In-Class Problems Week 9, Fri.

Problem 1.
A portion of a computer program consists of a sequence of calculations where the results are stored in variables, like this:

$$\begin{array}{rrcl}
 & \text{Inputs:} & & a, b \\
\text{Step 1.} & c & = & a+b \\
2. & d & = & a*c \\
3. & e & = & c+3 \\
4. & f & = & c-e \\
5. & g & = & a+f \\
6. & h & = & f+1 \\
 & \text{Outputs:} & & d, g, h
\end{array}$$

A computer can perform such calculations most quickly if the value of each variable is stored in a *register*, a chunk of very fast memory inside the microprocessor. Programming language compilers face the problem of assigning each variable in a program to a register. Computers usually have few registers, however, so they must be used wisely and reused often. This is called the *register allocation* problem.

In the example above, variables a and b must be assigned different registers, because they hold distinct input values. Furthermore, c and d must be assigned different registers; if they used the same one, then the value of c would be overwritten in the second step and we'd get the wrong answer in the third step. On the other hand, variables b and d may use the same register; after the first step, we no longer need b and can overwrite the register that holds its value. Also, f and h may use the same register; once $f+1$ is evaluated in the last step, the register holding the value of f can be overwritten.

(a) Recast the register allocation problem as a question about graph coloring. What do the vertices correspond to? Under what conditions should there be an edge between two vertices? Construct the graph corresponding to the example above.

(b) Color your graph using as few colors as you can. Call the computer's registers $R1$, $R2$ etc. Describe the assignment of variables to registers implied by your coloring. How many registers do you need?

(c) Suppose that a variable is assigned a value more than once, as in the code snippet below:

$$\begin{aligned}
&\dots \\
t &= r+s \\
u &= t*3 \\
t &= m-k \\
v &= t+u \\
&\dots
\end{aligned}$$

How might you cope with this complication?

Problem 2.

A basic example of a simple graph with chromatic number n is the complete graph on n vertices, that is $\chi(K_n) = n$. This implies that any graph with K_n as a subgraph must have chromatic number at least n. It's a common misconception to think that, conversely, graphs with high chromatic number must contain a large complete subgraph. In this problem we exhibit a simple example countering this misconception, namely a graph with chromatic number four that contains no *triangle*—length three cycle—and hence no subgraph isomorphic to K_n for $n \geq 3$. Namely, let G be the 11-vertex graph of Figure 1. The reader can verify that G is triangle-free.

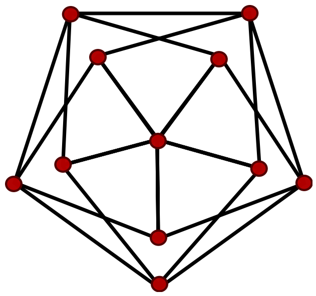

Figure 1 Graph G with no triangles and $\chi(G) = 4$.

(a) Show that G is 4-colorable.

(b) Prove that G can't be colored with 3 colors.

Problem 3.

In this problem, we examine an interesting connection between propositional logic and 3-colorings of certain special graphs. In the graph shown in Figure 2, designate the vertices connected in the triangle on the left as *color-vertices*. Since each color-vertex is adjacent to the other two, they must have different colors in any coloring of the graph. The colors assigned to the color-vertices will be called **T**, **F** and **N**. The dotted lines indicate edges to the color-vertex **N**.

(a) Prove that there exists a 3-coloring of the graph iff neither P nor Q are colored N.

(b) Argue that the graph in Figure 2 acts like a two-input OR-gate: a valid 3-coloring of the graph has the vertex labelled (P OR Q) colored **T** iff at least one of the vertices labelled P and Q are colored **T**.

(c) Changing the endpoint of one edge in Figure 2 will turn it into a two-input AND simulator. Explain.

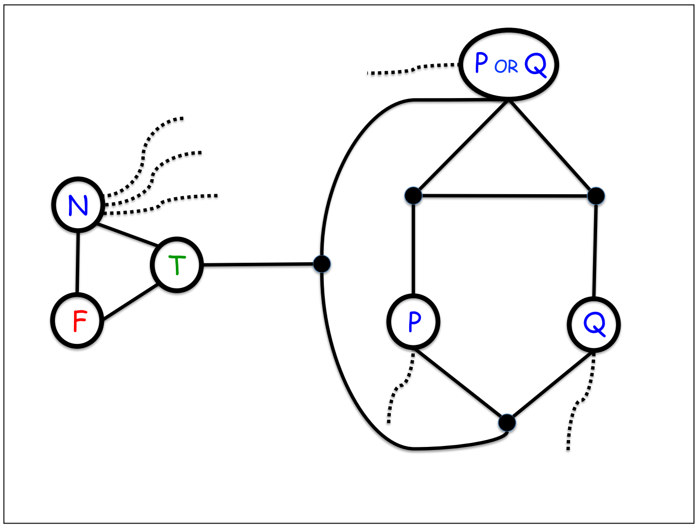

Figure 2 A 3-color OR-gate

Problem 4.

False Claim. *Let G be a graph whose vertex degrees are all $\leq k$. If G has a vertex of degree strictly less than k, then G is k-colorable.*

(a) Give a counterexample to the False Claim when $k = 2$.

(b) Underline the exact sentence or part of a sentence that is the first unjustified step in the following bogus proof of the False Claim.

Bogus proof. Proof by induction on the number n of vertices:

The induction hypothesis $P(n)$ is:

> Let G be an n-vertex graph whose vertex degrees are all $\leq k$. If G also has a vertex of degree strictly less than k, then G is k-colorable.

Base case: ($n = 1$) G has one vertex, the degree of which is 0. Since G is 1-colorable, $P(1)$ holds.

Inductive step: We may assume $P(n)$. To prove $P(n+1)$, let G_{n+1} be a graph with $n+1$ vertices whose vertex degrees are all k or less. Also, suppose G_{n+1} has a vertex v of degree strictly less than k. Now we only need to prove that G_{n+1} is k-colorable.

To do this, first remove the vertex v to produce a graph G_n with n vertices. Let u be a vertex that is adjacent to v in G_{n+1}. Removing v reduces the degree of u by 1. So in G_n, vertex u has degree strictly less than k. Since no edges were added, the vertex degrees of G_n remain $\leq k$. So G_n satisfies the conditions of the induction hypothesis $P(n)$, and so we conclude that G_n is k-colorable.

Now a k-coloring of G_n gives a coloring of all the vertices of G_{n+1}, except for v. Since v has degree less than k, there will be fewer than k colors assigned to the nodes adjacent to v. So among the k possible colors, there will be a color not used to color these adjacent nodes, and this color can be assigned to v to form a k-coloring of G_{n+1}.

■

(c) With a slightly strengthened condition, the preceding proof of the False Claim could be revised into a sound proof of the following Claim:

Claim. *Let G be a graph whose vertex degrees are all $\leq k$.*
*If $\langle$**statement inserted from below**$\rangle$ has a vertex of degree strictly less than k, then G is k-colorable.*

Circle each of the statements below that could be inserted to make the proof correct.

- G is connected and
- G has no vertex of degree zero and
- G does not contain a complete graph on k vertices and
- every connected component of G
- some connected component of G