# Operating Systems Project (CLO3)

## Complex Engineering Problem Attribute Justification

| Attribute | Justification |
|---|---|
| WP1 Depth of knowledge | Deep knowledge of multi-threaded programming is needed |
| WP3 Depth of analysis | Detailed analysis of the performance bottlenecks in a multi-threaded application is needed |
| WP7 Interdependence | The performance of the program needs to be maximized. If a large number of threads are chosen, there would likely be a benefit in CPU-based computations. However, the overhead of having multiple threads and the memory access lately can degrade. Therefore, the tradeoff between threads and performance needs to be analyzed carefully |

### Objective:

The objective of this project is to familiarize yourself with file handling, multithreaded programming, code analysis and optimization. Your code will create a histogram of characters present in a large text file

### Instructions:

Your program will read two text files:
A large text file with ASCII character encoding
A file which defines the character bins. This file will define one bin per line. The characters in a single bin will be comma-separated.

#### *File Input:*

Your program will need four command line arguments:
arg1: Path of the large text file (file 1)
arg2: Path of the file which specifies the histogram bins (file 2)
arg3: Integer which specifies the number of threads
arg4: Integer which specifies the block size. This is the granularity of data interleaving across threads in the case of interleaved partitioning (more of this later).

#### *Bins File Format and Histogram Example:*

Assuming your bins file looks like this:

```
a,b,c,d
1,2,3,4
```

And your text file looks like this:

```
Aabbceeeffff ggg
113456
```

Then the histogram will state that there are

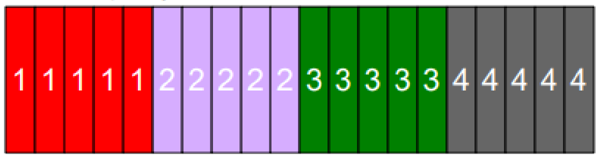
- Four characters in bin #1
  Aabbceeeffff ggg
  113456
- Four characters in bin #2
  Aabbceeeffff ggg
  113546
- Fifteen Characters which are not assigned to any bin (note that there is a line feed character)
  Aabbceeeffff ggg
  113546

You need to divide the text file data across the different threads. Each thread creates a small histogram of the input data it is assigned. Once all threads are done, they consolidate the data to give the histogram of the entire file. Note that data partitioning across threads can be done in two ways:

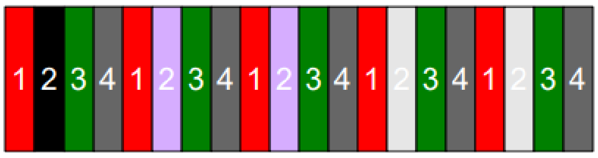
1. sectioned partitioning
2. interleaved partitioning.

In sectioned partitioning, adjacent threads do not access adjacent memory locations. Therefore DRAM performance suffers. In interleaved partitioning, adjacent threads access adjacent memory locations. Theoretically, interleaved partitioning should improve memory, and consequently, code performance. For your code, argument 4 is the size of each block in the case of interleaved partitioning.

Sectioned Partitioning ->



Interleaved Partitioning ->



## Submissions

You will need to submit two pieces of code. One which implements sectioned partitioning, and one which implements interleaved partitioning. The sectioned partitioning code should ignore argument 4. You need to make each code as performant as possible. For each code, you need to submit:

1. C file
2. Timing report. Segment your code into reasonable "operations" and profile the different operations to figure out which part of your code is taking the maximum amount of time. The timing analysis should be performed for multiple argument configurations: <number of threads> for sectioned partitioning, and <number of threads, block size> combination for interleaved partitioning. Following are the values you need to test for:
   a. <number of threads> in {1,2,4,8}
   b. <block size> in {1, 32, 256, 1024, 4096}
3. Explain and justify the timing differences you observe for the different configurations.
4. Create a graph which has the number of threads on the x-axis and the completion time on the y-axis. In the calculation of completion time, ignore the time needed for reading the large text file and storing this file in memory.
5. Report the steps you took to optimize the performance of your code