EEL4837 Programming for Electrical Engineers II – Spring 2023

Excursion 1: Circuit Analysis Tool

Due Date: 11:59 PM, Sunday, March 19, 2023

Submission Instructions

Submission should be through Gradescope. You need to submit a single zip file that contains:

1. One or several source code files of your circuit analysis tool. Your program should write all the outputs into a file named ***output.txt*** located in the same folder as your program. The output format should be a single row similar to what is shown in the slides:
    - $e_1\ e_2 \ldots e_n\ v_1\ v_2 \ldots v_n\ i_1\ i_2 \ldots i_n$
    - That is, you write the potentials, voltages, and currents with *one space between them.*
    - Each potential/voltage/current is a floating-point number in a decimal notation with *3 decimal places* after the point (e.g., 14.654).
2. A Bash script named ***compile.sh*** that compiles your program into a single executable file named ***exc1*** and located in the same folder as your code. Some steps in your Bash script for your reference only:
    a. The simplest compile script example (if one source code file only, feel free to use it):
        - `g++ -std=c++11 your_source_code_file_name.cpp -o exc1`
    b. With multiple code files, you can compile into object files first and link them together:
        - ```
          g++ -c file_1.cpp -o file_1.o
          g++ -c file_2.cpp -o file_2.o
          g++ file_1.o file_2.o -o exc1
          ```
    c. (Optional) On the ECE server, when creating the compile.sh file from scratch, you may need to change the permission to allow your script to execute:
        - `chmod +x compile.sh`
    d. Usage of compile script in the directory with your source files:
        - `./compile.sh`
    e. Execute the compiled executable file; it must read the file ***netlist.txt*** in the same directory as its input:
        - `./exc1`
3. A ***Readme*** file that explains how to compile, execute and run your code.
4. (Optional) Additional documentation that you wish to provide.
          ```

Description

The goal is to implement an elementary circuit analysis tool that reads a circuit from a netlist and computes the current, voltage potential, and voltage drop of each component.

In this excursion, you can assume that:

1. The format of the input netlist is the same as that in the Excursion 1 slides.
2. The circuit only consists of voltage sources and resistors.
3. Your program will not be tested on mal-formed or inconsistent netlists.

Guidelines

Taking the Excursion 1 slides as a reference, you may create several functions for reading a new netlist, counting the number of branches and nodes, creating several matrices, calculating all the parameters, etc. Here are some general steps for your reference:

1. Read a new netlist: one of the easiest ways is to use filestream. Note that we will strictly use the format of “branch label | source node label | destination node label | value” as shown in class.
2. Count the number of branches and nodes: you need to take the input from the previous step.
3. Create an incidence matrix: you may use the method introduced in class to derive a 2-D incidence matrix from a netlist.
4. Create a voltage coefficients matrix.
5. Create a current coefficients matrix.
6. Create a combined matrix.
7. Append input to the combined matrix.
8. Calculate all the parameters (e.g., using Gaussian elimination, LU factorization, or matrix inversion).

Grading

Total points: 100

1. Automated grading (75 pts)
    a. Correctness (60 pts): correct output values on well-formed, “legal” inputs.
    b. Robustness (15 pts): correct output on very simple and very complex “legal” inputs.
2. Manual grading (25 pts)
    a. Efficiency: the code is reasonable and not brute-force. No particular time/space complexity is expected.
    b. Elegance: clean code style, readable comments as needed, meaningful functions, etc.
3. Bonus points (20 pts): create and use a data structure for sparse matrices (as shown in the slides).

Note

1. Your code should be in C++ only. We encourage splitting up your code into multiple files/ functions/classes.
2. Your program should compile and run successfully on both the ECE Linux server and Gradescope.
3. You can only use C++ standard libraries (including STL). Do **not** use libraries that implement matrices for you. If you are not sure whether or not some libraries are allowed, please reach out to the TA or instructor for help.