Hamilton circuits (Section 2.2)

Under what circumstances can we be sure a graph has a Hamilton circuit?

Theorem 1. *K_n has a Hamilton circuit for $n \geq 3$.*

Proof. Let $v_1, \ldots, v_n$ be any way of listing the vertices in order. Then $v_1 - v_2 - \cdots - v_n - v_1$ is a Hamilton circuit since all edges are present. □

In general, having lots of edges makes it easier to have a Hamilton circuit.

Theorem 2. *If $G = (V, E)$ has $n \geq 3$ vertices and every vertex has degree $\geq n/2$ then G has a Hamilton circuit.*

Proof. First, we show that the graph is connected. Suppose G is not connected, so that G has at least two components. Then we could partition $V = V_0 \cup V_1$ into two non-empty pieces so there are no edges between V_0 and V_1. (V_0 and V_1 might not be components themselves, because there might be more than two components; instead V_0 and V_1 are unions of components.) Since $n = |V| = |V_0| + |V_1|$, we must have either $|V_0| \leq n/2$ or $|V_1| \leq n/2$. Say V_0 has size $\leq n/2$ and pick any $v \in V_0$. Then $deg(v) \geq n/2$, but every neighbor of v is contained in V_0 and is not v, so $deg(v) < n/2$; this is a contradiction. So G is connected.

We prove there is a Hamilton circuit by induction. Let p_m be the statement "As long as $m + 1 \leq n$, there is a path visiting $m + 1$ distinct vertices with no repetitions". p_0 is trivial—just take a single vertex.

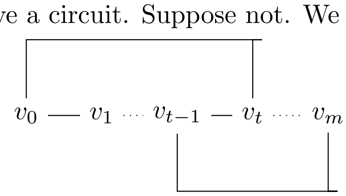
Suppose p_m is true, so we have a path

$$v_0 - v_1 - \cdots - v_m.$$

We want to show that we can extend this to a circuit with one more element. If v_0 is adjacent to any vertex not already in the path, we could just add it before v_0 and be done. Similarly, if v_m were adjacent to any vertex not already in the path, we could add it after v_m and be done.

So we have to consider the hard case. In this case, we have an important additional fact: all neighbors of v_0 and all neighbors of v_m are somewhere in the path.

We want to turn our path into a cycle. If v_0 is adjacent to v_m then we already have a circuit. Suppose not. We want to find the following arrangement:



because then we could break the link between v_{t-1} and v_t and have the circuit

$$v_t - \cdots - v_m - v_{t-1} - \cdots - v_1 - v_0 - v_t.$$

We know that v_0 has $n/2$ neighbors, all of them are in this path, and none are v_m. Let A be the vertices adjacent to v_0, so $|A| \geq n/2$. Let B be all the vertices which are adjacent to v_m, so $|B| \geq n/2$. Every vertex in B belongs on the path, so we can ask about the vertex immediately after it on the path. Let C be the set of vertices which are immediately after some vertex in B in the path. Then $|C| = |B| \geq n/2$. If $A \cap C = \emptyset$—if A and C are disjoint—then $|A \cup C| \geq n/2 + n/2 \geq n$, so $A \cup C$ would have to include all the vertices. But v_0 is in neither A nor C, so $A \cup C$ isn't all the vertices, so there is some vertex $v_t \in A \cap C$, and so $v_t \in A$ while $v_{t-1} \in B$.

Therefore (remember, we're still in the case where we can't just tack an element on at the beginning or end) we have turned our path into the circuit

$$v_t - \cdots - v_m - v_{t-1} - \cdots - v_1 - v_0 - v_t.$$

If $m + 1 = n$, we have included all the vertices, so we have a Hamilton circuit and we're done. If $m + 1 < n$, there must be some vertex not included in our circuit, and since G is connected, there must be some vertex w which isn't in our circuit but is adjacent to something in our circuit, say v_u. So we can rotate our circuit so v_u is the first vertex and then tack on w before it, say

$$w - v_u - v_{u+1} - \cdots - v_m - v_{t-1} - \cdots - v_1 - v_0 - v_t - \cdots - v_{u-1}.$$

This is a path with $m + 2$ elements, so we have shown p_{m+1}.

By induction, we know that for every m, p_{m+1} is true, so in particular, there is a path of length $m+1$. In particular, we have a path of length n, and, by the argument just given, we can turn this path into a circuit. □

We have given some examples of *necessary* conditions for Hamilton circuits—things that must be true if a graph has a Hamilton circuit—and *sufficient* conditions—things which, if true, guarantee that a graph has a Hamilton circuit. In the Euler case, we found conditions which were simultaneously necessary and sufficient; in the Hamilton circuit case, we don't know of any such conditions.

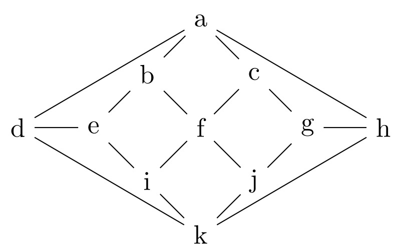
The conditions we've seen aren't the only possible ones. Here's another sufficient condition:

Theorem 3. *Suppose G is a planar graph and has a Hamilton circuit. Take any drawing of G on the plane. Then the Hamilton circuit creates an inside and an outside. For each i, let r_i be the regions inside the circuit with i edges on the boundary, and let r'_i be the regions outside the circuit with i edges on the boundary. Then*

$$\sum_i (i-2)(r_i - r'_i) = 0.$$

(Here I draw a small graph with a Hamilton circuit and did the calculation. This is easy to do: draw between 5 and 7 vertices in a circle with the edges between them and let that be the Hamilton circuit. Then draw some extra edges both inside and outside, without crossing. This always gives you a planar graph with a Hamilton circuit. Check that the equation holds. Don't forget the "outside" region—the region whose edges are the outside of the graph.)

This theorem gives a more complicated way of seeing that some graphs can't have a Hamilton circuit. Here's an example:



What are the regions in this drawing? There are nine regions, all of which have four edges forming their boundary. A Hamilton circuit would have to divide these nine regions so half were inside and half were outside, which is obviously impossible.

Graph Coloring (Section 2.3)

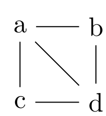
One example of a planar graph is a map, the sort we'd find in geography: imagine placing a vertex inside each country (or state, or provinice, or whatever) and drawing an edge between vertices which share a border. If we arrange so each edge crosses the border between those two countries, we can easily make sure the edges never cross, so the resulting graph is planar.

Most maps want to color all the countries in such a way that adjacent countries always get different colors. This corresponds to coloring the vertices of the graph.

Definition 4. A *coloring* of a graph is an assignment of a color to each vertex of a graph so that adjacent vertices have different colors.

The *chromatic number* of a graph is the smallest number of colors which must be used to color that graph.

The chromatic number of G is sometimes written $\chi(G)$ (this is the Greek letter "chi").

For instance, in the graph 

we could color a Red, b and c Blue, and d Green, showing that the chromatic number of this graph is at most 3.

In this case, it's easy to see that the chromatic number is exactly 3: we couldn't use two colors because a, b, d are a triangle, so all three need to get different colors.

It's easy to show that the chromatic number is small—if I show you how to color a graph using, say, three colors, you know the chromatic number is at most three. Showing that the chromatic number is large—that there's no way to color a graph more efficiently—can be hard.

Theorem 5. *G has chromatic number 1 if and only if G has no edges.*

Proof. Easy. If G has chromatic number 1, any edge would be between two vertices with the same color, so there can't be any edges. If G has no edges, we can just make all vertices red trivially. □

Theorem 6. *G has chromatic number 2 if any only if G is bipartite and has at least one edge.*

Proof. Suppose G has chromatic number 2. Then there is some way of coloring G using only two colors, say red and blue. Let R be the red vertices and B the blue vertices. Then $R \cup B$ is all the vertices, $R \cap B = \emptyset$, and all edges must have one end in R and one in B. So R and B are exactly the division into two halves.

Conversely, if G is bipartite, we can partition the vertices into V_0 and V_1 so that all edges have one end in V_0 and one in V_1. So we color V_0 red and V_1 blue. □

What is a graph which cannot be 2-colored? An example is thre triangle.

Is this the only example? No. Consider any odd cycle—an odd number of vertices arranged in a loop. Then there's no way to use only two colors. (It's easy to use three—number the vertices around the loop, color the evens red, the odds blue, and the last vertex, where you run into a conflict, can be colored green.)

A typical application of coloring is in scheduling problems. For instance, suppose we schedule delivery vans, and have to make deliveries to different places in specified time intervals. We could make our vertices be the deliveries that have to be made, and place an edge between two vertices when the times are close enough to create a conflict. Then the delivery vans we have are the colors: we assign a van to each delivery so that we don't assign the same van to do same things at the same time.