Student Number:

The University of Melbourne

Semester 1 Assessment 2021

School of Computing and Information Systems

COMP30024 Artificial Intelligence

Time for writing and upload: 3 hours 15 minutes.

This paper has 20 pages including this cover page.

Common Content Papers: None

Authorised Materials: Lecture notes, books, computer, on-line material

Instructions to Students:

- This paper counts for 70% of your final grade.
- There are 7 questions, with marks as indicated. Attempt all questions.
- Answer all the questions on the exam paper if possible, and then upload the completed exam paper containing your solutions. If you are unable to print the exam paper or electronically edit the exam paper, you may write on your own blank paper and then upload images of your written answers.
- You may upload your exam answers multiple times if you need to revise an answer at any time during the exam.
- You must not communicate with other students or seek assistance from anyone else taking whilst taking this exam, e.g. using messaging, chat rooms, email, telephone or face-to-face. Also, you must not assist anyone else taking the exam. You must not post answers to the questions or discussion of the questions online. Failure to comply with these instructions may be considered as academic misconduct.
- You are free to use the course materials and your laptop/PC in this exam but note that there is a strict time window for the exam hence you should be mindful of the time spent using such resources.
- Answer the questions as clearly and precisely as you can.
- Your writing should be clear. Unreadable answers will be deemed wrong. Excessively long answers or irrelevant information may be penalised.
- For numerical methods, marks will be given for applying the correct method. Students will not be heavily penalised for arithmetic errors.

Question 1 (10 marks)

Pick the most appropriate answer to each of the following questions. Please write your answer to each question in the boxes below. Each question is worth 2 marks.

Question	(a)	(b)	(c)	(d)	(e)
Answer					

(a) Which of the following statements is true in general:

1. Depth-first search always expands at least as many nodes as A* search with an admissible heuristic.
2. A perfectly rational backgammon agent never loses.
3. An advantage of Hill Climbing search is that it requires minimal memory.
4. Bi-directional search is always more efficient than uni-directional search.

(b) In a general CSP with n variables, each taking d possible values, what is the maximum number of times a backtracking search algorithm might have to backtrack, if it is running arc consistency and applying the Minimum Remaining Values (MRV) and (Least Constraining Value) LCV heuristics?

1. $O(1)$
2. $O(nd^2)$
3. $O(n^2d^3)$
4. $O(d^n)$

Question 1 (continued)

(c) You are given a game-tree for which you are the maximizer, and in the nodes in which you don't get to make a decision an action is chosen uniformly at random amongst the available options. Your objective is to maximize the probability that you win \$10 or more (rather than the usual objective to maximize your expected value). Note that Expectimax is a variant of Expectiminimax where nodes alternate between max nodes and chance nodes at different layers of the search tree (there are no min nodes). Then which of the following statements is true:

1. Running Expectimax will result in finding the optimal strategy to maximize the probability of winning \$10 or more.
2. Running Minimax, where chance nodes are considered minimizers, will result in finding the optimal strategy to maximize the probability of winning \$10 or more.
3. Running Expectimax in a modified game tree where every pay-off of \$10 or more is given a value of 1, and every pay-off lower than \$10 is given a value of 0 will result in finding the optimal strategy to maximize the probability of winning \$10 or more.
4. Running Minimax in a modified game tree where every pay-off of \$10 or more is given a value of 1, and every pay-off lower than \$10 is given a value of 0 will result in finding the optimal strategy to maximize the probability of winning \$10 or more.

(d) Consider that you need to run four auctions labelled (A), (B), (C) and (D), where the most important property for each auction is as follows:

(A) Hard to collude, (B) Winner pays second highest price, (C) Open-cry, (D) Decending

Which set of auction types below is the best match for these four auctions based on their respective desired properties?

1. (A) Vickery, (B) Dutch, (C) English, (D) First-price sealed-bid
2. (A) Vickery, (B) English, (C) Dutch, (D) First-price sealed-bid
3. (A) First-price sealed-bid, (B) Dutch, (C) English, (D) Vickery
4. (A) First-price sealed-bid, (B) Vickery, (C) English, (D) Dutch

Question 1 (continued)

(e) Figure 1-1 shows a robot arm that is made up of two sections Arm_1 and Arm_2. Arm_1 can rotate around the shoulder joint A, while Arm_2 is connected to Arm_1 at the elbow joint B, and can rotate around the joint B. The configuration of the robot arm can be specified by the angle θ_1 between the horizontal axis and Arm_1, and the angle θ_2 between Arm_1 and Arm_2. Both angles are measured in radians. There is also 1 fixed obstacle shown, which restricts the movement of the robot arm.

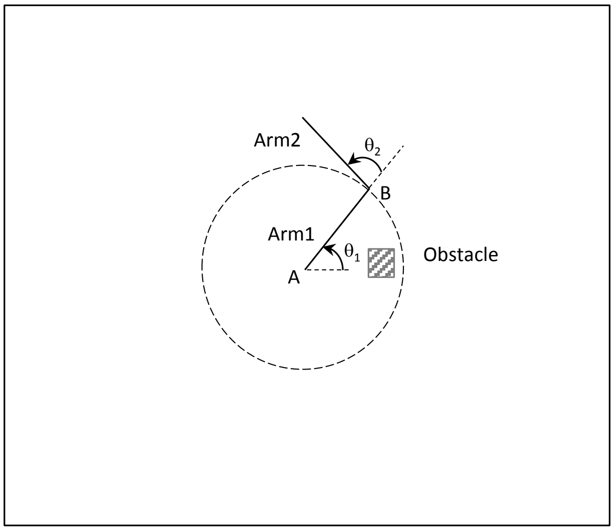


Figure 1-1

Which of the following four figures best represents the *configuration space* for this robot? The figures are labelled (1) to (4).

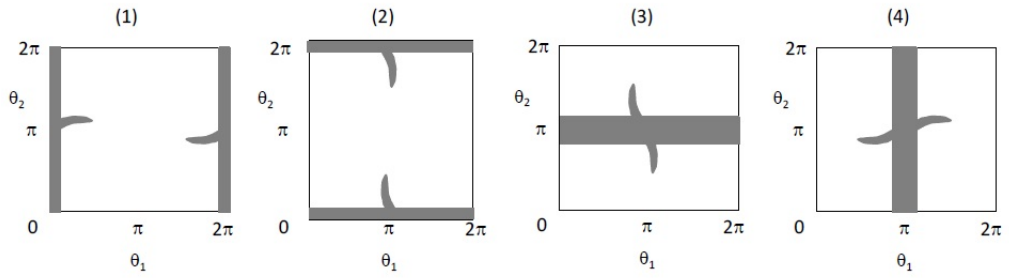


Figure 1-2

Question 2 (10 marks)

For each part of the following question you should write a brief answer in the box provided.

(a) [2 marks] The following is the weight update rule for gradient decent learning as described in the lectures

$$w_i \leftarrow w_i - \eta(z - t)f_i(s)$$

where η is the learning rate, and w is the vector of weights in the evaluation function. Why it is important to get the learning rate parameter η right?

Answer:

(b) [2 marks] In the weight update rule in part (a), what does a small difference between z and t indicate about the current weight w_i?

Answer:

Question 2 (continued)

(c) [6 marks] For each of the following three environments, what is the most appropriate type of agent (simple reflex agent, model-based reflex agent, goal-based agent, or utility-based agent)? Briefly justify your answer.

(i) An automated system that translates text from one language to another, e.g., Google translate
(ii) A movie recommendation system
(iii) An automated taxi driver that is required to reach the airport on time

Answer:

Question 3 (10 marks)

Consider the 3-ply game tree shown in Figure 3-1. Each node has an identifier (e.g., the root of the tree is node 1; it has three successor nodes 2, 12 and 22), and each terminal node has an associated value (e.g., the value of node 4 is 8).

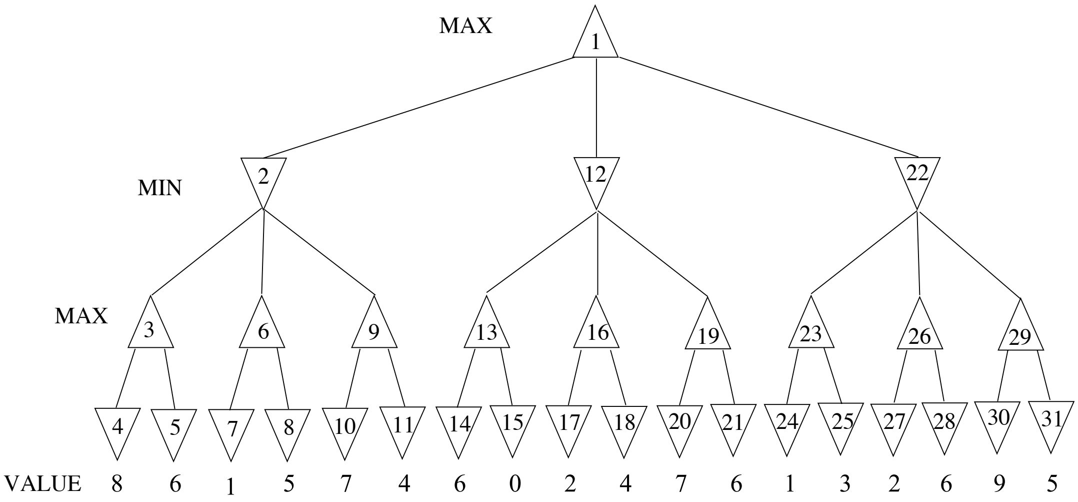


Figure 3-1

In the following questions, you are NOT required to redraw the search tree in your answer.

(a) [1 mark] What is the minimax value at node 1 after applying the minimax algorithm to this search tree? **Answer:**

(b) [4 marks] If the nodes are examined in the order shown by the identifier in each node in Figure 3-1, which nodes would be pruned if alpha-beta pruning is used? For each node that would be pruned, place a cross in the corresponding box below.

Answer:

Node	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Pruned																

Node	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Pruned															

(c) [2 marks] If we come up with the perfect ordering of the nodes, what is the maximum number of nodes that we can prune? **Answer:**

Question 3 (continued)

(d) [3 marks] What are *three properties* that characterise an effective mechanism design for an auction? Note: you should briefly define each property that you give in your answer.

Write your answer in the space provided below. If you require more space, please use the last page of the exam paper, and clearly indicate on this page that you have used the last page.

Answer:

Question 4 (10 marks)

If a search problem has only a single goal state, it is common to perform bidirectional search. In bidirectional search you build two search trees at the same time: the "forward" search tree is the one we have always worked with in COMP30024, while the "backward" search tree is the one that starts from the goal state, and calls a predecessor (rather than successor) function to work its way back to the start state. Both searches use the same cost function for transitioning between two states. There will now also be a backward heuristic, which for each state estimates the distance to the start state. Bidirectional search can result in significant computational advantages: the size of the search tree built grows exponentially with the depth of the search tree. By growing trees from the start and goal towards each other, these two trees could meet in the middle. Thus, one ends up with a computational complexity of just twice that of searching a tree of half the depth, which is a very significant saving.

Recall the pseudo-code for a standard A* graph search

Algorithm A* Search

```
Function Astar-Graph-Search(problem)
forward-closed ← empty set
forward-priority-queue
        ← Insert(Make-Node(Start-State(problem)), forward-priority-queue)
loop
  if forward-priority-queue is empty then return failure
  end if
  if forward-priority-queue is not empty then
    node ← Pop(forward-priority-queue)
    if (State(node) == Goal-State(problem) )  then return node
    end if
    if  State(node) is not in forward-closed  then
      add State(node) to forward-closed
      forward-priority-queue
              ← Insert-All(ExpandForward(node, problem), forward-priority-queue)
    end if
  end if
end loop
```

Now consider the following tentative pseudo-code for bidirectional A* search. We assume a consistent forward heuristic, and a consistent backward heuristic. Concatenation is a function that builds a plan (i.e., a sequence of actions) that goes from the start state to the goal state by combining a forward partial plan and a backward partial plan that end in the same state.

Question 4 (continued)

Algorithm Bidirectional A* Search

Function Bidirectional-Graph-Search(*problem*)
forward-closed ← empty set
backward-closed ← empty set
forward-priority-queue
    ← Insert(Make-Node(Start-State(problem)), forward-priority-queue)
backward-priority-queue
    ← Insert(Make-Node(Goal-State(problem)), backward-priority-queue)
loop
  if forward-priority-queue is empty AND backward-priority-queue is empty **then**
    return failure
  end if

  if there exist a node n1 in forward-priority-queue and a node n2 in backward
    priority queue such that State(n1) == State(n2) **then** {IF no:1}
    return Concatenation of n1 and n2
  end if

  if forward-priority-queue is not empty **then**
    node ← Pop(forward-priority-queue)
    if (State(node) == Goal-State(problem)) **then return** node
    end if
    if (State(node) is in backward-priority-queue) **then** {IF no:2}
      return Concatenation of node and matching node in backward-priority-queue
    end if
    if (State(node) is in backward-closed) **then** {IF no:3}
      return Concatenation of node and matching node in backward-closed
    end if
    if (State(node) is not in forward-closed) **then**
      add State(node) to forward-closed
      forward-priority-queue
          ← Insert-All(ExpandForward(node, problem), forward-priority-queue)
    end if
  end if

Question 4 (continued)

Algorithm Bidirectional A* Search (continued)

```
    if backward-priority-queue is not empty  then
      node ← Pop(backward-priority-queue)
      if (State(node) == Start-State(problem))  then return node
      end if
      if  ( State(node) is in forward-priority-queue )  then {IF no:4}
        return Concatenation of node and matching node in forward-priority-queue
      end if
      if  ( State(node) is in forward-closed )  then {IF no:5}
        return Concatenation of node and matching node in forward-closed
      end if
      if  ( State(node) is not in backward-closed )  then
        add State(node) to backward-closed
        backward-priority-queue
            ← Insert-All(ExpandBackward(node, problem), backward-priority-queue)
      end if
    end if

  end loop
```

The IF statements labeled 1, 2, 3, 4, 5 are modifications to try to connect both search trees.

(a) [2 marks] If we cut out all statements labeled 1, 2, 3, 4 or 5, will Bidirectional-Graph-Search return an optimal solution? Briefly justify your answer.

Answer:

Question 4 (continued)

(b) [2 marks] If amongst the numbered statements we only retain 1, is Bidirectional-Graph-Search guaranteed to be optimal? Briefly justify your answer.

Answer:

(c) [2 marks] If amongst the numbered statements we only retain 2, is Bidirectional-Graph-Search guaranteed to be optimal? Briefly justify your answer.

Answer:

Question 4 (continued)

(d) [2 marks] If amongst the numbered statements we only retain 3, is Bidirectional-Graph-Search guaranteed to be optimal? Briefly justify your answer.

Answer:

(e) [2 marks] If amongst the numbered statements we only retain 5, is Bidirectional-Graph-Search guaranteed to be optimal? Briefly justify your answer.

Answer:

Question 5 (10 marks)

There are five undergraduate CS classes and three lecturers who will be teaching these classes. A lecturer can only teach one class at a time. The classes are:

- Class 1 - Algorithms: meets from 8:00-9:00am
- Class 2 - Intro to AI: meets from 8:30-9:30am
- Class 3 - Databases: meets from 9:00-10:00am
- Class 4 - Operating Systems: meets from 9:00-10:00am
- Class 5 - Machine Learning: meets from 9:30-10:30am

and the instructors are:

- Professor A, who is available to teach Classes 3 and 4.
- Professor B, who is available to teach Classes 2, 3, 4, and 5.
- Professor C, who is available to teach Classes 1, 2, 3, 4, 5.

(a) [4 marks] Formulate this as a CSP problem in which there is one variable per class. State the variables and their domains, and the constraints. Constraints should be unary or binary and specified formally and precisely, but may be implicit rather than explicit.

Answer:

Question 5 (continued)

(b) [2 marks] Draw the constraint graph associated with your CSP.

Answer:

(c) [2 marks] Show the domains of the variables after running arc-consistency on this initial graph (after having already enforced any unary constraints).

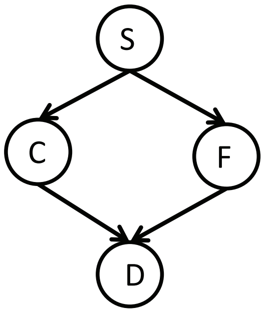
Answer:

(d) [2 marks] Give one solution to this CSP.

Answer:

Question 6 (10 marks)

Consider the Bayes network shown in Figure 6-1 for a lift at a university, where S = Semester, C = Crowded, F = FaultyDoor and D = DelayedLift are all Boolean random variables, i.e., they take the value either *true (t)* or *false (f)*. Also note that $P(c)$ is shorthand for $P(C = \text{true})$ and $P(\neg c)$ is shorthand for $P(C = \text{false})$.



P(S)
0.6

S	P(C|S)
t	0.8
f	0.5

S	P(F|S)
t	0.5
f	0.2

C	F	P(D|C,F)
t	t	0.9
t	f	0.8
f	t	0.6
f	f	0.1

Figure 6-1

Question 6 (continued)

[10 marks] Calculate the value of $P(f|d, s)$. If you cannot easily calculate the final value, try to simplify the expression as best you can. Justify your answer mathematically and clearly show your calculations.

Answer:

Question 7 (10 marks)

Someone is trying to send you a message in a very simple language. The language uses only two characters {A, B}. A message in the language is a string of A's and B's. Furthermore, messages in the language always start with the character A. For example, "AB" or "AABA" are both valid messages in the language.

Let S_i represent the character at position i in a message (S_1 denotes the first character in the message). For example, in the message "AB", $S_1 = A$ and $S_2 = B$. Each character in the message is generated at random (except for S_1), where the probability of a particular character being generated depends only on the previous character in the message. These conditional probabilities $P(S_{i+1}|S_i)$ are given as follows:
$P(S_{i+1} = A|S_i = A) = 0.6$, $P(S_{i+1} = B|S_i = A) = 0.4$, $P(S_{i+1} = A|S_i = B) = 0.2$, $P(S_{i+1} = B|S_i = B) = 0.8$

[10 marks] When a message is sent to you, noise can cause some characters to be confused with each other. For example, the message "ABB" might be received as "BBB", where the first character was corrupted by noise and thus confused in the received message.

Let S represent the character that is sent, and let R represent the character that is received. The "confusion" probability of S and R is defined by the probability distribution $P_c(R|S)$. These conditional probabilities are given as follows: $P_c(R = A|S = A) = 0.9$, $P_c(R = B|S = A) = 0.1$, $P_c(R = A|S = B) = 0.1$, $P_c(R = B|S = B) = 0.9$

What is the most likely valid message to have been transmitted if you receive the message "AB"?

Justify you answer mathematically and clearly show all your calculations and assumptions.

Question 7 (continued)

Answer:

END OF EXAM QUESTIONS

Extra space if needed to answer any question. If you write part of your answer here, please write the question number, and indicate at the corresponding question that you have used this space.

LAST PAGE OF EXAM