CS 473: Algorithms, Fall 2016
HW 8 (due Tuesday, November 1st at 8pm)

This homework contains three problems. **Read the instructions for submitting homework on the course webpage**.

Collaboration Policy: For this home work, each student can work in a group with up to three members. Only one solution for each group needs to be submitted. Follow the submission instructions carefully.

You may assume the following results in your solutions:

- Maximum flows and minimum cuts can be computed in $O(VE)$ time.
- Minimum-cost flows can be computed in $O(E^2 \log^2 V)$ time.
- Linear programming problems with integer coefficients can be solved in polynomial time.

For problems that ask for a linear-programming formulation of some problem, a full credit solution requires the following components:

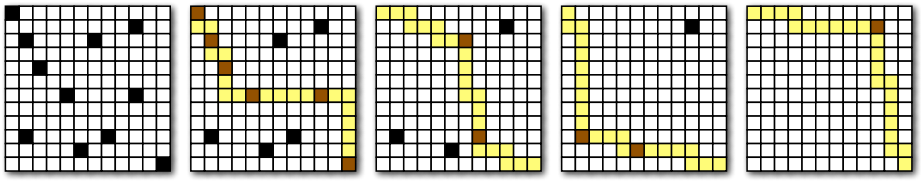
- A list of variables, along with a brief English description of each variable. (Omitting these English descriptions is a Deadly Sin.)
- A linear objective function (expressed either as minimization or maximization, whichever is more convenient), along with a brief English description of its meaning.
- A sequence of linear inequalities (expressed using $\leq$, $=$, or $\geq$, whichever is more appropriate or convenient), along with a brief English description of each constraint.
- A proof that your linear programming formulation is correct, meaning that the optimal solution to the original problem can always be obtained from the optimal solution to the linear program. This may be very short.

It is ***not*** necessary to express the linear program in canonical form, or even in matrix form. Clarity is much more important than formality.

For reductions to flow-based algorithms described in class or in the notes (for example: edge-disjoint paths, maximum bipartite matching, minimum-cost circulation) or to other standard graph problems (for example: reachability, minimum spanning tree, shortest paths), see HW 7.

---

1. Suppose we are given an $n \times n$ grid, some of whose cells are marked; the grid is represented by an array $M[1..n, 1..n]$ of booleans, where $M[i, j]$ = True if and only if cell (i, j) is marked. A monotone path through the grid starts at the top-left cell, moves only right or down at each step, and ends at the bottom-right cell. Our goal is to cover the marked cells with as few monotone paths as possible.



Greedily covering the marked cells in a grid with four monotone paths.

   - **Not to submit:** Describe an algorithm to find a monotone path that covers the largest number of marked cells.
   - **Not to submit:** There is a natural greedy heuristic to find a small cover by monotone paths: If there are any marked cells, find a monotone path Π that covers the largest number of marked cells, unmark any marked cells covered by Π, and recurse. Show that this algorithm does *not* always compute an optimal solution.
   - Describe and analyze an efficient algorithm to compute the smallest set of monotone paths that covers every marked cell.

2. Let $G = (V, E)$ be a graph with edge weights given by $c : E \to \mathbb{R}$. In the min-cost perfect matching problem the goal is to find a minimum cost perfect matching M in G (if there is one, otherwise the algorithm has to report that there is none) where the cost of a matching M is $\sum_{e \in M} c(e)$; note that the costs can be negative. In the maximum weight perfect matching problem the input is a graph $G = (V, E)$ and *non-negative* weights $w : E \to \mathbb{R}_+$ and the goal is to find a matching M of maximum weight. Note that a maximum weight matching may not be a maximum cardinality matching. Describe an efficient reduction from min-cost perfect matching to max-weight matching.

3. Consider a polyhedron P in n dimensions defined by a set of m inequalities $Ax \leq b$. Let $z \in \mathbb{R}^n$ be a point. Describe a polynomial-time algorithm to check whether z is a basic feasible solution of P.

The remaining problems are for self study. Do *NOT* submit for grading.

- See Problems 1 and 2 in HW 8 from Jeff's home work last spring. `https://courses.engr.illinois.edu/cs473/sp2016/hw/hw8.pdf`
- See HW 5 from Chandra's graduate algorithms course in Fall 2011. `https://courses.engr.illinois.edu/cs573/fa2011/Homework/hw5.pdf`