Homework 1: Optimizing Sparse Matrix Vector Multiplication

Sparse matrix vector multiplication (SPMV) is a very important computational kernel widely used in many scientific applications. Optimizing the performance of SPMV is a hot research topic in the HPC community. In this homework, you need to implement your own version of SPMV and optimize its performance.

What is a sparse matrix?

Read the WIKIPEDIA entry on sparse matrix: `http://en.wikipedia.org/wiki/Sparse_matrix`

What is sparse matrix vector multiplication?

Matrix vector multiplication of the form $y = Ax$ is SPMV if the input matrix A is sparse and the input vector x and the output vector y are dense.

Which sparse matrix format do we use?

We use the matrix market exchange format: `http://people.sc.fsu.edu/~jburkardt/data/mm/mm.html`

What are the inputs?

The course webpage (`http://inside.mines.edu/~bwu/CSCI_580_22FALL/homework.shtml`) provides 3 inputs, which will be used to test your code. The first input contains two files, but you only need to use the one named NLR.mtx.

What your code should do?

For each input:

- Step 1. Load the sparse matrix.
- Step 2. Randomly generate values (each being a double between 0.1 and 4.9) for the sparse matrix and the dense vector.

- Step 3. Perform parallel SPMV and produce the result vector. (Measure the execution time of this part.)
- Step 4. Test the correctness by comparing your result with the one produced by the reference implementation.

Note: your code should be written in C or C++. You should design a data structure to store the sparse matrix, which is critical for performance. You must implement a parallel program, which may use pthreads (https://computing.llnl.gov/tutorials/pthreads/), openmp (https://computing.llnl.gov/tutorials/openMP/) or Cilk (https://www.cilkplus.org/) for parallelization. You should find a reference implementation through Google search (e.g., the boost C++ library implementation) and integrate it into your C/C++ code to test correctness.

What to turn in?

- A tarball containing all your source code. After extracting your files, I should see a folder named as studentFirstName_studentLastName. You should provide a makefile[1], so that I can type "make" to compile your code. **IMPORTANT:** I will put the inputs into your folder. If I type "make run", your code should AUTOMATICALLY process each input (including producing the result and testing the correctness) and AT THE END print the total execution time of "step 3" for all inputs.
- A pdf document generated by LaTeX describing what kind of optimization(s) you applied.

Grading rules:

- You get 0 point if your code doesn't compile with GCC 5.0 or newer.
- You get 0 point if your result is incorrect.
- You get 5 points if your result is correct, but you don't apply any optimization.
- You get 10 points if your result is correct and you apply at least one optimization technique.

[1] https://makefiletutorial.com/