



University of Waterloo
Faculty of Mathematics

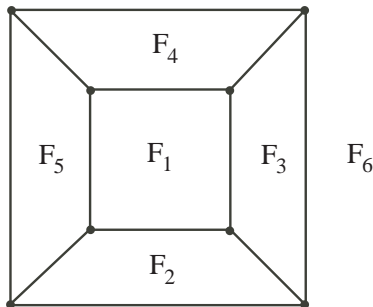
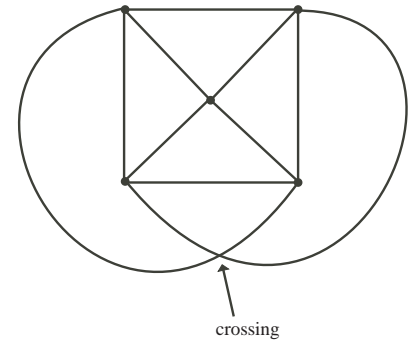


Centre for Education in
Mathematics and Computing

Senior Math Circles November 25, 2009 Graph Theory II

Planar Graphs

A graph that can be drawn in \mathbb{R}^2 without any edges crossing is **planar**. An example of a graph which is not planar is the complete graph on 5 vertices (the graph in which every vertex is adjacent to every other vertex; all $\binom{5}{2} = 10$ edges are present).

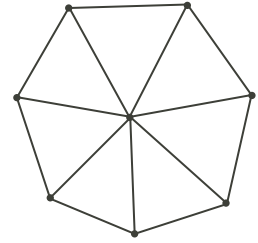


When we draw a planar graph, it divides \mathbb{R}^2 into different regions which are called *faces*. When talking about the faces of a graph, the infinite outer face is included. For example, the diagram here of a *cube graph* has 6 faces, denoted F_1, \dots, F_6 with F_6 the infinite outer face.

4-colour theorem: The faces of any planar graph can be coloured using at most 4 colours in total such that any two adjacent faces are not the same colour.

- The theorem was proposed in 1852, and was popularized by Cayley.
- It was proven in 1879 & 1880.
- Flaws were found in those proofs in 1890 & 1891.
- It remained open again until it was finally proven in 1976 by Appel & Haken.
 - computers were involved in checking a finite number of cases of proof.
 - proof overview: give a list of *reducible configurations*, show that induction can be used whenever a reducible configuration is found, and show that every graph contains a reducible configuration.
 - people are still coming up with simpler and more transparent proofs today.

Note that we only disallow faces which *share an edge* to be coloured the same — it is okay if like-coloured faces share a vertex. Otherwise we can come up with graphs (like the one pictured) which take an arbitrarily large number of colours.



In the problems at the end of this handout, we show that a weaker “6-colour theorem” is reasonably easy to prove.

In a connected planar graph, **Euler’s formula** states that

$$\text{the number of vertices} - \text{the number of edges} + \text{the number of faces} = 2$$

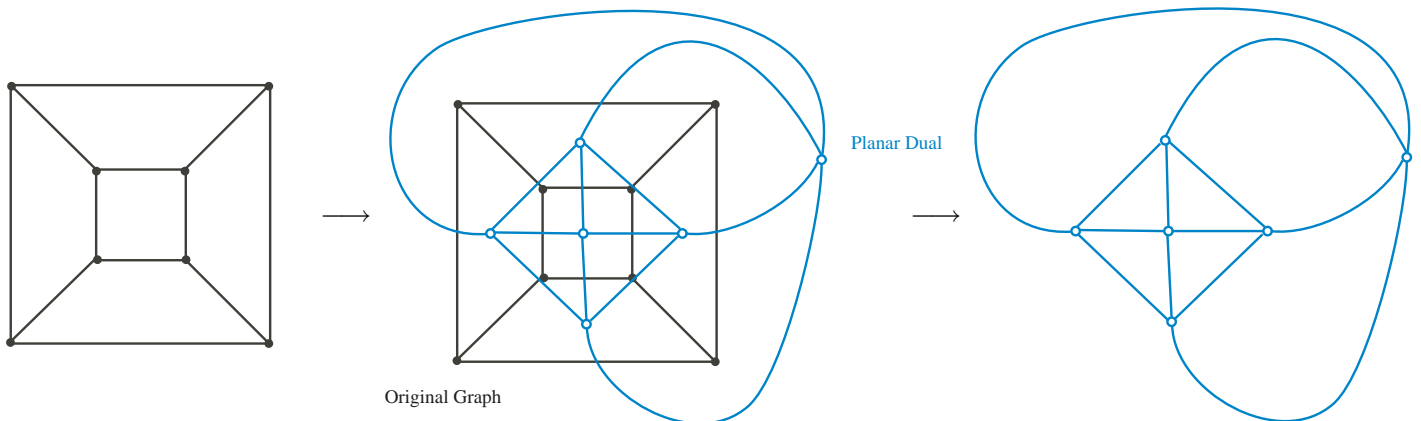
where the number of faces includes the infinite outer face. We will often abbreviate this as

$$|V| + |F| - |E| = 2$$

where $|V|$ is the number of vertices, $|F|$ is the number of faces, and $|E|$ is the number of edges.

Dual Graphs.

Every planar graph has a *dual graph*. For any connected planar graph G , the dual of its dual is again G . To obtain the dual of a graph (e.g., the cube graph shown with black vertices below), we put a white vertex in every face of G . Then, whenever two faces of G are next to each other, we connect the corresponding two white vertices with a *dual edge*. Note that we can delete the original graph and just deal with the dual graph by itself, as shown in the figure below.



We always have the following:

$$\begin{aligned} \text{number of original faces} &= \text{number of dual vertices} \\ \text{number of original edges} &= \text{number of dual edges} \\ \text{number of original vertices} &= \text{number of dual faces} \end{aligned}$$

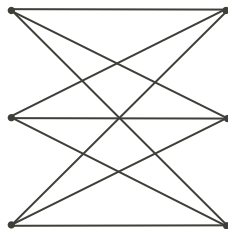
For example, to prove the first fact, just recall that we put one dual vertex in each face of the original graph. Another fact is that, if v is a vertex of the original graph of degree d , then the dual face corresponding to v has exactly d sides (see the diagram for an idea of the proof).

In the example above, the original graph had degree 3 at every vertex and every face was a square; this implies that the dual graph has triangular faces and degree 4 at every vertex, i.e. the dual graph is an octahedron.

Exercise. Compute the dual graphs of the tetrahedron, icosahedron, and dodecahedron. How are these graphs related to each other?

Duality has a nice effect on the 4-colour theorem. Instead of colouring faces of a planar graph, we can think about colouring vertices of the dual graph in such a way that adjacent vertices get different colours. (Why is this nice? For example, we can't colour the faces of non-planar graphs since they don't exist, but we can colour their vertices.) We call an assignment of colours to vertices a *proper colouring* if every adjacent pair gets different colours.

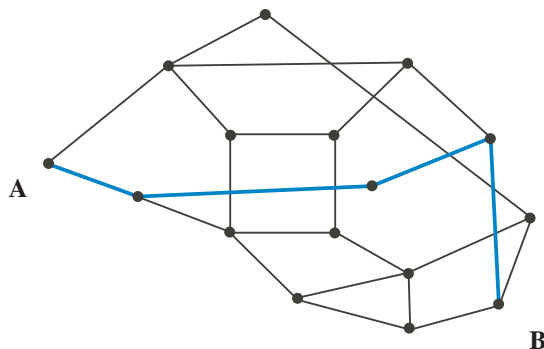
Exercise. What is the least total number of colours required to properly colour the following graph?



The graph in the exercise above is usually denoted $K_{3,3}$ and called the *complete bipartite graph*, since it has two (“bi-”) groups (“parts”) of three vertices ($(3,3)$) such that each vertex is connected to all vertices in the other part (“complete.”)

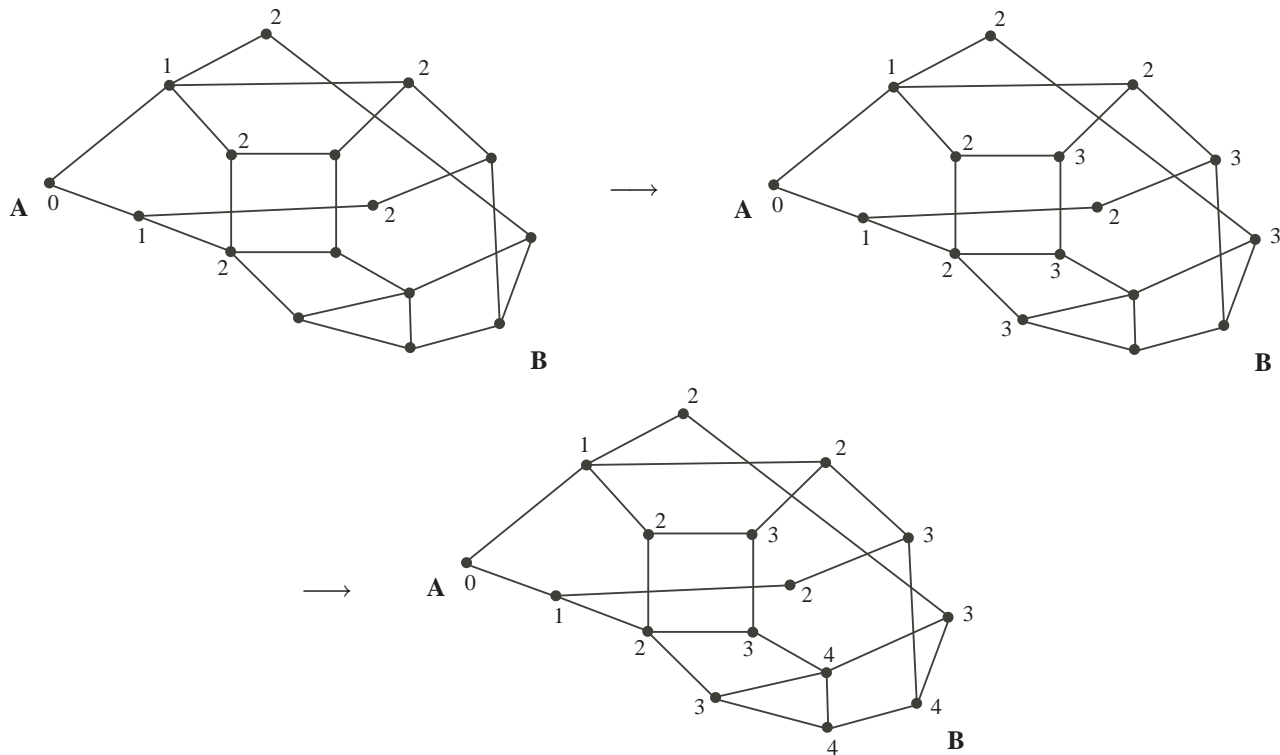
Breadth-First Search. Here is a natural question, related to applications like Google Maps. I give you a graph containing vertices A and B . What is the least number of steps (along edges) that will get me from vertex A to vertex B ?

In the example below, we see that there is a path P (bold, blue edges) of length 4. But how can we actually prove this is the shortest path?



We now give the most common all-purpose idea to find shortest paths in graphs, which is called *breadth-first search*. The idea is that we will label every vertex according to its *distance* from A , i.e. the label of v is the minimum number of steps to get from A to v . We need to compute all the small numbers first before computing the large numbers, and the term *breadth-first* refers to the fact that we are sort of searching outwards from A in layers.

First, we label A with the number 0. Then we label all neighbours of A with 1. Next, for all vertices v with label 1, for every unlabeled vertex w adjacent to v , we label w with 2. Then the unlabeled neighbours of 2-labeled vertices become 3-labeled, et cetera. You can prove by induction on the distance from A that this assigns the correct distance label to every vertex. The diagram below gives an example.



Implicit Search & Meet in the Middle. Graphs are hidden everywhere. In some cases, a “graph search” problem is the most natural way to think about a problem which says nothing about graphs!

An example is a *word ladder puzzle*. There are lots of variations on this idea, so we describe it in full detail. You are given a “start word” and an “end word,” both 4 letters long. (E.g. COLD for the start and HEAT for the end.) The goal is to quickly transform the start word into the end word, subject to the following rules: (i) you can only change one letter at a time, and (ii) all intermediate words must be actual English words. (So you can’t change COLD into ZOLD.)

In some cases, word ladders are easy. The given example has a solution with 4 transformations: COLD-HOLD-HELD-HEAD-HEAT. This is clearly optimal since we need to change each letter once. But for more complex situations, this reasoning won’t always figure out the shortest solution.

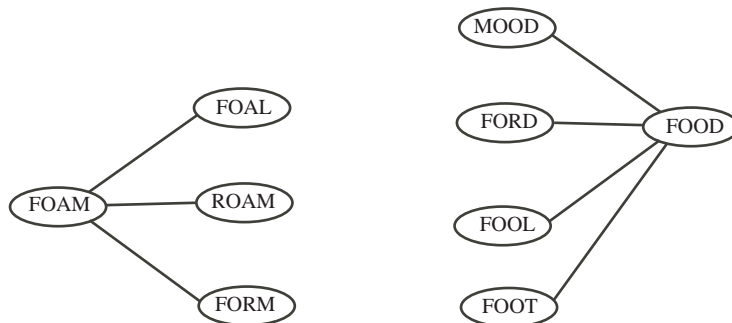
Here is a way of thinking about the problem in terms of graphs: define a vertex for every 4-letter English word, and connect two words by an edge whenever they differ only in one letter. Then we are just looking for a shortest path!

Exercise. Use breadth-first search to find a shortest path from POST to POTS.

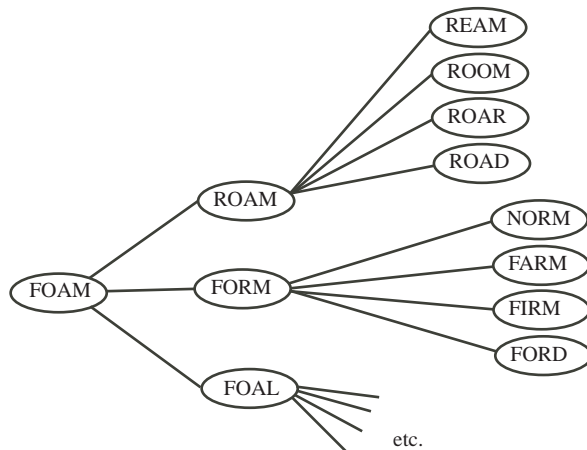
Another conceivable situation is that you have a Rubik's cube that you want to solve using the least number of possible operations (turns of a side). You could model this by a graph with a vertex for each possible state/picture of the cube, and vertices linked by all possible face turns. In principle, if you know how to do some programming, and a little graph theory, this will allow you to find the shortest path between the original (mixed) state of the cube and the final (solved) state where all faces have one colour.

We say “in principle” above because there are a lot of states to check, which takes up a lot of memory and uses a lot of processor time, even for today's modern computers. This complexity is important: if you ask Google Maps how to get to the supermarket because you are hungry, if it takes 2 weeks to compute the answer, then you may have starved by then. Therefore, it is useful to have algorithms which have better performance than breadth-first search. A simple improvement to breadth first-search is the **meet in the middle** algorithm which interleaves a forwards breadth-first search from the start with a backwards breadth-first search from the end. We start whenever both searches have a common member.

For example, let's find a shortest path from FOAM to FOOD. We start by searching one layer around FOAM: it is adjacent to ROAM, FORM, and FOAL. Then we go to FOOD and search one layer around it: it is adjacent to MOOD, FORD, FOOL, FOOT. *Remark:* as we go along, we are drawing tree structures which grow outwards from FOAM and FOOD, which are usually called *breadth-first search trees*.



We just grew the FOOD tree so we go back to add a layer to the FOAM tree, which is pictured below. At this point the word FORD is in both trees so we are done: a shortest path is FOAM-FORM-FORD-FOOD.



In practice, this meet in the middle search can result in a huge savings. Imagine some search problem where the distance was 16. Suppose for the sake of estimating that the tree increases in size by a factor of 3 nodes at each level. Then a BFS would result in a tree of size about $3^{16} \approx 43$ million, but meet-in the middle grows two trees each of size about $3^8 \approx 6500$, which is a huge savings in work.

Proofs of Euler's Formula.

To conclude, since Euler's formula is one of the most well-used theorems about planar graphs, we should do it justice by actually figuring out how to prove it. We sketch two proof ideas here.

Proof 1. Use induction. It is not so easy to figure out an appropriate induction variable, but a natural base case is *trees*: any tree has just one face, so $|V| + |F| - |E| = |V| + 1 - (|V| - 1) = 2$ as needed. Then complete the proof by arguing that whenever we add an edge to the graph, it increases the number of faces by exactly 1.

Proof 2. Use Problem 1 in the exercises below for a direct proof.

Problems

- Consider a connected planar graph G . Let T be a spanning tree of G . Let X be the set of all edges *not* in T .
 - Using the fact that T connects all vertices, show that in the dual graph G^* , X is acyclic.
 - Using the fact that T is acyclic, show that in the dual graph G^* , X connects all vertices. (Remember vertices of G^* correspond to faces of G .)
 - Deduce, using (a) and (b), that X is the edge set of a spanning tree of G^* .
- A colouring of the vertices of G is *proper* if, for all edges uv of G , the colour of u and the colour of v are different. The 4-colour theorem states that “every planar graph has a proper colouring using at most four colours in total.” We will prove the weaker but easier statement that “every planar graph has a proper colouring using at most *six* colours in total.”

Part 1. Assume the following statement is true:

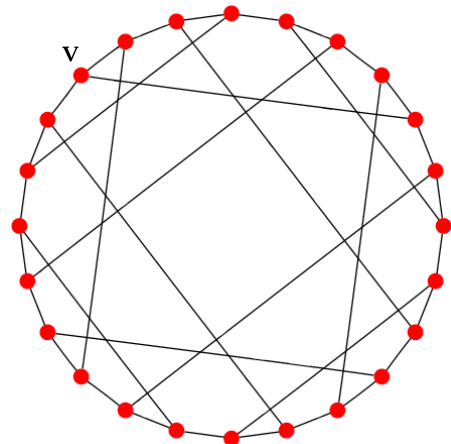
$$\underline{\text{Every planar graph has at least one vertex } v \text{ such that } \text{degree}(v) \leq 5.} \quad (\star)$$

Assuming that (\star) is true, show by induction that every planar graph has a proper colouring using at most *six* colours in total. (Hint: deleting v from G leaves a planar graph.)

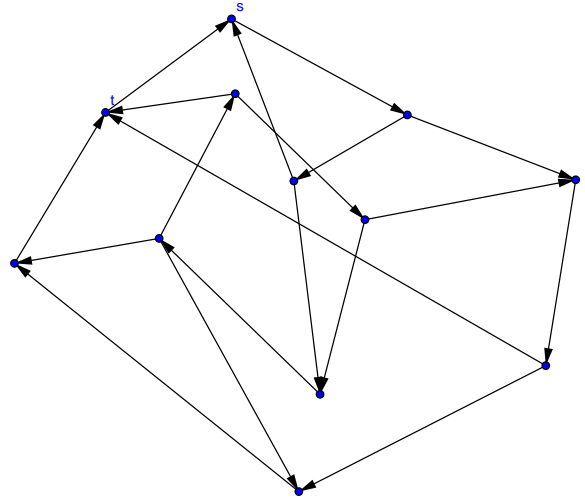
Part 2. We will prove (\star) .

- If G has no cycles, show that G has a vertex of degree at most 5. (Hint: $1 \leq 5$.)
 - If G has any cycles, then it is easy to argue that every face has at least three edges. Let $|E|$ denote the number of edges of G and $|F|$ denote the number of faces of G . Show that $|F| \leq \frac{2}{3}|E|$.
 - Suppose for the sake of contradiction that (\star) is false. Then there is a counterexample planar graph G in which every vertex has degree at least 6. Show that in this graph, $|V| \leq \frac{1}{3}|E|$, where $|V|$ denotes the number of vertices of G .
 - Complete the proof of (\star) by using Euler’s formula.
- Use breadth-first search to prove one direction of last week’s problem #6: if a graph has no cycles of odd length, construct a proper colouring using at most *two* colours.

- Find the furthest vertex (or vertices) from v in the graph on the left below. (It is called the *Nauru graph*.)



5. Modify the breadth-first search and meet-in-the-middle search algorithms to work on directed graphs. As an example, find the shortest distance from s to t in the directed graph on the right above.



6. The *4-letter word graph* has vertex set equal to the set of all 4-letter English words, and edges connecting two words when the letters in three positions are the same. (For example, MATH and MOTH are adjacent, POST and POTS are *not* adjacent, and an example of a path is COLD-HOLD-HELD-HEAD-HEAT.)
- Find a cycle of length 3.
 - Find a shortest path between JUNK and GOLD.
 - Prove the graph is not planar (hint: use the 4-colour theorem and consider all words of the form ?ATS).

Further resources:

- The “draw this figure without tracing any line more than once” puzzle from the first lecture is basically a question about whether the associated graph has a path which visits every edge once (but maybe some vertices more than once). It is called an Eulerian path. An Eulerian path exists if and only if the graph is connected, and at most two vertices have odd degree. An Eulerian cycle visits every edge exactly once *and* ends where it starts; an Eulerian cycle exists if and only if the graph is connected, and all vertices have even degree. See a proof at <http://www.stanford.edu/class/cme305/Lectures/Lecture1.pdf>