UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2003

BEng Honours Degree in Computing Part III
MEng Honours Degree in Information Systems Engineering Part IV
BSc Honours Degree in Mathematics and Computer Science Part III
MSci Honours Degree in Mathematics and Computer Science Part III
MSc in Advanced Computing
PhD
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
This paper is also taken for the relevant examinations for the Associateship of the Royal College of Science

PAPER C394=I4.40

ADVANCES IN ARTIFICIAL INTELLIGENCE

Wednesday 30 April 2003, 10:00
Duration: 120 minutes

Answer THREE questions

Paper contains 4 questions
Calculators required

1a Briefly explain the differences between *conditional planning* and *situated planning.*

b Consider the following robot path-finding problem: a robot is in a building and needs to find a path from one room (where it is located) to another. The layout of the building is described by the predicate

connected(Room, Door, OtherRoom)

which holds when *Room* is connected to *OtherRoom* via *Door*. The predicate *connected* is symmetric in the first and third arguments, that is, doors open both ways.

The robot can perform two actions

open(Door)

move(Room, OtherRoom)

The action *open(Door)* is possible if *Door* is closed and the robot is in a room directly connected via *Door* to another room. The outcome of the action is for *Door* to become open.

The action *move(Room, OtherRoom)* is possible if the robot is in *Room* and *Room* is directly connected via some open door to *OtherRoom*. The outcome of the action is for the robot to change location from *Room* to *OtherRoom.*

i. Represent the actions *open(Door), move(Room, OtherRoom)* as operators in STRIPS, specifying appropriate pre- and post-conditions. Use the predicate *in(Room)* to represent the fact that the robot is in *Room.*

ii. Given the initial state with

in(a) and

connected(a,d1,b) and connected(b,d2,c) and connected(b,d3,d) and

closed(d1) and open(d2) and open(d3)

and the goal state *in(c),* show how POP would compute the non-minimal plan (ordered left-to-right)

open(d1), move(a,b), move(b,d), move(d,b), move(b,c).

Show how the plan is constructed step by step, identifying preconditions and effects of actions, introduction of causal links, ordering constraints and threats, if any.

c Given the initial state and goal in part bii, apply GRAPHPLAN to compute a minimal plan. You may ignore the static properties (namely those properties that cannot be affected by actions). Return the computed graph explicitly, indicating add-links, delete-links and all actions (including the no-ops). Indicate explicitly the computed plan.

Parts a, bi, bii, and c carry, respectively, 20%, 10%, 35%, 35% of the marks.

2a Give the formal definition of *explanation* in Theorist. Briefly compare this definition with that of explanation in abductive logic programming, identifying similarities and differences.

b Consider the following abductive logic program:

T: *p(X) if q(X) and a(X)*

q(3)

r(5)

H: (all ground atoms in the predicates) *a, b*

IC: *if a(X) and b(X) then false*

if a(X) and r(X) and t(X) then false

i. What is the appropriate abductive proof procedure for computing explanations to observations, given <T,H,IC> above?

ii. State formally the property of completeness for this procedure. Does this property hold?

iii. For each of the observations O1 and O2 below, give a derivation, with respect to the abductive proof procedure in part i, adopting a left-most selection rule. For each derivation, give explicitly the computed set of hypotheses and the computed answer substitution, if any, or say explicitly that none exist, otherwise.

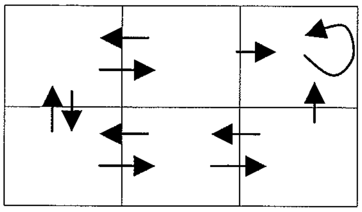
O1: *p(X)*

O2: *p(X) and b(X)*

c Give an example of a non-basic explanation in Theorist. Briefly justify the following statement: all explanations are basic in (any form of) abductive logic programming.

The three parts carry, respectively, 30%, 50%, 20% of the marks.

3a Consider a robot moving in the following grid:



The arrows indicate the possible actions (moves) that the robot can perform in each cell, and the (deterministic) effects of the actions. Below, given a cell (state) s in the grid and an action a possible in that cell, $\delta(s,a)$ indicates the cell in which the robot will be located after performing a in s.

For each cell s other than the right-most top-most cell $s_{3,2}$, the reward is given by $R(s)=0$. For the cell $s_{3,2}$, the reward is given by $R(s_{3,2})=120$.

i. Using the equation $Q(s,a)=R(s) + \gamma\, max_{a'}\, Q(\delta(s,a),a')$, compute the Q-values for each possible action in each cell of the grid. Use the discount factor $\gamma=0.5$.

ii. Using the Q-values computed in part i, give one optimal policy for the robot.

b Draw a suitable Bayesian network topology for the following set of (binary) random variables: *FrozenBattery, IcyWeather, CarDoesNotStart, EmptyTank.*

Suggest reasonable prior and conditional probabilities for the nodes of the network.

Exemplify *diagnostic*, *predictive* and *inter-causal* reasoning for the given network. One example for each kind of reasoning will suffice.

c Compare a planning agent using a conventional planner (such as POP) to a decision-theoretic agent.

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

4a Consider the problem of learning the concept *daughter* in inductive logic programming, given the description set D, classification set C, and background knowledge B (all given as sets of definite clauses):

D = {*daughter(julia,emily)*}

C = {*female(julia), mother(emily,julia)* }

B = { *∀X,Y [parent(X,Y) if mother(X,Y)],*

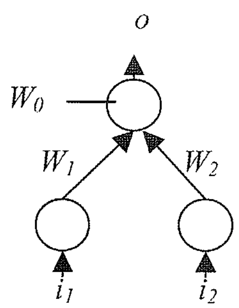
∀X,Y [parent(X,Y) if father(X,Y)] }

i. Apply inverse resolution to learn the clause

L = *∀X,Y [daughter(X,Y) if female(X) and parent(Y,X)]*

ii. Suppose now that B is empty, and C and D are as above. Apply inverse resolution to learn the clause L and a clause defining *parent.*

b Consider the following perceptron



where $o(i_1,i_2) = h(W_0+W_1i_1 + W_2i_2)$ and $h(x) = 1$ if $x > 0$ and 0 otherwise.

Using the equation

$\Delta W = ?\ (t - o(i_1,i_2))\ \boldsymbol{i}$, where t is the target and $?$ is the learning rate,

train the perceptron to learn $o(i_1,i_2) = (i_1 \text{ and } i_2)$, where $i_j=1$ stands for i_j is *true*, and $i_j=0$ stands for i_j is *false*, $j=1,2$.

In the training process:
- Use $? = 1$.
- Initialise the weights as follows: $W_0=0$, $W_1=2$, $W_2=1$.
- Consider the inputs i_1, i_2 in the following order: (*false,false*), (*false,true*), (*true,false*), (*true, true*).

c Briefly discuss the problem of assessing the performance of a learning system.

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