Assignment 3

Due: 28th May 2020, 11:30PM

Total Marks: 50

Note: Make different program files for a different question. The name of the file should be the same as the problem name (for example problem1, problem2, and so on). Then, make a zip file with all the programs. Finally, submit the zip file. The name of the zip file should be your firstname_lastname.zip (for example, Salimur_Choudhury.zip).

Expectations: It is a general expectation that your code should have enough comments. Variable and functions name are meaningful (e.g., camel notation). If you need to add any special instructions on how to run your code, you can add a readme file too. Please make sure that your program runs without error.

Problem 1 (25): Write a function **MyBipartiteGraph(N)** that the number of nodes of the graph as the input for the generation of a random bipartite graph. Then you should generate the bipartite graph and put random edges with random weights. There should be another function **TestBipartiteGraph(G)** to check if the generated graph G is bipartite or not. Finally, design an ILP for solving the maximum weighted bipartite matching problem. Please do research on the definition of the maximum weighted bipartite problem. You can find the definition from the following link as well
https://en.wikipedia.org/wiki/Maximum_weight_matching
Then, implement and solve the generated weighted bipartite graph problem in Gurobi and print the maximal weight found by the matching using ILP.

For this problem, along with your program file, you need to submit a separate document (pdf) where you will write the formal ILP formulations (mathematically). Please check the Linear programming lecture and see how generalized solutions for vertex cover and maximum flow were written. Please note that you have to write the generalized ILP, not an example one.

Problem 2 (25): Suppose, you have to build a college admission system that matches different colleges to different students. In the system, there will be 5000 students and 15 colleges in total. Each college has 30-40 spots available for student intake. Meanwhile, each student has a list of preferences for colleges and each college will have a list of preferences for students. Both the preference lists should be generated randomly and show preferences in decreasing order of priority. Now, your task is to design and implement an algorithm for matching colleges to students, where the final solution is stable. Again, you have to submit two files: a) one pdf explaining how your algorithm works. Please check the stable matching lecture and see how algorithm was written.

Then also explain why your algorithm will always terminate. b) A program that implement your algorithm. You can write the program in C/C++, Java or Python.