CME 305: Discrete Mathematics and Algorithms

Student: Laura Lyman (lymanla@stanford.edu)

HW#1 – Due at the beginning of class Thursday 01/26/17

1. Prove that at least one of G and $\overline{G}$ is connected. Here, $\overline{G}$ is a graph on the vertices of G such that two vertices are adjacent in $\overline{G}$ if and only if they are not adjacent in G.

   Proof. If G is connected, then at least one of G and $\overline{G}$ is connected and we are done. So suppose G is not connected. Let E and $\overline{E}$ denote the edge sets of G and $\overline{G}$ respectively. Let $u, v \in V$. If $(u, v) \in \overline{E}$, then there is a path (the edge) between u and v in $\overline{G}$. If $(u, v) \notin \overline{E}$, then $(u, v) \in E$. In this case, u and v must be on the same connected component of G, and since G is disconnected, it must have at least one more connected component. Let $b \in V$ be a vertex on this other connected component of G. Then $(u, b), (v, b) \notin E$ for otherwise b would be on the same connected component of u and v. Hence, $(u, b), (v, b) \in \overline{E}$. Since G is undirected, we therefore have a path $u \rightsquigarrow b \rightsquigarrow v$ in $\overline{E}$ between u and v. Hence, in all cases there exists a path in $\overline{E}$ between arbitrary $u, v \in V$, meaning $\overline{G}$ is connected when G is disconnected. Therefore at least one of G and $\overline{G}$ is connected. □

2. A vertex in G is *central* if its greatest distance from any other vertex is as small as possible. This distance is the *radius* of G.

   (a) Prove that for every graph G

   $$\text{rad } G \leq \text{diam } G \leq 2 \text{ rad } G$$

   Proof. Assume $|V| \geq 1$. (If not, rad $G =$ diam $G = 2$ rad G so the desired conclusion is satisfied.) Let $z \in V$ be a central vertex, so

   $$\max_{v \in V} d(z, v) = \min_{u \in V} \max_{v \in V} d(u, v) = \text{rad } G.$$

   That is, the greatest distance from z to another vertex is as small as possible in G. Note that a central vertex in G will always exist since the set of greatest distances from each vertex to any other vertex is a finite nonempty set, so that set has a minimum value and thus an associated vertex (which is central) for that minimum. Then

   $$\begin{aligned} \text{diam } G &= \max_{u \in V} \max_{v \in V} d(u, v) && \text{[definition of diameter]} \\ &\leq \max_{u \in V} \max_{v \in V} (d(u, z) + d(z, v)) && \text{[triangle inequality]} \\ &\leq \max_{u \in V} \max_{v \in V} d(u, z) + \max_{u \in V} \max_{v \in V} d(z, v) \end{aligned}$$

   where the last step follows since

   $$d(u, z) + d(z, v) \leq \max_{u \in V} \max_{v \in V} d(u, z) + \max_{u \in V} \max_{v \in V} d(z, v)$$

   for all $u, v \in V$, so in particular the inequality holds for the u, v that maximize $d(u, z) + d(z, v)$; that is,

   $$\max_{u \in V} \max_{v \in V} (d(u, z) + d(z, v)) \leq \max_{u \in V} \max_{v \in V} d(u, z) + \max_{u \in V} \max_{v \in V} d(z, v).$$

Therefore we have that

$$\begin{aligned}\text{diam } G &\le \max_{u\in V}\max_{v\in V} d(u,z) + \max_{u\in V}\max_{v\in V} d(z,v)\\ &= \max_{u\in V} d(u,z) + \max_{v\in V} d(z,v)\\ &= \text{rad } G + \text{rad } G = 2 \text{ rad } G\end{aligned}$$

since z is central. Furthermore, by definition of radius and diameter,

$$\text{rad } G = \min_{u\in V}\max_{v\in V} d(u,v) \le \max_{u\in V}\max_{v\in V} d(u,v) = \text{diam } G.$$

Hence, rad $G \le$ diam $G \le 2$ rad G. $\square$

(b) Prove that a graph G of radius at most k and maximum degree at most $d \ge 3$ has fewer than $\frac{d}{d-2}(d-1)^k$ vertices.

Proof. Let $\max_{v\in V} \deg(v) \le d$ for $d \ge 3$. Let $z \in V$ be a central vertex, so $\min_{u\in V}\max_{v\in V} d(u,v) = \max_{u\in V} d(z,u) = \text{rad}(G) \le k$. Let $S_i = \{v \in V \mid d(z,v) = i\}$. Since $d(z,u) \le \text{rad}(G) \le k$ for all $u \in V$, we have that $V = \cup_{i=0}^k S_i$.

Clearly $S_0 = \{z\}$. For $i \ge 1$ and $v \in S_i$, there exists some $w \in S^{i-1}$ such that $(w,v) \in E$. (Since $v \in S_i$, there exists some path $z \rightsquigarrow v_1 \rightsquigarrow \cdots \rightsquigarrow v_{i-1} \rightsquigarrow v$ of length i and this path between z and v has the shortest possible length. Then $z \rightsquigarrow v_1 \rightsquigarrow \cdots \rightsquigarrow v_{i-1}$ is a path of length $i-1$, and this path between z and v_{i-1} has the shortest possible length; if a path $z \overset{p}{\rightsquigarrow} v_{i-1}$ of shorter length existed, then $z \overset{p}{\rightsquigarrow} v_{i-1} \rightsquigarrow v$ would be a path of length less than i from z to v, contradicting that $v \in S_i$. Hence $v_{i-1} \in S_{i-1}$, meaning v will always have a neighbor in S_{i-1}.) By a similar argument, all neighbors of $v \in S_i$ must be contained in S_{i-1}, S_i or S_{i+1}.

Since $\deg(v) \le d$ for any $v \in S_i$, and v must have at least one neighbor in S_{i-1}, v can have at *most* $d-1$ neighbors in S_{i+1}. Furthermore, since all of the vertices in S_{i+1} must be connected to some $v \in S_i$, there are at most $d-1$ vertices in S_{i+1} per vertex in S_i; that is, $|S_{i+1}| \le (d-1)|S_i|$ for $i \ge 1$. For $i = 0$, since $\deg(z) \le d$ and each vertex in S_1 is connected to z, we have that $|S_1| \le d|S_0| = d$. So by finitely many iterations of these inequalities, we have for $i \ge 1$ that

$$|S_i| \le (d-1)|S_{i-1}| \le (d-1)^2|S_{i-2}| \le \cdots \le (d-1)^{i-1}|S_1| \le d(d-1)^{i-1}_.$$

Therefore

$$\begin{aligned}
|V| = |\cup_{i=0}^{k} S_i| = \sum_{i=0}^{k} |S_i| & \qquad \text{[since the } S_i \text{ are disjoint]} \\
\leq |S_0| + \sum_{i=1}^{k} d(d-1)^{i-1} & \qquad \text{[since } |S_i| \leq d(d-1)^{i-1}\text{]} \\
= 1 + d\sum_{j=0}^{k-1} (d-1)^j & \qquad \text{[letting } j = i-1\text{]} \\
= 1 + \frac{d - d(d-1)^k}{1-(d-1)} & \qquad \text{[geometric sum]} \\
= 1 + \frac{d}{d-2}((d-1)^k - 1) & \\
\leq \frac{d}{d-2} + \frac{d}{d-2}((d-1)^k - 1) & \qquad \text{[}d \geq 3 \Rightarrow \frac{d}{d-2} \geq 1\text{]} \\
= \frac{d(d-1)^k}{d-2} &
\end{aligned}$$

which completes the proof. □

3. A random permutation π of the set $\{1, 2, \ldots, n\}$ can be represented by a directed graph on n vertices with a directed arc (i, π_i), where π_i is the ith entry in the permutation. Observe that the resulting graph is just a collection of distinct cycles.

   (a) What is the expected length of the cycle containing vertex 1?

   (b) What is the expected number of cycles?

Proof. (a) There are $n!$ total permutations. Since permutations are bijections, vertex 1 must be contained in some unique cycle; that is, some vertex must be sent to vertex 1 (so vertex 1 is in some cycle) and vertex 1 cannot map/be mapped to by more than one vertex (so vertex 1 is not in more than one cycle). Let X_k be the event that vertex 1 is in a cycle of length k. Note that $1 \leq k \leq n$ since the smallest cycle (of length 1) corresponds to vertex 1 being isolated (vertex 1 is sent to itself) and the largest possible cycle contains all n of the vertices (and thus has length n). Then

$$E(\text{cycle length of the cycle containing vertex 1}) = \sum_{k=1}^{n} P(X_k)k.$$

Vertex 1 will correspond to a cycle of length 1 if the permutation fixes 1. In this case, since the remaining numbers $\{2, \ldots, n\}$ can be permuted in any fashion, there are $(n-1)!$ possible permutations that fix 1. Similarly, for vertex 1 to be in a cycle of length 2, the permutation must send 1 to some $i \neq 1$ (if i = 1 then the cycle has length 1) and then send i back to 1. So in this case, there are $(n-1)$ choices of where to send vertex 1 and only 1 choice of where to send vertex i. The remaining $(n-2)$ vertices can be permuted in any manner, meaning there are $(n-1)(n-2)! = (n-1)!$ possible ways for vertex 1 to be in a cycle of length 2. Continuing in this manner, for vertex 1 to be in a cycle of length k for $1 \leq k \leq n$, there are $(n-1)$ choices of where to send vertex 1, $(n-2)$ choices for where to send that next vertex, $(n-k+1)$ choices for the last vertex in the cycle, and 1 choice for where to send this last vertex (which must be sent back to vertex 1 to make a cycle of length k). The remaining $(n-k)!$ vertices can be permuted in any way, so there are $(n-1)(n-2)\cdots(n-k+1)(n-k)! = (n-1)!$ ways for 1 to be in a cycle of length k. Therefore, we have for all k such that $1 \leq k \leq n$,

$$\begin{aligned} P(X_k) &= \frac{\#\text{ of permutations that put vertex 1 in a cycle of length } k}{\text{total } \#\text{ of permutations}} \\ &= \frac{(n-1)!}{n!} = \frac{1}{n} \end{aligned}$$

which is constant. Therefore, the expected cycle length is

$$\sum_{k=1}^{n} P(X_k)k = \sum_{k=1}^{n} \frac{1}{n}k = \frac{1}{n}\sum_{k=1}^{n} k = \frac{1}{n}\frac{n(n+1)}{2} = \frac{n+1}{2}$$

completing the calculation for (a). □

Proof. (b) First we compute the total possible number of cycles with length k, noting again that $1 \leq k \leq n$. Given a selection of k nodes for a cycle of length k, there are $(n-k)!$ ways to permute the remaining $(n-k)$ nodes. Within the selected k nodes (call them $v_1, \ldots, v_k$), we count the number of ways to produce a k-cycle. There are $(k-1)$ options for where to send v_1 (sending $v_1 \mapsto v_1$ would result in a 1 cycle), $(k-2)$ options for where to send $\pi(v_1)$ (where π denotes the permutation), etc. meaning there are $(k-1)!$ possible k-cycles within these fixed k nodes. Hence, there are $(k-1)!(n-k)!$ total possible permutations that produce a cycle of length k on those fixed k nodes.

Since there are $\binom{n}{k}$ ways to select k nodes out of n total nodes, there are $\binom{n}{k}(k-1)!(n-k)!$ total possible cycles of length k. So out of $n!$ total permutations, the average permutation has

$$\frac{\binom{n}{k}(k-1)!(n-k)!}{n!} = \frac{n!(k-1)!(n-k)!}{k!(n-k)!n!} = \frac{1}{k}$$

cycles of length k. By linearity of expectation,

$$E(\text{total number of cycles}) = \sum_{k=1}^{n} E(\#\text{ of cycles of length } k) = \sum_{k=1}^{n} \frac{1}{k}$$

which is the nth harmonic number $H(n)$. □

4. Let $v_1, v_2, \ldots, v_n$ be unit vectors in $\mathbb{R}^n$. Prove that there exist $\alpha_1, \alpha_2, \ldots, \alpha_n \in \{-1, 1\}$ such that

$$||\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n||_2 \leq \sqrt{n}.$$

Proof. Consider the following algorithm

for $i = 1, \dots, n$ **do**

flip a coin and set $\alpha_i = \begin{cases} 1 & \text{if heads} \\ -1 & \text{if tails.} \end{cases}$

end for

which returns $\alpha_1, \dots, \alpha_n \in \{-1, 1\}$ in finite time with $P(\alpha_i = 1) = P(\alpha_i = -1) = \frac{1}{2}$. Observe that

$$X := ||\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n||_2^2 = \Big\langle \sum_{i=1}^n \alpha_i v_i, \sum_{j=1}^n \alpha_j v_j \Big\rangle = \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j \langle v_i, v_j \rangle$$

since the inner product is linear in each component. Furthermore,

$$\begin{aligned} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j \langle v_i, v_j \rangle &= \sum_{i=1}^n \alpha_i^2 \langle v_i, v_i \rangle + \sum_{i=1}^n \sum_{j \neq i} \alpha_i \alpha_j \langle v_i, v_j \rangle \\ &= \sum_{i=1}^n (1) ||v_i||_2^2 + \sum_{i=1}^n \sum_{j \neq i} \alpha_i \alpha_j \langle v_i, v_j \rangle \qquad [\text{since } \alpha_i^2 = 1] \\ &= n + \sum_{i=1}^n \sum_{j \neq i} \alpha_i \alpha_j \langle v_i, v_j \rangle \qquad [\text{since } ||v_i||_2^2 = 1.] \end{aligned}$$

By linearity of expectation,

$$E(X) = E(n + \sum_{i=1}^n \sum_{j \neq i} \alpha_i \alpha_j \langle v_i, v_j \rangle) = n + \sum_{i=1}^n \sum_{j \neq i} E(\alpha_i \alpha_j) \langle v_i, v_j \rangle.$$

For $i \neq j$ we have $E(\alpha_i \alpha_j) = E(\alpha_i) E(\alpha_j)$ by independence, and $E(\alpha_i) = E(\alpha_j) = \frac{1}{2}(1) + \frac{1}{2}(-1) = 0$. So each term $E(\alpha_i \alpha_j) = 0$ and therefore

$$E(||\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n||_2^2) = E(X) = n + \sum_{i=1}^n \sum_{j \neq i} (0) \langle v_i, v_j \rangle = n.$$

By the probabilistic method, there exist $\alpha_1, \dots, \alpha_n \in \{-1, 1\}$ such that

$$||\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n||_2^2 \leq n,$$

meaning

$$||\alpha_1 v_1 + \alpha_2 v_2 + \ldots + \alpha_n v_n||_2 \leq \sqrt{n}$$

for these $\alpha_1, \dots, \alpha_n \in \{-1, 1\}$. □

5. Consider a graph G on $2n$ vertices where every vertex has degree at least n. Prove that G contains a perfect matching.

Proof. By the lemma below, a (simple) graph on $2n$ vertices (with $2n \geq 3$ or equivalently $n \geq 2$) such that $\deg(v) \geq n$ for all $v \in V$ has a Hamiltonian cycle.

When $n = 1$, $|V| = 2$ and the two vertices each have degree 1 exactly; that is, G is a single edge. Then G trivially contains a perfect matching (the single edge) in this case.

For $n \geq 2$, G contains a Hamiltonian cycle. Thus, there is a cycle $\mathcal{C}$ in G that visits every node exactly once (except for the vertex that is its start/end). Since $\mathcal{C}$ is a cycle on $2n$ vertices, $\mathcal{C}$ has exactly $2n$ edges. Let M be the set of every other edge in $\mathcal{C}$, which is possible since $\mathcal{C}$ has an even number ($2n$) of edges. By construction, no two edges in M can be incident, meaning M is an independent edge set (matching).

Furthermore, every $v \in V$ is contained in $\mathcal{C}$ (definition of Hamiltonian), and every vertex in $\mathcal{C}$ is incident to exactly 2 edges in $\mathcal{C}$. Since M selects every other edge of $\mathcal{C}$, each $v \in \mathcal{C}$ (and hence each $v \in V$) will be incident to exactly one edge in M. Therefore, G contains a perfect matching M when $n \geq 2$, meaning we have shown that G contains a perfect matching in all cases. □

Lemma. Let G be a (simple) graph on $2n$ vertices (with $n \geq 2$) such that $\deg(v) \geq n$ for all $v \in V$. Then G contains a Hamiltonian cycle.

Proof. First note that G is connected. To see this, suppose for contradiction that G has more than one connected component. Pick two of the connected components and label them C_1 and C_2. Since $\deg(v) \geq n$ for all $v \in V$, each $v \in C_i$ needs at least another n vertices to connect to in C_i, meaning $|C_1|, |C_2| \geq n + 1$. Then the total number of vertices is at least $|C_1| + |C_2| \geq 2n + 2$, contradicting that there are $2n$ vertices since $2n < 2n + 2$. Thus, G has exactly one connected component.

Consider the longest path in G and denote it by $P := v_1 \rightsquigarrow \cdots \rightsquigarrow v_k$. Then v_1 cannot have some neighbor w with $w \notin \{v_1, \ldots, v_k\}$, for otherwise P is not the longest path since $w \rightsquigarrow v_1 \rightsquigarrow v_k$ is longer. Since $\deg(v_1) \geq n$, v_1 has at least n neighbors in $\{v_2, \ldots, v_k\}$; that is, there are n total v_i such that $(v_1, v_i) \in E$ with $i \in \{2, \ldots, k\}$. Equivalently, there are n total v_{i+1} such that $(v_1, v_{i+1}) \in E$ with $i \in \{1, \ldots, k-1\}$. By an analogous argument, there are at least n total v_i such that $(v_i, v_k) \in E$ for $i \in \{1, \ldots, k-1\}$. Note that $k \leq 2n$.

Since $\{1, \ldots, k-1\}$ has $k-1 \leq 2n-1 < 2n$ slots, by the Pidgeonhole principle, there is at least one $i \in \{1, \ldots, k-1\}$ that is picked such that both $(v_1, v_{i+1}) \in E$ and $(v_i, v_k) \in E$. Hence, $v_1 \rightsquigarrow v_{i+1} \rightsquigarrow v_{i+2} \rightsquigarrow \cdots \rightsquigarrow v_k \rightsquigarrow v_i \rightsquigarrow v_{i-1} \rightsquigarrow v_1$ is a cycle in G visiting each vertex exactly once (except for v_1), so G contains a Hamiltonian cycle. □

6. Let $G = (V, E)$ be a graph and $w : E \to R^+$ be an assignment of nonnegative weights to its edges. For $u, v \in V$ let $f(u, v)$ denote the weight of a minimum $u - v$ cut in G.

   (a) Let $u, v, w \in V$, and suppose $f(u, v) \leq f(u, w) \leq f(v, w)$. Show that $f(u, v) = f(u, w)$, i.e., the two smaller numbers are equal.

   (b) Show that among the $\binom{n}{2}$ values $f(u, v)$, for all pairs $u, v \in V$, there are at most $n - 1$ distinct values.

Proof. (a) Let $u, v, w \in V$ such that $f(u,v) \leq f(u,w) \leq f(v,w)$. Then $f(u,v) \leq f(u,w)$, so if we show that $f(u,w) \leq f(u,v)$, we will have proved that $f(u,v) = f(u,w)$. By definition, the minimum $u-v$ cut in G is a cut $S \subset V$ such that $u \in S, v \in V \setminus S$ and the sum $\sum_{s \in S, t \in V \setminus S} f(s,t)$ is minimized. Since $f(u,v)$ is the weight of the minimum $u-v$ cut, $f(u,v)$ is the value of the maximum flow that can be routed from u to v. Furthermore, the maximum flow that can be routed from u to v through w is $\min(f(u,w), f(v,w))$ since the maximum flow will be limited by the sub-route that can handle less flow. Then

$$\max(\text{flow from } u \text{ to } v \text{ through } w) \leq \max(\text{flow from } u \text{ to } v)$$

since the former is a specific case of the latter, meaning $\min(f(u,w), f(v,w)) \leq f(u,v)$. Since $f(u,w) \leq f(v,w)$ by assumption, $\min(f(u,w), f(v,w)) = f(u,w) \leq f(u,v)$ as desired. Thus $f(u,v) = f(u,w)$.

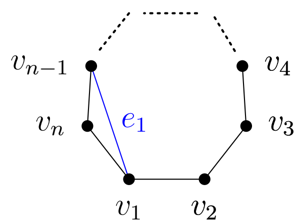
(b) We can represent the $\binom{n}{2}$ possible values of $f(u,v)$ by a complete graph K_n on $|V| = n$ vertices such that an edge (u,v) has nonnegative weight $f(u,v)$. Then the number of distinct values of $f(u,v)$ is equivalent to the number of distinct edge weights in this graph.

Lemma. Any circuit on n edges ($n \geq 3$) in this weighted K_n will have at least two edges with the same weight. (Here a *circuit* is a walk on a graph such that edges are not repeated, vertices may be repeated, and the starting/ending vertices are identical.)

Proof. We proceed by induction over n with $n \geq 3$. For $n = 3$, K_n has only one cycle (itself), and this cycle has three edges; without loss of generality, label them e_1, e_2, e_3 such that $f(e_1) \leq f(e_2) \leq f(e_3)$. Then by part (a), $f(e_1) = f(e_2)$, meaning at least two edges have the same weight. This completes the base case.

For strong induction, assume that any circuit on $k < n$ edges in this weighted K_n will have at least two edges with identical weights. Let $\mathcal{C}$ be an arbitrary circuit in K_n. If $\mathcal{C}$ has a sub-circuit of size k with $3 \leq k < n$, then by the inductive hypothesis, that sub-circuit has repeated edge weights and therefore $\mathcal{C}$ has repeated edge weights. So suppose $\mathcal{C}$ has no sub-circuits of size $k < n$, meaning $\mathcal{C}$ is just a cycle on n edges.

Since $\mathcal{C}$ is contained in K_n, all nodes are connected, so $e_1 := (v_1, v_{n-1})$ is an edge. (Note that e_1 is an edge in K_n but not an edge in $\mathcal{C}$). Then $v_1 \rightsquigarrow v_2 \rightsquigarrow \cdots \rightsquigarrow v_{n-1} \rightsquigarrow v_1$ is a cycle of size $(n-1)$, meaning it has at least two edges of the same weight (call them e_2, e_3) by the inductive hypothesis.



If $e_2, e_3 \neq e_1$, then e_2, e_3 are edges in $\mathcal{C}$ with identical weight (which is the desired conclusion). So suppose either e_2 or e_3 is e_1; that is, the addition of e_1 to the $(n-2)$ previous edges in $\mathcal{C}$ resulted in a redundant edge weight. Without loss of generality, let

$e_2 = e_1$, so $f(e_1) = f(e_2) = f(e_3)$ for edge e_3 in $\mathcal{C}$. Consider the triangle with edges $e_1, (v_{n-1}, v_n)$, and (v_1, v_n). By part (a), we know that the two edges with minimal weight in this triangle will have equal weights. If e_1 has maximum weight in the triangle (so $f(v_1, v_n), f(v_{n-1}, v_n) \leq f(e_1)$), then $f(v_1, v_n) = f(v_{n-1}, v_n)$, and since $(v_1, v_n), (v_{n-1}, v_n)$ are edges in $\mathcal{C}$, we have found at least two edges in $\mathcal{C}$ with equal weight. If e_1 does not have maximum weight in the triangle, then $f(e_1) = f(e_j)$ where e_j is the edge such that $f(e_j) = \min\{f(v_1, v_n), f(v_{n-1}, v_n)\}$. However, since $f(e_1) = f(e_3)$, we have $f(e_j) = f(e_3)$ where e_j, e_3 are two edges in the cycle $\mathcal{C}$. Hence, in all cases we have that $\mathcal{C}$ has edges with redundant edge weights. This completes the inductive step.

Therefore, all circuits on n edges within this weighted K_n have at least two edges of equal weight. □

With this lemma available, suppose for contradiction that there are more than $(n-1)$ distinct values of $f(u, v)$, so there are at least n distinct values of $f(u, v)$ and thus at least n *distinct* edge weights. Since all nodes are connected in K_n and there are only n nodes, any collection of n or more edges will form a circuit on n or more edges (edges may be repeated but vertices are not). So consider the circuit formed by the n or more edges with distinct edge weights. By the lemma this circuit must have at least two edges with the same weight, contradicting that the edge weights are distinct. Therefore, we conclude that there can be at most $(n-1)$ distinct edge weights on this weighted K_n, so there can be at most $(n-1)$ distinct values of $f(u, v)$ over all $\binom{n}{2}$ pairs of $u, v \in V$. □

7. Let T be a spanning tree of a graph G with an edge cost function c. We say that T has the *cycle property* if for any edge $e' \notin T$, $c(e') \geq c(e)$ for all e in the cycle generated by adding e' to T. Also, T has the *cut property* if for any edge $e \in T$, $c(e) \leq c(e')$ for all e' in the cut defined by e. Show that the following three statements are equivalent:

   (a) T has the cycle property.

   (b) T has the cut property.

   (c) T is a minimum cost spanning tree.

   Remark 1: Note that removing $e \in T$ creates two trees with vertex sets V_1 and V_2. A *cut* defined by $e \in T$ is the set of edges of G with one endpoint in V_1 and the other in V_2 (with the exception of e itself).

   Proof. Let T be a spanning tree of G.

   (a) $\Rightarrow$ *(b)* Suppose T has the cycle property. Let $e = (v, w) \in T$ and let $e' = (v', w') \in E$ be an edge in the cut defined by e; that is, $T \setminus e = T_1 \sqcup T_2$ where T_1, T_2 are trees and $v, v' \in T_1, w, w' \in T_2$, and $e' \neq e$. Then $e' \notin T$, for otherwise there would be a cycle in T. (Specifically, suppose $e' \in T$; without loss of generality, let $v' \in T_1$ and $w' \in T_2$. Then there is a path p_1 in T_1 from v to v' and a path p_2 in T_2 from w to w' since trees T_1, T_2 are connected. So $v \overset{p_1}{\rightsquigarrow} v' \overset{e'}{\rightsquigarrow} w' \overset{p_2}{\rightsquigarrow} w \overset{e}{\rightsquigarrow} v$ is a cycle in T, contradicting that trees are acyclic.) Since $e' \notin T$, and trees are *maximally acyclic*, $T \cup e'$ forms a cycle containing

e'; in particular, we have the cycle described previously: $v \overset{p_1}{\rightsquigarrow} v' \overset{e'}{\rightsquigarrow} w' \overset{p_2}{\rightsquigarrow} w \overset{e}{\rightsquigarrow} v$. Since T has the cycle property, $e' \notin T$, and e is in the cycle formed by adding e', $c(e) \leq c(e')$. Since e' was an arbitrary edge in the cut defined by e, we conclude that T has the cut property.

$(b) \Leftarrow (a)$ Suppose T has the cut property. Let $e' = (v', w') \notin T$. (If no such e' exists in G, then T trivially satisfies the cycle property.) Consider the cycle generated by adding e' to T, and let $e = (v, w)$ be an arbitrary edge in this cycle with $e \neq e'$. Let $T \setminus e = T_1 \sqcup T_2$; without loss of generality, let $v \in T_1, w \in T_2$. Then the cut defined by e is the set of edges between T_1 and T_2 (not including e). We show that e' must be in the cut defined by e.

Note that T_1, T_2 provide a partition of the vertices. If $v' \in T_1, w' \in T_2$ (or $v' \in T_2, w' \in T_1$) then e' is in the cut defined by e. Suppose $v', w' \in T_1$ (so e' is contained in T_1). Then $T_1 \sqcup T_2 \cup e' = (T_1 \cup e') \sqcup T_2 = T_1 \sqcup T_2$. Also, $T \cup e' \setminus e = (T \setminus e) \cup e' = T_1 \sqcup T_2 \cup e'$, meaning $T \cup e' \setminus e = T_1 \sqcup T_2$. However, by the lemma proved after this problem, $T \cup e' \setminus e$ is connected (because $T \cup e'$ is a connected graph containing a cycle with e, so we are just removing an edge from a cycle). Then the equality $T \cup e' \setminus e = T_1 \sqcup T_2$ implies that $T_1 \sqcup T_2$ is connected. This is a contradiction. The case for $v', w' \in T_2$ is symmetric. Hence, e' must be an edge across the cut defined by e. So any edge e in the cycle generated by adding e' to T has e' in its cut. By the cut property, $c(e) \leq c(e')$. Hence, T has the cycle property.

$(a) \Rightarrow (c)$ Let T have the cycle property. Consider *Kruskal's algorithm*, which always outputs a minimum cost spanning tree when given G.

```
T = {}
label edges e_1, ..., e_m such that c(e_1) ≤ ··· ≤ c(e_m)
for i = 1, ..., m do
  if T ∪ {e_i} does not have a cycle then
    T = T ∪ {e_i}
  end if
end for
```

Suppose for contradiction T is not a MST. Let T' be a MST outputted by Kruskal's algorithm. Then $T \neq T'$, so T' has some edges that do not belong to T. Let e' be the first edge selected by Kruskal's algorithm such that $e' \notin T$. Since trees are maximally acyclic, $T \cup e'$ has a cycle containing e'. Since $e' \notin T$ and T has the cycle property, $c(e') \geq c(e)$ for all edges e in this cycle, $e \neq e'$. Since Kruskal's algorithm visits all m edges in G in order of ascending weight and $c(e) \leq c(e')$, every $e \neq e'$ in the cycle was already evaluated by Kruskal's algorithm. Since e' was the *first* edge added to T' that was not an edge of T and each $e \neq e'$ was added before e', we conclude that all $e \neq e'$ in this cycle were members of T and T'. However, then e' would not have been added to T' in the algorithm, since all the other e in the cycle were edges in T' and adding e' to these edges created a cycle. This is a contradiction. Hence, $T = T'$, where T' is some MST of G, meaning T is a minimum cost spanning tree.

$(c) \Rightarrow (a)$ Let T be a minimum cost spanning tree. Suppose for contradiction that T does not have the cycle property. Then for some $e' \notin T$, there is an edge e in the cycle generated by adding e' to T such that $c(e') < c(e)$. Consider $T' = T \cup e' \setminus e$, which is also a spanning tree. (T' is connected by the lemma after this problem since $T \cup e'$ is connected and has a cycle with e, so we are just removing an edge from a cycle. T' is acyclic since $T \cup e'$ has only one cycle (since T is acyclic) and we removed an edge from that cycle. Also, $T' = T \setminus e \cup e'$ is a spanning tree since it is a tree still containing n vertices.)

Since $c(e') < c(e)$ and all other edges in T and T' are the same (and thus have the same weight), we have that T' is a lower cost spanning tree than T, which contradicts minimality. Thus, T must have the cycle property. □

Lemma. Let $G(V, E)$ be connected and let $\mathcal{C}$ be a cycle in G. Let $e \in \mathcal{C}$ be an edge in the cycle. Then $G(V, E \setminus e)$ is still connected.

Proof. Let $a, b \in V$; since G is connected, there exists a path $a \rightsquigarrow b$ in G. For $e = (v, w)$ in $\mathcal{C}$, there exists a path $v \rightsquigarrow w$ in $\mathcal{C}$ that does *not* use e (by going around the rest of the cycle). So if any portion of $a \rightsquigarrow b$ uses edge $e = (v, w)$, that portion can be replaced by the path $v \rightsquigarrow w$ in $E \setminus e$. Hence, any two vertices $a, b \in V$ can still be connected by a path in $G(V, E \setminus e)$. □

8. Given a graph $G = (V, E)$, a set of vertices $D \subseteq V$ is called a *dominating set* if every vertex in $V \setminus D$ is adjacent to a vertex in D. Suppose $|V| = n$ and the minimum degree of $G = \delta > 0$. Show that G contains a dominating set of size at most:

$$\frac{n(\log(1+\delta)+1)}{1+\delta}$$

Proof. Let $N(v)$ denote the neighborhood of $v \in V$, so $N(v) = \{w \in V \mid (v, w) \in E\}$. Consider the following algorithm:

$D = \emptyset$
for $v \in V$ **do**
  with some fixed probability $0 < p < 1$:
    $D = D \cup \{v\}$
end for
for $v \in V \setminus D$ **do**
  if $N(v) \cap D = \emptyset$ **then**
    $D = D \cup \{v\}$
  end if
end for

which adds vertices to D with probability p and then adds all of the vertices still "uncovered" by D into D. The algorithm terminates since V has finitely many vertices. Furthermore, D is not dominating $\Leftrightarrow$ there exists some $v \notin D$ with $N(v) \cap D = \emptyset$, which is impossible since the second *for* loop adds such vertices into D. Hence, D is dominating by construction.

Let D_1, D_2 be the vertices added to D in the first and second *for* loops respectively (so $D = D_1 \sqcup D_2$). Then $E(|v \text{ added to } D_1|) = np$ since there are n Bernoulli(p) trials. Furthermore, the probability that some $v \in V$ is left "uncovered" by D_1 is

$$P(v \text{ and } N(v) \text{ have no elements in } D_1).$$

Since $\min_{v \in V} \deg(v) = \delta$, $|N(v)| \geq \delta$ and so $|v \cup N(v)| \geq \delta + 1$. Each vertex in $v \cup N(v)$ is *not* included in D_1 with probability $(1-p)$, so $v \cup N(v)$ has no elements in D_1 with probability at most $(1-p)^{\delta+1}$ by independence. Hence

$$\begin{aligned} P(v \in D_2) &= P(v \text{ uncovered by } D_1) = P(v \text{ and } N(v) \text{ have no elements in } D_1) \\ &\leq (1-p)^{\delta+1} \end{aligned}$$

and by introducing indicator variables

$$E(|D_2|) = E(\sum_{v \in V} \mathbb{1}_{v \in D_2}) = \sum_{v \in V} P(v \in D_2) \leq n(1-p)^{\delta+1}.$$

So we compute

$$E(|D|) = E(|D_1| + |D_2|) = E(|D_1|) + E(|D_2|) \leq np + n(1-p)^{\delta+1}.$$

Note that generally $1 - p \leq e^{-p}$ on $[0,1]$ since $1 - p = e^{-p}$ at $p = 0$ and $\frac{d}{dp}(1-p) = -1 \leq -\frac{1}{e^p} = \frac{d}{dp}(e^{-p})$ on $[0,1]$. Also $(1-p) \leq e^{-p} \Rightarrow (1-p)^{\delta+1} \leq e^{-p(\delta+1)}$. Therefore

$$E(|D|) \leq np + n(1-p)^{\delta+1} \leq n(p + e^{-p(\delta+1)}).$$

We now optimize $0 < p < 1$ by selecting $p = \frac{\log(1+\delta)}{1+\delta}$ so that $e^{-p(\delta+1)} = e^{-\log(1+\delta)} = \frac{1}{e^{\log(1+\delta)}} = \frac{1}{1+\delta}$. Then

$$E(|D|) \leq \frac{n\log(1+\delta)}{1+\delta} + \frac{n}{1+\delta} = \frac{n(\log(1+\delta)+1)}{1+\delta}.$$

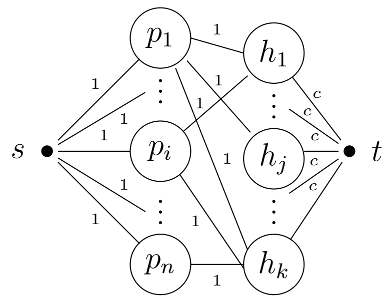
If all dominating sets D of G had size $|D| > \frac{n(\log(1+\delta)+1)}{1+\delta}$, then $E(|D|) > \frac{n(\log(1+\delta)+1)}{1+\delta}$ (which is a contradiction). Hence, not all dominating sets satisfy $|D| > \frac{n(\log(1+\delta)+1)}{1+\delta}$, meaning there exists some dominating set D of G with size at most $\frac{n(\log(1+\delta)+1)}{1+\delta}$. □

9. Consider the following scenario. Due to large-scale flooding in a region, paramedics have identified a set of n injured people distributed across the region who need to be rushed to hospitals. There are k hospitals in the region, and each of the n people needs to be brought to a hospital that is within a half-hour's driving time of their current location (so different people will have different options for hospitals, depending on where they are right now).

   At the same time, one doesn't want to overload any one of the hospitals by sending too many patients its way. The paramedics are in touch by cell phone, and they want to collectively work out whether they can choose a hospital for each of the injured people in such a way that the load on the hospitals is *balanced*: Each hospital receives at most $\lceil n/k \rceil$ people.

   Create a polynomial time algorithm that outputs an assignment of people to hospitals if a valid assignment exists and outputs no otherwise.

Proof. Consider the following flow diagram. Directed arrows are omitted since we assume flow is from left to right. The source s has n edges connected to it, with each edge connected to a single person p_i. The capacity on each of these n edges is 1 so that each p_i corresponds to a single patient. The $h_1, \ldots, h_k$ denote the k hospitals. Each p_i connects to some number of the $\{h_j\}_{j=1}^k$ via edges, where an edge (p_i, h_j) indicates that patient i can be sent to hospital j within half an hour.



The edges between the patients $\{p_i\}_{i=1}^n$ and the hospitals $\{h_j\}_{j=1}^k$ each have capacity 1 so that an edge represents sending a single patient to a hospital. Then we let $c := \lceil n/k \rceil$ and note that this value is constant given fixed (but arbitrary) n and k. Each hospital h_j then has a single edge connecting it to t with capacity c so that the flow through a hospital is capped at $\lceil n/k \rceil$, enforcing the constraint that each hospital cannot handle more than $\lceil n/k \rceil$ patients.

The units of flow then corresponds to the number of patients that can feasibly be assigned to a hospital given the constraints on location and the number of patients a hospital can handle. The maximum flow is the largest number of patients that can be assigned to the k hospitals with these constraints in place, and the max-flow route corresponds to how that assignment is made.

We run Ford-Fulkerson to compute the max-flow f. If $f \geq n$, then all n people can be assigned to a hospital; that is, a valid assignment of people to hospitals exists. If $f < n$ we know immediately that such an assignment is impossible.

In this case, Ford-Fulkerson is running on $O(n+k)$ nodes and $O(nk)$ edges. (In the worst-case, all n patients can be sent to all k hospitals so there are nk edges in the middle and $k + n$ edges on the sides, which is $O(nk)$ edges total.) Generally Ford-Fulkerson has a runtime of $O(nm)$, so the runtime in this problem is $O((n + k)nk) = O(n^2k^2)$, which is polynomial time. Therefore, Ford-Fulkerson on this flow network is a polynomial time algorithm that outputs an assignment of people to hospitals if a valid assignment exists and notifies us (when $f < n$) if no such assignment is feasible. □