# Operating System

Mitul Kabutarwala

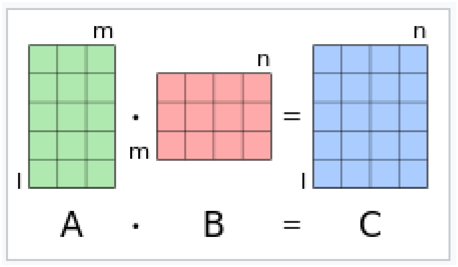
## Project Report

The project was to do matrix multiplication with multithreading in C programming language

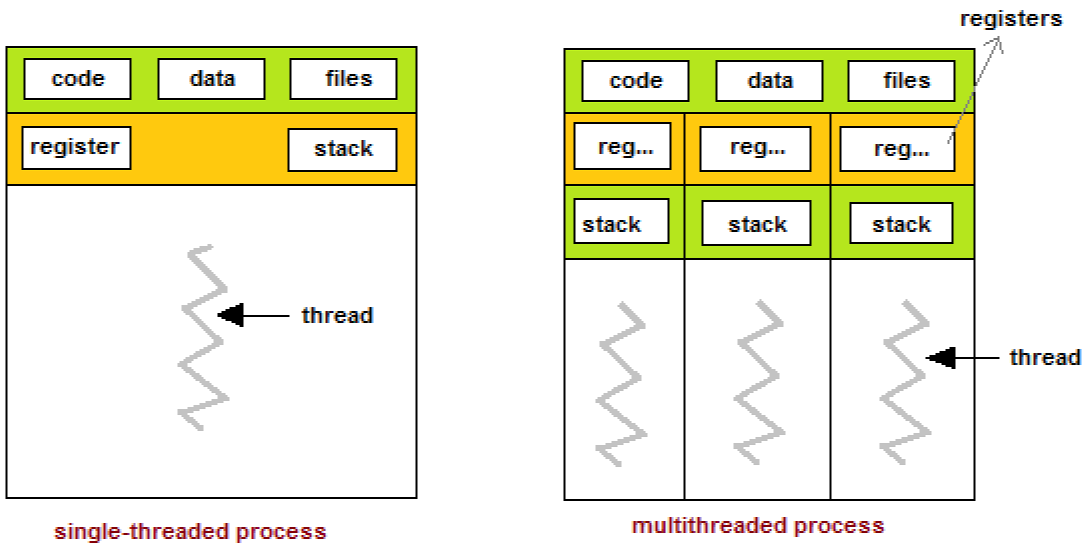
Matrix:- a matrix is a rectangular array or table of numbers, symbols, or expressions, arranged in rows and columns. For example, the dimension of the matrix below is m × n, because there are m rows and n columns [1].

$$\begin{array}{c|cccc|} & 1 & 2 & \cdots & n \\ 1 & a_{11} & a_{12} & \cdots & a_{1n} \\ 2 & a_{21} & a_{22} & \cdots & a_{2n} \\ 3 & a_{31} & a_{32} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ m & a_{m1} & a_{m2} & \cdots & a_{mn} \end{array}$$

Matrix multiplication:- In mathematics, matrix multiplication is a binary operation that produces a matrix from two matrices. For matrix multiplication, the number of columns in the first matrix must be equal to the number of rows in the second matrix. The resulting matrix, known as the matrix product, has the number of rows of the first and the number of columns of the second matrix. The product of matrices A and B is then denoted simply as AB [2].



Multithreading:- Multithreading is the ability of a central processing unit (CPU) (or a single core in a multi-core processor) to provide multiple threads of execution concurrently, supported by the **operating system**. This approach differs from multiprocessing. In a multithreaded application, the threads share the resources of a single or multiple core [3].



Pthread in C programming:- Pthreads are a simple and effective way of creating a multi-threaded application. This introduction to pthreads shows the basic functionality – executing two tasks in parallel and merging back into a single thread when the work has been done [4].

In cases where you must wait for a number of tasks to be completed before an overall task can proceed, **barrier synchronization** can be used. POSIX threads specifies a synchronization object called a **barrier**, along with barrier functions. The functions create the barrier, specifying the number of threads that are synchronizing on the barrier, and set up threads to perform tasks and wait at the barrier until all the threads reach the barrier. When the last thread arrives at the barrier, all the threads resume execution [5].

When the required number of threads have called **pthread_barrier_wait()** specifying the barrier, the constant PTHREAD_BARRIER_SERIAL_THREAD is returned to one unspecified thread and 0 is returned to each of the remaining threads. The barrier is then reset to the state it had as a result of the most recent **pthread_barrier_init()** function that referenced it [5].

C programming Code for Matrix multiplication with multithreading.

```
#include<stdio.h>
#include<stdlib.h>
#include<time.h>
#include<sys/time.h>
#include<math.h>
#include<pthread.h>

#define matrix_length     216

static struct timeval start,end;

// to calculate time take by program to execute
void start_timer()
{
           gettimeofday(&start,(struct timezone*)NULL);
}

float stop_timer()
{
           gettimeofday(&end,(struct timezone*)NULL);
           float elapsed_time = (float)(end.tv_sec - start.tv_sec)+(end.tv_usec -
start.tv_usec)/1000000.0;

return elapsed_time;
}

void start_timer();
float stop_timer();
pthread_barrier_t hold;
int a[matrix_length][matrix_length], b[matrix_length][matrix_length],
ab[matrix_length][matrix_length]; //creating 2d array for matrix a,b and ab
int value;

void mm(int *val)  //matrix multiplication process
{
           int i,j,k,temp;
           int start= (*val * matrix_length)/value;
           int end= ((*val+1) * matrix_length)/value;
           printf("%d\t%d\n",start,end);
       for(i=start;i<end;i++)
       {
          for(j=0;j<matrix_length;j++)
```

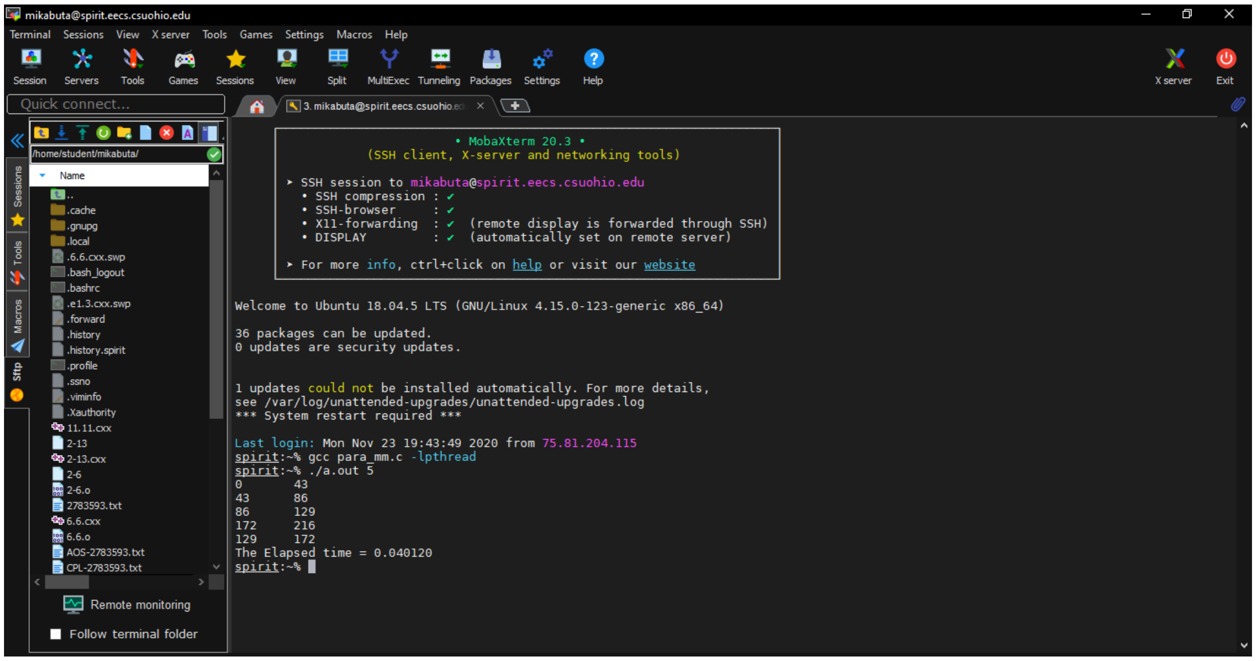
```
        {
                temp=0;
            for(k=0;k<matrix_length;k++)
            {
              temp=temp+(a[i][k]*b[k][j]);
            }
            ab[i][j]=temp;
        }
      }
pthread_barrier_wait(&hold);
}

int main(int argc,char *argv[])
{
        int i,j,k;
        float e_time;
        value = atoi(argv[1]);  //take argument
        pthread_t child[value];
        pthread_barrier_init(&hold,NULL,value+1);    //init pthread barrier
        for(i=0;i<matrix_length;i++){    //assigning value as 1 to every position in matrix
        for(j=0;j<matrix_length;j++)
                {
                        a[i][j]=1;
                        b[i][j]=1;
                }
        }
        int range[16];
        start_timer(); // Timer Started

        for(i=0;i<value;i++)
                {
                        range[i]=i;
                        if(pthread_create(&child[value],NULL,(void *)mm,&range[i])!=0)
                        {
                                printf("Error Creating Thread");
                        }
                }

        pthread_barrier_wait(&hold);    // pthread barrier
        e_time = stop_timer();
        printf("The Elapsed time = %f \n", e_time);

  return 0;
}
```

The code equally divides the thread into the given argument and keep the track on the time taken to solve the matrix. It also shows which thread is working on which part of the matrix.

Output

Command:-

- ✓ gcc para_mm.c -lpthread
- ✓ ./a.out 5



```
                    • MobaXterm 20.3 •
          (SSH client, X-server and networking tools)

 ➤ SSH session to mikabuta@spirit.eecs.csuohio.edu
   • SSH compression : ✔
   • SSH-browser     : ✔
   • X11-forwarding  : ✔  (remote display is forwarded through SSH)
   • DISPLAY         : ✔  (automatically set on remote server)

 ➤ For more info, ctrl+click on help or visit our website

Welcome to Ubuntu 18.04.5 LTS (GNU/Linux 4.15.0-123-generic x86_64)

36 packages can be updated.
0 updates are security updates.


1 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log
*** System restart required ***

Last login: Mon Nov 23 19:43:49 2020 from 75.81.204.115
spirit:~% gcc para_mm.c -lpthread
spirit:~% ./a.out 5
0       43
43      86
86      129
172     216
129     172
The Elapsed time = 0.040120
spirit:~% 
```

**Reference:**

1. Matrix, https://en.wikipedia.org/wiki/Matrix_(mathematics)
2. Matrix multiplication, https://en.wikipedia.org/wiki/Matrix_multiplication
3. Multithreading, https://en.wikipedia.org/wiki/Multithreading_(computer_architecture)#:~:text=In%20computer%20architecture%2C%20multithreading%20is,supported%20by%20the%20operating%20system.
4. Pthread in C programming, https://timmurphy.org/2010/05/04/pthreads-in-c-a-minimal-working-example/
5. Pthread barrier, https://docs.oracle.com/cd/E19120-01/open.solaris/816-5137/gfwek/index.html