CS 188 Fall 2024 — Introduction to Artificial Intelligence — Midterm Review

Q1. Ants: Escape!

An ant wakes up and finds itself in a spider's maze!

- The maze is an M-by-N rectangle.
- Legal actions: $\{Forward, TurnLeft, TurnRight\}$.
- Transition model: *Forward* moves the ant 1 square in the direction it's facing, unless there is a wall in front. The two turning actions rotate the ant by 90 degrees to face a different direction.
- Action cost: Each action costs 1.
- Start state: The ant starts at (s_x, s_y) facing North.
- Goal test: Returns true when the ant reaches the exit at $G = (g_x, g_y)$.

(a) **(i)** What's the minimum state space size S for this task?

$S =$ [ ]

(ii) Now suppose there are K ants, where each ant i must reach a distinct goal location G_i; any number of ants can occupy the same square; and **action costs are a sum of the individual ants' step costs**. What's the minimum state space size for this task, expressed in terms of K and S?

[ ]

(iii) Now suppose that each ant i can exit at any of the goal locations G_j, but no two ants can occupy the same square **if they are facing the same direction**. What's the minimum state space size for this task, expressed in terms of K and S?

[ ]

(iv) Now suppose, once again, that each ant i must reach its own exit at G_i, and no two ants can occupy the same square **if they are facing the same direction**. Let $H = \sum_i h_i^*$, where h_i^* is the optimal cost for ant i to reach goal G_i when it is the only ant in the maze. Is H admissible for the K-ant problem? Select all appropriate answers.

- ☐ Yes, because for any multiagent problem the sum of individual agent costs, with each agent solving a subproblem separately, is always a lower bound on the joint cost.
- ☐ Yes, because H is the exact cost for a relaxed version of the K-ant problem.
- ☐ Yes, because the "no two ants…" condition can only make the true cost larger than H, not smaller.
- ☐ No, because some ants can exit earlier than others so the sum may overestimate the total cost.
- ☐ No, it should be $\max_i$ rather than $\sum_i$.
- ○ None of the above

(b) The ant is alone again in the maze. Now, the spider will return in T timesteps, so the ant must reach an exit in T or fewer actions. Any sequence with more than T actions doesn't count as a solution.

In this part, we'll address this by solving the original problem and checking the resulting solution. That is, suppose p is a problem and A is a search algorithm; $A(p)$ returns a solution s, and $\ell(s)$ is the length (number of actions) of s, where $\ell(\mathit{failure}) = \infty$. Let p_T be p with the added time limit T. Then, given A, we can define a new algorithm $A'(p_T)$ as follows:

- $s \leftarrow A(p)$; **if** $\ell(s) \leq T$ **then return** s **else return** *failure*.

(i) ◯ T ◯ F Suppose A is an optimal algorithm for p, action costs are 1; then A' is optimal for p_T.

(ii) ◯ T ◯ F Suppose A is a complete algorithm for p; then A' is complete for p_T.

(iii) ◯ T ◯ F Suppose A is an optimal algorithm for p, and action costs may be any nonnegative real number; then A' is optimal for p_T.

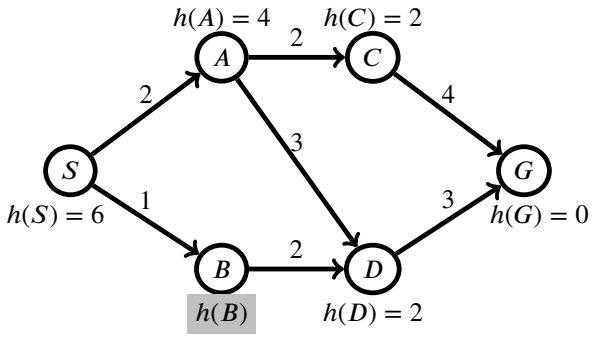
(c) Now we attempt to solve the time-limited problem by *modifying the problem definition* (specifically, the states, legal actions in each state, and/or goal test) appropriately so that regular, unmodified search algorithms will automatically avoid returning solutions with more than T actions.

(i) Is this possible *in general, for any problem* where actions costs are all 1? Mark all correct answers.

- ☐ Yes, by augmenting the state space only.
- ☐ Yes, by augmenting the state space and modifying the goal test.
- ☐ Yes, by modifying the goal test only.
- ☐ Yes, by augmenting the state space and modifying the legal actions.
- ☐ Yes, by modifying the legal actions only.
- ◯ No, it's not possible in general.

(ii) Subpart removed for time.

Q2. Informed Search



Search problem graph: S is the start state and G is the goal state.
Tie-break in alphabetical order.
$h(B)$ is unknown and will be determined in the subquestions.

(a) In this question, refer to the graph above where the optimal path is $S \to B \to D \to G$. For each of the following subparts, you will be asked to write ranges of $h(B)$. You should represent ranges as $\_\_ \leq h(B) \leq \_\_$. *Heuristic values can be any number including* $\pm\infty$. *For responses of* $\pm\infty$*, you can treat the provided inequalities as a strict inequality.* If you believe that there is no possible range, please write "None" in the left-hand box and leave the right box empty.

(i) What is the range for the heuristic to be admissible?

$\leq h(B) \leq$

(ii) What range of heuristic values for B would allow A* tree search to still return the optimal path ($S \to B \to D \to G$)?

$\leq h(B) \leq$

(iii) Now assume that the edges in the graph are undirected (which is equivalent to having two directed edges that point at both directions with the same cost as before). Regardless of whether the heuristic is consistent, what range of heuristic values for B would allow A* tree search to still return the optimal path ($S \to B \to D \to G$)?

$\leq h(B) \leq$

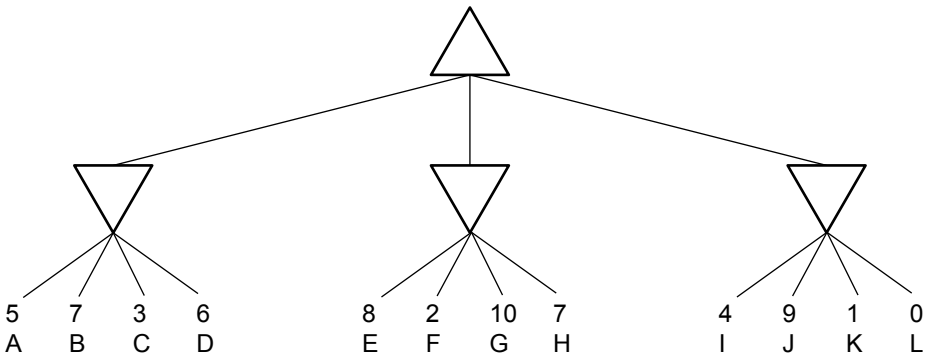
(b) Part b removed on this worksheet for cooking too hard, try it on your own time for extra practice.

Q3. Games

(a) Minimax and Alpha-Beta Pruning We have a two-player, zero-sum game with k rounds. In each round, the maximizer acts first and chooses from n possible actions, then the minimizer acts next and chooses from m possible actions. After the minimizer's k-th turn, the game finishes and we arrive at a utility value (leaf node). Both players behave optimally. Explore nodes from left to right.

(i) What is the total number of leaf nodes in the game tree, in terms of m, n, k? [ ]

(ii) In the minimax tree below $k = 1, n = 3, m = 4$.



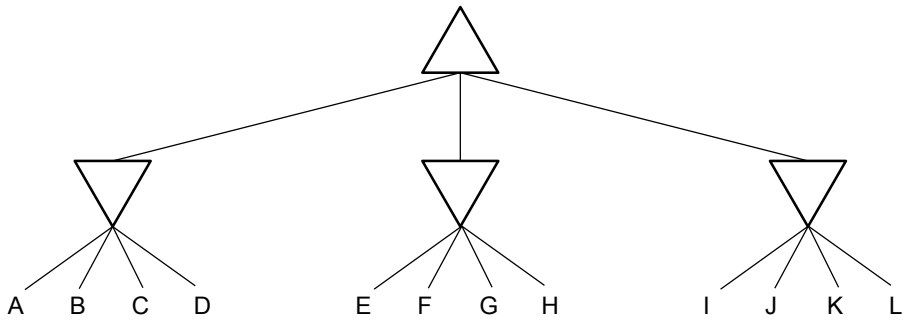
(1) What is the minimax value of the root? [ ]

(2) Which leaf nodes are pruned by alpha-beta? Mark the corresponding letters below.

☐ A ☐ B ☐ C ☐ D ☐ E ☐ F ☐ G ☐ H ☐ I ☐ J ☐ K ☐ L ○ None

(iii) When $k = 1$, in the *best case* (pruning the most nodes possible), which nodes can be pruned in the tree below?

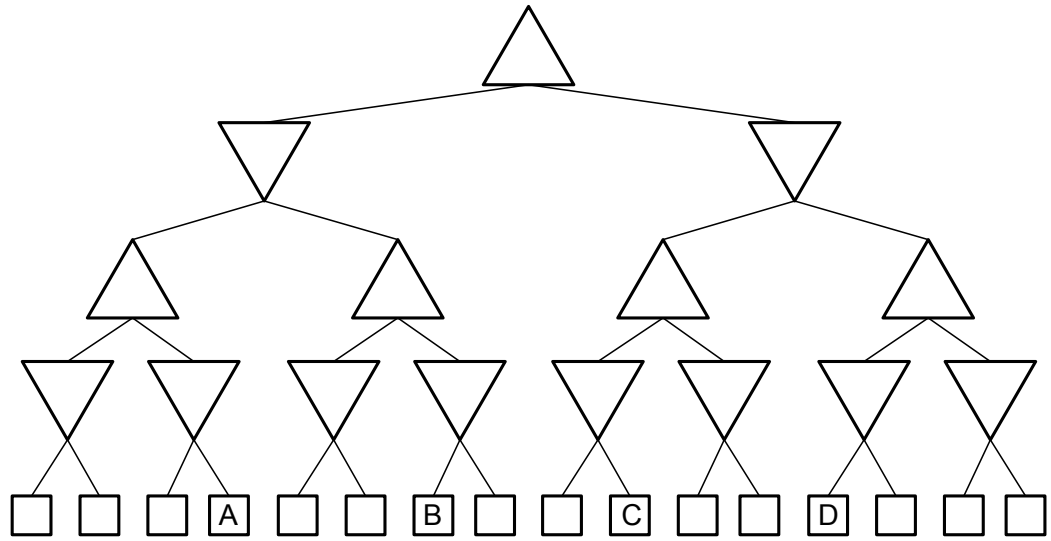
☐ A ☐ B ☐ C ☐ D ☐ E ☐ F ☐ G ☐ H ☐ I ☐ J ☐ K ☐ L ○ None



(iv) Now consider the same $k = 1$ but with a general m and n number of maximizer and minimizer actions respectively. How many leaf nodes would be pruned in the best case? Express your answer in terms of m and n.

[ ]

(v) When $k = 2, n = 2, m = 2$, in the best case, which of the leaves labeled A, B, C, D will be pruned?

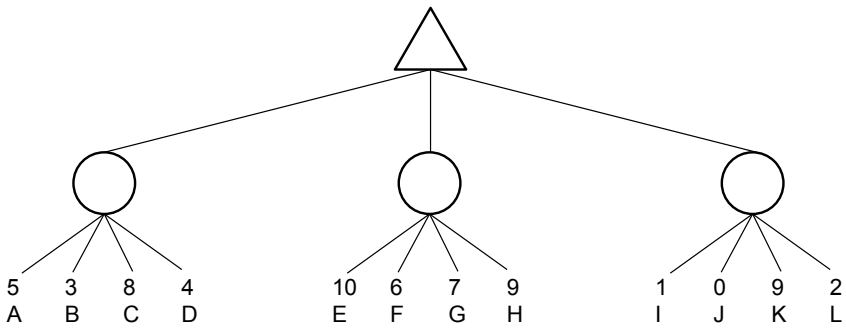


☐ A ☐ B ☐ C ☐ D ○ None

(b) Chance Nodes Our maximizer agent is now playing against a non-optimal opponent. In each round, the maximizer acts first, then the opponent acts next and chooses uniformly at random from m possible actions.

(i) Subpart removed for time (simple expectimax)

(ii) Consider the game tree below where we now **know** that the opponent always has $m = 4$ possible moves and chooses uniformly at random. **We also know that all leaf node utility values are less than or equal to $c = 10$.**



(1) What is the value of the root node? ☐

(2) Which nodes can be pruned?

☐ A ☐ B ☐ C ☐ D ☐ E ☐ F ☐ G ☐ H ☐ I ☐ J ☐ K ☐ L ○ None

(c) Now, let's generalize this idea for pruning on expectimax. We consider expectimax game trees where the opponent always chooses uniformly at random from m possible moves, and all leaf nodes have values no more than c. These facts are known by the maximizer player.

(i) Let's say that our depth-first traversal of this game tree is currently at a chance node and has seen k children of this node so far. The sum of the children seen so far is S. What is the largest possible value that this chance node can take on? (Answer in terms of m, c, k, and S)

(ii) [OPTIONAL]

Now, let's write an algorithm for computing the root value. Fill in the pseudocode below.

Note that m and c are constants that you should use in your pseudocode. To find the value at the root, we will start with a call to MAX-VALUE(*root*, $-\infty$).

```
1:  function MAX-VALUE(state, α)
2:      if state has no successors then return eval(state)
3:      v ← _____
4:      for each successor n of state do
5:          v ← _____
6:          _____
7:      return v
8:
9:  function EXP-VALUE(state, α)
10:     if state has no successors then return eval(state)
11:     S ← 0
12:     k ← 1
13:     for each successor n of state do
14:         S ← S + _____
15:         ci ← "expression from (c)(i) using m, c, k, and S"
16:         if _____ then
17:             return S/m
18:         k ← k + 1
19:     return S/m
```

Q4. CSPs: The Zookeeper

You are a newly appointed zookeeper, and your first task is to find rooms for all of the animals.

The zoo has three animals: the Iguana (I), Jaguar (J), and Koala (K).

Each animal needs to be assigned to one of four rooms: the North room (N), East room (E), South room (S), or West room (W), subject to the following constraints:

1. The jaguar cannot share a room with any other animal.
2. The iguana and koala must be in different rooms.
3. The koala can only be in the East room or the South room.

(a) Consider the first constraint: "The jaguar cannot share a room with any other animal."

Can this constraint be expressed using only binary constraints on the three variables I, J, K?

- ◯ Yes, it can be expressed as 1 binary constraint.
- ◯ Yes, it can be expressed as 2 different binary constraints.
- ◯ Yes, it can be expressed as 4 different binary constraints.
- ◯ No, this is necessarily a unary constraint.
- ◯ No, this is necessarily a higher-order constraint.

(b) Suppose we enforce unary constraints, and then assign the jaguar to the South room. The remaining values in each domain would be:

Iguana:	North, East, South, West
Jaguar:	South
Koala:	East, South

In the table below, mark each value that would be **removed** by running forward-checking after this assignment.

	North	East	South	West
Iguana	☐	☐	☐	☐
Koala		☐	☐	

(c) Regardless of your answer to the previous subpart, suppose we've done some filtering and the following values remain in each variable's domain:

Iguana:	East, West
Jaguar:	South
Koala:	East

Are all the arcs in this CSP consistent?

- ◯ Yes, all arcs are consistent.
- ◯ No, only $K \rightarrow I$ is not consistent.
- ◯ No, only $I \rightarrow K$ is not consistent.
- ◯ No, $K \rightarrow I$ and $I \rightarrow K$ are both not consistent.
- ◯ No, only $J \rightarrow I$ is not consistent.
- ◯ No, only $I \rightarrow J$ is not consistent.
- ◯ No, $J \rightarrow I$ and $I \rightarrow J$ are both not consistent.

The constraints, repeated here for your convenience:

1. The jaguar cannot share a room with any other animal.
2. The iguana and koala must be in different rooms.
3. The koala can only be in the East room or the South room.

(d) Regardless of your answer to the previous subpart, suppose we start over and just enforce the third constraint. Then the remaining values in each domain are:

Iguana:	North, East, South, West
Jaguar:	North, East, South, West
Koala:	East, South

What does the minimum remaining values (MRV) heuristic suggest doing next?

- ◯ Assign North or West to a variable next.
- ◯ Assign East or South to a variable next.
- ◯ Assign a value to Koala next.
- ◯ Assign a value to Iguana or Jaguar next.

(e) Again, consider the CSP after just enforcing the third constraint:

Iguana:	North, East, South, West
Jaguar:	North, East, South, West
Koala:	East, South

Which assignment would the least constraining value (LCV) heuristic prefer?

- ◯ Assign North to Jaguar.
- ◯ Assign East to Jaguar.
- ◯ LCV is indifferent between these two assignments.

(f) Suppose we add another constraint:

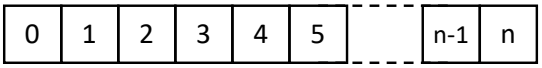
"The West room can contain at most one animal."

Can this constraint be expressed using only binary constraints on the three variables I, J, K?

- ◯ Yes, it can be expressed as 1 binary constraint.
- ◯ Yes, it can be expressed as 2 different binary constraints.
- ◯ Yes, it can be expressed as 3 different binary constraints.
- ◯ No, this is necessarily a higher-order constraint.

Q5. MDPs: Flying Pacman

Pacman is in a 1-dimensional grid with squares labeled 0 through n, inclusive, as shown below:



Pacman's goal is to reach square n as cheaply as possible. From state n, there are no more actions or rewards available.

At any given state, if Pacman is not in n, Pacman has two actions to choose from:

- **Run**: Pacman deterministically advances to the next state (i.e. from state i to state $i+1$). This action costs Pacman \$1.
- **Fly**: With probability p, Pacman directly reaches state n. With probability $1-p$, Pacman is stuck in the same state. This action costs Pacman \$2.

(a) Fill in the blank boxes below to define the MDP. i represents an arbitrary state in the range $\{0, \ldots, n-1\}$.

s	a	s'	$T(s,a,s')$	$R(s,a,s')$
i	Run	$i+1$		
i	Fly	i		
i	Fly			

For the next three subparts, assume that $\gamma = 1$.

Let π_R denote the policy of always selecting Run, and π_F denote the policy of always selecting Fly.

Compute the values of these two policies. Your answer should be an expression, possibly in terms of n, p, and/or i.

(b) What is $V^{\pi_R}(i)$?

(c) What is $V^{\pi_F}(i)$?

Hint: Recall that the mean of a geometric distribution with success probability p is $\sum_{k=1}^{\infty} k(1-p)^{k-1}p = 1/p$.

(d) Given the results of the two previous subparts, we can now find the optimal policy for the MDP.

Which of the following are true? Select all that apply. (Hint: consider what value of i makes $V^{\pi_R}(i)$ and $V^{\pi_F}(i)$ equal.)

Note: $\lceil x \rceil$ is the smallest integer greater than or equal to x.

- ☐ If $p < 2/n$, Fly is optimal for all states.
- ☐ If $p < 2/n$, Run is optimal for all states.
- ☐ If $p \geq 2/n$, Fly is optimal for all $i \geq \lceil n - 2/p \rceil$ and Run is optimal for all $i < \lceil n - 2/p \rceil$.
- ☐ If $p \geq 2/n$, Run is optimal for all $i \geq \lceil n - 2/p \rceil$ and Fly is optimal for all $i < \lceil n - 2/p \rceil$.
- ○ None of the above.

Regardless of your answers to the previous parts, consider the following modified transition and reward functions (which may not correspond to the original problem). As before, once Pacman reaches state n, no further actions or rewards are available.

For each modified MDP and discount factor, select whether value iteration will converge to a finite set of values.

(e) $\gamma = 1$

s	a	s'	$T(s, a, s')$	$R(s, a, s')$
i	Run	$i + 1$	1.0	+5
i	Fly	$i + 1$	1.0	+5

- ◯ Value iteration converges
- ◯ Value iteration does not converge
- ◯ Not enough information to decide

(f) $\gamma = 1$

s	a	s'	$T(s, a, s')$	$R(s, a, s')$
i	Run	$i + 1$	1.0	+5
i	Fly	$i - 1$	1.0	+5

- ◯ Value iteration converges
- ◯ Value iteration does not converge
- ◯ Not enough information to decide

(g) $\gamma < 1$

s	a	s'	$T(s, a, s')$	$R(s, a, s')$
i	Run	$i + 1$	1.0	+5
i	Fly	$i - 1$	1.0	+5

- ◯ Value iteration converges
- ◯ Value iteration does not converge
- ◯ Not enough information to decide

Q6. RL: Rest and ReLaxation

Consider the grid world MDP below, with unknown transition and reward functions.

A	B	C
D	E	F
G	H	I

The agent observes the following samples in this grid world:

s	a	s'	$R(s,a,s')$
E	East	F	-1
E	East	H	-1
E	South	H	-1
E	South	H	-1
E	South	D	-1

Reminder: In grid world, each non-exit action succeeds with some probability. If an action (e.g. North) fails, the agent moves in one of the cardinally adjacent directions (e.g. East or West) with equal probability, but will not move in the opposite direction (e.g. South).

Let p denote the probability that an action succeeds.

In this question, we will consider 3 strategies for estimating the transition function in this MDP.

Strategy 1: The agent does not know the rules of grid world, and runs model-based learning to directly estimate the transition function.

(a) From the samples provided, what is $\hat{T}$(E, South, H)?

◯ 0 ◯ 1/5 ◯ 2/5 ◯ 3/5 ◯ 4/5 ◯ 1/3 ◯ 2/3 ◯ 1/2 ◯ 1 ◯ Not enough information

(b) From the samples provided, what is $\hat{T}$(E, West, D)?

◯ 0 ◯ 1/5 ◯ 2/5 ◯ 3/5 ◯ 4/5 ◯ 1/3 ◯ 2/3 ◯ 1/2 ◯ 1 ◯ Not enough information

Strategy 2: The agent knows the rules of grid world, and runs model-based learning to estimate p. Then, the agent uses the estimated $\hat{p}$ to estimate the transition function.

(c) From the samples provided, what is $\hat{p}$, the estimated probability of an action succeeding?

◯ 0 ◯ 1/5 ◯ 2/5 ◯ 3/5 ◯ 4/5 ◯ 1/3 ◯ 2/3 ◯ 1/2 ◯ 1 ◯ Not enough information

(d) Based on $\hat{p}$, what is $\hat{T}$(E, West, D)?

◯ 0 ◯ 1/5 ◯ 2/5 ◯ 3/5 ◯ 4/5 ◯ 1/3 ◯ 2/3 ◯ 1/2 ◯ 1 ◯ Not enough information

(e) Select all true statements about comparing Strategy 1 and Strategy 2.

☐ Strategy 1 will usually require fewer samples to estimate the transition function to the same accuracy threshold.
☐ There are fewer unknown parameters to learn in Strategy 1.
☐ Strategy 1 is more prone to overfitting on samples.
◯ None of the above

The grid world and samples, repeated for your convenience:

A	B	C
D	E	F
G	H	I

s	a	s'	$R(s,a,s')$
E	East	F	-1
E	East	H	-1
E	South	H	-1
E	South	H	-1
E	South	D	-1

Strategy 3: The agent knows the rules of grid world, and uses an exponential moving average to estimate p. Then, the agent uses the estimated $\hat{p}$ to estimate the transition function.

(f) Consider this update equation: $\hat{p} \leftarrow (1-\alpha)\hat{p} + (\alpha)x$

Given a sample (s, a, s'), what value of x should be used in the corresponding update?

- ◯ $R(s, a, s')$
- ◯ 1.0 if the action succeeded, and 0.0 otherwise
- ◯ 1.0 if the action failed, and 0.0 otherwise
- ◯ $V(s)$
- ◯ $V(s')$

(g) Select all true statements about comparing Strategy 2 and Strategy 3.

- ☐ Strategy 2 gives a more accurate estimate, because it is the maximum likelihood estimate.
- ☐ Strategy 3 gives a more accurate estimate, because it gives more weight to more recent samples.
- ☐ Strategy 3 can be run with samples streaming in one at a time.
- ◯ None of the above

The rest of the question is independent from the previous subparts.

Suppose the agent runs Q-learning in this grid world, with learning rate $0 < \alpha < 1$, and discount factor $\gamma = 1$.

(h) After iterating through the samples once, how many learned Q-values will be nonzero?

◯ 0 ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ > 4

(i) After iterating through the samples repeatedly until convergence, how many learned Q-values will be nonzero?

◯ 0 ◯ 1 ◯ 2 ◯ 3 ◯ 4 ◯ > 4

Q7. Potpourri [OPTIONAL]

(a) Below is a list of task environments. For each of the sub-parts, choose all the environments in the list that falls into the specified type.

A: The competitive rock-paper-scissors game
B: The classical Pacman game (with ghosts following a fixed path)
C: Solving a crossword puzzle
D: A robot that removes defective cookies from a cookie conveyor belt

(i) Which of the environments can be formulated as *single-agent*? ☐ A ☐ B ☐ C ☐ D

(ii) Which of the environments are *static*? ☐ A ☐ B ☐ C ☐ D

(iii) Which of the environments are *discrete*? ☐ A ☐ B ☐ C ☐ D

(b) **(i)** ◯ T ◯ F Reflex agents cannot be rational.

(ii) ◯ T ◯ F There exist task environments in which no pure reflex agent can behave rationally.

(c) **(i)** ◯ T ◯ F If the costs can be arbitrarily large negative numbers in a search problem, then any optimal search algorithm in this problem will need to explore the entire state space.

(ii) ◯ T ◯ F Depth-first search always expands at least as many nodes as A* search with an admissible heuristic.

(d) **(i)** ◯ T ◯ F Local beam search with a beam size of 1 reduces to Hill climbing.

(ii) ◯ T ◯ F Local beam search with one initial state and no limit on the number of states retained reduces to depth-first search.