Operating Systems - Lab 08 Tasks

Ibrahim Johar Farooqi
23K-0074
BAI-4A
26 March 2025

Lab Questions:

Task 1 - a:
Declare three float arrays A, B and C each of size 1e7 (10000000) and perform the operation C = A + B for each element of A, B and C.
a) Write, compile, and run serial code.

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

#define N 10000000 //10 million

float A[N], B[N], C[N];

void initalise()
{
   for(int i = 0; i < N; i++)
   {
       //random values btw 0 to 100
       A[i] = (float)(rand() % 100);
       B[i] = (float)(rand() % 100);
   }
}

void serial_addition()
{
   for (int i = 0; i < N; i++)
   {
       C[i] = A[i] + B[i];
   }
}

int main()
{
   clock_t start, end;
```

```
    double time_taken;

    initalise(); //A & B initialised

    start = clock(); //start time

    serial_addition(); //computation performed here

    end = clock(); //end time

    time_taken = ((double)(end-start)) / CLOCKS_PER_SEC;

    printf("time taken (serial): %.6f secs\n", time_taken);

    return 0;
}
```

```
ibrahim-johar@ibrahim-johar-VirtualBox: ~/08
ibrahim-johar@ibrahim-johar-VirtualBox:~$ cd /home/ibrahim-johar/08/
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$ gcc -o q1_a q1_a.c -pthread
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$ ./q1_a
time taken (serial): 0.111969 secs
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$
```

Task 1 - b:

b) Write concurrent code where 10 worker threads will equally divide the computational workload.

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <time.h>

#define N 10000000  //10 million
#define NUM_THREADS 10 //num of worker threads

float A[N], B[N], C[N];

typedef struct {
    int start;
```

```
    int end;
} thread_data;

void initalise()
{
    for(int i = 0; i < N; i++)
    {
        //random values btw 0 to 100
        A[i] = (float)(rand() % 100);
        B[i] = (float)(rand() % 100);
    }
}

void parallel_addition(void* arg)
{
    thread_data* data = (thread_data*)arg;

    for(int i = data->start; i < data->end; i++)
    {
        C[i] = A[i] + B[i];
    }
    pthread_exit(NULL);
}

int main()
{
    pthread_t threads[NUM_THREADS];
    thread_data th_data[NUM_THREADS];

    clock_t start, end;
    double time_taken;

    initalise();

    start = clock();

    //creating threads
    int chunk_size = N / NUM_THREADS;

    for (int i = 0; i < NUM_THREADS; i++)
```

```
    {
        th_data[i].start = i * chunk_size;
        th_data[i].end = (i+1) * chunk_size;
        pthread_create(&threads[i], NULL, parallel_addition, &th_data[i]);
    }


    //joining threads
    for (int i = 0; i < NUM_THREADS; i++)
    {
        pthread_join(threads[i], NULL);
    }

    end = clock();


    time_taken = ((double)(end - start)) / CLOCKS_PER_SEC;

    printf("time taken (parallel w/ %d threads): %.5f seconds\n",
NUM_THREADS, time_taken);

    return 0;
}
```

```
ibrahim-johar@ibrahim-johar-VirtualBox: ~/08
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$ gcc -o g1_b q1_b.c -pthread
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$ ./q1_b
time taken (parallel w/ 10 threads): 0.01913 seconds
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$
```

Task 2:

Write a multithreaded program that calculates various statistical values for a list of numbers.This program will pass a series of numbers on the command line and will then create three separate worker threads. One thread will determine the average of the numbers, the second will determine the maximum value, and the

third will determine the minimum value. For example, suppose your program is passed the integers. (The array of numbers must be passed as parameter to threads, and the thread must return the calculated value to main thread).

90 81 78 95 79 72 85

The main thread will print:

The average value is 82.
The minimum value is 72.
The maximum value is 95.

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>
#include <limits.h>

typedef struct {
   int *numbers;
   int count;
} thread_data;

void* calc_avg(void* arg) //thread func -> average
{
   thread_data* data = (thread_data*)arg;

   int sum = 0;

   for (int i = 0; i < data->count; i++)
   {
       sum += data->numbers[i];
   }

   double* avg = malloc(sizeof(double));
   *avg = (double)sum / data->count;
   return avg;


}

void* calc_min(void* arg) //thread func -> minimum
{
   thread_data* data = (thread_data*)arg;
   int* min = malloc(sizeof(int));
```

```
    *min = INT_MAX; //initial large val

    for (int i = 0; i < data->count; i++)
    {
        if (data->numbers[i] < *min)
        {
            *min = data->numbers[i];
        }
    }
    return min;
}

void* calc_max(void* arg) //thread func -> maximum
{
    thread_data* data = (thread_data*)arg;
    int* max = malloc(sizeof(int));
    *max = INT_MIN; //initial small val

    for (int i = 0; i < data->count; i++)
    {
        if (data->numbers[i] > *max)
        {
            *max = data->numbers[i];
        }
    }
    return max;
}

int main(int argc, char* argv[])
{
    if (argc < 2)
    {
        printf("Usage: %s <list of numbers>\n", argv[0]);
        return 1;
    }

    //converting cmd-line args to an int array
    int count = argc - 1;
    int nums[count];
```

```
    for (int i = 0; i < count; i++)
    {
        nums[i] = atoi(argv[i+1]);
    }

    //data for threading
    thread_data data;
    data.numbers = nums;
    data.count = count;

    pthread_t threads[3];

    //3 worker threads
    pthread_create(&threads[0], NULL, calc_avg, &data);
    pthread_create(&threads[1], NULL, calc_min, &data);
    pthread_create(&threads[2], NULL, calc_max, &data);


    //collect results
    double* avg;
    int* min;
    int* max;

    pthread_join(threads[0], (void**)&avg);
    pthread_join(threads[1], (void**)&min);
    pthread_join(threads[2], (void**)&max);


    printf("The average value is %.2f\n", *avg);
    printf("The minimum value is %d\n", *min);
    printf("The maximum value is %d\n", *max);

    //free allocated memory
    free(avg);
    free(min);
    free(max);

    return 0;
}
```

```
ibrahim-johar@ibrahim-johar-VirtualBox: ~/08
ibrahim-johar@ibrahim-johar-VirtualBox:~$ cd /home/ibrahim-johar/08/
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$ gcc -o q2 q2.c -pthread
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$ ./q2 90 81 78 95 79 72 85
The average value is 82.86
The minimum value is 72
The maximum value is 95
ibrahim-johar@ibrahim-johar-VirtualBox:~/08$
```