CS4234: Optimisation Algorithms Tutorial 3

TSP, Max-Independent-Set, PS3

V1.7: A/Prof Steven Halim *September 11, 2023*

Discussion Points

Q1: In Lecture 04, Prof Halim did not show any approximation algorithm for the G-NR-TSP variant of TSP (that is, the general, non-metric version and without repeated vertex). He says that 'it is (NP-)hard even to approximate'. Why?

Q2: In class, we formulated the TSP (4 variants) in terms of a set of points V and a distance function d that gives the distance between any two points in V (thus, a complete graph with V^2 edges that surely has $(V-1)!$ Hamiltonian Cycles). What if the input to the problem is a graph $G = (V, E)$ with E weighted edges and $0 \leq |E| \leq V * (V-1)/2$?

Define a version of TSP for graphs, and explain whether or not it remains approximate-able using the techniques discussed in class. (Consider these: What if there is no Hamiltonian Cycle in the input to begin with? What if the input graph is actually disconnected?)

Q3: Think about the *Euclidean* TSP where each point has (x, y)-coordinates to identify it (for simplicity, let's assume that all x-coordinates of the n points are different). Imagine that Prof Halim only wants "cycles" that proceed in one direction, e.g., left-to-right. For example, a legal output is a cycle $(v_1, v_2, v_3, v_4, v_1)$ where for each $i < n$, we know that $v_i.x < v_{i+1}.x$. (Only in the last step of the cycle, going back from v_n to v_1, we are allowed to go to the left.) Is there an efficient algorithm to solve this *non-standard* version of TSP problem? Next, give an example where the cycle that is found in this case is very bad compared to the optimal TSP cycle.

Q4: Prof Halim needs to continually increase the number of (NP-)hard problems that has been exposed to CS4234 students so far (to have more interesting Midterm Test and Final Assessment). So let's study this yet other problem Max-Independent-Set (MIS). It is very similar to MVC and defined as follows: Given a graph $G = (V, E)$, pick the maximum-size set $I \subset V$ so that no two vertices in I share an edge.

You are told that MIS is also an NP-hard optimization problem (proof omitted, but you can reduce NP-hard VC to IS easily), but here you are given a network that is arranged as a grid, as in Figure 1:

Part 1. The grid has n vertices, and each of the vertices (except for those on the edges) has four neighbors. The goal of this question is to develop an algorithm for finding a maximum-sized Max-Independent-Set (MIS) for this graph (or an approximation). What is the MIS (and its size) of Figure 1 above?

Part 2. Consider the following greedy algorithm for graph $G = (V, E)$:

- Set $I = \emptyset$;
- Repeat until V is empty:
  - Choose any arbitrary vertex $u \in V$,
  - Add u to I,

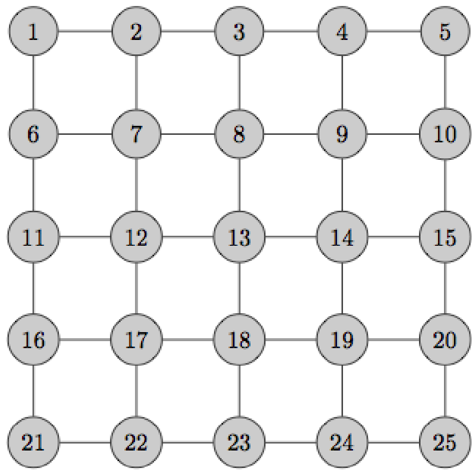

Figure 1: A Grid Graph of size 5×5.

- Delete u all the neighbors of $u \in V$.

Now argue that this algorithm is a 2.5-approximation of optimal on grid graph like in Figure 1 above. (But first, show that it is a correct algorithm that produces an independent set!)

Part 3. (Optional to save time, just read the modal answer): Write down MIS as an ILP!

Q5: One of the PS3 problems involves another new NP-hard problem that has not been discussed earlier (in lecture and/or previous tutorial). Which PS3 problem is it? What is the underlying NP-complete decision problem? What is the general idea to tackle this problem?