CSCI3160: Regular Exercise Set 10

Prepared by Yufei Tao

Problem 1. Consider a complete bipartite graph $G = (V, E)$:

- V has $2n$ vertices, including n black vertices and n white vertices.
- E has n^2 edges, including an edge between every black vertex and every white vertex.

Use G to explain why 2 is the best the approximation ratio that we can prove for the vertex cover algorithm discussed in our lecture.

Solution. It is easy to verify that our vertex cover algorithm picks all the $2n$ vertices. An optimal solution, however, should include only n vertices (e.g., all the black ones).

Problem 2*. Let $G = (V, E)$ be an input graph to the vertex cover problem. If G is a tree, describe an $O(|V|)$-time algorithm that finds an optimal vertex cover of G.

(Hint: Dynamic programming.)

Solution. Root the tree G at an arbitrary node. For each node u of the tree, define $T(u)$ the subtree rooted at u. In addition, define

- $\mathrm{OPT}(u, \mathit{yes})$ as the size of an optimal vertex cover of $T(u)$, provided that u belongs to the vertex cover.
- $\mathrm{OPT}(u, \mathit{no})$ as the size of an optimal vertex cover of $T(u)$, provided that u does not belong to the vertex cover.

If u is a leaf, then

$$\begin{aligned} \mathrm{OPT}(u, \mathit{yes}) &= 1 \\ \mathrm{OPT}(u, \mathit{no}) &= 0. \end{aligned}$$

If u is an internal node, then

$$\begin{aligned} \mathrm{OPT}(u, \mathit{yes}) &= 1 + \sum_{\text{child } v \text{ of } u} \min\{\mathrm{OPT}(v, \mathit{yes}), \mathrm{OPT}(v, \mathit{no})\} \\ \mathrm{OPT}(u, \mathit{no}) &= \sum_{\text{child } v \text{ of } u} \mathrm{OPT}(v, \mathit{yes}) \end{aligned}$$

Let r be the root of G. It is now rudimentary to compute $\mathrm{OPT}(r, \mathit{yes})$ and $\mathrm{OPT}(r, \mathit{no})$ in $O(|V|)$ time (go through the nodes in a bottom-up order). The optimal vertex cover size is $\min\{\mathrm{OPT}(r, \mathit{yes}), \mathrm{OPT}(r, \mathit{no})\}$. To obtain an optimal vertex, apply the piggyback technique.

Problem 3.** Prof. Goofy proposes the following algorithm to find a vertex cover of $G = (V, E)$:

algorithm max-deg-VC
Input: $G = (V, E)$
$S = \emptyset$
while E not empty **do**
    $v \leftarrow$ a vertex with the maximum degree in the current G
    add v to S
    remove from E all the edges of v

Show that the approximation ratio of this algorithm is greater than 2.

Solution. Let us construct a bipartite graph G as follows. The set L of left vertices is $\{u_1, u_2, ..., u_{16}\}$. To generate the right vertices, for each $i \in [2, 16]$, we create a group R_i which contains $s_i = \lfloor 16/i \rfloor$ vertices, denoted as $R_i[1]$, $R_i[2]$, ..., and $R_i[s_i]$, respectively. The set R of right vertices is the union of R_2, R_3, ..., R_{16}. The size of R is $\sum_{i=2}^{16} s_i = 34$.

Generate the edges of G as follows: for each group $i \in [2, 16]$, connect $R_i[j]$ ($j \in [1, s_i]$) to the i vertices $u_{i(j-1)+1}, u_{i(j-1)+2}, ..., u_{ij}$.

Running Prof. Goofy's algorithm, you will see that it picks all the 34 right vertices. As an optimal solution, we can pick the 16 left vertices.

Problem* 4 (Max-Cut). Let $G = (V, E)$ be a simple undirected graph. Given a subset $S \subseteq V$, a *cut* induced by S is the set of edges $e \in E$ such that e has a vertex in S and another vertex in $V \setminus S$. Let OPT_G be the maximum size of a cut that can be induced by any $S \subseteq V$. Design a poly($|V|$)-time (i.e., polynomial time in $|V|$) algorithm that returns a cut of size at least $\mathrm{OPT}_G/2$ in expectation.

(Hint: Random assignment.)

Solution. Start with an empty S. For each vertex $u \in S$, toss a fair coin. If the coin comes up heads, add u to S; otherwise, leave u out of S. It is easy to prove that each edge $\{u, v\} \in E$ contributes to the cut induced by S with probability 1/2. Hence, the cut has size $|E|/2$ in expectation, which is at least $\mathrm{OPT}_G/2$ by the trivial fact $|E| \geq \mathrm{OPT}_G$.