CMPSC 565: Design and Analysis of Algorithm **(Deadline: 12/05/2022-12/07/2022)**

Project: Leafy spanning trees

Instructor: Dr. Antonio Blanca *TA:* Jeremy Huang

To appreciate the notion of NP-completeness, and to get hands-on experience coping with hard problems, in this project you will try to solve and NP-complete problem.

You will form teams of up to three students for the project. Your task is to write a program to solve the Maximum Leaf Spanning Tree problem (defined below). The program should aim to find solutions that are as good as possible in as many situations as you can make it. You can use any technique, algorithm, heuristic, software package, or programming language you'd like.

You will also create hard instances that we will use to challenge the solutions of the other teams. Your team's performance in the project will be based on:

1. how well your program does on other teams' instances, and
2. how badly other programs do on your instances.

Specific details:

In the Maximum Leaf Spanning Tree problem, you are given a graph $G = (V, E)$. You are asked to find a spanning tree for G having as many leaf nodes as possible. The input size will be around $n := |V| \leq 100$ and $m := |E| \leq 2000$.

The deadline for submitting hard instances is 12/05/2022. After collecting hard instances, (within a day) we will post them for all teams to solve. The deadline for submitting solutions is 12/07/2022. After collecting solutions, we will judge teams' performances against each other.

You can write in any programming language of your choice (or even to mix different programs and to fine tune performances based on the hard instances). However, together with the solutions, you need to submit at least one program which is representative of your algorithm.

First submission:

Each team should submit two files: hard.in and hard.out. The file hard.in should contain at least 5 hard instances, but you are welcome to put more. The first line of hard.in should be a single number k corresponding to the number of instances or graphs. The description of k graphs should follow. The first line of the description for each graph contains two numbers n and m corresponding to the number of vertices and edges on the graph. The next m lines contain the description of each edge, consisting of two numbers which represent the unique id of the two vertices it connects. We will assume that vertices are numbered 0 to $n - 1$.

Example hard.in file:

```
2
3 3
0 1
0 2
1 2
4 4
0 1
1 2
2 3
0 3
```

The corresponding hard.out file should contain k solutions. The first line of each solution contains two numbers s and ℓ, where s is the number of leaves on your solution and ℓ is the number of edges on your tree. The next ℓ lines will contain each one edge consisting of two space separated sorted numbers corresponding to the nodes id's. Edges should appear lexicographically, from least to largest.

Example hard.out file:

```
2 2
0 1
1 2
2 3
0 3
1 2
2 3
```

Final submission:

After every team submits their hard.in and hard.out files, we will create and distribute a all-hard.in file that combines all your hard instances. For the second submission you will submit the corresponding all-hard.out file. You will also submit one pdf file with at least one program which is representative of your algorithm; make sure your code is well-commented and readable throughout.