From http://www.cs.uky.edu/~lewis/cs-heuristic/text/class/more-np.html

[...]

On to another graph problem. This time we shall examine one of a very different nature. In this problem we ask about coloring the vertices of a graph so that adjacent ones are distinct. Here is the definition.

Chromatic Number (COLOR). *Given a graph and an integer k, is there a way to color the vertices with k colors such that adjacent vertices are colored differently?*

This is the general problem for coloring. A special case, map coloring can always be done with four colors. But as we shall see presently, the general problem is *NP*-complete when we must use more than four colors.

Theorem 5. *COLOR is NP-complete.*

Proof.

1. To show that COLOR is in *NP*, again just guess the method of coloring vertices and check it out.

2. To show that the problem is *NP*-hard we shall reduce 3-SAT to COLOR.
Suppose that we have r clauses which contain $n \leq r \times 3$ propositional variables.
We need to construct a graph which can be colored with n+1 colors if and only if the clauses are satisfiable.

Begin by making all of the variables {v1, ... , vn} and their complements $\{\bar{\text{v}}1, \ldots, \bar{\text{v}}\text{n}\}$ vertices of the graph. Then connect each variable to its complement.
They must be colored differently, so color one of each pair *false* and the other *true*.

Now we will force the *true* colors to be different from each other.
Introduce a new collection of vertices {x1, ... , xn} and connect them all together.
The n xi's now form a clique. Connect each xi to all of the vj and their complements except when i = j. Thus if we have n different *true* colors (call them $t_1, \ldots, t_n$) we may color the xi's with these. And, since neither vj or its complement is connected to xi one of these may also be colored with *ti*. So far we have colored:

a each xi with *ti*,
b either v_i or its complement with *ti*, and the other with *false*.

An example for three variables is depicted in Figure 2. Since shades of gray are difficult to see, we have used three for the *true* colors and have drawn as squares all of the vertices to be colored with the false color. Note that v1 and v2 are true while v3 is false.

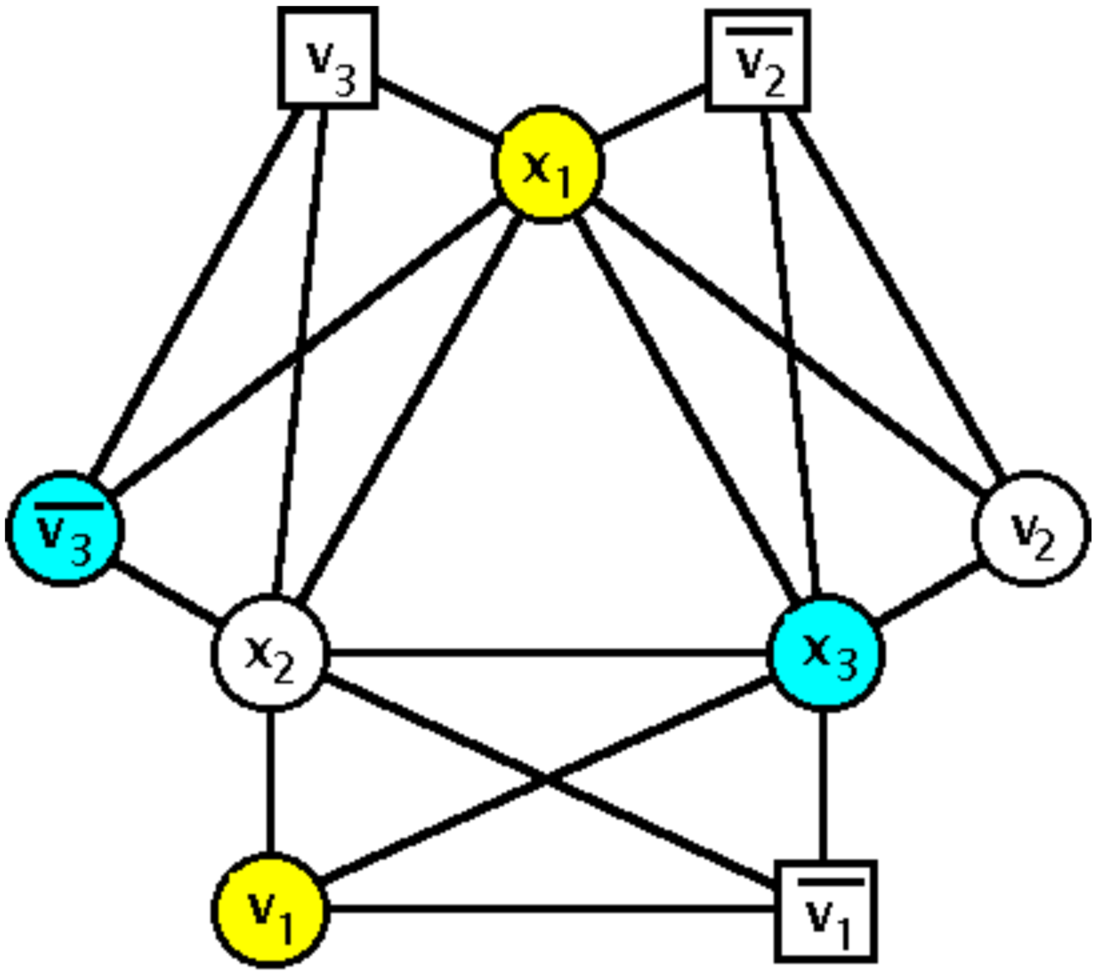

Figure 2 - Variables, True and False Colors

So far, so good. We have constructed a graph which cannot be colored with fewer than n+1 colors. And, the coloring scheme outlined above is the only one which will work. This is because the xi's must be different colors and either v_i or its complement has to be the (n+1)-st (*false*) color.

3-SAT enters at this point. Add a vertex for each clause and name them c1, ... , cr. Connect each of them to all the variables and their complements except for the three literals which are in the clause. We now have the following edges in our graph for all i and j between 1 and n, and k between 1 and r, except where otherwise noted.

a) $< v_i, \overline{v_i} >$,

b) $< x_i, x_j >$ for all $i \neq j$,

c) $< v_i, x_j >$ and $< \overline{v_i}, x_j >$ for all $i \neq j$, and

d) $< v_i, c_k >$ and $< \overline{v_i}, c_k >$ for literals not in c_k

Here's a recap. One of each variable and complement pair must be *false* and the other, one of the *true* colors. These *true*'s must be different because the xi's form a clique. Then, the clauses (the ci's) are connected to all of the literals *not* in the clause.

Suppose that there is a truth assignment to the variables which satisfies all of the clauses. Color each true literal with the appropriate *ti* and color its complement *false*. Examine one of the clauses (say, ci). One of its literals must have been colored with one of the *true* colors since the clause is satisfied. The vertex ci can be colored that way too since it is not connected to that literal. That makes exactly n+1 colors for all the vertices of the graph.

If there is no truth assignment which satisfies all of the clauses, then for each of these assignments there must be a clause (again, say ci) which has all its literals colored with the *false* or (n+1)-st color. (Because otherwise we would have a satisfying truth assignment and one of each literal pair must be colored false if n+1 colors are to suffice.) This means that ci is connected to vertices of every *true* color since it is connected to all those it does not contain. And since it is connected to all but three of the literal vertices, it must be connected to a vertex colored *false* also since there are at least three variables. Thus the graph cannot be colored with only n+1 colors.

Since constructing the graph takes polynomial time, we have shown that 3-SAT $\preccurlyeq_p$ COLOR and thus COLOR is *NP*-complete. □

An interesting aspect of the COLOR problem is that it can be almost immediately converted into a scheduling problem. In fact, one that is very familiar to anyone who has spent some time in academe. It is the problem of scheduling final examinations which we examined earlier.

Examination Scheduling (EXAM). *Given a list of courses, a list of conflicts between them, and an integer k; is there an exam schedule consisting of k dates such that there are no conflicts between courses which have examinations on the same date?*

Here is how we shall set up the problem. Assign courses to vertices, place edges between courses if someone takes both, and color the courses by their examination dates, so that no two courses taken by the same person have the same color.

[…]