# CMPSCI 687 Homework 1

Due September 20, 2018, 11pm Eastern Time

**Instructions:** This homework assignment consists of a written portion and a programming portion. Collaboration is not allowed on any part of this assignment. Submissions must be typed (hand written and scanned submissions will not be accepted). You must use LaTeX. The assignment should be submitted on Moodle as a .zip (.gz, .tar.gz, etc.) file containing your answers in a .pdf file and a folder with your source code. Include with your source code instructions for how to run your code. You may not use any reinforcement learning or machine learning specific libraries in your code (you may use libraries like C++ Eigen and numpy though). If you are unsure whether you can use a library, ask on Piazza. If you submit by September 25, you will not lose any credit. The automated system will not accept assignments after 11:55pm on September 25.

## Part One: Written (65 Points Total)

1. (Your grade will be a zero on this assignment if this question is not answered correctly) Read the class syllabus carefully, including the academic honesty policy. To affirm that you have read the syllabus, type your name as the answer to this problem.

   Elita Lobo

2. (15 Points) Given an MDP $M = (\mathcal{S}, \mathcal{A}, P, d_R, d_0, \gamma)$ and a fixed policy, $\pi$, the probability that the action at time $t = 0$ is $a \in \mathcal{A}$ is:

$$\Pr(A_0 = a) = \sum_{s \in \mathcal{S}} d_0(s)\pi(s, a). \tag{1}$$

   Write similar expressions (using only $\mathcal{S}, \mathcal{A}, P, R, d_0, \gamma$, and $\pi$) for the following:

   - The probability that the state at time $t = 3$ is either $s \in \mathcal{S}$ or $s' \in \mathcal{S}$.

$$\begin{aligned}\Pr(S_3 = s \cup S_3 = s') = &\sum_{s_0 \in \mathcal{S}} \sum_{a_0 \in \mathcal{A}} \sum_{S_1 \in \mathcal{S}} \sum_{a_1 \in \mathcal{A}} \sum_{S_2 \in \mathcal{S}} \sum_{A_2 \in \mathcal{A}} \sum_{s_3 \in \{s, s'\}} d_0(S_0 = s_0)\pi(S_0 = s_0, A_0 = a_0) \\ &* \mathcal{P}(S_0 = s_0, A_0 = a_0, S_1 = s_1)\pi(S_1 = s_1, A_1 = a_1) \\ &*\mathcal{P}(S_1 = s_1, A_1 = a_1, S_2 = s_2)\pi(S_2 = s_2, A_2 = a_2)\mathcal{P}(S_2 = s_2, A_2 = a_2, S_3 = s_3)\end{aligned} \tag{2}$$

   - The probability that the action at time $t = 16$ is $a' \in \mathcal{A}$ given that the action at time $t = 15$ is $a \in \mathcal{A}$ and the state at time $t = 14$ is

$s \in \mathcal{S}$.

$$\Pr(A_{16} = a' \mid A_{15} = a \cap S_{14} = s) = \frac{\mathcal{P}(A_{16} = a' \cap A_{15} = a \cap S_{14} = s)}{\mathcal{P}(A_{15} = a \cap S_{14} = s)}$$
$$= \sum_{a_{14} \in \mathcal{A}} \sum_{s_{15} \in \mathcal{S}} \sum_{s_{16} \in \mathcal{S}} \pi(S_{14} = s, A_{14} = a_{14}) \mathcal{P}(S_{14} = s, A_{14} = a_{14}, S_{15} = s_{15})$$
$$* \frac{\pi(S_{15} = s_{15}, A_{15} = a) \mathcal{P}(S_{15} = s_{15}, A_{15} = a, S_{16} = s_{16}) \pi(S_{16} = s_{16}, A_{16} = a')}{\sum \sum_{s_{15} \in \mathcal{S} a_{14} \in \mathcal{A}} \pi(S_{14} = s, A_{14} = a_{14}) \mathcal{P}(S_{14} = s, A_{14} = a_{14}, S_{15} = s_{15}) \pi(S_{15} = s_{15}, A_{15} = a)} \tag{3}$$

- The expected reward at time $t = 6$ given that the action at time $t = 3$ is $a \in \mathcal{A}$, and the state at time $t = 5$ is $s \in \mathcal{S}$.
$\mathbf{E}[\mathbf{R_6}|\mathbf{S_5} = \mathbf{s}] = \sum \sum \sum_{s_7 \in \mathcal{S} s_6 \in \mathcal{S} a_6 \in \mathcal{A}} \Pr(S_6 = s_6, A_6 = a_6, S_7 = s_7 \mid S_5 = s) \mathcal{R}(S_6 = s_6, A_6 = a_6, S_7 = s_7)$

$$= \sum_{a_5 \in \mathcal{A}} \sum_{s_6 \in \mathcal{S}} \sum_{a_6 \in \mathcal{A}} \sum_{s_7 \in \mathcal{S}} \pi(S_5 = s, A_5 = a_5) \mathcal{P}(S_5 = s, A_5 = a_5, S_6 = s_6)$$
$$\pi(S_6 = s_6, A_6 = a_6) \mathcal{P}(S_6 = s_6, A_6 = a_6, S_7 = s_7) \mathcal{R}(S_6 = s_6, A_6 = a_6, S_7 = s_7) \tag{4}$$

- The probability that the initial state was $s \in \mathcal{S}$ given that the state at time $t = 1$ is $s' \in \mathcal{S}$.

$$\Pr(S_0 = s \mid S_1 = s') = \frac{Pr(S_0 = s \cap S_1 = s')}{Pr(S_1 = s')}$$
$$= \frac{\sum_{A_0 \in \mathcal{A}} d_0(S_0 = s) \pi(S_0 = s, A_0 = a_0) \mathcal{P}(S_0 = s, A_0, S_1 = s)}{\sum \sum_{s_0 \in \mathcal{S} a_0 \in \mathcal{A}} d_0(S_0 = s_0) \pi(S_0 = s_0, A_0 = a_0) \mathcal{P}(S_0 = s_0, A_0 = a_0, S_1 = s)} \tag{5}$$

- The probability that the action at time $t = 5$ is $a \in \mathcal{A}$ given that the initial state is $s \in \mathcal{S}$, the state at time $t = 5$ is $s' \in \mathcal{S}$, and the action at time $t = 6$ is $a' \in \mathcal{A}$.

$$\Pr(A_5 = a \mid S_0 = s \cap S_5 = s' \cap A_6 = a') = \frac{\Pr(A_5 = a \cap S_5 = s' \cap A_6 = a')}{\Pr(S_5 = s' \cap A_6 = a')}$$
$$= \frac{\sum_{S_6 \in \mathcal{S}} \pi(S_5 = s', A_5 = a) \mathcal{P}(S_5 = s', A_5 = a, S_6 = s_6) \pi(S_6 = s_6, A_6 = a')}{\sum \sum_{a_5 \in \mathcal{A} s_6 \in \mathcal{S}} \pi(S_5 = s', A_5 = a_5) \mathcal{P}(S_5 = s', A_5 = a_5, S_6 = s_6) \pi(S_6 = s_6, A_6 = a')} \tag{6}$$

3. (3 Points) In 687-Gridworld, if we changed how rewards are generated so that hitting a wall (i.e., when the agent would enter an obstacle state, and is placed back where it started) results in a reward of $-1$, then what is

$\mathbf{E}[R_t|S_t = 17, A_t = \text{AL}, S_{t+1} = 17]$?

$$\begin{aligned}\mathbf{E}[R_t|S_t = 17, A_t = \text{AL}, S_{t+1} = 17] &= \Pr(S_t = 17, A_t = \text{AL}, \\ &\quad S_{t+1} = 17)R(S_t = 17, A_t = \text{AL}, S_{t+1} = 17) \\ &= (0.8/0.9) * (-10) + (0.1/(0.9) * 0) \ = -8.88888888 \end{aligned} \quad (7)$$

4. (2 Points) How many stochastic policies are there for an MDP with $|\mathcal{S}| < \infty$ and $|\mathcal{A}| < \infty$? (You may write your answer in terms of $|\mathcal{S}|$ and $|\mathcal{A}|$).

$$= A^S \quad (8)$$

5. (5 Points) Create an MDP (which may not have finite state or action sets) that does *not* have an optimal policy. The rewards for your MDP must be bounded.

   - State: Location of the robot in the 3X3 grid given by the coordinates of the robot $(row\_no, col\_no)$. The robot does not have any orientation.
   - Actions: Attempt Up (U), Attempt Down (D) , Attempt Left (L) , Attempt Right (R).
   - Environment Dynamics: The grid is a 3X3 grid.
     With probability 1.0 the robot moves in the specified direction from where it attempted to move.
     The robot starts from location $(0, 0)$ and the episode ends when the robot reaches location $(2, 2)$.
     Goal is the terminal state in this case.
   - Rewards:
     Following is the reward to state mapping.
     10 10 G
     10 -10 10
     R 10 10
     G = Goal (2,2)
     R - Robot (0,0)
     Each time the robot enters a state, it gets the reward associated with the state.

   - Since the total reward the robot can get is not bounded and the robot can keep circling around either of the corners to maximize its rewards (top-left corner and bottom-right corner), the MDP does not have an optimal policy.

6. (3 Points) Read about the Pendulum domain, described in Section 5.1 of this paper (Reinforcement Learning in Continuous Time and Space by Kenji Doya). Consider a variant where the initial state has the pendulum

hanging down with zero angular velocity always (a deterministic initial state where the pendulum is hanging straight down with no velocity) and a variant where the initial angle is chosen uniformly randomly in $[-\pi, \pi]$ and the initial velocity is zero. Which variant do you expect an agent to require more episodes to solve? Why?

- The variant where the pendulum is hanging down with zero velocity will require more episodes to solve as it would have higher number of reachable states as compared to the pendulum with initial angle chosen uniformly randomly. It would end up exploring a larger portion of the state-action space and would take more episodes to solve.

7. (1 Point) How many episodes do you expect an agent should need in order to find near-optimal policies for the gridworld and pendulum domains?

   - No of episodes required by the agent to find near optimal policies in pendulum domain will be greater than the number of episodes required by the agent in the gridworld domain because it has a larger state-action space.

8. (5 Points) Select a problem that we have not talked about in class, where the agent does not fully observe the state. Describe how this problem can be formulated as an MDP by specifying $(\mathcal{S}, \mathcal{A}, P, [d_r \text{ or } R], d_0, \gamma)$ (your specifications of these terms may use English rather than math, but be precise).

   - Consider an environment which consists of a house having two front doors ie- left door and right door and a tiger behind one of these doors.
     The states in this problem are $\{tiger - behind - left - door, tiger - behind - right - door\}$.
     At each time step, the agent should take one of these actions:
     $\{listen, open - left - door, open - right - door\}$.
     Probability of finding a tiger beyond the left door given that a roar was heard from the left door is 0.75.
     Probability of finding a tiger beyond the right door given that a roar was heard from the left door is 0.25.
     Similarly, probability of finding a tiger beyond the right door given that a roar was heard from right door is 0.75.
     Probability of finding a tiger beyond the left door given that a roar was heard from the right door is 0.25.
     These probabilities are a part of the state not perceived by the agent. The agent chooses its actions only based on the observations it makes. It tries to perceive the underlying state through the observations. The agent gets a reward of $-5$ for wrong opening (opening a door which has the tiger) and $+5$ for correct opening and $-1$ for listening.

Possible observations that the agent can make are: hear the tiger roar coming from the left door or hear the roar coming from the right door. Immediately after the agent opens a door and receives a reward or penalty, the problem resets, randomly relocating the tiger behind one of the two doors.

- States: $tiger - behind - left - door, tiger - behind - right - door$
- Actions:$listen, open - left - door, open - right - door$
- Rewards: $-50$ for wrong door opening, $+5$ for correct door opening and $-1$ to listen
- Observations: hear roar coming from left door, hear roar coming from right door
- Initial State distribution: $d_0(s) = 0.5 s \in \mathcal{S}$

9. (5 Points) Create an MDP for which there exist at least two optimal policies that have different variance of their returns. Describe the two optimal policies and derive the expected value and variance of their returns.

   - Consider a grid of size $1X3$ with a robot located on the second/middle cell. Cell 0 and cell 2 are the terminal states of the grid. On entering cell 1 , robot gets a reward of +10 whereas robot gets a reward 0.0 on entering/re-entering cell 2. However on entering cell 3, robot can get either a reward +8 or +12. Probability of getting +8 reward on entering cell3 is 0.5. Probability of getting +12 reward on entering cell 3 is +12.
     The robot can either move left L or right R. The probability of success of any action taken by the robot is 1.0.
   - States : Location of the robot in the grid. Start position of the robot is $(0, 1)$
   - Actions : $\{MoveleftL, MoverightR\}$
   - Rewards: +10 if the robot enters cell 1 $(0, 0)$ and either +8 or +12 if the robot enter cell 3 $(0, 2)$ ,each being equally probable.
   - Optimal policy 1: Always move left.
     Mean: 10
     Variance: 0
   - Optimal policy 2: Always move right
     Mean: 10
     Variance : 34.0

10. (2 Points) Create an MDP that always terminates, but which has no terminal states.

    - Consider a $4X4$ grid with water at $(4, 2)$.
    - State: Location of the robot

- Actions: Attempt Up, Attempt Down, Attempt Left, Attempt Right
- Transition Probability: The probability of success of any action is 0.85. The robot will stay at the same place with probability 0.15 which counts as re-entering the state.
  On entering/re-entering each state, there is a probability of 0.05 that the current state will transition the robot to terminal absorbing state or $S\_infinity$. (ie-Each cell/state has a bomb which will explode when the agent enters the state with 0.05 probability)
- Rewards: Agent gets a reward of -10 each time he enters or re-enters the water state and 0 otherwise. Each time the robot attempts to move outside the grid, it ends up re-entering the current state.
- Start position: $(0, 0)$
- Since each state has a probability of 0.05 of transitioning the robot to $S\_infinity$, this is an example of mdp which would terminate without a terminal state.

11. (2 Points) The sequence of states that results from running a fixed policy is a *Markov chain*. A state in a *Markov chain* has *period* $k$ if every return to the state must occur in multiples of $k$ time steps. More formally,

$$k = \gcd\{t > 0 : \Pr(S_t = s | S_0 = s) > 0\}.$$

Create an MDP and a policy that result in a state having a period of 3.

- Say we have a slot machine which dispenses money when the lever is pulled only at time steps which are multiples of 3 and if the lever was pulled 3 times since it last dispensed money.
  State here is defined by ($time\_step\%3, no\_of\_times\_lever\_was\_pulled\_since\_the\_machine\_last\_dispensed\_money\%$)
  Possible actions at any state are 1: Pulling the lever 2: Doing nothing.
  At any state $(t, n)$, pulling the lever would cause a transition to state $((t + 1)\%3, (n + 1)\%3)$ and doing nothing would cause the state to transition to state $((t + 1)\%3, n)$.
  Initial state is $(0, 0)$. By following a policy of pulling the lever at each time step, we can see that the lever returns to state(0,0) in multiple of 3 time steps.
- States: $(0, 0), (0, 1), (0, 2)(1, 0), (1, 1), (1, 2), (2, 0), (2, 1), (2, 2)$ . Initial state is $(0, 0)$
- Actions: 1: Pull lever 2. Do nothing
- Rewards: Gets $+10$ when entering/re-entering state $(0, 0)$ and $-1$ when agent chooses to do nothing.
- Policy: Pull lever at each time step.

12. (5 Points) A Markov chain is *irreducible* if it is possible to get to any state from any state. An MDP is *irreducible* if the Markov chain associated with

every deterministic policy is irredicible. A Markov chain is *aperiodic* if the period of every state is $k = 1$. The state of a Markov chain is *positive recurrent* if the expected time until the state recurs is finite. A Markov chain is *positive recurrent* if all states are positive recurrent. A Markov chain is *ergodic* if it is *aperiodic* and *positive recurrent*. An MDP is *ergodic* if the Markov chain associated with every deterministic policy is ergodic. Create an MDP that is ergodic, but *not* irreducible.

- Consider a 3X3 grid such that all cells other than corner cells are walls through which the agent cannot pass or enter.
  A state is defined by the location of the agent in the grid. There are 4 valid states in this grid $(0,0), (2,2), (0,2), (2,0)$.
  Agent can attempt to move UP, DOWN, LEFT or RIGHT.
  The agent will succeed in his action with probability 0.5 and will fail and stay in the same state with probability 0.5.
  Staying in a state counts as re-entering the state and the agent gets a reward of +1 each time the agent enters/re-enters a state.
  Hence in this scenario probability of moving from one valid state to another valid state is 0.0 ie- ($\mathcal{P}(s,a,s') = 0.0$ if $s \neq s'$) at any time step. The agent has no orientation
- States S : $(0,0), (0,2), (2,0), (2,2)$
- $d_0(s) = 0.25 s \in \mathcal{S}$
- Actions: $MoveUP, MoveDown, MoveLeft, MoveRight$
- Transition Probability: At any time step - ($\mathcal{P}(s,a,s') = 0.0$ if $s \neq s'$) $s \in \mathcal{S} a \in \mathcal{A}$
  ($\mathcal{P}(s,a,s) = 1.0$ $s \in \mathcal{S} a \in \mathcal{A}$
- Once the agent is in a state, it cannot reach any other state from the current state and the period of over state is 1. Hence this is an example of ergodic but not irreducible MDP.

13. (3 Points) Create an MDP that is not ergodic.

- Consider a grid of size $1X3$ with a robot located on the second/middle cell. Cell 0 and cell 2 are the terminal states of the grid. On entering cell 1 , robot gets a reward of +10 whereas robot gets a reward 0.0 on entering/re-entering cell 2. However on entering cell 3, robot gets a reward of +20. The robot is also penalized with a reward of -1 at each time step.
  The robot can either move left L or right R. The probability of success of any action taken by the robot is 1.0.
- State : Defined by location of the robot in the grid. Start position of the robot is $(0,1)$
- Actions : $\{MoveleftL, MoverightR\}$
- Rewards: +10 if the robot enters cell 1 $(0,0)$ and +20 if the robot enters cell 3 $(0,2)$.

- Irrespective of which deterministic policy is used, the robot is bound to move into any of the terminal states which will in turn transition it to S(infinity). State $(0, 1)$ in this case is not positive recurrent and hence this MDP is not ergodic.

14. (2 Points) Describe a real-world problem and how it can be reasonably modeled as an MDP where $R_t$ is *not* a deterministic function of $S_t, A_t$, and $S_{t+1}$.

    - Consider the case of an agent investing in stock market. State here is defined by the number of gold coins the agent has ie- $s(no\_of\_gold\_coins)$. MDP terminates when the agent cannot take any actions with the gold coins left.
      No of gold coins is randomly selected from the range $[1, 1000]$.
    - States: set of possible number of gold coins the agent can have. Let $s_i$ denote the state in which agent has i gold coins $1 \geq j \leq 1000$
    - Actions: Invest 100 gold coins , do not invest. Probability of succeeding while taking an action is 1.0. If the agent is at state $s_t$ and invests in stocks, he would reach state $s_(t - 100)$ and get a reward $R_t$.
    - Rewards: The agent gets a reward of $-10$ if he chooses not to invest and the current state is retained.
      Lets assume that the rewards for investing is received immediately after investing and these rewards are not in the form of gold coins so it does not get added to the gold coins the agent has.
      The reward at any time T is a stochastic function of the number of gold coins invested ie- There is a 0.25 probability that the reward is $2 * (no\_of\_gold\_coins\_invested)$ and 0.25 probability that the reward is 0 and 0.5 probability that the reward is equal to the number of gold coins invested. Hence the reward is a stochastic function of $A_t$ and is not a deterministic function of the current state $(s_t)$ , state reached after taking an action $(s_t + 1)$ and action $a_t$.

15. (2 Points) Describe a real-world problem and how it can be reasonably modeled as an MDP where $R_t$ is a deterministic function of $S_t$.

    - Say we have a car placed on the start of a long road of length L units. A state in this case is defined by distance covered by the car in terms of units and will be denoted by $s_x$ where $x$ is the distance covered by the car. $s_L$ is the terminal state of this MDP. $L = 10$
    - States: $s_0, s_1, s_2....s_L$
    - Actions: Move 1 unit forward, stay at the same location. Probability of success for any action is 1.0.
      If car at $s_x$ takes action - Move 1 unit forward, it will end up in state $s_{x+1}$

- Rewards: At any time t, if the car stays at the same location, the reward received is $-L$. if the car moves from state $S_t$ to $S_{t+1}$, the reward received is the ($distance\_covered\_by\_the\_car\_at\_state\_S_t$). An additional reward of +100 is received at the goal state.
- gamma=1
- Here the reward is a deterministic function of $S_t$. Intuition: When a car leaves a state , it receives reward for the distance covered so far.

16. (2 Points) Describe a real-world problem and how it can be reasonably modeled as an MDP where $R_t$ is a deterministic function of $S_{t+1}$.

    - Say we have a car placed on the start of a long road of length L units. A state in this case is defined by distance covered by the car in terms of units and will be denoted by $s_x$ where $x$ is the distance covered by the car. $s_L$ is the terminal state of this MDP. $L = 10$
    - States: $s_0, s_1, s_2....s_L$
    - Actions: Move 1 unit forward, stay at the same location. Probability of success for any action is 1.0.
      If car at $S_t$ takes action - Move 1 unit forward, it will end up in state $S_{t+1}$.
    - Rewards: At any time t, if the car stays at the same location, the reward received is $-L$. if the car moves from state $S_t$ to $S_{t+1}$, the reward received is the ($distance\_covered\_by\_the\_car\_at\_state\_S_{t+1}$). An additional reward of +100 is received at the goal state.
    - $\gamma = 1$
    - Here the reward is a deterministic function of $S_{t+1}$. Intuition: When a car enters a state , it receives reward for the distance covered so far.

17. (2 Points) Describe a real-world problem and how it can be reasonably modeled as an MDP where the reward function, $R$, would be known.

    - Say we have a agent in a car placed on the start of a long road of length L units. A state in this case is defined by distance covered by the car in terms of units and will be denoted by $s_x$ where $x$ is the distance covered by the car. $s_L$ is the terminal state of this MDP. $L = 10$
    - States: $\{s_0, s_1, s_2....s_L\}$
    - Actions: Move car 1 unit forward, Stop the car at the current location. Probability of success for any action is 1.0
    - Rewards: At any time t, if the car stays at the same location, the reward received is $-L$. if the car moves from state $S_t$ to $S_{t+1}$, the reward received is the ($distance\_covered\_by\_the\_car\_at\_state\_S_{t+1}$). An additional reward of +100 is received at the goal state. This reward scheme is known to the agent driving the car.

- $\gamma = 1$
- Here the reward is a deterministic function of $S_{t+1}$. Intuition: When a car enters a state , it receives reward for the distance covered so far. The reward function is known to the agent in this case.

18. (2 Points) Describe a real-world problem and how it can be reasonably modeled as an MDP where the reward function, $R$, would *not* be known.

    - Consider the case of an agent investing in stock market. State here is defined by the number of gold coins the agent has ie- $s(no\_of\_gold\_coins)$. MDP terminates when the agent cannot take any actions with the gold coins left.
      No of gold coins is randomly selected from the range $[1, 1000]$.
    - States: set of possible number of gold coins the agent can have. Let $s_i$ denote the state in which agent has i gold coins $1 \geq j \leq 1000$
    - Actions: Invest 100 gold coins , do not invest. Probability of succeeding while taking an action is 1.0. If the agent is at state $s_t$ and invests in stocks, he would reach state $s_(t - 100)$ and get a reward $R_t$.
    - Rewards: The agent gets a reward of -10 if he chooses not to invest and the current state is retained.
      Lets assume that the rewards for investing is received immediately after investing and these rewards are not in the form of gold coins so it does not get added to the gold coins the agent has.
      The rewards at any time t depend on the value of stocks in the market which fluctuates with time based on a number of external parameters as well as no of gold coins invested and is not known to the agent.

19. (2 Points) Describe a real-world problem and how it can be reasonably modeled as an MDP where the transition function, $P$, would be known.

    - Consider the case, where an agent has access to two slot machines - Machine A and Machine B. It is known to the agent that each time the lever of Machine A is pulled, it will dispense 2 gold coin with a probability 0.9 and each time the lever of Machine B is pulled, it will dispense 5 gold coins with probability 0.5. This information is encoded in the states's meta data. In this case, we have two possible states - $\{Success(S), Failure(F)\}$. Initial state is S. The actions available at any time t is $\{pull - lever - of - Machine - A, pull - lever - of - Machine - B\}$. Probability of success for any action is 1.0. At any time t, the agent can take one of the above mentioned actions. Current state transitions to state Success if , he receives gold coins from the machine he chose to play and it moves to state Failure otherwise. The agent receives a reward equal to the number of gold coins received when transitioning to Success state from either a failure state or success state and a reward of $-10$ on transitioning to failure state from either of the states. States: Success, Failure Actions: $\{pull - lever - of - Machine - A, pull - lever - of - Machine - B\}$

- In this case, the transition probability of moving from Success to Failure and vice versa via action $a_0/a_1$ is known.
- Rewards: The agent receives a reward equal to the number of gold coins received when transitioning to Success state from either a failure state or success state and a reward of $-10$ on transitioning to failure state from either of the states.

20. (2 Points) Describe a real-world problem and how it can be reasonably modeled as an MDP where the transition function, $P$, would *not* be known.

    - Consider a scenario where we have a robot and two boxes (box A and box B) filled with red and white balls.
      The distribution of red and white balls in both boxes may be different. At each time step t, robot is blinded and has to either pick a ball from box A or pick B with replacement. If it picks out a red ball, he enters Success state with probability 1.0 and gets a reward of +1 and if it picks a white ball, it enters Failure state with probability 1.0 and gets a reward of -2. If the robot is in Failure state and picks a white ball, he will re-enter Failure state and get a reward of -2.
      Similarly, if the robot is in success state and picks a red ball, it will re-enter success state and get a reward of +1.
    - States : $\{SuccessS, FailureF\}$ The robot is initially in state F.
    - Actions : $\{ChooseballfromBoxA, ChooseballfromBoxB\}$
    - Rewards: +1 if the robot picks a red ball (enters Success state) or -2 if he picks a white ball (enters Failure state)
    - In this case the transition probability of moving from Success state to Failure state on taking any of the available actions is not known ahead of time and can be deduced by the robot.

# Part Two: Programming (25 Points Total)

Implement the 687-Gridworld domain described in class and in the class notes. Have the agent select actions uniformly randomly.

- (5 Points) Have the agent uniformly randomly select actions. Run 10,000 episodes. Report the mean, standard deviation, maximum, and minimum of the observed discounted returns.
  Minimum discounted reward: -16.755427894066138
  Maximum discounted reward: 4.7829690000000005
  Mean discounted reward discounted reward: -0.3034602481733787
  Standard Deviation of discounted rewards: 1.315946608866517

- (5 Points) Find an optimal policy (you may do this any way you choose, including by reasoning through the problem yourself). Report the optimal policy here. Comment on whether it is unique.

- Optimal policy seems to be unique in this case

- Actions: $AR(AttemptRight)AL(AttemptLeft), AU(AttemptUP)AD(AttemptDown)$
- $Goal(G)Wall(W)$
- Optimal Policy:
  AR AR AR AR AD
  AR AR AR AR AD
  AU AU W AR AD
  AU AU W AR AD
  AU AL AR AR G
  Minimum discounted reward: 0.3433683820292515
  Maximum discount reward: 4.7829690000000000
  Mean discounted reward discounted reward: 3.9212725547333545
  Standard Deviation of discounted rewards: 0.6931065667160492
- Policy 2 which gives almost the same mean as optimal policy on running 10000000 trials for both policies.
  AR AR AR AR AD
  AR AR AR AR AD
  AR AU W AR AD
  AR AU W AR AD
  AU AL AR AR G
  Minimum discounted reward: 0.30903154382632636
  Maximum discounted reward: 4.7829690000000005
  Mean discounted reward discounted reward: 3.9212495602843256
  Standard Deviation of discounted rewards: 0.6934447618897589
- In state 11, Attempting up, can either lead to the robot moving to state 6 with probability 0.8 or staying in the same cell with probability 0.15 or moving to cell 12 with probability 0.05 whereas attempting right can move it to state 12 with probability 0.8, keep it in the same state with probability 0.1, move it to state 15 with probability 0.05 or move it to state 6 with probability 0.05.
  There is a small probability that the robot might end up in a position further away from the goal state than from its current position if it attempts right and hence attempt up is chosen as the optimal action for that state.
  The same logic is applied while choosing optimal action for state 15.
- In state 20, The robot will never get a negative reward (goes into water) if it attempts left although attempting left would take it longer to reach the goal. However if it attempts right so that it can reach faster to the goal, its reward irrespective of when it reaches the goal will always be negative (ie- positive reward obtained by reaching faster to the goal is always less than the reward obtained when passing through the water state).

  - In state 19, the agent can move closer to the goal with higher probability if it attempts up.

- (10 Points) Run the optimal policy that you found in the previous question for 10,000 episodes. Report the mean, standard deviation, maximum, and minimum of the observed discounted returns.

  - Minimum discounted reward: 0.3433683820292515
    Maximum discount reward: 4.7829690000000000
    Mean discounted reward discounted reward: 3.9212725547333545
    Standard Deviation of discounted rewards: 0.6931065667160492

- (5 Points) Using simulations, empirically estimate the probability that $S_{19} = 21$ (the state with water) given that $S_8 = 18$ (the state above the goal) when running the uniform random policy. Describe how you estimated this quantity (there is *not* a typo in this problem, nor an oversight).

  - At t=0 , placed agent at state 18 ie $S_0$=18. Ran 5000000 trials, and counted the number of trials for which $S_{11}$=21. The probability is calculated using the following formula:
    $(nooftrialsforwhichs_{11} = 21) \mid (totalnooftrials)$
  - Probability: 0.018693

- Steps to run the code:

- For optimal case: python run_policy.py True 10000

- For random policy: python run_policy.py False 10000

- For finding probability of $S_{19}$=21 given $S_8$=18 : python find_probability.py