Introduction to Artificial Intelligence

UNIZG FER, AY 2021/2022

Exercises, v2

7 Logic programming in Prolog

1 (T) Logic programming in Prolog boils down to defining a program as a sequence of definite clauses. **What is the advantage and what is the disadvantage of a Prolog program being made of definite clauses rather than FOL formulas?**

A The advantage is that the proof procedure is sound, the disadvantage is that the expressivity of definite clauses is limited

B The advantage is that proving formulas is of linear time complexity, the disadvantage is that the declarative meaning of definite clauses may deviate from their meaning in FOL

C The advantage is that derivations can be carried out effectively using backward chaining, the disadvantage is that some FOL formulas are not expressible as definite clauses

D The advantage is that the proof procedure is complete, the disadvantage is that it is not decidable, hence in some cases the proof procedure may not terminate

2 (T) Programs in Prolog are made of definite clauses, which correspond to rules and facts. **What is the difference between rules and facts in Prolog?**

A Rules have at least one negative literal and exactly one positive literal, while facts are unit clauses with only one positive literal

B Rules have at most one negative literal and at least one positive literal, while facts are unit clauses with only one positive literal

C Rules have an antecedent with at most one positive literal, while facts correspond to rules with an antecedent that is always false

D Rules have at least one negative literal and exactly one positive literal, while facts are unit clauses with at least one negative and exactly one positive literal

3 (P) Horn logic uses Horn clauses, which are a subset of FOL formulas. A building block of Prolog programs is a definite clause, which is a type of Horn clause. **Which of the following formulas can be written in the form of a definite clause or a conjunction of a number of such clauses**?

A $\neg P \rightarrow Q$ B $\neg(P_1 \wedge P_2) \rightarrow Q$ C $(P_1 \vee P_2) \rightarrow Q$ D $P \rightarrow (Q_1 \vee Q_2)$

4 (C) A Prolog knowledge base describes the relationships among biological species. It contains the following facts and rules:

```
subspecies(mammal, endotherm).
subspecies(bat, mammal).
subspecies(bird, endotherm).
descendant(X, Y) :- subspecies(X, Y).
descendant(X, Y) :- subspecies(X, Z), descendant(Z, Y).
flies(X) :- descendant(X, bird).
```

Using this knowledge base, we launch a query `flies(bat)`. Disappointingly, Prolog responds with a `False`. **How many nodes are there in Prolog's proof tree for this query?**

A 10 B 8 C 12 D 6

8 Expert systems

5 (T) Both Prolog and CLIPS use rules to infer new knowledge. However, the two frameworks do differ. **Which is one of the ways in which Prolog and CLIPS differ?**

- A Prolog uses variables, whereas CLIPS does not
- B CLIPS uses foward chaining, whereas Prolog uses backward chaining
- C CLIPS allows for the definition of facts, whereas Prolog does not
- D Both Prolog and CLIPS use backward chaining, but CLIPS does not use the resolution rule

6 (P) CLIPS is an expert system shell capable of running rule-based programs. Consider the following CLIPS program comprising two facts and two rules. **When this program is run, what will it print to screen?**

```
(assert (a 2))
(assert (b 2))
(defrule F (declare (salience 20))
   ?x1 <-(a ?v1) ?x2 <-(b ?v2) (test (< ?v1 200))
   => (retract ?x1) (retract ?x2) (assert (a ?v2)) (assert (b (+ ?v1 ?v2))))
(defrule output (a ?v) => (printout t ?v))
```

A 178 B 202 C 288 D 198

7 (C) An expert system's knowledge base contains the following rules:

(1) IF $A = a_1 \wedge B = b_2$ THEN $C = c_2$
(2) IF $F = f_3 \wedge B = b_2$ THEN $C = c_1$
(3) IF $E = e_1 \vee (D = d_1 \wedge E = e_2)$ THEN $A = a_1$
(4) IF $F = f_3 \vee Q = q_2$ THEN $D = d_1$
(5) IF $F = f_3$ THEN $E = e_2$
(6) IF $D = d_3 \vee G = g_1$ THEN $A = a_2 \wedge B = b_2$
(7) IF $A = a_1$ THEN $G = g_1 \wedge E = e_1$

We use the system to derive the value of variable C using backward chaining. In case of rule conflict, the rule with the lower ordinal number takes precedence. If queried by the system, the user will reply with $D = d_2$ and $F = f_3$. **What will the expert system do in the course of deriving the value of variable C?**

- A Fire 6 rules and derive $C = c_1$
- B Derive $E = e_1$ and subsequently derive $E = e_2$
- C Reject rule 2 and fire rule 1
- D Terminate with 6 facts in working memory