Network Flow and Matching

Edmonds-Karp Analysis

Recall: Edmonds-Karp is an efficient implementation of the Ford-Fulkerson method which selects shortest augmenting paths in the residual graph. It assigns a weight of 1 to every edge and runs BFS to find a breadth-first shortest path from s to t in G_f.

Monotonicity Lemma

Lemma. Let $\delta(v) = \delta_f(s, v)$ be the breadth-first distance from s to v in G_f. During the Edmonds-Karp algorithm, $\delta(v)$ increases monotonically.

Proof:

Suppose that augmenting a flow f on G produces a new flow f'. Let $\delta'(v) = \delta_{f'}(s, v)$. We will show that $\delta'(v) \geq \delta(v)$ by induction on $\delta'(v)$.

Base Case: $\delta'(v) = 0$. This implies that $v = s$, and since $\delta(s) = 0$ and distance can never be negative, it follows $\delta'(s) \geq \delta(s)$.

Inductive Case: Assume inductive hypothesis holds for any u where $\delta'(u) < \delta'(v)$. We will show that it is also hods for v.

Consider a breadth-first path $s \to \cdots \to u \to v$ in $G_{f'}$. We must have $\delta'(v) = \delta'(u) + 1$, since subpaths of shortest paths are also shortest paths. Also note that by our inductive assumption $\delta'(u) \geq \delta(u)$, because $\delta'(u) < \delta'(v)$. Certainly, $(u, v) \in E_{f'}$. We will now prove that $\delta'(v) \geq \delta(v)$ in both cases where $(u, v) \in E_f$ and $(u, v) \notin E_f$.

Case 1: $(u, v) \in E_f$. Here we have:

$$\begin{aligned} \delta(v) &\leq \delta(u) + 1 && \text{triangle inequality} \\ &\leq \delta'(u) + 1 && \text{inductive assumption} \\ &= \delta'(v) && \text{breadth-first path} \end{aligned} \tag{1}$$

Therefore $\delta'(v) \geq \delta(v)$ and monotonicity of $\delta(v)$ is established.

Case 2: $(u, v) \notin E_f$. Here, the only way $(u, v) \in E_{f'}$ is if the augmenting path p that produced f' from f must have included (v, u). Moreover, p is a breadth first path in G_f:

$$p = s \to \cdots \to v \to u$$

Thus, we have:

$$\begin{aligned} \delta(v) &= \delta(u) - 1 && \text{breadth-first path} \\ &\leq \delta'(u) - 1 && \text{inductive assumption} \\ &= \delta'(v) - 2 && \text{breadth-first path} \\ &< \delta'(v) \end{aligned} \tag{2}$$

thereby establishing monotonicity for this case, too. □

Counting Flow Augmentations

Theorem. The number of flow augmentations in the Edmonds-Karp algorithm is $O(VE)$.

Proof:

For an augmenting path p, define $c_f(p) = \min\{c_f(u, v) \in p\}$.

Let p be an augmenting path, and suppose that we have $c_f(p) = c_f(u, v)$ for edge $(u, v) \in p$. Then, we say that (u, v) is **critical**, and it disappears from the residual graph after flow augmentation. This is because during augmentation, the residual capacity of every edge in p decreases by $c_f(p)$ as that much new flow is pushed through the augmenting path. And since $c_f(u, v) - c_f(p) = 0$, the edge disappears after augmentation.

The first time an edge (u, v) is critical, we have $\delta(v) = \delta(u) + 1$ since p is a breadth-first path. After the augmentation, we must wait until (v, u) is on an augmenting path before (u, v) can be critical again. Let δ' be the distance function in the residual network when (v, u) is on an augmenting path. Then, we have:

$$\begin{aligned} \delta'(u) &= \delta'(v) + 1 && \text{breadth-first path} \\ &\geq \delta(v) + 1 && \text{monotonicity} \\ &= \delta(u) + 2 && \text{breadth-first path} \end{aligned} \tag{3}$$

Hence between each occurrence of an edge (u, v) as critical, $\delta(u)$ increases by at least 2. And since $\delta(u)$ starts out non-negative and can be at most $|V| - 1$ until the vertex is unreachable, each edge can be critical $O(V)$ times. And since the residual graph contains $O(E)$ edges, the total number of flow augmentations is $O(VE)$. □

Corollary. The Edmonds-Karp maximum-flow algorithm runs in $O(VE^2)$ time.

Proof: Breadth-First Search runs in $O(E)$ time, and there are $O(VE)$ augmentations. All other bookkeeping is $O(V)$ per augmentation.

Applications of Network Flow

Vertex Cover

Given an undirected graph $G = (V, E)$, we say that a set $S \subseteq V$ of vertices covers G, if for every edge $(u, v) \in E$, S contains either u or v. The Vertex Cover problem is now to find S such that S covers G and $|S|$ is minimal.

Vertex Cover is NP-Hard in general graphs but polynomial time solvable in bipartite graphs.

Bipartite Vertex Cover

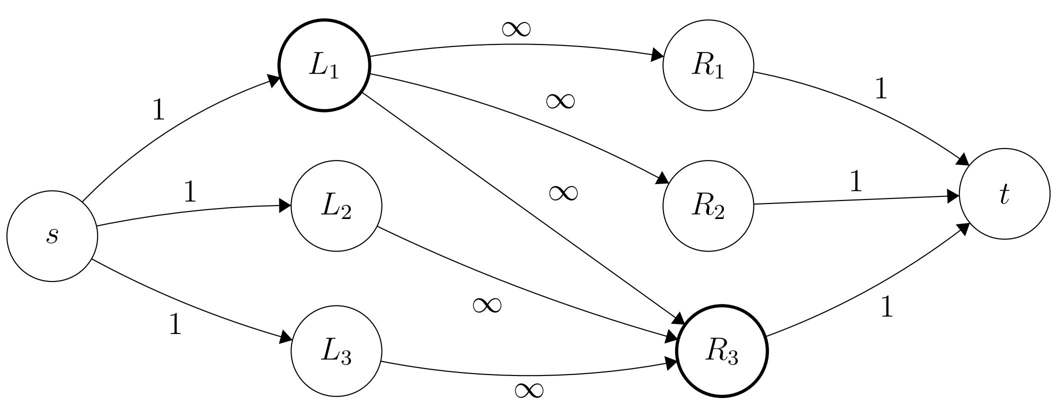
Given a bipartite graph $G = (L \sqcup R, E \subseteq L \times R)$, find the set S such that S covers G and $|S|$ is minimal.

Solution: Given G, define the following Flow Network H:

- Create a new source vertex s and add edges of capacity 1 from s to every vertex in L
- Create a new sink vertex t and add edges of capacity 1 from every vertex in R to t
- Direct all edges in E from L to R and assign each edge ∞ capacity

Run Maximum Flow in H and return the value.

For example, consider the following graph H constructed from $G = (\{L_1, L_2, L_3\} \sqcup \{R_1, R_2, R_3\}, E)$ where E consists of the shown edges:



In this example, the Maximum Flow is 2, and the minimal vertex cover is $Q = \{L_1, R_3\}$ and $|Q| = 2$.

Correctness of Bipartite Vertex Cover as Maximum Flow

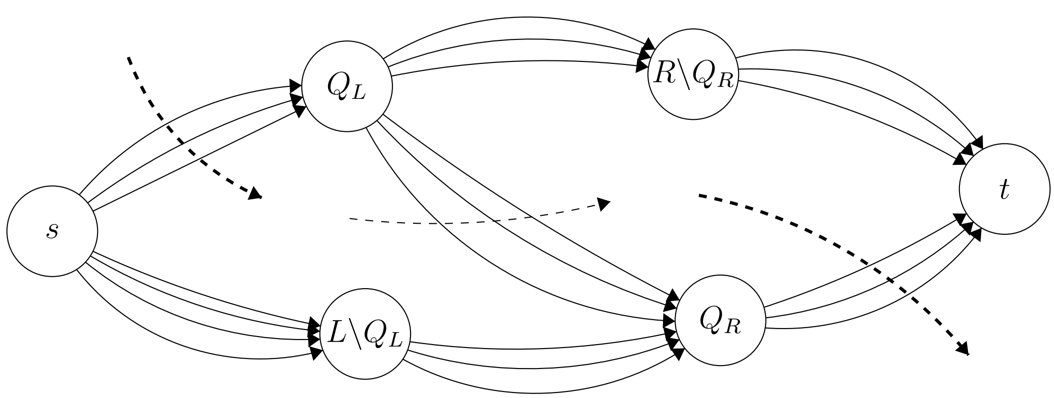
Claim 1: Every Vertex Cover Q of H defines an (S, T) cut of a finite value $c(S, T)$.

Proof: Let $Q = Q_L \sqcup Q_R$ where $Q_L = Q \cap L$ and $Q_R = Q \cap R$. Then define the cut (S, T) as follows:

$$S = \{s\} \cup Q_R \cup (L \backslash Q_L)$$

$$T = \{t\} \cup Q_L \cup (R \backslash Q_R)$$

Proof by picture:

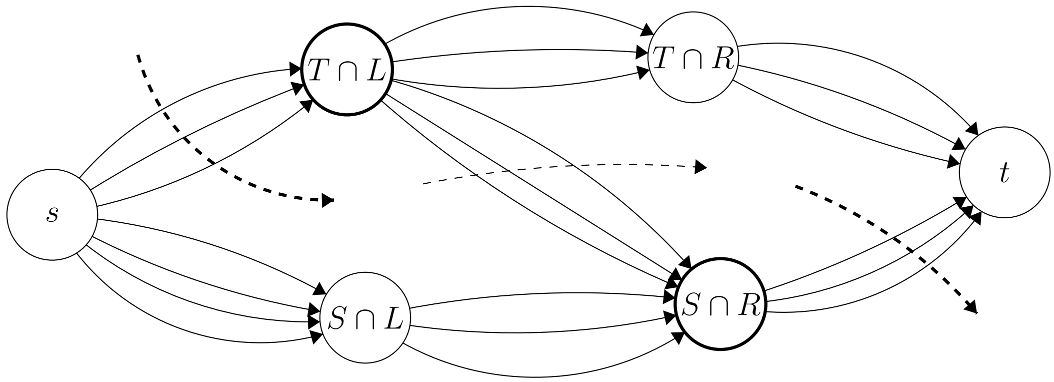


Note that there cannot be any edges going from $L\backslash Q_L$ to $R\backslash Q_R$ because if there were such an edge, both endpoint vertices would not be covered and would contradict that Q was a valid vertex cover. From the picture it is clear that (S, T) is indeed a cut in H. It is also clear that $c(S, T) = Q_L + Q_R$ because the only edges that cross the cut (S, T) are all the edges from s to Q_L and from Q_R to t, and each of them have capacity 1. $\square$

Claim 1 implies that $c(S^*, T^*) \leq |Q^*|$ where Q^* is the minimum Vertex Cover of G and (S^*, T^*) is the minimum cut in H.

Claim 2: For any *finite* cut (S, T) in H, the set $Q = (S \cap R) \cup (T \cap L)$ is a Vertex Cover of G.

Proof: Observe the following picture:



Note that there cannot be any edges going from $S \cap L$ to $T \cap R$ because that would make $c(S, T)$ infinite and contradict the assumption that the cut must be of finite capacity. From this picture, it is clear that every edge in G has at least one end point in either $T \cap L$ or $S \cap R$ and indeed $Q = (S \cap R) \cup (T \cap L)$ covers G. It is also clear that $|Q| = |S \cap R| + |T \cap L| = c(S, T)$. $\square$

Claim 2 implies that $|Q^*| \leq c(S^*, T^*)$ where (S^*, T^*) is the minimum cut in H and Q^* is the minimum Vertex Cover of G.

Punchline:

By claims 1 and 2, the size of the minimum Vertex Cover of G, $|Q^*|$ is **equal** to the size of minimum cut (S^*, T^*). And since the Maximum Flow is **equal** to the Minimum Cut, we can use Maximum Flow to solve Bipartite Vertex Cover in the way described above.

MIT OpenCourseWare
http://ocw.mit.edu

6.046J / 18.410J Design and Analysis of Algorithms
Spring 2015

For information about citing these materials or our Terms of Use, visit: http://ocw.mit.edu/terms.