Coding Project III

Find MST Using Kruskal's Algorithm - 10 Points

In this assignment you will use the provided code template to find the Minimum Spanning Tree of a graph using Kruskal's Algorithm. You will not only implement the algorithm itself, but you will also implement the `union` and `find` methods for union-find data structure.

Restrictions

- You must complete this assignment on your own; do not share your code with anyone and do not copy code from the Internet.
- Template code is provided and must be used. Note that the project template has changed; please be sure to use the current version of this assignment
- You code must be compatible with **python 3.10**
- No additional libraries may be imported beyond what is provided in the assignment template
- Do not modify the structure or program-flow of this assignment in any way – only add code where directed to do so by the code comments. Do not add functions, variables, or other code constructions except where told to do so – each individual component of your submission will be tested by the auto-grader when it is submitted

What is Provided

In addition to the template code, you have been given 2 files describing graphs: `small.txt` and `medium.txt`. Each has a solution file which has also been included in the assignment. You may choose which file you would like to use via the command-line argument -g.

The returned graph object has various functions and variables defined, but you should not need to access anything in this object directly. In the function `kruskal()`, notice that a sorted list of edges is already provided for you – all you need to do is access each edge within the provided loop. Note that an edge is a tuple composed of two vertex ids.

Union-Find

This data structure is covered in the text (Dasgupta 5.1.4). You will code two methods within the `unionFind` object as outlined in the text (the makeset functionality is already provided in the `unionFind` object's constructor). Follow the instructions in the code comment.

- For `find(p)`, you must use path compression.
- For `union(u,v)`, you must maintain both the rank and pi values for each vertex.

Submission

Gradescope will confirm if your submission passes the base case and creates the expected MST for `small.txt` and `medium.txt`. Your solution will be tested against three additional graph files and for proper implementation of path compression. Submit your code file (**mst.py**) ONLY to the Gradescope assignment on or before the posted due date. Do not submit a zip file, or any other files but **mst.py**. Late submissions will not be accepted.