CMPUT 675: Topics in Algorithms and Combinatorial Optimization (Fall 2009)

Lecture 12,13: Flows and Circulations, Matroids

Lecturer: Mohammad R. Salavatipour **Scriber:** Michael Joya
Date: Oct.13-15, 2009

1 Flows and Circulations cont'd

This lecture builds on the fundamentals of flows that were covered in the last week. We look at some related concepts to flow, namely circulations and b-transshipments, and see some variant problems. We then spend a lecture introducing matroids and examine some instances of these structures.

Some applications of maximum flow

Maximum Bipartite matching: We can solve problems of related types using flows. Eg. Consider augmenting a bipartite graph $G = (A \cup B, E)$ with a node s and t s.t. there is an edge from s to every $a \in A$ and an edge from every $b \in B$ to t, all with capacities 1. We also direct the edges between A and B to go from A to B. Then it is straightforward that in every maximum $s-t$-flow in this graph, the edges between A and B with non-zero amount of flow correspond to a maximum matching in the original graph G.

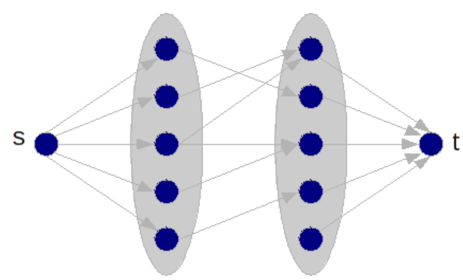

Figure 1: A bipartite graph with added nodes s and t.

Edge-disjoint paths: The problem of finding maximum number of edge-disjoint paths one can find between two given nodes s and t can be solved using max-flow by assigning a capacity of 1 to each edge. Then in any flow, the flow must follow edge-disjoint paths. Using max-flow-min-cut theorem, the number of such paths is equal to the minimum number of edges across any $s-t$-cut. This is known as Menger's theorem:

Theorem 1.1 (Menger) *In any graph $G(V, E)$ and any $u, v \in V$, the maximum number of edge-disjoint paths between u, v is equal to the minimum number of edges whose removal disconnects u, v (aka the connectivity of u, v).*

Definition 1.2 (Connectivity) *Connectivity of G is the minimum size of the set $U \subseteq V$ such that $G - U$ is connected [S03].*

Multisource, multisink flow: Suppose we are given a graph $G = (V, E)$ with a set of sources $\{s_1, \ldots, s_k\}$ and a set of sinks $t_1, \ldots, t_\ell$. Each edge has a capacity and our goal is to find a maximum amount of flow assuming that the flow originates from the sources and arrives at the sinks. The maximum flow problem in this multi-source multi-sink instance can be reduced to the single-source single-sink case by just adding a

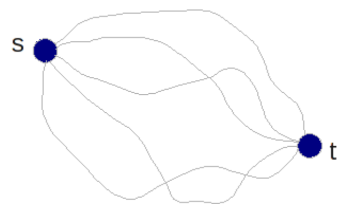

Figure 2: A set of edge-disjoint paths.

universal source s connected to all s_i's with directed edges with capacity ∞ and connect all the sinks t_j's to a new sink node t with capacity ∞; now compute a maximum $s-t$-flow in the new graph.

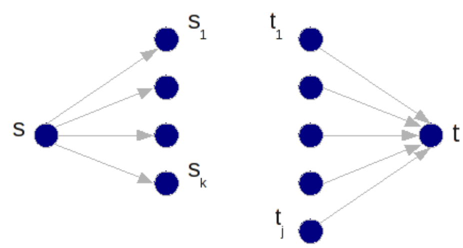

Figure 3: A flow with k sources and j sinks

2 Circulations

Given a digraph $G = (V, E)$ and any function $f : E \to \mathbb{R}^{\geq 0}$, the excess function $excess_f : V \to \mathbb{R}$ is defined as $excess_f(v) = f\big(s^{in}(v)\big)$.

Function f is called a circulation if $excess_f(v) = 0$ for every vertex v. So we have flow conservation everywhere. Note that in an $s-t$-flow we have $excess_f(v) = 0$ for all $v \neq s, t$ and $excess_f(s) = -excess_f(t)$.

Given a vector b (of size $|V|$), we say f is a b-transshipment if $excess_f(v) = b(v) \quad \forall v \in V$.

2.1 Relations of Circulations and Flows

Suppose we are given a digraph $G = (V, E)$ and two capacity bounds $d, c : E \to \mathbb{Q}^{\geq 0}$ with $d \leq c$. Our goal is to find a circulation f satisfying $d \leq f \leq c$.

This problem can be solved by reducing it to a flow problem.

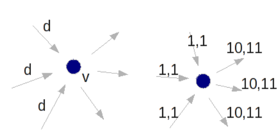

Figure 4: Nodes in a circulation.

For each edge $e \in E$ we define a new capacity as: $c'(e) = c(e) - d(e)$. We add two new vertices s, t to the set of nodes. For each $v \in V$ with $excess_d(v) > 0$ add an edge sv with capacity $c'(s, v) = excess_d(v)$

and for each vertex v with $excess_d(v) < 0$ add an edge vt with capacity $c'(v,t) = -excess_d(v)$. Call this new graph G'. We claim that G' has an $s-t$-flow f' with capacity constraint c' of value $|f'| = \sum_{v: excess_d(v)>0} excess_d(v)$ if and only if G has a flow f with $d \leq f \leq c$. To see this it is sufficient to take $f(e) = f'(e) + d(e)$ for each edge $e \in E$.

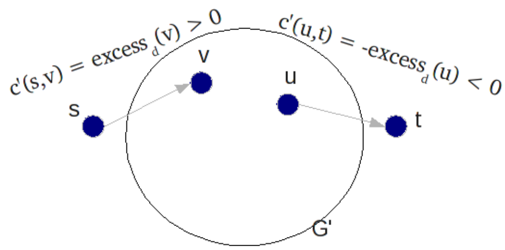

Figure 5: Depiction of intermediate step.

Claim 2.1 *G has a circulation $d \leq f \leq c$ iff G' has an $s-t$ flow with value f' (as above).*

Proof: An easy exercise. ■

We can also solve the max-flow problem using circulation. For that, suppose we are given a graph $G = (V, E)$ with source-sink pair s, t and capacities c. We create an edge ts with capacity ∞ and a demand d. The largest value of d for which there is a circulation in which the flow in on edge ts is at least d gives us a maximum $s-t$ flow.

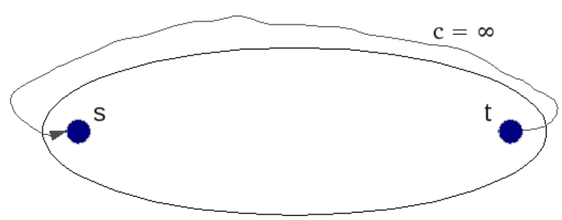

Figure 6: Make d as large as possible as long as there is a circulation.

2.2 Min-cost Flows and Circulations

Given a digraph $G = \langle V, E\rangle$ with cost/weight: $w : E \to \mathbb{R}$, any function $f : E \to \mathbb{R}^{\geq 0}$ has cost:

$$cost(f) := \sum_e f(e) \cdot w(e)$$

The **Min-cost-flow** problem is: Given $G = \langle V, E\rangle\ s, t \in V, w, \Phi, c : E \to R^{\geq 0}$, where w is the weight function, c is the capacity function and Φ a value, find a flow of value $\geq \Phi$ with minimum cost. The min-cost-max-flow problem is to find a max $s-t$-flow of minimum cost. The min-cost flow can be formulated as an LP:

$$\begin{array}{crl} \forall v \in V - \{s,t\} & f(\delta^{in}(v)) & = f(\delta^{out}(v)) \\ V * E & f(\delta^{out}(v)) & = f(\delta^{in}(t)) \\ & f(delta^{out}(s)) & \geq \Phi \end{array}$$

However, we will see that there are more efficient ways to solve this problem.

Similarly, one can define the min-cost-circulation problem: Given a graph $G = (V, E)$ with weights, capacities and demands on the edges$w, d, c : E \to \mathbb{R}^{\geq 0}$ the goal is to find a circulation $d \leq f \leq c$ with minimum cost. One can reduce the min-cost flow problem to min-cost circulation as follows: add a ts edge with capacity and demand Φ (which is the given flow requirement) and set the weight of this edge be zero $d(ts) = c(ts) = \Phi$, $w(ts) = 0$. Then a min-cost circulation corresponds to a min-cost flow with value Φ. If our goal is to find min-cost max-flow then we set $c(ts) = \infty$, $d(ts) = 0$, and $w(ts) = -1$ and for every other edge e we define $w(e) = 0$; then ask for a min-cost circulation.

Many problems are special cases of min-cost flow. For example, the shortest path problem is the special case of $\Phi = 1$: find a min-cost unit flow from s to t.

Residual Graph: The residual graph of a digraph $D = \langle V, E \rangle$ is the union of all the forward edges that have residual capacity, and all the backward edges such that the flow isn't minimal.

Definition 2.2 (Residual Graph) $G_f = \langle V, E_f \rangle$

$$\begin{array}{rllll} \forall e \in E: & e = uv & & & \\ & uv \in E_f & \text{if} & f(e) < c(e) & w_f(e) = w(e) \\ & vu \in E_f & \text{if} & f(e) > d(e) & w_f(e_f) = -w(e) \end{array}$$

Theorem 2.3 *Given $G = \langle V, E \rangle,\ w, d, c :$ let $d \leq f \leq c$ be a feasible circulation. Then f has a min-cost iff G_f has no negative-cost directed cycle.*

Proof: If $\mathcal{C}$ is a negative cycle in G_f, then for sufficiently small ϵ, we can replace f with

$$d \leq \underbrace{f - \epsilon}_{f'} \leq c$$

while f' remains feasible and and $cost(f') < cost(f)$.

Conversely, suppose every cycle in G_f has ≥ 0 weight and suppose f is not min-cost. Let f' be any feasible circulation and define $f^* = f' - f$. Observe that f^* is a circulation. Since we have flow conservation at every node (for flow f^*), f^* can be decomposed into a collection of cycles: $\mathcal{C}_1, ..., \mathcal{C}_n$:

$$f^* = f' - f = \sum_i \alpha_i \mathcal{C}_i \quad \alpha_i > 0$$

Therefore, $cost(f^*) = cost(f') - cost(f) = \sum_i \alpha_i cost(\mathcal{C}_i)$. If all $\mathcal{C}_i$'s have > 0 weight, then $c(f^*) > 0$, which implies that $cost(f') < cost(f)$ a contradiction. ■

This suggests an algorithm for finding the min-cost circulation: as long as there is a negative cycle $\mathcal{C}$ in G_f, find one and update f by increasing the flow in the other direction.

If C is the maximum capacity on any edge, W is the maximum cost and $m = |E|$ then the max # of iterations is $O(m \cdot W \cdot C)$. Each iteration can be performed using Bellman-Ford which takes $O(mn)$; so the total running time will be $O(m^2 n^2 CW)$ which is not a polynomial time algorithm.

We can improve the running time to strongly polynomial if instead of finding any negative cycle in each iteration, one finds a cycle $\mathcal{C}$ minimizing $\dfrac{w(\mathcal{C})}{|\mathcal{C}|}$

Theorem 2.4 (Goldberg and Tarjan) *Choosing a minimum mean cycle in each iteration of the above algorithm, the number of iterations is at most $O(nm^2 \cdot \log n)$.*

3 Matroids

Suppose that E is a finite ground set and $\mathcal{I}$ is a collection of subsets of E, called independent sets. We say $M = (E, \mathcal{I})$ is a matroid if the following two axioms hold:

1. For any $X \subseteq Y \subseteq E$: if $Y \in \mathcal{I}$ then $X \in \mathcal{I}$
2. For any $X, Y \subseteq \mathcal{I}$: if $|Y| > |X|$ then $\exists e \in Y \setminus X : X \cup \{e\} \in \mathcal{I}$

Sets in $\mathcal{I}$ are called *independent* sets. Any set not in $\mathcal{I}$ is called a *dependent* set. Observation: all maximal independent sets of $\mathcal{I}$ have the same size. They are called *bases* of the matroid.

Example: Uniform matroid Let $\mathcal{I} = \{X \subseteq E : |X| \leq k\}$ for some given integer k. ("All subsets with order less than or equal to k"). If $|E| = n$ this is the uniform matroid $U_{n,k}$. The bases are subsets $B \subseteq E$ with size $|B| = k$ exactly. It is easy to see that both axioms are satisfied by the subsets of $\mathcal{I}$. ■

Example: Linear matroid Let $A_{m \times n}$ be an $m \times n$ matrix. Let E be the set of indices of columns of A. For $X \subseteq E$, let A_x be the submatrix with columns indexed by X and define:

$$\mathcal{I} = \{X \subseteq E : \text{ columns of } A_x \text{ are linearly independent, i.e. } rank(A_x) = |X|\}$$

Axiom (1) is easy to see, as any subset of X will be linearly independent for a $X \in \mathcal{I}$. For 2) one needs to observe that if $X \subseteq Y$ and $|Y| > |X|$ and these columns are full-rank then there must be a column of Y not spanned by X; therefore one can add a column of matrix A_Y to A_X to obtain a larger matrix of full-column rank. ■

Example: Graphic matroids Given a graph $G = (V, E)$, then $M = (E, I)$ is a matroid where each $I \in \mathcal{I}$ is a forest (i.e: an acyclic collection of edges in G.) Consider the matroid axioms:

1. Any subset of a forest is a forest; thus it is closed under taking subsets.
2. If $c(V, X)$ denotes the number of connected components of the graph on vertices V and edges X then for any pair $X, Y \in \mathcal{I}$ with $|X| < |Y|$ we have $c(V, X) > c(V, Y)$; thus there is an edge in $Y - X$ such that $X \cup \{e\}$ is a forest (such an edge runs between two connected components of (V, X)).

If G is connected then bases of this matroid correspond to spanning trees of G. ■

Any minimal dependent set is called a *circuit*. In graphic matroids a circuits correspond to a cycle in G.

Example: Matching matroid Given graph $G = (V, E)$, say $\mathcal{I} = \{F \subseteq E : F \text{ is matching in } G\}$: One might ask whether this is a matroid or not. Although any subset of a matching is a matching too, the second axiom of matroid is not satisfied, for example assume that $X = \{bc\}$ while $Y = \{ab, cd\}$. Then clearly one cannot extend X to a larger matching by adding edges from Y.

However we can define a matroid in the following way:

$$\mathcal{I} = \{S \subseteq V : S \text{ is covered by some matching } M\}$$

One can check that matroid axioms are satisfied:

1. is easy to check

2. Suppose that $X, Y \in \mathcal{I}$, with $|X| < |Y|$, say M_1 is a matching covering the nodes in X and M_2 is a matching covering the nodes in Y. Our goal is to show there is a node $v \in Y \setminus X$ and a matching M' that covers $X \cup \{v\}$. If M_1 covers v too we are done. Otherwise consider $M_1 \Delta M_2$. There must be an alternating path from some vertex $v \in Y \setminus X$ to a vertex not in X. Then applying this path to M_1 gives a matching that covers $X \cup \{v\}$.

■

Recall that a minimal (inclusion-wise) dependent set is a circuit. By definition, if we remove any element from a circuit we obtain an independent set.

Theorem 3.1 *Given a matroid $M = (E, \mathcal{I})$, for every $I \in \mathcal{I}$ and $e \in E$, either $I + e \in \mathcal{I}$, or it contains a unique circuit.*

Proof: Suppose $I + e \notin \mathcal{I}$. In other words, assume that $I + e$ contains a circuit. Let $C = \{c : I + e - c \in \mathcal{I}\}$. First we claim that C is dependent. Suppose C is independent. It can be extended to a basis of $I + e$ of cardinality $|I|$, so it has the form $I + e - d$, which is contradicting the definition. Next we prove that C is minimal: removing any c from C makes C a subset of $\mathcal{I} + e - c$ which belongs to $\mathcal{I}$. Thus C is a minimal dependent set, i.e. a circuit. Thirdly, we prove that C is unique. Say D is another circuit in $I + e$, so $\exists c \in C - D$. Then $D \subseteq \underbrace{I + e - c}_{\in \mathcal{I}}$ but by the definition of C, we know that $D \in \mathcal{I}$ so D is not a circuit. ■

Consider the example of graphic matroid over a connected graph G. Every base is a spanning tree. If we have two spanning trees T_1 and T_2 then for every edge $e_1 \in T_1 \setminus T_2$, $T_2 + e_1$ has a unique cycle C. Also if we remove any edge $e_2 \in C \cap T_2$ from this cycle we obtain another spanning tree of the graph. This can be proved in general for matriods:

Lemma 3.2 *Let $M = (E, \mathcal{I})$ be a matroid and B_1, B_2 be bases. Let $x \in B_1 \setminus B_2$. Then $\exists y \in B_2 \setminus B_1$ such that $B_2 - x + y$ and $B_1 - y + x$ are bases.*

3.1 Rank functions

Let $M = (E, \mathcal{I})$ be a matroid and $A \subseteq E$. The rank function of M is a function $r_M : 2^{|E|} \to \mathbb{N}$ such that $r_M(A) = max\{|X| : X \subseteq A, X \in \mathcal{I}\}$. Note that all such subsets X of A have the same size; so the rank function is well-defined.

Example: Linear matroid Recall that in a linear matroid E is the set of indices of columns of a matrix $A_{m \times n}$). Here, $r_M(X)$ is exactly the rank of the matrix A_X in the algebraic sense. ■

Example: Graphic matroid Consider matroid $M = (E, \mathcal{I})$ over the graph $G = (V, E)$ with $\mathcal{I}$ being the set of forests of G. For every $F \subseteq E$ with $c(V, F)$ connected components, $r_M(F) = n - c(V, F)$. ■

We will prove later that:

Theorem 3.3 *Let E be a ground set, and $\mathcal{I}$ a collection of subsets of E that is closed under taking subsets. Consider a function r:*

$$r : 2^{|E|} \to \mathbb{N}$$

Then r is the rank function of a matroid if and only if for all $X, Y \subseteq E$:

1. $0 \leq r(X) \leq |X|$
2. *if $X \subseteq Y$ then $r(X) \leq r(Y)$*
3. *r is submodular, i.e. $r(X \cap Y) + r(X \cup Y) \leq r(X) + r(Y)$*

References

S03 A. SCHRIJVER, Combinatorial Optimization, *Springer 2003, pp. 148–195,237.*