Project Overview

This software was created to simulate simple DC circuits. Given a voltage source, a resistor, and a resistive load element, the simulator can calculate the current going through the circuit and use it to find the voltage at the load bus.

Class Overview

Bus

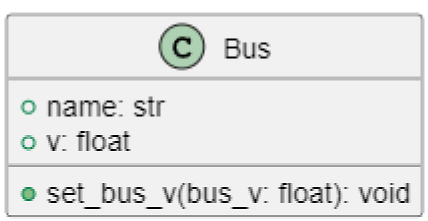

Figure 1: Bus Class

Attributes:

i. name (str): The user should provide this as an argument when defining the object.

ii. index (int): Each created bus automatically gets assigned an index value.

iii. v (float): Represents the voltage at the bus. For buses connected to a voltage source, the voltage updates when the source is created. For all other buses, the voltage updates during the power flow calculation.

Methods:

i. set_bus_v(bus_v: float): Sets the voltage at the bus.

Component

The Component class is a generic abstract class that contains electrical components.

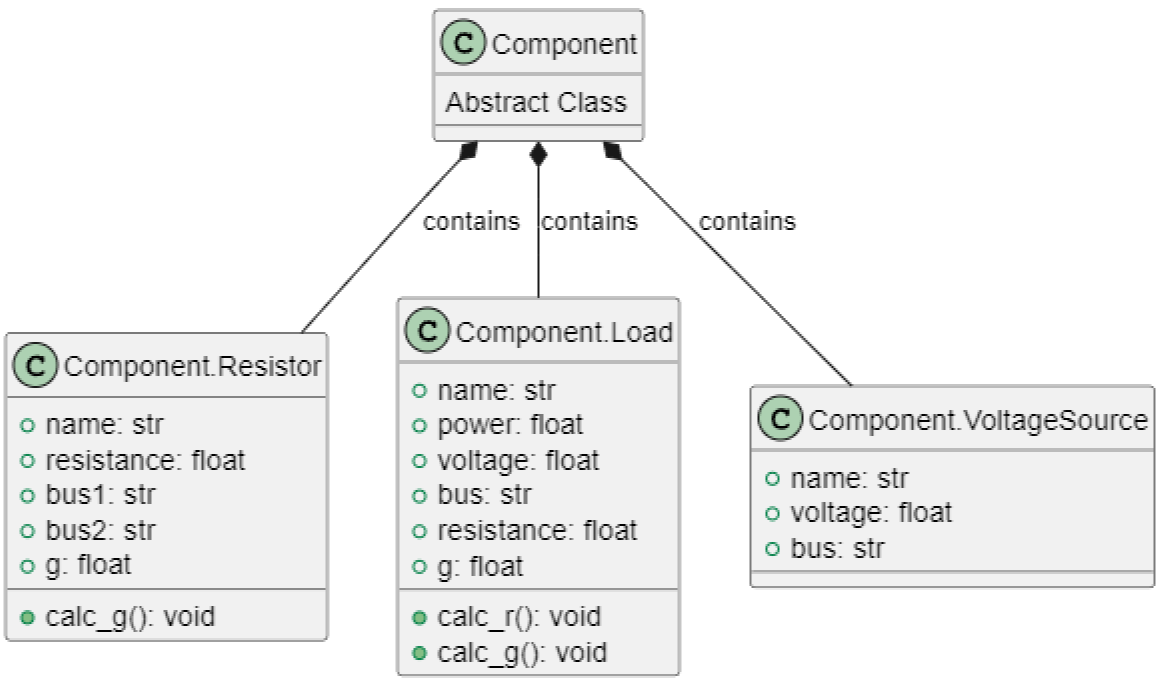

Figure 2: Component Class

Resistor

Attributes:

i. name (str): The user should provide this as an argument when defining the object.

ii. bus1 (str): The user should provide this as an argument when defining the object.

iii. bus2 (str): The user should provide this as an argument when defining the object.

iv. r (float): The user should provide this as an argument when defining the object.

v. g (float): It should be calculated internally using the calc_g method.

Methods:

i. calc_g(): Calculates the conductance value.

Load

Attributes:

i. name (str): The user should provide this as an argument when defining the object.

ii. bus1 (str): The user should provide this as an argument when defining the object.

iii. p (float): The user should provide this as an argument when defining the object.

iv. q (float): The user should provide this as an argument when defining the object.

v. g (float): It should be calculated internally using the calc_g method.

Methods:

i. calc_r(): Calculates the resistance of the load.

ii. calc_g(): Calculates the conductance value.

VoltageSource

Attributes:

i. name (str): The user should provide this as an argument when defining the object.

ii. bus1 (str): The user should provide this as an argument when defining the object.

iii. v (float): The user should provide this as an argument when defining the object.

Circuit

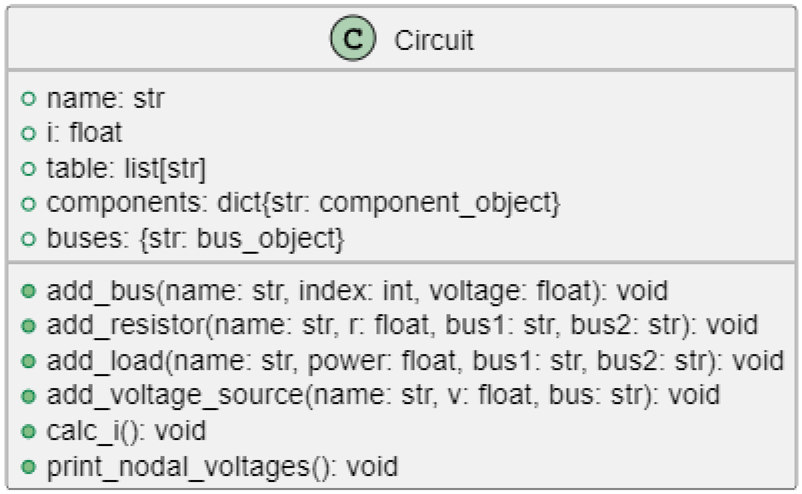

Figure 3: Circuit Class

Attributes:

i. name (str): The user should provide this as an argument when defining the object.

ii. buses (Dict[str, Bus]): A dictionary where each item has a bus name as the key and its corresponding Bus object as the value. The Bus object is created using the add_bus method of the Circuit class.

iii. components (Dict[str, component_name]): A dictionary where each item has a component name as the key and its corresponding Component object as the value. A Component object is created using any of the relative component methods.

iv. i (float): Current flowing through the circuit. It should be updated during the power flow calculation.

Methods:

i. add_bus(bus: str): Adds a bus to the circuit.

ii. add_resistor(name: str, bus1: str, bus2: str, r: float): Adds a resistor to the circuit.

iii. add_load(name: str, bus1: str, p: float, v: float): Adds a load to the circuit.

iv. add_voltage_source(name: str, bus1: str, v: float): Adds a voltage source to the circuit. v. set_i(i: float): Updates the i attribute.

vi. print_nodal_voltage(): Prints voltages at all buses.

vii. calc_i(): Calculates and prints the circuit current.

Solution

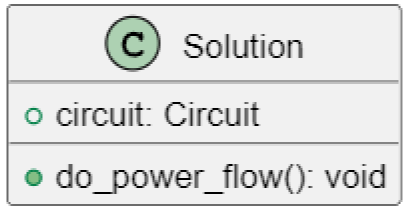

Figure 4: Solution Class

Attributes:

i. circuit (Circuit): When creating a solution object, you must pass a Circuit object as an argument.

Methods:

i. do_power_flow(): Solves the circuit by finding bus voltages and circuit current. First, calculate the current using element conductance values, then determine the voltage at bus B. This algorithm is designed specifically for this circuit, as the main goal is to understand the coding implementation of a simple circuit.

Relevant Equations

Since this simulator only solves simple linear DC circuits, the equations used are quite straightforward.

Load Resistance

For this software, we're only using two resistors, R_1 and R_2. R_2 is found by using the following relationship,

$$R_2 = \frac{V_{Load}^{\ 2}}{P_{Load}}.$$

Current

In general, the current through any series-connected circuit is given by the following equation.

$I = \frac{V}{R_{eq}}$, where $R_{eq} = \sum_{i=1}^{n} R_i$, or in words, the sum of all the individual resistive elements in the circuit.

We're only using two resistors so for our purposes,

$$I = \frac{V_{source}}{R_1 + R_2}.$$

Bus Voltage

After the current through the circuit is found, we can find the voltage at *Bus2* by using Ohm's law,

$$V_{bus2} = IR_1.$$

Example Case

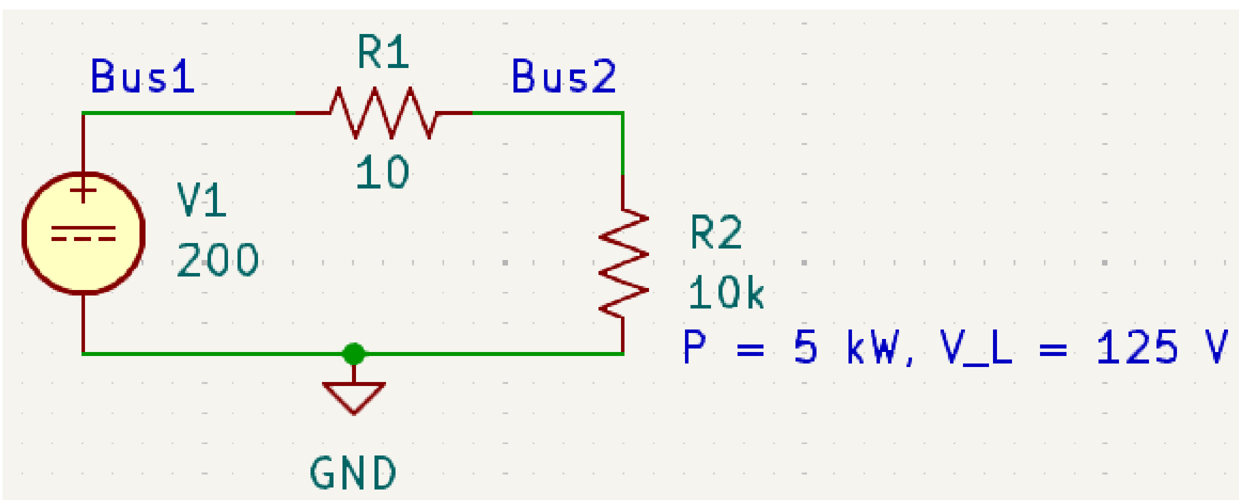

Figure 5: Example Circuit

Let's say we have the following circuit shown above. We want to find the voltage at *Bus2*. We have a resistive load, *R2*, operating at 125 V and consuming 5 kW of power. A 200 V DC source supplies the circuit, and a 10-ohm resistance is seen between the source and load. Following **Figure 6**, enter the information for your respective circuit, and the results will be displayed as seen in **Figure 7**.

```python
#  Example Case
circ2 = Circuit.Circuit("MyFirstCircuit")
circ2.add_bus("bus1")
circ2.add_bus("bus2")
circ2.add_voltage_source("V1", 200,"bus1")
circ2.add_resistor("R1", 10, "bus1", "bus2")
circ2.add_load("load1", 5000, 125, "bus2")
circ2.calc_i()
solution2 = Solution.Solution(circ2)
solution2.do_power_flow()
circ1.print_nodal_voltages()
```

Figure 6: Creating Circuit

```
Equivalent series resistance = 13.125 Ω
Circuit current = 15.238095238095237 A
Bus #0, 0 V
Bus #1, 200 V
Bus #2, 152.38095238095238 V
```

Figure 7: Example Results