CS 188 Spring 2005 — Introduction to AI — Stuart Russell — Midterm Solution

1. (12 pts.) True/False

(a) (2) *True/False*: There exists a task environment (PEAS) in which every agent is rational.
TRUE: e.g., let the performance measure be zero for every history.

(b) (2) *True/False*: Suppose agent A selects its action uniformly at random from the set of possible actions. There exists a deterministic, fully observable task environment in which A is rational.
TRUE: see (a). Many other environments have this property too.

(c) (2) *True/False*: No logical agent can behave rationally in partially observable environment.
FALSE: a logical agent can track the environment by inferring a description of the set of possible states, as in the wumpus world, and basing reflex rules on this description.

(d) (2) *True/False*: $\forall x, y \;\; x = y$ is satisfiable.
TRUE: satisfied by a model with exactly one object.

(e) (2) *True/False*: If θ unifies the atomic sentences α and β, then $\alpha \models \textsc{Subst}(\theta, \beta)$.
TRUE: If θ unifies α and β, then $\textsc{Subst}(\theta, \beta) \equiv \textsc{Subst}(\theta, \alpha)$, whcih is entailed by α for any θ.

(f) (2) *True/False*: In any finite state space, random-restart hillclimbing is an optimal algorithm.
TRUE (or maybe FALSE): it *will* eventually find the optimal solution, if necessary by generating it at random. But it has no way of knowing it has found the optimal solution, so strictly speaking it does not "return" it unless the time bound is finite—but in that case, it may not find it. The description in the book says that it returns the first goal it finds, which is obviously suboptimal; but one would not write the algorithm this way if optimal or near-optimal solutions were desired. We decided to give everyone full credit for this question.

2. (24 pts.) Search

Suppose there are two friends living in different cities on a map, such as the Romania map shown in Figure 3.2 of AIMA2e. On every turn, we can move each friend simultaneously to a neighboring city on the map. The amount of time needed to move from city i to neighbor j is equal to the road distance $d(i, j)$ between the cities, but on each turn the friend that arrives first must wait until the other one arrives (and calls the first on his/her cell phone) before the next turn can begin. We want the two friends to meet as quickly as possible. Let us formulate this as a search problem.

(a) (4) What is the state space? (You will find it helpful to define some formal notation here.)
States are all possible city pairs (i, j). The map is *not* the state space.

(b) (4) What is the successor function?
The successors of (i, j) are all pairs (x, y) such that $Adjacent(x, i)$ and $Adjacent(y, j)$.

(c) (2) What is the goal?
Be at (i, i) for some i.

(d) (4) What is the step cost function?
The cost to go from (i, j) to (x, y) is $max(d(i, x), d(j, y))$.

(e) (6) Let $SLD(i, j)$ be the straight-line distance between any two cities i and j. Which, if any, of the following heuristic functions are admissible? (If none, write NONE.)
(i) $SLD(i, j)$ (ii) $2 \cdot SLD(i, j)$ (iii) $SLD(i, j)/2$
In the best case, the friends head straight for each other in steps of equal size, reducing their separation by twice the time cost on each step. Hence (iii) is admissible.

(f) (4) *True/False*: There are completely connected maps for which no solution exists.
TRUE: e.g., a map with two nodes connected by one link.

3. (18 pts.) Propositional logic

(a) (9) Which of the following are entailed by the sentence $(A \lor B) \land (\neg C \lor \neg D \lor E)$?

i. $(A \lor B)$
ENTAILED: simple AND-elimination.

ii. $(A \lor B \lor C) \land (B \land C \land D \Rightarrow E)$
ENTAILED: $(B \land C \land D \Rightarrow E)$ is equivalent to $(\neg B \lor \neg C \lor \neg D \lor E)$, so this simply weakens the clause by introducing another disjunct.

iii. $(A \lor B) \land (\neg D \lor E)$
NOT ENTAILED: this removes the $\neg C$ literal, which strengthens the clause.

(b) (3) *True/False*: Every nonempty propositional clause, by itself, is satisfiable.
TRUE: a clause is a disjunction of literals, and its models are the union of the sets of models of the literals. Note that $False$ is unsatisfiable, but it is really another name for the empty clause. Also $A \land \neg A$ is unsatisfiable, but that's two clauses not one.

(c) (6) *True/False*: Every set of five 3SAT clauses is satisfiable, provided that each clause mentions exactly three distinct variables.
TRUE: a 3SAT clause rules out 1/8 of all possible models, so five clauses can rule out no more than 5/8 of the models.

4. (22 pts.) Logical knowledge representation

(a) (12) Which of the following are semantically and syntactically correct translations of "Everyone's zipcode within a state has the same first digit"?

i. $\forall x, s, z_1 \ [State(s) \land LivesIn(x,s) \land Zip(x) = z_1] \Rightarrow$
$[\forall y, z_2 \ LivesIn(y,s) \land Zip(y) = z_2 \Rightarrow Digit(1, z_1) = Digit(1, z_2)]$.
CORRECT

ii. $\forall x, s \ [State(s) \land LivesIn(x,s) \land \exists z_1 \ Zip(x) = z_1] \Rightarrow$
$[\forall y, z_2 \ LivesIn(y,s) \land Zip(y) = z_2 \land Digit(1, z_1) = Digit(1, z_2)]$.
INCORRECT: the existential either has the wrong scope or has $\Rightarrow$ as its main connective. The second universal has $\land$ as its main connective.

iii. $\forall x, y, s \ State(s) \land LivesIn(x,s) \land LivesIn(y,s) \Rightarrow Digit(1, Zip(x) = Zip(y))$.
INCORRECT: this is syntactically ill-formed because $Zip(x) = Zip(y)$ cannot be a term.

iv. $\forall x, y, s \ State(s) \land LivesIn(x,s) \land LivesIn(y,s) \Rightarrow Digit(1, Zip(x)) = Digit(1, Zip(y))$.
CORRECT: this is a more concise form of (i).

(b) (10) It was stated in lecture that a complete representation of the rules of chess in propositional logic would be unmanageably large—perhaps thousands of times larger than the first-order logic version. Which of the following are valid reasons for this?

i. The rules of chess are very complicated.
NOT VALID: this is equally true for the FOL representation.

ii. A chess game can go on for hundreds of moves.
VALID: a strict PL representation needs a complete copy of the axioms for every time step.

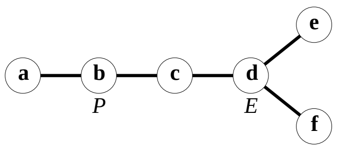
iii. There are several types of pieces.
NOT VALID: this is equally true for the FOL representation, which does not benefit much from the ability to quantify over (reified) types.

iv. There are several pieces of each type.
VALID: in the obvious formulation, PL needs separate axioms for all 16 pawns, for example, whereas FOL can quantify over pawns and colors. It's possible to use a predication on "squares that have pawns," which is much less profligate, although still needs one copy for each color.

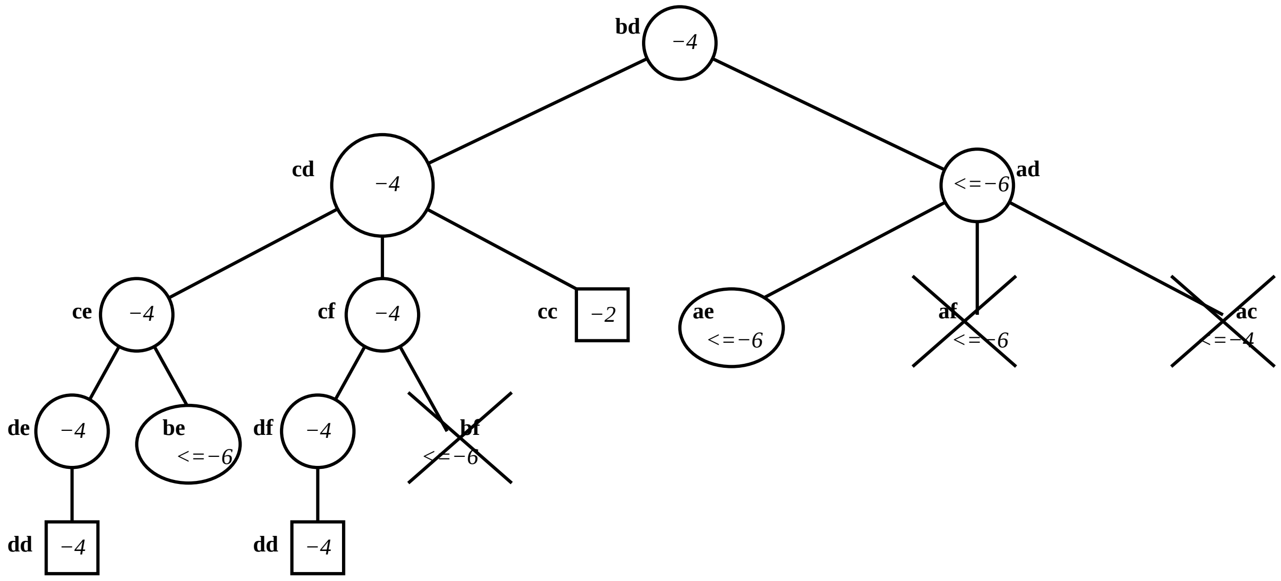
v. There are 64 squares on the board.
VALID: PL needs axioms for every combination of initial and final squares for moves.

5. (24 pts.) Game playing

Imagine that, in Q.2, one of the friends wants to avoid the other. The problem then becomes a two-player *pursuit-evasion* game. We assume now that the players take turns moving. The game ends only when the players are on the same node; the terminal payoff to the pursuer is minus the total time taken. (The evader "wins" by never losing.) Consider the following simple map, where the cost of every arc is 1 and initially the pursuer P is at node **b** and the evader E is at node **d**.



Here is a partially constructed game tree for this map. Each node is labelled with the P, E positions. P moves first. The values of the leaves marked "?" are currently unknown.



(a) (3) Mark the values of the terminal nodes.
See figure; the values are just (minus) the number of steps along the path from the root. This was stated clearly in the question, but some people confused values of game tree nodes with h-costs in search problems, and wrote 0 for the terminal nodes. Others used g-costs.

(b) (6) Inside each internal node, write the strongest fact you can infer about its value (either a number, one or more inequalities such as "≥ 14", or a "?").
See figure; note that there is both an upper bound and a lower bound for the left child of the root.

(c) (6) Can shortest-path lengths on the map be used to bound the values of the "?" leaves. If so, why and how? If not, why not?
The shortest-path length between the two players is a lower bound on the total capture time (here the players take turns, so no need to divide by two), so the "?" leaves have a capture time greater than or equal to the sum of the cost from the root and the shortest-path length. Notice that this bound is derived when the Evader plays very badly. The true value of a node comes from best play by both players, so we can get better bounds by assuming better play. For example, we can get a better bound from the cost when the Evader simply moves backwards and forwards rather than moving towards the Pursuer.

(d) (3) Mark inequalities on all the "?" leaves according to the method in (c). Remember the cost to get to each leaf as well as the cost to solve it.
See figure (we have used the simple bounds).

(e) (6) Now suppose the tree as given, with the leaf bounds from (d), was evaluated left-to-right. CIRCLE those nodes "?" nodes that would *not* need to be expanded further, given the bounds from part (d), and

CROSS OUT those that need not be considered at all.

See figure; notice that once the right child is known to have a value below –6, the remaining successors need not be considered.

(f) (10 extra credit) Can you say anything precise about who wins the game on a map that is a tree?

The pursuer always wins if the tree is finite. To prove this, let the tree be rooted as the pursuer's current node. (I.e., pick up the tree by that node and dangle all the other branches down.) The evader must either be at the root, in which case the pursuer has won, or in some subtree. The pursuer takes the branch leading to that subtree. This process repeats at most d times, where d is the maximum depth of the original subtree, until the pursuer either catches the evader or reaches a leaf node. Since the leaf has no subtrees, the evader must be at that node.