Last Time: 3-COL is NP-complete

Today: We want to show 3-COL $\leq_p$ PLANAR-3-COL

To show 3-COL $\leq_p$ PLANAR-3-COL, we need to take any three-colorable graph $G$ and turn it into a planar graph $\hat{G}$ that is also three-colorable.

Our first step is to take any three colorable graph, and place all the vertices of the graph on the perimeter of a circle. If any of the edges intersect each other, then the graph needs to be made planar.

We can remove these intersections using the following gadget:

This gadget makes any intersection and more importantly preserves the following property: Every valid three-coloring has opposite corners of the same color, and any such coloring of the corners can be extended to a valid three-coloring of the gadget. If we have four vertices in a graph, $u$, $v$, $x$, and $y$, and $x \rightarrow y$ intersects $u \rightarrow v$, we can resolve that intersection by placing a gadget at the intersection. In our gadget, we would have $x$ be the far left vertex of the gadget, and $u$ be the top vertex. The right vertex of the gadget would point at node $y$ and the bottom
vertex of the gadget would point at node v. By arranging our gadget in that way we can maintain
the three-coloring of the graph while also making an intersection planar. This gadget placement is
shown in the image below.

So now that we know how to manage a single intersection, what happens when we have
multiple intersections? If we have multiple intersections, we can chain crossings together. This
preserves the three color and planar properties. Below is an example of gadget chaining.

An important thing to note is that given the graph with vertices on the perimeter of a circle,
we must apply the gadget to the intersection closest to the edge of the circle first, and
subsequently work inwards.

**Lemma:** 3-COL \( \leq_p \) 4-COL \( \leq_p \) 5-COL ... etc.
- Simply connect each node in an N-colorable graph to another node. This node must be of
an (N+1)th color.

**Yes or No Question:** 4-COL \( \leq_p \) 3-COL?
- Yes: 4-COL \( \leq_p \) SAT \( \leq_p \) 3-SAT \( \leq_p \) 3-COL
Space Complexity

For a deterministic TM $M$, its space complexity is the function $f : \mathbb{N} \to \mathbb{N}$ such that $f(n) =$ max number of tape cells $M$ uses on any input of length $n$.

For a non-deterministic TM $N$, $f(n) =$ max number of tape cells $N$ uses on any computation path on any input of length $n$.

Definitions:

SPACE($f(n)$) = \{all languages that can be decided by a deterministic TM of space complexity $f(n)$\}

NPSPACE = $\bigcup$ NSPACE($n^k$)

Savitch's Theorem: For any $f(n) \geq n$, NSPACE($f(n)$) $\subseteq$ SPACE($f(n)^2$)

To prove this theorem, we construct a TM CANYIELD($C_1, C_2, t$), where $C_1$ and $C_2$ are states in a non-deterministic turing machine, and $t$ is a number of time steps. CANYIELD tests whether $C_2$ can be reached from $C_1$ in at most $t$ steps. For any given computation path in a non-deterministic TM, a space complexity of $O(f(n))$ implies a time complexity $\leq C^{f(n)}$ where C is a constant.

We run CANYIELD($C_{\text{start}}, C_{\text{accept}}, C^{f(n)}$)

Example: CANYIELD($C_1, C_2, t$) =

1. If $t = 0$, ACCEPT only if $C_1 = C_2$
2. If $t = 1$, ACCEPT if $C_1 \rightarrow C_2$ in 1 step
3. Else for all configurations $C_m \leftarrow O(f(n))$ space
   
   • call CANYIELD($C_1, C_m, t/2$)
   • call CANYIELD($C_m, C_2, t/2$)
   • if both accept, ACCEPT

4. REJECT

We will have $O(f(n))$ stack entries, with each entry at $\log(C^{f(n)}) \rightarrow O(f(n))$ for each entry. This equates to $O(f(n)^2)$ overall. Thus there is a polynomial relationship between NSPACE($f(n)$) and SPACE($f(n)$).