UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2003

MEng Honours Degrees in Computing Part IV
MSc 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 Associateship of the City and Guilds of London Institute

PAPER C491

KNOWLEDGE REPRESENTATION

Thursday 8 May 2003, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators not required

1 In this question, use $\mathrm{Th}(X)$ to denote the classical truth-functional consequences of a set of formulas X. $X \vdash \alpha$ is shorthand for $\alpha \in \mathrm{Th}(X)$.

a Define the extension of a Reiter default theory (W, D) and give its inductive characterisation.

b Consider knowledge bases constructed as follows. Let D be any set of Reiter default rules. A knowledge base is a set W of first-order formulas and has content $\mathrm{Cn}_D(W)$ where Cn_D is the consequence operator corresponding to the 'sceptical' or 'cautious' consequences of W under the default rules D: that is, α is in $\mathrm{Cn}_D(W)$ iff α is in the intersection of all the extensions of the default theory (W, D).

Construct a simple example (a set D of default rules, a knowledge base, and a query) to demonstrate that, in general:

i) There are three possible answers to a closed query on a knowledge base of this form. (A closed query is a query containing no free variables.)

ii) Cn_D is non-monotonic.

(An example with a single suitably chosen default rule will be enough. The 'knowledge base' can be very small.)

c Let A and B be sets of first-order formulas, and let Cn_D be the consequence operator defined in part (b). Show each of the following:

i) $A \subseteq \mathrm{Cn}_D(A)$

ii) When $A \subseteq B$, if E is an extension of (A, D) then E is an extension of (B, D).

iii) From part (ii), it follows that Cn_D satisfies the following property *'cut'*:

$$\text{If } A \subseteq B \subseteq \mathrm{Cn}_D(A) \text{ then } \mathrm{Cn}_D(B) \subseteq \mathrm{Cn}_D(A)$$

The three parts carry, respectively, 25%, 25%, 50% of the marks.

2a Define the *answer set* (stable model) semantics of a *normal* logic program.

b Show that

i) $\{p \leftarrow \mathsf{not}\, p\}$ has no stable models;

ii) $\{q \leftarrow \mathsf{not}\, r \; ; \quad r \leftarrow \mathsf{not}\, q\}$ has two stable models;

iii) $\{p \leftarrow \mathsf{not}\, r\}$ has one stable model.

c What is a *stable expansion* in autoepistemic logic?

d Consider the following logic program P:

$$s \leftarrow r, \mathsf{not}\, s, \mathsf{not}\, t$$
$$r \leftarrow p, \mathsf{not}\, q$$
$$q \leftarrow p, \mathsf{not}\, r$$
$$p \leftarrow \mathsf{not}\, t$$

Write down the standard translation of P to an equivalent autoepistemic logic theory. Explain (without proof) in what sense the normal logic program is equivalent to the autoepistemic logic theory.

e Use *splitting sets* to compute the answer sets (stable models) of the logic program P of part (d).

The five parts carry, respectively, 10%, 30%, 10%, 15%, 35% of the marks.

3a Consider the following simplified formulation of the Yale Shooting Problem.

$$s_0 \xrightarrow{\textsf{load}} s_1 \xrightarrow{\textsf{wait}} s_2 \xrightarrow{\textsf{shoot}} s_3$$

To simplify the question, situations are named here by constants s_o, s_1, s_2, s_3 rather than by means of a *result* function. The effect axioms and the initial state can then be expressed as follows:

$$\begin{aligned}
&\mathit{holds}(\mathsf{loaded}, s_1)\\
&\neg\mathit{holds}(\mathsf{alive}, s_3) \leftarrow \mathit{holds}(\mathsf{loaded}, s_2)\\
&\mathit{holds}(\mathsf{alive}, s_0)
\end{aligned}$$

The frame axioms representing the 'law of inertia' take the following form:

$$\mathit{holds}(f, s_{i+1}) \leftarrow \mathit{holds}(f, s_i), \neg ab(f, s_i, s_{i+1}) \tag{1}$$

$$\neg\mathit{holds}(f, s_{i+1}) \leftarrow \neg\mathit{holds}(f, s_i), \neg ab(f, s_i, s_{i+1}) \tag{2}$$

where f is alive or loaded and i is $0, 1, 2$.

Explain how *circumscription* can be used with the schemas (1) and (2) to give default persistence (or 'inertia') of fluents. What is circumscribed, and how does the circumscription determine what holds in the situations s_0, s_1, s_2, s_3?
Give the account in terms of models ('preferential entailment'). There is no need to formulate any second-order circumscription axioms. It is not necessary to construct the entire model(s) in every detail.

b Explain how default persistence can be obtained by using negation-by-failure with a modified form of axiom schemas (1) and (2) instead of circumscription. Structure your answer as follows.

i) First, re-formulate the theory of part (a) as an *extended* logic program. You will need to add a suitable definition for the *ab* predicate, either as a set of assertions or, better, as clauses representing the contrapositive forms of schemas (1) and (2).

ii) Translate the extended logic program into an equivalent *normal* logic program.

iii) In what sense are the extended and normal logic programs equivalent?
How does one formulate queries to determine whether alive holds in situation s_3?

(It is *not* necessary to prove any equivalence between the logic programs of this part and the circumscribed theory of part (a).)

c i) Minimal supported models for *stratified* programs can be obtained by using an iterated fixpoint construction. Describe this construction, and define what is meant by *minimal* and *supported* in this context.

ii) Discuss whether it is possible to use the iterated fixed point construction with your normal logic program from part (b).

The three parts carry, respectively, 40%, 35%, 25% of the marks.

4 Consider a domain in which there are agents $a, b, c, \ldots$ and locations $l_1, l_2, \ldots$ for some fixed (finite) number of each. Let the multi-valued fluents $\mathsf{loc}(x)$ and $\mathsf{car}(x)$ represent the location of agent x and the location of agent x's car, respectively: $\mathsf{loc}(x){=}l_1$ represents that agent x is at location l_1, and $\mathsf{car}(x){=}l_2$ that x's car is at location l_2. Consider two types of action: $\mathsf{walk}(x){=}l$ represents that agent x walks to location l, and $\mathsf{drive}(x){=}l$ that agent x drives its car to location l. Walking affects only the location of the agent, and driving affects both the location of the agent and the location of its car.

a Show how this example can be formulated as an action description in the language $\mathcal{C}/\mathcal{C}+$.

Several different agents might be walking or driving simultaneously. Comment on how this is allowed for in your formulation.

Ensure your formulation caters for the following constraints:

(i) an agent cannot drive its car if they are not both at the same location,

(ii) an agent cannot walk or drive to two different locations simultaneously,

(iii) an agent cannot walk and drive at the same time.

b Describe how the 'Causal Calculator' CCALC is used to perform computations on action descriptions in $\mathcal{C}/\mathcal{C}+$. Structure your answer as follows:

i) How is an action description in $\mathcal{C}/\mathcal{C}+$ translated to a 'causal theory'? Use the action description of part (a) to illustrate your answer.

ii) What is the *literal completion* of a definite causal theory? What is the significance of this completion in the case of $\mathcal{C}/\mathcal{C}+$ action descriptions? There is no need to define the term 'definite'.

iii) What method does the 'Causal Calculator' CCALC use to compute answers to queries?

Concentrate on explaining the main ideas, using the action description of part (a) as a source of examples. There is no need to show the computation of the example in every detail.

c Show how the walk-drive example can be formulated in the event calculus.

Explain how a narrative of walk and drive events is represented, and include the event calculus axioms for *holds_at* in your answer.

The three parts carry, respectively, 35%, 35%, 30% of the marks.