CS 466/666 Fall 2018 due Tuesday November 20, 5 PM

A. Lubiw

ASSIGNMENT 7

DO NOT COPY. ACKNOWLEDGE YOUR SOURCES.

1. [10 marks] The point of this question is to prove that if there is a polynomial-time constant-factor approximation algorithm for the Maximum Clique problem then there is a PTAS. Recall that the Maximum Clique problem (as an optimization problem) is to find the largest subset C of vertices such that every pair of vertices in C is joined by an edge in the graph.

   (a) [4] For a graph $G = (V, E)$ on n vertices define the k-th power of G, denoted G^k, as follows. The vertex set of G^k consists of the k-tuples $(v_1, v_2, \ldots, v_k)$ where $v_i \in V$. Note that G^k has n^k vertices. There is an edge between $(v_1, v_2, \ldots, v_k)$ and $(w_1, w_2, \ldots, w_k)$ if and only if for all $i = 1..k$, either $v_i = w_i$ or there is an edge (v_i, w_i) in G. (You will understand this better by making some small examples.)

   Prove that G has a clique of size t if and only if G^k has a clique of size t^k.

   (b) [6] Suppose there is a $\frac{1}{2}$-approximation algorithm A for the Maximum Clique problem that runs in time $O(p(n))$ where n is the number of vertices of the input graph and p is a polynomial function. Using part (a) and algorithm A, give a PTAS for the Maximum Clique problem. The input for your PTAS will be a graph G on n vertices and a value $\epsilon > 0$. What is the run-time of your PTAS, in terms of n, ϵ and $p(\ )$?

   (In fact, your argument will work for any constant-factor approximation.)

2. [10 marks] Recall the online bin packing problem: Given numbers $s_1, \ldots, s_n$ one-by-one, where $0 \le s_i \le 1$, pack them into a minimum number of unit-capacity bins. The decision of what bin to put item s_i in must be made without knowledge of items $s_{i+1}, \ldots, s_n$. In class we saw an algorithm, First Fit, that packs the items into at most 2OPT+1 bins, where OPT is the minimum number of bins required for the items (assuming perfect information about all the items).

   In this problem, we will suppose that the bins are also "online" in the sense that we pack items into the current bin until some item does not fit, then we close that bin and start a new bin. More precisely, the algorithm is:

   $j \leftarrow 1$

   for $i = 1 \ldots n$

       if s_i fits in bin j, put it in

       else $j \leftarrow j + 1$ and put s_i in bin j

   Prove that this algorithm uses at most 2OPT+1 bins, i.e., it is an asymptotic 2-approximation.

   Notation: let m be the number of bins used by the algorithm. Let b_j, $1 \le j \le m$, be $\sum\{s_i : s_i \text{ is placed in bin } j \text{ by the algorithm}\}$. Hint: Start by showing that $b_j + b_{j+1} > 1$.