CSC373 Fall'19
Assignment 4
Due Date: Dec 1, 2019, by 11:59pm

Instructions

1. Be sure to include your name and student number with your assignment. Typed assignments are preferred (e.g., PDFs created using LaTeX or Word), especially if your handwriting is possibly illegible or if you do not have access to a good quality scanner. Please submit a single PDF on MarkUS at `https://markus.teach.cs.toronto.edu/csc373-2019-09`.
2. You will receive 20% of the points for any (sub)problem for which you write "I do not know how to approach this problem." (you will receive 10% if you leave the question blank and do not write this or a similar statement). Not applicable to BONUS questions.
3. You may receive partial credit for the work that is clearly on the right track. But if your answer is largely irrelevant, you will receive 0 points.

Q1 [20 Points] Activity Selection

There are m students in a class, and a set of activities U happening in the class. Each student i is involved in a subset of activities $S_i \subseteq U$. We are told that for every activity in U, there are *exactly four* students involved in that activity.

We need to select some of the students as representatives. Our constraint is that for each activity in U, *at least three of the four students* involved in that activity must be selected. However, each student i already has some workload $w_i \geqslant 0$. So subject to that constraint, we want to minimize the total workload of the students selected as representatives.

(a) [5 Points] Write this problem as an *integer* program with 0-1 variables. Briefly explain what your variables are, and how an optimal solution to your program represents an optimal solution to our problem.

(b) [15 Points] Use LP relaxation and rounding to obtain a deterministic 2-approximation algorithm. Explain why your rounded solution is a feasible solution to the integer program and why it provides 2-approximation.

Q2 [20 Points] Coffee Shop Dilemma

$$
\begin{array}{ccccc}
(m,1) & \text{——} & (m,2) & \text{—}\cdots\text{—} & (m,n) \\
| & & | & & | \\
\vdots & & \vdots & \cdots & \vdots \\
| & & | & & | \\
(2,1) & \text{——} & (2,2) & \text{—}\cdots\text{—} & (2,n) \\
| & & | & \cdots & | \\
(1,1) & \text{——} & (1,2) & \text{—}\cdots\text{—} & (1,n)
\end{array}
$$

Your friends want to break into the lucrative coffee shop market by opening a new chain called *The Coffee Pot*. They have a map of the street corners in a neighbourhood of Toronto (shown above), and for each (i, j), an estimate $p_{i,j} \geqslant 0$ of the profit they can make if they open a shop on corner (i, j). If they open shops on multiple corners, their profits add up. So ideally, they would like to open a shop at every corner ("the dream").

However, if they open a shop on corner (i, j), municipal regulations forbid them from opening shops on *adjacent* corners $(i-1, j)$, $(i+1, j)$, $(i, j-1)$, and $(i, j+1)$ (whichever exist). As you can guess, they would like to select street corners where to open shops in order to maximize their profits!

(a) [5 Points] Consider the following greedy algorithm to try and select street corners. Give a precise counter-example to show that this greedy algorithm does not always find an optimal solution. Clearly state your counter-example (values of m, n, and $p_{i,j}$ for $1 \leqslant i \leqslant m$, $1 \leqslant j \leqslant n$), the solution found by the greedy algorithm, and a different solution with larger profit.

> $C \leftarrow \{(i,j) : 1 \leqslant i \leqslant m, 1 \leqslant j \leqslant n\}$ # C is the set of every available corner
> $S \leftarrow \varnothing$ # S is the current selection of corners
> **while** $C \neq \varnothing$:
>   pick $(i, j) \in C$ with the maximum value of $p_{i,j}$
>   # Add (i, j) to the selection and remove it (as well as all corners adjacent to it) from C.
> $S \leftarrow S \cup \{(i, j)\}$
> $C \leftarrow C \setminus \{(i,j), (i-1,j), (i+1,j), (i,j-1), (i,j+1)\}$
> **return** S

(b) [15 Points] Prove that the greedy algorithm from part (a) gives 4-approximation.

[Hint: Let S be the selection returned by the greedy algorithm and let T be an optimal solution. Show that for all $(i, j) \in T$, either $(i, j) \in S$ or there is an adjacent $(i', j') \in S$ with $p_{i',j'} \geqslant p_{i,j}$. What does this means for all $(i, j) \in S$ and their adjacent corners?]

Q3 [20 Points] Randomized Algorithm

Recall that a 3CNF formula $\varphi = C_1 \wedge \ldots \wedge C_m$ consists of a conjunction of m clauses, where each clause is a disjunction of *exactly* 3 literals. Each clause C_r has a corresponding weight $w_r \geqslant 0$. In the standard Exact Max-3-SAT problem, our goal was to maximize the total weight of clauses that are "satisfied" (i.e. in which at least one literal is true). Recall the naive randomized algorithm from class which provided 3/4-approximation and its derandomized deterministic version.

Now, consider a related problem, Exact Robust-Max-3-SAT, in which a clause is considered "satisfied" when *at least two literals* are true, and the goal is still to maximize the total weight of clauses that are "satisfied", but under the new definition of clause satisfaction.

(a) [5 Points] Give a randomized 1/2-approximation algorithm for Exact Robust-Max-3-SAT. Specifically, your algorithm should return a random truth assignment of variables such that the *expected* number of clauses "satisfied" is at least $m/2$. Argue correctness of your algorithm.

(b) [15 Points] Derandomize your algorithm to derive a deterministic algorithm which *always* returns a truth assignment of variables "satisfying" at least $m/2$ clauses. Write pseudocode for your derandomized algorithm, justify its correctness, and analyze its worst-case running time.