Cecilia Espadas

ECE 2774: Advanced Power Systems Analysis

Project 1: Simple Circuit Simulator

Project Overview

Purpose

The purpose of this simulator is to help model and solve basic DC circuits. Its purpose is to make it easy for users to understand and analyze the behavior of the circuit. It shows how voltage sources, resistors, and loads interact. It allows you to construct a DC circuit and provides the circuit current and the voltage at each bus.

Key Features and Functionality

The Simple Circuit Simulator is a modular circuit simulator with classes for buses, voltage source, resistor, and load. The circuit class combines these allowing the user to build and analyze different DC circuits. Using the solution class calculates the current flowing through the circuit and the voltages at each bus.

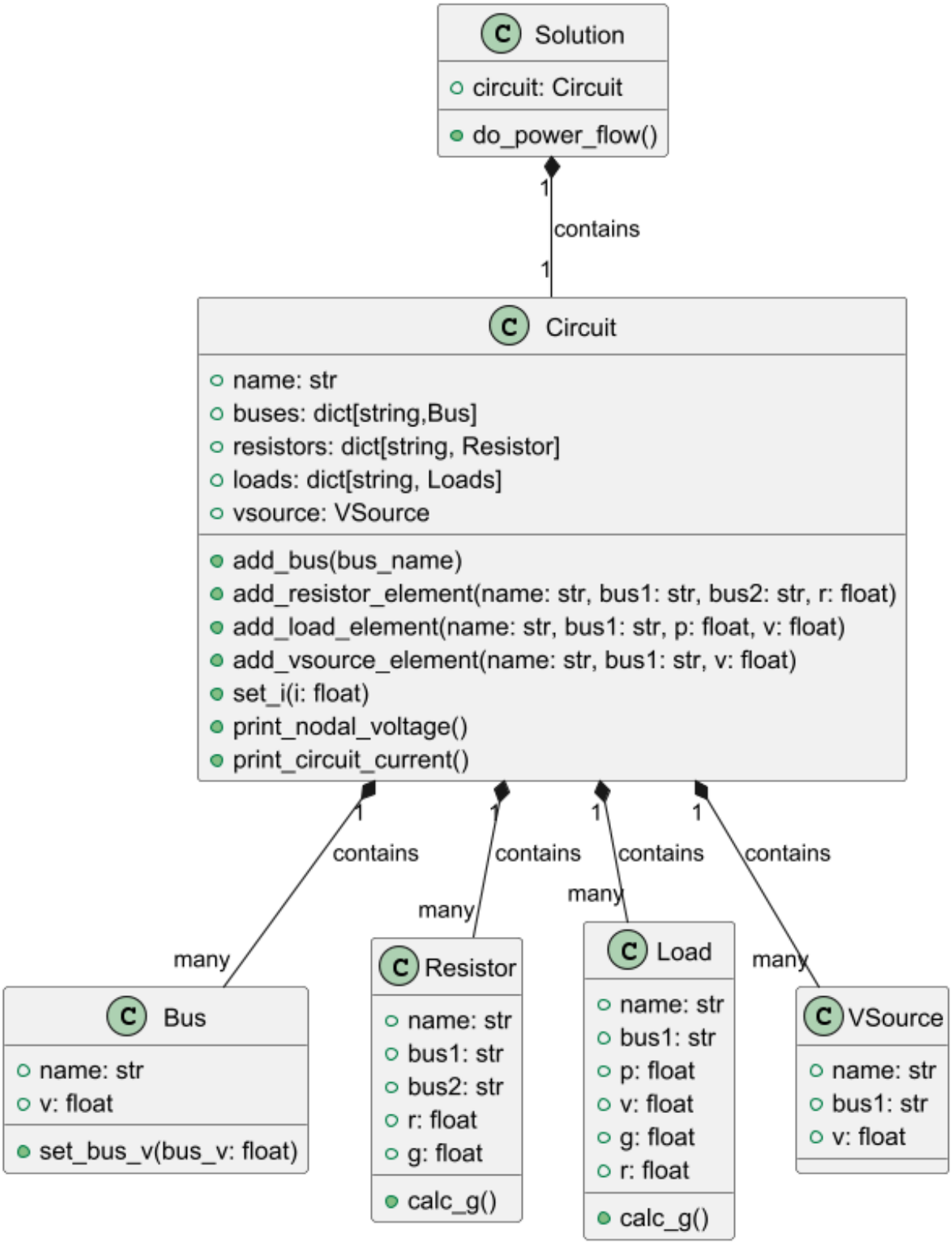
Problem

This simulator solves the problem of solving and analyzing circuit behavior. It automates the process of solving the current and bus voltages for the user. They can input their buses, voltage source, resistors, and the load and the simulator will analyze it for them.

Real-World Applications

The simulator could be used to assist in various power flow analyses of DC circuits. It could be applied to renewable energy DC microgrids and power networks.

Class Diagram



Class Specifications

Bus Class			
Purpose	**Method**		**Explanation**
This initializes the Bus class.	__init__(self, name: str, v: float = 0.0):		This method initializes a bus object with a user-defined name and an initial voltage.
	Attributes:	self.name = name	The name of the bus, provided by the user during object creation.
		self.v = v	The voltage at the bus, initialized to 0.0 by default. For voltage source-connected buses, the voltage is updated when the source is assigned. For other buses, it updates during the power flow calculation.
This method updates the voltage at a bus.	set_bus_v		This method sets the voltage at the bus. This method is used during the power flow analysis in the solution class when bus voltages are calculated.
This method provides a textual representation of the bus object.	__repr__		This method returns a string representation of the bus, displaying its name and voltage. It is useful for debugging and printing bus information in a readable format.

Resistor Class			
Purpose	**Method**		**Explanation**
This initializes the Resistor class.	__init__(self, name: str, v: float = 0.0):		This method initializes a resistor object with a user-defined name, two buses, and a resistance value. The conductance is calculated internally.
	Attributes:	self.name = name	The name of the resistor, provided by the user during object creation.
		self.bus1 = bus1	The first bus connected to the resistor, provided by the user.
		self.bus2 = bus2	The second bus connected to the resistor, provided by the user.
		self.r = r	The resistance value of the resistor in ohms, provided by the user.

		self.g = self.calc_g()	The conductance value, calculated internally using the calc_g method.
This method calculates the conductance of the resistor.	calc_g(self)		This method computes the conductance, G, of the resistor using the formula G= 1/R, where R is the resistance.
This method provides a textual representation of the resistor object.	__str__(self)		This method returns a formatted string representation of the resistor, displaying its name, buses, resistance, and conductance.

Load Class			
Purpose	**Method**		**Explanation**
This initializes the Load class.	__init__(self, name: str, bus1: str, p: float, v: float):		This method initializes a load object with a user-defined name, bus connection, real power, and voltage. The conductance and resistance are calculated internally.
	Attributes:	self.name = name	The name of the load, provided by the user during object creation.
		self.bus1 = bus1	The bus to which the load is connected, provided by the user.
		self.p = p	The real power of the load in watts, provided by the user.
		self.v = v	The voltage across the load in volts, provided by the user.
		self.g = self.calc_g()	The conductance value, calculated internally using the calc_g method.
		self.r = self.calc_r()	The resistance value, calculated internally using the calc_r method.
This method calculates the conductance of the load.	calc_g(self)		This method computes the conductance g of the load using the formula $G = P / V^2$, where P is the real power and V is the voltage.
This method calculates the resistance of the load.	calc_r(self)		This method computes the resistance r of the load using the formula $R = V^2 / P$, where V is the voltage and P is the real power.

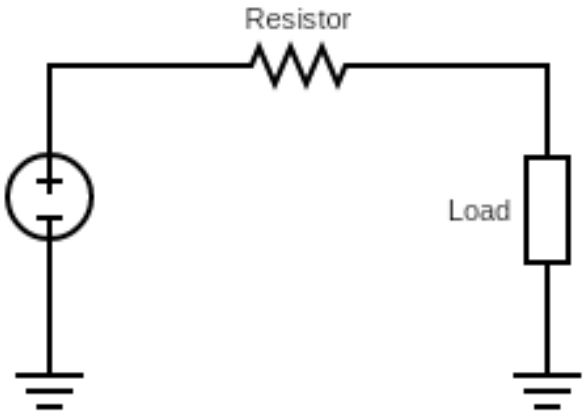
This method provides a textual representation of the load object.	__str__(self)	This method returns a formatted string representation of the load, displaying its name, bus, real power, voltage, resistance, and conductance.

Circuit Class			
Purpose	**Method**		**Explanation**
Initializes a new Circuit object.	__init__(self, name: str)		Sets up an empty circuit with dictionaries for buses, resistors, and loads. Initializes voltage source as None and current as 0.0.
	Attributes:	self.name = name	The name of the circuit, provided by the user when defining the object
		self.buses: Dict[str, Bus] = {}	A dictionary of buses, where the key is the bus name (str) and the value is a Bus object.
		self.resistors: Dict[str, Resistor] = {}	A dictionary of resistors, where the key is the resistor name (str) and the value is a Resistor object.
		self.loads: Dict[str, Load] = {}	A dictionary of loads, where the key is the load name (str) and the value is a Load object.
		self.vsource: VSource = None	A VSource object representing the voltage source of the circuit. Initially set to None.
		self.i: float = 0.0	The total current in the circuit, initialized to 0.0.
Adds a new bus.	add_bus(self, bus_name: str)		Checks if the bus exists before adding it to self.buses.
Adds a new resistor.	add_resistor_element(self, name: str, bus1: str, bus2: str, r: float)		Ensures both buses exist before adding the resistor. Stores it in self.resistors.
Adds a new load.	add_load_element(self, name: str, bus1: str, p: float, v: float)		Ensure the bus exists before adding the load. Stores it in self.loads.
Adds a voltage source.	add_vsource_element(self, name: str, bus1: str, v: float)		Ensures the bus exists before assigning a single voltage source to self.vsource.

Sets circuit current.	set_i(self, i: float)	Ensures a voltage source exists before updating the current.
Prints nodal voltages.	print_nodal_voltage(self)	Iterates through self.buses and prints each bus's voltage.
Prints circuit current.	print_circuit_current(self)	Displays the circuit's current value.
Returns a string representation of the circuit.	__repr__(self)	Displays key attributes for debugging and visualization.

Solution Class			
Purpose	**Method**		**Explanation**
This initializes the Solution class.	__init__(self, circuit: Circuit):		Initializes a Solution object, ensuring that a valid Circuit object is provided. If not, raises a TypeError.
	Attributes:	self.circuit = circuit	Holds the Circuit object passed to the Solution instance for later use in power flow calculations.
This method performs the power flow calculation.	do_power_flow(self)		Solve the circuit by calculating bus voltages and circuit current using the voltage source, resistors, and loads.
	Within do_power_flow(self)	Calculate G and R	Loops through the circuit's resistors and loads to compute the total conductance (1/R).
		Calculate I	Uses Ohm's Law (I = V * G) to compute the current, where V is the voltage source and G is the total conductance.
		Set V	Sets the voltage of the source bus equal to the source voltage and calculates the voltage at each bus using the voltage drops across the resistors

Equations



Conductance

$$G = \frac{1}{R}$$

- G = Conductance (S)
- R = Resistance (Ω)

$$G = \frac{P}{V^2}$$

- G = Conductance (S)
- P = Power (W)
- V = Voltage (V)

Ohm's Law - for current calculation

$$I = \frac{V}{R} = VG$$

- I = Current (A)
- V = Voltage (V)
- G= Conductance (S)

Total Resistance Calculation – for sum of all resistances

$$R_{total} = \sum R_{resistor} + \sum R_{load}$$

- $R_{resistor}$ = resistance of each resistor (Ω)
- R_{load} = resistance of each load (Ω)

Ohm's Law – for voltage drop

$$V_{drop} = IR_{resistor}$$

- V_{drop}= voltage drop across the resistor (V)
- I = current (A)
- $R_{resistor}$ = resistance of resistor (Ω)

Kirchoff's Voltage Law (KVL)

$$\sum V_{drop} = \sum V_{source}$$

- V_{drop} = voltage drop across each circuit element (V)

Example Case

Problem Definition

In this example case, a simple DC circuit is modeled and analyzed using the Simple Circuit Simulator. The circuit consists of a DC voltage source, a resistor, and a load. These elements are connected in series, with a bus between each connection. The example circuit has the following parameters:

- Bus A
- Bus B
- Voltage Source: 200 V source connected to Bus A
- Resistor: 50 Ω resistor connected between Bus A and Bus B
- Load: load with a power of 10,000 W and a nominal voltage of 500 V connected to Bus B

Use the main code below for this:

```
from circuit import Circuit

from solution import Solution

# create the Circuit object

circuit = Circuit("MyCircuit")

# Add components to the circuit

circuit.add_bus("BusA")

circuit.add_bus("BusB")

circuit.add_vsource_element("Va", "BusA", 200)

circuit.add_resistor_element("R1", "BusA", "BusB", 50)

circuit.add_load_element("Lb", "BusB", 10000, 500)

# Create the solution object

solution = Solution(circuit)

# Perform the power flow calculation

solution.do_power_flow()

# Print the results

solution.circuit.print_nodal_voltage()

solution.circuit.print_circuit_current()
```

Purpose	Method	Explanation
These methods build the circuit using user inputs.	circuit.add_bus	This method adds a bus to the circuit. All buses must be added first so that the resistors, load, and voltage source can be assigned to their buses. This method only requires that the user inputs the name of the bus. The buses are placed into a dictionary.
	circuit.add_vsource_element	This method adds the voltage source to the circuit. The method requires that the user inputs the name of the voltage source, the bus that it is connected to, and its voltage in volts. The voltage source is placed into a dictionary.
	circuit.add_resistor_element	This method adds a resistor to the circuit. The method requires that the user inputs the name of the resistor, the first bus the resistor is connected to and then the second bus the resistor is connected to, and the resistance in ohms. The resistor is placed into a dictionary.
	circuit.add_load_element	This method adds the load to the circuit. The method requires that the user inputs the name of the load, the bus the load is connected to, the power of the load in watts, and the nominal voltage in volts. The load is placed into the dictionary.
This method conducts the analysis of the simple circuit.	solution.do_power_flow()	This method solves the circuit by finding the bus voltages and circuit current. It calculates the total conductance in the circuit from the resistors and loads. It then uses this conductance to calculate the current via Ohm's Law, V=IG. It then determines the voltage at each bus. For the bus connected to the voltage source, the voltage is set equal to that of the source. For the bus connected to the load, the voltage is determined using Kirchoff's voltage law. The voltage drop across the resistor is found using the circuit current and the resistance of the resistor. The voltage at the bus connected to the load is found from

		subtracting the voltage drop from the voltage source.
These methods print the essential circuit information from the analysis.	solution.circuit.print_nodal_voltage()	This method prints the nodal voltage for the circuit, the voltages at each bus. It presents the voltage at each bus, according to the names utilized by the user, in volts.
	solution.circuit.print_circuit_current()	This method prints the circuit current in amps.

Solution Process

Current Through the Circuit:

1. The total resistance of the circuit is calculated by adding the resistance of the resistor and load. The total conductance is the reciprocal of the total resistance:

$$G = \frac{1}{R_{total}}$$

2. The total current through the circuit is calculated using the formula:

$$I = V_{source} G_{total}$$

   Where G_{total} is the total conductance and V_{source} is the voltage from the source.

Voltage at Each Bus:

1. The voltage at Bus A is set to the value of the voltage source.
2. The voltage drop across the resistor is calculated using Ohm's Law:

$$V_{drop,resistor} = IR_{resistor}$$

   Where I is the calculated current and $R_{resistor}$ is the resistance of the resistor in the circuit.

3. The voltage at Bus B is set by subtracting the voltage drop from the Bus A voltage,

$$V_{Bus\ B} = V_{Bus\ A} - V_{drop}$$

Expected Output

- Voltage at each bus:
  - Bus A: 200 V
  - Bus B: 66.67 V
- Total Current: 2.67 A