For this week's discussion, you will spend some time applying what you have learned about reinforcement learning.

In your initial post, address the following:

* Design a set of states, rewards, and rules for an intelligent agent playing a simple board game. You may choose any board game that you are familiar with. In your design, be sure to consider the following:
  + States, including starting and ending states, and possible actions
  + Rewards or penalties for reaching a state
  + Rules for navigating from one state to another
* Compare your approach to the Markov Decision Process (MDP) that you learned about in this module. What similarities and differences do you see between your approach and the MDP?

In your response posts to your peers, address the following:

* Evaluate the thoroughness of your classmates' designs. What did they do well? Were there any states that they missed? Do their rewards and penalties make sense, and help move the agent to the goal?

Insert childhood memory of playing the board game mousetrap. And some of the missing pieces I or my cousins lost after taking the cheese. Good times, Good times.

Attempting to design a reinforcement learning agent based around “Mouse Trap” involves us defining states, actions, & rules. In addition to that, we have to analyze the similarities to & from what is listed in Markov Decision process or the MDP. The Markov Decision Process (MDP) forms the basis for reinforcement learning, which is how an intelligent agent learns through trial and error. The game starts with each agent’s token on the starting position on the board, with the mousetrap not being fully built. All players can move freely, as each space triggers an event that starts the building of the trap, it’s construction progress also defines the state. The game is finished when any agent wins by trapping the opposition or being caught themselves.

The agent’s actions begin with the roll of the dice, with a possible outcome being based on the designated landing space, either building the trap or activating it if the requirements have been met. Rewards and penalties guide the agent’s decisions, with positive rewards for actions such as advancing closer to the end of the board (+10), building parts of the trap (+20), trapping an opponent (+50), and winning the game (+100). Conversely, penalties are assigned for landing on spaces that help opponents build the trap (-10), getting the agent’s token caught in the trap (-20), or losing the game (-50). Navigation is influenced by dice rolls, but the agent can strategize by prioritizing advancement or targeting spaces that facilitate trap-building. It can also delay activating the trap until opponents are within the trap zone.

When looking at the framework like this, I believe it closely aligns with the Markov Decision Process, as it involves states, actions, rewards, & shows transitions that are reflected via dice rolls. Agents aim to maximize cumulative rewards, a mirror effect of MDP’s goal. However additional complexities are also introduced by relying on unpredictable & unmanipulated dice rolls. Some other factors would be game rules & other player’s actions.

In conclusion, MDP principles provide a foundational framework for designing an intelligent agent for Mouse Trap, the dynamic nature of this game requires strategy & a bit of cunning.

Team, H. C. (n.d.). *How to play mouse trap rules and instructions - hasbro*. Hasbro Instructions. https://instructions.hasbro.com/en-us/instruction/Mouse-Trap-Game

Das, A. (2017, March 26). *The very basics of reinforcement learning*. Medium. Retrieved from https://becominghuman.ai/the-very-basics-of-reinforcement-learning-154f28a79071

Beysolow, II, Taweh. *Applied Reinforcement Learning with Python : With OpenAI Gym, Tensorflow, and Keras*, Apress L. P., 2019.*ProQuest Ebook Central*, <https://ebookcentral.proquest.com/lib/snhu-ebooks/detail.action?docID=5880718>.

Evening Frank, how goes the mid-week for you?

I feel crazy, I had to go look up Snakes & Ladders. Learned that the original name was snakes & ladders, whereas I grew up with chutes & ladders. Had to rack my mind around that one for a second.

After reading through your laid out design for the learning agent in chutes & ladders is clear & easily understandable. By defining the states as board positions, actions as dice rolls, & and the transitions based on snakes & ladders. The reward system is logical, positive rewards for climbing ladders or making it to the end in the final square, while you still might incur a penalty for setbacks like sliding down the snakes. That said, there are areas for improvement. While board positions work well as states, adding information about nearby snakes or ladders could make the agent’s decision-making more strategic. For instance, knowing if a snake or ladder is within the next few spaces could help the agent plan its moves more effectively.

If I had to find something to comment on, maybe having a better reward structure or more detailed. Penalties for sliding down snakes could vary based on the length of the snake, while rewards might add a +1 or +2 based on the height. To give a bit more variability & ad to the game’s feedback system.

Your comparison to MDP is quite strong, detailed, particularly in highlighting how dice rolls add randomness while transitions follow a fixed board layout. Overall, the framework is solid, doing a great job of applying reinforcement learning principles to Snakes & ladders.

Hey Nathan, busy week we got here. I believe this discussion post was one of the more challenging ones I’ve had since attending SNHU. Your design is well thought out, particularly how you defined the game states & it’s reward system. By beginning with an empty board and ending once the board is filled, you outlined the state. Intermediate configurations or turn by turn play is a different state, constantly changing based on the players actions. The reward is logical enough, assuming positive rewards for winning, perhaps in a set, with penalties for losing, and no such rewards for a draw. The suggestion of smaller rewards for strategic moves, such as controlling the center or creating a fork, adds depth to the design by encouraging optimal play. This draws back to different states based on intermediate configurations.

Some differences should be noted, like the adding a bit more refinement to said model by including sub states. Maybe unique scenarios where each player can guarantee victory in a few moves. Though overall, I think your approach is thorough, effectively applying learning principles into a simple game.