Assignment 2 Spice Simulation

Deepak Charan S ee23b022

September 2024

1 Introduction

I am Deepak Charan S and this is my report for the second assignment of EE2703 course (Applied Programming Lab).

2 Description of my solution

2.1 Overall Approach:

My train of thought was to visualise the KCL and KVL equations forming out of the circuit and accordingly fill it in my matrix.
In our EE2015 course (Electrical circuits and Networks), we had learnt the matrix construction and the approach was that:

- In KCL equations, the diagonal elements = sum of conductances connected to that node
- Non diagonal elements (take row i, column j where i != j) = overall conductance between node i and j
- current sources directly goes into the solution matrix (taking care of signs)
- if any voltage source connected to node, assume a current 'Is' flowing out of it and accordingly plug it in the matrix
- normal KVL equations for independent voltage sources

2.2 Parsing:

1. Using the file pointer and readlines(), I broke down the file into multiple lines and used the line.strip() to remove newline characters.
2. Then I checked at what lines are '.circuit' and '.end' present (which would be my area of focus)
3. Then in each line, I checked for the presence of comments and removed them

4. Using the first letter of the first character, I identified the components and accordingly stored them in different dictionaries/lists

2.3 Construction and solving of the matrix:

- While parsing through the lines, I counter the number of distinct nodes (apart from GND) to figure out how many equations I would need (AKA size of my matrix)
- I then used a numpy array with the corresponding dimensions to construct two matrices
- After construction of the matrix, I filled in the entries using the algorithm mentioned in my 'Overall Approach' section
- Then I used numpy.linalg.solve() to solve the equations and accordingly separated the solution matrix into 2 dictionaries (1 for voltage at each node and 1 for current through the voltage sources)

2.4 Error Handling:

* Invalid files and malformed files (no '.start' or '.end') are checked
* If any component which isnt a voltage/current source or a resistor is present, I raise 'ValueError'
* when there is an invalid arrangement of current/ voltage sources, the matrix would produce a zero row. Hence, I check for the prescence of that and raised the desired error.
* Any free node or current sources connected to a floating node would cause a singular matrix, hence I raised a ValueError for that
* If there isnt a ground, no voltage can be found out (we need a referance). Hence, I raised an error for that

3 Extra Test cases:

In order to test the robustness of my code, I have tried it against multiple test cases (which I made got it online/made it myself/got it from friends) which are as follows:

- *parallel.ckt*: To test against a parallel resistor network
- *mul_dc.ckt*: To test against multiple Voltage sources
- *supernode.ckt*: To test against voltage sources placed between two nodes which arent 'GND' (this helped me realise a logical error in my earlier version)

- *free_node.ckt*: If there is a component which isnt connected to anything, the matrix will have no solution. To test it and raise the appropriate error, I used this file
- *floating_node.ckt*: If one end of the resistor isnt connected to anything, no current would flow through it so I used this file to check if voltage is same across both the ends (current source in place of a resistor would raise an error and I check for it as well)

4 References

- https://www.geeksforgeeks.org/valueerror-the-truth-value-of-an-array-with-more-than-one-element-is-ambiguous-use-a-any-or-a-all/: To check for a zero row in my matrix
- https://www.allaboutcircuits.com/textbook/reference/chpt-7/example-circuits-and-netlists/: To get some sample test cases from online (mul_dc.ckt)
- I had also asked my roommate, N Deenabandhan (EE23B021) for suggestions regarding to the construction of the matrix and our friend group also shared test cases with each other to verify our codes.