VERTEX COLORING: WELSH-POWELL ALGORITHM

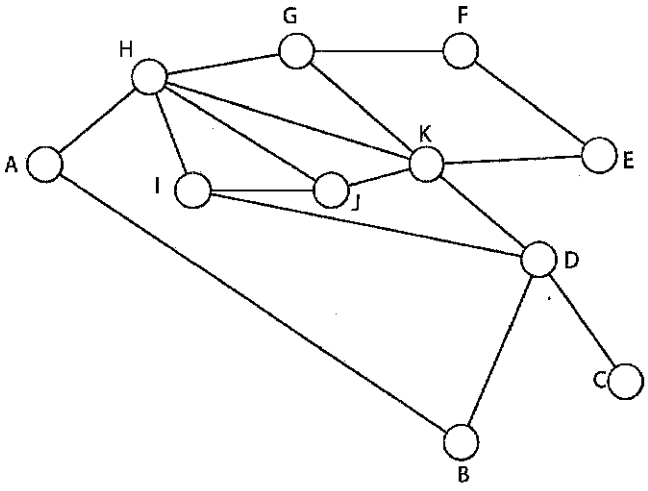
One algorithm that gives a good solution to a vertex-coloring problem is the *Welsh-Powell algorithm*. It may not always give the best solution, but it will usually perform better than just coloring the vertices without a plan will.

The Welsh-Powell algorithm consists of the following steps:

1. Find the valence for each vertex.
2. List the vertices in order of descending valence (you can break ties any way you wish).
3. Color the first vertex in the list (the vertex with the highest valence) with color 1.
4. Go down the list and color every vertex not connected to the colored vertices above the same color. Then cross out all colored vertices in the list.
5. Repeat the process on the uncolored vertices with a new color – always working in descending order of valence until all the vertices have been colored.

The idea is that, by starting with the vertices that have the highest valences, you will be able to take care of the vertices with the largest number of conflicts as early as possible.

EXAMPLE:



Vertex	Valence
A	2
B	2
C	1
D	4
E	2
F	2
G	3
H	5
I	3
J	3
K	5

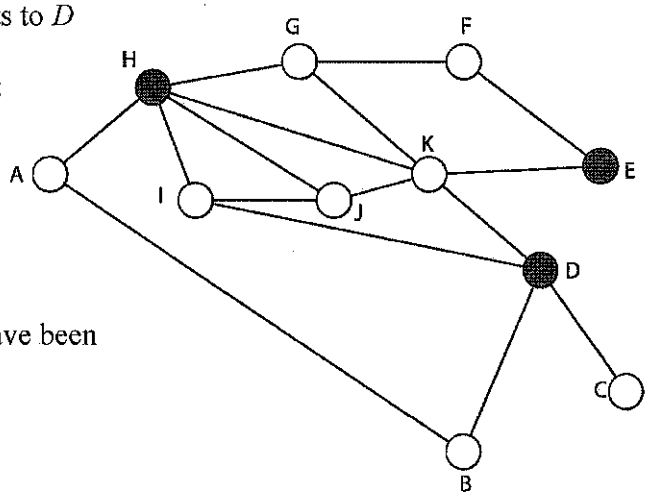
Arrange the list of vertices in descending order of valence. If there's a tie, you can randomly choose how to break it. (To get the ordering below, we used alphabetical order to break ties.) The new order will be:

H, K, D, G, I, J, A, B, E, F, C

Now we color the vertices, in the order listed above. Here's the thought process for the first color:

H color red
K don't color red since it connects to *H*
D color red
G don't color red since it connects to *H*
I don't color red since it connects to *H*
J don't color red since it connects to *H*
A don't color red since it connects to *H*
B don't color red since it connects to *D*
E color red
F don't color red since it connects to *E*
C don't color red since it connects to *D*

Now the graph should look like this:



If we now ignore the vertices that have been already colored, we're left with:

K, G, I, J, A, B, F, C

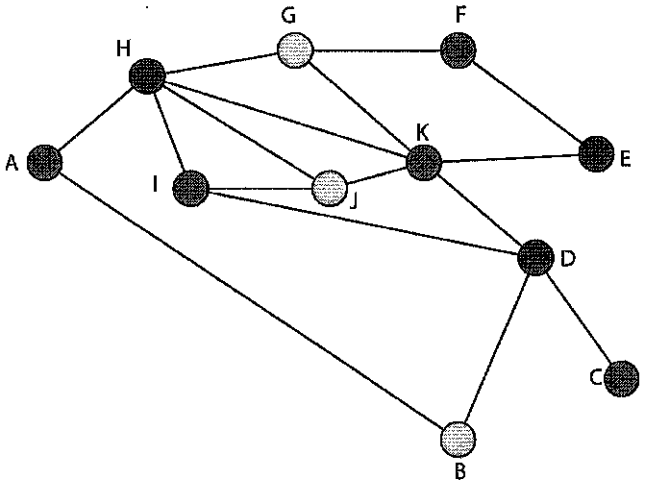
We can repeat the process now with a second color (blue):

K color blue
G don't color blue since it connects with *K*
I color blue
J don't color blue since it connects with *I*
A color blue
B don't color blue since it connects with *A*
F color blue
C color blue

Again, cross out the colored vertices, leaving you with *G, J, B*, and start at the top of the new list with a new color (yellow):

G color yellow
J color yellow
B color yellow

The final colored graph is shown at right.



We see that by using Welsh-Powell we can get away with using just three colors. In this case, three colors turns out to be the optimal solution since the graph contains at least one triangle (for example, there's the triangle *HIJH*), and it will always take at least 3 colors to color a graph with a triangle in it.

QUESTION

Why would a graph with a triangle in it require at least three colors?

EXERCISE

What would happen if you don't reorder the list of vertices according to their valence? That is to say, what if you just start going through the vertices in alphabetical order? How many colors would be necessary? Color the graph below to find the answer.