The Englishman lives in the red house.
The Spaniard owns the dog.
The Norwegian lives in the first house on the left.
The green house is immediately to the right of the ivory house.
The man who eats Hershey bars lives in the house next to the man with the fox.
Kit Kats are eaten in the yellow house.
The Norwegian lives next to the blue house.
The Smarties eater owns snails.
The Snickers eater drinks orange juice.
The Ukrainian drinks tea.
The Japanese eats Milky Ways.
Kit Kats are eaten in a house next to the house where the horse is kept.
Coffee is drunk in the green house.
Milk is drunk in the middle house.

Discuss different representations of this problem as a CSP. Why would one prefer one representation over another?

6.8 Consider the graph with 8 nodes $A_1, A_2, A_3, A_4, H, T, F_1, F_2$. $A$ is connected to $A_{i+1}$ for all each $A_i$ is connected to $H$, $H$ is connected to $T$, and $T$ is connected to each $F_i$. Find a 3-coloring of this graph by hand using the following strategy: backtracking with conflict-directed \textbf{backjumping}, the variable order $A_1, H, A_4, F_1, A_2, F_2, A_3$, and the value order $R, G, B$.

6.9 Explain why it is a good heuristic to choose the variable that is most constrained but the value that is least constraining in a CSP search.

6.10 Generate random instances of map-coloring problems as follows: scatter $n$ points on the unit square; select a point $X$ at random, connect $X$ by a straight line to the nearest point $Y$ such that $X$ is not already connected to $Y$ and the line crosses no other line; repeat the previous step until no more connections are possible. The points represent regions on the map and the lines connect neighbors. Now try to find $k$-colorings of each map, for both $k = 3$ and $k = 4$, using min-conflicts, backtracking, backtracking with forward checking, and backtracking with MAC. Construct a table of average run times for each algorithm for values of $n$ up to the largest you can manage. Comment on your results.

6.11 Use the AC-3 algorithm to show that arc consistency can detect the inconsistency of the partial assignment $\{ WA = \text{green}, V = \text{red}\}$ for the problem shown in Figure 6.1.

6.12 What is the worst-case complexity of running AC-3 on a tree-structured CSP?

6.13 AC-3 puts back on the queue every arc $(X_k, X_i)$ whenever any value is deleted from the domain of $X_i$, even if each value of $X_k$ is consistent with several remaining values of $X_i$. Suppose that, for every arc $(X_k, X_i)$, we keep track of the number of remaining values of $X_i$ that are consistent with each value of $X_k$. Explain how to update these numbers efficiently and hence show that arc consistency can be enforced in total time $O(n^2d^2)$.,
6.14 The **Tree-CSP-Solver** (Figure 6.10) makes arcs consistent starting at the leaves and working backwards towards the root. Why does it do that? What would happen if it went in the opposite direction?

6.15 We introduced Sudoku as a CSP to be solved by search over partial assignments because that is the way people generally undertake solving Sudoku problems. It is also possible, of course, to attack these problems with local search over complete assignments. How well would a local solver using the *min-conflicts* heuristic do on Sudoku problems?

6.16 Define in your own words the terms constraint, backtracking search, arc consistency, backjumping, min-conflicts, and cycle cutset.

6.17 Suppose that a graph is known to have a cycle cutset of no more than $l$ nodes. Describe a simple algorithm for finding a minimal cycle cutset whose run time is not much more than $O(l^k)$ for a CSP with $n$ variables. Search the literature for methods for finding approximately minimal cycle cutsets in time that is polynomial in the size of the cutset. Does the existence of such algorithms make the cycle cutset method practical?