CS 684: Approximation and Network Algorithms	**10/24/01**
Network Design with higher connectivity	**scribe: Alexander Druyan**

1 Network Design with higher connectivity

In the network design with higher connectivity problem, we are given an (undirected) graph $G = (V, E)$ with edge costs $c_e \geq 0$ for all edges $e \in E$, and connectivity requirements $r_{ij} \geq 0$ for each pair $i, j \in V$ of vertices[1]. We want to find a min-cost subgraph of G such that for all vertices $i, j \in V$ and cuts S, if $i \in S$ and $j \notin S$ then the number of edges in $\delta(S)$ in the subgraph is $\geq r_{ij}$.

For any cut S, define $f(S)$ to be

$$f(S) = \max_{i \in S, j \notin S} r_{ij}$$

So for the network design with higher connectivity we need to have $f(S)$ edges leaving S. This problem can be very conveniently written as an LP. Also, note that in the case of this problem, it is not clear what is a good economic interpretation: Sets would have to pay to have enough entering edges, but does a set pay the same to all its entering edges?

2 Augmentation Problem

Suppose that the set of edges $E_0 \subseteq E$ are already included in the start of a solution for the network design with higher connectivity problem instance, but we still do not have a complete solution. This is due to the fact that there must still be some sets which do not have sufficiently many edges. Denote by $\mathcal{S}$ a subset of those sets, i.e., the subset of the sets which are missing edges, or formally,

$$\mathcal{S} \subseteq \{S \quad \text{s.t.} \quad \delta_{E_0}(S) < f(S)\}$$

Goal: find a min-cost subset of the edges $F \subseteq E \backslash E_0$, with $\delta_F(S) \geq 1$ for all $S \in \mathcal{S}$.

Here is a possible way to approach this:

Look at all sets which do not have enough edges, give them one more edge, iterate.

This is clearly greedy, since once you select an edge, you commit to it. So now, being able to get this F, we add it to E_0 and repeat. This is still missing two important things:

[1] We assume that the requirements are symmetric, i.e. $r_{ij} = r_{ji}$ for all i, j.

- What $\mathcal{S}$ to pick: to pick $\mathcal{S}$, find sets which are missing the most edges. Let

$$k = \max_S f(S) - \delta_{E_0}(S)$$

  where S goes over all subsets, including the empty set $\emptyset$. We call any set S a *biggest deficiency set* if it has $f(S) - \delta_{E_0}(S) = k$. We let $\mathcal{S}_k$ be the set of all deficiency sets, i.e.,

$$\mathcal{S}_k = \{S \quad \text{s.t.} \quad f(S) - \delta_{E_0}(S) = k\}$$

  Run augmentation algorithm if $k > 0$.

- The above algorithm will have computational difficulty - it is hard to compute k and hard to list $\mathcal{S}_k$.

  If the set is deficient then there is some i, j pair that is displaced. Consider each $i, j \in V$. Look at min-cuts in (V, E_0) where i and j are on different sides of the cut, and compute

$$\max_{i,j} [r_{ij} - \text{min (i, j)-cut}]$$

  which will yield the value of k. The above computation requires finding n^2 max-flows. Note that the number of max-flow computations can be decreased to n.

Theorem 2.1 *If the min-cost of the original network design problem is OPT and the current deficiency is $k = \max_S f(s) - \delta_{E_0}(S)$ then we find $F \subseteq E \backslash E_0$ solving augmentation problem for $\mathcal{S}_k$ of cost $\leq \frac{2}{k} OPT$.*

Instead of proving this theorem, we look at a corollary:

Corollary 2.2 *The resulting network design problem has an approximate guarantee which is $(\frac{2}{R} + \frac{2}{R-1} + \cdots + \frac{2}{1})OPT = 2H(R)OPT \approx 2ln(R)OPT$, where $R = max_{i,j} r_{ij}$.*

Proof: We will use the augmentation problem for $\mathcal{S}_k$ for $k = R, \ldots, 1$. Summing up the bounds given by the theorem above, we get the claimed bound.

There is a weakness to this approach, that the computational requirements are relatively large, and the approximation ratio is not that good.

The main technical fact used in this lecture will be the following. FACT 0: If two sets A and B are maximally violated, then either $A \cap B$ is also maximally violated, or both $A \setminus B$ and $B \setminus A$ is also maximally violated.

We will not prove this fact here, but will prove the FACTS we felt as unproven in lecture using this FACT 0.

3 Primal Dual Algorithm for the Augmentation Problem

There is a price p_S for each $S \in \mathcal{S}_k$. We start off the algorithm with zero prices, i.e., with setting $p_S = 0$ for all S.

We maintain the following constraints:

1. For all $e \in E \backslash E_0$ we have
$$\sum_{S \text{ s.t. } e \in \delta(S)} p_s \leq c_e$$
(edge is not overpaid).

2. Subset $F \subseteq E \backslash E_0$ s.t. $e \in F$
$$\sum_{S \text{ s.t. } e \in \delta(S)} p_s = c_e$$

FACT 1: Minimal sets in $\mathcal{S}_k$, that have $\delta_F(S) = 0$ are disjoint.

Proof: The minimal sets mentioned in this fact, are the maximally violated sets subject to $E_0 \cup F$. By FACT 0 the minimal sets in $\mathcal{S}_k$ are always disjoint as if A and B are intersection maximally violated sets than either $A \cap B$ or $A \setminus B$ and $B \setminus A$ are also maximally violated, and hence in either case A and B were not minimal.

Algorithm 3.1 *Primal-Dual Algorithm for network design with higher connectivity*

1. *take minimal sets in $\mathcal{S}_k$ which are not yet augmented (i.e., $\delta_F(S) = 0$)*

2. *uniformly increase p_S on all such sets (Note that due to FACT 1 we are not increasing p for too many sets S. There can be at most $|V|$ disjoint sets..)*

3. *if e is paid for, include it in F (so that we do not end up over-paying for some edge)*

4. *repeat until there are no minimal sets found*

5. *Cleanup: consider $e \in F$ in reverse order of being added. Delete e if it is not needed for augmentation.*

In the iteration i of the above algorithm, let $\mathcal{M}_i$ denote the minimal sets at that iteration, and Δ_i the increase in prices. Note that the termination condition of the loop (1)-(4) is exactly when $\mathcal{M}_i = \emptyset$.

The algorithm at step (1) requires an ability to find minimal sets. So we need an algorithm which will give us those sets. This is accomplished by simply running the minimal-cut algorithm for every i, j pair.

FACT 2: How to get hold of the minimal sets

- Minimal sets needed can be computed by n^2 max-flow computations, and

- Edge usefulness of edge e can be tested in n^2 max-flow computations (consider F without e and recompute k and see if it goes down).

Sets are offering money, all sets are paid for.

Lemma 3.2 *labellm:price-bound* $\sum_S p_S \leq OPT/k$*, so not capable of offering more than* OPT/k*.*

Constraint (1) prevents you from offering too much money.
Proof: In order to be able to do any comparisons to the OPT, we first have to understand what the OPT is like. Let $O \subseteq E$ be the optimal set of edges. Look now at the set $O \backslash E_0$, which is at least a candidate solution for our initial augmentation problem. Clearly,

$$\delta_{O\backslash E_0}(S) \geq k \text{ for } S \in \mathcal{S}_k$$

for the current most needy sets, since

$$\delta_{O\backslash E_0}(S) \geq f(S) - \delta_{E_0}(S) = k \text{ if } S \in \mathcal{S}_k$$

Now,

$$\begin{aligned} OPT &= \sum_{e\in O} c_e \\ &\geq \sum_{e\in O\backslash E_0} c_e \\ &\geq \sum_{e\in O\backslash E_0} \sum_{e\in\delta(S)} y_S \\ &= \sum_S y_S \cdot \delta_{O\backslash E}(S) \\ &\geq \sum_S y_S \cdot k \end{aligned}$$

which proves the lemma. Note that this is linear programming weak duality.

Lemma 3.3 *If F is a solution found for augmentation problem then*

$$\sum_{e\in F} c_e \leq 2 \sum_S y_s$$

<u>FACT 3</u>: $\cup\mathcal{M}_i$ in the algorithm creates "laminar" sets, i.e., 2 sets are either disjoint or one contains the other.
Proof: Any set S that is minimally violated at some stage $\mathcal{M}_i$ was also violated at all earlier stages $j < i$. By FACT 0 all violated sets $S' \in \mathcal{M}_i$ are either contained in S or are disjoint from S.

Lemma 3.4 *At each iteration i,*

$$\sum_{S\in\mathcal{M}_i} \delta_F(S) \leq 2|\mathcal{M}_i|$$

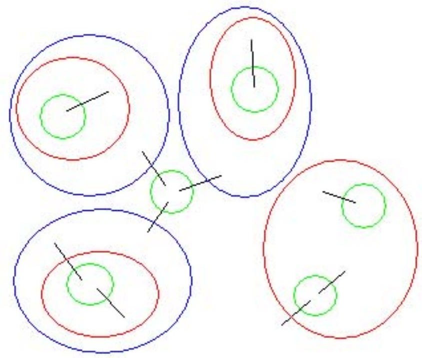

Figure 1: The reasons why the edges were not deleted.

Proof: On every edge that we chose not to delete, there was a reason. For each edge in $\delta_F(S)$ for $S \in \mathcal{M}_i$, here is a pictorial "reason" why e was not deleted:

We deleted edges in reverse order of when they were added. At stage i when $\mathcal{M}_i$ were the minimal violated sets none of the edges leaving the sets $S \in \mathcal{M}_i$ were added yet. Therefore any "reason" set for there edges in $\delta_F(S)$ for a set $S \in \mathcal{M}_i$ must have been violated at stage i, and hence it will contain some set in $\mathcal{M}_i$. Using FACT 0 we get a stronger fact the reason sets are laminar with $\mathcal{M}_i$. In fact, we can even prove the following fact.

FACT 4: We can have reason sets so that $\mathcal{M}_i \cup$ reasons is a laminar system.

What is so nice about a laminar system?

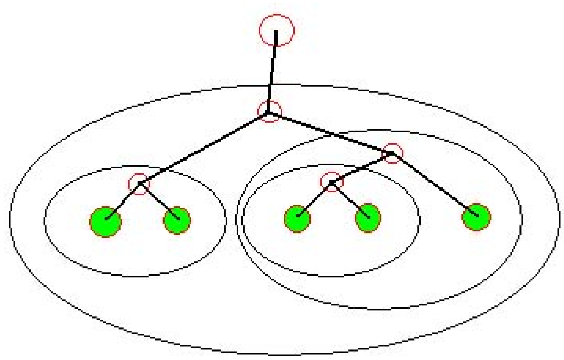

Figure 2: The tree corresponding to a laminar set.

Consider the set-system of the "reason" sets only. From the set system we get a corresponding tree. Each set gets a node, and there is an edge between two nodes i, j iff set S_i is contained in S_j and there is no set S smaller than S_j which contains S_i but is contained in S_j.

Then all minimal sets are nodes (but not all nodes are minimal sets). In the corresponding tree, the reason sets are nodes.

1. All leaves in this tree naturally correspond to some set in in $\mathcal{M}_i$ (leaves are the minimum ones) as the sets in $\mathcal{M}_i$ are disjoint and the "reason" sets were laminar containing some of them. Note that there are sets in $\mathcal{M}_i$ that are not leaves.

2. All sets in $S \in \mathcal{M}_i$ correspond to some node in the tree, as there is an edge $e \in F$ that leaves S, and we included a "reason" for e. This reason set must contain S.

3. $\delta_F(S) \leq$ degree of the corresponding tree node in the tree. This is true as the reason sets have exactly one edge of F leaving them (otherwise they don't serve are "reasons" for not deleting an edge, and each such set has its own leaving edge. So the leaves of the tree must have just one leaving edge. For other nodes in the tree, let S be the corresponding reason set, and $S' \subset S$ be the corresponding set $S' \in \mathcal{M}_i$, then S' can have a leaving edge that leaves S' and one leaving edge that comes from each of S' children in the tree.

From last time we have the following lemma.

Lemma 3.5 *In any tree, and any some subset G of the nodes such that all leaves are in G we have that*

$$\sum_{S \in G} \delta_F(S) \leq 2|G|$$

We use this lemma for the tree corresponding to our laminar set system, and where G is the set of nodes in $\mathcal{M}_i$ to get the bound claimed by lemma 3.4. ☐

Now we want to use this degree bound to prove the main theorem 2.1. This is done just the same way as last time, see the proof of Theorem 3.0.3 in the last lecture notes. We use our lemma 3.4 in place of the Lemma 3.0.2 there, and in the last inequality we get to bound $2\sum_S p_S$ by $2 \cdot OPT/k$ using Lemma **??**.