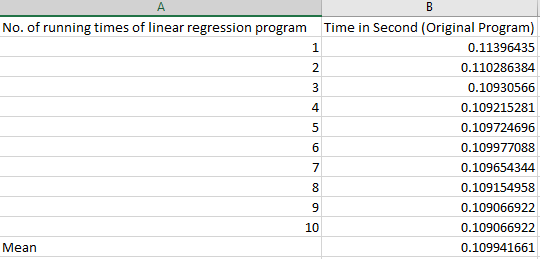


As shown in the above figure,

* Red color represents first guess which m = 2.3 and c = 30
* Yellow color represent second guess with m = 1 and c = 30
* Pink color represent third guess with m = 1 and c = 22
* Green color represent best guess with m = 1.31 and c = 39.46

1. Remove any extraneous print statements from the program and find its mean running time.

Answer:



1. Create a modified version of the program that performs each of the evaluations on a different thread.

Answer:

#include <stdio.h>

#include <math.h>

#include <pthread.h>

#include <time.h>

#define n\_threads 8

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This program takes an initial estimate of m and c and finds the associated

\* rms error. It is then as a base to generate and evaluate 8 new estimates,

\* which are steps in different directions in m-c space. The best estimate is

\* then used as the base for another iteration of "generate and evaluate". This

\* continues until none of the new estimates are better than the base. This is

\* a gradient search for a minimum in mc-space.

\*

\* To compile:

\* cc -o 3\_e\_MultiThread 3\_e\_MultiThread.c -lm -pthread

\*

\* To run:

\* ./3\_e\_MultiThread >3\_e\_MultiThread.csv

\*

\* Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int i;

double bm = 1.3;

double bc = 10;

double be;

double dm[8];

double dc[8];

double e[8];

double step = 0.01;

double best\_error = 999999999;

int best\_error\_i;

int minimum\_found = 0;

double om[] = {0,1,1, 1, 0,-1,-1,-1};

double oc[] = {1,1,0,-1,-1,-1, 0, 1};

typedef struct point\_t {

double x;

double y;

} point\_t;

int n\_data = 1000;

point\_t data[];

double residual\_error(double x, double y, double m, double c) {

double e = (m \* x) + c - y;

return e \* e;

}

double rms\_error(double m, double c) {

int i;

double mean;

double error\_sum = 0;

for(i=0; i<n\_data; i++) {

error\_sum += residual\_error(data[i].x, data[i].y, m, c);

}

mean = error\_sum / n\_data;

return sqrt(mean);

}

void \*linearRegression\_thread(void \*args)

{

int \*a = args;

int i = \*a;

//printf("\nThread number: %d", i);

dm[i] = bm + (om[i] \* step);

dc[i] = bc + (oc[i] \* step);

e[i] = rms\_error(dm[i], dc[i]);

if(e[i] < best\_error) {

best\_error = e[i];

best\_error\_i = i;

}

}

// Calculate the difference between two times. Returns zero on

// success and the time difference through an argument. It will

// be unsuccessful if the start time is after the end time.

int time\_difference(struct timespec \*start,

struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

int main() {

int i;

be = rms\_error(bm, bc);

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

pthread\_t p\_threads[8];

be = rms\_error(bm, bc);

while(!minimum\_found) {

for(i=0;i<8;i++) {

pthread\_create(&p\_threads[i], NULL, (void\*)linearRegression\_thread, &i);

pthread\_join(p\_threads[i], NULL);

}

// printf("\nBest m,c is %lf,%lf with error %lf in direction %d\n", dm[best\_error\_i], dc[best\_error\_i], best\_error, best\_error\_i);

if(best\_error < be) {

be = best\_error;

bm = dm[best\_error\_i];

bc = dc[best\_error\_i];

} else {

minimum\_found = 1;

}

}

printf("\nMinimum m,c is %lf,%lf with error %lf\n", bm, bc, be);

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

//printf("\nTime elapsed was %lldns or %0.9lfs or %0.9lfmin\n", time\_elapsed, (time\_elapsed/1.0e9), ((time\_elapsed/1.0e9)/60));

return 0;

}

point\_t data[] = {

{65.06,140.50},{68.29,128.16},{65.16,115.78},{77.08,144.62},

{80.58,148.61},{69.14,116.53},{65.57,114.91},{82.72,137.98},

{83.65,124.22},{79.38,144.31},{78.26,132.38},{87.08,145.27},

{94.45,147.08},{92.35,164.15},{29.71,80.37},{ 4.60,55.88},

{93.22,160.01},{41.28,104.89},{70.80,148.31},{65.29,130.26},

{99.15,171.42},{ 3.63,57.34},{24.39,60.47},{ 5.30,63.83},

{83.72,129.70},{ 7.17,54.62},{ 2.55,51.94},{81.91,149.27},

{65.84,125.98},{52.85,102.69},{53.55,111.53},{68.67,133.94},

{71.18,133.57},{31.95,72.01},{68.73,111.86},{72.26,120.95},

{69.11,100.24},{57.60,107.51},{93.09,163.24},{97.86,152.47},

{76.21,142.09},{57.24,122.45},{54.07,109.88},{83.78,140.43},

{24.99,76.38},{51.46,97.96},{73.46,126.86},{84.30,126.29},

{12.04,47.93},{ 6.50,54.26},{33.87,73.33},{79.90,149.55},

{25.17,81.80},{37.22,79.55},{38.65,89.41},{39.91,100.49},

{76.60,157.45},{18.16,60.40},{65.87,126.29},{71.88,112.63},

{70.80,116.68},{80.05,135.15},{71.60,130.84},{92.36,182.25},

{81.46,133.35},{47.37,82.88},{78.32,140.80},{35.59,84.76},

{98.55,159.78},{69.63,136.22},{ 6.85,31.94},{68.92,135.57},

{39.62,75.64},{45.22,95.17},{56.10,113.68},{33.40,90.58},

{51.25,113.20},{31.87,71.31},{91.54,174.46},{34.49,91.73},

{49.83,102.67},{16.86,48.98},{30.06,88.34},{98.04,196.30},

{31.25,86.63},{38.39,91.43},{36.81,90.52},{29.41,87.37},

{30.49,84.56},{23.21,74.83},{86.64,149.79},{98.24,187.36},

{44.74,80.44},{89.56,162.75},{88.07,155.61},{65.31,139.57},

{ 5.98,36.29},{85.06,153.94},{32.95,79.00},{43.81,101.89},

{62.22,111.47},{18.81,50.86},{65.82,113.31},{34.13,68.03},

{28.85,70.54},{71.38,141.34},{35.08,81.06},{41.36,109.69},

{40.12,76.78},{77.54,133.76},{50.18,106.00},{69.88,128.59},

{85.04,147.04},{66.82,128.00},{ 4.62,65.65},{78.07,145.26},

{33.81,86.56},{30.94,86.44},{78.63,158.12},{30.00,94.75},

{68.43,134.65},{66.31,122.03},{72.19,140.43},{43.52,112.56},

{58.36,129.89},{63.49,109.82},{72.93,140.32},{ 0.91,32.13},

{92.36,166.03},{58.59,132.02},{19.62,65.12},{73.70,127.87},

{ 7.10,49.29},{96.54,188.39},{11.72,29.18},{85.16,146.33},

{86.71,149.72},{82.62,137.78},{ 0.54,36.93},{43.02,101.71},

{56.52,96.81},{83.48,141.45},{87.95,150.02},{41.46,79.96},

{14.22,41.84},{ 6.09,45.81},{70.28,124.44},{70.57,126.73},

{21.65,60.07},{55.38,112.19},{ 1.10,46.57},{99.19,175.35},

{20.13,74.04},{36.30,79.86},{15.00,64.84},{55.51,111.39},

{39.08,81.62},{26.40,52.68},{68.19,136.41},{15.15,66.96},

{10.81,41.02},{45.70,113.27},{42.56,109.98},{80.27,131.75},

{45.41,95.82},{31.53,73.92},{40.84,93.37},{54.49,100.20},

{98.15,165.46},{ 8.63,51.82},{ 2.49,53.66},{73.88,147.16},

{13.36,63.33},{64.39,141.51},{84.32,157.95},{67.96,107.69},

{ 6.78,39.40},{77.95,141.91},{51.91,109.56},{47.62,92.81},

{27.17,63.62},{61.71,117.83},{38.62,86.65},{60.72,124.18},

{67.81,134.12},{31.53,71.56},{14.00,61.96},{19.50,59.01},

{90.95,135.89},{91.03,158.12},{33.65,82.03},{12.33,39.88},

{83.04,132.29},{50.85,112.28},{70.55,139.16},{64.07,121.96},

{35.31,92.95},{ 3.02,59.14},{46.36,105.52},{ 9.08,51.06},

{97.16,170.35},{ 5.22,48.76},{37.20,90.63},{60.93,127.67},

{89.32,178.38},{69.67,138.27},{19.36,63.13},{33.04,71.81},

{39.92,65.58},{50.92,124.15},{26.92,77.66},{29.14,82.75},

{ 0.85,40.60},{27.69,81.38},{81.94,142.14},{31.80,82.99},

{32.47,87.54},{ 4.43,39.37},{86.44,160.01},{62.87,132.53},

{48.20,103.22},{95.55,163.90},{82.81,152.15},{41.02,71.00},

{51.46,91.99},{42.61,92.40},{ 9.37,55.95},{12.68,62.08},

{41.45,102.49},{67.42,123.38},{83.35,157.06},{25.04,62.00},

{17.95,67.64},{74.23,142.58},{ 2.20,42.77},{85.41,152.29},

{63.83,111.25},{35.59,72.75},{ 9.84,59.26},{39.47,99.05},

{16.96,54.75},{27.80,92.37},{68.88,146.45},{80.19,160.65},

{41.22,99.46},{47.87,102.96},{30.06,77.15},{13.92,61.54},

{40.95,107.20},{48.85,98.44},{20.23,71.85},{37.48,87.76},

{46.93,100.97},{91.46,166.75},{66.78,127.65},{69.80,128.92},

{56.99,113.89},{64.99,118.91},{60.11,133.15},{11.00,71.54},

{ 8.09,61.76},{33.73,89.91},{14.97,47.57},{ 1.82,55.82},

{55.44,124.87},{18.43,54.83},{51.17,107.27},{14.12,58.21},

{61.49,123.07},{64.13,109.73},{42.54,92.26},{65.15,144.90},

{53.75,105.92},{57.63,114.92},{13.89,45.67},{52.03,101.50},

{63.30,116.07},{89.52,143.82},{63.83,113.06},{80.08,143.83},

{ 6.92,47.55},{57.40,103.92},{64.21,125.58},{46.81,106.30},

{91.30,151.08},{76.66,155.83},{47.48,112.92},{54.00,99.69},

{40.35,94.01},{77.68,150.21},{87.39,143.66},{76.78,146.43},

{54.80,105.64},{23.96,75.02},{45.90,86.75},{17.85,40.30},

{44.08,99.76},{62.84,119.24},{ 4.66,40.51},{61.01,125.84},

{96.70,168.86},{54.75,112.30},{ 8.70,53.95},{95.79,174.98},

{99.36,150.00},{69.48,127.58},{44.39,102.81},{43.75,98.47},

{45.91,95.15},{11.07,50.20},{85.58,141.73},{26.65,72.63},

{16.74,71.23},{43.53,83.72},{ 1.76,42.14},{23.78,79.11},

{26.59,84.39},{43.96,99.13},{47.07,120.85},{90.53,150.59},

{63.14,121.81},{17.75,73.96},{38.85,90.89},{77.91,145.78},

{64.09,115.44},{90.91,171.76},{53.97,123.30},{36.48,77.64},

{91.35,179.25},{70.46,131.28},{84.31,156.60},{53.08,103.03},

{26.80,62.81},{74.76,160.33},{21.42,60.42},{77.34,148.78},

{17.14,58.22},{69.61,127.58},{90.26,159.43},{69.55,132.52},

{82.69,132.50},{96.81,158.98},{99.48,182.04},{34.32,96.27},

{ 9.71,55.14},{46.94,102.50},{71.31,135.96},{66.98,121.76},

{98.22,165.52},{89.51,166.77},{ 6.33,30.69},{65.11,122.80},

{82.86,156.93},{16.40,66.05},{35.25,94.44},{94.61,147.92},

{96.45,166.53},{12.94,45.40},{36.88,76.80},{42.23,102.36},

{38.52,79.49},{28.72,69.83},{ 5.47,47.33},{31.42,76.78},

{89.61,149.18},{54.33,120.28},{29.57,82.48},{ 3.24,37.35},

{39.97,92.40},{38.31,88.57},{64.15,101.51},{66.60,124.22},

{ 8.39,53.51},{53.52,120.09},{54.00,104.21},{69.22,116.75},

{66.90,125.93},{16.18,60.34},{36.52,106.85},{ 3.83,57.76},

{63.60,111.98},{16.08,64.93},{14.51,60.59},{55.61,91.17},

{36.35,104.47},{22.41,63.89},{32.32,78.96},{44.95,91.99},

{12.10,66.00},{76.41,146.29},{57.89,105.09},{22.74,83.64},

{68.93,135.75},{56.80,101.13},{44.07,95.81},{ 2.22,39.62},

{34.64,80.94},{91.59,175.08},{ 7.54,56.50},{90.29,148.76},

{ 0.18,59.80},{59.07,108.26},{24.80,83.05},{58.62,123.01},

{16.02,56.86},{38.71,92.88},{99.87,174.65},{59.02,109.10},

{82.68,153.35},{24.15,99.81},{64.58,117.31},{21.24,69.38},

{30.70,80.02},{ 1.41,40.58},{59.40,111.46},{52.91,94.94},

{80.90,151.48},{ 9.73,51.73},{70.74,131.92},{92.53,157.93},

{93.11,160.17},{78.60,142.44},{39.51,103.00},{27.05,76.90},

{53.77,102.32},{82.71,163.25},{32.13,74.88},{73.27,134.77},

{18.37,66.43},{96.87,160.13},{14.74,59.16},{29.79,66.13},

{81.48,156.13},{50.93,132.81},{34.14,90.11},{97.06,177.46},

{43.59,118.51},{87.77,155.51},{26.78,56.57},{67.37,135.73},

{43.72,82.53},{78.90,151.03},{67.47,128.75},{73.54,142.62},

{83.58,146.88},{60.97,105.98},{61.39,121.73},{97.62,170.99},

{22.87,75.14},{14.33,60.35},{10.18,41.30},{71.87,138.38},

{ 6.67,47.75},{ 0.18,31.24},{63.19,127.26},{30.39,85.15},

{ 5.40,37.14},{30.71,94.33},{80.79,127.64},{60.59,107.25},

{13.56,65.29},{87.52,167.00},{14.84,53.76},{41.12,94.91},

{10.69,58.47},{53.25,89.18},{81.43,152.09},{33.06,79.09},

{15.51,51.17},{79.41,141.15},{70.48,139.28},{94.07,168.01},

{51.12,107.53},{ 5.67,45.68},{58.79,131.63},{19.47,73.96},

{31.67,71.86},{26.69,74.47},{71.45,147.52},{86.49,138.97},

{34.95,73.25},{73.30,148.44},{71.11,133.22},{94.75,166.25},

{88.11,147.28},{19.30,67.43},{90.05,164.25},{15.78,61.88},

{25.32,70.27},{28.72,74.55},{79.45,132.86},{25.88,72.08},

{31.14,86.32},{35.98,84.84},{ 5.71,44.50},{76.81,142.72},

{57.22,111.49},{87.46,139.57},{14.92,62.58},{83.44,141.89},

{23.73,58.30},{50.44,86.37},{20.16,70.79},{38.51,83.48},

{40.26,75.05},{15.57,53.06},{13.83,71.11},{19.29,73.30},

{91.73,146.04},{31.42,95.31},{24.64,72.44},{36.80,91.84},

{30.03,98.86},{11.40,51.19},{ 9.15,45.77},{65.77,120.82},

{90.64,148.02},{49.00,115.15},{33.13,82.17},{11.11,47.41},

{57.31,127.10},{22.37,75.81},{39.67,95.67},{29.11,75.90},

{45.84,86.16},{45.13,92.70},{ 5.48,50.66},{24.90,85.23},

{63.62,106.70},{82.49,142.53},{71.25,141.47},{36.64,87.11},

{ 1.96,51.52},{50.21,125.63},{61.50,110.30},{93.82,163.58},

{95.33,150.35},{ 9.32,63.49},{62.76,120.10},{ 3.44,63.32},

{41.49,97.46},{74.59,128.03},{77.61,137.92},{68.64,121.30},

{62.10,121.86},{34.90,93.88},{30.26,96.06},{85.65,165.07},

{54.94,128.21},{33.16,61.94},{ 4.82,50.78},{14.96,67.98},

{41.74,94.46},{86.33,148.59},{78.50,151.64},{78.74,124.22},

{83.50,154.44},{69.30,150.92},{86.91,133.87},{57.76,101.58},

{ 9.27,58.05},{52.10,108.71},{ 6.70,52.52},{15.06,75.65},

{40.28,100.24},{72.78,141.77},{56.15,109.93},{34.21,91.19},

{46.13,95.81},{90.38,157.10},{79.42,137.37},{23.85,75.78},

{16.94,61.69},{28.30,63.59},{45.94,116.74},{21.90,72.69},

{86.71,144.20},{50.15,101.35},{47.52,104.48},{76.50,133.92},

{16.59,70.95},{30.07,81.69},{95.78,162.28},{79.85,139.57},

{32.16,84.87},{59.81,100.58},{12.73,55.16},{68.40,143.18},

{88.39,142.70},{70.92,141.63},{16.91,65.12},{65.99,138.01},

{29.08,65.78},{94.66,143.54},{17.46,51.83},{40.45,100.97},

{80.61,154.88},{55.99,101.51},{50.64,94.60},{89.71,154.05},

{76.69,128.98},{71.69,135.85},{ 6.01,40.29},{96.97,153.79},

{51.33,104.76},{26.38,78.48},{74.84,124.85},{87.23,154.70},

{43.79,101.79},{36.34,86.15},{93.97,152.26},{91.93,168.48},

{99.95,153.69},{ 6.24,47.10},{33.57,94.58},{25.15,90.07},

{ 2.71,47.98},{55.65,96.69},{56.66,103.53},{15.84,75.38},

{13.10,49.82},{31.59,87.78},{10.07,52.90},{64.69,128.15},

{ 0.29,51.37},{54.63,117.56},{16.48,47.12},{68.05,111.33},

{65.05,117.98},{50.73,105.93},{38.21,92.02},{84.19,152.96},

{60.62,120.01},{71.76,130.65},{99.68,172.79},{16.20,50.34},

{16.99,79.00},{29.19,59.02},{69.67,116.57},{ 2.04,45.22},

{53.52,113.83},{38.61,86.66},{20.82,71.20},{45.51,91.72},

{83.51,133.29},{95.95,166.47},{99.30,187.23},{57.30,129.30},

{54.98,98.10},{60.47,112.79},{57.56,119.35},{18.50,62.58},

{75.27,140.67},{ 5.79,48.06},{94.17,157.17},{25.72,77.59},

{66.55,135.05},{53.82,122.03},{44.12,103.61},{86.49,159.15},

{20.06,64.54},{97.78,170.13},{61.95,119.09},{58.91,128.99},

{75.55,152.04},{28.76,63.60},{61.54,100.40},{74.51,125.14},

{ 6.62,30.47},{45.73,90.74},{79.17,140.18},{42.13,94.38},

{ 3.52,42.69},{20.27,53.88},{47.35,118.66},{64.33,122.29},

{14.34,63.88},{48.22,104.79},{79.32,140.01},{20.79,81.66},

{21.07,49.96},{48.95,125.70},{99.68,178.12},{50.76,107.99},

{59.08,111.82},{32.86,83.53},{37.71,104.74},{ 6.95,41.30},

{61.04,126.11},{23.82,75.61},{ 3.64,39.77},{11.31,73.23},

{65.13,128.77},{45.97,87.42},{51.46,105.02},{50.48,106.56},

{29.63,85.41},{73.86,144.97},{56.80,127.73},{83.76,154.70},

{78.47,134.95},{69.16,121.95},{60.20,102.58},{89.65,144.45},

{47.31,93.28},{87.81,155.29},{98.46,180.88},{ 8.62,66.81},

{18.21,62.82},{16.90,62.50},{66.64,125.01},{35.03,90.13},

{ 1.19,42.82},{39.25,87.90},{95.51,185.50},{52.60,130.29},

{42.18,98.66},{43.05,101.51},{66.40,133.64},{93.74,170.39},

{51.13,109.28},{72.79,138.29},{45.04,114.16},{15.80,70.59},

{94.90,151.00},{99.25,176.44},{65.26,137.40},{36.04,72.00},

{ 0.40,43.60},{86.91,138.43},{41.29,105.27},{56.72,116.01},

{20.01,65.99},{ 1.44,24.72},{60.94,134.19},{17.93,49.10},

{51.69,111.06},{98.77,171.60},{18.49,49.95},{98.53,168.61},

{39.04,91.25},{33.96,73.93},{58.72,122.23},{54.30,94.70},

{19.38,77.47},{59.93,100.50},{57.00,99.68},{19.08,49.37},

{62.19,125.66},{69.26,126.43},{63.67,124.79},{33.97,73.46},

{64.05,129.47},{90.41,160.25},{90.68,141.75},{58.43,120.99},

{47.72,98.06},{57.18,115.01},{21.72,71.98},{78.91,144.12},

{96.99,161.88},{90.98,144.82},{99.32,167.24},{92.02,168.45},

{76.83,146.88},{13.22,55.99},{22.96,85.34},{ 8.79,58.31},

{39.57,99.22},{95.41,166.55},{87.15,159.87},{77.74,155.73},

{30.21,80.03},{15.00,65.07},{ 1.25,36.22},{85.49,163.08},

{ 6.02,49.47},{38.29,90.72},{ 8.87,44.71},{21.80,68.59},

{69.88,115.45},{14.01,52.20},{15.63,62.15},{ 2.54,44.44},

{29.85,77.08},{83.24,154.98},{27.89,75.98},{87.91,160.12},

{66.17,134.31},{93.36,157.11},{64.99,130.27},{60.52,131.00},

{ 5.75,41.05},{87.46,144.19},{47.47,94.07},{ 8.26,43.29},

{42.36,100.58},{12.30,60.00},{16.23,60.55},{24.60,67.52},

{18.49,64.78},{80.04,139.33},{46.05,105.21},{16.89,58.52},

{ 6.63,62.86},{48.83,96.30},{39.23,100.05},{96.70,187.16},

{68.76,122.37},{55.28,120.83},{11.54,39.88},{ 6.99,41.18},

{96.16,173.94},{15.73,46.98},{91.07,145.49},{16.07,80.57},

{40.17,99.22},{64.93,118.55},{38.08,83.04},{ 6.19,54.52},

{86.09,150.91},{56.36,104.25},{41.95,97.61},{98.49,166.66},

{79.59,141.04},{32.42,80.76},{61.50,115.50},{40.85,88.12},

{39.26,97.38},{52.55,125.88},{72.67,152.23},{23.07,75.82},

{12.61,41.49},{89.50,132.51},{90.08,173.93},{54.88,116.36},

{41.88,88.54},{68.23,112.00},{13.83,37.32},{57.71,108.79},

{ 7.49,57.40},{80.46,126.17},{54.78,112.41},{85.85,156.33},

{39.31,84.84},{ 0.57,11.17},{92.41,145.81},{55.63,119.56},

{65.66,108.36},{72.85,136.35},{36.00,93.64},{45.38,97.11},

{99.11,144.04},{27.37,82.63},{82.92,153.95},{17.88,74.85},

{23.32,60.05},{68.17,134.30},{75.78,134.99},{80.26,149.31},

{11.84,49.92},{98.01,156.92},{ 1.82,38.12},{17.27,67.28},

{ 2.30,45.03},{87.16,152.81},{ 5.12,47.75},{47.91,103.87},

{45.76,107.49},{51.05,121.73},{43.92,104.41},{ 2.24,23.42},

{19.45,72.35},{ 4.27,46.17},{19.93,62.42},{49.57,99.46},

{88.25,160.42},{ 6.47,36.61},{20.82,60.18},{24.10,66.11},

{97.18,175.27},{13.49,43.49},{93.97,161.19},{97.02,143.79},

{94.51,162.20},{29.71,87.40},{72.56,128.83},{24.59,74.45},

{82.46,150.71},{69.11,131.86},{14.59,47.06},{93.10,154.28},

{11.42,58.70},{49.15,117.48},{65.69,130.75},{66.85,132.81},

{30.30,66.09},{33.49,82.24},{19.09,78.38},{76.50,146.34},

{ 2.48,22.05},{93.19,175.64},{52.16,122.32},{89.50,157.08},

{ 2.70,34.81},{ 3.19,31.68},{58.98,110.55},{63.36,110.02},

{90.50,154.14},{72.47,124.43},{26.44,75.60},{78.04,147.11},

{20.76,66.40},{26.99,78.87},{10.23,40.71},{34.27,77.02},

{41.62,111.74},{57.40,139.36},{16.31,76.09},{93.55,170.94},

{30.47,65.32},{78.87,153.63},{69.79,132.46},{62.10,129.93},

{55.54,107.90},{44.31,116.79},{33.50,80.09},{30.62,86.17},

{ 9.22,55.76},{57.81,111.98},{40.80,85.30},{ 1.13,51.94},

{37.18,73.32},{11.92,64.57},{ 5.96,40.39},{29.06,85.72},

{45.79,93.94},{33.97,80.99},{34.65,71.49},{44.26,77.08},

{47.49,104.46},{66.18,145.70},{27.32,76.18},{83.78,154.25},

{ 3.86,49.52},{ 2.95,46.28},{53.75,120.20},{26.85,71.84},

{26.51,73.85},{99.02,164.59},{66.62,120.46},{ 4.36,44.96},

{76.90,140.62},{21.49,69.85},{23.49,83.78},{99.09,165.36},

{60.04,132.84},{23.10,54.47},{40.68,86.11},{31.91,106.31},

{37.12,82.67},{39.39,86.60},{40.18,105.70},{51.35,129.06},

{58.58,112.39},{31.78,68.54},{65.77,125.66},{50.22,113.81},

{22.42,52.72},{73.70,148.23},{27.37,99.01},{41.00,94.43},

{56.35,110.68},{85.69,159.14},{52.28,111.81},{25.12,91.74},

{29.30,75.16},{32.79,95.38},{12.83,43.35},{83.30,136.94},

{41.34,77.86},{42.87,89.04},{13.24,62.89},{62.62,138.13},

{14.40,65.55},{27.11,81.45},{38.38,61.89},{17.27,72.58},

{29.53,85.39},{81.42,136.71},{12.21,48.86},{74.83,120.84},

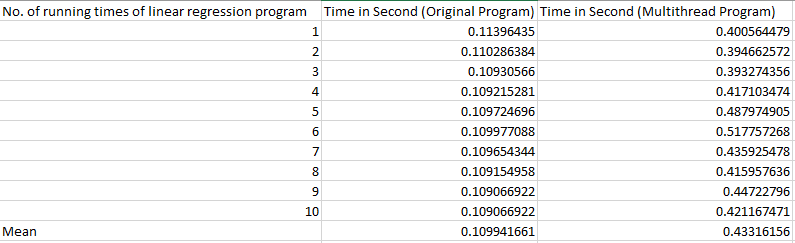
{74.97,138.85},{11.93,41.04},{22.62,64.23},{89.69,139.61},

{40.38,76.74},{65.74,124.89},{71.41,149.81},{50.76,99.00}

};

1. Calculate the mean running time of the multithread version of the program and use to make comments on the relative performance of the 2 versions.

Answer:



According to the above table, the mean running time to run the original program took 0.109941661 seconds whereas in the multithread version it took 0.43316156 seconds which means multithread program took more time to run the program then the original program because in multi-thread program each of the evaluation that is eight values for m and c performes on a different thread. So, it takes more time to run then the original program.

# CUDA

## Password Cracking

1. Create another version of the two-initials-four-digits password cracker. It should use separate CUDA thread to process each possible pair of the initials, i.e. thread 0 should process AA through to thread 675 processing ZZ. Each thread must therefore perform the 10000 iterations that explore the four-digit integers.

Answer: Code of Password Cracking of CUDA version.

#include <stdio.h>

#include <cuda\_runtime\_api.h>

#include <time.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

This program gives an example of a poor way to implement a password cracker

in CUDA C. It is poor because it acheives this with just one thread, which

is obviously not good given the scale of parallelism available to CUDA

programs.

The intentions of this program are:

1) Demonstrate the use of \_\_device\_\_ and \_\_global\_\_ functions

2) Enable a simulation of password cracking in the absence of library

with equivalent functionality to libcrypt. The password to be found

is hardcoded into a function called is\_a\_match.

Compile and run with:

nvcc -o 1\_a\_crack 1\_a\_crack.cu

./1\_a\_crack

Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

This function returns 1 if the attempt at cracking the password is

identical to the plain text password string stored in the program.

Otherwise,it returns 0.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

\_\_device\_\_ int is\_a\_match(char \*attempt) {

char plain\_password1[] = "AA1111";

char plain\_password2[] = "BB2222";

char plain\_password3[] = "SS3333";

char plain\_password4[] = "ZZ4444";

char \*a = attempt;

char \*b = attempt;

char \*c = attempt;

char \*d = attempt;

char \*p1 = plain\_password1;

char \*p2 = plain\_password2;

char \*p3 = plain\_password3;

char \*p4 = plain\_password4;

while(\*a == \*p1) {

if(\*a == '\0')

{

printf("Password: %s\n",plain\_password1);

break;

}

a++;

p1++;

}

while(\*b == \*p2) {

if(\*b == '\0')

{

printf("Password: %s\n",plain\_password2);

break;

}

b++;

p2++;

}

while(\*c == \*p3) {

if(\*c == '\0')

{

printf("Password: %s\n",plain\_password3);

break;

}

c++;

p3++;

}

while(\*d == \*p4) {

if(\*d == '\0')

{

printf("Password: %s\n",plain\_password4);

return 1;

}

d++;

p4++;

}

return 0;

}

\_\_global\_\_ void kernel() {

char a1,b2,c3,d4;//variables

char password[7];

password[6] = '\0';

//block id threrad id initilized

int i = blockIdx.x+65;

int j = threadIdx.x+65;

char firstMatch = i;

char secondMatch = j;

password[0] = firstMatch;

password[1] = secondMatch;

for(a1='0'; a1<='9'; a1++){

for(b2='0'; b2<='9'; b2++){

for(c3='0'; c3<='9'; c3++){

for(d4='0'; d4<='9'; d4++){

password[2] = a1;

password[3] = b2;

password[4] = c3;

password[5] = d4;

if(is\_a\_match(password)) {

}

else {

//printf("tried: %s\n", password);

}

}

}

}

}

}

//time difference

int time\_difference(struct timespec \*start,

struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

int main() {

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

//kernal function that determine block and threads to use

kernel <<<26,26>>>();

cudaThreadSynchronize();

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

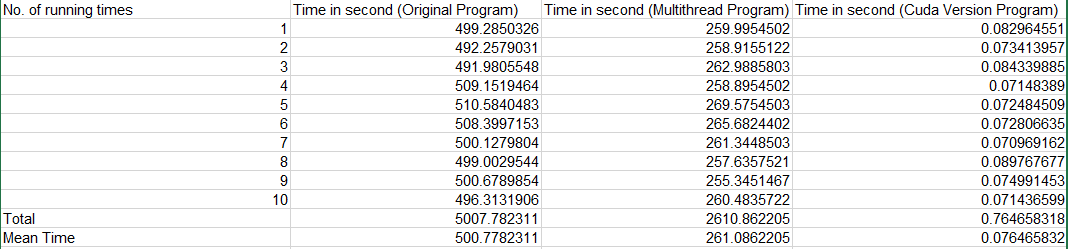
printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed/1.0e9));

return 0;

}

1. Compare the mean running time of the CUDA version with the original and multithread versions.

Answer:



In this program we have not give encrypted password because the nvidia graphics does not support the encrypted password.

As shown in the above table, the original program took 500.7782311 seconds to run the program similarly, the multithread program took 261.0862205 seconds which is half taken time than the original program and the CUDA version program took 0.076465832 seconds which is less time taken than the other 2 programs. This is because the CUDA version of the password cracking is faster than the other two program which is original version and multithread version program and also the CUDA replaces loop and the program runs parallel over 675 threads. So, because of this reason the running time for CUDA password cracking takes almost 80% lesser time than original.

## Image Processing

1. Create another version of the edge detection program that utilises CUDA. Each CUDA thread should process an individual pixel, therefore given an image that is 100 \* 72 this will require 7200 thread.

Answer: Code of Image Processing of CUDA version

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <GL/glut.h>

#include <GL/gl.h>

#include <malloc.h>

#include <signal.h>

#include <cuda\_runtime\_api.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Displays two grey scale images. On the left is an image that has come from an

image processing pipeline, just after colour thresholding. On the right is

the result of applying an edge detection convolution operator to the left

image. This program performs that convolution.

Things to note:

- A single unsigned char stores a pixel intensity value. 0 is black, 256 is

white.

- The colour mode used is GL\_LUMINANCE. This uses a single number to

represent a pixel's intensity. In this case we want 256 shades of grey,

which is best stored in eight bits, so GL\_UNSIGNED\_BYTE is specified as

the pixel data type.

To compile adapt the code below wo match your filenames:

nvcc -o 2\_a 2\_a.cu -lglut -lGL -lm

./2\_a

Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define width 100

#define height 72

unsigned char results[width \* height];

//dataset

unsigned char image[] = {255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,0,255,255,0,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,0,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,

0,0,0,0,0,0,0,255,255,255,0,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,0,0,0,0,0,0,0,255,255,255,0,0,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,0,255,0,0,0,0,0,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,

0,0,0,255,255,255,0,0,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,0,0,0,255,255,0,0,0,0,0,0,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,0,0,0,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,0,0,0,0,255,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,0,0,

0,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,255,

255,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,0,0,0,0,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,255,255,255,0,0,0,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,0,0,

0,0,0,0,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,255,255,255,0,0,0,0,0,255,255,255,255,0,0,0,255,255,255,

255,0,0,0,255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,

255,255,255,0,0,0,0,255,255,0,0,0,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,

255,255,0,0,0,0,255,255,255,255,255,0,0,0,255,255,255,255,0,

0,0,0,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,255,255,255,255,255,0,0,0,255,255,0,0,0,255,255,

255,255,255,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,255,

255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,

0,255,255,255,255,255,255,0,0,0,0,255,255,255,255,255,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,

0,0,0,0,255,255,0,0,0,0,255,255,255,255,0,0,0,0,255,

255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,255,255,255,0,0,0,0,255,255,255,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,255,255,255,255,0,0,0,0,0,255,

255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,255,255,255,255,

0,0,0,0,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,0,

0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,255,255,0,0,

0,0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,0,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,255,0,0,0,

255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,0,0,0,

0,255,255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,255,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,255,255,0,0,0,255,255,255,255,255,255,0,0,

255,255,255,0,0,0,255,255,255,255,0,0,0,0,0,255,255,255,255,

255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,255,0,0,0,

0,255,255,255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,255,0,

0,0,255,255,255,255,255,255,0,0,0,255,255,0,0,0,255,255,255,

255,255,0,0,0,255,255,255,0,0,0,255,255,255,255,0,0,0,0,

0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,0,255,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,0,0,

0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,255,

0,0,0,255,255,255,255,0,0,0,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,

0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,255,0,0,0,

0,0,0,0,0,255,255,0,0,0,255,255,255,255,255,0,0,0,255,

255,255,255,0,0,0,0,255,255,255,0,0,0,0,255,255,255,0,0,

0,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,0,

0,0,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,0,

0,0,0,255,255,0,0,0,0,255,255,0,0,0,0,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,255,255,255,255,0,0,255,255,255,255,255,0,0,0,0,0,255,255,

255,255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,0,0,0,

0,0,0,0,0,255,255,0,0,0,0,255,0,0,0,0,255,255,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,255,255,

255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,255,255,255,255,255,0,0,255,255,255,255,255,0,0,

0,0,0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,

255,255,0,0,0,0,0,0,0,0,255,255,0,0,0,0,255,255,0,

0,0,255,255,255,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,255,255,255,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,255,

255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,

255,0,0,0,0,0,255,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,

0,0,0,0,0,255,255,0,0,0,0,255,255,255,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,255,255,255,255,255,255,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255,255,

0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,

0,0,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,255,0,0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255,

255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,0,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,0,255,255,255,255,255,255,255,255,

255,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,

0,0,0,255,255,0,255,255,0,255,255,255,255,255,255,255,255,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0

};

\_\_global\_\_ void detect\_edges(unsigned char \*input, unsigned char \*output) {

int i = (blockIdx.x \* 72) + threadIdx.x;

int x, y; // the pixel of interest

int a, b, c, d; // the pixels adjacent to x,y used for the calculation

int r; // the result of calculate

y = i / width;;

x = i - (width \* y);

if (x == 0 || y == 0 || x == width - 1 || y == height - 1) {

output[i] = 0;

} else {

b = i + width;

d = i - 1;

f = i + 1;

h = i - width;

r = (input[i] \* 4) + (input[a] \* -1) + (input[b] \* -1) + (input[c] \* -1)

+ (input[d] \* -1);

if (r >= 0) {

output[i] = 255;

} else {

output[i] = 0;

}

}

}

void tidy\_and\_exit() {

exit(0);

}

void sigint\_callback(int signal\_number){

printf("\nInterrupt from keyboard\n");

tidy\_and\_exit();

}

static void display() {

glClear(GL\_COLOR\_BUFFER\_BIT);

glRasterPos4i(-1, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image);

glRasterPos4i(0, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results);

glFlush();

}

static void key\_pressed(unsigned char key, int x, int y) {

switch(key){

case 27: // escape

tidy\_and\_exit();

break;

default:

printf("\nPress escape to exit\n");

break;

}

}

int time\_difference(struct timespec \*start, struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

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

printf("image dimensions %dx%d\n", width, height);

unsigned char \*d\_results;

unsigned char \*d\_image;

cudaMalloc((void\*\*)&d\_image, sizeof(unsigned char) \* (width \* height));

cudaMalloc((void\*\*)&d\_results, sizeof(unsigned char) \* (width \* height));

cudaMemcpy(d\_image, &image, sizeof(unsigned char) \* (width \* height), cudaMemcpyHostToDevice);

signal(SIGINT, sigint\_callback);

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

detect\_edges<<<100,72>>>(d\_image, d\_results);

cudaThreadSynchronize();

cudaMemcpy(&results, d\_results, sizeof(unsigned char) \* (width \* height), cudaMemcpyDeviceToHost);

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed,

(time\_elapsed/1.0e9));

cudaFree(&d\_image);

cudaFree(&d\_results);

glutInit(&argc, argv);

glutInitWindowSize(width \* 2,height);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_LUMINANCE);

glutCreateWindow("Cuda Image Processing High Performance Computing");

glutDisplayFunc(display);

glutKeyboardFunc(key\_pressed);

glClearColor(0.0, 1.0, 0.0, 1.0);

glutMainLoop();

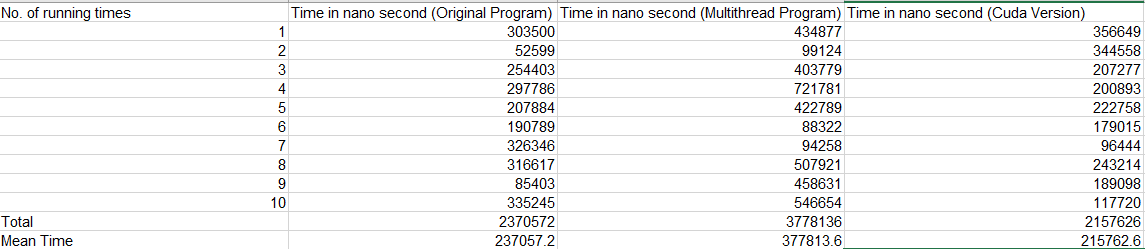
tidy\_and\_exit();

return 0;

}

1. Compare the mean running time of the CUDA version with the original program and multithreaded version. Comment of the effect of invoking more threads than cores.

Answer:



From the above figure we can see that, the original image processing program took 237057.2 nano seconds to run similarly, multithread program took 377813.6 nano seconds which is slightly more than the original program and the CUDA version took 215762.6 nano seconds in average. The time taken by the CUDA version is less than the other two program because the CUDA has more processing core than the other so the thread processes an individual pixel so there is 7200 pixels which allows to run each thread block to 7200 threads in parallel.

## Linear Regression

1. Create another version of the linear regression program that utilises CUDA. There should be 1000 CUDA threads that can evaluates a single data point in parallel. The main algorithm therefore needs to call the CUDA kernel 8 times for each minimisation iteration.

Answer: Code of linear regression of CUDA version

#include <stdio.h>

#include <math.h>

#include <time.h>

#include <unistd.h>

#include <cuda\_runtime\_api.h>

#include <unistd.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This program takes an initial estimate of m and c and finds the associated

\* rms error. It is then as a base to generate and evaluate 8 new estimates,

\* which are steps in different directions in m-c space. The best estimate is

\* then used as the base for another iteration of "generate and evaluate". This

\* continues until none of the new estimates are better than the base. This is

\* a gradient search for a minimum in mc-space.

\*

\* To compile:

\* nvcc -o 3\_a\_LinearRegression 3\_a\_LinearRegression.cu -lm

\*

\* To run:

\* ./3\_a\_LinearRegression

\*

\* Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

typedef struct point\_t {

double x;

double y;

} point\_t;

int n\_data = 1000;

\_\_device\_\_ int d\_n\_data = 1000;

point\_t data[] = {

{65.06,140.50},{68.29,128.16},{65.16,115.78},{77.08,144.62},

{80.58,148.61},{69.14,116.53},{65.57,114.91},{82.72,137.98},

{83.65,124.22},{79.38,144.31},{78.26,132.38},{87.08,145.27},

{94.45,147.08},{92.35,164.15},{29.71,80.37},{ 4.60,55.88},

{93.22,160.01},{41.28,104.89},{70.80,148.31},{65.29,130.26},

{99.15,171.42},{ 3.63,57.34},{24.39,60.47},{ 5.30,63.83},

{83.72,129.70},{ 7.17,54.62},{ 2.55,51.94},{81.91,149.27},

{65.84,125.98},{52.85,102.69},{53.55,111.53},{68.67,133.94},

{71.18,133.57},{31.95,72.01},{68.73,111.86},{72.26,120.95},

{69.11,100.24},{57.60,107.51},{93.09,163.24},{97.86,152.47},

{76.21,142.09},{57.24,122.45},{54.07,109.88},{83.78,140.43},

{24.99,76.38},{51.46,97.96},{73.46,126.86},{84.30,126.29},

{12.04,47.93},{ 6.50,54.26},{33.87,73.33},{79.90,149.55},

{25.17,81.80},{37.22,79.55},{38.65,89.41},{39.91,100.49},

{76.60,157.45},{18.16,60.40},{65.87,126.29},{71.88,112.63},

{70.80,116.68},{80.05,135.15},{71.60,130.84},{92.36,182.25},

{81.46,133.35},{47.37,82.88},{78.32,140.80},{35.59,84.76},

{98.55,159.78},{69.63,136.22},{ 6.85,31.94},{68.92,135.57},

{39.62,75.64},{45.22,95.17},{56.10,113.68},{33.40,90.58},

{51.25,113.20},{31.87,71.31},{91.54,174.46},{34.49,91.73},

{49.83,102.67},{16.86,48.98},{30.06,88.34},{98.04,196.30},

{31.25,86.63},{38.39,91.43},{36.81,90.52},{29.41,87.37},

{30.49,84.56},{23.21,74.83},{86.64,149.79},{98.24,187.36},

{44.74,80.44},{89.56,162.75},{88.07,155.61},{65.31,139.57},

{ 5.98,36.29},{85.06,153.94},{32.95,79.00},{43.81,101.89},

{62.22,111.47},{18.81,50.86},{65.82,113.31},{34.13,68.03},

{28.85,70.54},{71.38,141.34},{35.08,81.06},{41.36,109.69},

{40.12,76.78},{77.54,133.76},{50.18,106.00},{69.88,128.59},

{85.04,147.04},{66.82,128.00},{ 4.62,65.65},{78.07,145.26},

{33.81,86.56},{30.94,86.44},{78.63,158.12},{30.00,94.75},

{68.43,134.65},{66.31,122.03},{72.19,140.43},{43.52,112.56},

{58.36,129.89},{63.49,109.82},{72.93,140.32},{ 0.91,32.13},

{92.36,166.03},{58.59,132.02},{19.62,65.12},{73.70,127.87},

{ 7.10,49.29},{96.54,188.39},{11.72,29.18},{85.16,146.33},

{86.71,149.72},{82.62,137.78},{ 0.54,36.93},{43.02,101.71},

{56.52,96.81},{83.48,141.45},{87.95,150.02},{41.46,79.96},

{14.22,41.84},{ 6.09,45.81},{70.28,124.44},{70.57,126.73},

{21.65,60.07},{55.38,112.19},{ 1.10,46.57},{99.19,175.35},

{20.13,74.04},{36.30,79.86},{15.00,64.84},{55.51,111.39},

{39.08,81.62},{26.40,52.68},{68.19,136.41},{15.15,66.96},

{10.81,41.02},{45.70,113.27},{42.56,109.98},{80.27,131.75},

{45.41,95.82},{31.53,73.92},{40.84,93.37},{54.49,100.20},

{98.15,165.46},{ 8.63,51.82},{ 2.49,53.66},{73.88,147.16},

{13.36,63.33},{64.39,141.51},{84.32,157.95},{67.96,107.69},

{ 6.78,39.40},{77.95,141.91},{51.91,109.56},{47.62,92.81},

{27.17,63.62},{61.71,117.83},{38.62,86.65},{60.72,124.18},

{67.81,134.12},{31.53,71.56},{14.00,61.96},{19.50,59.01},

{90.95,135.89},{91.03,158.12},{33.65,82.03},{12.33,39.88},

{83.04,132.29},{50.85,112.28},{70.55,139.16},{64.07,121.96},

{35.31,92.95},{ 3.02,59.14},{46.36,105.52},{ 9.08,51.06},

{97.16,170.35},{ 5.22,48.76},{37.20,90.63},{60.93,127.67},

{89.32,178.38},{69.67,138.27},{19.36,63.13},{33.04,71.81},

{39.92,65.58},{50.92,124.15},{26.92,77.66},{29.14,82.75},

{ 0.85,40.60},{27.69,81.38},{81.94,142.14},{31.80,82.99},

{32.47,87.54},{ 4.43,39.37},{86.44,160.01},{62.87,132.53},

{48.20,103.22},{95.55,163.90},{82.81,152.15},{41.02,71.00},

{51.46,91.99},{42.61,92.40},{ 9.37,55.95},{12.68,62.08},

{41.45,102.49},{67.42,123.38},{83.35,157.06},{25.04,62.00},

{17.95,67.64},{74.23,142.58},{ 2.20,42.77},{85.41,152.29},

{63.83,111.25},{35.59,72.75},{ 9.84,59.26},{39.47,99.05},

{16.96,54.75},{27.80,92.37},{68.88,146.45},{80.19,160.65},

{41.22,99.46},{47.87,102.96},{30.06,77.15},{13.92,61.54},

{40.95,107.20},{48.85,98.44},{20.23,71.85},{37.48,87.76},

{46.93,100.97},{91.46,166.75},{66.78,127.65},{69.80,128.92},

{56.99,113.89},{64.99,118.91},{60.11,133.15},{11.00,71.54},

{ 8.09,61.76},{33.73,89.91},{14.97,47.57},{ 1.82,55.82},

{55.44,124.87},{18.43,54.83},{51.17,107.27},{14.12,58.21},

{61.49,123.07},{64.13,109.73},{42.54,92.26},{65.15,144.90},

{53.75,105.92},{57.63,114.92},{13.89,45.67},{52.03,101.50},

{63.30,116.07},{89.52,143.82},{63.83,113.06},{80.08,143.83},

{ 6.92,47.55},{57.40,103.92},{64.21,125.58},{46.81,106.30},

{91.30,151.08},{76.66,155.83},{47.48,112.92},{54.00,99.69},

{40.35,94.01},{77.68,150.21},{87.39,143.66},{76.78,146.43},

{54.80,105.64},{23.96,75.02},{45.90,86.75},{17.85,40.30},

{44.08,99.76},{62.84,119.24},{ 4.66,40.51},{61.01,125.84},

{96.70,168.86},{54.75,112.30},{ 8.70,53.95},{95.79,174.98},

{99.36,150.00},{69.48,127.58},{44.39,102.81},{43.75,98.47},

{45.91,95.15},{11.07,50.20},{85.58,141.73},{26.65,72.63},

{16.74,71.23},{43.53,83.72},{ 1.76,42.14},{23.78,79.11},

{26.59,84.39},{43.96,99.13},{47.07,120.85},{90.53,150.59},

{63.14,121.81},{17.75,73.96},{38.85,90.89},{77.91,145.78},

{64.09,115.44},{90.91,171.76},{53.97,123.30},{36.48,77.64},

{91.35,179.25},{70.46,131.28},{84.31,156.60},{53.08,103.03},

{26.80,62.81},{74.76,160.33},{21.42,60.42},{77.34,148.78},

{17.14,58.22},{69.61,127.58},{90.26,159.43},{69.55,132.52},

{82.69,132.50},{96.81,158.98},{99.48,182.04},{34.32,96.27},

{ 9.71,55.14},{46.94,102.50},{71.31,135.96},{66.98,121.76},

{98.22,165.52},{89.51,166.77},{ 6.33,30.69},{65.11,122.80},

{82.86,156.93},{16.40,66.05},{35.25,94.44},{94.61,147.92},

{96.45,166.53},{12.94,45.40},{36.88,76.80},{42.23,102.36},

{38.52,79.49},{28.72,69.83},{ 5.47,47.33},{31.42,76.78},

{89.61,149.18},{54.33,120.28},{29.57,82.48},{ 3.24,37.35},

{39.97,92.40},{38.31,88.57},{64.15,101.51},{66.60,124.22},

{ 8.39,53.51},{53.52,120.09},{54.00,104.21},{69.22,116.75},

{66.90,125.93},{16.18,60.34},{36.52,106.85},{ 3.83,57.76},

{63.60,111.98},{16.08,64.93},{14.51,60.59},{55.61,91.17},

{36.35,104.47},{22.41,63.89},{32.32,78.96},{44.95,91.99},

{12.10,66.00},{76.41,146.29},{57.89,105.09},{22.74,83.64},

{68.93,135.75},{56.80,101.13},{44.07,95.81},{ 2.22,39.62},

{34.64,80.94},{91.59,175.08},{ 7.54,56.50},{90.29,148.76},

{ 0.18,59.80},{59.07,108.26},{24.80,83.05},{58.62,123.01},

{16.02,56.86},{38.71,92.88},{99.87,174.65},{59.02,109.10},

{82.68,153.35},{24.15,99.81},{64.58,117.31},{21.24,69.38},

{30.70,80.02},{ 1.41,40.58},{59.40,111.46},{52.91,94.94},

{80.90,151.48},{ 9.73,51.73},{70.74,131.92},{92.53,157.93},

{93.11,160.17},{78.60,142.44},{39.51,103.00},{27.05,76.90},

{53.77,102.32},{82.71,163.25},{32.13,74.88},{73.27,134.77},

{18.37,66.43},{96.87,160.13},{14.74,59.16},{29.79,66.13},

{81.48,156.13},{50.93,132.81},{34.14,90.11},{97.06,177.46},

{43.59,118.51},{87.77,155.51},{26.78,56.57},{67.37,135.73},

{43.72,82.53},{78.90,151.03},{67.47,128.75},{73.54,142.62},

{83.58,146.88},{60.97,105.98},{61.39,121.73},{97.62,170.99},

{22.87,75.14},{14.33,60.35},{10.18,41.30},{71.87,138.38},

{ 6.67,47.75},{ 0.18,31.24},{63.19,127.26},{30.39,85.15},

{ 5.40,37.14},{30.71,94.33},{80.79,127.64},{60.59,107.25},

{13.56,65.29},{87.52,167.00},{14.84,53.76},{41.12,94.91},

{10.69,58.47},{53.25,89.18},{81.43,152.09},{33.06,79.09},

{15.51,51.17},{79.41,141.15},{70.48,139.28},{94.07,168.01},

{51.12,107.53},{ 5.67,45.68},{58.79,131.63},{19.47,73.96},

{31.67,71.86},{26.69,74.47},{71.45,147.52},{86.49,138.97},

{34.95,73.25},{73.30,148.44},{71.11,133.22},{94.75,166.25},

{88.11,147.28},{19.30,67.43},{90.05,164.25},{15.78,61.88},

{25.32,70.27},{28.72,74.55},{79.45,132.86},{25.88,72.08},

{31.14,86.32},{35.98,84.84},{ 5.71,44.50},{76.81,142.72},

{57.22,111.49},{87.46,139.57},{14.92,62.58},{83.44,141.89},

{23.73,58.30},{50.44,86.37},{20.16,70.79},{38.51,83.48},

{40.26,75.05},{15.57,53.06},{13.83,71.11},{19.29,73.30},

{91.73,146.04},{31.42,95.31},{24.64,72.44},{36.80,91.84},

{30.03,98.86},{11.40,51.19},{ 9.15,45.77},{65.77,120.82},

{90.64,148.02},{49.00,115.15},{33.13,82.17},{11.11,47.41},

{57.31,127.10},{22.37,75.81},{39.67,95.67},{29.11,75.90},

{45.84,86.16},{45.13,92.70},{ 5.48,50.66},{24.90,85.23},

{63.62,106.70},{82.49,142.53},{71.25,141.47},{36.64,87.11},

{ 1.96,51.52},{50.21,125.63},{61.50,110.30},{93.82,163.58},

{95.33,150.35},{ 9.32,63.49},{62.76,120.10},{ 3.44,63.32},

{41.49,97.46},{74.59,128.03},{77.61,137.92},{68.64,121.30},

{62.10,121.86},{34.90,93.88},{30.26,96.06},{85.65,165.07},

{54.94,128.21},{33.16,61.94},{ 4.82,50.78},{14.96,67.98},

{41.74,94.46},{86.33,148.59},{78.50,151.64},{78.74,124.22},

{83.50,154.44},{69.30,150.92},{86.91,133.87},{57.76,101.58},

{ 9.27,58.05},{52.10,108.71},{ 6.70,52.52},{15.06,75.65},

{40.28,100.24},{72.78,141.77},{56.15,109.93},{34.21,91.19},

{46.13,95.81},{90.38,157.10},{79.42,137.37},{23.85,75.78},

{16.94,61.69},{28.30,63.59},{45.94,116.74},{21.90,72.69},

{86.71,144.20},{50.15,101.35},{47.52,104.48},{76.50,133.92},

{16.59,70.95},{30.07,81.69},{95.78,162.28},{79.85,139.57},

{32.16,84.87},{59.81,100.58},{12.73,55.16},{68.40,143.18},

{88.39,142.70},{70.92,141.63},{16.91,65.12},{65.99,138.01},

{29.08,65.78},{94.66,143.54},{17.46,51.83},{40.45,100.97},

{80.61,154.88},{55.99,101.51},{50.64,94.60},{89.71,154.05},

{76.69,128.98},{71.69,135.85},{ 6.01,40.29},{96.97,153.79},

{51.33,104.76},{26.38,78.48},{74.84,124.85},{87.23,154.70},

{43.79,101.79},{36.34,86.15},{93.97,152.26},{91.93,168.48},

{99.95,153.69},{ 6.24,47.10},{33.57,94.58},{25.15,90.07},

{ 2.71,47.98},{55.65,96.69},{56.66,103.53},{15.84,75.38},

{13.10,49.82},{31.59,87.78},{10.07,52.90},{64.69,128.15},

{ 0.29,51.37},{54.63,117.56},{16.48,47.12},{68.05,111.33},

{65.05,117.98},{50.73,105.93},{38.21,92.02},{84.19,152.96},

{60.62,120.01},{71.76,130.65},{99.68,172.79},{16.20,50.34},

{16.99,79.00},{29.19,59.02},{69.67,116.57},{ 2.04,45.22},

{53.52,113.83},{38.61,86.66},{20.82,71.20},{45.51,91.72},

{83.51,133.29},{95.95,166.47},{99.30,187.23},{57.30,129.30},

{54.98,98.10},{60.47,112.79},{57.56,119.35},{18.50,62.58},

{75.27,140.67},{ 5.79,48.06},{94.17,157.17},{25.72,77.59},

{66.55,135.05},{53.82,122.03},{44.12,103.61},{86.49,159.15},

{20.06,64.54},{97.78,170.13},{61.95,119.09},{58.91,128.99},

{75.55,152.04},{28.76,63.60},{61.54,100.40},{74.51,125.14},

{ 6.62,30.47},{45.73,90.74},{79.17,140.18},{42.13,94.38},

{ 3.52,42.69},{20.27,53.88},{47.35,118.66},{64.33,122.29},

{14.34,63.88},{48.22,104.79},{79.32,140.01},{20.79,81.66},

{21.07,49.96},{48.95,125.70},{99.68,178.12},{50.76,107.99},

{59.08,111.82},{32.86,83.53},{37.71,104.74},{ 6.95,41.30},

{61.04,126.11},{23.82,75.61},{ 3.64,39.77},{11.31,73.23},

{65.13,128.77},{45.97,87.42},{51.46,105.02},{50.48,106.56},

{29.63,85.41},{73.86,144.97},{56.80,127.73},{83.76,154.70},

{78.47,134.95},{69.16,121.95},{60.20,102.58},{89.65,144.45},

{47.31,93.28},{87.81,155.29},{98.46,180.88},{ 8.62,66.81},

{18.21,62.82},{16.90,62.50},{66.64,125.01},{35.03,90.13},

{ 1.19,42.82},{39.25,87.90},{95.51,185.50},{52.60,130.29},

{42.18,98.66},{43.05,101.51},{66.40,133.64},{93.74,170.39},

{51.13,109.28},{72.79,138.29},{45.04,114.16},{15.80,70.59},

{94.90,151.00},{99.25,176.44},{65.26,137.40},{36.04,72.00},

{ 0.40,43.60},{86.91,138.43},{41.29,105.27},{56.72,116.01},

{20.01,65.99},{ 1.44,24.72},{60.94,134.19},{17.93,49.10},

{51.69,111.06},{98.77,171.60},{18.49,49.95},{98.53,168.61},

{39.04,91.25},{33.96,73.93},{58.72,122.23},{54.30,94.70},

{19.38,77.47},{59.93,100.50},{57.00,99.68},{19.08,49.37},

{62.19,125.66},{69.26,126.43},{63.67,124.79},{33.97,73.46},

{64.05,129.47},{90.41,160.25},{90.68,141.75},{58.43,120.99},

{47.72,98.06},{57.18,115.01},{21.72,71.98},{78.91,144.12},

{96.99,161.88},{90.98,144.82},{99.32,167.24},{92.02,168.45},

{76.83,146.88},{13.22,55.99},{22.96,85.34},{ 8.79,58.31},

{39.57,99.22},{95.41,166.55},{87.15,159.87},{77.74,155.73},

{30.21,80.03},{15.00,65.07},{ 1.25,36.22},{85.49,163.08},

{ 6.02,49.47},{38.29,90.72},{ 8.87,44.71},{21.80,68.59},

{69.88,115.45},{14.01,52.20},{15.63,62.15},{ 2.54,44.44},

{29.85,77.08},{83.24,154.98},{27.89,75.98},{87.91,160.12},

{66.17,134.31},{93.36,157.11},{64.99,130.27},{60.52,131.00},

{ 5.75,41.05},{87.46,144.19},{47.47,94.07},{ 8.26,43.29},

{42.36,100.58},{12.30,60.00},{16.23,60.55},{24.60,67.52},

{18.49,64.78},{80.04,139.33},{46.05,105.21},{16.89,58.52},

{ 6.63,62.86},{48.83,96.30},{39.23,100.05},{96.70,187.16},

{68.76,122.37},{55.28,120.83},{11.54,39.88},{ 6.99,41.18},

{96.16,173.94},{15.73,46.98},{91.07,145.49},{16.07,80.57},

{40.17,99.22},{64.93,118.55},{38.08,83.04},{ 6.19,54.52},

{86.09,150.91},{56.36,104.25},{41.95,97.61},{98.49,166.66},

{79.59,141.04},{32.42,80.76},{61.50,115.50},{40.85,88.12},

{39.26,97.38},{52.55,125.88},{72.67,152.23},{23.07,75.82},

{12.61,41.49},{89.50,132.51},{90.08,173.93},{54.88,116.36},

{41.88,88.54},{68.23,112.00},{13.83,37.32},{57.71,108.79},

{ 7.49,57.40},{80.46,126.17},{54.78,112.41},{85.85,156.33},

{39.31,84.84},{ 0.57,11.17},{92.41,145.81},{55.63,119.56},

{65.66,108.36},{72.85,136.35},{36.00,93.64},{45.38,97.11},

{99.11,144.04},{27.37,82.63},{82.92,153.95},{17.88,74.85},

{23.32,60.05},{68.17,134.30},{75.78,134.99},{80.26,149.31},

{11.84,49.92},{98.01,156.92},{ 1.82,38.12},{17.27,67.28},

{ 2.30,45.03},{87.16,152.81},{ 5.12,47.75},{47.91,103.87},

{45.76,107.49},{51.05,121.73},{43.92,104.41},{ 2.24,23.42},

{19.45,72.35},{ 4.27,46.17},{19.93,62.42},{49.57,99.46},

{88.25,160.42},{ 6.47,36.61},{20.82,60.18},{24.10,66.11},

{97.18,175.27},{13.49,43.49},{93.97,161.19},{97.02,143.79},

{94.51,162.20},{29.71,87.40},{72.56,128.83},{24.59,74.45},

{82.46,150.71},{69.11,131.86},{14.59,47.06},{93.10,154.28},

{11.42,58.70},{49.15,117.48},{65.69,130.75},{66.85,132.81},

{30.30,66.09},{33.49,82.24},{19.09,78.38},{76.50,146.34},

{ 2.48,22.05},{93.19,175.64},{52.16,122.32},{89.50,157.08},

{ 2.70,34.81},{ 3.19,31.68},{58.98,110.55},{63.36,110.02},

{90.50,154.14},{72.47,124.43},{26.44,75.60},{78.04,147.11},

{20.76,66.40},{26.99,78.87},{10.23,40.71},{34.27,77.02},

{41.62,111.74},{57.40,139.36},{16.31,76.09},{93.55,170.94},

{30.47,65.32},{78.87,153.63},{69.79,132.46},{62.10,129.93},

{55.54,107.90},{44.31,116.79},{33.50,80.09},{30.62,86.17},

{ 9.22,55.76},{57.81,111.98},{40.80,85.30},{ 1.13,51.94},

{37.18,73.32},{11.92,64.57},{ 5.96,40.39},{29.06,85.72},

{45.79,93.94},{33.97,80.99},{34.65,71.49},{44.26,77.08},

{47.49,104.46},{66.18,145.70},{27.32,76.18},{83.78,154.25},

{ 3.86,49.52},{ 2.95,46.28},{53.75,120.20},{26.85,71.84},

{26.51,73.85},{99.02,164.59},{66.62,120.46},{ 4.36,44.96},

{76.90,140.62},{21.49,69.85},{23.49,83.78},{99.09,165.36},

{60.04,132.84},{23.10,54.47},{40.68,86.11},{31.91,106.31},

{37.12,82.67},{39.39,86.60},{40.18,105.70},{51.35,129.06},

{58.58,112.39},{31.78,68.54},{65.77,125.66},{50.22,113.81},

{22.42,52.72},{73.70,148.23},{27.37,99.01},{41.00,94.43},

{56.35,110.68},{85.69,159.14},{52.28,111.81},{25.12,91.74},

{29.30,75.16},{32.79,95.38},{12.83,43.35},{83.30,136.94},

{41.34,77.86},{42.87,89.04},{13.24,62.89},{62.62,138.13},

{14.40,65.55},{27.11,81.45},{38.38,61.89},{17.27,72.58},

{29.53,85.39},{81.42,136.71},{12.21,48.86},{74.83,120.84},

{74.97,138.85},{11.93,41.04},{22.62,64.23},{89.69,139.61},

{40.38,76.74},{65.74,124.89},{71.41,149.81},{50.76,99.00}

};

int time\_difference(struct timespec \*start, struct timespec \*finish, long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

double residual\_error(double x, double y, double m, double c) {

double e = (m \* x) + c - y;

return e \* e;

}

double rms\_error(double m, double c) {

int i;

double mean;

double error\_sum = 0;

for(i=0; i<n\_data; i++) {

error\_sum += residual\_error(data[i].x, data[i].y, m, c);

}

mean = error\_sum / n\_data;

return sqrt(mean);

}

\_\_device\_\_ double d\_residual\_error(double x, double y, double m, double c) {

double e = (m \* x) + c - y;

return e \* e;

}

\_\_global\_\_ void d\_rms\_error(double \*m, double \*c, double \*error\_sum\_arr, point\_t \*d\_data) {

int i = threadIdx.x + blockIdx.x \* blockDim.x;

error\_sum\_arr[i] = d\_residual\_error(d\_data[i].x, d\_data[i].y, \*m, \*c);

}

int main() {

int i;

double bm = 1.3;

double bc = 10;

double be;

double dm[8];

double dc[8];

double e[8];

double step = 0.01;

double best\_error = 999999999;

int best\_error\_i;

int minimum\_found = 0;

double om[] = {0,1,1, 1, 0,-1,-1,-1};

double oc[] = {1,1,0,-1,-1,-1, 0, 1};

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

cudaError\_t error;

double \*d\_dm;

double \*d\_dc;

double \*d\_error\_sum\_arr;

point\_t \*d\_data;

be = rms\_error(bm, bc);

//Allocate memory for d\_dm

error = cudaMalloc(&d\_dm, (sizeof(double) \* 8));

if(error){

fprintf(stderr, "cudaMalloc on d\_dm returned %d %s\n", error,

cudaGetErrorString(error));

exit(1);

}

//Allocate memory for d\_dc

error = cudaMalloc(&d\_dc, (sizeof(double) \* 8));

if(error){

fprintf(stderr, "cudaMalloc on d\_dc returned %d %s\n", error,

cudaGetErrorString(error));

exit(1);

}

//Allocate memory for d\_error\_sum\_arr

error = cudaMalloc(&d\_error\_sum\_arr, (sizeof(double) \* 1000));

if(error){

fprintf(stderr, "cudaMalloc on d\_error\_sum\_arr returned %d %s\n", error,

cudaGetErrorString(error));

exit(1);

}

//Allocate memory for d\_data

error = cudaMalloc(&d\_data, sizeof(data));

if(error){

fprintf(stderr, "cudaMalloc on d\_data returned %d %s\n", error,

cudaGetErrorString(error));

exit(1);

}

while(!minimum\_found) {

for(i=0;i<8;i++) {

dm[i] = bm + (om[i] \* step);

dc[i] = bc + (oc[i] \* step);

}

//Copy memory for dm to d\_dm

error = cudaMemcpy(d\_dm, dm, (sizeof(double) \* 8), cudaMemcpyHostToDevice);

if(error){

fprintf(stderr, "cudaMemcpy to d\_dm returned %d %s\n", error,

cudaGetErrorString(error));

}

//Copy memory for dc to d\_dc

error = cudaMemcpy(d\_dc, dc, (sizeof(double) \* 8), cudaMemcpyHostToDevice);

if(error){

fprintf(stderr, "cudaMemcpy to d\_dc returned %d %s\n", error,

cudaGetErrorString(error));

}

//Copy memory for data to d\_data

error = cudaMemcpy(d\_data, data, sizeof(data), cudaMemcpyHostToDevice);

if(error){

fprintf(stderr, "cudaMemcpy to d\_data returned %d %s\n", error,

cudaGetErrorString(error));

}

for(i=0;i<8;i++) {

//Host variable storing the array returned from the kernel function.

double h\_error\_sum\_arr[1000];

//Stores the total sum of the values from the error sum array.

double error\_sum\_total;

//Stores the mean of the total sum of the error sums.

double error\_sum\_mean;

//Call the rms\_error function using 100 blocks and 10 threads.

d\_rms\_error <<<100,10>>>(&d\_dm[i], &d\_dc[i], d\_error\_sum\_arr, d\_data);

cudaThreadSynchronize();

//Copy memory for d\_error\_sum\_arr

error = cudaMemcpy(&h\_error\_sum\_arr, d\_error\_sum\_arr, (sizeof(double) \* 1000), cudaMemcpyDeviceToHost);

if(error){

fprintf(stderr, "cudaMemcpy to error\_sum returned %d %s\n", error,

cudaGetErrorString(error));

}

//Loop through the error sum array returned from the kernel function

for(int j=0; j<n\_data; j++) {

//Add each error sum to the error sum total.

error\_sum\_total += h\_error\_sum\_arr[j];

}

//Calculate the mean for the error sum.

error\_sum\_mean = error\_sum\_total / n\_data;

//Calculate the square root for the error sum mean.

e[i] = sqrt(error\_sum\_mean);

if(e[i] < best\_error) {

best\_error = e[i];

best\_error\_i = i;

}

//Reset the error sum total.

error\_sum\_total = 0;

}

//printf("best m,c is %lf,%lf with error %lf in direction %d\n",

//dm[best\_error\_i], dc[best\_error\_i], best\_error, best\_error\_i);

if(best\_error < be) {

be = best\_error;

bm = dm[best\_error\_i];

bc = dc[best\_error\_i];

} else {

minimum\_found = 1;

}

}

//Free memory for d\_dm

error = cudaFree(d\_dm);

if(error){

fprintf(stderr, "cudaFree on d\_dm returned %d %s\n", error,cudaGetErrorString(error));

exit(1);

}

//Free memory for d\_dc

error = cudaFree(d\_dc);

if(error){

fprintf(stderr, "cudaFree on d\_dc returned %d %s\n", error,cudaGetErrorString(error));

exit(1);

}

//Free memory for d\_data

error = cudaFree(d\_data);

if(error){

fprintf(stderr, "cudaFree on d\_data returned %d %s\n", error,cudaGetErrorString(error));

exit(1);

}

//Free memory for d\_error\_sum\_arr

error = cudaFree(d\_error\_sum\_arr);

if(error){

fprintf(stderr, "cudaFree on d\_error\_sum\_arr returned %d %s\n", error,cudaGetErrorString(error));

exit(1);

}

printf("minimum m,c is %lf,%lf with error %lf\n", bm, bc, be);

//Get the system time after we have run the linear regression function.

clock\_gettime(CLOCK\_MONOTONIC, &finish);

//Calculate the time spent between the start time and end time.

time\_difference(&start, &finish, &time\_elapsed);

//Output the time spent running the program.

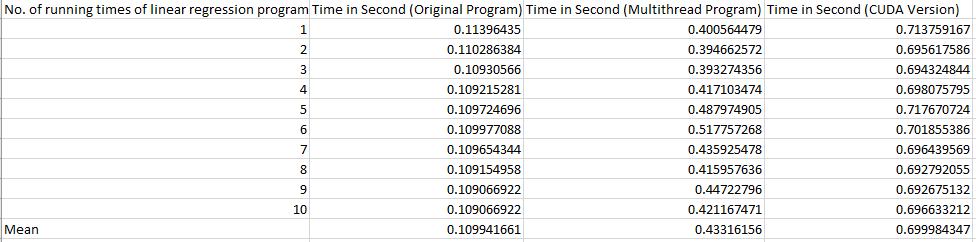
printf("Time elapsed was %lldns or %0.9lfs or %0.9lfmin\n", time\_elapsed,(time\_elapsed/1.0e9),(time\_elapsed/1.0e9/60));

return 0;

}

1. Compare the mean running time of the CUDA version with the original program and multithreaded versions.

Answer:



From the table we found out that the running time of the original program was 0.109941661 seconds similarly, the running time of multithread program was 0.43316156 seconds whereas, the CUDA version program running time is 0.699984347 seconds. As per my information CUDA version program run time should be less than the original and multithread program but here in the above table it is completely different because in CUDA version there are 1000 CUDA threads evaluating a single data point in parallel and the main CUDA kernel is called 8 times each for minimization iteration.

# MPI

## Password Cracking

1. Create another version of the two-initials-four-digits password cracker. A master should share the work out to two compute instances i.e. instance 1 should explore AA through to MZ, whilist instance 2 should explore NA through to NZ.

Answer: Code for Password Cracking using MPI

#include <stdlib.h>

#include <stdio.h>

#include <ctype.h>

#include <errno.h>

#include <sys/stat.h>

#include <string.h>

#include <time.h>

#include <pthread.h>

#include <math.h>

#include <crypt.h>

#include <mpi.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Demonstrates how to crack an encrypted password using a simple

"brute force" algorithm. Works on passwords that consist only of 2 uppercase

letters and a 2 digit integer. Your personalised data set is included in the

code.

Compile with:

mpicc -o 1\_a\_MPIPasswordCracking 1\_a\_MPIPasswordCracking.c -lcrypt

If you want to analyse the results then use the redirection operator to send

output to a file that you can view using an editor or the less utility:

mpirun -n 3 ./1\_a\_MPIPasswordCracking

Dr Kevan Buckley, University of Wolverhampton, 2019

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int n\_passwords = 4;

// encrypted passwords are AA1111, BB2222, SS3333, ZZ4444

char \*encrypted\_passwords[] = {

"$6$KB$YJ2cyUdgpCT/xIjdOjTkhYBZ9ZMGhECA6s0Oa2qcqet4QG99J1KOp1kU8KM96NAv.MUefq7znOGoBPT/R/Qof1",

"$6$KB$joc9OPGjxifQU65CL8CBjrzktkRaTaCh6V4EGA3GDVQNU1o6SXjZkPEzFv7sY6AKVKygbtHTPQ8sW.gvlkd0x.",

"$6$KB$EU/6W9xieuBxRicwU8jww278fq5AB4pO3PoA1jAZ4.a.zuLiV/q.5N2FvdJunOcAjjMKi/ygDMe8kRkCbdr01.",

"$6$KB$ShfuEhmrWupfgZMOqNv3Zj4Lqu8H064/byysYxm/hDthUkQEHNimhxLx1BDwc8c2yGHg7kF9RncwvObSWCfd2."

};

/\*\*

Required by lack of standard function in C.

\*/

void crack(char \*salt\_and\_encrypted, char start, char finish);

void substr(char \*dest, char \*src, int start, int length){

memcpy(dest, src + start, length);

\*(dest + length) = '\0';

}

/\*\*

This function can crack the kind of password explained above. All

combinations

that are tried are displayed and when the password is found, #, is put

at the

start of the line. Note that one of the most time consuming operations

that

it performs is the output of intermediate results, so performance

experiments

for this kind of program should not include this. i.e. comment out the

printfs.

\*/

void crack(char \*salt\_and\_encrypted, char start, char finish){

int x, y, z, w; // Loop counters

char salt[7]; // String used in hashing the password. Need space for \0

char plain[7]; // The combination of letters currently being checked

char \*enc; // Pointer to the encrypted password

int count = 0; // The number of combinations explored so far

substr(salt, salt\_and\_encrypted, 0, 6);

for(x= start; x<= finish; x++){

for(y='A'; y<='Z'; y++){

for(z=0; z<=99; z++){

for(w=0; w<=99; w++){

sprintf(plain, "%c%c%02d%02d",x, y, z, w);

enc = (char \*) crypt(plain, salt);

count++;

if(strcmp(salt\_and\_encrypted, enc) == 0){

printf("#%-8d%s %s\n", count, plain, enc);

} else {

//printf(" %-8d%s %s\n", count, plain, enc);

}}

}

}

}

printf("%d solutions explored\n", count);

}

int time\_difference(struct timespec \*start,

struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

// Calculate the difference between two times. Returns zero on

// success and the time difference through an argument. It will

// be unsuccessful if the start time is after the end time.

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

{

int size, rank;

struct timespec start, finish;

long long int time\_elapsed;

MPI\_Init(NULL, NULL);

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

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

if(size != 3)

{

if(rank == 0)

{

printf("This program needs exactly 3 process to run\n");

exit(-1);

}

}

else

{

for(int i=0; i<n\_passwords; i++)

{

if(rank==0){

clock\_gettime(CLOCK\_MONOTONIC, &start);

MPI\_Send(&encrypted\_passwords[i][0],512,MPI\_BYTE,1,0,MPI\_COMM\_WORLD);

MPI\_Send(&encrypted\_passwords[i][0],512,MPI\_BYTE,2,0,MPI\_COMM\_WORLD);

}

else

{

if(rank==1){

char password[512];

MPI\_Status status;

MPI\_Recv(&password[0], 512,MPI\_BYTE,0,0,MPI\_COMM\_WORLD,&status);

crack(password, 'A', 'M');

}

if(rank==2){

char password[512];

MPI\_Status status;

MPI\_Recv(&password[0], 512,MPI\_BYTE,0,0,MPI\_COMM\_WORLD,&status);

crack(password, 'N', 'Z');

}

}

}

}

MPI\_Barrier(MPI\_COMM\_WORLD);

if(rank==0)

{

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was %lldns or %0.9lfs or %0.9lfmin\n",time\_elapsed,(time\_elapsed/1.0e9),((time\_elapsed/1.0e9)/60));

}

MPI\_Finalize();

return 0;

}

1. Compare the mean running time of MPI version with the original, multithread and CUDA versions.

Answer:

The original program took 8.7804793519 minutes to run whereas the MPI version took 8.7711888339 minutes to run which is almost same. The MPI program in the first two run took around six minutes which increased to 8 minutes and the time was stable. This might have occurred due the problem in the computer.

## Image Processing

1. Create another version of the edge detection program that uses 4 MPI instances to each process a quarter of the image in a horizontal band. A master instance should share out the work and collate the results back into one image.

Answer: Code of Image Processing using MPI version

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <GL/glut.h>

#include <GL/gl.h>

#include <malloc.h>

#include <signal.h>

#include <mpi.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Displays two grey scale images. On the left is an image that has come from an

image processing pipeline, just after colour thresholding. On the right is

the result of applying an edge detection convolution operator to the left

image. This program performs that convolution.

This is a version of the edge detection program that uses 4 MPI instances to

each process a quarter of the image in a horizontal band. A master instance shares

out the work and collates the results back into one image.

Things to note:

- A single unsigned char stores a pixel intensity value. 0 is black, 256 is

white.

- The colour mode used is GL\_LUMINANCE. This uses a single number to

represent a pixel's intensity. In this case we want 256 shades of grey,

which is best stored in eight bits, so GL\_UNSIGNED\_BYTE is specified as

the pixel data type

To compile adapt the code below wo match your filenames:

mpicc -o 2a\_MPIProgram 2a\_MPIProgram.c -lglut -lGL -lm

mpirun -n 5 -quiet ././2a\_MPIProgram

Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define width 100

#define height 72

unsigned char image[], results[width \* height];

int startIndex, endIndex;

void detect\_edges(unsigned char \*in, unsigned char \*out) {

int i;

int n\_pixels = (width \* height);

for(i=0;i<n\_pixels;i++) {

int x, y; // the pixel of interest

int a, b, c, d; // the pixels adjacent to x,y used for the calculation

int r; // the result of calculate

y = i / width;

x = i - (width \* y);

if (x == 0 || y == 0 || x == width - 1 || y == height - 1) {

results[i] = 0;

} else {

b = i + width;

d = i - 1;

f = i + 1;

h = i - width;

r = (in[i] \* 4) + (in[a] \* -1) + (in[b] \* -1) + (in[c] \* -1)

+ (in[d] \* -1);

if (r > 0) { // if the result is positive this is an edge pixel

out[i] = 255;

} else {

out[i] = 0;

}

}

}

}

void tidy\_and\_exit() {

exit(0);

}

void sigint\_callback(int signal\_number){

printf("\nInterrupt from keyboard\n");

tidy\_and\_exit();

}

static void display() {

glClear(GL\_COLOR\_BUFFER\_BIT);

glRasterPos4i(-1, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, image);

glRasterPos4i(0, -1, 0, 1);

glDrawPixels(width, height, GL\_LUMINANCE, GL\_UNSIGNED\_BYTE, results);

glFlush();

}

static void key\_pressed(unsigned char key, int x, int y) {

switch(key){

case 27: // escape

tidy\_and\_exit();

break;

default:

printf("\nPress escape to exit\n");

break;

}

}

// Calculate the difference between two times. Returns zero on

// success and the time difference through an argument. It will

// be unsuccessful if the start time is after the end time.

int time\_difference(struct timespec \*start, struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

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

signal(SIGINT, sigint\_callback);

printf("image dimensions %dx%d\n", width, height);

int account = 0;

int size, rank;

MPI\_Init(NULL, NULL);

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

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

if(size != 5) {

if(rank == 0) {

printf("This program needs 5 processes\n");

}

} else {

if(rank ==0){

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start);

MPI\_Send(&results[0], 1800, MPI\_UNSIGNED\_CHAR, 1, 0, MPI\_COMM\_WORLD);

MPI\_Send(&results[1800], 1800, MPI\_UNSIGNED\_CHAR, 2, 0, MPI\_COMM\_WORLD);

MPI\_Send(&results[3600], 1800, MPI\_UNSIGNED\_CHAR, 3, 0, MPI\_COMM\_WORLD);

MPI\_Send(&results[5400], 1800, MPI\_UNSIGNED\_CHAR, 4, 0, MPI\_COMM\_WORLD);

MPI\_Recv(&results[0], 1800, MPI\_UNSIGNED\_CHAR, 1, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv(&results[1800], 1800,MPI\_UNSIGNED\_CHAR, 2, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv(&results[3600], 1800,MPI\_UNSIGNED\_CHAR, 3, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv(&results[5400], 1800,MPI\_UNSIGNED\_CHAR, 4, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

clock\_gettime(CLOCK\_MONOTONIC, &finish);

time\_difference(&start, &finish, &time\_elapsed);

printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed, (time\_elapsed/1.0e9));

glutInit(&argc, argv);

glutInitWindowSize(width \* 2,height);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_LUMINANCE);

glutCreateWindow("6CS005 Image Progessing Courework");

glutDisplayFunc(display);

glutKeyboardFunc(key\_pressed);

glClearColor(0.0, 1.0, 0.0, 1.0);

glutMainLoop();

tidy\_and\_exit();

} else {

if(rank == 1){

startIndex = 0;

endIndex = 1799;

MPI\_Recv(&results[0], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

detect\_edges(image, results);

MPI\_Send(&results[0], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD);

}

else if(rank == 2){

startIndex = 1800;

endIndex = 3599;

MPI\_Recv(&results[1800], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

detect\_edges(image, results);

MPI\_Send(&results[1800], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD);

}

else if(rank == 3){

startIndex = 3600;

endIndex = 5399;

MPI\_Recv(&results[3600], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

detect\_edges(image, results);

MPI\_Send(&results[3600], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD);

}

else if(rank == 4){

startIndex = 5400;

endIndex = 7199;

MPI\_Recv(&results[5400], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

detect\_edges(image, results);

MPI\_Send(&results[5400], 1800, MPI\_UNSIGNED\_CHAR, 0, 0, MPI\_COMM\_WORLD);

}

}

}

MPI\_Finalize();

return 0;

}

unsigned char image[] = {255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,255,0,255,255,0,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,0,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,

0,0,0,0,0,0,0,255,255,255,0,255,255,255,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,0,0,0,0,0,0,0,255,255,255,0,0,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,0,255,0,0,0,0,0,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,

0,0,0,255,255,255,0,0,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,255,255,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,255,255,0,0,0,0,0,0,0,0,

0,0,0,255,0,0,0,255,255,0,0,0,0,0,0,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,0,0,0,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,255,0,0,0,0,255,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,0,0,

0,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,255,

255,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,0,0,0,0,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,255,255,255,0,0,0,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,0,0,

0,0,0,0,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,255,255,255,0,0,0,0,0,255,255,255,255,0,0,0,255,255,255,

255,0,0,0,255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,

255,255,255,0,0,0,0,255,255,0,0,0,255,255,255,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,

255,255,0,0,0,0,255,255,255,255,255,0,0,0,255,255,255,255,0,

0,0,0,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,255,255,255,255,255,0,0,0,255,255,0,0,0,255,255,

255,255,255,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,255,

255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,

0,255,255,255,255,255,255,0,0,0,0,255,255,255,255,255,0,0,0,

255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,

0,0,0,0,255,255,0,0,0,0,255,255,255,255,0,0,0,0,255,

255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,255,255,255,0,0,0,0,255,255,255,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,255,255,255,255,0,0,0,0,0,255,

255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,255,255,255,255,

0,0,0,0,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,

0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,0,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,0,

0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,255,255,0,0,

0,0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,

255,0,0,0,0,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,255,0,0,0,

255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,0,0,0,

0,255,255,0,0,0,0,255,255,255,255,0,0,0,255,255,255,255,255,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,255,255,255,255,0,0,0,0,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,255,255,0,0,0,255,255,255,255,255,255,0,0,

255,255,255,0,0,0,255,255,255,255,0,0,0,0,0,255,255,255,255,

255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,255,0,0,0,

0,255,255,255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,255,0,

0,0,255,255,255,255,255,255,0,0,0,255,255,0,0,0,255,255,255,

255,255,0,0,0,255,255,255,0,0,0,255,255,255,255,0,0,0,0,

0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,0,255,

255,255,0,0,0,0,255,255,255,255,0,0,0,0,0,0,255,255,255,

255,255,255,255,255,255,255,255,255,255,0,0,0,255,255,255,255,0,0,

0,255,255,255,255,0,0,0,255,255,255,255,255,0,0,0,255,255,255,

0,0,0,255,255,255,255,0,0,0,255,255,255,0,0,0,255,255,255,

255,255,0,0,0,0,0,255,255,255,255,255,0,0,0,0,0,255,255,

0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,255,0,0,0,

0,0,0,0,0,255,255,0,0,0,255,255,255,255,255,0,0,0,255,

255,255,255,0,0,0,0,255,255,255,0,0,0,0,255,255,255,0,0,

0,255,255,255,255,0,0,0,0,0,0,0,0,0,0,255,255,255,0,

0,0,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,0,0,

0,0,0,255,255,0,0,0,0,255,255,0,0,0,0,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,255,255,0,0,0,0,0,0,0,

0,0,0,0,255,255,255,255,0,0,0,0,255,255,255,255,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,255,255,255,255,0,0,255,255,255,255,255,0,0,0,0,0,255,255,

255,255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,0,0,0,

0,0,0,0,0,255,255,0,0,0,0,255,0,0,0,0,255,255,0,

0,0,0,0,0,0,0,0,0,255,255,255,255,0,0,0,0,255,255,

255,255,255,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,0,

0,0,0,0,0,255,255,255,255,255,0,0,255,255,255,255,255,0,0,

0,0,0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,0,0,

255,255,0,0,0,0,0,0,0,0,255,255,0,0,0,0,255,255,0,

0,0,255,255,255,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

0,0,0,255,255,255,255,255,255,255,255,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,255,255,

255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,0,0,

255,0,0,0,0,0,255,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,255,255,255,255,255,255,0,0,0,0,0,255,255,255,255,255,

0,0,0,0,0,255,255,0,0,0,0,255,255,255,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,255,255,255,255,255,255,0,0,0,0,0,

255,255,255,255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,

0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255,255,

0,0,0,0,0,255,255,255,255,255,255,0,0,0,0,255,255,0,0,

0,0,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,255,0,0,0,0,0,255,255,255,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,255,255,

255,255,0,0,0,0,0,255,255,0,0,0,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,0,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,

0,0,255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,255,0,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,

0,0,0,0,0,0,0,0,0,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,255,255,255,255,0,255,255,255,255,255,255,255,255,

255,0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,255,255,

255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,255,255,255,0,0,0,0,255,255,255,255,255,255,0,0,0,

0,0,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,255,255,255,255,255,0,0,0,0,255,255,255,255,

255,255,0,0,0,0,255,255,255,255,255,255,255,255,255,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,

255,255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,255,255,255,255,255,255,255,255,255,

255,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,255,255,255,255,

255,255,255,255,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,255,255,255,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,0,

0,0,0,255,255,0,255,255,0,255,255,255,255,255,255,255,255,0,0,

0,255,255,255,255,255,255,255,255,255,255,255,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,255,255,255,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,255,255,0,0,0,0,0,0,255,255,255,255,255,255,255,255,255,255,

255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,255,0,0,0,

0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

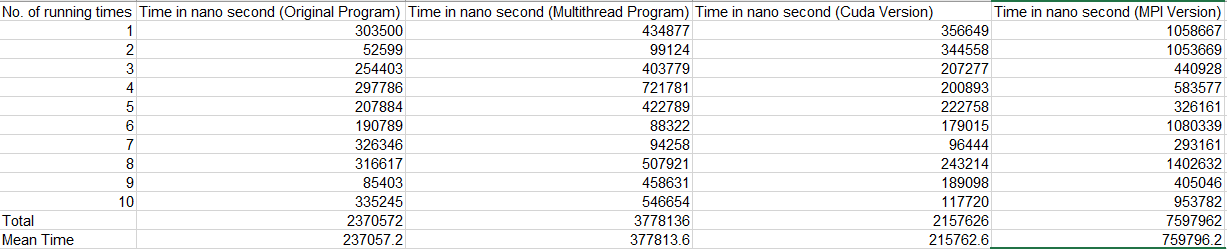
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,

0,0,0,255,255,255,0,0,0,0,0,0,0,0,0,0,0,0,0

};

1. Compare the mean running time of the MPI version with the original, multithread and CUDA versions.

Answer:



As shown in the above table the mean running time for the original program of image processing was 237057.2 nano seconds, the multithread program mean running time was 377813.6 nano seconds, CUDA version program mean time was 215762.6 nano seconds whereas the MPI Version of mean running time is 759796.2 nano seconds. The mean running time of MPI is higher than the other programs because in MPI version to create process it takes more process cycle so it consumes more time to run the program comparing to other.

## Linear Regression

1. Create another version of the linear regression program that uses 8 MPI instances, to each compute the error associated with a specific regression line. The master instance should run the main algorithm and use a different instance to compute the error associated with each m, c being explored.

Answer: Code for MPI linear regression

#include <stdio.h>

#include <math.h>

#include <time.h>

#include <mpi.h>

#include <malloc.h>

#include <sys/stat.h>

#include <stdlib.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* This program takes an initial estimate of m and c and finds the associated

\* rms error. It is then as a base to generate and evaluate 8 new estimates,

\* which are steps in different directions in m-c space. The best estimate is

\* then used as the base for another iteration of "generate and evaluate". This

\* continues until none of the new estimates are better than the base. This is

\* a gradient search for a minimum in mc-space.

\* This is a version of the linear regression program that uses 8 MPI instances,

\* to each compute the error associated with a specific regression line. The master

\* instance runs the main algorithm and uses a different instance to compute the

\* error associated with each m, c being explored

\* To compile:

\* mpicc -o 3\_a\_MPILinearRegression 3\_a\_MPILinearRegression.c -lm

\*

\* To run:

\* mpirun -n 9 ./3\_a\_MPILinearRegression

\*

\* Dr Kevan Buckley, University of Wolverhampton, 2018

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

typedef struct point\_t

{

double x;

double y;

} point\_t;

int n\_data = 1000;

point\_t data[];

double residual\_error (double x, double y, double m, double c)

{

double e = (m \* x) + c - y;

return e \* e;

}

//error function

double rms\_error (double m, double c)

{

int i;

double mean;

double error\_sum = 0;

for (i = 0; i < n\_data; i++)

{

error\_sum += residual\_error (data[i].x, data[i].y, m, c);

}

mean = error\_sum / n\_data;

return sqrt (mean);

}

//time function

int time\_difference(struct timespec \*start, struct timespec \*finish,

long long int \*difference) {

long long int ds = finish->tv\_sec - start->tv\_sec;

long long int dn = finish->tv\_nsec - start->tv\_nsec;

if(dn < 0 ) {

ds--;

dn += 1000000000;

}

\*difference = ds \* 1000000000 + dn;

return !(\*difference > 0);

}

//main function

int main () {

struct timespec start, finish;

long long int time\_elapsed;

clock\_gettime(CLOCK\_MONOTONIC, &start); //strting time

//variables

int rank, size;

int i;

double bm = 1.3;

double bc = 10;

double be;

double dm[8];

double dc[8];

double e[8];

double step = 0.01;

double best\_error = 999999999;

int best\_error\_i;

int minimum\_found = 0;

double pError = 0;

double baseMC[2];

double om[] = { 0, 1, 1, 1, 0, -1, -1, -1 };

double oc[] = { 1, 1, 0, -1, -1, -1, 0, 1 };

//Initilizing MPI

MPI\_Init (NULL, NULL);

MPI\_Comm\_size (MPI\_COMM\_WORLD, &size);//determining size of group

MPI\_Comm\_rank (MPI\_COMM\_WORLD, &rank);//determingn rank of group

be = rms\_error (bm, bc);

if (size != 9)

{

if (rank == 0)

{

printf

("This program is designed to run with exactly 9 processes.\n");

return 0;

}

}

while (!minimum\_found)

{

if (rank != 0)

{

i = rank - 1;

dm[i] = bm + (om[i] \* step);

dc[i] = bc + (oc[i] \* step);

pError = rms\_error (dm[i], dc[i]);

//sending count of element

MPI\_Send (&pError, 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD);

MPI\_Send (&dm[i], 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD);

MPI\_Send (&dc[i], 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD);

//receving count of element

MPI\_Recv (&bm, 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv (&bc, 1, MPI\_DOUBLE, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv (&minimum\_found, 1, MPI\_INT, 0, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

}

else

{

for (i = 1; i < size; i++)

{

//recving count of element

MPI\_Recv (&pError, 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv (&dm[i-1], 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

MPI\_Recv (&dc[i-1], 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD, MPI\_STATUS\_IGNORE);

if (pError < best\_error)

{

best\_error = pError;

best\_error\_i = i - 1;

}

}

// printf ("best m,c is %lf,%lf with error %lf in direction %d\n",

// dm[best\_error\_i], dc[best\_error\_i], best\_error, best\_error\_i);

if (best\_error < be)

{

be = best\_error;

bm = dm[best\_error\_i];

bc = dc[best\_error\_i];

}

else

{

minimum\_found = 1;

}

for (i = 1; i < size; i++)

{

MPI\_Send (&bm, 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD);

MPI\_Send (&bc, 1, MPI\_DOUBLE, i, 0, MPI\_COMM\_WORLD);

MPI\_Send (&minimum\_found, 1, MPI\_INT, i, 0, MPI\_COMM\_WORLD);

}

}

}

if(rank==0) {

printf ("minimum m,c is %lf,%lf with error %lf\n", bm, bc, be);

clock\_gettime(CLOCK\_MONOTONIC, &finish); //stopping time

time\_difference(&start, &finish, &time\_elapsed);

//printing time elapsed

printf("Time elapsed was %lldns or %0.9lfs\n", time\_elapsed,

(time\_elapsed/1.0e9));

}

MPI\_Finalize(); //closing MPI

return 0;

}

//data set

point\_t data[] = {

{65.06,140.50},{68.29,128.16},{65.16,115.78},{77.08,144.62},

{80.58,148.61},{69.14,116.53},{65.57,114.91},{82.72,137.98},

{83.65,124.22},{79.38,144.31},{78.26,132.38},{87.08,145.27},

{94.45,147.08},{92.35,164.15},{29.71,80.37},{ 4.60,55.88},

{93.22,160.01},{41.28,104.89},{70.80,148.31},{65.29,130.26},

{99.15,171.42},{ 3.63,57.34},{24.39,60.47},{ 5.30,63.83},

{83.72,129.70},{ 7.17,54.62},{ 2.55,51.94},{81.91,149.27},

{65.84,125.98},{52.85,102.69},{53.55,111.53},{68.67,133.94},

{71.18,133.57},{31.95,72.01},{68.73,111.86},{72.26,120.95},

{69.11,100.24},{57.60,107.51},{93.09,163.24},{97.86,152.47},

{76.21,142.09},{57.24,122.45},{54.07,109.88},{83.78,140.43},

{24.99,76.38},{51.46,97.96},{73.46,126.86},{84.30,126.29},

{12.04,47.93},{ 6.50,54.26},{33.87,73.33},{79.90,149.55},

{25.17,81.80},{37.22,79.55},{38.65,89.41},{39.91,100.49},

{76.60,157.45},{18.16,60.40},{65.87,126.29},{71.88,112.63},

{70.80,116.68},{80.05,135.15},{71.60,130.84},{92.36,182.25},

{81.46,133.35},{47.37,82.88},{78.32,140.80},{35.59,84.76},

{98.55,159.78},{69.63,136.22},{ 6.85,31.94},{68.92,135.57},

{39.62,75.64},{45.22,95.17},{56.10,113.68},{33.40,90.58},

{51.25,113.20},{31.87,71.31},{91.54,174.46},{34.49,91.73},

{49.83,102.67},{16.86,48.98},{30.06,88.34},{98.04,196.30},

{31.25,86.63},{38.39,91.43},{36.81,90.52},{29.41,87.37},

{30.49,84.56},{23.21,74.83},{86.64,149.79},{98.24,187.36},

{44.74,80.44},{89.56,162.75},{88.07,155.61},{65.31,139.57},

{ 5.98,36.29},{85.06,153.94},{32.95,79.00},{43.81,101.89},

{62.22,111.47},{18.81,50.86},{65.82,113.31},{34.13,68.03},

{28.85,70.54},{71.38,141.34},{35.08,81.06},{41.36,109.69},

{40.12,76.78},{77.54,133.76},{50.18,106.00},{69.88,128.59},

{85.04,147.04},{66.82,128.00},{ 4.62,65.65},{78.07,145.26},

{33.81,86.56},{30.94,86.44},{78.63,158.12},{30.00,94.75},

{68.43,134.65},{66.31,122.03},{72.19,140.43},{43.52,112.56},

{58.36,129.89},{63.49,109.82},{72.93,140.32},{ 0.91,32.13},

{92.36,166.03},{58.59,132.02},{19.62,65.12},{73.70,127.87},

{ 7.10,49.29},{96.54,188.39},{11.72,29.18},{85.16,146.33},

{86.71,149.72},{82.62,137.78},{ 0.54,36.93},{43.02,101.71},

{56.52,96.81},{83.48,141.45},{87.95,150.02},{41.46,79.96},

{14.22,41.84},{ 6.09,45.81},{70.28,124.44},{70.57,126.73},

{21.65,60.07},{55.38,112.19},{ 1.10,46.57},{99.19,175.35},

{20.13,74.04},{36.30,79.86},{15.00,64.84},{55.51,111.39},

{39.08,81.62},{26.40,52.68},{68.19,136.41},{15.15,66.96},

{10.81,41.02},{45.70,113.27},{42.56,109.98},{80.27,131.75},

{45.41,95.82},{31.53,73.92},{40.84,93.37},{54.49,100.20},

{98.15,165.46},{ 8.63,51.82},{ 2.49,53.66},{73.88,147.16},

{13.36,63.33},{64.39,141.51},{84.32,157.95},{67.96,107.69},

{ 6.78,39.40},{77.95,141.91},{51.91,109.56},{47.62,92.81},

{27.17,63.62},{61.71,117.83},{38.62,86.65},{60.72,124.18},

{67.81,134.12},{31.53,71.56},{14.00,61.96},{19.50,59.01},

{90.95,135.89},{91.03,158.12},{33.65,82.03},{12.33,39.88},

{83.04,132.29},{50.85,112.28},{70.55,139.16},{64.07,121.96},

{35.31,92.95},{ 3.02,59.14},{46.36,105.52},{ 9.08,51.06},

{97.16,170.35},{ 5.22,48.76},{37.20,90.63},{60.93,127.67},

{89.32,178.38},{69.67,138.27},{19.36,63.13},{33.04,71.81},

{39.92,65.58},{50.92,124.15},{26.92,77.66},{29.14,82.75},

{ 0.85,40.60},{27.69,81.38},{81.94,142.14},{31.80,82.99},

{32.47,87.54},{ 4.43,39.37},{86.44,160.01},{62.87,132.53},

{48.20,103.22},{95.55,163.90},{82.81,152.15},{41.02,71.00},

{51.46,91.99},{42.61,92.40},{ 9.37,55.95},{12.68,62.08},

{41.45,102.49},{67.42,123.38},{83.35,157.06},{25.04,62.00},

{17.95,67.64},{74.23,142.58},{ 2.20,42.77},{85.41,152.29},

{63.83,111.25},{35.59,72.75},{ 9.84,59.26},{39.47,99.05},

{16.96,54.75},{27.80,92.37},{68.88,146.45},{80.19,160.65},

{41.22,99.46},{47.87,102.96},{30.06,77.15},{13.92,61.54},

{40.95,107.20},{48.85,98.44},{20.23,71.85},{37.48,87.76},

{46.93,100.97},{91.46,166.75},{66.78,127.65},{69.80,128.92},

{56.99,113.89},{64.99,118.91},{60.11,133.15},{11.00,71.54},

{ 8.09,61.76},{33.73,89.91},{14.97,47.57},{ 1.82,55.82},

{55.44,124.87},{18.43,54.83},{51.17,107.27},{14.12,58.21},

{61.49,123.07},{64.13,109.73},{42.54,92.26},{65.15,144.90},

{53.75,105.92},{57.63,114.92},{13.89,45.67},{52.03,101.50},

{63.30,116.07},{89.52,143.82},{63.83,113.06},{80.08,143.83},

{ 6.92,47.55},{57.40,103.92},{64.21,125.58},{46.81,106.30},

{91.30,151.08},{76.66,155.83},{47.48,112.92},{54.00,99.69},

{40.35,94.01},{77.68,150.21},{87.39,143.66},{76.78,146.43},

{54.80,105.64},{23.96,75.02},{45.90,86.75},{17.85,40.30},

{44.08,99.76},{62.84,119.24},{ 4.66,40.51},{61.01,125.84},

{96.70,168.86},{54.75,112.30},{ 8.70,53.95},{95.79,174.98},

{99.36,150.00},{69.48,127.58},{44.39,102.81},{43.75,98.47},

{45.91,95.15},{11.07,50.20},{85.58,141.73},{26.65,72.63},

{16.74,71.23},{43.53,83.72},{ 1.76,42.14},{23.78,79.11},

{26.59,84.39},{43.96,99.13},{47.07,120.85},{90.53,150.59},

{63.14,121.81},{17.75,73.96},{38.85,90.89},{77.91,145.78},

{64.09,115.44},{90.91,171.76},{53.97,123.30},{36.48,77.64},

{91.35,179.25},{70.46,131.28},{84.31,156.60},{53.08,103.03},

{26.80,62.81},{74.76,160.33},{21.42,60.42},{77.34,148.78},

{17.14,58.22},{69.61,127.58},{90.26,159.43},{69.55,132.52},

{82.69,132.50},{96.81,158.98},{99.48,182.04},{34.32,96.27},

{ 9.71,55.14},{46.94,102.50},{71.31,135.96},{66.98,121.76},

{98.22,165.52},{89.51,166.77},{ 6.33,30.69},{65.11,122.80},

{82.86,156.93},{16.40,66.05},{35.25,94.44},{94.61,147.92},

{96.45,166.53},{12.94,45.40},{36.88,76.80},{42.23,102.36},

{38.52,79.49},{28.72,69.83},{ 5.47,47.33},{31.42,76.78},

{89.61,149.18},{54.33,120.28},{29.57,82.48},{ 3.24,37.35},

{39.97,92.40},{38.31,88.57},{64.15,101.51},{66.60,124.22},

{ 8.39,53.51},{53.52,120.09},{54.00,104.21},{69.22,116.75},

{66.90,125.93},{16.18,60.34},{36.52,106.85},{ 3.83,57.76},

{63.60,111.98},{16.08,64.93},{14.51,60.59},{55.61,91.17},

{36.35,104.47},{22.41,63.89},{32.32,78.96},{44.95,91.99},

{12.10,66.00},{76.41,146.29},{57.89,105.09},{22.74,83.64},

{68.93,135.75},{56.80,101.13},{44.07,95.81},{ 2.22,39.62},

{34.64,80.94},{91.59,175.08},{ 7.54,56.50},{90.29,148.76},

{ 0.18,59.80},{59.07,108.26},{24.80,83.05},{58.62,123.01},

{16.02,56.86},{38.71,92.88},{99.87,174.65},{59.02,109.10},

{82.68,153.35},{24.15,99.81},{64.58,117.31},{21.24,69.38},

{30.70,80.02},{ 1.41,40.58},{59.40,111.46},{52.91,94.94},

{80.90,151.48},{ 9.73,51.73},{70.74,131.92},{92.53,157.93},

{93.11,160.17},{78.60,142.44},{39.51,103.00},{27.05,76.90},

{53.77,102.32},{82.71,163.25},{32.13,74.88},{73.27,134.77},

{18.37,66.43},{96.87,160.13},{14.74,59.16},{29.79,66.13},

{81.48,156.13},{50.93,132.81},{34.14,90.11},{97.06,177.46},

{43.59,118.51},{87.77,155.51},{26.78,56.57},{67.37,135.73},

{43.72,82.53},{78.90,151.03},{67.47,128.75},{73.54,142.62},

{83.58,146.88},{60.97,105.98},{61.39,121.73},{97.62,170.99},

{22.87,75.14},{14.33,60.35},{10.18,41.30},{71.87,138.38},

{ 6.67,47.75},{ 0.18,31.24},{63.19,127.26},{30.39,85.15},

{ 5.40,37.14},{30.71,94.33},{80.79,127.64},{60.59,107.25},

{13.56,65.29},{87.52,167.00},{14.84,53.76},{41.12,94.91},

{10.69,58.47},{53.25,89.18},{81.43,152.09},{33.06,79.09},

{15.51,51.17},{79.41,141.15},{70.48,139.28},{94.07,168.01},

{51.12,107.53},{ 5.67,45.68},{58.79,131.63},{19.47,73.96},

{31.67,71.86},{26.69,74.47},{71.45,147.52},{86.49,138.97},

{34.95,73.25},{73.30,148.44},{71.11,133.22},{94.75,166.25},

{88.11,147.28},{19.30,67.43},{90.05,164.25},{15.78,61.88},

{25.32,70.27},{28.72,74.55},{79.45,132.86},{25.88,72.08},

{31.14,86.32},{35.98,84.84},{ 5.71,44.50},{76.81,142.72},

{57.22,111.49},{87.46,139.57},{14.92,62.58},{83.44,141.89},

{23.73,58.30},{50.44,86.37},{20.16,70.79},{38.51,83.48},

{40.26,75.05},{15.57,53.06},{13.83,71.11},{19.29,73.30},

{91.73,146.04},{31.42,95.31},{24.64,72.44},{36.80,91.84},

{30.03,98.86},{11.40,51.19},{ 9.15,45.77},{65.77,120.82},

{90.64,148.02},{49.00,115.15},{33.13,82.17},{11.11,47.41},

{57.31,127.10},{22.37,75.81},{39.67,95.67},{29.11,75.90},

{45.84,86.16},{45.13,92.70},{ 5.48,50.66},{24.90,85.23},

{63.62,106.70},{82.49,142.53},{71.25,141.47},{36.64,87.11},

{ 1.96,51.52},{50.21,125.63},{61.50,110.30},{93.82,163.58},

{95.33,150.35},{ 9.32,63.49},{62.76,120.10},{ 3.44,63.32},

{41.49,97.46},{74.59,128.03},{77.61,137.92},{68.64,121.30},

{62.10,121.86},{34.90,93.88},{30.26,96.06},{85.65,165.07},

{54.94,128.21},{33.16,61.94},{ 4.82,50.78},{14.96,67.98},

{41.74,94.46},{86.33,148.59},{78.50,151.64},{78.74,124.22},

{83.50,154.44},{69.30,150.92},{86.91,133.87},{57.76,101.58},

{ 9.27,58.05},{52.10,108.71},{ 6.70,52.52},{15.06,75.65},

{40.28,100.24},{72.78,141.77},{56.15,109.93},{34.21,91.19},

{46.13,95.81},{90.38,157.10},{79.42,137.37},{23.85,75.78},

{16.94,61.69},{28.30,63.59},{45.94,116.74},{21.90,72.69},

{86.71,144.20},{50.15,101.35},{47.52,104.48},{76.50,133.92},

{16.59,70.95},{30.07,81.69},{95.78,162.28},{79.85,139.57},

{32.16,84.87},{59.81,100.58},{12.73,55.16},{68.40,143.18},

{88.39,142.70},{70.92,141.63},{16.91,65.12},{65.99,138.01},

{29.08,65.78},{94.66,143.54},{17.46,51.83},{40.45,100.97},

{80.61,154.88},{55.99,101.51},{50.64,94.60},{89.71,154.05},

{76.69,128.98},{71.69,135.85},{ 6.01,40.29},{96.97,153.79},

{51.33,104.76},{26.38,78.48},{74.84,124.85},{87.23,154.70},

{43.79,101.79},{36.34,86.15},{93.97,152.26},{91.93,168.48},

{99.95,153.69},{ 6.24,47.10},{33.57,94.58},{25.15,90.07},

{ 2.71,47.98},{55.65,96.69},{56.66,103.53},{15.84,75.38},

{13.10,49.82},{31.59,87.78},{10.07,52.90},{64.69,128.15},

{ 0.29,51.37},{54.63,117.56},{16.48,47.12},{68.05,111.33},

{65.05,117.98},{50.73,105.93},{38.21,92.02},{84.19,152.96},

{60.62,120.01},{71.76,130.65},{99.68,172.79},{16.20,50.34},

{16.99,79.00},{29.19,59.02},{69.67,116.57},{ 2.04,45.22},

{53.52,113.83},{38.61,86.66},{20.82,71.20},{45.51,91.72},

{83.51,133.29},{95.95,166.47},{99.30,187.23},{57.30,129.30},

{54.98,98.10},{60.47,112.79},{57.56,119.35},{18.50,62.58},

{75.27,140.67},{ 5.79,48.06},{94.17,157.17},{25.72,77.59},

{66.55,135.05},{53.82,122.03},{44.12,103.61},{86.49,159.15},

{20.06,64.54},{97.78,170.13},{61.95,119.09},{58.91,128.99},

{75.55,152.04},{28.76,63.60},{61.54,100.40},{74.51,125.14},

{ 6.62,30.47},{45.73,90.74},{79.17,140.18},{42.13,94.38},

{ 3.52,42.69},{20.27,53.88},{47.35,118.66},{64.33,122.29},

{14.34,63.88},{48.22,104.79},{79.32,140.01},{20.79,81.66},

{21.07,49.96},{48.95,125.70},{99.68,178.12},{50.76,107.99},

{59.08,111.82},{32.86,83.53},{37.71,104.74},{ 6.95,41.30},

{61.04,126.11},{23.82,75.61},{ 3.64,39.77},{11.31,73.23},

{65.13,128.77},{45.97,87.42},{51.46,105.02},{50.48,106.56},

{29.63,85.41},{73.86,144.97},{56.80,127.73},{83.76,154.70},

{78.47,134.95},{69.16,121.95},{60.20,102.58},{89.65,144.45},

{47.31,93.28},{87.81,155.29},{98.46,180.88},{ 8.62,66.81},

{18.21,62.82},{16.90,62.50},{66.64,125.01},{35.03,90.13},

{ 1.19,42.82},{39.25,87.90},{95.51,185.50},{52.60,130.29},

{42.18,98.66},{43.05,101.51},{66.40,133.64},{93.74,170.39},

{51.13,109.28},{72.79,138.29},{45.04,114.16},{15.80,70.59},

{94.90,151.00},{99.25,176.44},{65.26,137.40},{36.04,72.00},

{ 0.40,43.60},{86.91,138.43},{41.29,105.27},{56.72,116.01},

{20.01,65.99},{ 1.44,24.72},{60.94,134.19},{17.93,49.10},

{51.69,111.06},{98.77,171.60},{18.49,49.95},{98.53,168.61},

{39.04,91.25},{33.96,73.93},{58.72,122.23},{54.30,94.70},

{19.38,77.47},{59.93,100.50},{57.00,99.68},{19.08,49.37},

{62.19,125.66},{69.26,126.43},{63.67,124.79},{33.97,73.46},

{64.05,129.47},{90.41,160.25},{90.68,141.75},{58.43,120.99},

{47.72,98.06},{57.18,115.01},{21.72,71.98},{78.91,144.12},

{96.99,161.88},{90.98,144.82},{99.32,167.24},{92.02,168.45},

{76.83,146.88},{13.22,55.99},{22.96,85.34},{ 8.79,58.31},

{39.57,99.22},{95.41,166.55},{87.15,159.87},{77.74,155.73},

{30.21,80.03},{15.00,65.07},{ 1.25,36.22},{85.49,163.08},

{ 6.02,49.47},{38.29,90.72},{ 8.87,44.71},{21.80,68.59},

{69.88,115.45},{14.01,52.20},{15.63,62.15},{ 2.54,44.44},

{29.85,77.08},{83.24,154.98},{27.89,75.98},{87.91,160.12},

{66.17,134.31},{93.36,157.11},{64.99,130.27},{60.52,131.00},

{ 5.75,41.05},{87.46,144.19},{47.47,94.07},{ 8.26,43.29},

{42.36,100.58},{12.30,60.00},{16.23,60.55},{24.60,67.52},

{18.49,64.78},{80.04,139.33},{46.05,105.21},{16.89,58.52},

{ 6.63,62.86},{48.83,96.30},{39.23,100.05},{96.70,187.16},

{68.76,122.37},{55.28,120.83},{11.54,39.88},{ 6.99,41.18},

{96.16,173.94},{15.73,46.98},{91.07,145.49},{16.07,80.57},

{40.17,99.22},{64.93,118.55},{38.08,83.04},{ 6.19,54.52},

{86.09,150.91},{56.36,104.25},{41.95,97.61},{98.49,166.66},

{79.59,141.04},{32.42,80.76},{61.50,115.50},{40.85,88.12},

{39.26,97.38},{52.55,125.88},{72.67,152.23},{23.07,75.82},

{12.61,41.49},{89.50,132.51},{90.08,173.93},{54.88,116.36},

{41.88,88.54},{68.23,112.00},{13.83,37.32},{57.71,108.79},

{ 7.49,57.40},{80.46,126.17},{54.78,112.41},{85.85,156.33},

{39.31,84.84},{ 0.57,11.17},{92.41,145.81},{55.63,119.56},

{65.66,108.36},{72.85,136.35},{36.00,93.64},{45.38,97.11},

{99.11,144.04},{27.37,82.63},{82.92,153.95},{17.88,74.85},

{23.32,60.05},{68.17,134.30},{75.78,134.99},{80.26,149.31},

{11.84,49.92},{98.01,156.92},{ 1.82,38.12},{17.27,67.28},

{ 2.30,45.03},{87.16,152.81},{ 5.12,47.75},{47.91,103.87},

{45.76,107.49},{51.05,121.73},{43.92,104.41},{ 2.24,23.42},

{19.45,72.35},{ 4.27,46.17},{19.93,62.42},{49.57,99.46},

{88.25,160.42},{ 6.47,36.61},{20.82,60.18},{24.10,66.11},

{97.18,175.27},{13.49,43.49},{93.97,161.19},{97.02,143.79},

{94.51,162.20},{29.71,87.40},{72.56,128.83},{24.59,74.45},

{82.46,150.71},{69.11,131.86},{14.59,47.06},{93.10,154.28},

{11.42,58.70},{49.15,117.48},{65.69,130.75},{66.85,132.81},

{30.30,66.09},{33.49,82.24},{19.09,78.38},{76.50,146.34},

{ 2.48,22.05},{93.19,175.64},{52.16,122.32},{89.50,157.08},

{ 2.70,34.81},{ 3.19,31.68},{58.98,110.55},{63.36,110.02},

{90.50,154.14},{72.47,124.43},{26.44,75.60},{78.04,147.11},

{20.76,66.40},{26.99,78.87},{10.23,40.71},{34.27,77.02},

{41.62,111.74},{57.40,139.36},{16.31,76.09},{93.55,170.94},

{30.47,65.32},{78.87,153.63},{69.79,132.46},{62.10,129.93},

{55.54,107.90},{44.31,116.79},{33.50,80.09},{30.62,86.17},

{ 9.22,55.76},{57.81,111.98},{40.80,85.30},{ 1.13,51.94},

{37.18,73.32},{11.92,64.57},{ 5.96,40.39},{29.06,85.72},

{45.79,93.94},{33.97,80.99},{34.65,71.49},{44.26,77.08},

{47.49,104.46},{66.18,145.70},{27.32,76.18},{83.78,154.25},

{ 3.86,49.52},{ 2.95,46.28},{53.75,120.20},{26.85,71.84},

{26.51,73.85},{99.02,164.59},{66.62,120.46},{ 4.36,44.96},

{76.90,140.62},{21.49,69.85},{23.49,83.78},{99.09,165.36},

{60.04,132.84},{23.10,54.47},{40.68,86.11},{31.91,106.31},

{37.12,82.67},{39.39,86.60},{40.18,105.70},{51.35,129.06},

{58.58,112.39},{31.78,68.54},{65.77,125.66},{50.22,113.81},

{22.42,52.72},{73.70,148.23},{27.37,99.01},{41.00,94.43},

{56.35,110.68},{85.69,159.14},{52.28,111.81},{25.12,91.74},

{29.30,75.16},{32.79,95.38},{12.83,43.35},{83.30,136.94},

{41.34,77.86},{42.87,89.04},{13.24,62.89},{62.62,138.13},

{14.40,65.55},{27.11,81.45},{38.38,61.89},{17.27,72.58},

{29.53,85.39},{81.42,136.71},{12.21,48.86},{74.83,120.84},

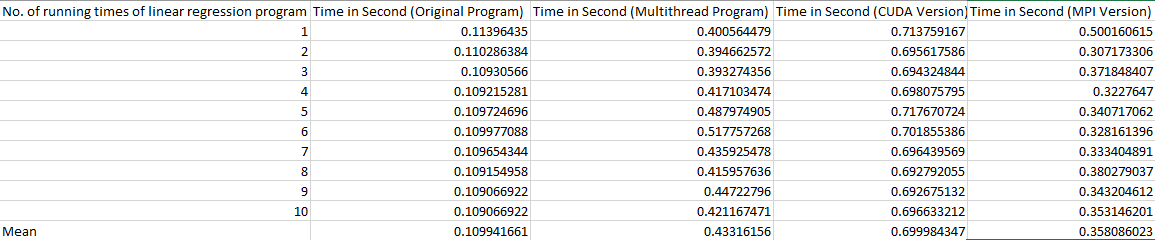
{74.97,138.85},{11.93,41.04},{22.62,64.23},{89.69,139.61},

{40.38,76.74},{65.74,124.89},{71.41,149.81},{50.76,99.00}

};

1. Compare the mean running time of the MPI version with the original, multithread and CUDA versions.

Answer:



According to the above table the mean running time of the original program was 0.109941661 seconds, multithread version was 0.43316156 seconds, CUDA version was 0.699984347 seconds, MPI version is 0.358086023 seconds. Comparing the running time of the original program and MPI version, MPI version took more time to run because in linear regression of MPI it use 8 MPI instances which means each instance compute the error that is attached with a specific regression line. Because of this reasons MPI version takes more time to run.

# Verbose Repository Log

commit 33fe35c31fee2abdc1950dade68ba386fd061d2a

Author: Nancyd01 <dahalnancy1@gmail.com>

Date: Mon May 6 18:01:30 2019 +0545

Password cracking using POSIX

commit 5006b67cd13b32f8f6764b8ddef7154abb3f1c75

Author: Nancyd01 <dahalnancy1@gmail.com>

Date: Mon May 6 17:55:19 2019 +0545

Linear regression using POSIX uplaoded

commit 5a83656a418cb2ce4c9c975649646f57f452b38c

Author: Nancyd01 <dahalnancy1@gmail.com>

Date: Mon May 6 17:53:55 2019 +0545

Image processing MPI uplaoded

commit c014a1683a5d0255db9982a71816be6a34513eaa

Author: Nancyd01 <dahalnancy1@gmail.com>

Date: Mon May 6 17:51:23 2019 +0545

MPI files added

commit 2a0f7896699bd4da919d366d39a0dbd4e18c794e

Author: Nancyd01 <dahalnancy1@gmail.com>

Date: Mon May 6 17:49:24 2019 +0545

Cuda file uploaded