# Operating Systems–2: Spring 2024
## Programming Assignment 3: Dynamic Matrix Squaring
## Submission Deadline: 3rd March 2024, 9:00 pm

**Goal:-** This assignment aims to perform parallel matrix multiplication through a Dynamic mechanism in C++.

**Details:-** This assignment is an extension of the programming assignment 1. Consider a square matrix A. The goal of the problem is to find the square of matrix A in parallel using the dynamic mechanism (as discussed in class). Assume that the matrices are given in row-major order.

In this programming assignment, you will build upon your previous work measuring the performance of calculating the square of matrix A in parallel in C++. Previously, you have allocated the rows of the matrix to threads statically (in either mixed or chunk mode).

Here, in this assignment, the sequence of rows will be allocated dynamically to the threads. Each thread increments a shared counter *C* by a value *rowInc (row Increment)* to claim the set of rows of the C matrix it is responsible for. The value *rowInc* can be seen similarly to the chunk size of assignment 1, where the chunk size was fixed statically.

There will be synchronization issues when threads compete to increment the counter C. To solve this problem, you must implement different mutual exclusion algorithms to increment C as discussed in the class, which are: (a) **TAS**, (b) **CAS**, (c) **Bounded CAS,** and (d) **atomic increment** (provided by the C++ atomic library).

The main program will read **N, K, rowInc,** and the **matrix A** from an input file. Here, the variable N represents the number of rows of A; K represents the number of threads; rowInc is the row Increment explained above.

**Input File:-** As mentioned above, the input will consist of four parameters: **N, K, rowInc, and the matrix A**. Please name the input file as *inp.txt*

**Output File:-** Your program should output the following in an outfile file named *out.txt:*
*(i) The resulting square matrix produced by each mutual exclusion method:* (a) **TAS**, (b) **CAS**, (c) **Bounded CAS,** and (d) **atomic increment**
*(ii) Time taken to compute the square matrix using each mutual exclusion method.*

**Report Details:-** As part of this assignment, you must prepare a report describing your program's low-level design. Further, you must perform the following experiments described below, measure the performance, and provide the observations you recorded in the tabular format.

You will get 4 plots for the first three experiments, one for each mutual exclusion method. Please name the curves:-
a) TAS

b) CAS
c) Bounded CAS
d) Atomic

**1. Time vs. Size, N:** The y-axis will show the time taken to compute the square matrix in this graph. The x-axis will be the values of N (size on input matrix) varying from 16 to 2048 (size of the matrix will vary as 16*16, 32*32, 64*64, 128*128….) in the power of 2. *Please have the rowInc fixed at 16 and K at 16 for all these experiments*.

**2. Time vs. rowInc, row Increment:** Like the previous graph, the y-axis will show the time to compute the square matrix. The x-axis will be the rowInc varying from 1 to 32 (in powers of 2, *i.e.*, 1,2,4,8,16,32). *Please fix N at 2048 and K at 16 for all these experiments***.**

**3. Time vs. Number of threads, K:** Again, like the previous graphs, the y-axis will show the time taken to do the matrix squaring, and the x-axis will be the values of K, the number of threads varying from 2 to 32 (in powers of 2, *i.e.*, 2,4,8,16,32). *Please have N fixed at 2048 and rowInc at 16 for all these experiments.*

**4. Time vs. Algorithms:** Again, like the previous graphs, the y-axis will show the time taken to do the matrix squaring, and the x-axis will be different algorithms -
a) Static rowInc
b) Static mixed
c) Dynamic with TAS
d) Dynamic with CAS
e) Dynamic with Bounded CAS
f) Dynamic with Atomic.

*Please have N fixed at 2048, K fixed at 16, and rowInc at 16 for all these experiments*. Please plot a **bar graph** for this experiment.

*To handle temporary outliers, ensure that each point in the above plots is averaged over 5 times. Thus, you will run your experiment 5 times for each point on the x-axis.*

You must write the observation you gained based on the plots described above and mention any anomalies, if you observe in the report as well.

**Deliverables:-**

You will have to submit the following as a part of this assignment:

1. A report describing the following:
(a) Low-level design of your program. You must explain the implementation of the all the Mutual exclusion algorithm techniques.

(b) The graph plots are described above, as well as their analysis. Please name this report file as Assgn3_Report-<Roll No>.pdf

2. Prepare a README file that contains the instructions on how to execute your submitted file. The file should be named Assgn3_ReadMe-<Roll No>.txt

3. Name the source code file in the following format: Assgn3_Src-<Roll No>.cpp

4. Combine the source code, report, input file, and README as a zip archive file and name it Assgn3-<Roll No>.zip. Upload this zip file.

Please see the instructions given above before uploading your file. **Please follow this naming convention.** Your assignment will **NOT be evaluated** if there is any deviation from the instructions posted there, especially with regard to the naming convention.

**Marking Scheme:**

The evaluation scheme for this assignment will be as follows:

| S. No | Section | Marks |
|---|---|---|
| 1. | Program Design & Report | 50 |
| 2. | Program Execution | 40 |
| 3. | Code Indentation and Documentation | 10 |
| **Total** | | **100** |

**Please keep in mind that all the submissions are subject to plagiarism checks.**