NATIONAL UNIVERSITY OF SINGAPORE

SCHOOL OF COMPUTING

FINAL EXAM FOR
Semester 2 AY2012/13

CS3243: INTRODUCTION TO ARTIFICIAL INTELLIGENCE

May 2, 2013 Time Allowed: 2 Hours

INSTRUCTIONS TO CANDIDATES

1. This examination paper contains **FIVE (5)** parts and comprises **TWELVE (12)** printed pages, including this page.
2. Answer **ALL** questions as indicated.
3. This is a **RESTRICTED OPEN BOOK** examination.
4. Please fill in your **Matriculation Number** below.

MATRICULATION NUMBER: ______________________

EXAMINER'S USE ONLY		
Part	Mark	Score
I	16	
II	14	
III	4	
IV	6	
V	10	
TOTAL	50	

In Part I, II, III, IV, and V, you will find a series of short essay questions. For each short essay question, give your answer in the reserved space in the script.

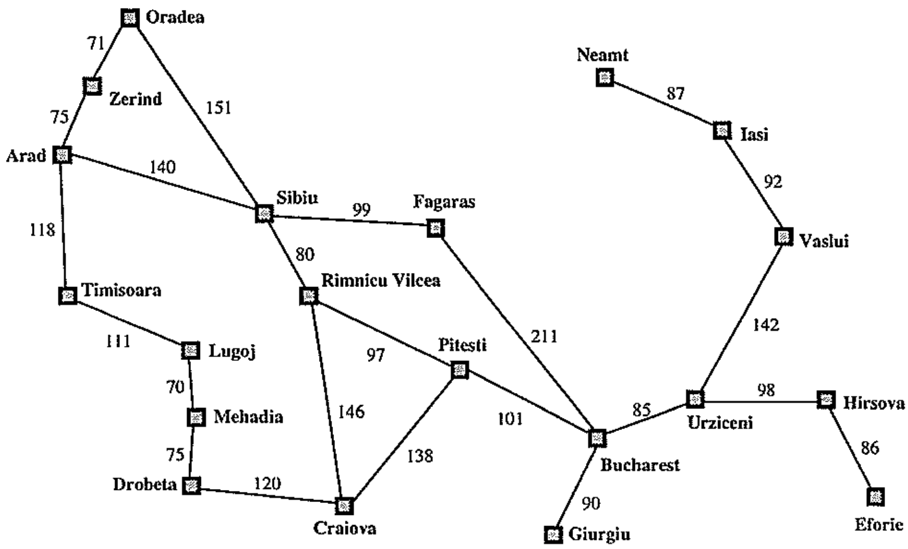
Part I
Uninformed and Informed Search

(16 points) Short essay questions. Answer in the space provided on the script.

Refer to the Figure 1 below. Apply the A* search algorithm using graph search to find a path from CRAIOVA to FAGARAS, using the evaluation function $f(n) = g(n) + h(n)$ where

$$h(n) = \max\left(h_{SLD}(\text{Fagaras}) - h_{SLD}(n), \frac{h_{SLD}(n) - h_{SLD}(\text{Fagaras})}{2} \right)$$

and $h_{SLD}(n)$ is the straight-line distance from any city n to Bucharest given in Figure 3.22 of AIMA 3rd edition (reproduced in Figure 1).



Values of h_{SLD} - straight-line distances to Bucharest

Arad	366
Bucharest	0
Craiova	160
Drobeta	242
Eforie	161
Fagaras	176
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	100
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Figure 1: Graph of Romania.

1. (8 points) Trace the A* search algorithm using GRAPH SEARCH by showing the nodes in the frontier at the end of each iteration of the outer loop. Pay very careful attention to the following instructions when presenting your solution:
   - Recall from page 93 of AIMA 3rd edition (specifically, last line of text) that the A* search algorithm is identical to uniform-cost search (reproduced from Figure 3.14 of AIMA 3rd edition in Figure 2 below) except that A* uses $g + h$ instead of g.
   - For each node n in the frontier, give the corresponding 3-tuple $(g(n), h(n), f(n))$.
   - At the end of each iteration of the outer loop, list the nodes in the frontier in nondecreasing order of f value.
   - AFTER the goal node is found (i.e., last iteration of the outer loop), you must also list the nodes in the frontier.

function UNIFORM-COST-SEARCH(*problem*) **returns** a solution, or failure

```
node ← a node with STATE = problem.INITIAL-STATE, PATH-COST = 0
frontier ← a priority queue ordered by PATH-COST, with node as the only element
explored ← an empty set
loop do
   if EMPTY?(frontier) then return failure
   node ← POP(frontier)  /* chooses the lowest-cost node in frontier */
   if problem.GOAL-TEST(node.STATE) then return SOLUTION(node)
   add node.STATE to explored
   for each action in problem.ACTIONS(node.STATE) do
      child ← CHILD-NODE(problem, node, action)
      if child.STATE is not in explored or frontier then
         frontier ← INSERT(child, frontier)
      else if child.STATE is in frontier with higher PATH-COST then
         replace that frontier node with child
```

Figure 2: Uniform-cost search algorithm.

Solution: The node (denoting the initial state) in the frontier before entering the outer loop is provided.

FRONTIER:

Craiova(0,16,16)
End of Iteration 1:
End of Iteration 2:
End of Iteration 3:
End of Iteration 4:
End of Iteration 5:
End of Iteration 6:
End of Iteration 7:
End of Iteration 8:

2. (1 point) Give the solution path from Craiova to Fagaras that is produced by the A* search algorithm.

Solution:

3. (3 points) Give the solution path from Craiova to Fagaras that is produced by the greedy best-first search algorithm. Does greedy best-first search incur lower, same, or higher solution path cost than A*? Does greedy best-first search incur lower, same, or higher search cost (i.e., number of EXPANDED nodes) than A*? You don't have to explain your answer.

Solution:

4. (3 points) Give the solution path from Craiova to Fagaras that is produced by the uniform-cost search algorithm given in Figure 3.14 of AIMA 3rd edition (reproduced in Figure 2). Does uniform-cost search incur lower, same, or higher solution path cost than A*? Does uniform-cost search incur lower, same, or higher search cost (i.e., number of EXPANDED nodes) than A*? You don't have to explain your answer.

Solution:

5. (1 point) Let $h_1(n) = h_{SLD}(n)$. Prove that $h_1(n)$ is NOT an admissible heuristic for A* tree search to find a path from Craiova to Fagaras.

Solution:

Part II
Adversarial Search

(14 points) Short essay questions. Answer in the space provided on the script.

1. (4 points) Consider the minimax search tree shown in the solution space below; the utility function values are specified with respect to the MAX player and indicated at all the leaf (terminal) nodes. Suppose we use alpha-beta pruning algorithm, given in Figure 5.7 of AIMA 3rd edition (reproduced in Figure 3), in the direction from left to right to prune the search tree. Mark (with an "X") all arcs that are pruned by alpha-beta pruning.

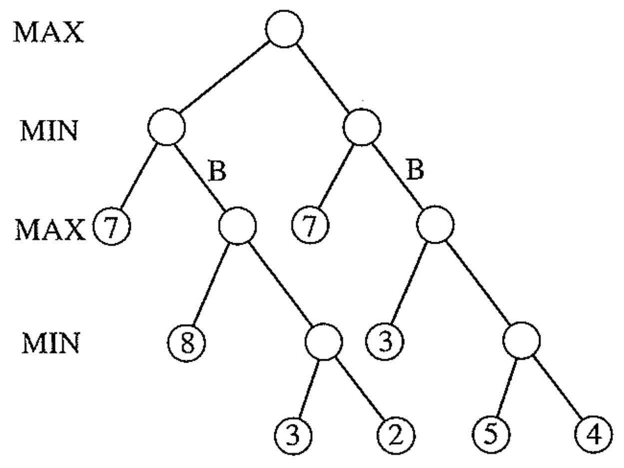
```
function ALPHA-BETA-SEARCH(state) returns an action
  v ← MAX-VALUE(state, −∞, +∞)
  return the action in ACTIONS(state) with value v
```

```
function MAX-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← −∞
  for each a in ACTIONS(state) do
    v ← MAX(v, MIN-VALUE(RESULT(s,a), α, β))
    if v ≥ β then return v
    α ← MAX(α, v)
  return v
```

```
function MIN-VALUE(state, α, β) returns a utility value
  if TERMINAL-TEST(state) then return UTILITY(state)
  v ← +∞
  for each a in ACTIONS(state) do
    v ← MIN(v, MAX-VALUE(RESULT(s,a) , α, β))
    if v ≤ α then return v
    β ← MIN(β, v)
  return v
```

Figure 3: Alpha-beta pruning algorithm.

Solution:



2. (1 point) What is the minimax value at the root node?

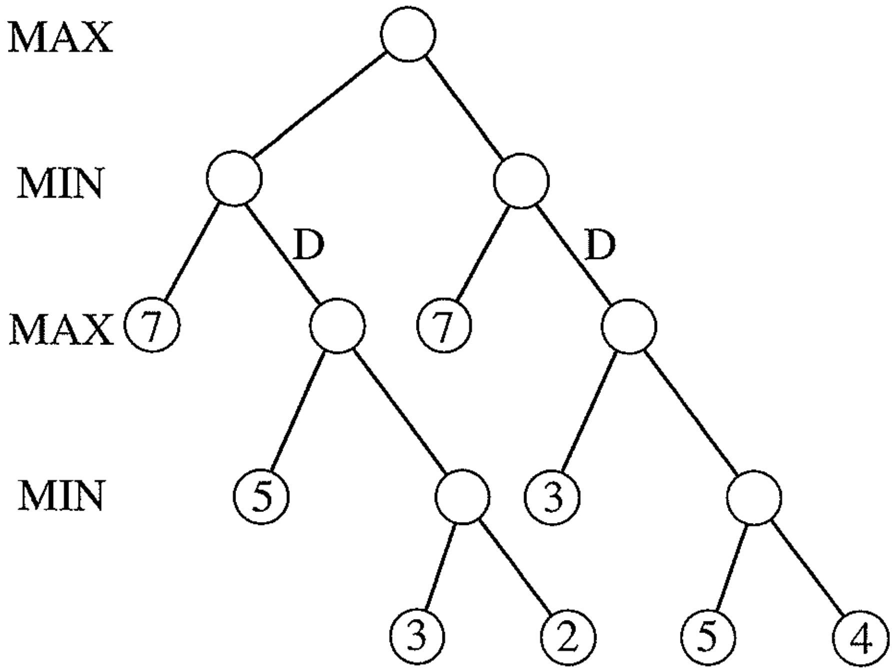
Solution:

3. (2 points) Suppose the MIN player decides to perform action B in his first turn, as shown in the solution space ABOVE in question 1, and play optimally thereafter. However, the MAX player does not know about this and continues to assume that the MIN player is acting optimally. What then is MAX player's payoff value when starting from the root of the tree?

Solution:

4. (4 points) Consider the minimax search tree shown in the solution space below; the utility function values are specified with respect to the MAX player and indicated at all the leaf (terminal) nodes. Suppose we use alpha-beta pruning algorithm, given in Figure 5.7 of AIMA 3rd edition (reproduced in Figure 3), in the direction from left to right to prune the search tree. Mark (with an "X") all arcs that are pruned by alpha-beta pruning.

Solution:



5. (1 point) What is the minimax value at the root node?

Solution:

6. (2 points) Suppose the MIN player decides to perform action D in his first turn, as shown in the solution space ABOVE in question 4, and play optimally thereafter. However, the MAX player does not know about this and continues to assume that the MIN player is acting optimally. What then is MAX player's payoff value when starting from root of the tree?

Solution:

Part III
Inference in First-Order Logic

(4 points) Short essay questions. Answer in the space provided on the script.

function UNIFY(x, y, θ) **returns** a substitution to make x and y identical
  inputs: x, a variable, constant, list, or compound expression
    y, a variable, constant, list, or compound expression
    θ, the substitution built up so far (optional, defaults to empty)

  if θ = failure **then return** failure
  else if $x = y$ **then return** θ
  else if VARIABLE?(x) **then return** UNIFY-VAR(x, y, θ)
  else if VARIABLE?(y) **then return** UNIFY-VAR(y, x, θ)
  else if COMPOUND?(x) **and** COMPOUND?(y) **then**
    return UNIFY(x.ARGS, y.ARGS, UNIFY(x.OP, y.OP, θ))
  else if LIST?(x) **and** LIST?(y) **then**
    return UNIFY(x.REST, y.REST, UNIFY(x.FIRST, y.FIRST, θ))
  else return failure

function UNIFY-VAR(var, x, θ) **returns** a substitution

  if $\{var/val\} \in \theta$ **then return** UNIFY(val, x, θ)
  else if $\{x/val\} \in \theta$ **then return** UNIFY(var, val, θ)
  else if OCCUR-CHECK?(var, x) **then return** failure
  else return add $\{var/x\}$ to θ

Figure 4: Unification algorithm.

1. (3 points) Give the most general unifier (if one exists) for the pair of sentences $Q(y, G(A, z))$ and $Q(G(x, B), y)$ that is PRODUCED by the unification algorithm given in Fig. 9.1 of AIMA 3rd edition (reproduced in Figure 4). You don't have to show your derivation.

Solution:

2. (1 point) We know that the most general unifier (MGU) is unique up to renaming and substitution of variable(s). Give another such MGU (if one exists) that is equivalent to that in question 1 above by SUBSTITUTION OF VARIABLE(S). You don't have to show your derivation.

Solution:

Part IV
Uncertainty

(6 points) Short essay questions. Answer in the space provided on the script.

For a long time, Bryan has been trying to train his baby daughter, Cara, to be capable of localizing herself among the COM1 and COM2 buildings. Cara has some initial belief (i.e, equal prior probability) about whether she is in COM1 or COM2:

$$\mathbf{P}(In\_COM1) = 0.5, \quad \mathbf{P}(In\_COM2) = 0.5\,.$$

When sensing in COM1, Cara will be able to observe with 0.8 probability that she is in COM1:

$$\mathbf{P}(Sense\_COM1 \mid In\_COM1) = 0.8, \quad \mathbf{P}(Sense\_COM2 \mid In\_COM1) = 0.2\,.$$

When sensing in COM2, Cara is absolutely uncertain (i.e., equal probability of being in COM1 and COM2) due to extremely poor lighting conditions:

$$\mathbf{P}(Sense\_COM1 \mid In\_COM2) = 0.5, \quad \mathbf{P}(Sense\_COM2 \mid In\_COM2) = 0.5\,.$$

1. (4 points) Suppose at this time, Cara senses that she is in $COM1$. Given this sensing information, calculate the posterior beliefs that (a) Cara is in $COM1$, and (b) Cara is in $COM2$ using Bayes rule. Show your derivation. No marks will be given if you do not show your derivation. Give your answer up to 4 decimal places.

Solution:

(a) $\mathbf{P}(In\_COM1 \mid Sense\_COM1) =$ (b) $\mathbf{P}(In\_COM2 \mid Sense\_COM1) =$

2. (2 points) Cara prefers to be more certain that she is in $COM1$ before informing Bryan; she has been scolded many times in the past for localizing wrongly. In particular, Cara wants to be at least 80% certain that she is in $COM1$. Suppose Cara has consecutively sensed m MORE TIMES that she is in $COM1$. What should the minimum value of m be in order to achieve at least 80% certainty that Cara is in $COM1$?

Solution: $m =$

Part V
Learning from Examples

(10 points) Short essay questions. Answer in the space provided on the script.

In this question, we will build a decision tree using the 8-example training set (see Table 1) to understand the factors affecting the grades of students enrolled in an AI class. This scenario is purely hypothetical.

Example	Input Attributes				Goal
Student	*AttendClass*	*MidtermGrade*	*TetrisPerf*	*LoveAI*	*AIGrade*
Prabhu	*Sometimes*	*Average*	*Poor*	*Yes*	*Pass*
Etkin	*Sometimes*	*Poor*	*Average*	*No*	*Fail*
Hancheng	*Rarely*	*Good*	*Average*	*No*	*Fail*
XuNuo	*Sometimes*	*Good*	*Average*	*Yes*	*Pass*
TrongNghia	*Always*	*Average*	*Good*	*Yes*	*Pass*
Arik	*Rarely*	*Poor*	*Good*	*Yes*	*Fail*
Yehong	*Rarely*	*Average*	*Good*	*Yes*	*Fail*
Ruofei	*Sometimes*	*Good*	*Poor*	*No*	*Fail*

Table 1: Examples for the *AIGrade* domain.

1. (3 points) What is the entropy of the goal attribute '*AIGrade*' on the whole set of examples shown in Table 1? Give your answer up to 4 decimal places.

Solution:

$H(Goal) =$

What is the information gain with choosing the input attribute '*AttendClass*' as the root of the decision tree? Give your answer up to 4 decimal places.

Solution:

$Gain(AttendClass) =$

What is the information gain with choosing the input attribute '*LoveAI*' as the root of the decision tree? Give your answer up to 4 decimal places.

Solution:

$Gain(LoveAI) =$

2. (5 points) Using the DECISION-TREE-LEARNING algorithm given in Fig. 18.5 of AIMA 3rd edition (reproduced in Figure 5) and information gain as the IMPORTANCE function in this algorithm, draw the resulting decision tree that is induced by the 8-example training set. You are required to label the non-leaf nodes with the input attributes, the leaf nodes with the AI grade (i.e. either '*Pass*' or '*Fail*'), and the branches with the values of the chosen attributes.

function DECISION-TREE-LEARNING(*examples*, *attributes*, *parent_examples*) **returns** tree

  if *examples* is empty **then return** PLURALITY-VALUE(*parent_examples*)
  else if all *examples* have the same classification **then return** the classification
  else if *attributes* is empty **then return** PLURALITY-VALUE(*examples*)
  else
    $A \leftarrow \text{argmax}_{a \in attributes}$ IMPORTANCE(a, *examples*)
    tree ← a new decision tree with root test A
    for each value v_k of A **do**
      $exs \leftarrow \{e : e \in examples$ **and** $e.A = v_k\}$
      subtree ← DECISION-TREE-LEARNING(*exs*, *attributes* − A, *examples*)
      add a branch to *tree* with label ($A = v_k$) and subtree *subtree*
    return *tree*

Figure 5: Decision tree learning algorithm.

Solution:

3. (2 points) Give the logical expression (in disjunctive normal form) that corresponds to the decision tree produced by the DECISION-TREE-LEARNING algorithm. Let '*Pass*' and '*Fail*' be associated with True and False, respectively.

Solution:

END OF PAPER