CS 31: Algorithms (Spring 2019): Lecture 17

Date: 21st May, 2019

Topic: Graph Algorithms 7: Applications of Flows

Disclaimer: These notes have not gone through scrutiny and in all probability contain errors. Please notify errors on Piazza/by email to deeparnab@dartmouth.edu.

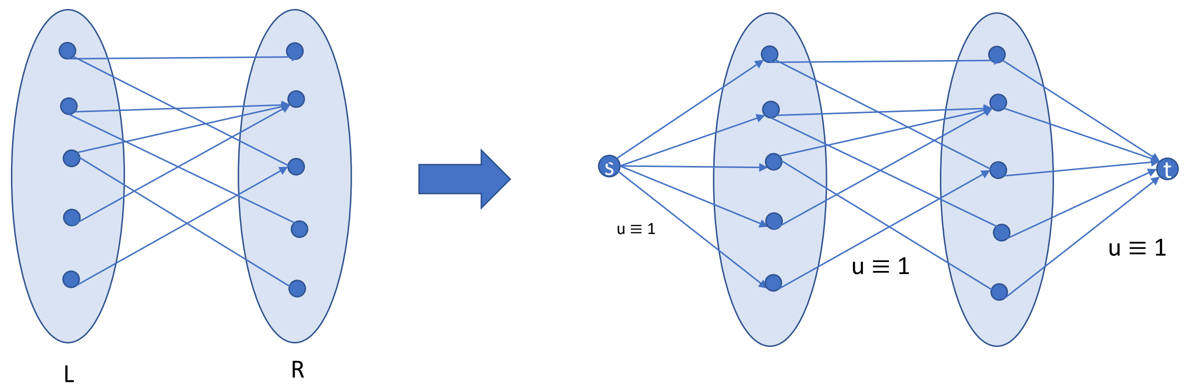
Flows have numerous applications. We sample some.

1 Matchings in Bipartite Graphs

We can use flows to find matchings in bipartite graphs. Recall, a matching M in any graph G is a collection of edges which don't share an end point.

MAXIMUM CARDINALITY BIPARTITE MATCHING
Input: A *bipartite* graph $G = (L \cup R, E)$.
Output: A maximum matching M in G.

Given the graph G, we construct a flow network $\mathcal{N} = (G, s, t, u)$ as follows. We introduce a source node s and sink node t. We add edge (s, ℓ) for all $\ell \in L$ and add edge (r, t) for all $r \in R$. We set $u(e) = 1$ for all these edges and the edges $e \in G$. The figure below shows an illustration.



Lemma 1. The value of the maximum flow in $\mathcal{N}$ is the size of the maximum cardinality matching in G.

Proof. Let M be the maximum matching in G. Let $|M| = k$, and let $M = \{(\ell_1, r_1), (\ell_2, r_2), \ldots, (\ell_k, r_k)\}$. We can send a flow of value k in $\mathcal{N}$ by sending them along the paths (s, ℓ_i, r_i, t).

Conversely, if there is a flow of value k in $\mathcal{N}$, then using integrality of flow we may assume this flow has $f(e) \in \{0, 1\}$ on all edges. Thus, we have k edge disjoint paths from s to t in $\mathcal{N}$. Focusing on the edges of G in these paths, we get a matching of size k. □

Theorem 1. MAXIMUM CARDINALITY BIPARTITE MATCHING can be solved in $O(nm)$ time.

We can also derive a theorem you may have seen in previous courses: Hall's Theorem. A matching in a graph G is *perfect* if all vertices participate as endpoints in the matching. This theorem states a necessary and sufficient condition for a bipartite graph G to have a perfect matching. A definition: given any subset $S \subseteq L$, we define $\Gamma S := \{r \in R : \exists \ell \in S, (\ell, r) \in E\}$ to be the set of neighbors of S.

Theorem 2 (Hall's Theorem)**.** A bipartite graph $G = (L \cup R, E)$ with $|L| = |R|$ has a perfect matching if and only if for all $S \subseteq L$, $|\Gamma S| \geq |S|$.

Proof. We consider the network $\mathcal{N}$ defined above with one extra change: for all $e \in G$, we set $u(e) = \infty$. Note that the maximum flow doesn't change since the total capacity incoming into any $\ell \in L$ is 1, and also the total capacity out going from any $r \in R$ is also 1. Thus, the infinite capacity in the "middle" doesn't help in sending more flow. We see that it makes our arguments easier.

Now, we know that G has a perfect matching if and only if the maximum s, t flow in $\mathcal{N}$ is of value $|L|$. Using the max-flow-min-cut theorem, we get G has a perfect matching iff the minimum s, t cut in $\mathcal{N}$ is of value $|L|$.

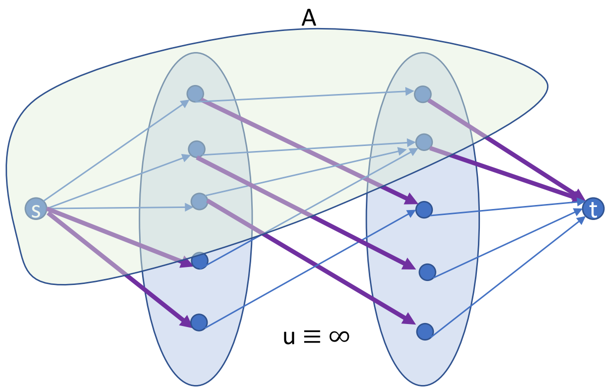
Let us consider an s, t cut in $\mathcal{N}$. Let A be the subset inducing the cut; $A = s \cup S \cup T$ where $S \subseteq L$ and $T \subseteq R$. Note, $t \notin A$. The capacity of this cut is as follows.

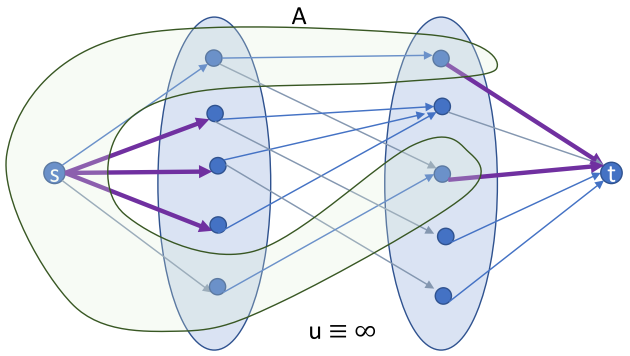
$$u(\partial^+ A) = \begin{cases} \infty & \text{if } \Gamma S \not\subset T \\ (|L| - |S|) + |T| & \text{otherwise} \end{cases}$$

To see this, note that if $\Gamma S \not\subset T$, then there is an edge $(\ell, r) \in \partial^+ A$ of capacity ∞ (this is why we defined it so). If $\Gamma S \subset T$, then the only edges in $\partial^+ A$ are of the form (s, ℓ) for $\ell \notin S$ and (r, t) for $r \in T$.

Now, if A were the minimum s, t cut, then observe that we would pick $T = \Gamma S$ (we would like to pick T as minimal as possible). Therefore, the value of the minimum s, t cut is precisely $\min_{S \subseteq L}(|L| - |S| + |\Gamma S|) = |L| + \min_{S \subseteq L}(|\Gamma S| - |S|)$.

See the figure below for an illustration.





Putting everything together, we get G has a perfect matching if and only if

$$|L| + \min_{S \subseteq L} (|\Gamma S| - |S|) = |L|$$

or, in other words, for every $S \subseteq L$, we have $|\Gamma S| \geq |S|$. $\square$

2 A Project Selection Problem

In this problem we are given a *directed acyclic graph* $G = (V, E)$. Each node v_i corresponds to a project, and has an associated value p_i. This value could be *positive*, that is, completing this project gives you p_i units of revenue, or it could be *negative*, completing this project leads to a loss of p_i units.

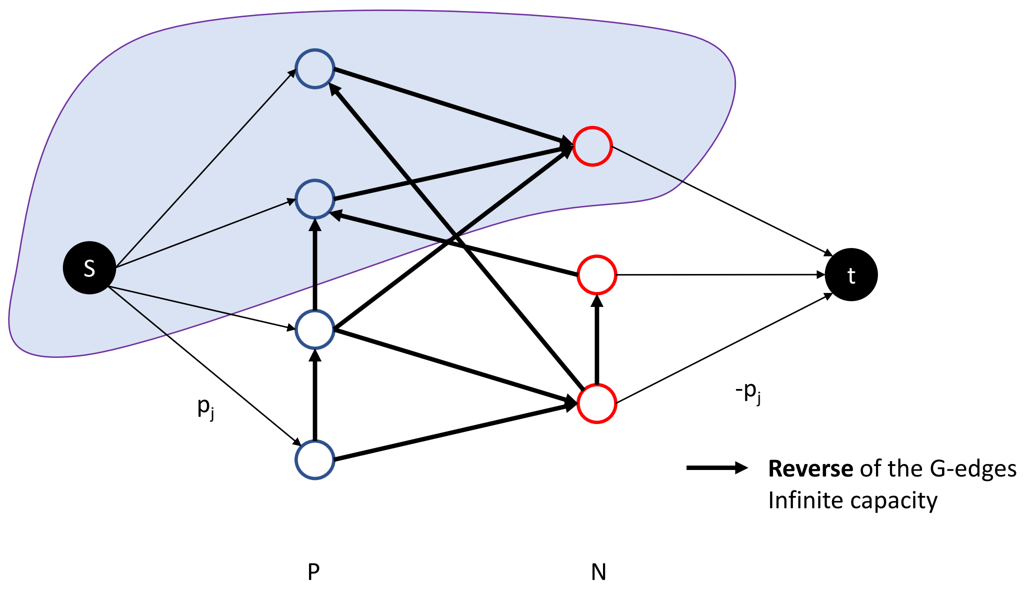
The edges (v_i, v_j) are *precedence constraints* – to perform a project v_j one must complete the project v_i as well. This is similar to a problem in your PSet. Indeed, if you think of these projects as classes you need to take in Dartmouth. Then you can imagine one vertex titled "Major" which has some value (hopefully, significantly positive for everyone). However, there is a node titled "CS 31" which points to it. And it may (or hopefully may not) have negative value. However, to complete the "Major" project, you need to complete the "CS 31" project.

The objective of the problem is to select a subset of projects $S \subseteq V$ to maximize the total value $\sum_{i \in S} p_i$. The constraint is: if a vertex $v \in S$, then for all edges (u, v), the vertex $u \in S$ as well. Such a set is called ***valid***. How will we attack this problem?

To begin with let us consider two *trivial* solutions which should form our benchmarks. One, we pick no project. This gives us a total value of 0. Second, we pick every project $S = V$. The total value we get is $\sum_{v \in V} p_v$. Let's call this last term Θ. Note that this could be positive, negative, or zero.

Let me first describe the *reduction* to the minimum s,t-cut problem. Given the input above, we describe a flow network $N = (H, s, t, u)$. $H = (V \cup s \cup t)$, that is, we have the original vertices plus a source and a sink. We partition V into two groups: $P \subseteq V$ with $p_i > 0$, that is, the projects which give positive value. We add an edge (s, v_i) for each $v \in P$ of capacity p_i. The remaining vertices N, for not positive, have $p_i \leq 0$. For each vertex $v_j \in N$, we add an edge (v_j, t) of capacity $-p_j$. Note that the capacities are non-negative.

Next, for every edge (v_i, v_j) in the graph G, we add the ***reverse*** edge (v_j, v_i) in the graph H. We assign a capacity ∞ to each such edge. See the figure below as illustration.



Now, our algorithm for the project selection problem is the following. We obtain a minimum s,t-cut for the network N. If this minimum cut is $\partial^+ S$, then we choose $S \setminus s$ as the set of projects. We claim this is the best solution. This will follow from the following claims.

Claim 1. If $u(\partial^+ S) \neq \infty$, then the set $S \setminus s$ is a *valid* set. The total value of the set $S \setminus s$ is precisely $\Theta - u(\partial^+ S)$.

Proof. Let's first prove that $S \setminus s$ is valid. Suppose not. Suppose there is a project $v \in S \setminus s$ and another project $u \notin S$, such that $(u, v) \in G$. But then, $(v, u) \in H$. And that would make the capacity of $u(\partial^+ S) = \infty$. Thus, $S \setminus s$ is a valid set.

The total value of the set is the $\sum_{v \in S\text{cap}P} p_v + \sum_{v \in S\text{cap}N} p_v$. Note that for every $v \in S\text{cap}N$, we have $\sum_{v \in S\text{cap}N}(-p_v)$ equal the capacity of the edges from S to t. Note that

$\sum_{v \in S \mathsf{cap} P} p_v = \Theta - \sum_{v \notin S \mathsf{cap} P} p_v = \Theta$ minus the total capacity of edges from the source s to outside S. Thus, $\sum_{v \in S \mathsf{cap} P} p_v + \sum_{v \in S \mathsf{cap} N} p_v = \Theta - u(\partial^+ S)$. □

Claim 2. Let $A \subseteq V$ be any valid solution for the project selection problem. Then $A \cup s$ induces an s, t-cut of capacity $u(\partial^+ S) = \Theta - \mathsf{val}(A)$.

Proof. Since A is valid, there is no edge from $v \in A$ to $u \notin A$; thus the capacity of $\partial^+(A \cup s)$ is precisely the capacity of the edges from s to $u \in P \setminus A$ and $N \mathsf{cap} A$ to t. The capacity of these edges precisely is $\Theta - \mathsf{val}(A)$. □