CS188 Spring 1993 — Intro to AI — Stuart Russell — Final solutions

1. (12 pts.) Definitions Provide brief, *precise* definitions of the following:

(a) Turing test: a test designed to indicate whether or not a system can be said to be intelligent. If a computer is approximately indistinguishable from a human when interrogated over a terminal, it passes.

(b) Unifier: a substitution for the variables in two atomic sentences that makes them identical.

(c) Coercible world: a world that can be forced into a known state even if it can't be sensed.

(d) Partial-order planner: a planner that entertains plan descriptions containing steps that need not be ordered with respect to each other.

(e) Singly-connected network: a belief network in which every pair of nodes are connected by at most one directed path.

(f) Perceptron: a "one-layer" neural network: the inputs nodes are connected directly to output nodes with no hidden units.

2. (20 pts.) Logical Inference
From "Horses are animals", it follows that "The head of a horse is the head of an animal". Demonstrate that this inference is valid by carrying out the following steps:

(a) (6) Translate the premise and the conclusion into the language of first-order logic. Use three predicates:

- $\texttt{Head-of}(x, y)$: x is the head of y.
- $\texttt{Horse}(x)$: x is a horse.
- $\texttt{Animal}(x)$: x is an animal.

$\forall x(\texttt{Horse}(x) \rightarrow \texttt{Animal}(x))$

$\forall x \forall y(\texttt{Horse}(y) \wedge \texttt{Head-of}(x, y) \rightarrow \exists z(\texttt{Animal}(z) \wedge \texttt{Head-of}(x, z)))$

(b) (7) Negate the conclusion, and convert the premise and the negated conclusion into conjunctive normal form.

$\neg\forall x \forall y(\texttt{Horse}(y) \wedge \texttt{Head-of}(x, y) \rightarrow \exists z(\texttt{Animal}(z) \wedge \texttt{Head-of}(x, z)))$

$\neg\forall x \forall y(\neg(\texttt{Horse}(y) \wedge \texttt{Head-of}(x, y)) \vee \exists z(\texttt{Animal}(z) \wedge \texttt{Head-of}(x, z)))$

$\exists x \exists y \neg(\neg(\texttt{Horse}(y) \wedge \texttt{Head-of}(x, y)) \vee \exists z(\texttt{Animal}(z) \wedge \texttt{Head-of}(x, z)))$

$\exists x \exists y(\neg\neg(\texttt{Horse}(y) \wedge \texttt{Head-of}(x, y)) \wedge \neg\exists z(\texttt{Animal}(z) \wedge \texttt{Head-of}(x, z)))$

$\exists x \exists y(\texttt{Horse}(y) \wedge \texttt{Head-of}(x, y) \wedge \forall z \neg(\texttt{Animal}(z) \wedge \texttt{Head-of}(x, z)))$

$\exists x \exists y(\texttt{Horse}(y) \wedge \texttt{Head-of}(x, y) \wedge \forall z(\neg\texttt{Animal}(z) \vee \neg\texttt{Head-of}(x, z)))$

$\exists x \exists y \forall z(\texttt{Horse}(y) \wedge \texttt{Head-of}(x, y) \wedge (\neg\texttt{Animal}(z) \vee \neg\texttt{Head-of}(x, z)))$

$\forall z(\texttt{Horse}(b) \wedge \texttt{Head-of}(a, b) \wedge (\neg\texttt{Animal}(z) \vee \neg\texttt{Head-of}(a, z)))$

$\texttt{Horse}(b) \wedge \texttt{Head-of}(a, b) \wedge (\neg\texttt{Animal}(z) \vee \neg\texttt{Head-of}(a, z))$

(1) $\texttt{Horse}(b)$

(2) $\texttt{Head-of}(a, b)$

(3) $\neg\texttt{Animal}(z) \vee \neg\texttt{Head-of}(a, z)$

$\forall x(\texttt{Horse}(x) \rightarrow \texttt{Animal}(x))$

$\forall x(\neg\texttt{Horse}(x) \vee \texttt{Animal}(x))$

(4) $\neg\texttt{Horse}(x) \vee \texttt{Animal}(x)$

(c) (7) Use resolution to show that the conclusion follows from the premise.

Resolve (1) and (4), with unifier $\{x/b\}$, to yield

(5) Animal(b)

Resolve (3) and (5), with unifier $\{z/b\}$, to yield

(6) $\neg$Head-of(a, b)

Resolve (2) and (6), with unifier $\{\}$, to yield the empty clause.

3. (17 pts.) Search in games

(a) (3) The search is depth-first. You can follow the code and see that the recursive call to `backed-up-value` doesn't terminate until the stack reaches the depth limit. In this code, the recursion stack contains the sequence of moves and states.

(b) (4) See figure

(c) (3) See figure

(d) (2) See figure

(e) (2) See figure

(f) (3)

4. (16 pts.) Decision theory and the value of information

(a) (3) Choose the action with the highest expected outcome utility. Outcomes occur with probability $P(W_j|A_i, K)$, so we have
$argmax_{A_i} \sum_j U(W_j)P(W_j|A_i, K)$

(b) (4) ii) is correct. i) is ill-formed, because the first A_i is not in the scope of any max or sum, and inadequate because the last conditional probability omits the dependence on the action. iii) reverses the order of the averaging and summation: it essentially suggests that the choice of action will occur before the information is known, so the whole expression will always be 0.

(c) (6) If I don't peek, I win C/k on average. If I peek, I have a $1/k$ chance of finding the prize, in which case I choose the same door and win C; I have a $(k-1)/k$ chance of finding that the first door is empty, in which case I choose some other door (the new best action) with an expected winnings of $C/(k-1)$. Hence my expected total winnings are

$$\frac{1}{k} \times C + \frac{k-1}{k} \times \frac{C}{k-1} = \frac{2C}{k}$$

The value of information is the difference between this and my winnings without the peek, i.e. C/k.

(d) (3) False. After the first peek, it would be silly to pay for a second if I happen to find the prize with the first.

5. (12 pts.) Learning in agents

(a) (6)

i. (3) See figure

ii. (3) See figure

(b) (6) Basic idea: agent should check to see if the percept contains a recommended action, If not, it calls LP on the current examples to get a function, calls the function on the percept and returns the reult of the function. If the percept contains an action, it stores the percept with the previous examples and returns the action from it.

6. (15 pts.) Natural language

(a) B and C. A fails because it allows *either* two straight adverbs (slowly slowly) *or* a single prepositional phrase.

(b) eg "I ran quickly from my house along a river".

Pronoun → "I"
V → "ran"
Adv → "quickly"
Prep → "from", "along"
Det → "my", "a"
Noun → "house", "river"

(c) See figure

(d)
i. Exhibited by B,C
ii. Exhibited by B,C
iii. Exhibited by A,C
iv. Exhibited by B,C
v. Exhibited by A (maybe C also)

7. (18 pts.) Perception and robotics using belief networks

(a) i) and iii) are correct. ii) has the connection from the X to O reversed, which says that the sensor value causes the actual position.

(b) iii) is best because it has a subset of the arrows in i) but is still correct. i) has the state variables (X, V, F) fully connected to the next time step, which simply says that the current state depends on the previous state in some arbitrary way.

(c) ii)

(d) If we add nodes for each future time step until we get to the desired time, then calculate the posterior probability of the state variables given the evidence up to the present time, we will have predicted the trajectory.

(e) Two parts: 1) Find the ball in the camera array; simply locate the region of highest intensity. This can be done efficiently by random sampling until closely spaced high values are found (this method is robust against noise in the image also). 2) Convert from image (x',y') coordinates to real coordinates (x,y,z). The easiest way to do this is to calculate the distance of the ball from the camera using its apparent size (diameter inversely proportional to distance) and the fact that the pingpong ball has a known actual size. The simple geomtery suffices to recover x,y,z.

(f) For the purposes of hitting the ball, we want relative coordinates because the arm motion is also relative to the robot. But to decide where to hit it to, we will need absolute coordinates so it ends up on the table. What you *don't* want to do is calculate the ball position in absolute coordinates, calculate the robot arm motion in absolute coordinates, and subtract to get the appropriate motor commands to send to the arm. It would also be necessary to track both arm and ball continuously to make sure they meet.