# CSE 421 Winter 2021
# Homework 7

**Due: Friday, March 5, 2021, 6:00 pm**

**Problem 1:**
Let $G = (V, E)$ be an arbitrary flow network, with source $s$, sink $t$, and capacity $c_e$ for each directed edge $e \in E$. Let $f$ be a maximum flow in $G$ that has flow value $v(f)$. Let $(A, B)$ be a minimum $st$-cut in $G$ and let $k$ be the number of edges directed from $A$ to $B$ that cross this cut. Suppose that we add 1 to the capacity $c_e$ of *every* edge in $E$ to get a new capacity $c'_e$ for each edge $e$ in $G$. Let $f'$ be a maximum flow in the graph $G$ using capacities given by the $c'_e$. Prove or disprove each of the following statements.

- We always have $v(f') = v(f) + k$.
- It is sometimes the case that $v(f') < v(f) + k$.
- It is sometimes the case that $v(f') > v(f) + k$.

**Problem 2:**
A service provider has deployed a wireless network in a city via of a number of base station towers at fixed locations in the city. The stations were installed at different times so they have a variety of power/reach of their signal and can each support a variety numbers of clients. Each client they wish to serve is also assumed to be served via an antenna at a fixed and known location. The goal of the provider is to serve as many of their clients as possible by allocating their clients to talk with with specific based stations.

Suppose that the locations of each of the $B$ base station towers and each of the $n$ client antennas are at known coordinates in the $xy$-plane. Suppose that base stations 1 through $B$ have signal radius $r_1, \ldots, r_B$, respectively (that is, base station $b$ can communicate with any client within Euclidean distance $r_b$ of its location) and, respectively, can handle a load of up to $L_1, \ldots, L_B$ clients at the same time,

Give a polynomial-time algorithm that will figure out the maximum number of clients that the service provider can communicate with at a time, and an allocation of clients to base station towers that will achieve this maximum.

**Problem 3:**
In emergency planning we often need to set up escape routes ahead of time in case of a disaster. In this problem you are given a directed graph $G = (V, E)$ representing a set of possible 1-way roads. Each node in $V$ is either a populated node in $P$, a safe node in $S$, or an intersection node in $I$. (These are disjoint sets of nodes whose union is $V$.)

(a) An *escape plan* consists of a collection $C$ of directed paths in $G$ such that

- each path in $C$ begins at a node in $P$ and ends at a node in $S$,
- for each node $u \in P$, there is exactly one path in $C$ that begins at $u$, and
- no two paths in $C$ share any edges of $G$.

Give a polynomial-time algorithm to determine whether or not an escape plan exists.

(b) Suppose also that every node $v$ not in $P$ has a capacity $c_v$. A *congestion-free* escape plan is an escape plan in which

- the directed path beginning at node $u \in P$ does not go through any other populated node, and
- for every node $v$ not in $P$, the total number of paths containing $v$ is at most $c_v$.

Show how to determine in polynomial-time whether or not such a congestion-free escape plan exists and produce such a plan if one does.

**Problem 4 (Extra credit):**
We saw how repeatedly augmenting along shortest paths (given by a BFS each time) produced a better runtime analysis for general network flow. In this problem you will do something similar in the special case of finding a maximum matching in a bipartite graph $G = (V, E)$ where the two sides of $V$ are $X$ and $Y$, $|V| = n$ and $|E| = m$. The idea here will be to augment along many shortest paths at the same time. In order for this to work, the paths better not share any edges. Your algorithm will add the slightly stronger condition that the paths do not share any vertices and will include as many shortest paths as possible at each step.

More precisely, at each round the new algorithm will simultaneously augment along all paths in some *maximal set* $P_{short}$ of vertex-disjoint shortest augmenting paths in the flow graph associated with the bipartite matching problem. (In other words, if $d = d_f(s, t)$ is the length of the shortest augmenting path in the residual graph $G_f$ then every path in $P_{short}$ has length $d$, no two paths in $P_{short}$ share any vertices other than $s$ or $t$ and any other $st$-path of length $d$ in $G_f$ shares at least one vertex other than $s$ and $t$ with some path in $P_{short}$.)

(a) Show how to find a set $P_{short}$ and do all the augmentations on its paths to get a new flow $f'$ in time $O(m+n)$.

(b) Prove that $d_{f'}(s, t) \geq d_f(s, t) + 2$.

(c) Given two matchings $M'$ and $M$ on graph $G$ we can define the symmetric difference, $M' \oplus M$, of $M'$ and $M$ to be the graph consisting of edges that occur in exactly one of the two matchings. $M' \oplus M$ consists of a collection of vertex-disjoint paths and cycles (why?).
Show that if $M'$ is a maximum matching and flow $f$ corresponds to a matching of $M$ then

i. $M' \oplus M$ contains exactly $|M'| - |M|$ vertex-disjoint odd-length paths,
ii. Any path of odd length $k$ in $M' \oplus M$ corresponds to an augmenting path of length $k + 2$ in $G_f$.
iii. Use these two properties to prove that once all augmenting paths in $G_f$ are of length at least $k+2$ then at most $n/k$ additional augmentations will produce a maximum flow (and hence maximum matching).

(d) Use the above analysis with a suitable value of $k$ to show that this algorithm computes a maximum matching in $O(m\sqrt{n})$ time