UNIVERSITY OF DUBLIN
TRINITY COLLEGE

Faculty of Engineering and Systems Sciences

DEPARTMENT OF COMPUTER SCIENCE

B.A.(Mod.) Computer Science

Junior Sophister Examination **Trinity Term 2002**

3BA2 Artificial Intelligence

Thursday 23rd May MANSION HOUSE 09.30 - 12.30

T. Fernando, P. Cunningham and M. Brady

Answer **five** questions, at least **one** from each section. Each question is worth 20 marks.

Section A

1. (a) What is a *Representation and Reasoning System* and what does it have to do with the *Symbol System Hypothesis* and the *Church-Turing Thesis*? (3 or 4 well-chosen lines should do.)

   (b) What is an *interpretation* and what does it have to do with your answer to part (a)? (2 or 3 well-chosen lines should do.)

   (c) What is *logical consequence* and what does it have to do with your answer to parts (a) and (b)? (2 or 3 well-chosen lines should do.)

   (d) What is the *Halting Problem* and what does it have to do with your answer to parts (a), (b) and (c)? (2 or 3 well-chosen lines should do.)

2. Consider the following Prolog knowledge base.

```
partner(batman,robin).
partner(X,Y) :- partner(Y,X).
```

   (a) Give (examples of) terms t and t' such that Prolog says `yes` to the query `partner(`t`,`t'`).` If no such terms exist, explain why not.

(b) Give terms t and t' such that Prolog answers `no` to the query `partner(`t`,`t'`)`. If no such terms exist, explain why not.

(c) Give terms t and t' such that Prolog answers neither `yes` nor `no` to `partner(`t`,`t'`)`. If no such terms exist, explain why not.

(d) Modify the knowledge base above so that the modified knowledge base satisfies all of the following conditions (i), (ii), (iii) and (iv).

(i) Prolog answers `yes` to the query

```
| ?- partner(batman,robin), partner(robin,batman).
```

(ii) Prolog answers `no` to the query

```
| ?- partner(batman,batman).
```

(iii) The modified knowledge base has exactly one clause mentioning `batman` and `robin`, and that clause is a fact (i.e. an atom). All its other clauses mention neither `batman` nor `robin`.

(iv) All clauses in the modified knowledge base are definite clauses (no *cuts*, no `findall`, no `setof` ...).

(You are allowed to modify the knowledge base only once to satisfy (i)-(iv) above.) *Hint.* A modified knowledge base with three clauses will do the job.

3. (a) What is an *integrity constraint* and what does it have to do with *models* of a knowledge base? How does it introduce *inconsistency*?

(b) Give at least two different uses of integrity constraints.

(c) What are *minimal conflicts* and what do they have to do with integrity constraints?

(d) Consider the following Prolog clauses for meta-interpreting a knowledge base encoded as in our text, *Computational Intelligence* (i.e. using meta-predicates `claus` and `assumable` to record the content of the knowledge base, and the assumable facts, respectively).

```
dprove(true,D).
dprove(oand(A,B),D) :-
    dprove(A,DA), dprove(B,DB), append(DA,DB,D).
dprove(G,[G]) :- assumable(G).
dprove(G,D) :-
    claus(G,B), dprove(B,D).

append([],L,L).
append([H|T],L,[H|R]) :- append(T,L,R).
```

True or false: Prolog answers the query `dprove(false,C)` by instantiating `C` to a minimal conflict. Justify your answer.

4. (a) How does STRIPS draw the distinction between *primitive* and *derived* relations?

(b) How does STRIPS characterize an action? Explain with an example.

(c) Can we always avoid mentioning derived relations in the STRIPS pre-condition list of an action? Justify your answer.

(d) Consider the action *do-anything-you-can* in which an agent does anything it can. Can STRIPS characterize this action? Can the *situation calculus*? Justify your answer.

5. (a) Let us agree to encode a fact `p` as the list `[p]` and a rule `h:-b1,...,bn` as the list `[h,b1,...,bn]` so that a knowledge base of facts and rules can be encoded as a list of lists. For example, under this encoding, the knowledge base

```
a.
b.
c:-a,b.
```

becomes the list `[[a],[b],[c,a,b]]`.

(i) Define a predicate `lc(G,KB)` that checks that the fact `G` is a logical consequence of the knowledge base encoded by the list `KB`. For instance, Prolog should answer `yes` to the query

```
| ?- lc(c, [[a],[b],[c,a,b]]).
```

You may assume that no variables appear in the knowledge base encoded by `KB`.

(ii) Does your predicate work 'top-down' or 'bottom-up'? Explain the difference at stake here.

(iii) What complication arises if the knowledge base encoded by `KB` contains variables. Outline (briefly) how your definition of `lc(G,KB)` would have to be modified to handle variables.

(b) Consider the knowledge base

```
nat(zero).
nat(succ(X)) :- nat(X).
```

Describe 'top-down' and 'bottom-up' approaches to the query

```
| ?- nat(succ(succ(zero))).
```

If one or the other is not feasible, explain why.

6. (a) Explain the difference between Lazy and Eager approaches to Machine Learning giving brief examples of both.

(b) Nearest Neighbour classification is perhaps the simplest machine learning technique. Explain how it works.

(c) Nearest Neighbour classification is commonly implemented as k-Nearest Neighbour. Explain what the k refers to and explain its role in the algorithm you described in (b) above.

Section B

7. Explain in one sentence what distinguishes a 'stable' sort from an 'unstable' one. Write a Prolog program to sort a list of terms into order. The most frequently occurring words should be placed at the start of the output list, with the least frequently occurring words at the end of the list. For example, the list
[the,quick,brown,fox,jumps,over,the,lazy,but,not,quick,dog] sorts to, for example,
[quick,the,brown,but,dog,fox,jumps,lazy,not,over]

8. Peter van Roy has stated: Prolog = imperative language + unification + backtracking. What has this got to do with Prolog as a *logic programming language?*
Discuss this and support your arguments with technical details (and examples, if appropriate).