CS4234: Optimisation Algorithms Tutorial 1

M(W)VC, (I)LP, Max-Clique, Graph-Coloring, PS1 Debrief

V1.7: A/Prof Steven Halim *August 28, 2023*

Preliminaries

As this is the first tutorial session, let's use the first few minutes for the tutor (Lee Zheng Han (2 classes: T1+T2) and Teoh Jun Jie (1 class: T3) for this S1 AY2023/24) to know slightly more about you.

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 that 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, with lp_solve, 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: Prof Halim wants to add a few new features involving various NP-hard optimization problems to his VisuAlgo project (assume that nobody has proven whether $P = NP$ (or not) as of this tutorial session). 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 Prof Halim's requirements:

1. VisuAlgo user *must* be able to draw his/her own graph input, as per VisuAlgo standard.
2. Prof Halim dislikes graphs that are drawn with edge crossings, so he has put on a check in his "Edit 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 JavaScript routine has to quickly compute the answer in not more than 1 second. Longer than that, Prof Halim says that the visitors will press Alt+F4 instead.

What is the *best* algorithm that will you implement as JavaScript 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?

Q5: Follow up of Q4 above. All Prof Halim's requirements are identical, just that you are another FYP student who is assigned to do GRAPH-COLORING visualization (which also has not appeared in `https://visualgo.net/` yet...). The decision version of GRAPH-COLORING is defined as follows: "Given a graph $G = (V, E)$ and an integer k, can we assign a color (an integer in $[1..k]$) to each vertex such that no two adjacent vertices (that share an edge) are of the same color? The GRAPH-COLORING optimization variant seeks for coloring with the smallest possible k.

PS1 Debrief: To be summarized briefly: What are the underlying problems+(alternative) solutions of PS1 Task A/B/C/D? What are the required algorithms for each task? PS: E/F/G/H are optional.