Linear Programming Algorithm for the Multiway Cut Problem

Adam Groce
18.434: Seminar in Theoretical Computer Science
Prof. M. X. Goemans

March 2, 2006

1 The Problem

Recall from last time the multiway cut problem: given a graph with weighted edges and a set of terminals $S = \{s_1, s_2, \ldots, s_k\} \subseteq V$, find the minimum weight set of edges $E' \subseteq E$ which, when removed, leaves all terminals separated from all other terminals. Last time, a combinatorial algorithm was given with an approximation factor of $2 - \frac{2}{k}$. This lecture will show a randomized linear programming algorithm with an approximation factor of $\frac{3}{2}$.

2 The Linear Programming Relaxation

Let Δ_k denote the $k-1$ dimensional simplex; that is, the surface in $\mathbb{R}^k$ defined by $\{x \in \mathbb{R}^k | x \geq 0 \text{ and } \sum_i x^i = 1\}$, 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. Each terminal will be mapped to a different unit vector. Let x_v represent the point to which vertex v is mapped. Define the length of an edge (u, v) to be

$$d(u, v) = \frac{1}{2} \sum_{i=1}^{k} |x_u^i - x_v^i|$$

Now consider the relaxation:

$$\begin{aligned}
\text{minimize} \quad & \sum_{(u,v)\in E} c(u,v)d(u,v) & \\
\text{subject to} \quad & d(u,v) = \frac{1}{2}\sum_{i=1}^{k} |x_u^i - x_v^i|, & (u,v) \in E \\
& x_v \in \Delta_k, & v \in V \\
& x_{s_i} = e_i, & s_i \in S
\end{aligned}$$

An integer solution to this relaxation maps each vertex of G to a unit vector. Each vertex represents a component of the graph after E' is removed. Edges within one component have length 0, and edges between components (i.e., those in E') have length 1. The function being minimized is therefore equal to the cost of E'.

It is not clear that the above is a true linear program, due to the absolute values. However, this is not a problem. To create an equivalent true linear program, replace the first constraint with:

$$\begin{aligned}
x_{uv}^i &\geq x_u^i - x_v^i,\ 1 \leq i \leq k \\
x_{uv}^i &\geq x_v^i - x_u^i,\ 1 \leq i \leq k \\
d(u,v) &= \frac{1}{2}\sum_{i=1}^{k} x_{uv}^i
\end{aligned}$$

Because of the minimization, any optimal solution must satisfy $x^i_(uv) = |x_u^i - x_v^i|$.

We may assume, without loss of generality, that for each edge $(u,v) \in E$, x_u and x_v differ in at most two coordinates.

Proof. Along any edge (u,v) where x_u and x_v differ in more than two coordinates, insert a new vertex w and replace (u,v) with (u,w) and (w,v). Assign both (u,w) and (w,v) the same cost as (u,v). This does not change the cost of the optimal integral solution.

Now consider the optimal fractional solution. Since d is a valid distance function, $d(u,w) + d(w,v) \geq d(u,v)$. Therefore the cost of the optimal solution cannot decrease because of the addition of w. Now, let i be the coordinate in which the difference between x_u^i and x_v^i is minimal (disregarding coordinates where $x_u^i = x_v^i$). Without loss of generality assume $x_u^i < x_v^i$ and let $\alpha = x_v^i - x_u^i$. There must be a coordinate j such that $x_u^j \geq x_v^j + \alpha$. Consider the solution with $x_w^i = x_u^i$ and $x_w^j = x_v^j + \alpha$. All other coordinates of x_w are equal to those of x_v. This gives $x_w \in \Delta_k$ and $d(u,v) = d(u,w) + d(w,v)$.

x_v and x_w differ in only two coordinates, and x_w and x_u differ in fewer coordinates than x_u and x_v. Repeated application of this process will give a solution with the same cost and with the desired property. $\square$

3 The Algorithm

Take an optimal solution to the relaxation with edges whose endpoints differ in at most two coordinates, and let OPT denote its cost. Define $E_i = (u,v) \in E | x_u^i \neq x_v^i$. (Note that each edge will lie in two of these sets.) Define $W_i = \sum_{e \in E_i} c(e)d(e)$. Without loss of generality, assume that W_k is the greatest of $W_1, \ldots, W_k$. Also define $B(s_i, \rho) = v \in V | x_v^i \geq \rho$.

The algorithm operates as follows. First, pick ρ at random in $(0,1)$ and an ordering σ from $(1, 2, \ldots, k-1, k)$ and $(k-1, k-2, \ldots, 1, k)$. Then partition V into $V_1, \ldots, V_k$ as follows. Proceed in the order given by σ. Each V_i should contain all vertices in $B(s_i, \rho)$ that have not already been assigned to a previous V_i. At the end, assign all unused vertices to V_k. The sets $V_1, \ldots, V_k$ are the components after removing the cut, and edges between vertices in two different sets are in the cut.

More formally, the algorithm is:

1. Compute an optimal solution to relaxation.

2. Renumber the terminals so that W_k is largest among $W_1, \ldots, W_k$.

3. Pick uniformly at random $\rho \in (0,1)$ and
   $\sigma \in (1, 2, \ldots, k-1, k), (k-1, k-2, \ldots, 1, k)$.

4. For $i = 1$ to $k-1$: $V_{\sigma(i)} \leftarrow B(s_i, \rho) - \bigcup_{j<i} V_{\sigma(j)}$.

5. $V_k \leftarrow V - \bigcup_{i<k} V_i$.

6. Let C be the set of edges that run between sets in the partition
   $V_1, \ldots, V_k$. Output C.

4 Proof of the Approximation Factor

Let C be the cut produced by the algorithm, $c(C)$ be the cost of C, and OPT be the cost of the optimal solution to the linear program. We will show that $E[c(C)]$, the expected value of $c(C)$, is at most $(1.5 - \frac{1}{k}) \times OPT$.

Lemma 1. *If $e \in E_k$, then $Pr[e \in C] \leq d(e)$.*

Proof. The endpoints of E differ in coordinates i and k. Since V_k is determined without considering the coordinates of the points left over, and all coordinates except i and k are equal, the only way that endpoints u and v will end up in different sets is if one (but not the other) is in V_i. This occurs if and only if ρ is between x_u^i and x_v^i. This has probability $d(e)$. □

Lemma 2. *If $e \in E - E_k$, then $Pr[e \in C] \leq 1.5d(e)$.*

Proof. The endpoints of E, u and v, differ in coordinates i and j. Let β be the interval $[x_u^j, x_v^j]$ and let α be the part of $[x_u^i, x_v^i]$ that does not overlap with β. u and v can each end up in either V_i, V_j, or V_k. Assume without loss of generality that α is closer to 0 than β. (If not, switching the values of i and j makes it so.) u and v end up in different sets if and only if $\rho \in \beta$, or $\rho \in \alpha$ and $\sigma(i) < \sigma(j)$. Therefore $\Pr[e \in C] = |\beta| + \frac{|\alpha|}{2} \leq 1.5d(e)$ (because $|\alpha| \leq |\beta| = d(e)$). □

Theorem 1. *$E[c(C)] \leq (1.5 - \frac{1}{k}) \times OPT$.*

Proof. First, note that $\sum_{i=1}^{k} W_i = 2 \cdot OPT$, and W_k was chosen to be the greatest, so $W_k \geq \frac{2}{k} \cdot OPT$.

$$\begin{aligned}
E[c(C)] &= \sum_{e \in E} c(e)\Pr[e \in C] = \sum_{e \in E - E_k} c(e)\Pr[e \in C] + \sum_{e \in E_k} c(e)\Pr[e \in C] \\
&\leq 1.5 \sum_{e \in E - E_k} c(e)d(e) + \sum_{e \in E_k} c(e)d(e) = 1.5 \sum_{e \in E} c(e)d(e) - 0.5 \sum_{e \in E_k} c(e)d(e) \\
&\leq 1.5 \cdot OPT - 0.5 \left(\frac{2}{k} \cdot OPT\right) \\
&\leq (1.5 - \frac{1}{k}) \cdot OPT
\end{aligned}$$

□