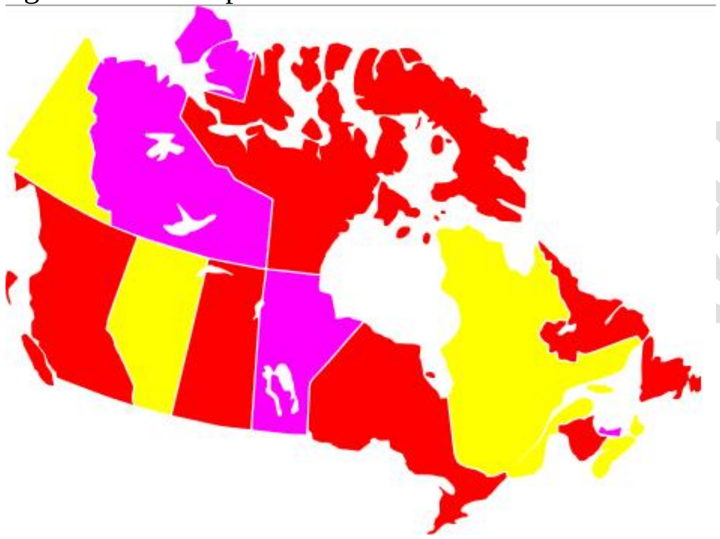
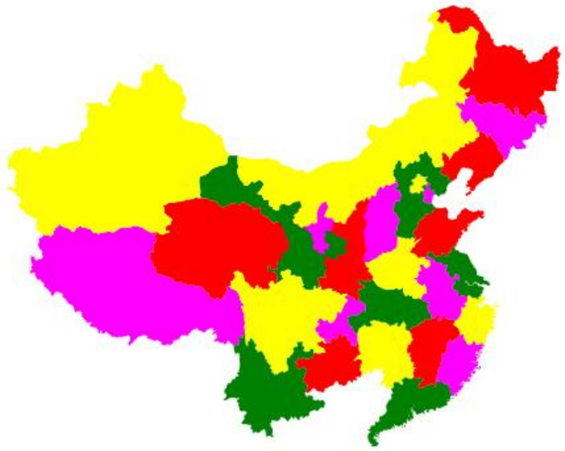
Graph Coloring

- We have seen several problems where it doesn't seem like graph theory should be useful.
  - But graphs are often very good at capturing the important stuff about a problem, so we can see it clearly.
  - This is going to be another one of those.
- When drawing a map, we want to be able to distinguish different regions.
  - One of the ways that's done is by drawing them in different colors.
  - …with the rule that regions that touch each other must be different colors: otherwise we couldn't see the border.
- e.g. I colored the provinces of Canada with three colors:



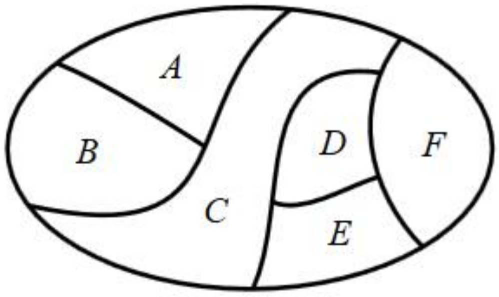
  - But it took four colors for China:



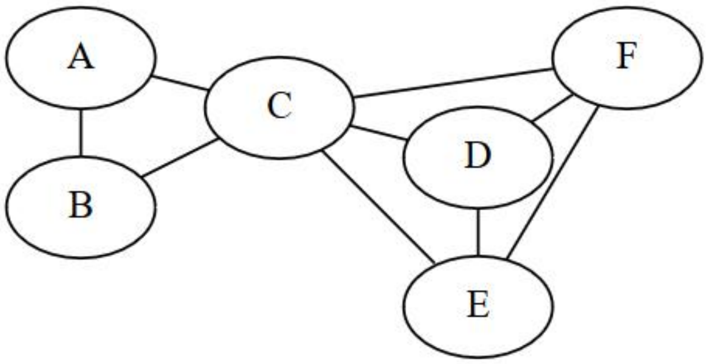
  - Could I have colored China with three colors?
  - Are there countries that take five colors? Six?
  - Could there be such countries if they had weird-enough borders?
- Again, we can represent this problem as a graph to get the useful information, while throwing away details we don't care about.

- Each region will be a vertex.
- Connect vertices with edges if they share a border (that's larger than a point: they can touch at a corner and we don't care).

- For example…
  - We start with this map with six regions:



  - We reduce it to this graph:



- Now the question becomes: how many colors do we need to color this (or any other) graph so that no two adjacent vertices have the same color?
  - A solution (colored graph) is called a *coloring*.
  - The minimum number of colors needed for a coloring of a graph is its *chromatic number*.
- ***Theorem:* Every bipartite graph has chromatic number two.**

  Proof: By definition, bipartite graphs can have their vertices partitioned into two sets with no edges within the sets. Color the vertices from each partition the same color, and we have a coloring.

- In general, the chromatic number could be large.
  - For example, a K_n requires n colors since every vertex is adjacent to every other.
- But, if we get our graph from a map, it will be planar.
  - As long as we don't allow enclaves and exclaves, and we won't.

    - So the question becomes: what is the maximum chromatic number for a planar graph?
- Easy bound: it's at least four.
    - The graph K_4 is planar and definitely requires four colors.
- It's easy to show that the chromatic number of a planar graph is no more than six.
    - We know there is a vertex of degree five or less from an earlier theorem.
    - Remove that vertex, and by induction, the rest of the graph can be colored.
    - Color it a different color than the adjacent five.
- Getting ≤ 5 isn't much harder.
    - There's a little trick to show that the five adjacent vertices didn't have to be different colors.
    - … but don't worry about the details.
- That leaves us with a question: are there planar graphs that require five colors?
    - It turns out: no.
- ***Theorem:* [The Four-Color Theorem] The chromatic number of a planar graph is at most four.**

  Proof: It took more than 100 years between conjecture and proof for this theorem. The proof involved reducing the planar graphs to about 2000 examples where if the theorem was false, it was shown one of these would be a counter-example. These each had 4-colourings found by computer.

  Obviously, we won't go into the details.

- So, it turns out we can color any (planar) map with four colors.
    - The four-color theorem is one that seems true: a little experimenting would convince most people that the chromatic number can be at most four.
    - But the proof turned out to be very difficult.
- But remember:
    - The chromatic number of non-planar graphs can be more than four: for example n for K_n.
    - The chromatic number of non-planar graphs can also be small: 2 for $K_{10000, 100000}$.

Coloring Applications

- Actually coloring maps is a good motivation for the chromatic number problem, but maybe not all that practical.
    - Coloring the maps we actually have on the planet doesn't need a big theorem.
    - It's hard to see why we would care about non-planar graphs.
- There are other places where a coloring of a graph tells us something interesting.
    - These are really just examples of where the problem comes up in an unexpected place.

- Scheduling exams
  - Let the vertices of the graph be the courses, and connect vertices with an edge if there are any students in both courses.
  - We'll end up with a graph like this (but bigger):
  - If we can color this with n colors, then we need n exam slots.
  - … And courses in each color can be scheduled at the same time.
- Register allocation
  - Registers are pieces of memory in a processor that are much faster than standard RAM.
  - When your code is compiled by an optimizing compiler, it will attempt to use the registers when appropriate to speed up your code.
  - E.g. the index on a `for` loop is probably used a lot: maybe use a register for it?
  - But there is a limited number registers.
  - Create a graph: vertices are variables in the code; edges join them if the variables are used in the same segment of code.
  - If we can color the graph with n colors, then we can use n registers for all of the variables.
  - Variables with the same color can share a register, swapping them out when necessary.
- Sudoku
  - Create one vertex for each of the 81 squares.
  - Join vertices if they cannot have the same value assigned: if they are in the same row/column/square.
  - Create some extra edges to enforce the pre-filled values.
  - Find a nine-coloring.
- If we might actually want to find colorings to solve real problems, how quickly can we find them?
  - Again, we don't know of any fast algorithm.
  - Apparently, we can find a coloring of n vertices in $O(2.445_n)$ or $O(2_n n)$.
  - There are some reasonably-fast approximation algorithms: we can color a graph quickly if we are willing to accept a few more colors than necessary.