6.4110/16.420
Representation, Inference and Reasoning in AI

Final Exam

Solutions

December 19, 2022

Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, use the margins.

You are permitted to use two sheets of paper with notes on (both sides), and a calculator and a timer. If you use your phone for the calculator and timer, please restrict yourself to these functions.

Name: ______________________________

MIT email: ______________________________

Question	Points	Score
1	10	
2	10	
3	12	
4	10	
5	18	
6	10	
7	10	
8	20	
Total:	100	

1 Logical Inference

1. *Note: Nashville has a copy of the Parthenon!*

   (a) (3 points) Consider the following axioms:

   1. Everyone who has been in Nashville has been at a Taylor Swift concert.
   2. If you have been in Nashville or in Athens, you have seen the Parthenon.
   3. There is a tourist who has been in Nashville or in Athens

   Formulate the axioms above in first-order logic using the predicates $\texttt{InAthens}(x)$, $\texttt{InNashville}(x)$, $\texttt{Tourist}(x)$, $\texttt{AtTaylorSwift}(x)$, and $\texttt{SeenParthenon}(x)$. Feel free to use the abbreviations IA, IN, T, ATS, and SP for the predicates.

   Solution:

   - $\forall x.\texttt{InNashville}(x) \Rightarrow \texttt{AtTaylorSwift}(x)$.
   - $\forall x.(\texttt{InAthens}(x) \vee \texttt{InNashville}(x)) \Rightarrow \texttt{SeenParthenon}(x)$.
   - $\exists x.\texttt{Tourist}(x) \wedge (\texttt{InAthens}(x) \vee \texttt{InNashville}(x))$.

(b) (5 points) Given these axioms, already converted to clausal form:

1. $(\neg \texttt{InAthens}(x) \vee \texttt{SeenParthenon}(x))$
2. $(\neg \texttt{InNashville}(y) \vee \texttt{SeenParthenon}(y))$
3. $(\neg \texttt{Tourist}(z) \vee \texttt{InNashville}(z) \vee \texttt{InAthens}(z))$
4. $(\neg \texttt{Tourist}(z) \vee \neg \texttt{InNashville}(z) \vee \neg \texttt{InAthens}(z))$
5. $(\neg \texttt{AtTaylorSwift}(z) \vee \texttt{Tourist}(z))$
6. $\texttt{AtTaylorSwift}(Tomas)$

Prove that Tomas has seen the Parthenon, using resolution.

Solution:

- Suppose $\neg$`SeenParthenon`(Tomas).
- From (6) and (5), we conclude (7) `Tourist`(Tomas).
- From (7) and (3) we conclude (8) `InAthens`(Tomas) $\vee$ `InNashville`(Tomas).
- From (8) and (1) we conclude (9) `InNashville`(Tomas) $\vee$ `SeenParthenon`(Tomas).
- From (9) and (2) we conclude (10) `SeenParthenon`(Tomas).
- From (6) and (10) we conclude False. (Okay to omit this step.) □

(c) (2 points) If we did not assert `AtTaylorSwift`(Tomas), but instead asserted `SeenParthenon`(Tomas), could we prove `AtTaylorSwift`(Tomas) using resolution? If yes, please show using resolution. If not, please explain why not.

Solution: Note that $\neg$`AtTaylorSwift`(Tomas) does not resolve with any of the axioms. We have no way of concluding that someone is at a Taylor Swift concert.

2 Search: PDDL and Heuristics

2. Let's do something like Towers of Hanoi in PDDL, but with arbitrarily many discs and pegs. The goal is generally to move a stack of disks from one peg to another, with the constraint that one never puts a bigger disk on top of a smaller disk.

   We will declare every disk to be smaller than every peg (which will allow us to put any disk onto an empty peg).

   Here are the predicates.

   - `(clear ?x)` : there is no other disk on top of `?x`, which can be a disk or an empty peg
   - `(on ?x ?y)` : disk `?x` is resting directly on `?y`, which can be a disk or a peg
   - `(smaller ?x ?y)` : disk `?x` is smaller than `?y`, which can be a disk or a peg

   (a) (4 points) We only need one operator. What are the effects?

```
(define (domain hanoi)
  (:predicates (clear ?x) (on ?x ?y) (smaller ?x ?y))

  (:action move
    :parameters (?disc ?from ?to)
    :precondition (and (smaller ?disc ?to) (on ?disc ?from)
       (clear ?disc) (clear ?to))
    :effect
```

Solution:

```
(and (clear ?from)
     (on ?disc ?to)
     (not (on ?disc ?from))
     (not (clear ?to))))
  )
```

   (b) (2 points) Given an initial state

```
(on d3 peg1)
(on d2 d3)
(on d1 d2)
(clear peg2)
(clear peg3)
(smaller d1 d2)
(smaller d1 d3)
(smaller d2 d3)
(smaller d1 peg1) (smaller d2 peg1) (smaller d3 peg1)
(smaller d1 peg2) (smaller d2 peg2) (smaller d3 peg2)
(smaller d1 peg3) (smaller d2 peg3) (smaller d3 peg3)
```

   At what depth in the Relaxed Planning Graph (RPG) do all the fluents in this goal first appear?
   `(and (on d3 peg3) (on d2 d3) (on d1 d2))`

Solution: They all appear at level 3 after the following actions:

```
move(d1 peg1 peg2)
move(d2 peg1 peg2)  ; (clear peg2) is still there...
move(d3 peg1 peg3)
```

Note that `(on d2 d3) (on d1 d2)` remain true, since nothing is ever removed in the RPG.

(c) (2 points) You are considering making a relaxed version of the problem for the purposes of a heuristic (without using the relaxed planning graph). Your idea is to do it by dropping one precondition from the operator. For each of the preconditions below, indicate whether, if you were to delete it from the operator and use the length of the solution to the resulting planning problem as a heuristic, the heuristic would be admissible. Explain your answers.

i. `(smaller ?disc ?to)`

> **Solution:** It is admissible, since this is a strict relaxation.

ii. `(clear ?disc)`

> **Solution:** It is admissible, since this is a strict relaxation.

(d) (1 point) Which of the two "precondition dropping relaxations" above would produce smaller heuristic values? Assume all the discs start out in a valid configuration on one peg. Explain.

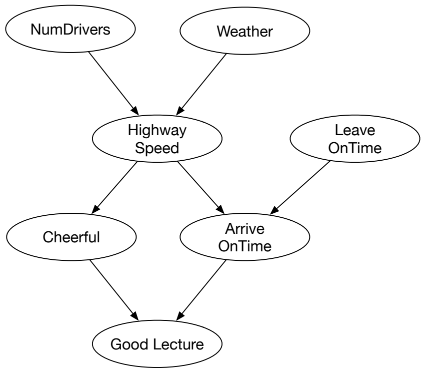
> **Solution:** `(clear ?disc)`. Since it allows us to move discs in any order, one can always move n discs in n steps.

(e) (1 point) Given two admissible heuristics h_1 and h_2 such that for all s, $h_1(s) \leq h_2(s)$, and for some state s, $h_1(s) < h_2(s)$, which one would we generally prefer to use in A* search? Explain.

> **Solution:** h_2 : generally we want to have values closer to the true values, which will be the larger values if the heuristics are admissible.

3 Traffic

3. Consider the following Bayesian network, which is about my morning commute.



The intended meaning of the state variables is:

- **NumDrivers**(N) : how many drivers are trying to make the same trip I am this morning (discretized into 4 values)
- **Weather**(W) : is it clear, raining, or snowy?
- **HighwaySpeed**(H) : the average speed of the cars on my highway (discretized into 4 values)
- **LeaveOnTime**(L) : whether I left my house on time
- **ArriveOnTime**(A) : whether I arrive at my office on time for my 9:30AM class
- **Cheerful**(C) : whether I am happy and cheerful when I get into my office
- **GoodLecture**(G) : whether I deliver a good lecture

(a) (4 points) Draw the factor graph corresponding to this Bayesian network.

Solution:

(b) (1 point) Could I use belief propagation to exactly compute the marginal distributions on **Weather** given **GoodLecture**? ◯ Yes √ **No**

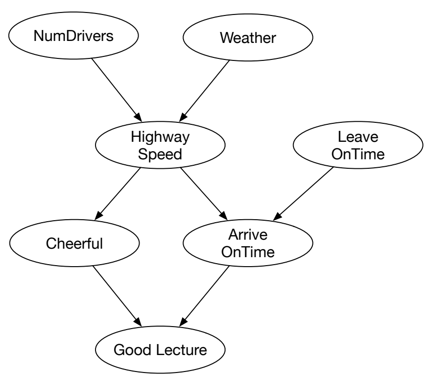
(c) (1 point) Could I use belief propagation to exactly compute the marginal distributions on **Weather** given that I'm **Cheerful** and **ArriveOnTime**? √ **Yes** ◯ No

(d) (2 points) We are interested in using sampling to estimate the marginal distribution of **Weather** given the *extremely rare* event that **GoodLecture** = **False**. Rejection sampling would be inefficient in this case. Explain why.

> **Solution:** Most of the samples would be rejected.

(e) (2 points) Name and briefly describe a sampling method that is likely to be more efficient.

> **Solution:** Gibbs sampling lets us draw samples from the conditional distribution by fixing the observed nodes and repeatedly sampling from the distribution of each node conditioned on the values of the rest of the nodes in the network.



(f) (2 points) Add each of the following variables to the Bayesian network above with appropriate arrows.

i. **Month**(M) : What month are we in?

> **Solution:** As a parent of **Weather**

ii. **Road surface**(R) : are they doing road-surfacing work on my highway today?

> **Solution:** As a parent of **HighwaySpeed**.

4 Sequential estimation

4. *These problems should be solvable intuitively. There is no need to show work.* Consider an HMM, with two states, a and b, and two observations, $\triangle$ and $\square$.

(a) (4 points) We will first consider a case where the state doesn't change

$$P(S_t = s \mid S_{t-1} = s) = 1$$

the observation probabilities are

$$P(O_t = \triangle \mid S_t = a) = P(O_t = \square \mid S_t = b) = .9$$

and the initial state distribution is $P(S_0 = a) = .05$.
To help get started on the questions below, we note that $P(S_0 = a \mid O_0 = \triangle) = 0.32$.

i. What is the most likely state at time 0 given $O_0 = \triangle$?

Solution: b

ii. What is the most likely state at time 1 given that $O_0 = \triangle$ and $O_1 = \triangle$?

Solution: a

iii. What is the most likely state sequence S_0, S_1, S_2 given $O_0 = \triangle, O_1 = \triangle, O_2 = \triangle$?

Solution: a, a, a

(b) (4 points) Now, what if we just change the transition model to be

$$P(S_t = s' \mid S_{t-1} = s) = 0.5$$

so that the new state at every time step is just a coin-flip.

i. What is the most likely state at time 0 given $O_0 = \triangle$?

Solution: b

ii. What is the most likely state at time 1 given that $O_0 = \triangle$ and $O_1 = \triangle$?

Solution: a

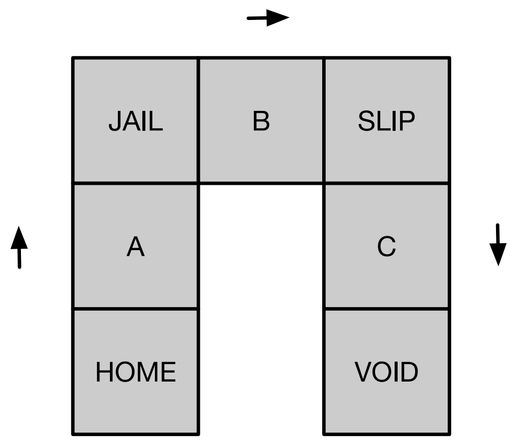
iii. What is the most likely state sequence S_0, S_1, S_2 given $O_0 = \triangle, O_1 = \triangle, O_2 = \triangle$?

Solution: b, a, a

(c) (2 points) When using a Kalman filter, if we want to recover the maximum likelihood trajectory, can we discard the history of observations and just record the most likely states while filtering?
$\surd$ **Yes** ◯ No

5 Monotony

5. Here's a terrible board game.



- You start in **HOME** and the game ends when you reach **VOID**.
- It is very boring and you have no choices to make.
- You move clockwise around the board. In **HOME** and the squares A, B, and C, you always just move to the next square.
- In **JAIL**, with probability .5 you stay in **JAIL** and with probability .5 you move to **B**.
- In **SLIP**, with probability .5 you go to **HOME** and with probability .5 you move to **C**.
- You get reward -1 on every step until your miserable existence ends in **VOID**.

(a) (5 points) Write a system of 4 equations in 4 unknowns relating the undiscounted ($\gamma = 1.0$) values of **HOME**, **JAIL**, **SLIP** and **VOID**.

Solution:

$$\begin{aligned} V_H &= -2 + V_J \\ V_J &= -1 + .5(-1 + V_S) + .5V_J \\ V_S &= -1 + .5(-1 + V_V) + .5V_H \\ V_V &= 0 \end{aligned}$$

Now, imagine that you can choose actions!

- In any state, you can select the *escape* action at a reward of -3, or do the normal action described above (which we will call *go*).
- In the **JAIL** state, if you select the *escape* then you move to the right with probability 1. in the **SLIP** state, if you *escape*, then you move down with probability 1.

(b) (5 points) Below are 10 iterations of Q-value iteration, showing the Q values for the *go* (G) action in all states and for the *escape* (E) action in **JAIL** and **SLIP**. One of the values is missing.

Note that we omitted columns for `Q(H,E)`, `Q(A, E)`, `Q(B, E)`, *and* `Q(C, E)` *because in those states* `E` *has the same effect as* `G` *and costs more, so it clearly isn't worth choosing.*

Q(H, G)	Q(A, G)	Q(J, G)	Q(J, E)	Q(B, G)	Q(S, G)	Q(S, E)	Q(C, G)
-1	-1	-1	-3	-1	-1	-3	-1
-2	-2	-2	-4	-2	-2	-4	-1
-3	-3	-3	-5	-3	-2.5	-4	-1
-4	-4	-4	-6	-3.5	-3.0	-4	-1
-5	-5	XXX	-6.5	-4.0	-3.5	-4	-1
-6	-5.75	-5.375	-7.0	-4.5	-4.0	-4	-1
-6.75	-6.375	-5.9375	-7.5	-5.0	-4.5	-4	-1
-7.375	-6.9375	-6.4687	-8.0	-5.0	-4.875	-4	-1
-7.9375	-7.46875	-6.7344	-8.0	-5.0	-5.1875	-4	-1
-8.4687	-7.73437	-6.8672	-8.0	-5.0	-5.4687	-4	-1

Compute the missing value (XXX), showing your work in detail!

Solution:

$$
\begin{aligned}
Q_3(B, E) &= -3 + \max(Q_3(S, G), Q_3(S, E) = -3 + \max(-3, -4) = -6 \\
V_2(B) &= \max(Q_3(B, G), Q_3(B, E)) = \max(-3.5, -6) = -3.5 \\
V_3(J) &= \max(Q_3(J, G), Q_3(J, E)) = \max(-4, -6) = -4 \\
Q_4(J, G) &= 0.5(-1 + V_3(J)) + 0.5(-1 + V_3(B)) = 0.5(-1 - 4) + 0.5(-1 - 3.5) = 4.75
\end{aligned}
$$

(c) (2 points) What is the optimal policy?

Solution: *escape* from **SLIP**, but just *go* in **JAIL**.

Time for capitalism! We will assume the same transitions and termination condition, but change the actions and rewards. Instead of *escaping* the void, we will try to make money. The squares A, B, and C start out as empty lots, but if you visit one and it's empty, you can do the action *invest*, which has reward -5, and changes the state of that square to a rental property. If you land on a square with a rental property, you get reward $+10$. All other actions have reward 0.

(d) (3 points) Is it necessary to add discounting to this problem for the values to be well defined? Why or why not?

> **Solution:** No, because you end up in the **VOID** with probability 1 and the game ends, so all values are finite even without discounting.

(e) (3 points) What is the state space for this new problem?

> **Solution:** $\{$Home, A, Jail, B, Slip, C, Void$\} \times \{$A-rented, A-empty$\} \times \{$B-rented, B-empty$\} \times \{$C-rented, C-empty$\}$

6 Frogger

6. You are a small frog, in a 1D world, with states s in the interval $[0, 1]$. You can take one of four possible actions:

   - *big right (BR)* which moves you to a new location drawn from a Gaussian distribution with mean $s + .2$ and standard deviation 0.1.
   - *small right (SR)* which moves you to a new location drawn from a Gaussian distribution with mean $s + .1$ and standard deviation 0.01.
   - *big left (BL)* which moves you to a new location drawn from a Gaussian distribution with mean $s - .2$ and standard deviation 0.1.
   - *small left (SL)* which moves you to a new location drawn from a Gaussian distribution with mean $s - .1$ and standard deviation 0.01.

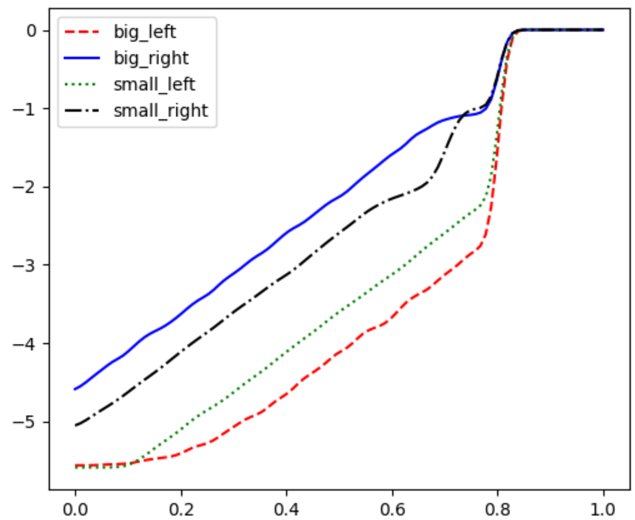
   However, any action result that would move you to a state less than 0.0 instead puts you at 0.0 and any action result that would move to a state greater than 1.0 instead puts you at 1.0.

   If you are in the *goal region* then any action yields a reward of 0 and the game terminates. All other actions have reward -1. The goal region will have slightly different definitions in the parts below.

   There is no discounting ($\gamma = 1.0$)

(a) (4 points) Easy Frogger: the *goal region* is $[0.8, 1.0]$.

In the figure below, the x axis corresponds to the state s and the y axis corresponds to the expected optimal value. We have plotted $Q(s, \text{BR})$, $Q(s, \text{SR})$, $Q(s, \text{BL})$, and $Q(s, \text{SL})$.



For each of the following questions, give an intuitive explanation, you do not have to compute the values exactly.

i. Explain why $Q(s = 0.4, \text{BR})$ has the value it has. In particular, why is it less than -2.?

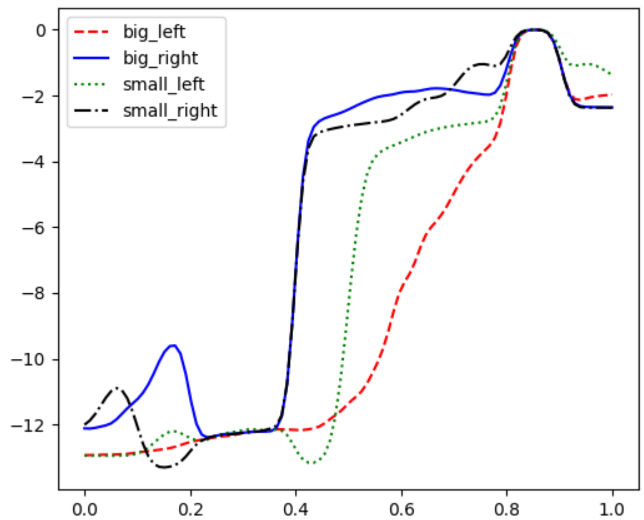
Solution: Because of variance in the outcome of BR, two steps is not guaranteed to reach the goal.

ii. Explain why $Q(s = 0.4, \text{SR})$ has the value it has. In particular, explain its relationship to $Q(s = 0.4, \text{BR})$.

Solution: Remember that the Q value of an action is the value from taking that action and then acting optimally. So, even theough SR is not optimal, we can recover quickly via the optimal policy.

Hard Frogger: the *goal region* is now $[0.8, 0.9]$, and we add a region of mud $[.2, .4]$. Inside the muddy region, no matter what action you select, the transition will be *as if* you had selected action BL (slide backwards).

The Q value functions are shown below.



(b) (3 points) Specify the optimal policy that would be derived from these Q functions. You can write it as a math expression or an informal piece of code. Try to find the simplest form of the policy; don't worry about getting the numeric parts exactly right.

> **Solution:** Approximate transitions. There are lots of ties.
> ```
> if 0.00 < s < 0.10: SR
> if 0.10 < s < 0.75: BR
> if 0.50 < s < 0.90: SR
> if 0.90 < s < 1.00: SL
> ```

(c) (3 points) Provide an explanation in words for why the best action choice at .75 is what it is; why is it better than the second-best action choice?

> **Solution:** You want to land in the goal region, which goes from .8 to .9. If you take a big step to the right, you move .2 in expectation so you are likely to overshoot. So, it's better to just move .1.

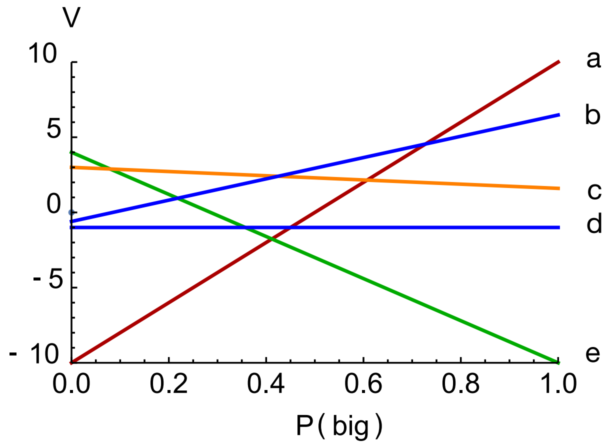
7 Truth in advertising

7. You are selling beautiful river stones as door-stops. Your e-commerce software is bad, though, and only lets you sell two different products, a *big stone* and a *small stone*. You are in a hurry because of the holiday season, and you need to go through your stone collection, decide how to label each one, and package them up for shipping.

   You will model the process, for a single stone, as a POMDP, with

   - States: $\{\text{big-stone}, \text{small-stone}\}$
   - Actions:
     - ship-big: label this stone as a big stone and ship it
     - ship-small: label this stone as a small stone and ship it
     - make-small: grind the stone down to try to make it smaller
     - inspect: measure the stone to determine its size
   - Transitions:
     - inspect does not change the state of the stone
     - ship-big and ship-small do not change the state of the stone, but do terminate the game
     - make-small: if the stone was originally big, then it becomes small with probability 0.9 and stays big with probability 0.1; if it was originally small, it stays small.
   - Observations
     - There are three possible observations: looks-big, looks-small, and hazy.
     - All actions besides inspect generate observation hazy with probability 1.
     - The inspect action, on a big stone generates observations with distribution: $\{\text{looks-big} : 0.7, \text{looks-small} : 0.1, \text{hazy} : 0.2\}$
     - The inspect action, on a small stone generates observations with distribution: $\{\text{looks-big} : 0.2, \text{looks-small} : 0.5, \text{hazy} : 0.3\}$
   - Rewards
     - Rewards for actions inspect and make-small are -1 in all states
     - Reward for action ship-small is -10 if the stone is big and 4 if the stone is actually small.
     - Reward for action ship-big is $+10$ if the stone is big and -10 if the stone is actually small.
   - Discount factor $\gamma = 1$

This figure indicates α vectors for several simple policy trees.



For each policy tree below, indicate which vector it corresponds to:

(a) (1 point) ship-big
◯ a ◯ b ◯ c ◯ d √ **e**

(b) (1 point) ship-small
√ **a** ◯ b ◯ c ◯ d ◯ e

(c) (1 point) inspect
◯ a ◯ b ◯ c √ **d** ◯ e

(d) (1 point) make-small then ship-small
◯ a ◯ b √ **c** ◯ d ◯ e

(e) (1 point) inspect then
if $o =$ looks-big then ship-big;
if $o =$ looks-small then ship-small;
otherwise (make-small then ship-small).
◯ a √ **b** ◯ c ◯ d ◯ e

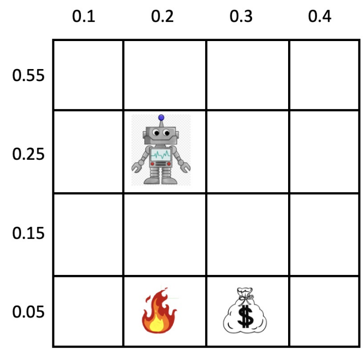
(f) (4 points) Justify your choice for part (e) above by computing the value for $P(big) = 1.0$.

Solution: Compute the reward for the $P(big) = 1$ case.

(g) (1 point) Which policy tree is never helpful? Indicate its associated α vector.
◯ a ◯ b ◯ c √ **d** ◯ e

8 Flames or fortune

8. Consider the following gridworld containing a robot, a large positive reward (the money, worth +1000 points) and a large negative reward (the flames, worth -1000, points). The robot is trying to obtain the money and avoid the flames.



The robot's orientation is fixed, but it can move up, down, left and right, and each action that does not result in fortune or flame costs -1 points. However, it cannot directly observe its (x, y) position — it can only observe whether there are any walls next to it in the four compass directions. For instance, if the robot were in the top right corner, it would see a wall to the north and to the east. It would also know unambiguously that it is in that cell — no other cells have a wall to the north and to the east.

There are only walls around the boundary of the gridworld — there are no internal walls. The problem ends when the robot obtains the money or is consumed by flames.

Let us also assume the robot's motion dynamics are perfect — when it attempts to move in a given direction (e.g., move up), the motion occurs deterministically, unless there is a wall in the way. Let us also assume that the robot's sensing is perfect — it always sees the walls it is adjacent to, and never sees walls that aren't there. Recall that on any given step, *the robot moves then observes*, so it will start with a motion.

Finally, assume the robot's (unobservable) start location is the one drawn in the figure, at $(1, 2)$. (We will use the x, y coordinate system convention, so the bottom-left corner is $(0, 0)$.)

The robot will have to select an action based on its initial belief, *then* it will receive an observation, etc.

(a) (3 points) For the purposes of this part, assume the robot starts out with an initial belief that is uniform over all 16 locations. How many distinct belief states are reachable, including the start belief state? (Recall that a belief state is a distribution over the underlying states.)

Solution: 22 belief states. 1 uniform belief, 1 uniform belief over the four internal squares, 4 beliefs where one of the coordinates is known exactly and the other coordinate is distributed uniformly between two of the edge cells, and 16 beliefs where both coordinates are known exactly.

20 belief states is also acceptable if we don't count the belief that results when the robot moves into the positive or negative reward squares and the game ends.

Now, assume that when the robot wakes up, it doesn't know where it is, but it does have a prior on its location indicated by the marginal distribution on the columns $(.1, .2, .3, .4)$ and on the rows $(.55, .25, .15, .05)$ as indicated in the figure.

(b) (2 points) What action would the *most likely state* method select initially?

> **Solution:** Down

(c) (4 points) What would the posterior belief state be after the robot (in its actual location as shown in the figure) executed a *down* action and got the observation corresponding to its resulting actual state?

> **Solution:**
> ```
> 0 0 0 0
> 0 x y 0
> 0 z w 0
> 0 0 0 0
> ```
> (x, y, z, w) = normalized (.11, .165, .05, .075)

(d) (2 points) What action would the *most likely state* method select from this belief?

> **Solution:** down

Now let's consider the *most likely observation* (MLO) strategy, in which we search in a determinized version of belief space, assuming that we will always get the most likely observation after each action.

In all the following questions, assume we start with belief b_0:

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & .3 & .4 & 0 \\ 0 & .1 & .2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

(e) (2 points) What is the most likely observation after we move *left*? Explain why.

> **Solution:** no walls in any direction, because the .6 of our probability mass will be in states that generate that observation

(f) (2 points) In the MLO belief-space search, what is the one-step reward for moving *down* from belief b_0?

> **Solution:** .1 * -1000 + .2 * 1000 + .7 * -1

(g) (3 points) In the MLO belief-space search, what is an optimal open-loop plan from b_0, and its approximate value (to the nearest 100)?

> **Solution:** left, right, down, down
>
> value approx 1000

(h) (2 points) Given the robot's *actual* initial state, will the robot end up executing the whole action sequence found in the previous step? Explain.

> **Solution:** No. On its first step it will get an observation that is not the MLO, so it will need to replan.