Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Symbolic AI

Date: Monday 23rd May 2011

Time: 09:45 - 11:45

Please answer Question ONE in Section A and TWO Questions from Section B.

For full marks your answers should be concise as well as accurate. Marks will be awarded for reasoning and method as well as being correct.

This is a CLOSED book examination

The use of electronic calculators is NOT permitted

Section A

You should answer question 1: this question carries 30 marks

1. a) Consider the following Prolog program.

```
p([H | T], X) :-
    p(H, T, X).

p(X, [], X).
p(X, [H | T], Y) :-
    H < X,
    !,
    p(H, T, Y).
p(X, [_H | T], Y) :-
    p(X, T, Y).
```

   i. What steps would this program carry out, and what would be the result, if you called it with

   `| ?- p([1,3,6,4], X).`

   [6 marks]

   ii. What steps would it carry out, and what would be the result, if you called it with

   `| ?- p(L, 9).`

   [2 marks]

   b) Consider the following Prolog program:

```
r(0).
r(I) :-
    assertz(p(I)),
    J is I-1,
    r(J).
```

   i. What steps would this program carry out, and what would be the result, if you called it with

   `| ?- r(4).`

   [6 marks]

   ii. What steps would it carry out, and what would be the result, if you reversed the order of the clauses in this program and called it with the same argument? [4 marks]

c) Consider the following Prolog program:

```
q([], L, L).
q(L, [], L).
q([H | T0], [H | T1], [H | T]) :-
    q(T0, T1, T).
q([H0 | T0], [H1 | T1], [H0 | T]) :-
    H0 @< H1,
    q(T0, [H1 | T1], T).
q([H0 | T0], [H1 | T1], [H1 | T]) :-
    H0 @< H1,
    q([H0 | T0], T1, T).
```

i. What steps would this program carry out, and what variable bindings would be obtained, if you called it with

```
q([agree=thirdsing, cat=np, definite=yes],
  [case=subject, cat=D],
  L).
```

[6 marks]

ii. What steps would it carry out, and what variable bindings would be obtained, if you called it with

```
q([agree=thirdsing, cat=np, definite=yes],
  [cat=D, case=subject],
  L).
```

[4 marks]

iii. What are the advantages when using a program like this for comparing feature sets of assuming that the sets are represented as *ordered* lists?
[2 marks]

Section B

Answer two questions from this section. Each question carries 35 marks.

2\. a) What is the difference between a 'context-free' grammar and a 'feature-based' grammar? You should illustrate your answer with examples that would be easier to account for using a feature-based grammar than with a context-free grammar. [10 marks]

b) *Briefly* outline the basic top-down and bottom-up algorithms for using a simple grammar to analyse the structure of an input sentence. Show the steps that *one* of these algorithms would perform when using the grammar below to analyse the sentence '*I know it died*'. [10 marks]

```
s ==> [np, vp].                   word('i', pronoun).
np ==> [pronoun].                 word('it', pronoun).
vp ==> [verb].                    word('know', verb).
vp ==> [verb, s].                 word('died', verb).
```

c) What is the fundamental rule of chart parsing? [2 marks] Show the steps that a left-corner chart parser would perform when analysing the sentence above with the given grammar. [8 marks]

d) Explain why grammars that consist solely of sets of rewrite rules have difficulty with situations where items occur in marked/non-canonical positions. [5 marks]

3. a) Natural language understanding systems often translate the input text into an expression in some logic. What is the purpose of this step? [8 marks].

b) State the 'principle of compositionality'. [2 marks] Explain why the sentences below pose a challenge for this principle. [8 marks]

(3) a. I broke a glass jam jar.
 b. I got caught in a jam.
 c. I want to buy a jar of jam.

c) Show how the grammar below would lead to the logical form like `forall(A, man(A) => exists(B, die(B) & patient(B, A)))` for the sentence '*Every man dies*'. [10 marks] What is the significance of the term `patient` in this logical form? [7 marks]

```
[cat=s, meaning=VP:NP]
         ==> [[cat np, meaning=NP, [cat=vp, meaning=VP]].
[cat=np, meaning=DET:N]
         ==> [[cat=det, meaning=DET, [cat=noun, meaning=NP]].
[cat=vp, meaning=V]] ==> [[cat=verb, meaning=V]].

word('every',
     [cat=det,
      meaning=lambda(P, lambda(Q, forall(X, (P:X => Q:X))))].
word('man', [cat=noun, meaning=lambda(U, man(U))]).
word('dies',
     [cat=verb,
      meaning=lambda(A, A:lambda(Y, exists(Z, die(Z) & patient(Z, Y))))])).
```

4. a) Explain why standard first-order logic is inadequate as a framework for sentences like (4)(a)–(c) [8 marks].

   (4) a. The man entered a shop.
       b. The president of the USA is the most powerful person on earth.
       c. She saw him.

   b) Russell extended first-order logic by introducing ‘reference terms’, so that (4)(a) would have a logical form like `exists(X, shop(X) & exists(Y, enter(Y) & agent(Y, ref(lambda(Z, man(Z)))) & object(Y, X)))`. What is wrong with Russell’s suggestion that this sentence would be true if there was exactly one man and that man entered a shop? [10 marks]

   c) Let `minutes[S]` denote ‘the speaker’s view of the minutes’. How does the rule

```
exists(X, (minutes[S] => man(X))
          & forall(Y, (minutes[S] => man(Y)) => (X = Y))
          & exists(Y, shop(Y)
                      & exists(Z, enter(Z)
                                  & agent(Z, X)
                                  & object(Z, Y))))
```

   provide a better account of (4)(a)? Your answer should explain the role of the minutes in this expression, and should discuss why the rule refers to the speaker’s view of the minutes rather than just the minutes. [10 marks]

   d) Assume that the meanings of ‘*he*’ and ‘*him*’ are dealt with by using the term `ref(lambda(X, male(X) & salient(X))`, and that ‘*John*’ and ‘*Bill*’ are both known to be the names of male individuals. What would this account of reference say about (4)(d). [7 marks]

   (4) d. John hit Bill. Then he kicked him.

5. a) What is 'Horn-clause logic'? [3 marks] What advantages does Horn-clause logic have over full first-order logic as a programming language, and what disadvantages does it have as a language for representing the meanings of natural language sentences? [7 marks]

b) The following program provides a basic implementation of the 'model generation' approach to theorem proving for first-order logic. Explain what each element of this program is for, and show how it could be used to derive `s` from {`p, (p => q or r), (q => s), (r => s)`}. [15 marks]

```
prove(P) :-
    P.
prove(P) :-
    (R or S),
    (R => P),
    (S => P).

(P => Q) :-
    assert(P),
    (prove(Q) ->
        retract(P);
        (retract(P), fail)).
```

c) The clause for `P => Q` in this program implements 'constructive implication'. Explain the difference between this and 'material implication'. [5 marks]

d) What is meant by the statement that first-order logic is semi-decidable? [2 marks] Give an example of a rule and a query that would send the simple algorithm above into an infinite loop. How would you attempt to block such loops? [3 marks]