Assignment 03: Building an parallel matrix multiplier

In this assignment you will be tasked with building a parallel matrix multiplier. For this assignment we will implement the simpler but less efficient stripe multiplier. there is a more efficient block multiplier but it is much more complex to implement. The parallel stripe multiplier will multiply matrix A by matrix B. To perform this task matrix A is divided into horizontal stripes and a stripe is handed to each node. Each node recieves a full copy of matrix B. Each node will perform matrix multiplication on it's stripe of A and the full matrix B. When these results have been calculated all nodes will take part in a gather operation to send all the data back to the coordinator node wherein it will be reassembled into a full matrix.

For the simplification of this assignment you are restricted to square matricies (NxN where N >= 2) and N must be a multiple of 8. I will be testing this with 4 MPI nodes and 8 MPI nodes and small and large matricies to see if your code is capable of handling any reasonable size.

I will provide a simple C++ program that you can use to build matrix files as one dimensional arrays. I highly recommend using a one dimensional array to represent your matricies as two dimensional would introduce complexities when trying to send and recieve data from nodes.

Tasks and marking scheme:

01) create a main method that will initialise and finalise MPI, and will determine the world rank and world size. if the rank is zero the process should be the coordinator otherwise a participant (5%)

02) create a coordinator(int world_rank, int world_size, int argc, char** argv) function that does the following tasks (25%)

- determines if the correct number of arguments have been passed on the command line (should be 4, program name, two matrix filenames, and a size for N) it should test if both files exist and read them. if there is a error at any stage broadcast a message of 0 to indicate failure to the other nodes. if all of the above succeeds then broadcast a message of 1 to indicate success.
- determine the size of a full matrix, stripe and N and send these values to all nodes.
- take part in a scatter operation to send a stripe of matrix A to each node and use a broadcast to send a full copy of matrix B to each node.
- perform the matrix multiplication on the stripe and take part in a gather operation to get all calculated stripes back to the coordinator.
- print out the calculated matrix to console

03) create a particpant(int world_rank, int world_size) function that does the following tasks (15%)

- wait for a broadcast from the coordinator. if we get a failure value then exit
- get the size of a stripe, matrix, and N from the coordinator.
- take part in a scatter operation to get a stripe of matrix A and use a broadcast to

recieve the matrix B.

- multiple the stripe with the matrix and take part in a gather operation to send the calculated stripes back to the coordinator.

04) create a multiplyStripe(double *stripeA, double *matrixB, double *stripeC, int total_size, int stripe_size, int matrix_size) function to take in arrays representing the stripe of A and the matrix B. it should multiply these together to produce a stripe of C. Use the dotProduct function to perform multiplication of each element. (10%)

05) create a printMatrix(double *matrix, int n, int total_size) function to print out a matrix to console. it will print out total_size elements arranged in rows of n elements. (5%)

06) create a dotProduct(double *matA, double *matB, int row, int column, int n) function to perform the dot product of a row in matA with a column in matB. this should return a double of the result (10%)

07) modify your code to be able to multiply three matricies together or two matricies together. when multiplying three matricies together A by B by C you must multiply A and B first before you multiply the result by C. matrix multiplication is not commutative. (30%)

Notes:

You have three weeks to submit this assignment. Submission date is 25th April at 23:55. you must submit via moodle. assignments that are submitted from 23:56 onwards will have the standard late penalties applied. You are required to submit a single CPP file that contains the entire code of your program

There are a range of penalties in addition to the standard late submission penalties that I can apply to assignment. This is to encourage you to submit your project in a form that makes it easy to run and correct.

- failure of code to compile -30% Using a standard makefile or compiler I should only have to link your code against the MPI libraries and it should run without modification
- the use of external libraries to MPI -20%
- filename name should be firstname_lastname_studentnumber where firstname, lastname and studentnumber are your own details.
- archives should only be in one of the following formats zip/rar/tar.bz2/tar.gz/7z -10%