CS426 Parallel Computing
Project 3 – OpenMP

Due Wednesday, July 25th 2021 at 23:59

In this project, you will use OpenMP to implement the matrix's rank using gaussian elimination.

Problem Statement

Rank of a matrix is the number of linearly independent vectors of that matrix. To calculate the rank of a matrix you will use gaussian elimination. You can find the details about gaussian elimination in the following link.

https://en.wikipedia.org/wiki/Gaussian_elimination

Here is an example:

Consider the following matrix:

[[1,2,3,4],[2,3,4,5],[2,4,6,8],[3,1,2,4],[4,1,2,3],[2,4,6,8]]

Using gaussian elimination one can find its rank as 4. To implement the algorithm You need to start from the first row and check if it is dependent on the other rows. If it is, that row will be eliminated and rank will decrease by one. Following algorithm shows how to check the dependency of rows i,j:

result = matrix[i] * matrix[j][0] -matrix[j] * matrix[i][0]

if (All elements in the result vector are zero)
row i and j are dependent
else
row i,j are independent

Consider the first row as the base and let us check its dependency with second and third rows:

result = [1,2,3,4] * 2 - [2,3,4,5] * 1 = [0,1,2,3]
first and second rows are independent

result = [1,2,3,4] * 2 - [2,4,6,8] * 1 = [0,0,0,0]
first and third rows are dependent

After checking the dependency of the first row with all other rows we will move on to the second row and we will repeat the same procedure. We will not take the first row into account anymore. Plus the dependent row will be eliminated. Now we get into the parallel part. If the matrix's dimension is (n,n), for checking the dependency of two rows we have to check all the elements of the rows. Write your code which can do this part in parallel. The output of your program will be the rank of the matrix printed on the screen.

Language

Use C language with OpenMP for coding the program. (You can refer to http://openmp.org for openMP's specifications.) You may use the following command to compile your files:

gcc –fopenmp filename.c

Testing

- Generate the input matrix with random integer numbers. Use (n,n) as the dimension of your matrix where n is set as 100,1000,10000,100000. You need to give the dimension as the input argument of your program.
- Write a serial version of the program and compare the timing results with the parallel version.
- You can use the function **omp_get_wtime()** to produce timing results. This function is used in a similar fashion as **get_walltime**(). Choose the appropriate data type to store the value returned by **omp_get_wtime**(). Generating the input matrix and etc. should not be included in the time measurement.

Report

Write a short report containing:

1. Explain the used parallelization strategy
2. Draw a graph which shows the timing results of serial and parallel versions.
3. Short discussion about the results.
4. Answer to the following question.

   Are there any other parts in the algorithm that can be implemented in parallel? If so, how would you use OpenMP to implement your design?

Submission

Put all relevant code, makefile, shell script and your report into a zip file.

Name the zip file: name_surename_ID.zip

Submit the zip file to the correct assignment on Moodle.