FACOLTÀ DI INGEGNERIA DELL'INFORMAZIONE

Artificial Intelligence 2009–10

Test 2 – 26th February 2010

RULES: You have 2 hours to complete the test
You can use no texts, written notes, computers, calculators, mobile phones
Please write your name and student ID on all sheets
Please answer Question 3 on a separate sheet to speed up the marking process

1. State Space (10 points)

1.1 Explain the difference between *tree search* and *graph search*, clarifying: (i) why and when we need graph search; (ii) what improvements to the tree search algorithm are required to implement graph search (you are not asked to describe the whole algorithm, only the relevant modifications).

1.2 Suggest a state space representation for the problem stated below: define a data structure for representing the states, specify the allowable actions, represent the initial state and the goal, and describe a solution. Finally say what search strategy you would choose.

> There are a lake (of unlimited size), two cans (of 4 and 7 litres, respectively), and a container (of 12 litres). You are allowed to fill a can with water from the lake or from the container, and to pour the content of a can into the container or into the other can or even back into the lake. The goal is to fill the container with exactly 6 litres of water from the river. The cans and the container are initially empty.

2. CSP (10 points)

2.1 Define the concept of a Constraint Satisfaction Problem (CSP); in particular explain clearly the meaning of the following terms: *variable*, *domain*, *value*, *assignment*, *constraint*, *consistent assignment*, *solution*.

2.2 Define a map-colouring problem on a map of your choice (with at least 5 countries) as a CSP, specifying all variables, domains, and constraints. Then choose a search strategy for the solution of the problem and justify your choice. Specify and concisely define the domain-independent heuristics that you would adopt to speed up the search.

3. Logic (12 points)

3.1 Formalize the following sentences into first order logic formulae:

a) if Serena Williams teams with her sister Venus, then nobody can beat any of them;

b) Serena is stronger than Venus;

c) if Serena doesn't team with Venus, then there are tennis players who can beat her (Serena);

d) if nobody can beat Venus, then Serena is teaming with her.

3.2 Transform the first order logic formulae into clauses.

3.3 Try to prove by refutation with the resolution technique that sentence d is a logical consequence of sentences a - c (watch out for hidden assumptions on 'stronger'). If the procedure fails, explain why.

SOLUTION OF 1.2

Representation of states: <Can1,Can2,Cont>

$0 \le Can1 \le 4$

$0 \le Can2 \le 7$

$0 \le Cont \le 12$

A set of actions (sufficient to solve the problem):

FillCan1:	<0,Can2,Cont>	→	<4,Can2,Cont>
FillCan2:	<Can1,0,Cont>	→	<Can1,7,Cont>
EmptyCan1:	<Can1,Can2,Cont>	→	<0,Can2,Cont>
EmptyCan2:	<Can1,Can2,Cont>	→	<Can1,0,Cont>
Can1ToCont:	<Can1,Can2,Cont> ∧ Cont+Can1 ≤ 12	→	<0,Can2,Cont+Can1>
Can2ToCont:	<Can1,Can2,Cont> ∧ Cont+Can2 ≤ 12	→	<Can1,0,Cont+Can2>
ContToCan1:	<0,Can2,Cont> ∧ Cont ≥ 4	→	<4,Can2,Cont–4>
ContToCan2:	<Can1,0,Cont> ∧ Cont ≥ 7	→	<Can1,7,Cont–7>

Initial state: <0,0,0>

Goal states: <∗,∗,6>

A solution:

<0,0,0>	FillCan2
<0,7,0>	Can2ToCont
<0,0,7>	ContToCan1
<4,0,3>	FillCan2
<4,7,3>	Can2ToCont
<4,0,10>	EmptyCan1
<0,0,10>	ContToCan1
<4,0,6>	(goal)

SOLUTION OF 3

3.1

a) Team(S,V) => ¬ ∃ x (Beat(x,S) ∨ Beat(x,V))

b) Stronger(S,V)

c) ¬ Team(S,V) => ∃x (Beat(x,S))

d) ¬ ∃x Beat(x,V) => Team(S,V)

3.2

a1) ¬ Team(S,V) ∨ ¬ Beat(x,S)

a2) ¬ Team(S,V) ∨ ¬ Beat(x,V)

b) Stronger(S,V)

c) Team(S,V) ∨ Beat(K,S)

d) Beat(H,V) ∨ Team(S,V)

3.3

By negating thesis d we obtain:

¬d) ¬ ∃x Beat(x,V) ∧ ¬Team(S,V) from which we obtain the following clauses

¬d1) ¬ Beat(x,V)

¬d2) ¬ Team(S,V)

If we define Stronger(a,b) as ∀x (Beat(x,a) => Beat(x,b)) we have:

b*) ¬ Beat(x,S) ∨ Beat(x,V)

and we proceed with the resolution as follows:

b* + c = 1: Team(S,V) ∨ Beat(K,V)
1 + ¬d1 = 2: Team(S,V)
2 + ¬d2 = {}.

If we define Stronger(a,b) as Beat(a,b) we have:

b°) Beat(S,V)
we obtain the empty clause immediately from b° + ¬d1.