UNIVERSITY OF LONDON

IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 1997

BEng Honours Degree in Computing Part III
BSc Honours Degree in Mathematics and Computer Science Part III
MSci Honours Degree in Mathematics and Computer Science Part III
MSc Degree in Advanced Computing
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the
Diploma of Membership of Imperial College
Associateship of the Royal College of Science
Associateship of the City and Guilds of London Institute

PAPER 3.29

LOGIC PROGRAMMING - FOUNDATIONS

Thursday, April 24th 1997, 10.00 - 12.00

Answer THREE questions

For admin. only: paper contains 4 questions

1 For any set **S** of FOL (first-order logic) sentences, let **C**(**S**) denote the set of clauses obtained by converting **S** in the standard way to clausal form.

Let **A** be some set of FOL sentences and let T be some FOL sentence.

a Express in terms of **C**, **A** and T the set of clauses from which the derivation by resolution of the empty clause $\square$ is sufficient to establish **A** $\models$ T.

b Explain carefully why deriving $\square$ does indeed establish **A** $\models$ T.

c For the particular case

$$\mathbf{A} = \{\ (\forall XY)(\text{path}(X, Y)\ \text{ iff }\ (\text{go}(Y) \text{ if } \text{go}(X)))\ \}$$
$$T = (\forall XYZ)(\text{path}(X, Z)\ \text{ if }\ \text{path}(X, Y), \text{path}(Y, Z))$$

use the method stated in part a to establish **A** $\models$ T.
Note —you do **not** need to show the details of converting to clausal form.

The three parts carry, respectively, 10%, 20% and 70% of the marks.

2a The *immediate consequence function* $T_\mathbf{P}$ for a definite program **P** is defined by

$$T_\mathbf{P}(\mathbf{I}) = \{\ q \mid (q \text{ :- body}) \in \mathbf{G}(\mathbf{P}),\ \text{body} \subseteq \mathbf{I}\ \}$$

Explain what the terms **I** and **G**(**P**) denote here.

b Prove both the following:

i) $(\forall \mathbf{IJ})\ (\mathbf{I} \subseteq \mathbf{J} \rightarrow T_\mathbf{P}(\mathbf{I}) \subseteq T_\mathbf{P}(\mathbf{J}))$

ii) $(\forall \mathbf{I})\ (T_\mathbf{P}(\mathbf{I}) \subseteq \mathbf{I} \rightarrow \mathbf{I}$ is a model of $\mathbf{G}(\mathbf{P}))$

Note— in both cases use the $T_\mathbf{P}$ definition, basic properties of $\subseteq$ and consideration of how a clause in **G**(**P**) is made true or false by an interpretation.

c This program **P** is defined over the Herbrand domain **H**={chris, bob, logic}:

```
likes(chris, bob).
likes(X, Y) :- teaches(X, Y).
likes(X, Y) :- likes(Z, Y), teaches(X, Z).

teaches(bob, chris).
teaches(chris, logic).
```

Construct the least fixpoint $T_\mathbf{P}\uparrow\omega$, showing the iterates obtained.
Note —you may abbreviate the program's symbols if you wish.

What is the logical significance of the atoms contained in $T_\mathbf{P}\uparrow\omega$?

The three parts carry, respectively, 10%, 60% and 30% of the marks.

3a What is meant by the terms *computation rule* and *search rule* ?

b Explain what is meant by a *fair* computation rule and a *fair* search rule. Illustrate the benefit of each of these by a simple concrete example.

c Draw the SLD-tree obtained when Prolog evaluates the query

Q: ? p([a, E, b, F], [])

using the palindrome-testing program

```
P: | p(X, X).
   | p([U | X], X).
   | p([U | X], Z) :- p(X, [U | Z]).
```

Explain what is meant by an ***answer substitution***, and state the particular one computed in this example.

d A query can be represented by a list Query of calls treated as ordinary terms. Thus, **Q** in part c can be represented by just the unit list [p([a, E, b, F], [])]. A record of how the query is solved can be represented by a list Calls of the calls successively selected (by Prolog) for evaluation. For instance, the record of how **Q** is solved takes the form Calls = [p([a, E, b, F], []), ...].

Write a Prolog program defining the relation trace(Query, Trace) which holds when Query is solved (by Prolog) with answer substitution θ and Trace=Callsθ, where Calls is the record as defined above.

State the value of Trace obtained by solving the query ?trace(**Q**, Trace) using your "trace" program accompanied by the clauses of **P** given in part c.

Hint — this is a simple program comprising just one base case and one recursion jointly defining the "trace" predicate. Make use of Prolog's built-in predicate clause(H, B) which, given H, returns the body B of a program clause matching H :- B.

The four parts carry, respectively, 10%, 20% , 30% and 40% of the marks.

Turn over ...

4a i) Construct the *completed definition* of the single clause B(1) :- not C(X) and comment upon the steps involved.

ii) A completion **Comp(P)** comprises, in general, both the completed definitions of the relations mentioned in the program **P** and a set **E** of additional axioms. What computational mechanism is modelled by **E** and why is this significant?

b i) Construct **Comp(P1)** for this program **P1** with Herbrand domain {1, 2}:

```
A :- not B(1).
A :- C(1).
B(1) :- not C(X).
C(2).
```

ii) Sketch the evaluation of the query ?A under an *unsafe* computation rule.

iii) Determine whether or not **Comp(P1)** correctly predicts the outcome, and comment upon the result.

iv) Sketch the evaluation of the same query using (safe) SLDNF.

c In respect of any definite program **P**, SLDNF has this completeness property:

for any atom q in the Herbrand Base,
if **Comp(P)** $\models \neg q$ then ?not q succeeds from **P**

Construct a program **P2** in which the only non-definite clause is

q :- n(a), p(a), not n(a).

sufficient to show that SLDNF is not similarly complete for normal programs. In what way might SLDNF be modified such as to extend the class of normal programs satisfying the completeness property? Show that the modification would bring your program **P2** into this class.

The three parts carry, respectively, 20%, 60% and 20% of the marks.

End of Paper