CS4234: Optimisation Algorithms Tutorial 1

Min(-W)-Vertex-Cover, (Integer) LP, Max-Clique, PS1

V1.2: Steven Halim *August 27, 2018*

Preliminaries

As this is the first tutorial session, let's use the first few minutes for the tutor (Wei Liang, all tutorial groups 1+2+3) to know slightly more about you and for you all to know him a bit.

Discussion Points

Q1: Can we solve Min-Vertex-Cover in polynomial time if all vertices in the input graph have degrees not more than 2? What if we want to solve Min-Weight-Vertex-Cover variant on the same graph type?

Q2: In lecture, we have seen a Dynamic Programming (DP) solution to solve the Min-Vertex-Cover on a (Binary) Tree. How about the following Greedy Algorithm (solution for CLRS Ex 35.1-4): First, take any leaf in the tree, then add its parent to the (minimal) vertex cover, then delete the leaf and parent and all associated edges. Repeat this process until there is no vertex remain in the tree. Is this a correct Greedy Algorithm? Hint: Check `https://visualgo.net/en/mvc`, 'MVC on Tree', the 'Greedy MVC on Tree' option. Can it be used on the Min-Weight-Vertex-Cover variant?

Q3: A carpenter makes tables and chairs. Each table can be sold for a profit of 25 SGD and each chair for a profit of 11 SGD. The carpenter can afford to spend up to 40 hours per week working and takes 5 hours to make a table and 3 hours to make a chair. The owner requires that the carpenter makes at least 3 times as many chairs as tables (so that he can package it as dining table set). Tables take up 4 times as much storage space as chairs and there is room for at most 10 tables each week.

Formulate this problem as a Linear Programming problem (write in standard form) and solve it (graphically, with Excel, or with a (Simplex? or Brute Force?) program :O).

Now if the solution is a floating point numbers, please round them so that we have an Integer solution. Are you sure your Integer solution is the optimal one?

Q4: Recall the basic idea behind the Simplex algorithm is that it starts at some vertex and then examines the value of the objective function at the neighboring vertices. How does the Simplex algorithm finds neighbors exactly?

Q5: The year is 2020. Steven has added a lot of features in VisuAlgo and he has started the foray towards visualization of various NP-hard optimization problems (assume that nobody has proven whether $P = NP$ (or not) by then). You are one of his Final Year Project (FYP) student who is assigned to do Max-Clique visualization (which has not appeared in `https://visualgo.net/` yet...). If you forget, the decision version of Clique is defined as follows: "Given a graph $G = (V, E)$ and an integer k, is there a subset $C \subseteq V$ of size k such that C is a clique in G? The Max-Clique optimization variant seeks for the subset C with the largest possible k in G. Here are Steven's requirements:

1. VisuAlgo user *must* be able to draw his/her own graph input, as per VisuAlgo standard.
2. Steven dislikes graphs that are drawn with edge crossings, so he has put on a check in his "Draw Graph" feature so that only graphs without edge crossing are allowed in VisuAlgo.

3. VisuAlgo must be very responsive, i.e. upon clicking "Show Max-Clique" button, a Java?Script (the year is 2020, remember) routine has to quickly compute the answer in not more than 1 second. Longer than that, Steven says that the visitors will press Alt+F4 instead.

What is the *best* algorithm that will you implement as Java?Script routine of your MAX-CLIQUE visualization in VisuAlgo? What is the time complexity? Is your algorithm seeks for an approximate solution or an optimal solution? What are the rough limit of n and m, the number of vertices and edges, that you will set in your visualization?

PS1: What are the meaning of the constraints in Version I, II, III, and IV? Therefore, what are the required algorithms for each version? This is your last chance to ask your tutor about PS1 that will due on Saturday, 01 September 2018, 07.59am.