Lecture 14 (Oct. 25): Multi-Cut Problem

Lecturer: Mohammad R. Salavatipour *Scribe: Weitian Tong*

14.1 Multiway cut problem and a minimum-cut-based algorithm

Multiway Cut Problem

Input: A graph $G = (V, E)$ with an assignment of cost to each edge $c : E \to \mathbb{R}^+$ and a set of terminals $S = \{s_1, s_2, \ldots, s_k\} \subseteq V$.

Goal: Find a minimum-cost collection of edges that separate each s_i from other terminals.

Definition 1 *An s_i-cut is a set of edges that separates s_i from all other terminals.*

One greedy approach to solve this problem involves using minimum s_i-cut. If we remove the edges in any s_i-cut, we can separate s_i from other terminals.

Minimum-cut-based Algorithm

1. **for** $i \leftarrow 1$ to k **do**
2. Let C_i be a minimum s_i-cut.
3. Let C_k be the costliest cut among all the s_i-cuts, $i = 1, 2, \ldots, k$.
4. **return** $C = \cup_{i=1}^{k-1} C_i$.

Theorem 1 *The Minimum-cut-based Algorithm is a $(2 - \frac{2}{k})$-approximation algorithm.*

Proof. Let A be an optimal solution. Then G-A has at least k components with each s_i in one of them. Actually, G-A contains exactly k components, otherwise there must exist some component that contains no terminals and could obtain a smaller solution by not deleting the edges that separate this component from at least one other component. Suppose $G_1, G_2, \ldots, G_k$ are components of G-A. Let $A_i = \delta(G_i)$, which means $A = \cup_{i=1}^{k} A_i$. Of course, each A_i is an s_i-cut. Thus, we have $c(C_i) \leq c(A_i), i = 1, 2, \ldots, k$. Since each edge in A appears exactly two A_i's,

$$\sum_{i=1}^{k} c(C_i) \leq \sum_{i=1}^{k} c(A_i) = 2c(A) = 2OPT.$$

Note that $C = \cup_{i=1}^{k-1} C_i$ is also a feasible solution since for each $i \leq k-1$, C_i separate s_i from s_k. Because C_k is the costliest cut of $C_1, \ldots, C_k$, $c(C_k) \geq \frac{1}{k} \sum_{i=1}^{k} c(C_i)$, which means

$$\sum_{i=1}^{k-1} c(C_i) \leq (1 - \frac{1}{k}) \sum_{i=1}^{k} c(C_i) \leq (2 - \frac{2}{k}) OPT.$$

Hence, it is a $(2 - \frac{2}{k})$-approximation algorithm. ■

14.2 Multiway cut problem and an LP rounding algorithm

Now we introduce a better approximation algorithm for the multiway cut problem via LP rounding. Another way of looking at the multiway cut problem is finding an optimal partition of V, say $V_1, V_2, \ldots, V_k$, such that $s_i \in V_i, i = 1, 2, \ldots, k$ and the cost of $\cup_{i=1}^{k} \delta(V_i)$ is minimized.

To formulate the problem as an integer program, we need to define some sets of variables. For each vertex $v \in V$, we have k boolean variables x_v^i such that $x_v^i = 1$ if and only if v is assigned to the set V_i. For each edge $e \in E$, we create a boolean variable z_e^i such that $z_e^i = 1$ if and only if $e \in \delta(V_i)$. Since if $e \in \delta(V_i)$, it is also the case that $e \in \delta(V_j)$ for some $j \neq i$, the objective function of the integer program is then

$$\frac{1}{2} \sum_{e \in E} c_e \sum_{i=1}^{k} z_e^i.$$

Now we consider the constraints for the integer program. Obviously, we have $x_{s_i}^i = 1, i = 1, \ldots, k$ since each s_i must be assigned to V_i and we can also have $\sum_{i=1}^{k} x_u^i = 1$ for any vertex $u \in V$ since u must be contained in some V_i. Because for any edge $e = (u, v)$, $e \in \delta(V_i)$ if and only if exactly one of its endpoints is in V_i, we have $z_e^i \geq |x_u^i - x_v^i|$. Then the overall integer program is as follows:

$$\begin{aligned}
\text{minimize} \quad & \frac{1}{2} \sum_{e \in E} c_e \sum_{i=1}^{k} z_e^i & \\
\text{subject to} \quad & \sum_{i=1}^{k} x_u^i = 1, & \forall u \in V, \\
& z_e^i \geq x_u^i - x_v^i, & \forall e = (u, v) \in E, \\
& z_e^i \geq x_v^i - x_u^i, & \forall e = (u, v) \in E, \\
& x_{s_i}^i = 1, & i = 1, \ldots, k, \\
& x_u^i \in \{0, 1\}, & \forall u \in V, i = 1, \ldots, k.
\end{aligned} \tag{14.1}$$

Since the relaxed linear program of this integer program is closely related with the l_1-metric for measuring distances in Euclidean space, we give the definition of l_1-metric below.

Definition 2 *l_1-metric is a metric space where for any $x = (x^1, \ldots, x^n), y = (y^1, \ldots, y^n) \in \mathbb{R}^n$ the distance between them is $||x - y||_1 = \sum_{i=1}^{n} |x^i - y^i|$.*

Let Δ_k denote the $k-1$ dimensional simplex, that is, the surface in $\mathbb{R}^k$ defined by $\{x \in \mathbb{R}^k | x \geq 0 \ \& \ \sum_{i=1}^{k} x^i = 1\}$, where x is a vector and x^i is the ith coordinate of x. The LP relaxation will map each vertex of G to a point in Δ_k, and especially map each terminal to a unit vector. Let x_v represent the point to which vertex v is mapped. Thus, the relaxed linear program is as follows:

$$\begin{aligned}
\text{minimize} \quad & \frac{1}{2} \sum_{e=(u,v) \in E} c_e ||x_u - x_v||_1 & \\
\text{subject to} \quad & x_v \in \Delta_k, & \forall v \in V, \\
& x_{s_i} = e_i, & i = 1, \ldots, k,
\end{aligned} \tag{14.2}$$

For any $r \in [0, 1]$ and $1 \leq i \leq k$, let $B(s_i, r)$ be the set of vertices corresponding to the points x_v in a ball of radius r around s_i under the measure of l_1-metric, that is, $B(s_i, r) = \{v \in V | \ \frac{1}{2}||s_i - x_v||_1 \leq r\}$.

Randomized-LP-rounding Algorithm

1. Let x be an optimal LP solution to (14.2).
2. Pick $r \in (0,1)$ uniformly at random.
3. Pick a random permutation π of $\{1, 2, \ldots, k\}$.
4. **for** $i \leftarrow 1$ to $k-1$ **do**
5. $\quad V_{\pi(i)} \leftarrow B(S_{\pi(i)}, r) - \cup_{j<i} V_{\pi(j)}$
6. $V_{\pi(k)} = V - \cup_{j<k} V_{\pi(j)}$
7. **return** $\cup_{i=1}^{k} \delta(V_i)$

Theorem 2 *The randomized-LP-rounding algorithm is a $\frac{3}{2}$-approximation algorithm.*

To prove this theorem, we need to introduce some useful lemmas first.

Lemma 1 *$\forall u, v \in V$ and any index l, $|x_u^l - x_v^l| \leq \frac{1}{2}||x_u - x_v||_1$.*

Proof. Without loss of generality, assume that $x_u^l \geq x_v^l$. Then

$$\begin{aligned} |x_u^l - x_v^l| &= x_u^l - x_v^l = (1 - \sum_{j \neq l} x_u^j) - (1 - \sum_{j \neq l} x_v^j) \\ &= \sum_{j \neq l} (x_u^j - x_v^j) \\ &\leq \sum_{j \neq l} |x_u^j - x_v^j| \end{aligned}$$

Thus we have

$$2|x_u^l - x_v^l| \leq |x_u^l - x_v^l| + \sum_{j \neq l} |x_u^j - x_v^j| = \sum_{j=1}^{k} |x_u^j - x_v^j| = ||x_u - x_v||_1,$$

which implies $|x_u^l - x_v^l| \leq \frac{1}{2}||x_u - x_v||_1$. ■

Lemma 2 *$u \in B(s_i, r)$ if and only if $1 - x_u^i \leq r$.*

Proof.

$$\begin{aligned} u \in B(s_i, r) &\Leftrightarrow \frac{1}{2}||s_i - x_u||_1 \leq r \Leftrightarrow \frac{1}{2}\sum_{j=1}^{k} |x_u^j - x_v^j| \leq r \\ &\Leftrightarrow \frac{1}{2}\sum_{j \neq i} x_u^j + \frac{1}{2}(1 - x_u^i) \leq r \\ &\Leftrightarrow \frac{1}{2}(1 - x_u^i) + \frac{1}{2}(1 - x_u^i) \leq r \\ &\Leftrightarrow 1 - x_u^i \leq r. \end{aligned}$$

■

Lemma 3 *For each edge $e = (u, v)$, $\mathbf{Pr}$[e is in cut] $\leq \frac{3}{4}||x_u - x_v||_1$.*

Proof. We say that an index i *settles* edge (u, v) if i is the first index in the random permutation such that at least one of $u, v \in B(s_i, r)$. We say that an index i *cuts* edge (u, v) if exactly one of $u, v \in B(s_i, r)$. Let S_i be the event that i settles (u, v) and X_i be the event that i cuts (u, v). Thus, $\mathbf{Pr}[$ e is in cut$] = \sum_{i=1}^{k} \mathbf{Pr}[S_i \wedge X_i]$. Note that S_i depends on the random permutation, while X_i is independent of the randomized permutation.

By lemma 2, we have

$$\mathbf{Pr}[X_i] = \mathbf{Pr}[min(1 - x_u^i, 1 - x_v^i) \leq r < max(1 - x_u^i, 1 - x_v^i)] = |x_u^i - x_v^i|.$$

Let $l = argmin_i(min(1 - x_u^i, 1 - x_v^i))$, that is , s_l is the closest terminal to one of u, v. We can claim that any index $i \neq l$ cannot settle the edge $e = (u, v)$ if l comes before i in permutation π, since if at least one of $u, v \in B(s_i, r)$, then at least one of $u, v \in B(s_l, r)$. Note that the probability that l comes before i in the randomized permutation π is $\frac{1}{2}$. Hence for $i \neq l$, we have

$$\begin{aligned}
\mathbf{Pr}[S_i \wedge X_i] &= \mathbf{Pr}[S_i \wedge X_i | l >_\pi i]\mathbf{Pr}[l >_\pi i] + \mathbf{Pr}[S_i \wedge X_i | l <_\pi i]\mathbf{Pr}[l <_\pi i] \\
&= \frac{1}{2}\mathbf{Pr}[S_i \wedge X_i | l >_\pi i] + 0 \\
&\leq \frac{1}{2}\mathbf{Pr}[X_i | l >_\pi i]
\end{aligned}$$

Since the event X_i is independent of the randomized permutation, $\mathbf{Pr}[X_i | l >_\pi i] = \mathbf{Pr}[X_i]$ and therefore for $i \neq l$,

$$\mathbf{Pr}[S_i \wedge X_i] \leq \frac{1}{2}\mathbf{Pr}[X_i] = \frac{1}{2}|x_u^i - x_v^i|.$$

We also have that $\mathbf{Pr}[S_l \wedge X_l] \leq \mathbf{Pr}[X_l] \leq |x_u^l - x_v^l|$. Therefore, we have

$$\begin{aligned}
\mathbf{Pr}[e \text{ is in cut}] &= \sum_{i=1}^{k} \mathbf{Pr}[S_i \wedge X_i] \\
&\leq |x_u^l - x_v^l| + \frac{1}{2}\sum_{i \neq l} |x_u^i - x_v^i| \\
&= \frac{1}{2}|x_u^l - x_v^l| + \frac{1}{2}||x_u - x_v||_1 \\
&\leq \frac{1}{4}||x_u - x_v||_1 + \frac{1}{2}||x_u - x_v||_1 \qquad \text{By lemma 1} \\
&= \frac{3}{4}||x_u - x_v||_1
\end{aligned}$$

■

Now using the above three lemma, we can prove the theorem 2.

Proof. Let Z_{uv} be a boolean variable which is 1 if u and v are in different parts of the partition. Then the total cost of the cut returned by this algorithm is $W = \sum_{e=(u,v)\in E} c_e Z_{uv}$, which have the expectation

$$\begin{aligned}
E[W] &= E\left[\sum_{e=(u,v)\in E} c_e Z_{uv}\right] \\
&= \sum_{e=(u,v)\in E} c_e E[Z_{uv}] \\
&= \sum_{e=(u,v)\in E} c_e \mathbf{Pr}[e \text{ is in cut }] \\
&\leq \sum_{e=(u,v)\in E} c_e \frac{3}{4} ||x_u - x_v||_1 \\
&= \frac{3}{2} * \frac{1}{2} \sum_{e=(u,v)\in E} c_e ||x_u - x_v||_1 \\
&\leq \frac{3}{2} OPT
\end{aligned}$$

■