LAB 01: MULTITHREDED PROGRAMMING

1. Mục tiêu:

- SV tìm hiểu và sử dụng thư viện Posix Thread trên linux
- Viết một chương trình multithread đơn giản minh họa tính hiệu quả của chương trình khi được song song hóa.
  - Các hàm Posix threads cơ bản:
    - pthread_create(), pthread_attr_init(), pthread_attr_destroy(), pthread_exit().
    - pthread_join(), pthread_detach(), pthread_attr_getdetachsate(),
    - pthread_attr_setdetachstate().
    - Pthread_mutex_init(), pthread_mutex_destroy(), pthread_mutexattr_init() ,
    - pthread_mutexattr_destroy(), pthread_mutex_lock(), pthread_mutex_unlock().

2. Nội dung:

2.1. Yêu cầu:

- Sinh viên hoàn thiện chương trình nhân hai ma trận vuông NxN bằng multithread programming sử dụng thư viện lập trình POSIX pThread trên hệ điều hành Linux.

2.2. Đoạn code mẫu của chương trình nhân hai ma trận NxN tuần tự:

```
/* Matrix multiplication */

/* source code from Quinn's book */

#include <stdio.h>

#include <stdlib.h>

/* N <= 10000*/

#define N 1000

#define THRESHOLD 65535

double a[N][N], b[N][N], c[N][N];

int main() {

        init_matrix( a, N, N, 1);

        init_matrix( b, N, N, 1);
```

```
        mm( 0, 0,
                0, 0,
                0, 0,
                N,
                N,
                N);

        //print_to_file(double *mt, int row, int col, char *name) {
        print_to_file(a, N, N, "inA");
        print_to_file(b, N, N, "inB");
        print_to_file(c, N, N, "outC");
        return 0; //OK
}
void init_matrix(double *MT, int row, int col,int seed) {
    int i,j;
    for (i=0;i<row;i++) {
        for (j=0;j<col;j++) {
            MT[i*col+j] = 1;
        }
    }
}
void mm(int crow, int ccol , /* go'c cua block C*/
            int arow, int acol , /* go'c of A block*/
            int brow, int bcol , /* go'c of B block*/
```

```
                int l,          /* block A is l x m*/
                int m,          /* block B is m x n*/
                int n)          /* block C is l x n*/
{
    int lhalf[3], mhalf[3], nhalf[3]; /*kich thuoc cua khoi 1/4*/
    int i,j,k;
    double *aptr, *bptr, *cptr;
    if (m*n > THRESHOLD) {

        /* Neu A, B khong vua khoi thi giam di 1/2*/
        lhalf[0] = 0; lhalf[1] = l/2; lhalf[2] = l - l/2;
        mhalf[0] = 0; mhalf[1] = m/2; mhalf[2] = m - m/2;
        nhalf[0] = 0; nhalf[1] = n/2; nhalf[2] = n - n/2;

        for (i=0; i<2; i++)
            for (j=0;j<2;j++)
                for (k=0;k<2;k++)
                    mm(   crow + lhalf[i], ccol + mhalf[j],
                          arow + lhalf[i], acol + mhalf[k],
                          brow + mhalf[k], bcol + nhalf[j],
                          lhalf[i+1], mhalf[k+1], nhalf[j+1] );
    }else {
      /* B fits in cache -- do standard multiply */
        for (i=0; i<l ; i++)
```

```
            for (j=0;j<n;j++) {
                cptr = &c [crow+i] [ccol + j];
                aptr = &a [arow+i] [acol];
                bptr = &b [brow] [bcol + j];

                for (k=0; k<m; k++) {
                    *cptr += *(aptr++) * *bptr;
                    bptr += N;
                }
            }
        }
}

void print_to_file(double *mt, int row, int col, char *name) {
        FILE *f;
        f= fopen(name,"w");
        if (f == NULL){
            printf("Cannot write to file %s\n",name);
            exit(1);
        }
        int i,j;
        for (i=0;i<row;i++) {
            for (j=0;j<col;j++)
                fprintf(f,"%6.0f ", *(mt + i*col +j));
```

```
            fprintf(f,"\n");
        }
        fclose(f);
}
void print(double *mt, int row, int col) {
        int i,j;
        for (i=0;i<row;i++) {
                for (j=0;j<col;j++)
                        printf("%6.0f ", *(mt + i*col +j));
                        printf("\n");
        }
}
```

2.3. Nội dung bài thực hành:

Hoàn thiện chương trình nhân hai ma trận NxN song song dùng thư viện lập trình POSIX pThread.

Gợi ý:

- Tạo thêm nhiều thread.

3. Bài tập

SV thực hiện các bài tập sau:

3.1. Hiện thực 3 giải thuật sắp xếp đã yêu cầu, SV cần tìm hiểu các hàm Posix threads để hoàn tất hàm main() trong chương trình.
3.2 Viết chương trình nhân hai vector
3.3 Viết chương trình nhân hai ma trận
Hướng dẫn: SV sử dụng hàm pthread_mutex_lock() và pthread_mutex_unlock() trong trường hợp đồng bộ dữ liệu dùng chung giữa các thread.
3.4 Viết chương trình chơi cờ caro trên bàn cờ N*N. Đề xuất mô hình multithread cho giải thuật tìm kiếm nước đi tốt nhất và khảo sát tính hiệu quả, thời gian chạy của giải thuật được song song hóa bằng multithread.

3.5 Viết chương trình đặt N quân hậu vào bàn cờ tổng quát N *N ô. Đề xuất mô hình multithread cho giải thuật đặt hậu và khảo sát tính hiệu quả, thời gian chạy của giải thuật được song song hóa bằng multithread.