


# Assignment 2. MDP, Bayesian Network, and HMM


Gary Geunbae Lee  
CSED342-01 Artificial Intelligence

Contact: TA Heejin Do (heejindo@postech.ac.kr)

## General Instructions

This (and every) assignment has **written** and **programming** parts.

 This icon means a written answer is expected. There are no partial credits. Submit `name_studentID.pdf` containing your answers to the written part.

 This icon means you should write code. You can add other helper functions outside the answer block if you want. DO NOT make changes to files other than `submission.py`. You should write your answers in `submission.py` between

```
# BEGIN_YOUR_ANSWER
```

and

```
# END_YOUR_ANSWER
```

Your code will be evaluated on two types of test cases, **basic** and **hidden**, which you can see in `grader.py`. Basic tests, which are fully provided to you, do not stress your code with large inputs or tricky corner cases. Hidden tests are more complex and do stress your code. The inputs of hidden tests are provided in `grader.py`, but the correct outputs are not. To run all the tests, type

```
python grader.py
```

This will tell you only whether you passed the basic tests. On the hidden tests, the script will alert you if your code takes too long or crashes, but does not say whether you got the correct output. You can also run a single test (e.g., `1a-1-basic`) by typing

```
python grader.py 1a-1-basic
```

We strongly encourage you to read and understand the test cases, create your own test cases, and not just blindly run `grader.py`.

---



The search algorithms explored in the previous assignment are effective when you know exactly the results of your actions. Unfortunately, the real world is not so predictable. One of the key aspects of an effective AI is the ability to reason in the face of uncertainty.

Markov decision processes (MDPs) can be used to formalize uncertain situations. In this assignment, you will implement algorithms to find the optimal policy in these situations. You will then formalize a modified version of Blackjack as an MDP, and apply your algorithm to find the optimal policy.

## Problem 1. Blackjack and Value Iteration

You will be creating a MDP to describe a modified version of Blackjack. (Before reading the description of the task, first check how `util.ValueIteration.solve` finds the optimal policy of a given MDP such as `util.NumberLineMDP`.)

For our version of Blackjack, the deck can contain an arbitrary collection of cards with different values, each with a given multiplicity. For example, a standard deck would have card values  $[1, 2, \dots, 13]$  and multiplicity 4. You could also have a deck with card values  $[1, 5, 20]$ . The deck is shuffled (each permutation of the cards is equally likely).

The game occurs in a sequence of rounds. Each round, the player either (i) takes the next card from the top of the deck (costing nothing), (ii) peeks at the top card (costing `peekCost`, in which case the card will be drawn in the next round), or (iii) quits the game. (Note: it is not possible to peek twice in a row; if the player peeks twice in a row, then `succAndProbReward()` should return `[]`.)

The game continues until one of the following conditions becomes true:

- The player quits, in which case her reward is the sum of the cards in her hand.
- The player takes a card, and this leaves her with a sum that is strictly greater than the threshold, in which case her reward is 0.

- The deck runs out of cards, in which case it is as if she quits, and she gets a reward which is the sum of the cards in her hand.

In this problem, your state  $s$  will be represented as a triple:

`(totalCardValueInHand, nextCardIndexIfPeeked, deckCardCounts)`

As an example, assume the deck has card values  $[1, 2, 3]$  with multiplicity 1, and the threshold is 4. Initially, the player has no cards, so her total is 0; this corresponds to state `(0, None, (1, 1, 1))`. At this point, she can take, peek, or quit.

- If she takes, the three possible successor states (each has  $1/3$  probability) are

`(1, None, (0, 1, 1))`  
`(2, None, (1, 0, 1))`  
`(3, None, (1, 1, 0))`

She will receive reward 0 for reaching any of these states.

- If she instead peeks, the three possible successor states are

`(0, 0, (1, 1, 1))`  
`(0, 1, (1, 1, 1))`  
`(0, 2, (1, 1, 1))`

She will receive reward `-peekCost` to reach these states. From `(0, 0, (1, 1, 1))`, taking yields `(1, None, (0, 1, 1))` deterministically.

- If she quits, then the resulting state will be `(0, None, None)` (note setting the deck to `None` signifies the end of the game).

As another example, let's say her current state is `(3, None, (1, 1, 0))`.

- If she quits, the successor state will be `(3, None, None)`.
- If she takes, the successor states are `(3 + 1, None, (0, 1, 0))` or `(3 + 2, None, None)`. Note that in the second successor state, the deck is set to `None` to signify the game ended with a bust. You should also set the deck to `None` if the deck runs out of cards.

### Problem 1a [5 points]

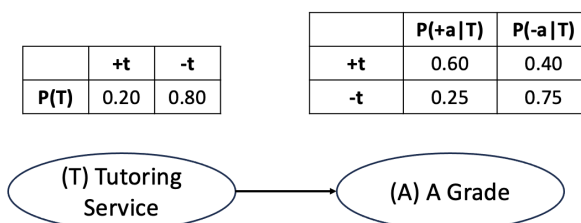
Your task is to implement the game of Blackjack as an MDP by filling out the `succAndProbReward()` function of class `BlackjackMDP`.

### Problem 1b [5 points]

Now, we'll implement a variation of value iteration specialized in acyclic MDPs. Your task is to implement `ValueIterationDP`, which should use dynamic programming to compute the optimal values. The time complexity of the algorithm should be linear to the number of all possible transitions in an acyclic MDP. The implemented algorithm should be faster than the normal value iteration.

## Problem 2: Bayesian Network

In research about student performance, students at a certain university may or may not have access to personalized Tutoring Services. Researchers believe that students who receive personalized Tutoring are more likely to excel academically (have an A grade) than those who do not, but exceptional performance can still occur among all students. Using the power of probability, the researchers model the situation with the Bayesian network below.



### Problem 2a [1.5 points]

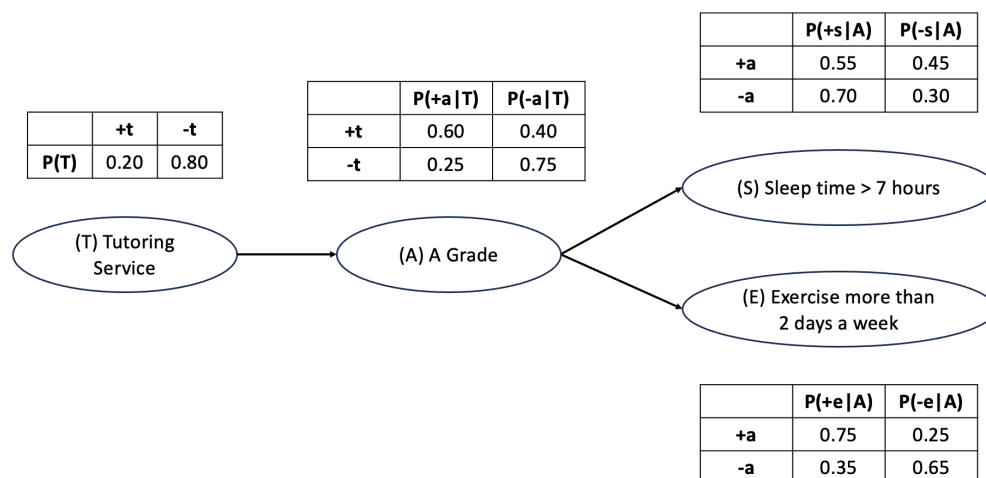
What is  $P(+a)$ , the marginal probability that a student gets an A grade?

### Problem 2b [1.5 points]

What is the conditional probability  $P(+t | +a)$ ?

### Problem 2c [2 points]

We can make better inferences if there is more evidence. We will expand on the Bayes net by introducing two new random variables: whether the student sleeps more than 7 hours a day (S), and whether the student exercises more than two days a week (E). The following figure shows the expanded Bayes net and conditional distributions.



Compute the following quantities.

$$(1) P(+t, +a, +s, +e) =$$

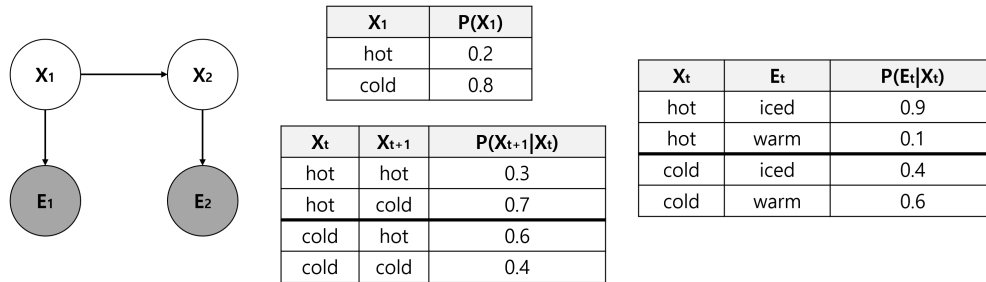
$$(2) P(+t, -a, +s, +e) =$$

$$(3) P(-s, +a, +t) =$$

$$(4) P(+s | +a, +t) =$$

### Problem 3: Hidden Markov Models

We want to estimate the weather (either hot or cold) based on whether Sophia drinks an **iced** or **warm** coffee. Based on the following Hidden Markov Model, and suppose that  $E_1 = \text{iced}$  (Sophia drinks iced coffee) and  $E_2 = \text{warm}$  (Sophia drinks warm coffee) are observed, answer the questions below.



#### Problem 3a [2 points]

Compute the probability distribution  $P(X_2, E_1 = \text{iced}, E_2 = \text{warm})$ , using the Forward algorithm. Show both the  $X_2 = \text{hot}$  and  $X_2 = \text{cold}$  cases.

#### Problem 3b [3 points]

Compute the maximum probability sequence  $X_1, X_2$  using the Viterbi algorithm. Tell the state of  $X_1, X_2$  in the maximum probability sequence.