CS 4345 (Operating Systems) [Summer 2020]

Assignment – 1

Due date: Sunday, 28 June 2020, 11:00 p.m.

Row-major dot product of two matrices:

Given two numerical matrices (which is row-column arrangements of numbers) A and B, where matrix A contains m rows and n columns and matrix B contains k rows and n columns, the ***row-major dot product*** of A and B is defined as a matrix P, where P contains m rows and k columns. Entries in matrix P for row i and column j (the cell (i, j)) is the **dot product** of ith row vector from A and jth row vector from B. For example, if A is a **3**×3 matrix $\begin{bmatrix}1 & 1 & 2\\ 3 & 4 & 5\\ 5 & 6 & 7\end{bmatrix}$ and B is a **4**×3 matrix $\begin{bmatrix}2 & 3 & -1\\ 5 & 3 & 2\\ 1 & 4 & 0\\ -1 & 2 & 3\end{bmatrix}$, then the row-major dot product of A and B is given by matrix P = $\begin{bmatrix}3 & 12 & 5 & 7\\ 13 & 37 & 15 & 20\\ 21 & 57 & 29 & 28\end{bmatrix}$, which is a 3×4 matrix. It is to be noted that the two matrices are row-major dot product compatible when they have same number of columns. Also note how the individual cell values of P are calculated. For example, the value 7 at cell (1, 4) in P is the dot product of row-vector 1 of A and row-vector 4 of B. That is, (1 1 2)•(-1 2 3) = 1*(-1) + 1*2 + 2*3 = (-1) + 2 + 6 = 7

Multi-threaded implementation of the above problem:

In this problem, you are required to write a multi-threaded program to find the row-major dot product of two matrices. The following provides a design guideline:

1. Calculate each cell of the product matrix P in a separate worker thread. This will involve determining the dimension of the product matrix. The number of worker threads is same as number of cells in the product matrix.
2. The main (or, parent) thread will initialize the matrices A and B and allocate space to hold the matrix P. Ideally, these matrices should be declared as global data so that each worker thread has access to these matrices.
3. The parent thread will create required number of worker threads. Each worker thread is responsible for calculating one cell value of the product matrix. For example, a worker thread responsible for cell (i, j) of the product matrix is passed the row i from first matrix and row j from the second matrix to calculate the dot product.
4. Once all worker threads have completed, the main thread will output the product matrix P. This requires the main thread to wait for all worker threads to finish before it can output the value of the matrix product. [*Hint: you may check use of join() or yield() to make the main thread waiting.*]

Program requirements:

The initial matrices A and B are in a file. Your program should read the data from the file. **The file name is**

provided as command-line argument at execution time. That is, the user may use the following command, after compilation, on a console to execute your program:

>> java <program-name> <datafile-name>

Caution: your program must not prompt (ask) the user to enter a filename, nor there should be any "hard-coding" of the input file path. The program is not allowed to ask the user to enter matrix values either. Matrices must be read from input file. Testing of your code will be done using a file with same format but different dimension and data.

Data file format:

The data file format is as follows: first line has 4 integer numbers, separated by single spaces, specifying dimension of two matrices. For example, 3 2 4 2 means matrix A has dimension 3 × 2 and matrix B has dimension 4 × 2. Your program should read this line at the beginning to determine whether two matrices are row-major dot product compatible and **indicate (display) the decision with appropriate message**.

Then there is a line gap followed by first matrix cell values (written in row-column arrangement). Each column is separated by two spaces and each row ends with a line break. After first matrix data, there is a line gap followed by second matrix data. Check the sample data file **Matrices.txt**

The program then reads the matrices, completes calculation as specified above, and produces the output on console in a matrix format (standard row-column format). You can assume that the data file will have integer value matrices only. However, the given data file is just a sample. Instructor may test your program using another, but similar, data file where the matrices have larger dimensions.

[MUST] Multi-threading requirements**:

Each worker thread displays the beginning and ending of their work together with the calculated value. For example, the worker thread calculating cell (i, j) of the product matrix should display the following: "Thread <i, j> starts calculation" (to indicate beginning of thread run) and "Thread <i, j> returns <cell-value>" to indicate end of that thread's run. [*The values of i, j, cell-values should reflect appropriate values.*]

[*Hint: these print statements should be within the run() method of the worker thread*]

All exceptions must be caught properly (use of proper try-catch blocks).

Submission: Each source code file should contain the following as coding comments at the top: course number (CS4345), Semester/Year (Summer 2020), assignment identifier (Assignment 1) and your name as author of the program. You may use regular Java comments or Javadoc com. Submit your source file(s) through BlazeVIEW submission box.