EECS 118 Term Project Report

Part 2: Graph Solver

Aaron Chang (33197863)

Tun Myat (51705354)

Project Report for Graph solver

Introduction:

The question this program intents to solve.

21. find s where is_path(s, A, B) and no_nodes(s, t) and total_weight(s, u) and t=C and u=D

The goal of the program:

- Draw the graph from CSV input file.
- Figure out all the paths that satisfy.
  - Number of nodes = user input C
  - Total weight of edges = user input D
- Generate the result on the CSV output file.
  - Include all the paths that satisfy those user input conditions.

Explanation:

- Run the program in this format: python main.py <the_graph>.csv
- The program will read <the_graph>.csv file and draw into graph.

	A	B	C	D
1	1	2	3	green
2	1	3	3	red
3	2	3	3	yellow
4	2	4	3	white
5	2	5	3	black
6	3	4	3	green
7	4	5	3	yellow
8	4	6	3	blue
9	5	7	3	red
10	5	8	3	white
11	7	8	3	red
12	4	7	3	red
13	2	8	3	yellow
14	3	7	3	green

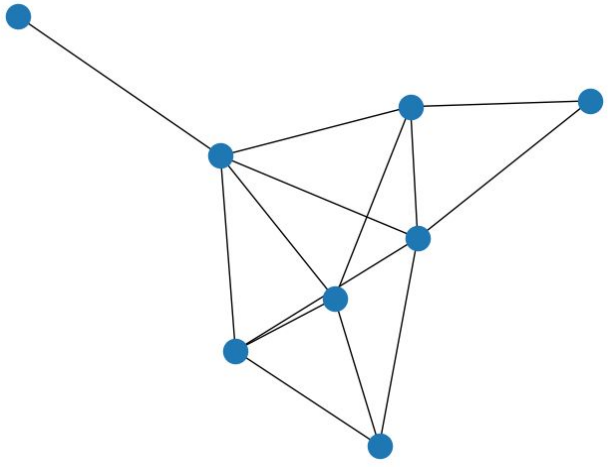

Fig 1: A test graph of graph.csv file (Node 1, Node 2, weight, color) and the output graph

```
Type: Graph
Number of nodes: 8
Number of edges: 14
Average degree:   3.5000
```

Fig 2: Information of the test output graph

- The program is using NetworkX library.
- all_simple_paths() function is being used to find all the possible paths.
  - As the path goes on, the total weight of each path is being added overtime.
  - Some weight values might be using decimal number. Since float is not accurate, it is not easy to check with float for exact number. So, I put an absolute value for two decimal number while checking weight.
  - The number of nodes for each path is also counted by using len(path) function.
  - If the total weight is equal to user input D and the number of nodes is equal to user input C, the program will create result.csv file and write all the paths row by row.

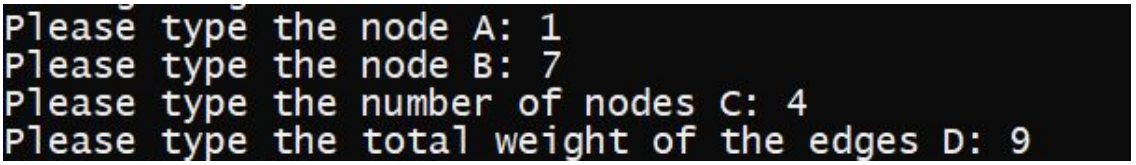

Fig 3: A test plan of user input

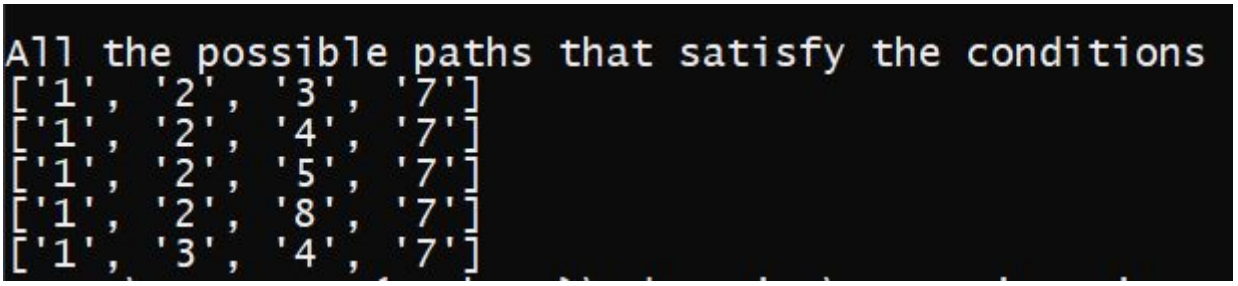

Fig 4: The result from the test plan

	A	B	C
1	Path_1		
2	1	2	
3	2	3	
4	3	7	
5	Path_2		
6	1	2	
7	2	4	
8	4	7	
9	Path_3		
10	1	2	
11	2	5	
12	5	7	
13	Path_4		
14	1	2	
15	2	8	
16	8	7	
17	Path_5		
18	1	3	
19	3	4	
20	4	7	

Fig 5: Information of the test result written on result.csv file

❖ If there is no equal path, it will end the program without writing anything into csv file.

```
Type: Graph
Number of nodes: 8
Number of edges: 14
Average degree:   3.5000
Please type the node A: 2
Please type the node B: 4
Please type the number of nodes C: 5
Please type the total weight of the edges D: 6
There is no result.
NULL
```

Fig 6: Information of the test result when there is no path satisfy

More test plans:

1. Test plans with float weight

	A	B	C	D
1	1	2	0.3	green
2	1	3	0.3	red
3	2	3	0.3	yellow
4	2	4	0.3	white
5	2	5	0.3	black
6	3	4	0.3	green
7	4	5	0.3	yellow
8	4	6	0.3	blue
9	5	7	0.3	red
10	5	8	0.3	white
11	7	8	0.3	red
12	4	7	0.3	red
13	2	8	0.3	yellow
14	3	7	0.3	green
15				

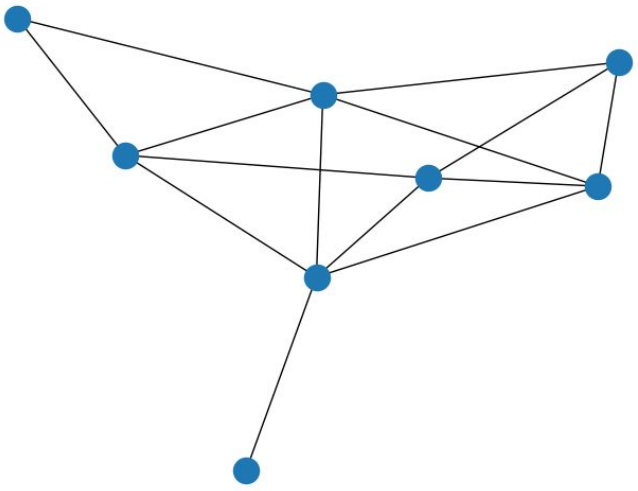

Fig 7: A test graph of graph.csv file (Node 1, Node 2, weight, color) and the output graph

```
Type: Graph
Number of nodes: 8
Number of edges: 14
Average degree:   3.5000
```

Fig 8: Information of the test output graph

```
Please type the node A: 1
Please type the node B: 7
Please type the number of nodes C: 4
Please type the total weight of the edges D: 0.9
```

Fig 9: A test plan of user input

```
All the possible paths that satisfy the conditions
['1', '2', '3', '7']
['1', '2', '4', '7']
['1', '2', '5', '7']
['1', '2', '8', '7']
['1', '3', '4', '7']
```

Fig 10: The result from the test plan

	A	B	C
1	Path_1		
2	1	2	
3	2	3	
4	3	7	
5	Path_2		
6	1	2	
7	2	4	
8	4	7	
9	Path_3		
10	1	2	
11	2	5	
12	5	7	
13	Path_4		
14	1	2	
15	2	8	
16	8	7	
17	Path_5		
18	1	3	
19	3	4	
20	4	7	

Fig 11: Information of the test result written on result.csv file

2. Test plans with float weight

	A	B	C	D
1	1	2	0.3	green
2	1	3	0.3	red
3	2	3	0.3	yellow
4	2	4	0.3	white
5	2	5	0.3	black
6	3	4	0.3	green
7	4	5	0.3	yellow
8	4	6	0.3	blue
9	5	7	0.3	red
10	5	8	0.3	white
11	7	8	0.3	red
12	4	7	0.3	red
13	2	8	0.3	yellow
14	3	7	0.3	green
15				

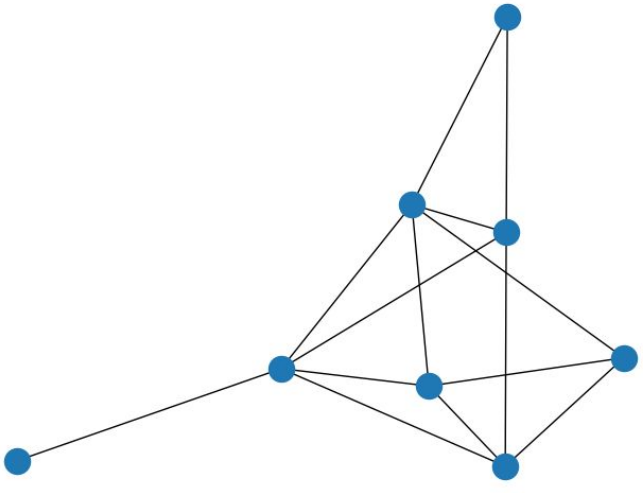

Fig 12: A test graph of graph.csv file (Node 1, Node 2, weight, color) and the output graph

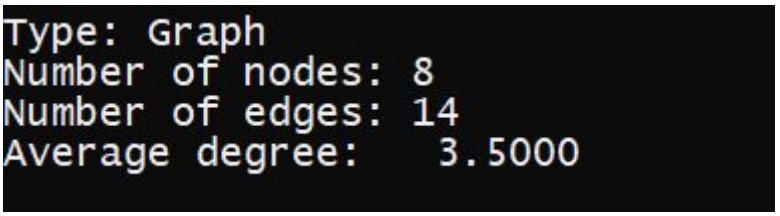

Fig 13: Information of the test output graph

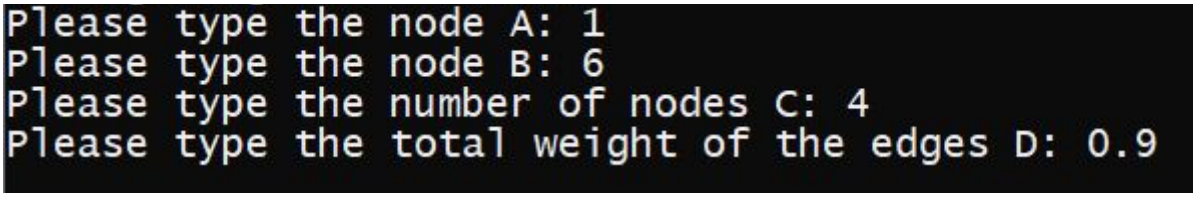

Fig 14: A test plan of user input

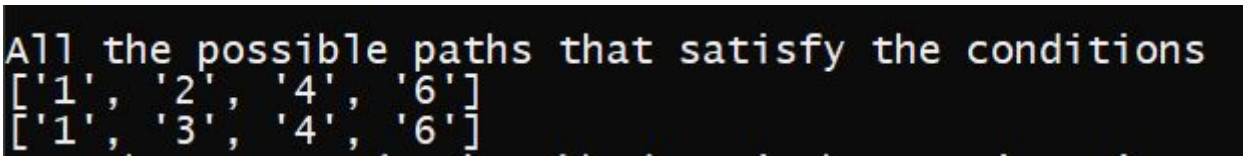

Fig 15: The result from the test plan

	A	B
1	path_1	
2	1	2
3	2	4
4	4	6
5	path_2	
6	1	3
7	3	4
8	4	6

Fig 16: Information of the test result written on result.csv file