Lecture 8: Cuts and Distances

In this lecture, we will look at three cut problems and the algorithmic techniques that have been developed to tackle them. In all the examples below, we are given an undirected graph $G = (V, E)$. Each edge $e \in E$ has cost $c(e)$.

- **Minimum s,t-cut Problem.** We are input a pair of nodes s and t. The goal is to find the minimum cost set of edges F such that in the graph $(V, E \setminus F)$, s and t are disconnected. That is, F is an s,t-cut.
- **Multiway cut Problem.** We are input a collection of nodes $\{s_1, \ldots, s_k\}$, and the goal is to find a minimum cost set of edges F such that in the graph $(V, E \setminus F)$ each s_i lies in a different connected component.
- **Multicut Problem.** We are input a collection of pairs $\{(s_1, t_1), \ldots, (s_k, t_k)\}$ and the goal is to find a minimum cost set of edges F such that in the graph $(V, E \setminus F)$, each s_i is disconnected from its pair t_i. Note that s_i and s_j might lie in the same component.

Note that the problems are in increasing order of generality. In the next lecture we'll encounter another cut problem, the sparsest cut problem.

Cut induced distances.

Given the cut edges F, we can define a distance function d among all pairs of vertices as follows. Set $d(e) = 1$ for all $e \in F$ and $d(e) = 0$ for all $e \notin F$. For any pair (u, v), $d(u, v)$ is the shortest path with these edge lengths.

All of the cut problems above can be cast as finding a suitable "distance" function among the pairs of vertices in V. For instance, the multicut problem can be restated as follows.

$$\begin{array}{lll} \min & \sum_{e \in E} c(e)d(e) & d(e) \in \{0,1\} \\ \text{such that} & \text{distance induced by } d \text{ between } s_i \text{ and } t_i \ \geq 1 & \forall i = 1, ..., k \end{array}$$

One can move to a linear programming formulation by relaxing $d(e) \in [0, 1]$, and having a constraint for each s_i, t_i path p. Or one can write the following compact LP using the fact that d forms a distance and hence satisfies triangle inequality. For completeness, let us add all edges (u, v) not in E with $c(u, v)$ set to 0 for these edges. Then the LP is as follows

$$\begin{aligned} \min \quad & \sum_{e \in E} c(e)d(e) && d(e) \in [0,1] && (1) \\ \text{such that} \quad & d(s_i, t_i) \geq 1 && \forall i = 1...k && (2) \\ & d(u,v) \leq d(u,w) + d(w,v) && \forall u, v, w \in V && (3) \end{aligned}$$

The discussion above says that

Claim 1. *For the multicut problem,* $\texttt{opt} \geq \texttt{lp}$ *for the LP* (1)*.*

Let us see how the above can be used to give algorithms for the three problems above. We start with the s,t-min cut problem.

Minimum s,t Cut.
Minimum s,t-cut Algorithm.

1. Solve LP (1) with just one constraint of the form (2).
2. Pick r uniformly between $[0,1)$.
3. Let $S := \{v : d(s,v) \leq r\}$. Return the cut $F = \delta(S)$.

Note that the solution returned is a valid s,t-cut of cost `alg`, say.

Theorem 1. *The expected cost of the mincut algorithm is at most* `lp`*.*

Proof. Fix any edge $e = (u,v)$, let us calculate the probability $e \in F$. Without loss of generality, let $d(s,u) \leq d(s,v)$. Then $e \in F$ iff $d(s,u) \leq r < d(s,v)$. That is,

$$\mathbf{Pr}[e \in F] = \mathbf{Pr}[r \in [d(s,u), d(s,v))] = d(s,v) - d(s,u) \leq d(u,v)$$

where the second equality follows from the fact that r is chosen uniformly at random between $[0,1)$, and the second inequality follows from triangle inequality. Thus, $\mathbf{E}[\texttt{alg}] \leq \sum_{(u,v)} c(u,v)d(u,v) = \texttt{lp}$. □

Of course, the expected value of the algorithm cannot be strictly less than `lp`. In fact, this shows that any cut returned by the algorithm has value equal to `lp`.

Multiway Cut.

The algorithm for multiway cut is slightly different. Instead of randomly sampling from $[0,1)$, we will sample from $[0,1/2)$.

Minimum Multiway cut Algorithm.

1. Solve LP (1) with just constraints of the form (2) for every (s_i, s_j) pair.
2. Pick r uniformly between $[0,1/2)$.
3. Let $S_i := \{v : d(s_i,v) \leq r\}$. Return the cut $F = \bigcup_i^k \delta(S_i)$.

It should be clear the algorithm returns a valid multiway cut. This is because no S_i will contain an s_j since the distance $d(s_i, s_j) \geq 1$. Once again, fix an edge (u, v) and let us calculate the probability it is in the cut F. Here is an observation which will be useful.

Claim 2. *For any vertex u, there is at most one s_u such that $d(s_u, u) < 1/2$.*

Proof. All of this follow from triangle inequality. If there are two s_u and $s_{u'}$ with $d(s_u, u) < 1/2$ and $d(s_{u'}, u) < 1/2$, then $d(s_u, s_{u'}) < 1$ which is a contradiction. □

Lemma 1. *In the above multiway cut algorithm, for any edge (u, v), $\mathbf{Pr}[(u, v) \in F] \leq 2d(u, v)$.*

Proof. (u, v) is cut when 1) $u \in S_u$ and $v \notin S_u$ for some S_u, or 2) $v \in S_v$ and $u \notin S_v$ for some S_v. However, since $r < 1/2$, Claim 2 implies there is an unique S_u, if any, for which 1) is true and there is an unique S_v, if any, for which 2) is true. Furthermore, for 1) to occur, we must have $r \in [d(s_u, u), 1/2)$ and for 2) to occur we must have $r \in [d(s_v, v), 1/2)$. Thus, the probability that at least one of them occur is at most

$$\mathbf{Pr}[(u, v) \in F] \leq 2 \cdot (1/2 - d(s_u, u)) + 2 \cdot (1/2 - d(s_v, v)) = 2\,(1 - d(s_u, u) - d(s_v, v)) \tag{4}$$

Now, triangle inequality gives us $1 \leq d(s_u, s_v) \leq d(s_u, u) + d(u, v) + d(v, s_v)$. Putting this in (4), we get the lemma. □

The above lemma implies a 2-approximation. In fact, we can get a $2(1 - 1/k)$ approximation by taking the union of the cheapest $(k - 1)$ cuts rather than k of them.

Multicut. We will look at two algorithms for the multicut problem. The first will be in the spirit as we saw above using a random radius. The other will be a deterministic "region growing" algorithm. Let us start with the randomized idea. What should our algorithm be? Inspired by the mincut algorithm, let's say we run k rounds of it. That is, pick a radius $r \in [0, 1/2)$ and go over the pairs in some order. Each time let $S_i = \{u : d(s_i, u) \leq r_i\}$ and add $\delta(S_i)$ to F. We remove the edges (u, v) when u and v both lie in some S_j for $j < i$. We can do this because both s_i and t_i cannot lie in a set S_j since $r < 1/2$. F be the final subset of edges

Fix an edge $e = (u, v)$; what is the probability it is in F? When we go in the order defining S_i, three things can happen to this edge. Either u, v both lie in S_i, exactly one of these lie in S_i, or none of these two lie in S_i. We say (u, v) is *taken care of* by i if one of the first two occur. If the second occurs, we say (u, v) is cut by i. Note that if (u, v) is taken care of by any i, $(u, v) \notin F$: this will be crucial. So, (u, v) is cut by our algorithm, if there exists i such that i cuts (u, v), and none of the $j < i$ take care of (u, v).

To be precise, define $\mathcal{E}_i(u, v)$ to be the event that $d(s_i, u) \leq r$ and $d(s_i, v) > r$, and let $\mathcal{E}'_i(u, v)$ be the event that $r < \min(d(s_i, u), d(s_i, v))$. From our discussion in the previous paragraph, we get

$$\mathbf{Pr}[(u, v) \in F] = \mathbf{Pr}\left[\exists i :\ \mathcal{E}_i(u, v) \wedge \mathcal{E}'_{i-1}(u, v) \wedge \cdots \mathcal{E}'_1(u, v)\right] \tag{5}$$

For a fixed i, we have

$$\mathbf{Pr}\left[\mathcal{E}_i(u, v) \wedge \mathcal{E}'_{i-1}(u, v) \wedge \cdots \mathcal{E}'_1(u, v)\right] = \mathbf{Pr}\left[r \in [d(s_i, u), d(s_i, v)] \ \wedge\ r < d(s_j, u), r < d(s_j, v),\ \forall j < i\right]$$

Consider the following pathological case: $d(s_i, u) = 1 - (i/k)$ and $d(s_i, v) = 1 - ((i-1)/k)$. In this case, the latter probability in the above expression simply becomes $\mathbf{Pr}[r \in [d(s_i, u), d(s_i, v)]$ for all i. Union bound then gives the $\mathbf{Pr}[(u, v) \in F] \leq k \cdot d(u, v)$ – which would imply a k-approximation and nothing better. (In fact, the probability in (5) becomes 1, that is the edge (u, v) is cut with certainty).

The above example is pathological because of the order: if we had chosen a different order, say the order $\{k, k-1, \ldots, 1\}$, then with probability 1, k takes care of (u, v) and with probability $1/k$ it cuts it. This motivates the following algorithm which we call the CKR algorithm after Calinescu, Karloff, and Rabani (the inventors).

Minimum multicut algorithm (CKR)

1. Solve LP (1).
2. Pick r uniformly between $[0, 1/2)$. Let σ be a random permutation of $\{1, 2, \ldots, k\}$. Initialize $F \to \emptyset$
3. Let $S_i := \{u : d(s_i, u) \leq r\}$. Let $E[S_i] := \{(u, v) : u, v \in S_i\}$.
4. In the order of σ, add $\delta\left(S_{\sigma(i)}\right) \setminus \bigcup_{j<i} E[S_{\sigma(j)}]$ to F.

Claim 3. *F separates all s_i, t_i pairs.*

Proof. Note that when $\delta\left(S_{\sigma(i)}\right) \setminus \bigcup_{j<i} E[S_{\sigma(j)}]$ is added to F, $s_{\sigma(i)}$ disconnects from all vertices outside $S_{\sigma(i)}$, except maybe those in $S_{\sigma(j)} : j < i$ which contain $s_{\sigma(i)}$. (Those which don't contain $s_{\sigma(i)}$ are disconnected by induction.) However, in that case, they don't contain $t_{\sigma(i)}$. In any case, $s_{\sigma(i)}$ is disconnected from $t_{\sigma(i)}$.

□

Theorem 2. *The expected cost of the multicut returned by the above algorithm is at most $O(\log k)\mathtt{lp}$.*

Proof. Fix an edge (u, v). The proof of the theorem follows if we prove $\mathbf{Pr}[(u, v) \in F] = O(\log k) d(u, v)$. Note that the probability is now both over our choice of r *and* the random permutation of the terminals. Recall, $\mathcal{E}_i(u, v)$ is the event that $d(s_i, u) \leq r$ and $d(s_i, v) > r$, and $\mathcal{E}'_i(u, v)$ is event that $r < \min(d(s_i, u), d(s_i, v))$. As in the discussion above, we have

$$\mathbf{Pr}[(u, v) \in F] = \mathbf{Pr}_{\sigma, r}\left[\exists i : \ \mathcal{E}_{\sigma(i)}(u, v) \bigwedge_{j<i} \mathcal{E}'_{\sigma(j)}(u, v)\right] \tag{6}$$

Fix an i between 1 and k. Note that $\bigwedge_{j<i} \mathcal{E}'_{\sigma(j)}(u, v)$ occurs only if $r < d(s_{\sigma(j)}, u)$ for all $j < i$. But $\mathcal{E}_{\sigma(i)}(u, v)$ occurs only if $r \geq d(s_{\sigma(i)}, u)$. So, we can upper bound the probability in the RHS above as

$$\mathbf{Pr}_{\sigma, r}\left[\mathcal{E}_{\sigma(i)}(u, v) \bigwedge_{j:<i} \mathcal{E}'_{\sigma(j)}(u, v)\right] \leq \mathbf{Pr}_{\sigma, r}\left[r \in [d(s_{\sigma(i)}, u), d(s_{\sigma(i)}, v)) \ \bigwedge_{j<i} \{d(s_{\sigma(i)}, u) < d(s_{\sigma(j)}, u)\}\right]$$

Note that the two events in the second expression are independent: the first depends only on r, the second only on σ, they were chosen independently. So, we get

$$\mathbf{Pr}[(u,v) \in F] \leq \sum_{i=1}^{k} \left(\mathbf{Pr}_r \left[d(s_{\sigma(i)}, u) \leq r < d(s_{\sigma(i)}, v) \right] \cdot \mathbf{Pr}_\sigma \left[\bigwedge_{j<i} \{ d(s_{\sigma(i)}, u) < d(s_{\sigma(j)}, u) \} \right] \right)$$

We know $\mathbf{Pr}_r \left[d(s_{\sigma(i)}, u) \leq r < d(s_{\sigma(i)}, v) \right] \leq 2d(u,v)$. Let $N_i(u) := \{ j : d(s_j, u) < d(s_i, u) \}$ denote the terminals whose distance to u is smaller than that of s_i's. Let $n_i = |N_i|$. Note that the n_i's are a permutation of $0, 1, \ldots, (k-1)$. The probability $\mathbf{Pr}_\sigma \left[\bigwedge_{j:\sigma(j)<\sigma(i)} \{ d(s_{\sigma(i)}, u) < d(s_{\sigma(j)}, u) \} \right]$ is then $1/(n_{\sigma(i)} + 1)$; it happens only if $\sigma(i)$ lies first among the terminals $N_{\sigma(i)}(u) \cup \{\sigma(i)\}$ in the ordering σ. So, we get,

$$\mathbf{Pr}[(u,v) \in F] \leq \sum_{i=1}^{k} 2d(u,v)/(n_{\sigma(i)} + 1) = 2H_k \cdot d(u,v)$$

□

We now describe another algorithm for the multicut problem. This algorithm, which we'll call the GVY algorithm, is the first algorithm for the problem due to Garg, Vazirani and Yannakakis. This uses a technique called region growing which is a useful technique and has applications in other partitioning problems.

We start with a couple of definition. Given a solution to (1), and a parameter $r \in [0, 1/2)$, let $S_i(r) := \{u : d(s_i, u) \leq r\}$. Recall $\delta S_i(r) := \{(u,v) : u \in S_i(r), v \notin S_i(r)\}$ and $E[S_i(r)] = \{(u,v) : u, v \in S_i(r)\}$. Define

$$\texttt{Vol}_i(r) := \frac{\texttt{lp}}{k} + \sum_{(u,v) \in E[S_i(r)]} c(u,v) d(u,v) + \sum_{(u,v) \in \delta S_i(r)} c(u,v) \cdot (r - d(s_i, u)) \tag{7}$$

This denotes the total "lp-mass" in a ball of radius r around s_i.

Lemma 2. *(Region growing lemma) For every i, there exists a $r_i \in [0, 1/2)$ such that*

$$\sum_{(u,v) \in \delta S_i(r), v \notin S_i(r)} c(u,v) \leq 2 \ln(2k) \cdot \texttt{Vol}_i(r_i)$$

The GVY algorithm returns the set $F := \bigcup_{i=1}^{k} \delta S_i(r_i)$. The following shows another $O(\log k)$-approximation for multicut.

Lemma 3. $c(F) \leq 4 \ln(2k) \cdot \texttt{lp}$

Proof. $c(F) \leq \sum_{i=1}^{k} c(\delta S_i(r_i)) \leq (2 \ln(2k)) \sum_{i=1}^{k} \texttt{Vol}_i(r_i)$. Substituting the definition of $\texttt{Vol}_i(r_i)$, we get

$$c(F) \leq 2 \ln(2k) \left(\texttt{lp} + \sum_{(u,v) \in \cup_{i=1}^{k} E[S_i(r_i)]} c(u,v) d(u,v) + \sum_{i=1}^{k} \sum_{(u,v) \in \delta S_i(r_i)} c(u,v)(r_i - d(s_i, u)) \right)$$

Let F_1 be the edges belonging to exactly one $\delta S_i(r_i)$. F_2 be the edges belonging to two of these. For the first type of edges, we have the coefficient of $c(u,v)$ to be $(r_i - d(s_i,u)) \leq d(s_i,v) - d(s_i,u) \leq d(u,v)$. For the second type of edges, we have the coefficient of $c(u,v)$ to be $(r_i - d(s_i,u)) + (r_j - d(s_j,v)) \leq 1 - d(s_i,u) - d(s_j,v) \leq d(u,v)$. In all, we get $c(F) \leq 2\ln(2k) \cdot (\texttt{lp}+\texttt{lp}) = 4\ln(2k) \cdot \texttt{lp}$. $\square$

Proof of Region growing lemma: The crucial observation is that

$$\frac{d\texttt{Vol}_i(r)}{dr} = c(\delta S_i(r))$$

If the claim is false, then we would get $d\texttt{Vol}_i(r)/dr > 2\ln(2k) \cdot \texttt{Vol}_i(r)$. So,

$$\int_{\texttt{Vol}(0)}^{\texttt{Vol}(1/2)} \frac{d\texttt{Vol}(r)}{\texttt{Vol}(r)} > 2\ln(2k) \int_0^{1/2} dr$$

Integrating, we get $\ln\left(\frac{\texttt{Vol}(1/2)}{\texttt{Vol}(0)}\right) > \ln(2k)$ implying $\texttt{Vol}(1/2) > \frac{\texttt{lp}}{k} \cdot e^{\ln(2k)} = 2\texttt{lp}$. This is a contradiction since the total lp value cannot exceed $2\texttt{lp}$. $\square$