Computer Science Department
Stanford University
Comprehensive Examination in Artificial Intelligence
Autumn 1998

Solution Samples

1 Search

a. Several answers are possible. In one, nodes are partial colorings of the map, and edges denote adding a color for a blank country. For each country in C, pick a color not yet used for that country, and use **neighbor** to check if the coloring is valid. Continue depth-first with the next country if the coloring is valid, and backtrack otherwise.

A better idea is to spend the n^2 time it takes to create a graph of the countries by using the **neighbor** predicate. One can then use a depth-first or breadth-first search on a spanning tree of the graph. Finding neighbors during search is a more expensive option, since work is done anew in case of backtracking.

The graph solution is better because it essentially conforms to the most-constrained-variable heuristic. Other solutions are possible.

b. Failure can be declared by the first algorithm above when backtracking from a country for which the last available color has been tried. The second algorithm may find out earlier if it checks the degree of every node in the graph.

c. Yes. Heuristic repair algorithms are specifically designed for constraint satisfaction problems, of which the map coloring problem is an instance. How well it works, however, depends on the initialization.

d. You may be tempted to answer "no," because no cost has been defined for a solution. The correct answer, however, is "yes," because appropriate costs can be defined. For instance, heuristic repair can be seen as a tree search where each node is a complete coloring, and improperly-colored nodes are selected for modification. Then, the cost of a solution could be the number of changes necessary to repair the map.

2 Robotics

a. A straight line. The parameter is the turning angle of the screw.

b. A torus. The parameters are polar angles of the bob in some reference system.

3 Logic-Based Knowledge Representation

a. $\forall xy\,(\text{Woman}(x) \wedge \text{Man}(y) \wedge \forall z(\text{Woman}(z) \Rightarrow \text{Hates}(y,z)) \Rightarrow \neg\text{Loves}(x,y)$

b. $\forall s\,\text{HOLD}(\text{Result}(\text{CleanRoom}, s), \forall x(\text{Toy}(x) \Rightarrow \text{OffFloor}(x)))$

c.

$$\forall x\,(\text{Dog}(x) \Rightarrow \exists y(\text{Cat}(y) \wedge \text{Corresponds}(x,y)))$$
$$\forall x\,(\text{Cat}(x) \Rightarrow \exists y(\text{Dog}(y) \wedge \text{Corresponds}(x,y)))$$
$$\forall xyz\,\text{Corresponds}(x,y) \wedge \text{Corresponds}(x,z) \Rightarrow y = z$$
$$\forall xyz\,\text{Corresponds}(x,y) \wedge \text{Corresponds}(z,y) \Rightarrow x = z$$

4 Logic

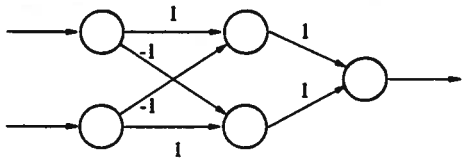
a. A FO model is a tuple (D,F,R,p,q) where the domain D is a set, F is a set of functions defined on D (each with its own arity), R is a set of relations defined on D (each with its own arity), and the function p (q) maps the function (predicate, resp.) symbols in the logic to F (R, resp.). The variable assignment, which maps variables to D, is also sometimes considered part of the model.

b. (i) yes, (ii) no, (iii) no, (iv) yes.

5 Learning

a. Yes. The additional units can only change the weights that inputs are multiplied by, so they can be subsumed by the two hidden units.

b. No. In a standard feed-forward architecture, each input unit can connect to both hidden units. This gives greater expressive power. For instance, XOR cannot be computed by the perceptron, or equivalently (from the previous answer) by the network in the question figure. On the other hand, XOR can be computed with a two-layer neural network:



Thresholds are zero in this network.

6 Planning

a. On(B, TABLE) ∧ Clear(B) ∧ On(A, TABLE) ∧ On(C, A) ∧ Clear(C).

b. On(C, TABLE) ∧ On(B, C) ∧ On(A, B) ∧ Clear(A). The last clause is optional.

c. Op(ACTION: Move(x, y, z),
PRECONDITION: On(x, y) ∧ Clear(x) ∧ Clear(z),
EFFECT: On(x, z) ∧ Clear(y) ∧ ¬ Clear(z))

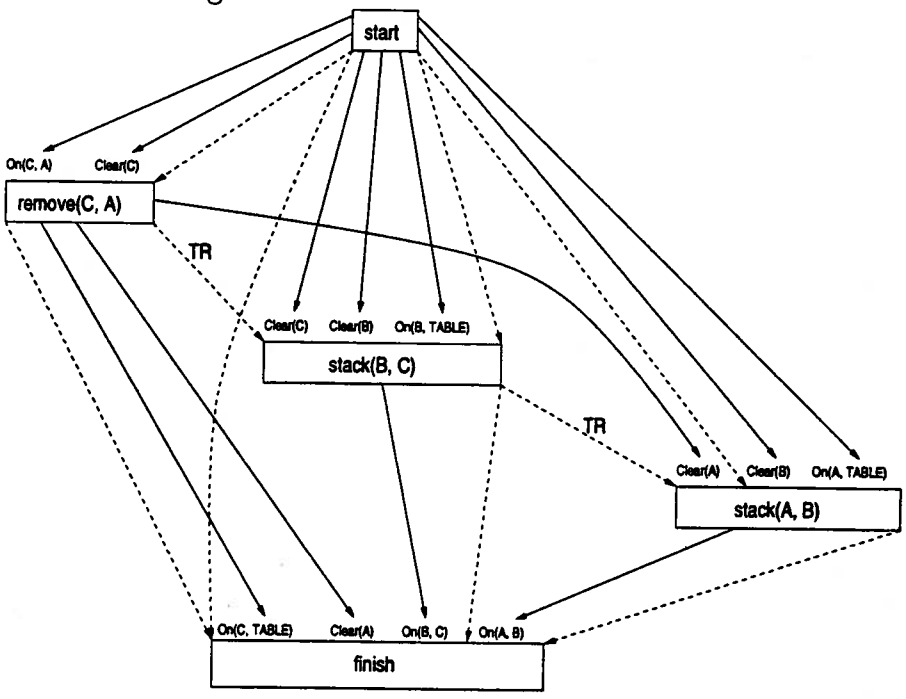
Op(ACTION: Remove(x, y),
PRECONDITION: On(x, y) ∧ Clear(x),
EFFECT: On(x, TABLE) ∧ Clear(y))

Op(ACTION: Stack(x, y),
PRECONDITION: On(x, TABLE) ∧ Clear(x) ∧ Clear(y),
EFFECT: On(x, y) ∧ ¬ Clear(y))

d. Yes, STRIPS can get lucky if the clauses in the goal staqte are given in the order On(C, TABLE), On(B, C), On(A, B), and if STRIPS creates subgoals top-down.

e. No. STRPIS will still attempt both bad moves mentioned in the description of the Sussman anomaly.

f. Here is a solution diagram.



The resulting plan is Remove(C, A), Stack(B, C), Stack(A, B).