csp-exercises

March 26, 2019

1 6. Constraint Satisfaction Problems

6.1 How many solutions are there for the map-coloring problem in Figure Section **??**? How many solutions if four colors are allowed? Two colors?

6.2 Consider the problem of placing k knights on an $n \times n$ chessboard such that no two knights are attacking each other, where k is given and $k \leq n^2$.

1. Choose a CSP formulation. In your formulation, what are the variables?
2. What are the possible values of each variable?
3. What sets of variables are constrained, and how?
4. Now consider the problem of putting *as many knights as possible* on the board without any attacks. Explain how to solve this with local search by defining appropriate ACTIONS and RESULT functions and a sensible objective function.

6.3 [crossword-exercise]Consider the problem of Section **??** (not solving) crossword puzzles fitting words into a rectangular grid. The grid, which is given as part of the problem, specifies which squares are blank and which are shaded. Assume that a list of words (i.e., a dictionary) is provided and that the task is to fill in the blank squares by using any subset of the list. Formulate this problem precisely in two ways:

1. As a general search problem. Choose an appropriate search algorithm and specify a heuristic function. Is it better to fill in blanks one letter at a time or one word at a time?
2. As a constraint satisfaction problem. Should the variables be words or letters?

Which formulation do you think will be better? Why?

6.4 [csp-definition-exercise]Give precise formulations for each of the following as constraint satisfaction problems:

1. Rectilinear floor-planning: find non-overlapping places in a large rectangle for a number of smaller rectangles.
2. Class scheduling: There is a fixed number of professors and classrooms, a list of classes to be offered, and a list of possible time slots for classes. Each professor has a set of classes that he or she can teach.

3. Hamiltonian tour: given a network of cities connected by roads, choose an order to visit all cities in a country without repeating any.

6.5 Solve the cryptarithmetic problem in Figure Section **??** by hand, using the strategy of backtracking with forward checking and the MRV and least-constraining-value heuristics.

6.6 [nary-csp-exercise] Show how a single ternary constraint such as "$A + B = C$" can be turned into three binary constraints by using an auxiliary variable. You may assume finite domains. (*Hint:* Consider a new variable that takes on values that are pairs of other values, and consider constraints such as "X is the first element of the pair Y.") Next, show how constraints with more than three variables can be treated similarly. Finally, show how unary constraints can be eliminated by altering the domains of variables. This completes the demonstration that any CSP can be transformed into a CSP with only binary constraints.

6.7 [zebra-exercise] Consider the following logic puzzle: In five houses, each with a different color, live five persons of different nationalities, each of whom prefers a different brand of candy, a different drink, and a different pet. Given the following facts, the questions to answer are "Where does the zebra live, and in which house do they drink water?"

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_i is connected to A_{i+1} for all i, 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 backjumping, the variable order A_1, H, A_4, F_1, A_2, F_2, A_3, T, 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 $\{WA = green, V = red\}$ for the problem shown in Figure Section **??**.

6.12 Use the AC-3 algorithm to show that arc consistency can detect the inconsistency of the partial assignment $\{WA = red, V = blue\}$ for the problem shown in Figure Section **??**.

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

6.14 [ac4-exercise] 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 $O(n^2d^2)$.

6.15 The Tree-CSP-Solver (Figure Section **??**) 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?

6.16 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?

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

6.18 Define in your own words the terms constraint, commutativity, arc consistency, backjumping, min-conflicts, and cycle cutset.

6.19 Suppose that a graph is known to have a cycle cutset of no more than k nodes. Describe a simple algorithm for finding a minimal cycle cutset whose run time is not much more than $O(n^k)$ 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?

6.20 Consider the problem of tiling a surface (completely and exactly covering it) with n dominoes (2×1 rectangles). The surface is an arbitrary edge-connected (i.e., adjacent along an edge, not just a corner) collection of $2n$ 1×1 squares (e.g., a checkerboard, a checkerboard with some squares missing, a 10×1 row of squares, etc.).

1. Formulate this problem precisely as a CSP where the dominoes are the variables.
2. Formulate this problem precisely as a CSP where the squares are the variables, keeping the state space as small as possible. (*Hint:* does it matter which particular domino goes on a given pair of squares?)
3. Construct a surface consisting of 6 squares such that your CSP formulation from part (b) has a *tree-structured* constraint graph.
4. Describe exactly the set of solvable instances that have a tree-structured constraint graph.

ˆ1. @Ginsberg+al:1990 discuss several methods for constructing crossword puzzles. @Littman+al:1999 tackle the harder problem of solving them.