CS 251 - Problem 5.3

Sarah Eisenstat (seisenst)

September 26, 2008

Definitions

Clique Cover

Clique cover, a decision problem asking whether it's possible to cover a graph with k cliques, is a problem that is known to be **NP**-complete. Formally, it is defined as follows:

- **Input:** A graph $G = (V, E)$ and an integer k.
- **Output:** "YES" if there exists a way to partition G into k complete subgraphs; "NO" otherwise.

Algorithm

Let $D = \{c_{u,v} \mid u, v \in V\}$. Then let $d_1 < d_2 < \ldots < d_m$ be the distinct elements of D in increasing order. (In other words, sort them.) We iterate through the d_i's in increasing order, and do the following for each d_i:

Construct a graph G_i for each d_i such that $G_i = (V, E_i)$, where E_i consists of all edges $\{u, v\}$ with cost $c_{u,v} \leq d_i$. Then construct a maximum independent set I_i for the graph G_i. If $|I_i| > k$, then we move on to the next G_i. Otherwise, we construct a k-clustering of the graph from I_i as follows:

Let $V' = V \backslash I_i$. Iterate over each $v_j \in I_i$, selecting all members of V' which are neighbors of v_j in G_i to construct the cluster V_j. In other words, let $V_j = \{v_j\} \cup \{u \in V' \mid \{u, v_j\} \in E_i\}$. Then let $V' = V' \backslash V_j$, and repeat for the next j. Clearly, this algorithm produces a k-cluster. But we must analyze the results to determine what kind of bound this approximation algorithm offers.

Analysis

Let's start from the optimal solution — a division into clusters so that the maximum value edge in each cluster is less than or equal to the value OPT. Because the value of the clustering must be one of the edge weights c_e, we know that it must be equal to d_i for some i. Consider the structure of G_i for that particular i. Because we have a k-clustering, we know that for each edge e in each cluster, $c_e \leq OPT = d_i$. This means that there is an edge in G_i for each one of those edges e. In other words, the k-clustering in G corresponds to a clique cover of size k in G_i.

So what does this mean? Well, if we have a clique cover of size k in G_i, then any independent set must have size $\leq k$, because it can contain at most one member of each of the cliques. Therefore, no matter how I_i is selected, $|I_i| \leq k$, and so our algorithm will construct a clustering using I_i.

What is the value of this result? Well, we know that for each V_j constructed by the algorithm, all of the elements are neighbors of v_j in the graph G_i. But there can only be an edge in G_i if the original edge in G has weight $\leq OPT$. That means that all members of V_j are distance $\leq OPT$ from v_j.

But what is the distance between two $u, w \in V_j$, where neither u nor w is equal to v_j? Well, by the triangle inequality, $c_{u,w} \leq c_{u,v_j} + c_{v_j,w} \leq 2 \cdot OPT$. Therefore, in each cluster, the longest edge is at most $2 \cdot OPT$. Hence, the value of our clustering is at most $2 \cdot OPT$.

Bound

We wish to prove that there does not exist a $(2 - \epsilon)$-approximation for k-clustering, for any $\epsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$. We will do so by reducing from clique cover, a problem that is known to be NP-complete.

Assume, for the sake of contradiction, that there exists a polynomial time algorithm A that is a $(2 - \epsilon)$-approximation algorithm for k-clustering. For our reduction, we will start with an instance of the clique cover problem — that is, a graph $G = (V, E)$ and some integer k. We wish to determine whether there exists a way to partition V into k subgraphs such that each subgraph is a complete graph.

We construct an instance of the k-clustering problem as follows:

$$\begin{aligned} \text{Graph:} \quad & G' = (V, V \times V) \\ \text{Edge weights:} \quad & c_{u,v} = 1 \text{ if } \{u, v\} \in E,\ c_{u,v} = 2 \text{ otherwise} \\ \text{Integer:} \quad & k \end{aligned}$$

We then run A on this instance of the k-clustering problem. What are the possible results? Well, the only two edge weights are 1 and 2, so it must return one of those two values.

- **Case 1:** If it returns 1, then we know that there exists a way to divide the graph up into k subgraphs such that the longest edge in each subgraph is at most 1. But because of the way we have constructed the edge weights, we know that that means that all of the edges in each of the k subgraphs also existed in the original graph G — in other words, we have found a way to partition the graph into k sections such that each section is a complete graph. Hence, we return "YES" for the original clique cover problem.

- **Case 2:** If it returns 2, then we know that $2 \leq (2-\epsilon) \cdot OPT$. That means that $OPT \geq \frac{2}{2-\epsilon}$, so $OPT > 1$. This means that there is no way to partition G' into k clusters such that each cluster contains edges of maximum length 1. Because edge lengths are 1 if they existed in the original graph G, this means that there is no way to divide up G into k groups such that each subgraph is a complete graph. Thus, we return "NO" for the original clique cover problem.

Clearly, this results in an algorithm for solving clique cover. However, we know that clique cover is an **NP**-complete problem. Therefore, since this algorithm is polynomial-time, we know that unless **P** = **NP**, we have a contradiction. Hence, our assumption was wrong, and there exists no $(2-\epsilon)$-approximation for k-clustering.