Chapter 13: Network Flows and Applications

Network: directed graph with source S and target T. Non-negative edge weights represent capacities.

Assume no edges into S or out of T. (If necessary, we can pull back to S' and extend to T' to accomplish this. Use sum of capacities out of S and into T to assign capacities to new edges.)

Network flow: assignment of non-negative numbers to edges respecting capacities and satisfying conservation of flow.

If P and Q are sets of vertices, flow(P, Q) = sum of flow values along all (p, q) edges.

Definition: A vertex cut (P, Q) is a partition of V with $S \in P$, $T \in Q$.

Lemma: For any vertex cut (P, Q) we have

$$\text{flow}(P, Q) - \text{flow}(Q, P) = \text{flow}(S, V) = \text{flow}(V, T).$$

Proof: We have

$$\text{flow}(P, Q) = \text{flow }(P, V) - \text{flow}(P, P)$$

and

$$\text{flow}(Q, P) = \text{flow}(V, P) - \text{flow}(P, P),$$

therefore

$$\text{flow}(P, Q) - \text{flow}(Q, P) = \text{flow}(P, V) - \text{flow}(V, P) = \text{flow}(S, V).$$

Some more detail:

$$\text{flow}(P, V) = \text{flow}(S, V) + \sum_{p \in P \setminus \{S\}} \text{flow}(p, V)$$

and

$$\text{flow}(V, P) = \sum_{p \in P \setminus \{S\}} \text{flow}(V, p) = \sum_{p \in P \setminus \{S\}} \text{flow}(p, V)$$

by conservation of flow, hence the cancelation. Similarly, we have

$$\text{flow}(P, Q) = \text{flow }(V, Q) - \text{flow}(Q, Q)$$

and

$$\text{flow}(Q,P) = \text{flow}(Q,V) - \text{flow}(Q,Q),$$

therefore

$$\text{flow}(P,Q) - \text{flow}(Q,P) = \text{flow}(V,Q) - \text{flow}(Q,V) = \text{flow}(V,T).$$

In both cases we used conservation of flow and the assumption of no flow into S or out of T.

Goal: find f maximizing flow(S,V).

Definition: capacity(P,Q) = sum of capacities of all (p,q) edges.

Theorem: let P and Q be any partition of vertices in which $S \in P$ and $T \in Q$. Then flow$(S,V) \leq$ capacity(P,Q).

Proof:

$$\text{flow}(S,V) = \text{flow}(P,Q) - \ \text{flow}(Q,P) \leq \text{flow}(P,Q) \leq \ \text{capacity}(P,Q).$$

Corollary: If flow(S,V) = capacity(P,Q) for some cut (P,Q) then flow(S,V) is maximal and capacity(P,Q) is minimal.

Corollary: To prove that a given flow assignment f has maximal flow(S,V), all we have to do is find a cut (P,Q) in which the (p,q) edges have flow equal to capacity and the (q,p) edges have flow equal to zero. In this case we have

$$\text{flow}(S,V) = \text{flow}(P,Q) - \text{flow}(Q,P) = \text{capacity}(P,Q).$$

Given an existing flow, here is a method of augmenting it:

Find pseudo-path from S to T.

Define slack along forward edges e as capacity$(e)-$flow(e). Define slack along backward edges as flow(e). Slack values are all non-negative. Let δ be the smallest slack value value. If $\delta > 0$, add δ to the forward edges and subtract δ from the backward edges. This preserves conservation of flow along the path and increases flow(S,V) without exceeding capacities or creating negative flow values.

An algorithm for augmenting a flow to a maximal flow: turning the crank.

Here is one turn of the crank (a.k.a. the flow augmentation algorithm): Initialize $P = \{S\}$ and $Q = V - \{S\}$. (Labelled and unlabelled vertices.) Label S with $(*, \infty)$. The first coordinate is predecessor, the second coordinate is used to compute min slack.

Continue to do the following: examine every frontier edge pq. Classify as forward or backwards edge, and calculate the slack δ_{pq}. If $\delta_{pq} = 0$, move on to the next frontier edge. If $\delta_{pq} > 0$, do the following: assuming that the label of p is $(predecessor, minslack)$, then create the following label for q: $(p, \min(minslack(p), \delta_{pq}))$. Now add q to P and delete q from Q. If $T \in P$, stop. Otherwise, keep on going.

There are two possible stopping conditions when you turn the crank:

Stop Condition 1: $\delta_{pq} = 0$ along all frontier edges at some point in the algorithm.

Stop Condition 2: T gets added to P.

Theorem: If we achieve Stop Condition 1, then the current flow assignment is maximal. But if we achieve Stop Condition 2, then we can strictly augment the flow.

Proof: In Stop Condition 1 we have identified a cut (P, Q) in which flow = capacity along (p, q) edges and flow = 0 along (q, p) edges. This implies maximal flow and minimal cut. In Stop Condition 2 we can identify a pseudo-path from S to T using predecessor vertices, and the second coordinate of the T-label calculates the minimum slack along this pseudo-path. This minimum slack value will be positive, and we can use it to strictly augment the existing flow.

Corollary: If we keep turning the crank this way, we will eventually find a maximal flow and a minimal cut.

Proof: We can't keep increasing flow(S, V) indefinitely. So at some point we achieve Stop Condition 1.

Section 13.3: Flows and Connectivity

Theorem 13.3.5: Let s and t be distinct vertices in a digraph D. Then the maximum number of edge-disjoint st paths is equal to the minimum number of edges in an st separating set.

Proof: It is clear that any st separating set of edges must have at least as many edges as there are edge-disjoint paths between s and t. So we must have to show that the minimum number of edges is less than or equal to the maximum number of paths.

Create a network N from D by adding a vertex S with in-degree 0 an a single edge into s and a vertex T with out-degree 0 and a single edge into it from t. Assign capacity 1 to all pre-existing edges, and capacity ∞ to the two new edges. Find a flow in this network with $\text{flow}(S, V) = \text{capacity}(P, Q)$. If we remove all (p, q) edges then there can be no surviving ST paths, hence no surviving st paths. Therefore the minimum st edge separating set has less than or equal to the number of such edges, which equal to the capacity of (P, Q), which is equal to $\text{flow}(S, V)$. On the other hand, the flow defines a set of edge-disjoint st paths, so the maximal number of them is greater than or equal to $\text{flow}(S, V)$. QED.

Theorem 13.3.9: Let s and t be distinct vertices in a simple graph G. Then the maximum number of edge-disjoint st paths is equal to the minimum number of edges in an st separating set.

Proof: It is clear that any st separating set of edges must have at least as many edges as there are edge-disjoint paths between s and t. So we must have to show that the minimum number of edges is less than or equal to the maximum number of paths.

Create a network N from G as follows: Add two new vertices S and T. Create a directed edge $S \to s$ with capacity ∞ and a directed edge $t \to T$ with capacity ∞. For every edge pq in G, create a directed edge $p \to q$ and a directed edge $q \to p$, capacities equal to 1. Find a flow in this network with $\text{flow}(S, V) = \text{capacity}(P, Q)$. If we remove all pq edges from G, we will separate s from t in G. Moreover, the number of pq edges in G is equal to the capacity of (P, Q) in N. Hence the minimum number of edges in an st-separating set of G is less than or equal to $\text{capacity}(P, Q)$. On the other hand, the capacity of (P, Q) is equal to the flow, and this flow defines edge-disjoint paths from s to t in N. These do not, however, define edge-disjoint st path in G, so we need to be careful here. The lemma below shows that we can use the edge-disjoint st paths in N to define non-overlapping edge-disjoint st paths in N, which correspond to edge-disjoint st paths in G. So the minimum number of edges in an st-separating set of edges in G is less than or equal to the maximum number of edge-disjoint st paths in G.

Lemma: Every collection of k edge-disjoint st paths in N gives rise to a collection of k non-overlapping edge-disjoint paths in N.

Proof: We have a collection of edges defining k paths. We will reduce the total number of overlapping edges to zero by deleting some edges and redistributing the rest. Find two intersecting paths P and Q, and let x be the first vertex encountered on path Q on an overlapping edge, and let y be the first vertex encountered on Q after leaving a consecutive block of overlapping edges. Modify the two paths in question as follows: Take P from s to y, then take Q from y to t. Also, take Q from s to x, then take P from x to t. We have discarded the shared edges joining x to y. Now keep on going until no more overlapping edges.

Notation: $\kappa_e(s,t)$ is the minimum number of edges whose removal separates s and t in a simple graph.

Lemma: Let G be a simple graph and $s \neq t$ in G. Then $\kappa_e(s,t)$ is equal to the maximum flow(s,V) in the network obtained by adding $S \to s$, $t \to T$, and introducing the usual capacities of 1 and ∞.

Proof: We must have $\kappa_e(s,t) \geq$ the maximum number of edge-disjoint st paths, hence $\kappa_e(s,t) \geq$ the minimum size of an st cut, hence $\kappa_e(s,t) \geq$ the minimum capacity of an st cut, hence $\kappa_e(s,t) \geq$ the maximum flow. Note that if f produces maximum flow and generates minimal cut (P,Q), then removing all the (p,q) edges separates s and t, hence there is an st-separating set of edges of size capacity(P,Q), hence size flow(S,V), hence $\kappa_e(s,t) \leq$ flow(S,V). So these two values are equal.

Corollary 13.3.12: Let s be an arbitrary vertex in a simple graph G. Then

$$\kappa_e(G) = \min_{t \in V_G - \{s\}} \{\text{flow}_{st}(s,V)\}$$

where flow$_{st}(s,V)$ is maximum flow in a network with $S \to s$, $t \to T$, and the usual capacities.

Proof: It suffices to show that

$$\kappa_e(G) = \min_{t \in V_G - \{s\}} \{\kappa_e(s,t)\}.$$

Let $t \neq s$ be given. Then removing $\kappa_e(s,t)$ edges separates s from t, hence separates G, hence

$$\kappa_e(G) \leq \kappa_e(s,t).$$

This proves that

$$\kappa_e(G) \le \min_{t \in V_G - \{s\}} \{\kappa_e(s,t)\}.$$

Suppose it is possible to disconnect G by removing $j < \min_{t\in V_G-\{s\}}\{\kappa_e(s,t)\}$ vertices. Then removing these j vertices separates s from some other vertex t_0, hence $\kappa_e(s,t_0) \le j < \kappa_e(s,t_0)$. Contradiction. Therefore we cannot have strict inequality.

Consequence: We can calculate $\kappa_e(G)$ algorithmically using this – see Algorithm 13.3.1.

Using Network Flows to Prove the Vertex Forms of Menger's Theorem

The idea is to force internally disjoint paths. This can be done by introducing bottleneck edges with capacity 1. Let S and T be source and target. After every non-ST vertex x introduce x_B and $x \to X_B$. Replace every edge xy by $x_B \to y$. Now prove the vertex forms of Menger's Theorems in the same way we proved the edge forms (leave as exercise). Also, we can develop an algorithm for computing $\kappa_v(G)$ by computing the max flow in the network defined by two non-adjacent vertices and then computing the minimum of these values (leave as exercise).

Section 13.4: Matchings, Transversals, and Vertex Covers

Matchings

Let G be a graph. A matching in a graph is a subset of edges which do not share endpoints.

Maximum Matching in a bipartite graph: a matching with maximal number of edges.

Application: see Application 13.4.1, page 561.

Theorem 1: Given a bipartite graph $G(X,Y)$, form network by adding source S and target T, directing edges from S to X, X to Y, and Y to T, and setting $k(e) = 1$ in all cases. Then there is a one-to-one correspondence between matchings and flow assignments, and the size of a matching is equal to the value of the corresponding flow.

Proof: Let M be a matching in $G(X,Y)$. Assign a flow value of 1 to all corresponding edges in the network. Assign flows out of S and into T

accordingly to preserve conservation of flow. Get flow with value equal to $|M|$. Conversely, given a flow assignment, the $G(A, B)$ edges with positive flow cannot share endpoints, so gives rise to a matching. Hence the one-to-one correspondence.

Transversals

Transversal of a collection of sets $\{A_1, \ldots, A_r\}$: a vector $(a_1, \ldots, a_r)$ such that $a_i \in A_i$ for each i and $a_i \neq a_j$ whenever $i \neq j$. Also called a SDR (system of distinct representatives).

Application: see Application 13.4.3, page 564.

X-saturated matching in a bipartite graph $G(X, Y)$: a matching M in which every vertex in X belongs to an edge in M.

Theorem 2: There is a one-to-one correspondence between transversals of $\{A_1, \ldots, A_r\}$ and X-saturated matchings in the bipartite graph $G(X, Y)$, where $X = \{A_l, \ldots, A_r\}$, $Y = A_1 \cup \cdots \cup A_r$, and there is an edge between $x \in X$ and $y \in Y$ iff $y \in x$.

Proof: Transversals correspond to matching edges.

Method for finding a transversal: form $G(X, Y)$ as in the theorem, then find a maximal flow with flow value $|X|$, if possible. Use the method outlined in Theorem 1 to construct the corresponding matching, and used the correspondence described in Theorem 2 to construct the transversal.

Theorem 3 (Hall's Theorem): Let $G(X, Y)$ be a bipartite graph. Then $G(X, Y)$ has an X-saturated matching if and only if for each $W \subseteq X$ there are at least $|W|$ vertices incident to W in $G(X, Y)$, i.e. $|N(W)| \geq |W|$.

Proof: Assume $G(X, Y)$ has an X-saturated matching M. Given $W \subseteq X$, every vertex of W lies in a different edge of M, so there have to be at least $|W|$ neighbors of W in $G(X, Y)$.

Conversely, suppose for each $W \subseteq X$ there are at least $|W|$ neighbors of W in G. Form the network as in Theorem 1. We will show that the minimal cut-capacity of the network is $|X|$, which implies that there is a flow f with $|f| = |X|$, which can be used to construct a matching of size X which is therefore X-saturated.

First, note that the capacity of the cut $(\{S\}, V \backslash \{S\})$ is $|X|$, so the minimal capacity is $\leq |X|$. Now let (P, Q) be a minimal vertex cut. To finish the proof, we will show that capacity$(P, Q) \geq |X|$, hence capacity$(P, Q) = |X|$. Write $P = \{S\} \cup P_0$ and $Q = \{T\} \cup Q_0$. The capacity of (P, Q) is the number of edges from S into Q_0 plus the number of edges from P_0 into Q_0 plus the number of edges from P_0 into T, or $\langle S, Q_0\rangle + \langle P_0, Q_0\rangle + \langle P_0, T\rangle$ for short. This is clearly equal to

$$\langle S, Q_0 \cap X\rangle + \langle P_0 \cap X, Q_0 \cap Y\rangle + \langle P_0 \cap Y, T\rangle.$$

Some simplifications:

1.
$$\langle S, Q_0 \cap X\rangle = |Q_0 \cap X|.$$

2.
$$\langle P_0 \cap X, Q_0 \cap Y\rangle \geq |Q_0 \cap N(P_0 \cap X)| = |N(P_0 \cap X)| - |P_0 \cap N(P_0 \cap X)| \geq$$
$$|N(P_0 \cap X)| - |P_0 \cap Y| \geq |P_0 \cap X| - |P_0 \cap Y|.$$

3.
$$\langle P_0 \cap Y, T\rangle = |P_0 \cap Y|.$$

Therefore
$$\text{capacity}(P, Q) \geq |Q_0 \cap X| + |P_0 \cap X| = |X|.$$

Graph Factorizations

A factor of a graph is a subgraph that includes all vertices in its edges. A k-regular graph is a graph in which every vertex has degree k. A k-factorization of a graph $G = (V, E)$ is a partition of E into k-regular 1-factors of G. For example, a 1-factorization of the complete bipartite graph $K(\{1, 2, 3\}, \{4, 5, 6\})$ is $\{E_1, E_2, E_3\}$, where $E_1 = \{\{1, 4\}, \{2, 5\}, \{3, 6\}\}$, $E_2 = \{\{1, 5\}, \{2, 6\}, \{3, 4\}\}$, $E_3 = \{\{1, 6\}, \{2, 4\}, \{3, 5\}\}$.

Theorem 4 (König's 1-Factorization Theorem): Every k-regular bipartite graph $G(X, Y)$ is 1-factorable when $k \geq 1$.

Proof: By induction on k. When $k = 1$, $G(X, Y)$ is a collection of vertex-disjoint edges and is its own 1-factor. Now assume every k-regular bipartite graph $G(X, Y)$ has a partition into k 1-factors. Let $G(X, Y)$ be a $(k + 1)$-regular bipartite graph. We will show that $G(X, Y)$ has an X-saturated matching M and that $|X| = |Y|$. Then M is a 1-factor and

$$H(X, Y) = G(X, Y) - M$$

is a k-regular bipartite graph. By the induction hypothesis, the edges of $H(X, Y)$ can be partitioned into k 1-factors, which implies that the edges of $G(X, Y)$ can be partitioned into $k + 1$ 1-factors.

We will prove that $G(X, Y)$ has an X-saturated matching M using Hall's theorem. Let $W \subseteq X$ be given. Consider the collection of edges from W to $N(W)$. Keeping track of the endpoints in W, there are $k|W|$ edges from W to $N(W)$. Keeping track of the endpoints in $N(W)$, there are $\leq k|N(W)|$ edges from W to $N(W)$. Therefore $k|W| \leq k|N(W)|$, which implies that $|W| \leq |N(W)|$. Hence by Hall's theorem there is an X-saturated matching M. This of course implies that $|X| \leq |Y|$ (or we could just say that $|X| \leq |N(X)| \leq |Y|$). A similar argument shows that $|Y| \leq |X|$. Therefore $|X| = |Y|$ and we have our X-saturated matching M. //

Theorem 5 (Petersen's 2-Factorization Theorem): Every $2k$-regular graph G is 2-factorable when $k \geq 1$.

Proof: Write $G = (V, E)$ where $V = \{v_1, \ldots, v_n\}$. Without loss of generality G is connected. Then it has a closed Euler trail C which traverses each edge of G exactly once. Regard this as sequence of directed edges $e_1, e_2, \ldots, e_r$. Write $e_i = v_{\alpha_i} \to v_{\beta_i}$. The tail of e_i is v_{α_i} and the head of e_i is v_{β_j}. Then each vertex of G appears as tail vertex exactly k times and as a head vertex exactly k times. Let $H(X, Y)$ be the bipartite graph with vertex partition $X = \{x_1, \ldots, x_n\}$ and $Y = \{y_1, \ldots, y_n\}$, where there is an edge $\{x_i, y_j\}$ in $H(X, Y)$ iff there is an edge $v_i \to v_j$ in C. Then $H(X, Y)$ is k-regular and has a factorization into k 1-factors. Consider the 1-factor $\{x_1, y_{\sigma(1)}\}, \ldots, \{x_n, y_{\sigma(n)}\}$, where σ is a permutation of $\{1, \ldots, n\}$. Then in C we have, for any $j \leq n$, the directed cycle $v_j \to v_{\sigma(j)} \to v_{\sigma^2(j)} \to \cdots \to v_j$. Therefore the edges in G corresponding to the edges in $\{x_1, y_{\sigma(1)}\}, \ldots, \{x_n, y_{\sigma(n)}\}$ can be partitioned into distinct cycles and form a 2-factor of G. The 2-factors of G generated in this way do not share edges since C traverses each edge of G exactly once, and every edge of G appears in a 2-factor since it appears in C, therefore the 1-factorization of $H(X, Y)$ induces a 2-factorization of G. //

Vertex Covers

A vertex cover of a graph G is a set of vertices C such that every edge of G has at least one endpoint in C. In other words, every edge of G can be "seen" by every vertex in C. A minimum vertex cover is a cover of minimum size.

Lemma: Let M be any matching (collection of vertex-disjoint edges) in G, and let C be any cover of C. Then $|M| \leq |C|$.

Proof: Every edge of M has an endpoint in C. Let M_1 be the edges which have 1 endpoint in C, and let M_2 be the edges which have 2 endpoints in C. Since the edges of M do not share vertices, the total number of vertices in C which appear as endpoints in M is equal to $|M_1| + 2|M_2|$. Therefore

$$|C| \geq |M_1| + 2|M_2| \geq |M_1| + |M_2| = |M|.$$

Corollary: Whenever $|M| = |C|$ in G, M is a maximum matching and C is a minimum vertex cover.

Proof: Let M_1 be maximum matching and let C_1 be a minimum cover. Then

$$|M| \leq |M_1| \leq |C_1| \leq |C|.$$

Since $|M| = |C|$, this implies $|M_1| = |M|$ and $|C_1| = |C|$. Therefore M is maximum and C is minimum.

Theorem 6 (König 1931): Let $G(X, Y)$ be a bipartite graph. Then $|M| = |C|$ is attainable.

Proof: The proof in the textbook is not correct, to put it mildly. Here is a correct proof: Let $C = C_X \cup C_Y$ be a minimum size vertex cover, where $C_X \subseteq X$ and $C_Y \subseteq Y$. Let $A = G(C_X, Y - C_Y)$ denote the subgraph of $G(X, Y)$ consisting of edges from C_X to $Y - C_Y$, and let $B = G(X - C_X, Y)$ denote the subgraph of $G(X, Y)$ consisting of edges from $X - C_X$ to Y. If we can find a C_X-saturated matching of A and a C_Y-saturated matching of B, then the union of these will be a matching of $G(X, Y)$ of size $|C_X \cup C_Y|$. We need only verify $|W| \leq |N_A(W)|$ for all $W \subseteq C_X$ and $|W| \leq |N_B(W)|$ for all $W \subseteq C_Y$, then invoke Hall's Theorem. By symmetry it will suffice to prove the former condition only.

Consider $W \subseteq C_X$. We claim that $C' = (C_X \backslash W) \cup N_A(W) \cup C_Y$ is a vertex cover of $G(X, Y)$. To see this, let $\{x, y\}$ be an arbitrary edge in $G(X, Y)$. If

$x \in C_X \backslash W$ or $y \in N_A(W)$ or $y \in C_Y$ then $\{x, y\}$ has an endpoint in C'. It is not possible to have $x \in W$ and $y \notin N_A(W)$ and $y \notin C_Y$ by definition of $N_A(W)$. It is also not possible to have $x \notin C_X$ and $y \notin N_A(W)$ and $y \notin C_Y$ because $C_X \cup C_Y$ is a cover of $G(X, Y)$. This exhausts the possibilities. So C' is a cover of $G(X, Y)$. As such, $|C'| \geq |C|$ by minimality of C. This implies $|N_A(W)| \geq |W|$. So we are done.

0-1 Matrices and the König-Egerváry Theorem

Theorem 7 (König-Egerváry 1931): Let A be a 0-1 matrix. Then the maximum number of 1s in A, no two of which lie in the same row or column, is equal to the minimum of rows and columns that together contain all the 1s in A.

Proof: We need a bipartite graph model in which matchings correspond to 1s which do not share rows or columns and in which covers correspond to rows and columns that see all 1s. The rows and columns seem to be acting as vertices, and it is natural to form a bipartite graph $G(\{r_1, \ldots, r_j\}, \{c_1, \ldots, c_k\})$ with an edge from r_i to c_j if and only if $a_{ij} = 1$. A matching corresponds to a collection of 1s, no two of which lie in the same row or column. A vertex cover corresponds to a collection of rows and columns that see all the 1s. Now apply Theorem 6.

Application: See the application to the Bottleneck Problem, Application 13.3.4, page 569.