Math 3322: Graph Theory[1] Mikhail Lavrov

Lecture 24: Graph coloring and the greedy algorithm

November 7, 2024 *Kennesaw State University*

1 The chromatic number

Today we will talk about coloring graphs.

Informally, this means painting every vertex of a graph some color. We can formally define a **coloring** of G as a function $f\colon V(G) \to S$, where S is the set of colors: we call $f(v)$ the color of vertex v. Usually, it will not matter what S is: we can be poetic and pick a set like $S = \{\text{red}, \text{blue}, \text{purple}, \text{violet}\}$, or practical and pick a set like $S = \{1, 2, 3, 4\}$.

It *will* matter how large S is: how many colors we use. If $|S| = k$ (in which case, it is convenient to set $S = \{1, 2, 3, \ldots, k\}$), we say that f is a k-coloring of G. We like to use as few colors as possible...

...but right now, we haven't said anything that requires us to use more than one color. We say that a **proper coloring** of G is a coloring in which adjacent vertices receive different colors. This goes back to the origins of graph coloring: coloring maps, in which adjacent regions should have different colors to distinguish them. However, graph coloring shows up in many applications: if edges represent "conflicts" between vertices (which is a very flexible interpretation), and we want to partition the vertices into non-conflicting groups, then we're looking for a proper coloring.

We say that G is **k-colorable** if it has a proper k-coloring. The **chromatic number** of G, denoted $\chi(G)$, is the minimum value of k for G is k-colorable. Once again, we're faced with an optimization problem, and so we can make some familiar observations:

- If we find any proper k-coloring of G, this is a proof by demonstration that G is k-colorable: that $\chi(G) \leq k$.
- Proving that $\chi(G) \geq k$ seems very hard. The brute-force approach is to show that none of the many many $(k-1)$-colorings of G are proper.

I am talking about graph coloring as though it is a new problem, but we have already seen one aspect of it near the beginning of the semester. A graph G is 2-colorable if and only if it is bipartite, and a bipartition is very nearly the same thing as a proper 2-coloring: just color vertices by which side of the bipartition they're on.

2 Greedy coloring

We are often interested in proving that all graphs of a certain type are k-colorable for some k. Here, it is not enough to draw a picture, because there could be infinitely many graphs of that type to

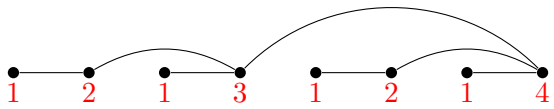
[1] This document comes from an archive of the Math 3322 course webpage: http://misha.fish/archive/3322-fall-2024

color. Usually, the way we do this is to find an algorithm that tells us how to color the graphs we care about, and then prove that the algorithm never uses too many colors.

The simplest graph coloring algorithm is the greedy coloring algorithm. This does the following:

1. Number the vertices $v_1, v_2, \ldots, v_n$ in an arbitrary order. (We will usually illustrate this by drawing the graph so that the vertices are $v_1, v_2, \ldots, v_n$ from left to right.)
2. Color the vertices in that order, giving v_i the first color[2] that does not appear on any of its neighbors among $v_1, v_2, \ldots, v_{i-1}$.

This is "greedy" because it does the first thing it can think of without thinking about future consequences. Sometimes this backfires. The graph below is a tree, so it is bipartite—and 2-colorable. But if we go from left to right, we get a 4-coloring (labels in red):



The truth about the greedy algorithm is that the result heavily depends on the order of vertices $v_1, v_2, \ldots, v_n$ that we picked at the start. In the example above, I had to carefully order the vertices to get the tree to be 4-colored. On the other hand, if you want the tree to be 2-colored, you also have to order the vertices at least a little bit carefully.

The following bound on chromatic number is what we're guaranteed to get even if we take no care in ordering the vertices:

Theorem 2.1. *For any graph G, $\chi(G) \leq \Delta(G) + 1$.*

Proof. We will prove that the greedy algorithm never uses more than $\Delta(G)+1$ colors. (Here, $\Delta(G)$ is the maximum degree of G.) In other words, if our colors come from the set $S = \{1, 2, \ldots, \Delta(G)+1\}$, the greedy algorithm will never get stuck.

When we are coloring vertex v_i, it has at most $\Delta(G)$ neighbors among $v_1, v_2, \ldots, v_{i-1}$. (It could have fewer for two reasons: it's possible that v_i has fewer than $\Delta(G)$ neighbor, and it's possible that not all of them come before v_i in the ordering we picked.)

Among those neighbors, there can be at most $\Delta(G)$ colors. Therefore there is at least one color from the set $S = \{1, 2, \ldots, \Delta(G) + 1\}$ that does *not* show up on any of them. We can continue by giving v_i the first such color.

Since the greedy algorithm never fails to pick a color from S, at the end we will have a proper $(\Delta(G) + 1)$-coloring of G. □

In the remainder of the lecture, we will apply the greedy coloring algorithm to two special cases of the graph coloring problem. By putting some thought into the order we color the vertices, we will be able to do better than the general bound in Theorem 2.1.

[2]The "first color" assuming the colors are numbered $1, 2, 3, \ldots$, so that they have a definite order.

3 Greedily coloring planar graphs

Coloring a map is the origin of graph coloring, and when we color a map, we are usually coloring a planar graph. (Put a vertex in each region on the map. Draw an edge between vertices if their regions share a border. If the regions are connected—which is not always true in real-world maps—then these edges can be drawn without crossing.)

By carefully ordering the vertices of a planar graph, we can get the greedy algorithm to perform better. Here, the heuristic we will use is that **it is good to leave low-degree vertices until the end**. If a vertex v does not have many neighbors, then even if v's neighbors all get different colors, there will be a color left for v.

What kind of low-degree vertices will planar graphs have? We can show the following:

Lemma 3.1. *Every planar graph G has minimum degree $\delta(G) \leq 5$.*

Proof. This is true for every planar graph with at most 6 vertices because at that point, you can't have any degrees bigger than 5.

For planar graphs with $n \geq 3$ vertices and m edges, we have the inequality $m \leq 3n - 6$. However, if every vertex had degree 6 or more, we would have $m \geq \frac{1}{2}(6n) = 3n$ by the handshake lemma, and we cannot have $3n \leq 3n - 6$. Therefore not all vertices have degree 6 or more: there must be a vertex with degree 5 or less. □

Just having a small minimum degree would not help us. What *does* help us is that when you remove a vertex of minimum degree, we are left with a smaller planar graph, for which the following holds. This lets us pick a good vertex ordering:

Lemma 3.2. *Every planar graph G has a vertex ordering $v_1, v_2, \ldots, v_n$ in which each vertex is adjacent to at most 5 of the vertices that come before it.*

Proof. We will prove this by induction on n. When n is small (say, $n \leq 6$), any vertex ordering will do.

Assume that the lemma is true for all $(n-1)$-vertex planar graphs, and let G be an n-vertex planar graph. By Lemma 3.1, G has a vertex v with $\deg(v) \leq 5$.

Apply the induction hypothesis to find a vertex ordering $v_1, v_2, \ldots, v_{n-1}$ of $G - v$. We can extend it to a vertex ordering of G by setting $v_n = v$. This vertex ordering does what we want for the first $n - 1$ vertices by the induction hypothesis, and for the last vertex because $\deg(v) \leq 5$.

By induction, we can find such a vertex ordering for all n. □

Using this lemma, we can get a bound on the chromatic number of arbitrary planar graphs!

Theorem 3.3. *Every planar graph G has $\chi(G) \leq 6$.*

Proof. Using the vertex ordering given by Lemma 3.2 and 6 colors, apply the greedy coloring algorithm to color G. We will show that at every step in the greedy algorithm, one of the 6 colors is available to color the next vertex.

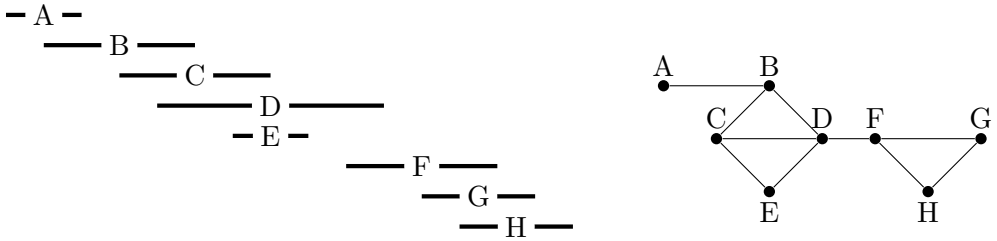
Since each vertex v_i is adjacent to at most 5 of the vertices that come before it, there will be at least one of the colors from $S = \{1, 2, 3, 4, 5, 6\}$ that does not appear on any of v_i's already-colored neighbors. So we can always continue by assigning v_i some color from S. When we're done, we will have given G a proper 6-coloring. □

Much more is true! In fact, there is a famous theorem called the Four Color Theorem that $\chi(G) \leq 4$ for every planar graph G. However, this is much harder to prove, to the point that all known proofs require a computer to check many cases.

4 Greedily coloring interval graphs

Recall from the previous lecture that an **interval graph** is a graph defined by a set of intervals $\{[a_1, b_1], [a_2, b_2], \ldots, [a_n, b_n]\}$. For each interval $[a_i, b_i]$, there is a vertex v_i; two vertices are adjacent whenever the intervals overlap.

Here is an example of a set of intervals and their interval graph:



If planar graphs are the historically important application of graph coloring, then interval graphs are the modern application. There are two important applications that boil down to coloring an interval graph:

- Event scheduling. Here, we imagine that each interval $[a_i, b_i]$ represents an event starting at time a_i and ending at time b_i. We want to put the events in rooms, but we have a limited supply of rooms available, so we want to use as few as possible. However, if two events are happening at the same time, they must be in different rooms.
- Register allocation: an application to computer science. Here, each interval $[a_i, b_i]$ represents a variable in a computer program, which is initialized at step a_i in the code, and last used at step b_i. There are special parts of computer memory called *registers* which are particularly easy to access, but their number is limited. We want to see if the variables can be stored in registers, so that variables which are in use simultaneously occupy different registers.

With interval graphs, the greedy algorithm turns out to perform well with the natural ordering of the vertices. We sort the intervals $[a_i, b_i]$ by their starting point, so that $a_1 < a_2 < \cdots < a_n$. Then, we color the vertices $v_1, v_2, \ldots, v_n$ in this order.

How well does this do? Here's how well:

Theorem 4.1. *If G is an interval graph, then $\chi(G) = \omega(G)$ ($\omega(G)$ is the clique number of G).*

Proof. Suppose we are coloring vertex v_i during the greedy algorithm. Which vertices among $v_1, \ldots, v_{i-1}$ are adjacent to this vertex: which of the intervals that come before $[a_i, b_i]$ overlap with it?

Well, they all start before a_i, so to overlap with $[a_i, b_i]$, they must end after a_i. This means that they all contain the point a_i itself. But if we're looking at a set of intervals that all contain a_i, then they form a clique, because they all overlap! So, together with v_i, we are looking at $\omega(G)$ or fewer vertices: v_i has at most $\omega(G) - 1$ neighbors that come before it.

Therefore the greedy algorithm will never use more than $\omega(G)$ colors.

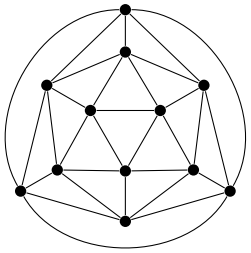
(Conversely, it will have to use at least $\omega(G)$ colors, because the vertices of a clique will all need different colors. More on this in the next lecture.) □

5 Practice problems

1. Draw a tree that, when the vertices are unfortunately ordered, ends up being given 5 different colors by the greedy algorithm.

   (Think about how the 4-color example works; don't try to reinvent the wheel.)

2. Here is a plane embedding of the icosahedral graph.



   (a) Find a proper 4-coloring of this graph.

   (b) Prove that there is no proper 3-coloring of this graph: the chromatic number is exactly 4.

3. The **lowest degree last** vertex ordering of an n-vertex graph G is constructed as follows, recursively. We choose the last vertex, v_n, to be any vertex whose degree is the lowest degree in G. To find the ordering $v_1, v_2, \ldots, v_{n-1}$ of the other vertices, we find the lowest degree last ordering of the $(n-1)$-vertex graph $G - v_n$.

   This is exactly the vertex ordering which we described in Lemma 3.2 for planar graphs.

   Prove that if G is a tree, then the greedy coloring algorithm, using the lowest degree last ordering, will never use more than 2 colors.

4. When we take the union of graphs with the same vertex set, we just keep that set of vertices and include an edge if it is contained in any of the graphs.

   (a) Prove that if G is the union of k forests, then the minimum degree of G is $2k-1$ or less. (Think about the proof of Lemma 3.1.)

   (b) Prove that if G is the union of k forests, then it is $2k$-colorable.

   (c) Prove that the complete graph K_{2k} is the union of k forests, so we can't prove any better result than what we got in part (b).

5. (a) Suppose that the greedy algorithm uses k colors on a graph G. Prove that G must have at least $1 + 2 + \cdots + (k-1) = \binom{k}{2}$ edges.

   (Hint: whenever a color other than color 1 is used, there are some edges that have to exist to cause this.)

   (b) Prove that if G has m edges, then $\chi(G) \leq 1 + \sqrt{2m}$.