CS 251 - Problem 16.8

Sarah Eisenstat (seisenst)

October 31, 2008

Max k-Cut

- **Input:** Graph $G = (V, E)$, edge weights w_e, integer k.
- **Output:** A partition of V into sets $S_1, S_2, \ldots S_k$.
- **Goal:** Maximize the sum of the weights of the edges between the different partitions.

Randomized Algorithm

We consider the following randomized algorithm for Max k-Cut:

1. initialize $S_1, \ldots S_k$ to $\emptyset$
2. for each vertex v_i in V:
   (a) pick a number j uniformly at random from $\{1, \ldots, k\}$
   (b) add v_i to the set S_j
3. output $S_1, \ldots, S_k$

What is the expected value of this algorithm? Well, for each edge e, let W_e be a random variable representing the amount of weight that edge e contributes to the final value. Then we have:

$$\mathrm{E}\left[COST\right] = \mathrm{E}\left[\sum_{e \in E} W_e\right] = \sum_{e \in E} \mathrm{E}\left[W_e\right]$$

What is $\mathrm{E}\left[W_e\right]$? Well, if $e = \{u, v\}$, then we know that $W_e = 0$ if and only if v ends up in the same set as u. Because v's set is chosen uniformly at random, the probability that v ends up in the same set as u is $1/k$. Therefore:

$$\mathrm{E}\left[COST\right] = \sum_{e \in E} w_e \left(1 - \frac{1}{k}\right) = \left(1 - \frac{1}{k}\right) \cdot \sum_{e \in E} w_e$$

Because $k \geq 2$, and $\sum_{e \in E} w_e \geq OPT$, we know that:

$$\mathrm{E}\left[COST\right] \geq \left(1 - \frac{1}{k}\right) \cdot OPT \geq \frac{1}{2} \cdot OPT$$

Derandomization

What happens when we derandomize this algorithm? It becomes the following:

1. initialize $S_1, \ldots S_k$ to $\emptyset$
2. for each vertex v_i in V:
   (a) set j^* to 0
   (b) for each j in $\{1, \ldots k\}$:
      i. calculate $\alpha_j = \mathrm{E}\left[COST \mid S_1, \ldots, S_j \cup \{v_i\}, \ldots, S_k\right]$
      ii. if $j^* = 0$ or $\alpha_j > \alpha_{j^*}$, set $j^* = j$
   (c) add v_i to S_{j^*}
3. output $S_1, \ldots, S_k$

What sort of bounds does this algorithm give us? Well, at every point, we are choosing to put v_i in the set S_j that maximizes α_j. So for the particular j^* that we pick, we know that $\alpha_{j^*} \geq \alpha_j$. Because of the way that we've calculated the α values, we have:

$$\mathrm{E}\left[COST \mid S_1, \ldots, S_k\right] = \frac{1}{k}\left(\alpha_1 + \ldots + \alpha_k\right) \leq \frac{1}{k} \cdot k \cdot \alpha_{j^*} = \alpha_{j^*}$$

Hence, the expected cost only increases as we go. This means that the output of this derandomized algorithm has cost greater than or equal to the expected cost of the randomized algorithm. Therefore, by our previous analysis, the result is better than $(1 - 1/k) \cdot OPT$.

How does this compare to the natural greedy algorithm? Well, at any given point, the vertex set can be partitioned into three sets: A, the vertices that have already been assigned to groups; B, the current vertex v_i; and C, the rest of the vertices. For each choice of j, let E_j be the event that we have divided $v_1, \ldots, v_{i-1}$ into some fixed partitions $S_1, \ldots, S_k$, and that we have chosen partition S_j as the location for vertex v_i. Then we can break up each α_j into three pieces:

$$\begin{aligned}\alpha_j = \mathrm{E}\left[COST \mid E_j\right] = \mathrm{E}\left[\sum_{u,v \in V} W_{u,v} \mid E_j\right] &= \sum_{u,v \in V} \mathrm{E}\left[W_{u,v} \mid E_j\right] \\ = \sum_{u,v \in A} \mathrm{E}\left[W_{u,v} \mid E_j\right] + \sum_{u \in A, v \in B} \mathrm{E}\left[W_{u,v} \mid E_j\right] &+ \sum_{u \in V, v \in C} \mathrm{E}\left[W_{u,v} \mid E_j\right]\end{aligned}$$

The first part of that sum is fixed, because all of the vertices in A have already been partitioned. Therefore, since we're choosing our j based on the largest α_j, we can safely ignore this part of the sum, because it occurs in all α_j's.

How about the third part of the sum? Well, because C is the set of vertices whose location has not been fixed yet, we know that for each edge $\{u, v\}$, v has probability $1/k$ of being in the same group as u. Therefore,

$$\sum_{u \in V, v \in C} \mathrm{E}\left[W_{u,v} \mid E_j\right] = \sum_{u \in V, v \in C} \left(1 - \frac{1}{k}\right) w_{u,v}$$

Note that this is the same regardless of where v_i is located, so once again we know that this part of the sum does not affect our choice of the j maximizing α_j.

So the only part of the sum that would influence our choice of a location for v_i is:

$$\sum_{u \in A, v \in B} \mathrm{E}\left[W_{u,v} \mid E_j\right]$$

In the event E_j, the locations for both A and B have already been chosen. Therefore:

$$\sum_{u \in A, v \in B} \mathrm{E}\left[W_{u,v} \mid E_j\right] = \sum_{u \in A \setminus S_j} w_{u,v_i}$$

Hence, by picking the j that maximizes α_j, we are picking the j that maximizes the cost of the edges between v_i and the elements of other groups. This is the straightforward greedy algorithm sought in Problem 2.3.