
CS 213 — Multiprocessor Architecture and Programming

Instructor: Elaheh Sadredini
Programming Assignment 2 - OpenMP

Released: November 15

Due Date: November 25 - 11:59pm

1 Sparse Matrix Vector Multiplication (SpMV) with OpenMP

The purpose of this part is to improve the performance of sparse matrix-vector multiplication using OpenMP.

You need to implement the operation $x_{i+1} = Ax_i$, where A is your sparse matrix and x is the vector (store result of the product of A and x into x at the end of i^{th} iteration and use it in the $i + 1^{th}$ iteration).

You are asked to write two programs for each of the following sparse matrix formats:

- Compressed sparse row (CSR, CRS or Yale format)
- Compressed sparse column (CSC or CCS)

If you are not familiar with these formats, please see [Wikipedia](#) or find other online resources.

Each program should have a sequential version and an OpenMP parallel version of the algorithm. The matrix A should be read as input and the vector x should be initialized to all 1s. Three sample matrices are provided with this assignment. You can assume that the sparse matrix is a square matrix.

Try different number of threads (2 to 32) to record the time required to perform the multiplication. Prepare a table with your results and write a short report (no more than a page) discussing your results. Include the speedup in each of the two cases. You should try to optimize your parallel program.

1.1 Input

The command to execute your program should be:

```
./spmv matrixFileName m n
```

where "matrixFileName" is the matrix file name (i.e., matrix1.txt), m is the number of iterations, and n is the number of threads.

Note: you code will be tested with this exact command. If your code does not run with this command, you do not get the implementation credit.

Input matrix format: The first line of the file contains three numbers: first dimension of the matrix, second dimension of the matrix, and total number of non-zero entries. The rest of the file is one line per non-zero in the format: **i j A(i,j)**: that is, row, column, and value.

1.2 What to submit

- (a) Sequential and parallel implementation of CSR and CSC format. Files should be named as "CSRSeq.c", "CSROpenMP.c", "CSCSeq.c", and "CSCOpenMP.c".
- (b) Calculate the output vector for three input matrices after 8 iterations ($i=0$ to $i=7$). Report the output of the resulting vector, **one value per line**. The files should be named as "CSRVec1.txt", "CSRVec2.txt", "CSRVec3.txt", and "CSCVec1.txt", "CSCVec2.txt", "CSCVec3.txt" (the numbers correspond to input matrix name).
- (c) Try different number of threads (2 to 32) for each implementation and record the time required to perform the multiplication. Prepare a table with your results and write a short report (no more than a page) discussing your results. Include the speedup in each of the two cases.

1.3 Grading

Table 1

| Task | Percentage |
|---|------------|
| Implementing Sequential CSR | 5% |
| Implementing Sequential CSC | 5% |
| CSR optimoization with OpenMP | 15% |
| CSC optimoization with OpenMP | 15% |
| Correct output vectors for CSR (for 3 given matrices) | 15% |
| Correct output vectors for CSC (for 3 given matrices) | 15% |
| Your report and discussion | 30% |