CS 188 Spring 2021 | Introduction to Artificial Intelligence | MT Review: Logic Soluti

Q1. Propositional Logic

(a) Which of the following are correct?

In all cases, the question can be resolved easily by referring to the definition of entailment.

(i) $(A \wedge B) \implies C \models (A \implies C) \vee (B \implies C)$.

$(A \wedge B) \implies C \models (A \implies C) \vee (B \implies C)$ is true because the RHS is false only when both disjuncts are false, i.e., when A and B are true and C is false, in which case the LHS is also false. This may seem counterintuitive, and would not hold if $\implies$ is interpreted as "causes."

(ii) $(A \vee B) \wedge (\neg C \vee \neg D \vee E) \models (A \vee B) \wedge (\neg D \vee E)$.

$(A \vee B) \wedge (\neg C \vee \neg D \vee E) \models (A \vee B) \wedge (\neg D \vee E)$ is false; removing a disjunct allows fewer models.

(iii) $(A \iff B) \wedge (\neg A \vee B)$ is satisfiable.

$(A \iff B) \wedge (\neg A \vee B)$ *is* satisfiable; RHS is entailed by LHS so models are those of $A \iff B$.

(iv) $(A \iff B) \iff C$ has the same number of models as $(A \iff B)$ for any fixed set of proposition symbols that includes A, B, C.

$(A \iff B) \iff C$ does have the same number of models as $(A \iff B)$; half the models of $(A \iff B)$ satisfy $(A \iff B) \iff C$, as do half the non-models, and there are the same numbers of models and non-models.

(b) Minesweeper, the well-known computer game, is closely related to the Pacman world. A minesweeper world is a rectangular grid of N squares with M invisible mines scattered among them. Any square may be probed by the agent; instant death follows if a mine is probed. Minesweeper indicates the presence of mines by revealing, in each probed square, the *number* of mines that are directly or diagonally adjacent. The goal is to probe every unmined square.

(i) Let $X_{i,j}$ be true iff square $[i, j]$ contains a mine. Write down the assertion that exactly two mines are adjacent to [1,1] as a sentence involving some logical combination of $X_{i,j}$ propositions.

This is a disjunction with 28 disjuncts, each one saying that two of the neighbors are true and the others are false. The first disjunct is

$$X_{2,2} \wedge X_{1,2} \wedge \neg X_{0,2} \wedge \neg X_{0,1} \wedge \neg X_{2,1} \wedge \neg X_{0,0} \wedge \neg X_{1,0} \wedge \neg X_{2,0}$$

The other 27 disjuncts each select two different $X_{i,j}$ to be true.

(ii) Generalize your assertion from the previous part by explaining how to construct a CNF sentence asserting that k of n neighbors contain mines. How many disjuncts would we need to use?

There will be $\binom{n}{k}$ disjuncts, each saying that k of the n symbols are true and the others false.

(iii) How can an agent use DPLL to prove that a given square does (or does not) contain a mine, ignoring the global constraint that there are exactly M mines in all? Formulate this as precisely as you can.

For each of the cells that have been probed, take the resulting number n revealed by probing the square and construct a sentence with $\binom{q}{n}$ disjuncts, where q is the number of neighbors that square has, where q is usually 8. Conjoin all the sentences together. Then use DPLL to answer the question of whether this sentence entails $X_{i,j}$ for the particular i, j pair you are interested in.

(iv) Suppose that we are no longer ignoring the global constraint as mentioned in the previous part, and we construct it using your formulation. How does the number of clauses depend on M and N? Suggest a way to modify DPLL so that the global constraint does not need to be represented explicitly.

To encode the global constraint that there are M mines altogether in a grid with N squares, we can construct a disjunction with $\binom{N}{M}$ disjuncts. Remember, $\binom{N}{M} = \frac{N!}{M!(N-M)!}$. So for a Minesweeper game with 100 cells and 20 mines, this will be more than 10^{39}, and thus cannot be represented in any computer. However, we can represent the global constraint within the DPLL algorithm itself. We add the parameter *min* and *max* to the DPLL function; these indicate the minimum and maximum number of unassigned symbols that must be true in the model. For an unconstrained problem the values 0 and N will be used for these parameters. For a minesweeper problem the value M will be used for both *min* and *max*. Within DPLL, we fail (return false) immediately if *min* is less than the number of remaining symbols, or if *max* is less than 0. For each recursive call to DPLL, we update *min* and *max* by subtracting one when we assign a true value to a symbol.

Q2. First Order Logic

Consider a vocabulary with the following symbols:

- $Occuption(p, o)$: Predicate. Person p has occuption o.
- $Customer(p1, p2)$: Predicate. Person $p1$ is a customer of person $p2$.
- $Boss(p1, p2)$: Predicate. Person $p1$ is a boss of person $p2$.
- $Doctor, Surgeon, Lawyer, Actor$: Constants denoting occupations.
- $Emily, Joe$: Constants denoting people.

Use these symbols to write the following assertions in first-order logic:

(v) Emily is either a surgeon or a lawyer.
$O(E, S) \vee O(E, L)$

(vi) Joe is an actor, but he also holds another job.
$O(J, A) \wedge \exists p\ p \neq A \wedge O(J, p)$

(vii) All surgeons are doctors.
$\forall p\ O(p, S) \Rightarrow O(p, D)$

(viii) Joe does not have a lawyer (i.e., is not a customer of any lawyer).
$\neg \exists p\ C(J, p) \wedge O(p, L)$

(ix) Emily has a boss who is a lawyer.
$\exists p\ B(p, E) \wedge O(p, L)$

(x) There exists a lawyer all of whose customers are doctors.
$\exists\ pO(p, L) \wedge \forall\ qC(q, p) \Rightarrow O(q, D)$

(xi) Every surgeon has a lawyer.
$\forall p\ O(p, S) \Rightarrow \exists\ qO(q, L) \wedge C(p, q)$