Neatly print your name(s), NetID(s), and the alias(es) you used for Homework 0 in the boxes above. Please also tell us whether you are an undergraduate or a grad student by circling U or G, respectively. Staple this sheet to the top of your homework.
It was the best of times, it was the worst of times, it was the age of wisdom, it was the age of foolishness, it was the epoch of belief, it was the epoch of incredulity, it was the season of Light, it was the season of Darkness, it was the spring of hope, it was the winter of despair, we had everything before us, we had nothing before us, we were all going direct to Heaven, we were all going direct the other way—in short, the period was so far like the present period, that some of its noisiest authorities insisted on its being received, for good or for evil, in the superlative degree of comparison only.

— A Tale of Two Cities, Charles Dickens

**Required Problems**

1. **Rectangles are Forever.**

   **[20 Points]**

   A rectangle in the plane $r$ is called *neat*, if the ratio between its longest edge and shortest edge is bounded by a constant $\alpha$. Given a set of rectangles $R$, the induced graph $G_R$, has the rectangles of $R$ as vertices, and it connect two rectangles if their intersection is not empty.

   (a) **[5 Points]** (hard?) Given a set $R$ of $n$ neat rectangles in the plane (not necessarily axis parallel), describe a polynomial time algorithm for computing an independent set $I$ in the graph $G_R$, such that $|I| \geq \beta |X|$, where $X$ is the largest independent set in $G_R$, and $\beta$ is a constant that depends only on $\alpha$. Give an explicit formula for the dependency of $\beta$ on $\alpha$. What is the running time of your algorithm?

   (b) **[5 Points]** Let $R$ be a set of rectangles which are axis parallel. Show a polynomial time algorithm for finding the largest independent set in $G_R$ if all the rectangles of $R$ intersects the $y$-axis.

   (c) **[10 Points]** Let $R$ be a set of axis parallel rectangles. Using (b), show to compute in polynomial time an independent set of rectangles of size $\Omega(k^c)$, where $k$ is the size of the largest independent set in $G_R$ and $c$ is an absolute constant. (Hint: Consider all vertical lines through vertical edges of rectangles of $R$. Next, show that by picking one of them “cleverly” and using (b), one can perform a divide and conquer to find a large independent set. Define a recurrence on the size of the independent set, and prove a lower bound on the solution of the recurrence.)

2. **Greedy algorithm does not work for coloring. Really.**

   **[20 Points]**

   Let $G$ be a graph defined over $n$ vertices, and let the vertices be ordered: $v_1, \ldots, v_n$. Let $G_i$ be the induced subgraph of $G$ on $v_1, \ldots, v_i$. Formally, $G_i = (V_i, E_i)$, where $V_i = \{v_1, \ldots, v_i\}$ and

   $$E_i = \{uv \in E \mid u, v \in V_i \text{ and } uv \in E(G)\}.$$ 

   The greedy coloring algorithm, colors the vertices, one by one, according to their ordering. Let $k_i$ denote the number of colors the algorithm uses to color the first $i$ vertices.

   In the $i$-th iteration, the algorithm considers $v_i$ in the graph $G_i$. If all the neighbors of $v_i$ in $G_i$ are using all the $k_{i-1}$ colors used to color $G_{i-1}$, the algorithm introduces a new color (i.e.,
\[k_i = k_{i-1} + 1\] and assigns it to \(v_i\). Otherwise, it assigns \(v_i\) one of the colors \(1, \ldots, k_{i-1}\) (i.e., \(k_i = k_{i-1}\)).

Give an example of a graph \(G\) with \(n\) vertices, and an ordering of its vertices, such that even if \(G\) can be colored using \(O(1)\) (in fact, it is possible to do this with two) colors, the greedy algorithm would color it with \(\Omega(n)\) colors. (Hint: consider an ordering where the first two vertices are not connected.)

3. **Greedy coloring does not work even if you do it in the right order.**

   **[20 Points]**

   Given a graph \(G\), with \(n\) vertices, let us define an ordering on the vertices of \(G\) where the min degree vertex in the graph is last. Formally, we set \(v_n\) to be a vertex of minimum degree in \(G\) (breaking ties arbitrarily), define the ordering recursively, over the graph \(G \setminus v_n\), which is the graph resulting from removing \(v_n\) from \(G\). Let \(v_1, \ldots, v_n\) be the resulting ordering, which is known as min last ordering.

   (a) **[10 Points]** Prove that the greedy coloring algorithm, if applied to a planar graph \(G\), which uses the min last ordering, outputs a coloring that uses 6 colors.\(^1\)

   (b) **[10 Points]** Give an example of a graph \(G_n\) with \(O(n)\) vertices which is 3-colorable, but nevertheless, when colored by the greedy algorithm using min last ordering, the number of colors output is \(n\).

4. **[20 Points] Graph coloring revisited**

   (a) **[5 Points]** Prove that a graph \(G\) with a chromatic number \(k\) (i.e., \(k\) is the minimal number of colors needed to color \(G\)), must have \(\Omega(k^2)\) edges.

   (b) **[5 Points]** Prove that a graph \(G\) with \(m\) edges can be colored using \(4\sqrt{m}\) colors.

   (c) **[10 Points]** Describe a polynomial time algorithm that given a graph \(G\), which is 3-colorable, it computes a coloring of \(G\) using, say, at most \(n/(5 \log n)\) colors.

5. **[20 Points] An Easy Problem 5**

   Let \(G = (V, E)\) be an undirected graph. For any \(k \geq 1\), define \(G^{(k)}\) to be the undirected graph \((V^{(k)}, E^{(k)})\), where \(V^{(k)}\) is the set of all ordered \(k\)-tuples of vertices from \(V\) and \(E^{(k)}\) is defined so that \((v_1, v_2, \ldots, v_k)\) is adjacent to \((w_1, w_2, \ldots, w_k)\) if and only if for each \(i\) (\(1 \leq i \leq k\)) either vertex \(v_i\) is adjacent to \(w_i\) in \(G\), or else \(v_i = w_i\).

   (a) **[10 Points]** Prove that the size of the maximum clique in \(G^{(k)}\) is equal to the \(k\)-th power of the size of the maximum clique in \(G\).

   (b) **[10 Points]** Argue that if there is an approximation algorithm that has a constant approximation ratio for finding a maximum-size clique, then there is a fully polynomial time approximation scheme for the problem.

---

\(^1\)There is a quadratic time algorithm for coloring planar graphs using 4 colors (i.e., follows from a constructive proof of the four color theorem). Coloring with 5 colors requires slightly more cleverness.