EN.520.650.01.SP22: Machine Intelligence
Homework - 1 Solutions

---

1. For each of the following activities, give a PEAS description of the task environment and characterize it in terms of the properties listed in Section 2.3.

a. Playing soccer
b. Knitting a sweater
c. Bidding on an item at an auction

Solution:

PEAS description of the task environments

Agent Type	Performance Measure	Environment	Actuators	Sensors
Playing soccer	Score goals and defend well against the opponent	Field, opponent, ball, Weather Conditions, Audience	Kicking, throwing, dribbling, tackling, passing	cameras, ball sensor, location sensors, other players locator
Knitting a sweater	Comfort, good-fit, durability, look	Recipient's body	Needles, yarn, jointed-arms	Pattern sensor
Bidding on an item at an auction	cost, value, durability, uniqueness, necessity, quality	auctioneer, items, other bidders	speaker, display items	camera, price monitor

Characteristics of the task environments

Agent Type	Observable /Not	Deterministic/ Stochastic	Episodic/ Sequential	Static/Dynamic	Discrete/ Continuous	Single-agent/ Multi-agent
Playing soccer	Yes	Stochastic	Sequential	Dynamic	Continuous	Multi
Knitting a sweater	Yes	Deterministic	Sequential	Static	Continuous	Single

Bidding on an item at an auction	Partially	Stochastic	Sequential	Dynamic	Continuous	Single

---

2. To what extent are the following computer systems instances of artificial intelligence:
• Supermarket bar code scanners.
• Web search engines.
• Voice-activated telephone menus.
• Spelling and grammar correction feature in word processing programs.
• Internet routing algorithms that respond dynamically to the state of the network.

Solutions

★ **Supermarket bar code scanners are not AI.** This is an example of a lookup, that commonly uses efficient Database Management frameworks. No learning is required.
★ **Web search engines are AI.** In order to build good web search engines, we need to use deep learning algorithms for Natural Language Processing, Retrieval, and recommendations.
★ **Voice-activated telephone menus are AI.** In order to build Voice-activated telephone menus, we need robust and generalized Automatic Speech Recognition (ASR) units that typically use deep learning algorithms nowadays.
★ **Spelling and grammar correction features in word processing programs are AI.** Researchers use learning algorithms to go beyond what non-AI spell-checkers can provide. They can not only identify spelling mistakes but also identify correctly spelled words used in the wrong contexts. (e.g. Grammarly, which is trained using large test corpus)
★ **Internet routing algorithms that respond dynamically to the state of the network** are AI because they are capable of adapting to new situations and learning from previous states.

---

3. Examine the AI literature to discover whether the following tasks can currently be solved by computers:

a. Playing a decent game of table tennis (Ping-Pong). [Yes]

b. Driving in the center of Cairo, Egypt. [No]
c. Driving in Victorville, California. [Maybe]
d. Buying a week's worth of groceries at the market. [No]
e. Buying a week's worth of groceries on the Web. [Yes]
f. Playing a decent game of bridge at a competitive level. [Yes]
g. Discovering and proving new mathematical theorems. [No]
h. Writing an intentionally funny story. [Currently no, Maybe in a few years]
i. Giving competent legal advice in a specialized area of law. [Yes]
j. Translating spoken English into spoken Swedish in real-time. [Yes]
k. Performing a complex surgical operation. [No]

Explanations for the currently infeasible tasks

★ **Driving in the center of Cairo, Egypt is infeasible** because of heavy traffic in Cairo. Egypt is even considering building a new capital city to reduce Cairo's heavy traffic [For those who are fascinated about geography - https://youtu.be/9-ThusbaRW8]. Self-driving cars by Waymo are currently in action in San Francisco. Hopefully, we will be able to use Self-driving cars all around the world in near future.

★ **Buying a week's worth of groceries at the market is infeasible** for computers right now because the computer would have to know what it (or you) wants, it would have to be able to identify foods without bar codes such as apples.

★ **Discovering and proving new mathematical theorems would be a challenge** for computers right now. Computers can solve mathematical theorems, but discovering them is a whole different story. A computer would have to be self-aware to discover anything.

★ **Writing an intentionally funny story would be a challenge for computers right now.** Detecting subtle emotional states like sarcasm and humor is a challenge for AI algorithms. Thus, generating such humorous long natural language is incredibly tough. You can input jokes into a computer, but it won't know how to write a funny story with new material.

★ **Performing a complex surgical operation is a challenge for computers right now** because the task requires high domain knowledge which doctors gain after years of practice.

---

4. The missionaries and cannibals problem is usually stated as follows. Three missionaries and three cannibals are on one side of a river, along with a boat that

can hold one or two people. Find a way to get everyone to the other side without ever leaving a group of missionaries in one place outnumbered by the cannibals in that place.

A. **Formulate the problem precisely, making only those distinctions necessary to ensure a valid solution. Draw a diagram of the complete state space.**

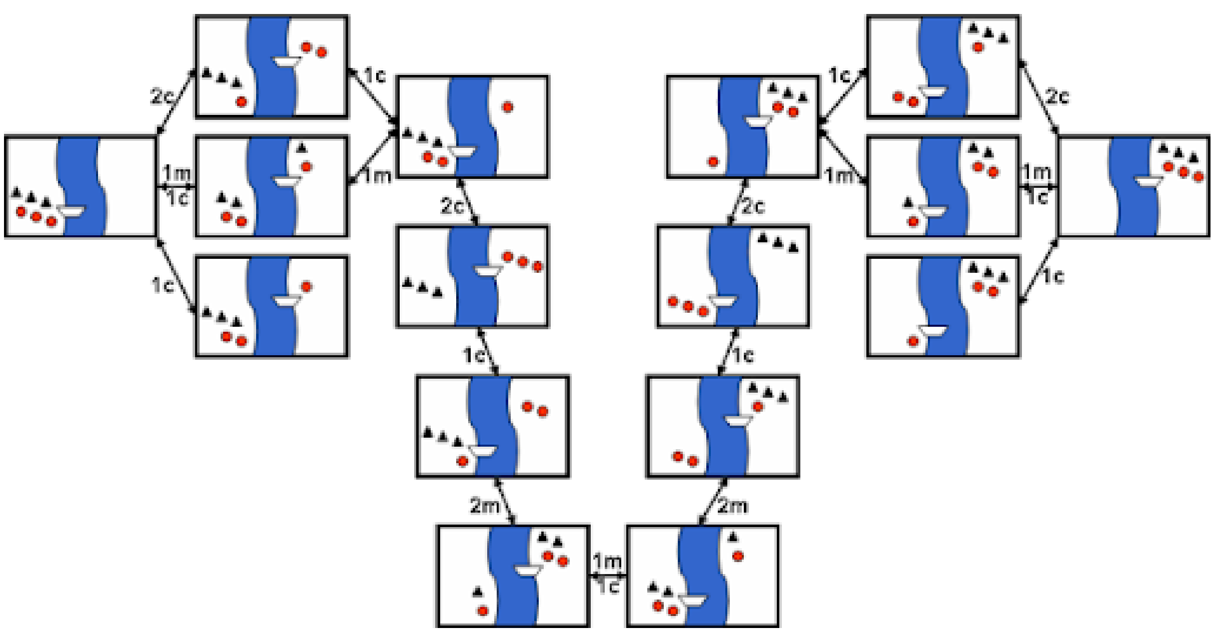
Solution:

The given data are

1. There are three missionaries and three cannibals on the left bank of a river.
2. They wish to cross over to the right bank using a boat that can only carry two at a time.
3. The number of cannibals on either bank must never exceed the number of missionaries on the same bank, otherwise, the missionaries will become the cannibals' dinner.
4. Plan a sequence of crossings that will take everyone safely across.

For solving this, we use a 3-tuple (m, c, b) to represent the state of one side of the river, since the other side can be easily inferred. Here, m stands for the number of missionaries, c, the number of cannibals, and b, whether the boat is at this side of the river. Initially, we have the state (3, 3, 1) and the goal state should be (0, 0, 0).

State Space Diagram:



Here, Actions are taken as
1m --- one missionary crosses the river

1c --- one cannibal crosses the river
2m ---- two missionaries cross the river
2c --- two cannibals cross the river
1m, 1c --- one missionary and one cannibal cross the river

B. **Implement and solve the problem optimally using an appropriate search algorithm. Is it a good idea to check for repeated states?**

Solution: Yes, it is a good idea for the repeated states to solve the problem optimally using an appropriate search algorithm. Here, all the links in the state space are bidirectional, indicating the graph is full of cycles. However, all the cycles correspond to the case where we undo an action immediately after it is performed.

C. **Why do you think people have a hard time solving this puzzle, given that the state space is so simple?**

Solution: Even though the state space is simple, people have a hard time solving this puzzle because most of the moves are either illegal or reverted to the previous state.

---

5. Which of the following are true and which are false? Explain your answers.

a. Depth-first search always expands at least as many nodes as A* search with an admissible heuristic.

FALSE. The depth-first search may possibly, sometimes by good luck, expand fewer nodes than A* search with an admissible heuristic. Sometimes it is logically possible that DFS may find the goal directly without back-tracking, but A* has to consider alternatives to ensure optimality.

b. h(n)=0 is an admissible heuristic for the 8-puzzle.

TRUE. h(n)=0 never overestimates the remaining optimal distance to the goal node.

c. A* is of no use in robotics because percepts, states, and actions are continuous.

FALSE. The continuous space can be abstracted into discrete states. For example, A* (or, its variants) are widely used for navigation.

d. Breadth-first search is complete even if zero-step costs are allowed.

TRUE. If there exists a targeted node that occurs at a finite depth d, it can be found in $O(b^d)$ steps. However, it may not be optimal. Complete only refers that 'will find a goal when one exists'.

e. Assume that a rook can move on a chessboard any number of squares in a straight line, vertically or horizontally, but cannot jump over other pieces. Manhattan distance is an admissible heuristic for the problem of moving the rook from square A to square B in the smallest number of moves.

FALSE. A rook can cross the board in one move, so Manhattan is pessimistic and hence not an admissible heuristic.

---

6. Prove each of the following statements or give a counterexample:

A. Breadth-first search is a special case of uniform-cost search.

TRUE. When all step costs are equal, $g(n) \propto depth(n)$, so uniform-cost search reproduces breadth-first search.

B. Depth-first search is a special case of best-first tree search.

TRUE. Depth-first search is the best-first search with $f(n) = - depth(n)$.

C. Uniform-cost search is a special case of A search.

TRUE. We know that, for A* search $f(n) = g(n) + h(n)$, and for uniform-cost search, $f(n) = g(n)$. Thus, for $h(n) = 0$, uniform-cost search will produce the same result as A* search.

---

7. Consider a best-first search algorithm in which the evaluation function is $f(n) = (2-w)*g(n) + w*h(n)$. For what values of w is this complete? For what values is it optimal, assuming that h is admissible? What kind of search does this perform for w=0, w=1, and w=2?

<u>Solutions:</u>

The algorithm is complete whenever $0 \leq w < 2$.

★ w = 0 gives f(n) = 2*g(n). This behaves exactly like uniform-cost search — the factor of two makes no difference in the ordering of the nodes.
★ w = 1 gives A* search.
★ w = 2 gives f(n) = 2*h(n), i.e., greedy best-first search.

If h(n) is admissible, the algorithm is guaranteed to be optimal. We also have f(n) = (2−w) * [g(n) + w/(2−w)*h(n)] which behaves exactly like A* search with a heuristic w/(2−w)*h(n). For w ≤ 1, this is always less than h(n) and hence admissible, provided h(n) is itself admissible. So, the condition for optimality is 0 ≤ w ≤ 1.

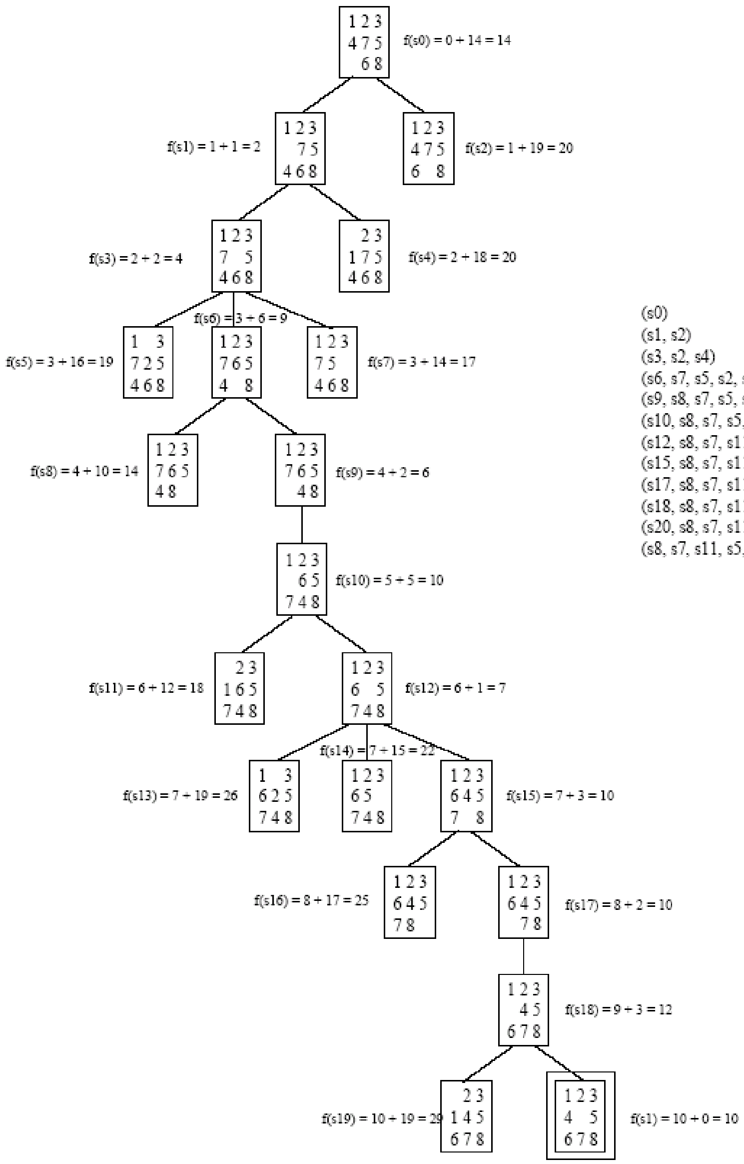
8.

A. Invent a heuristic function for the 8-puzzle that sometimes overestimates, and show how it can lead to a suboptimal solution on a particular problem. (You can use a computer to help if you want.)

Solutions: Consider a heuristic h(n) that returns 0 if the state is a goal and otherwise returns a random number between 1 and N where N is large enough to ensure that sometimes it overestimates. For the example below, let N=20 and suppose that we start with the following state

```
123
475
 68
```

The following A* trace shows how a sub-optimal solution would be found. Note that when the goal is found, all unexpanded nodes have an estimated cost greater than the actual cost of the goal that is found.



(s0)
(s1, s2)
(s3, s2, s4)
(s6, s7, s5, s2, s4)
(s9, s8, s7, s5, s2, s4)
(s10, s8, s7, s5, s2, s4)
(s12, s8, s7, s11, s5, s2, s4)
(s15, s8, s7, s11, s5, s2, s4, s14, s13)
(s17, s8, s7, s11, s5, s2, s4, s14, s16, s13)
(s18, s8, s7, s11, s5, s2, s4, s14, s16, s13)
(s20, s8, s7, s11, s5, s2, s4, s14, s16, s13, s19)
(s8, s7, s11, s5, s2, s4, s14, s16, s13, s19)

B. Prove that if h never overestimates the cost to reach the goal by more than a positive constant c, then A* search using h returns a solution whose cost does not exceed that of the minimum-cost path by more than c.

<u>Solutions:</u>

Suppose h(n) <= h*(n)+c as given and let G2 be a goal that is sub-optimal by more than c, i.e. f(G2) = g(G2) > C* +c. Now consider any node n on a path to an optimal goal. We have f(n) = g(n) + h(n) <= g(n) + h*(n) + c <= C* + c <= f(G2).
So, G2 will never be expanded before an optimal node is expanded because f(n) < f(G2).

9. Assume that there are 8 non-identical objects (as shown in the figure), and 9 locations. You can place any number of objects in any location or you can choose to keep a location empty. The objects are stackable in any order, and the gripper can also be placed in any location. Show that the possible number of combinations is in the order of billions.

Solutions:

At first, we have to place 8 non-identical objects in 9 locations.
For the time being, let's just forget about non-identical objects. We have to find out the number of ways to place 8 similar objects in 9 locations.

This is exactly the same as placing 8 balls in 9 buckets.

To solve this, we have just to 'divide' those 8 balls into 9 ways, which can be done by placing (9-1) = 8 dividers in between 8 balls. And, this is possible in (8+8)!/(8!*8!) = ${}^{16}C_8$ ways.

Now, since the balls/objects are non-identical, this number will be multiplied by 8!.

So, the total number of possible combinations to place 8 non-identical objects in 9 locations is 16!/8! = 518,918,400. On top of this, we can place the gripper in any of the 9 locations. So, our total number of possible combinations is 518,918,400*9 = 4,670,265,600, which is over 4.6B.