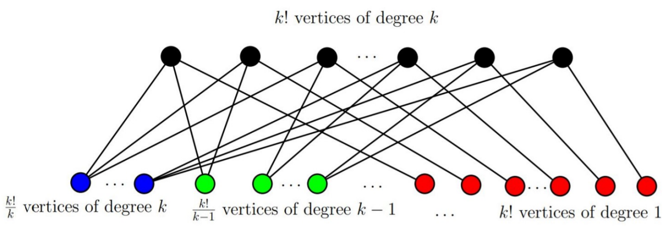
School of Computer Science and Technology
University of Science and Technology of China

Exercise Sheet 3 for
Design and Analysis of Algorithms
Autumn 2022

Due 28 Oct 2022 at 16:59

Exercise 1 *30*

Consider the algorithm AnotherGreedyVC given in Lecture 8 for the minimum vertex cover problem. Use the following example to show that the approximation ratio of AnotherGreedyVC is $\omega_n(1)$, where n is the number of vertices in the input graph.



Solution:

- According to the given example, $OPT = k!$ which contains all black vertices.
- The worst case is we choose the bottom vertices $C = k!(1 + \frac{1}{2} + ... + \frac{1}{k})$.
- $\frac{C}{OPT} = (1 + \frac{1}{2} + ... + \frac{1}{k})$, and the total number of vertices is $n = k!(1 + 1 + \frac{1}{2} + ... + \frac{1}{k})$
- Therefore, $\frac{C}{OPT} = \omega_n(1)$

Exercise 2 *30*

Consider the set cover problem. Let U be a set of n elements. Let $\mathcal{S} = \{S_1, \ldots, S_m\}$ be a collection of subsets of U such that $\cup_{i=1}^{m} S_i = U$. Our goal is to select as few subsets as possible from $\mathcal{S}$ such that their union covers U.
Consider the following algorithm SetCover for this problem. The algorithm takes as input U and $\mathcal{S}$, and does the following:

(a) Initialize $C = \emptyset$.

(b) While U contains elements not covered by C:

  (i) Find the set S_i containing the greatest number of uncovered elements

  (ii) Add S_i to C.

To analyze the above algorithm, let $k = \text{OPT}$ be the number of sets in the optimal solution. Let $E_0 = U$ and let E_t be the set of elements not yet covered after step t.

(a) Show that $|E_{t+1}| \leq |E_t| - |E_t|/k$.

(b) Show that the algorithm SetCover is a $(\ln n)$-approximation algorithm for the set cover problem.

Hint: Show that the algorithm SetCover finishes within OPT $\cdot \ln n$ steps.
Solution:

(a) According to the definition, the optimal solution sets must contains the E_t when step t, and in this time, the number of solution sets are most k, we can set it k'. Therefore, there is at least one set which not yet select has the number of not covered elements is $|E_t|/k'$
So, we can get $|E_{t+1}| \leq |E_t|/k' \leq |E_t| - |E_t|/k$

(b) As we proved in (a), $|E_{t+1}| \leq |E_t| - |E_t|/k \leq |E_{t-1}|(1-1/k)^2 \leq |E_{t-2}|(1-1/k)^3 \leq ... \leq |E_0|(1-1/k)^t$
Set$|E_0|(1-1/k)^t = n \exp^{-t/k} \leq 1$, which means when the algorithm runs the $t+1$ steps, we can achieve the solution.
$n \exp^{-t/k} \leq 1$, $n \leq \exp^{t/k}$, $lnn \leq t/k$, $t \geq klnn$. Therefore, the algorithm SetCover is a $(\ln n)$-approximation algorithm for the set cover problem.

Exercise 3 *40*
Consider the max cut problem. Given an undirected n-vertex graph $G = (V, E)$ with positive integer edge weights w_e for each $e \in E$, find a vertex partition $(A, \bar{A})$ such that the total weight of edges crossing the cut is maximized, where $\bar{A} = V \setminus A$ and the weight of $(A, \bar{A})$ is defined to be $w(A, \bar{A}) := \sum_{u \in A, v \in \bar{A}} w_{uv}$.

Consider the following algorithm MaxCut.

(a) Start with an arbitrary partition of V.

(b) Pick a vertex $v \in V$ such that moving it across the partition would yield a greater cut value.

(c) Repeat step (b) until no such v exists.

Now analyze the performance guarantee of the algorithm.

(a) Suppose that the maximum edge weight is $\lceil n^{10} \rceil$. Show that the algorithm runs in polynomial time.

(b) Let $(S, \bar{S})$ be partition output by the algorithm MaxCut. Show that for any vertex $v \in S$, it holds that

$$\sum_{u \in \bar{S}, (u,v) \in E} w_{u,v} \geq \frac{1}{2} \sum_{u:(u,v) \in E} w_{u,v}$$

(c) Show that the algorithm is MaxCut is a 1/2-approximation algorithm for the max cut problem.

Hint: Use the fact that $\sum_{e \in E} w_e \geq$ OPT, where OPT is the total weight of the optimal solution.
Solution:

(a) According to the definition, the algorithm will check at most every edge of the graph. So the complexity of the running time is $O(|E|)$

(b) According to the algorithm step (b), the weights sum of max-cut which is the output of the algorithm is not less than the weights sum of other edges ($u \in V, v \in S$), otherwise, the algorithm can continue run the step (b) and not stop. Therefore, for any vertex $v \in S$, it holds that

$$\sum_{u \in \bar{S}, (u,v) \in E} w_{u,v} \geq \frac{1}{2} \sum_{u:(u,v) \in E} w_{u,v}$$

(c) As we proved in (b),

$$\sum_{u \in \bar{S}, v \in S, (u,v) \in E} w_{u,v} \geq \frac{1}{2} \sum_{u \in V, v \in S, (u,v) \in E} w_{u,v}$$

If we sum over all vertices,

$$\sum_{v} \sum_{u \in \bar{S}, v \in S, (u,v) \in E} w_{u,v} \geq \frac{1}{2} \sum_{v} \sum_{u \in V, v \in S, (u,v) \in E} w_{u,v}$$

The left hand side is exactly twice the value of the cut, while the right hand side (sum of degree cuts) counts every edge twice.

$$2W_{cut} \geq \frac{1}{2}(2 \times \sum_{e \in E} w_e) \geq OPT$$

So we get $W_{cut} \geq \frac{1}{2}OPT$. The algorithm is MaxCut is a 1/2-approximation algorithm for the max cut problem.
