Analysis of Algorithms	**Problem Set 2**
CS 6820, Fall 2020	**Due Friday, October 16, 6:00pm**

Solve ALL FOUR of the problems on this problem set.

In solving the following problems you are allowed to use, without proof, any facts from the lectures or assigned readings in the Fall 2020 version of CS 6820. Hand in your solutions electronically using CMS. Scanned PDF files are allowed, if you prefer to hand-write your solutions. Each group is responsible for submitting just one solution set.

Almost all of the problems are designed to be challenging — particularly those marked with an asterisk — and it is acceptable if you fail to solve all of the problems. In fact, the expectation is that many groups will not produce four correct solutions.

(1) *(10 points)* Recall that in class we saw that the number of phases of Dinitz's algorithm (i.e., the number of outer loop iterations, each of which involves performing one blocking flow computation) is bounded above by n, the number of vertices in the flow network. Show that this bound is tight, up to constant factors. In other words, for a positive constant $c > 0$, construct an infinite sequence of flow networks, $G_1, G_2, G_3, \ldots$, such that:

1. $n(G_i)$, the number of vertices in G_i, tends to infinity as i tends to infinity;
2. when Dinitz's algorithm is used to compute a maximum flow in G_i, it performs at least $c \cdot n(G_i)$ blocking flow computations before terminating.

(2) *(10 points)* This exercise explores the running time of Dinitz's algorithm when the input is a flow network with $\{0, 1\}$-valued edge capacities. Throughout the exercise, we will assume that the network satisfies $n = O(m)$; this assumption is satisfied whenever the network has no isolated vertices, and it is convenient for analyzing running times since it implies that $O(m + n) = O(m)$.

(2a) *(1 point)* Consider an execution of Dinitz's algorithm in a flow network with $\{0, 1\}$-valued edge capacities, and let G_f denote the residual graph at the end of any phase. Prove that the residual capacities in G_f are $\{0, 1, 2\}$-valued.

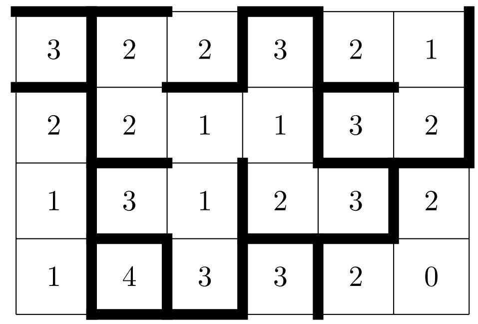
(2b) *(1 point)* Suppose that the residual graph G_f at the start of a phase of Dinitz's algorithm has $\{0, 1, 2\}$-valued edge capacities. Prove that the phase runs in time $O(m)$. *(Remark: This was proven in class for $\{0, 1\}$-valued capacities on October 5.)*

(2c) *(4 points)* Prove that the number of phases of Dinitz's algorithm in a graph with $\{0, 1\}$-valued edge capacities is bounded above by $O(m^{1/2})$. Consequently the entire algorithm runs in time $O(m^{3/2})$.

(2d*) *(4 points)* Prove that the number of phases of Dinitz's algorithm in a graph with $\{0, 1\}$-valued edge capacities is *also* bounded above by $O(n^{2/3})$. Consequently the entire algorithm runs in time $O(m \cdot \min\{m^{1/2}, n^{2/3}\})$.

(3) *(10 points)* Consider a puzzle in which you are given an n_1-by-n_2 rectangular grid, with an integer in the range $\{0, \ldots, 4\}$ written inside each grid cell. You are asked to select a subset F of the edges of the grid, such that for each grid cell the number written inside the cell matches the number of elements of F that belong to the cell's boundary. The following figure shows an example of a puzzle and a valid solution to the puzzle.

3	2	2	3	2	1
2	2	1	1	3	2
1	3	1	2	3	2
1	4	3	3	2	0



Design and analyze a polynomial-time algorithm that, when given such a puzzle, either outputs a valid solution or decides (correctly) that there is no valid solution.

HINT: *It might be easiest to start by solving the special case in which each grid cell is labeled with either 0 or 1.*

(4*) *(10 points)* You are organizing the garbage collection schedule for a rural area consisting of n towns. You have a single garbage truck. On any given day, the truck can visit only one town. The goal is to design a schedule so that no town accumulates too much garbage during any interval between consecutive garbage truck visits.

Let $g(i, j) \geq 0$ denote the amount of new garbage produced in town i on day j. We will assume that the total amount of garbage produced in a day is always bounded above by 1.

$$\forall j \quad \sum_{i=1}^{n} g(i, j) \leq 1.$$

A garbage pick-up schedule specifies which town the truck visits each day. A schedule is *admissible* if the total amount of garbage that builds up in any town between consecutive visits is less than 2. More formally, if i is any town and j_0, j_1 are two dates such that the garbage truck makes no visits to the town during the interval $\{j_0, j_0 + 1, \ldots, j_1\}$ then

$$\sum_{j_0 \leq j \leq j_1} g(i, j) < 2.$$

Design and analyze a polynomial-time algorithm to compute an admissible schedule, given as input the values $g(i, j)$ for every town i and date j. (Note that this means the problem is making the unrealistic assumption that the amount of garbage that will produced in each town is known in advance.) Your solution should include a proof that an admissible schedule always exists; most likely this proof will arise as a by-product of analyzing your algorithm.