UNIVERSITY OF LONDON

IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 1999

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

Monday, April 26th 1999, 2.00 – 4.00

Answer THREE questions

For admin. only:
paper contains 4 questions

1a Describe briefly but carefully the nature of an inference step in general resolution and then explain how SLD-resolution is a special case of general resolution.

b Given a program consisting of the two facts

```
p(1)
q(2)
```

explain the circumstances in which it could give this SLD-tree:

```
? p(X), q(1)
|
■
```

c For a certain definite program and a query ?q(X), every possible SLD-refutation binds X to some ground term. Moreover, one of the possible SLD-trees contains two refutations whilst another one contains four; none contains fewer than two. What are the maximum and minimum numbers of distinct solutions that this query can have?

d A certain SLD-tree contains this step:

```
:
? p(X), q(X, a)
|
? p(a), q(Z, a)
:
```

Write down two program clauses—having no common instance—either of which might be used to construct this step, depending upon the computation rule used.

e The solution obtained from an SLD-refutation is formally known as the computed answer substitution, comprising the bindings of the initial query variables that result from the composition of the refutation's unifiers.

Illustrate how this applies in the case of the following query and program:

```
?append(X, X, [a, b, a, b])

append([], Z, Z)
append([U | X], Y, [U | Z])  if  append(X, Y, Z)
```

The five parts carry, respectively, 20%, 5%, 15%, 20% and 40% of the marks.

2 For the following indefinite program **P**, the Herbrand domain is taken to be **H** = {peter, tony, gordon, charles, loan}:

```
C1:  good(X)  v  naughty(X)
C2:  naughty(peter)  v  declares(peter, Y)

C3:  sacks(tony, peter)  v  good(peter)
C4:  sacks(gordon, charles)  if  sacks(tony, peter)

C5:  ¬ declares(peter, loan)
```

The characters appearing in this program bear no relation to any living persons.

a How many atoms are there in the Herbrand base **B**(**P**), and how many Herbrand interpretations does **P** have ? Justify your answers.

b Explaining your answers in each case, write down

i) any minimal Herbrand model of **P** that contains good(peter),
ii) any minimal Herbrand model of **P** that does not contain good(peter).

c **P** can be reformulated as the following normal program **Q**, intended for execution by SLD extended with the finite-failure rule:

```
good(X)  if  fail naughty(X)
naughty(peter)  if  fail declares(peter, Y)

sacks(tony, peter)  if  fail good(peter)
sacks(gordon, charles)  if  sacks(tony, peter)
```

i) Write down the completion **comp**(**Q**) and decide whether it implies good(tony).
Note—the equality theory need not be written down explicitly.

ii) Sketch the evaluations of the queries ?good(tony) and ?good(X), and explain the difference between the results obtained.

The three parts carry, respectively, 20%, 40% and 40% of the marks.

Turn over ...

3 The *immediate consequence function* T_P for a definite program **P** is defined by

$$T_P(I) = \{ q \mid (q \text{ if body}) \in G(P), \text{body} \subseteq I \}$$

Also, $T_P\uparrow\omega$ is the smallest set that includes $T_P^k(\phi)$ for all $k \geq 0$, and
$T_P\downarrow\omega$ is the smallest set that includes $T_P^k(B(P))$ for all $k \geq 0$.

a What do **I** and **G(P)** denote in the above definition of $T_P(I)$?
What is the logical and computational significance of $T_P\uparrow\omega$?

b Show that $T_P(I) \subseteq I$ holds in the case that **I** is a Herbrand model of **P**.

c Show by an example that, given two definite programs **P** and **Q**, the following does not necessarily hold:

$$T_P\uparrow\omega \cup T_Q\uparrow\omega = T_{P\cup Q}\uparrow\omega$$

d For a definite program **P** the finite-failure set **FF(P)** is defined as $B(P) - T_P\downarrow\omega$.
For the case where **P** comprises just the clause

```
p if p, q
```

and $B(P) = \{p, q\}$, determine **FF(P)** and explain why the result does not agree with Prolog's evaluation of the query ?p.

e Explain the significance of a case in which, for a definite program **P**,

$$T_P\uparrow\omega = T_P\downarrow\omega$$

f Show that the following holds for any definite programs **P** and **Q** having a common Herbrand Base:

$$T_P\downarrow\omega \cup T_Q\downarrow\omega \subseteq T_{P\cup Q}\downarrow\omega$$

The six parts carry, respectively, 20%, 20%, 10%, 25%, 10% and 15% of the marks.

4a The standard Prolog interpreter is said to have a *depth-first* search strategy.

i) Precisely what is meant by depth-first ?
ii) State one significant advantage of this strategy.

b Depth-first search is said to be *unfair*. Explain what this means, and give a simple example showing the disadvantage caused by this unfairness.

c Write in Prolog an interpreter **I** which simulates a *breadth-first* search strategy (which is necessarily fair) capable of finding all solutions to a query **Q** given as data an object-level definite program **P**. Its main predicate should be demo(Queries) in which Queries is a list of queries each represented by a list of atoms. For example, in order to simulate the breadth-first evaluation of

```
? p(X), q(X, Y)
```

the initial query posed to your interpreter would be

```
? demo([[p(X), q(X, Y)]])
```

You may make free use of any Prolog primitives. If you require other predicates performing basic list-processing operations you need not write clauses defining them *provided* you instead specify their intended meanings in English.

Besides giving the code, write a short explanation of how it is intended to work. Marks can be gained from this even if your coding runs into difficulties.

d Express formally, by a statement relating **Q**, **P**, **I** and demo, a condition for your interpreter **I** being a correct evaluator.

The four parts carry, respectively, 25%, 15%, 50% and 10% of the marks.

End of Paper