The University of Nottingham

SCHOOL OF COMPUTER SCIENCE

A LEVEL 2 MODULE, AUTUMN SEMESTER 2010-2011

PLANNING AND SEARCH

Time allowed TWO hours

Candidates may complete the front cover of their answer book and sign their desk card but must NOT write anything else until the start of the examination period is announced

Answer FOUR out of SIX questions

Only silent, self contained calculators with a Single-Line Display are permitted in this examination.

Dictionaries are not allowed with one exception. Those whose first language is not English may use a standard translation dictionary to translate between that language and English provided that neither language is the subject of this examination. Subject specific translation dictionaries are not permitted.

No electronic devices capable of storing and retrieving text, including electronic dictionaries, may be used.

DO NOT turn your examination paper over until instructed to do so

1. This question is on genetic algorithms.

   (a) Define the terms chromosome, fitness function, crossover and mutation as used in genetic algorithms. Explain how genetic algorithms work, in English or in pseudocode. (10 marks)

   (b) A genetic algorithm is to be used to evolve a binary string of length n containing only 1s. The initial population is a randomly generated set of binary strings of length n.

      i. Give a suitable fitness function for this problem. (2 marks)

      ii. Will the offspring of parents with a high fitness value generally also have a high fitness value, given your fitness function? Explain your answer. (2 marks)

   (c) Suppose the problem is to evolve a binary string of length n which is symmetric. If the string positions are numbered from 0, then a symmetric string will have a 1 in position i if and only if there is a 1 in position $(n-1)-i$. For example, 001100 is symmetric since it has a 1 at index 2 and a 1 at index $(6-1)-2=3$. Similarly, 110011 is symmetric, and 011011 is not. The initial population is a randomly generated set of binary strings of length n, where n is an even number.

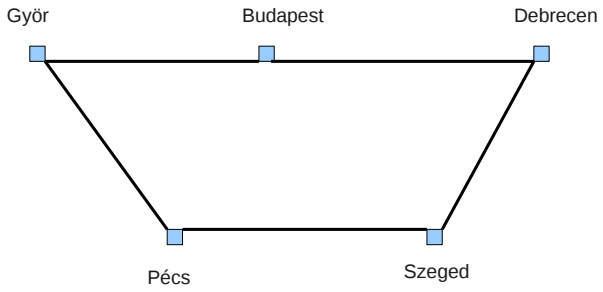
      i. Give a suitable fitness function for this problem. (3 marks)

      ii. Will the offspring of parents with a high fitness value generally also have a high fitness value, given your fitness function? Explain your answer. (3 marks)

   (d) If the population size in a genetic algorithm is restricted to 1, what search algorithm does it correspond to? Explain your answer. (5 marks)

2. This question is on search with non-deterministic actions.

   Consider the following search problem. The set of states corresponds to a set of cities on the map of Hungary below (the state Budapest corresponds to being in Budapest, etc.). Actions correspond to following an edge (driving along the road) from one city to another. All actions apart from two are deterministic and have the expected result, namely driving from X to Y results in being in Y. The only exceptions are actions which involve driving out of Győr: the action of driving from Győr to Budapest has two possible outcomes, one being in Budapest and another being in Pécs. Similarly, the action of driving from Győr to Pécs also has two possible outcomes, one being in Budapest and another being in Pécs. The initial state is being in Győr and the goal is to reach Szeged.



   (a) For each state, give a list of possible actions. (2 marks)

   (b) For each state and action, define the function $Result(state, action)$. (2 marks)

   (c) Draw the and-or search tree for this search problem. To reduce the size of the tree, don't expand the nodes for the states you have already visited. (7 marks)

   (d) Explain how an and-or search algorithm works, in English or pseudocode. (10 marks)

   (e) Draw a subtree which is a solution to the problem and state the corresponding plan. (4 marks)

3. This question is on SATPlan algorithm.

   Consider the following planning problem:

   Initial state: $Have(Cake)$

   Goal: $Have(Cake) \wedge Eaten(Cake)$

   $Eat(Cake)$:
   PRECOND: $Have(Cake)$
   EFFECT: $\neg Have(Cake) \wedge Eaten(Cake)$

   $Bake(Cake)$:
   PRECOND: $\neg Have(Cake)$
   EFFECT: $Have(Cake)$

   (a) Give a description of this planning problem in terms of propositional formulas, suitable as an input to the SATPlan algorithm when searching for a plan consisting of 2 actions. You need to state:
      i. That the initial state description holds at time 0. (2 marks)
      ii. That the goal description holds at time 2. (1 marks)
      iii. Successor state axioms for times 1 and 2 for fluents $Have(Cake)$ and $Eaten(Cake)$. (8 marks)
      iv. Precondition axioms for times 0 and 1 for the two actions. (2 marks)
      v. Action exclusion axioms for times 0 and 1. (2 marks)

   (b) Give an assignment which satisfies all the formulas above, and show that the formulas are indeed true for this assignment. (8 marks)

   (c) Extract the solution from this assignment. (2 marks)

4. (a) Explain how regression planning works, in English or pseudocode. (10 marks)

   (b) Consider the following planning problem. An agent is at home and has no food. The goal is to be at home and have food. Actions available to the agent are:

   $Buy(Food)$:
   PRECOND: $At(x) \wedge Sells(x, Food)$
   EFFECT: $Have(Food)$

   $Go(x, y)$:
   PRECOND: $At(x)$
   EFFECT: $\neg At(x) \wedge At(y)$

   The initial state is $At(Home) \wedge Sells(S1, Food) \wedge Sells(S2, Food)$.

   The goal state is $At(Home) \wedge Have(Food)$.

   Solve this problem using regression planning; trace the algorithm, explaining each step. (10 marks)

   (c) How are a *classical search problem* and a *classical planning problem* formulated? Explain the difference between them. (5 marks)

5. This question is on partial order planning.

   (a) Explain what is a totally ordered plan and what is a partially ordered plan. (3 marks)

   (b) Explain how partial order planning works (in English or pseudocode). (10 marks)

   (c) Solve the following blocks world problem below using partial order planning; trace the search from the initial empty plan to a complete solution, explaining each step. (12 marks)

   The actions are:

   $Move(b, x, y)$:
   PRECOND: $On(b, x) \wedge Clear(b) \wedge Clear(y)$
   EFFECT: $On(b, y) \wedge Clear(x) \wedge \neg On(b, x) \wedge \neg Clear(y)$

   $Movetotable(b, x)$:
   PRECOND: $On(b, x) \wedge Clear(b)$
   EFFECT: $On(b, Table) \wedge Clear(x) \ \wedge \neg On(b, x)$

   The initial state is $On(B, A) \wedge On(A, Table) \wedge On(C, Table) \wedge Clear(B) \wedge Clear(C)$. The goal is $On(C, B) \wedge On(B, Table)$.

6. This question is on HTN planning.

   Consider the following (incomplete) planning domain description. A fence consists of 3 panels, numbered 1 to 3. The agent can be positioned in front of any of the panels. The agent being in front of panel i is represented by $At(i)$. The agent can move itself one panel to the left and one panel to the right. When it is in front of an unpainted panel, it can paint it. The planning domain is described by the following hierarchical task network (where the refinements for $Go(x, 1)$ action are not given):

   $Left(x)$:
   PRECOND: $At(x) \wedge x > 1$
   EFFECT: $\neg At(x) \wedge At(x - 1)$

   $Right(x)$:
   PRECOND: $At(x) \wedge x < 3$
   EFFECT: $\neg At(x) \wedge At(x + 1)$

   $Paint(x)$:
   PRECOND: $At(x) \wedge \neg Painted(x)$
   EFFECT: $Painted(x)$

   $Refinement(PaintAll,$
   PRECOND: $At(x)$
   STEPS: $[Go(x, 1), Paint(1), Right(1), Paint(2), Right(2), Paint(2), Right(2), Paint(3)])$

   $Refinement(Go(x, 1),$
   ??? ???)

   (a) Add suitable refinements for the $Go(x, 1)$ high-level action. (5 marks)

   (b) Explain how hierarchical search works, in English or pseudocode. (10 marks)

   (c) Trace hierarchical search on the following example: the planning domain is as given above plus your refinements for the $Go(x, 1)$ action. In the initial state the agent is in front of panel 2 and the fence is not painted, and in the goal state it is painted. For simplicity, assume that the only refinement of the top-level action *Act* is *PaintAll*. If you could not do part (a), just assume that $Go(x, 1)$ is a primitive action which makes $At(1)$ true. (5 marks)

   (d) Compare the complexity of solving an arbitrary planning problem using forward state-space search with primitive actions, and solving it using hierarchical planning. (5 marks)