Introduction to Reinforcement Learning

Swati Mishra

Applications of Machine Learning (4AL3)

Fall 2024

McMaster University

ENGINEERING

Tips on Planning for Final Project

- All project milestones can be submitted at **anytime** before the deadline. You do not have to wait until the deadline to finish it.

- Milestone III is all about making **small improvements** to your project. Most of your work is finished in Milestone II. As an example, if you experiment with 2 normalization techniques; Standard Scalar and Min Max, you can report results of 1 technique in milestone II and results of another technique in milestone III.

- There are 3 parts: Data + Model + Evaluation (distribute the workload evenly across team members).

- Try to stick with topics that have been covered in the class. Trying to learn something entirely new may take way more time and effort.

- Do not be too ambitious. It’s more important to complete the project. Please talk to me when in doubt.

Reinforcement Learning

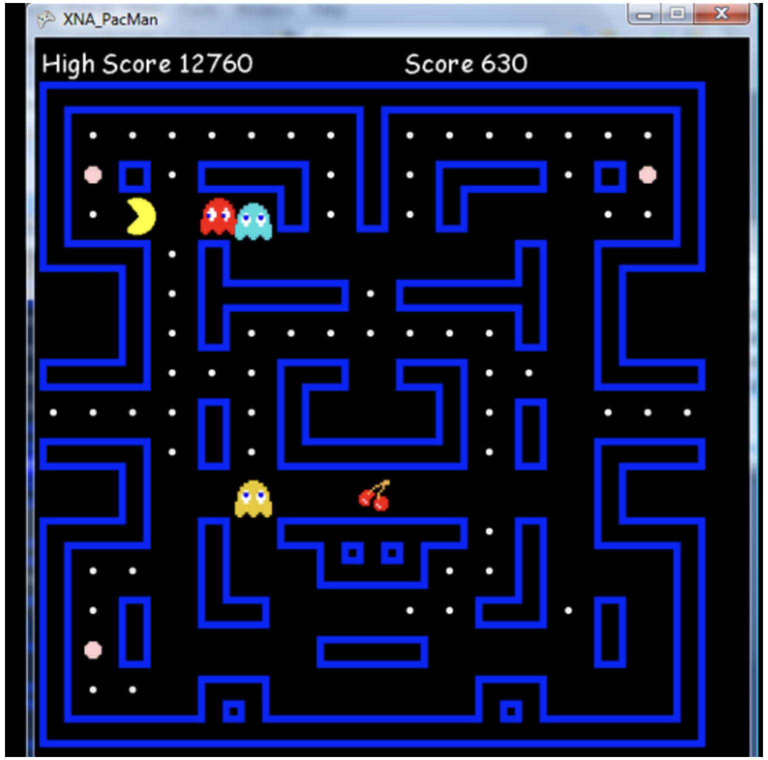
- A software agent
  - Makes observations,
  - Takes actions within an environment,
  - Receives a reward:
    - The reward maybe positive – award for taking the right /acceptable action.
    - The reward maybe negative – punishment for taking the wrong/inacceptable action.

- The goal of the agent is to maximize the reward and minimize the punishment.

- If you think about it, the process is entirely trial and error.

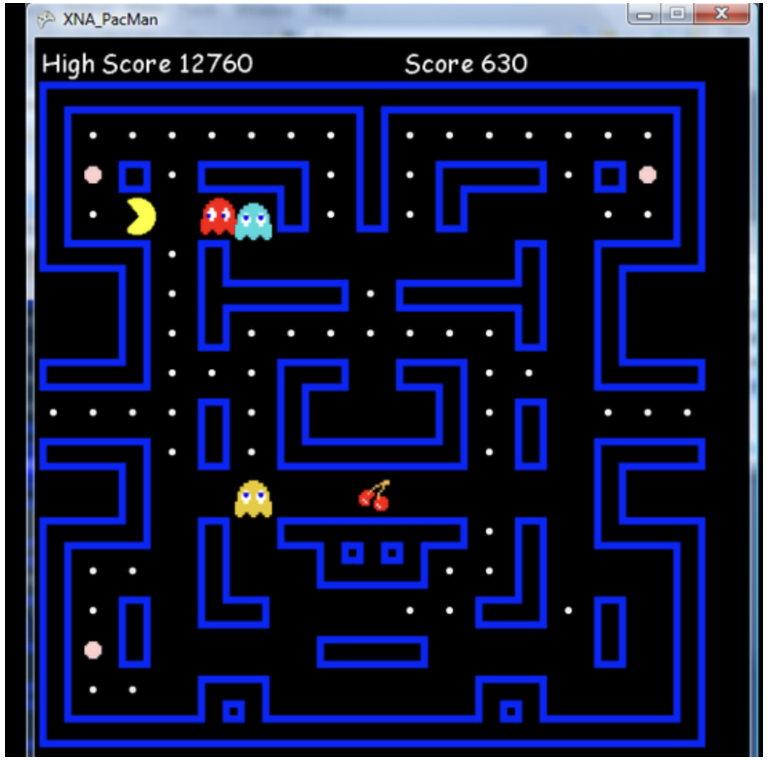
Reinforcement Learning

- Think Atari game Pac-man!
- The **environment** is
- The **agent** is
- The **actions** are
- The **reward** is
- The **observation** is



Reinforcement Learning

- Think Atari game Pac-man!
- The **environment** is the simulation of the Atari game.
- The **agent** is something that control the Ms. Pac-Man movements.
- The **actions** are possible directions in which the movements can occur (Up, Left, Right, Down).
- The **reward** is points earned or the game score.
- The **observation** is screenshots of the game stages.



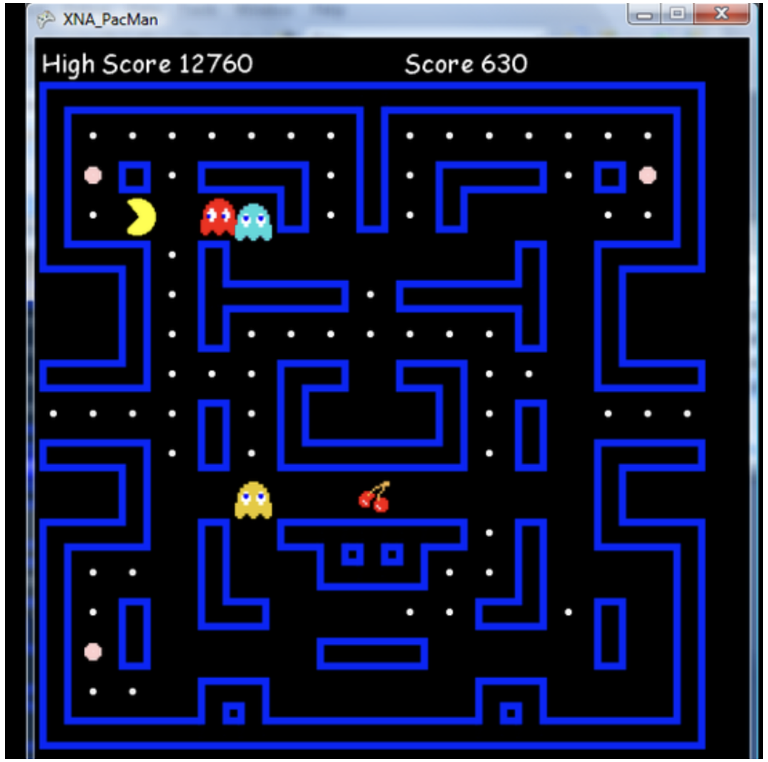
Picture Source: Wikipedia

Reinforcement Learning: Policy

- RL systems need a **policy function** to determine the behavior of the agent

- A policy defines the learning agent's way of behaving at a given time. It is defined as the **mapping from** perceived states of the **environment to actions** that the agent must takes when those states are encountered.

- Policy function can be a simple hash look up table, or a complex like some sort of search algorithm.



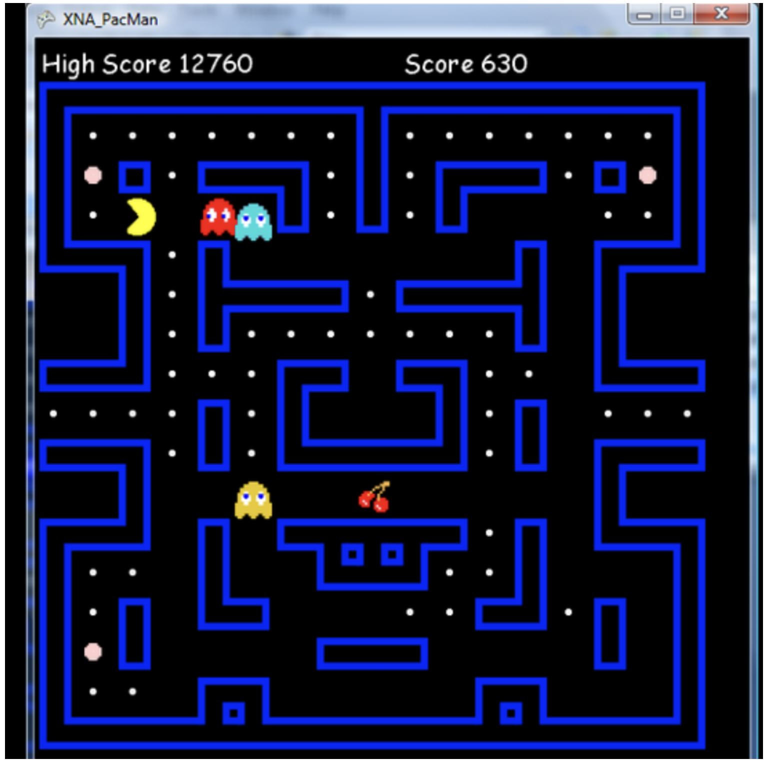
McMaster University

Reinforcement Learning: Policy

- RL systems need a **policy function** to determine the behavior of the agent

- A policy defines the learning agent's way of behaving at a given time. It is defined as the **mapping from** perceived states of the **environment to actions** that the agent must takes when those states are encountered.

- Policy function can be a simple hash look up table, or a complex like some sort of search algorithm.

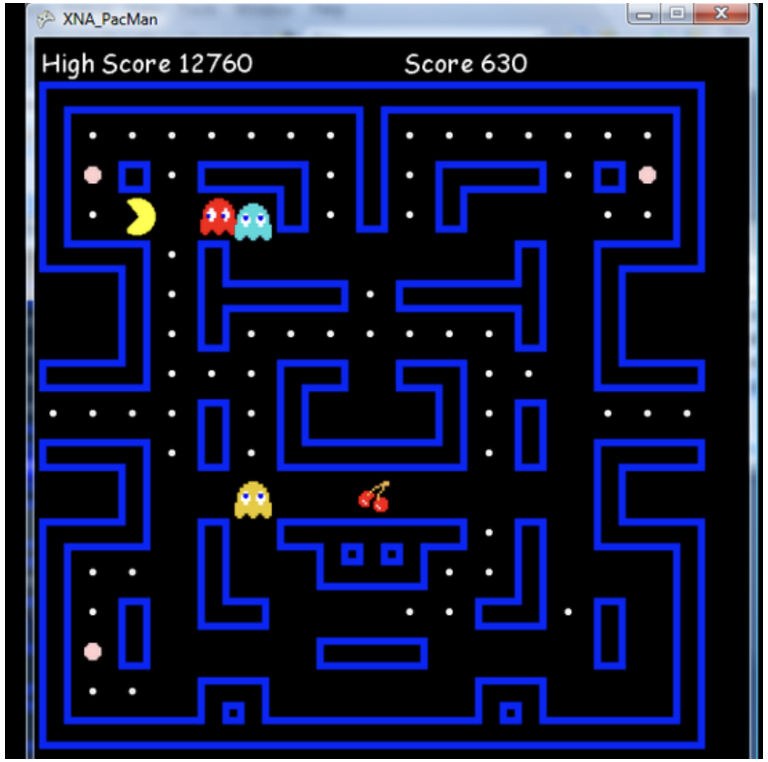


Reinforcement Learning: Policy

- RL systems need a **policy function** to determine the behavior of the agent

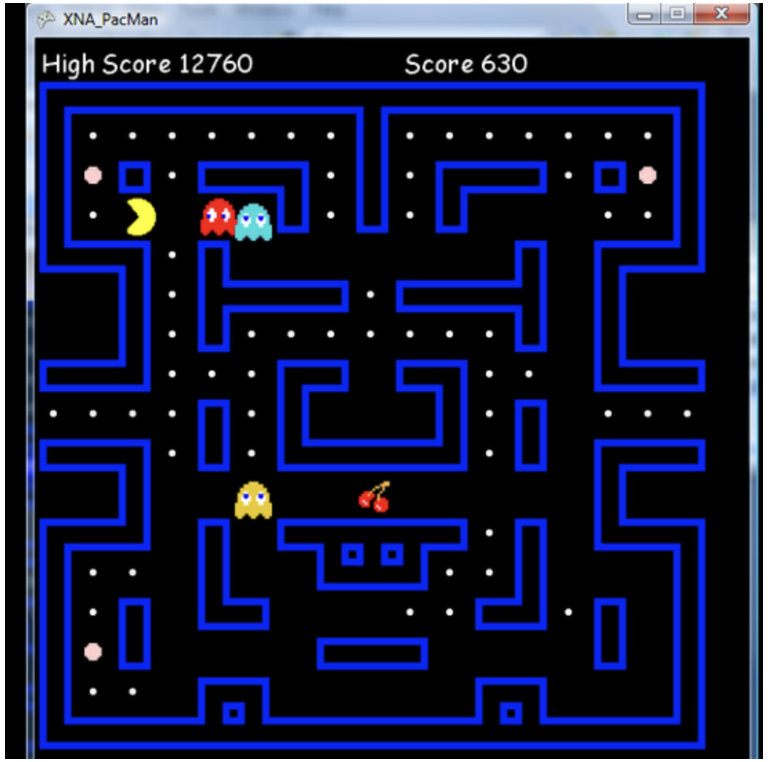
- A policy defines the learning agent's way of behaving at a given time. It is defined as the **mapping from** perceived states of the **environment to actions** that the agent must takes when those states are encountered.

- Policy function can be a simple hash look up table, or a complex like some sort of search algorithm.



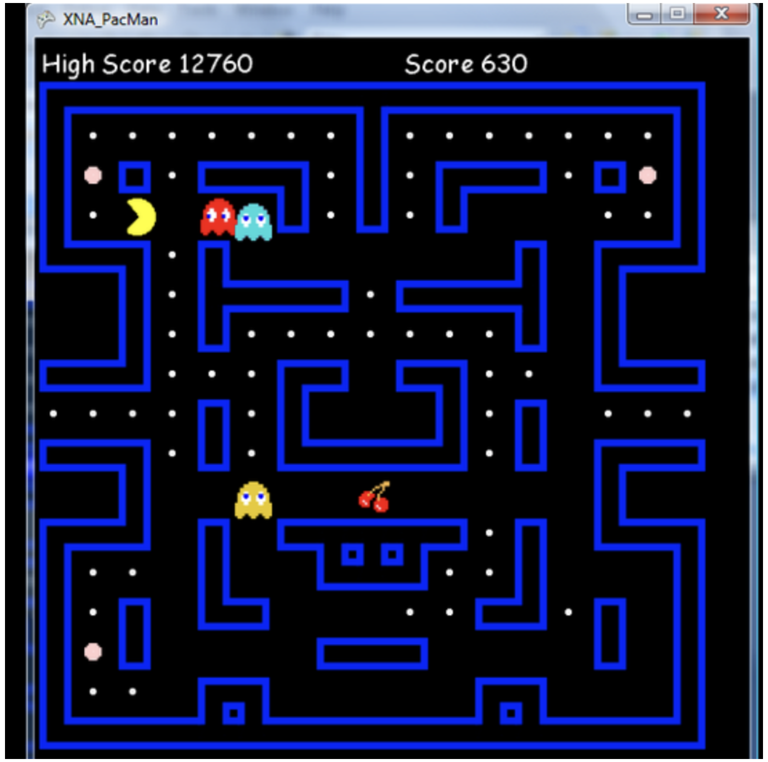
Reinforcement Learning: Policy

- RL systems need a **policy function** to determine the behavior of the agent
- A policy defines the learning agent’s way of behaving at a given time. It is defined as the **mapping from** perceived states of the **environment to actions** that the agent must takes when those states are encountered.
- Policy function can be a simple hash look up table, or a complex like some sort of search algorithm.



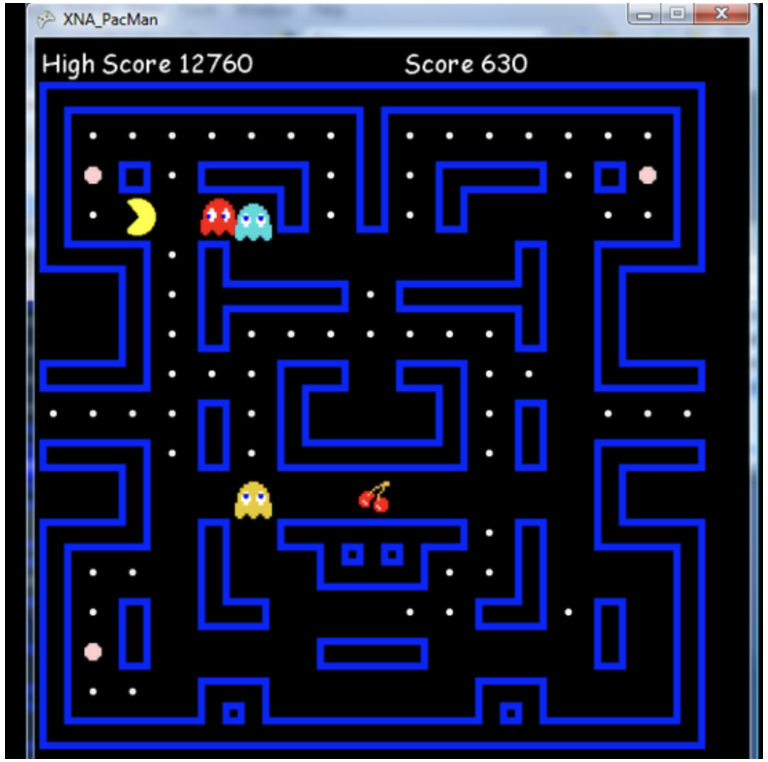
Reinforcement Learning: Reward

- At each time step, the agent receives a “numeric” reward, and it’s goal is to maximize it.
- Reward at a given time step depends on the current action, and current observation.
- The agent may learn a policy based on reward, not the other way round.
- If in the current state, the policy selected gives low rewards, then policy is changed for the future state.
- Reward signal indicates what is good in current state.



Reinforcement Learning: Value

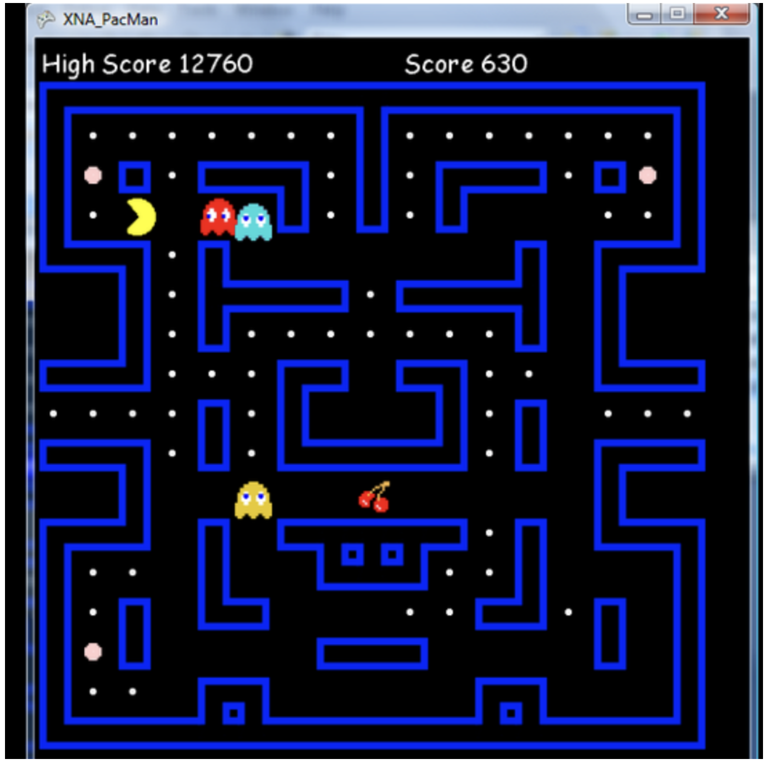
- Value function indicates what is advantageous in the long run.
- It indicates the long-term-desirability of a given state in the environment.
- A state with low immediate reward does not necessarily yield bad score, because maybe it is helpful in the long run.
- Value function is estimated through observations and reward.
- Usually, the goal is to seek a state of highest value, not highest reward.



Picture Source: Wikipedia

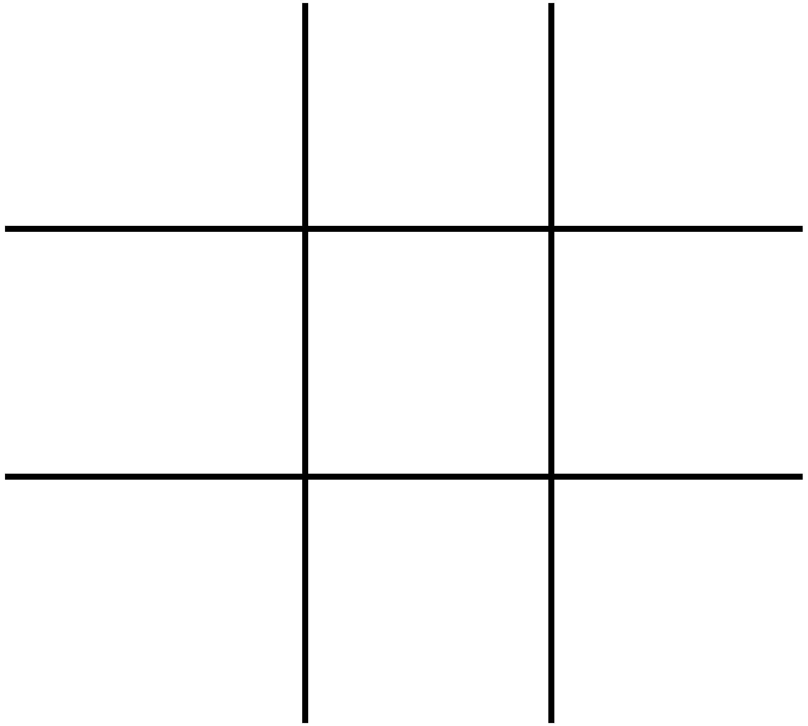
Reinforcement Learning: Model

- This is the simulation of the environment and mimic its behavior in real time.
- For a given state and action, a model might predict the next state and next reward.
- Models are useful for planning a set of actions to anticipate the states an agent might experience even before they are encountered.
- Model-based methods use a model and planning, model-free methods do not use model and planning.



Picture Source: Wikipedia

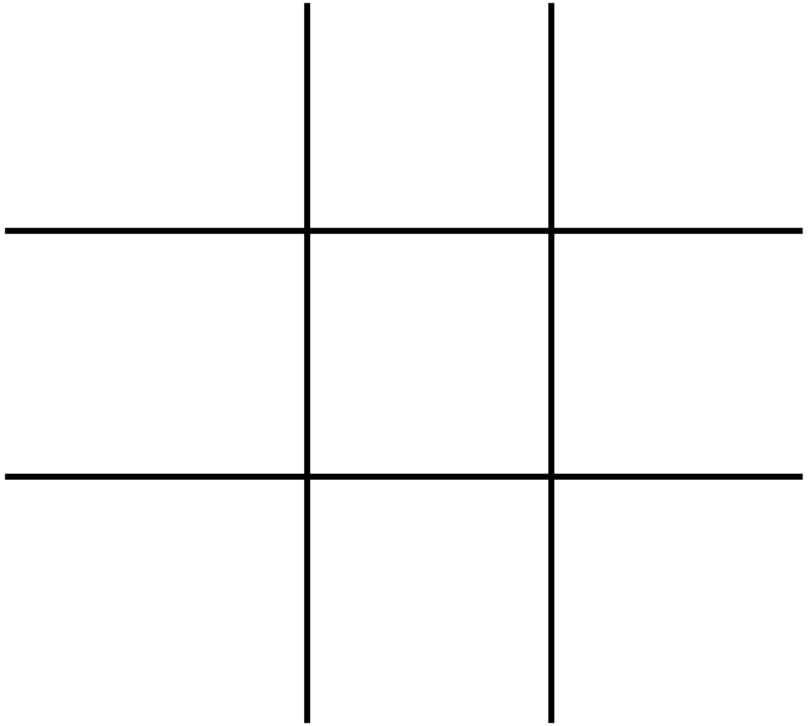
Reinforcement Learning



Let’s play Tic-Tac-Toe!

Reinforcement Learning

Starting Position: Empty board



Minmax solution from game theory:

$$\max_{x \in X} \min_{y \in Y} f(x, y) = \min_{y \in Y} \max_{x \in X} f(x, y)$$

Reinforcement Learning

Starting Position: Empty board

Possible moves : **0**
Move by A : **0**

0	0	0
0	0	0
0	0	0

Minmax solution from game theory:

$$\max_{x \in X} \min_{y \in Y} f(x, y) = \min_{y \in Y} \max_{x \in X} f(x, y)$$

Reinforcement Learning

Starting Position: Empty board

Possible moves : **0**
Move by A : **0**

Possible moves : **X**
Move by B : **X**

0	X	X
X	X	X
X	X	X

Minmax solution from game theory:

$$\max_{x \in X} \min_{y \in Y} f(x, y) = \min_{y \in Y} \max_{x \in X} f(x, y)$$

Reinforcement Learning

Starting Position: Empty board

Possible moves : **0**
Move by A : **0**

Possible moves : **X**
Move by B : **X**

Possible moves : **0**
Move by A : **0**

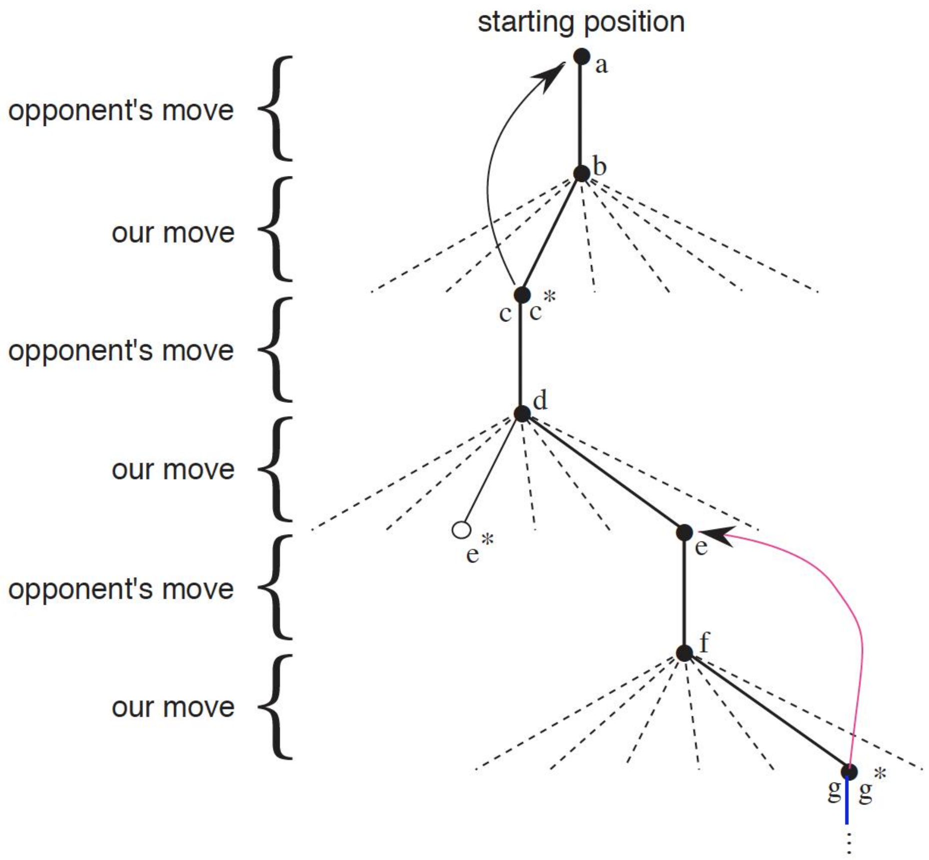
0	X	0
0	0	0
0	0	0

Minmax solution from game theory:

$$\max_{x\in X}\min_{y\in Y} f(x,y) = \min_{y\in Y}\max_{x\in X} f(x,y)$$

McMaster University

Reinforcement Learning



0	X	0
0	0	0
0	0	0

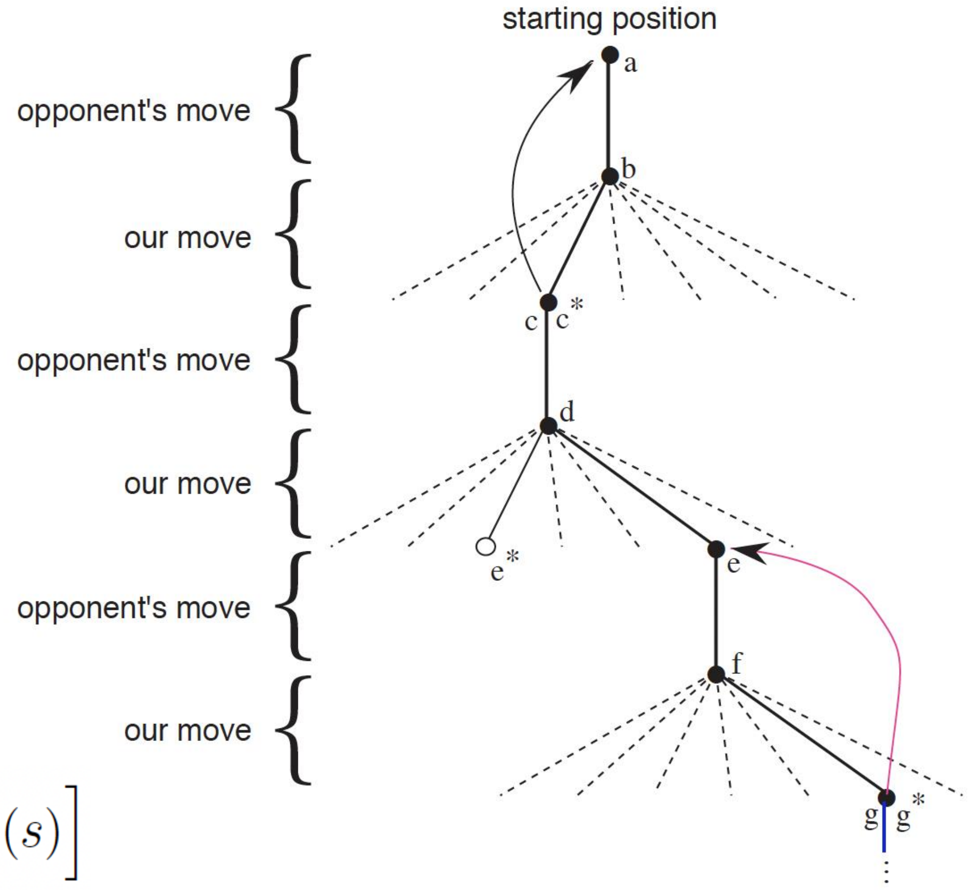
Minmax solution from game theory:

$$\max_{x\in X}\min_{y\in Y} f(x,y) = \min_{y\in Y}\max_{x\in X} f(x,y)$$

Reinforcement Learning: Learning

- The idea is to play many games with the opponent.
- We observe the result from each move and many times, select states that lead to greatest value.
- As we progress, we change the values of our current state.
- To make more accurate estimates of the probability of success (e.g. maximizing the reward), we “back up” the value of state after each greedy move.
- s = state before greedy move,
  s’ = state after the move,
  V = estimated value of s

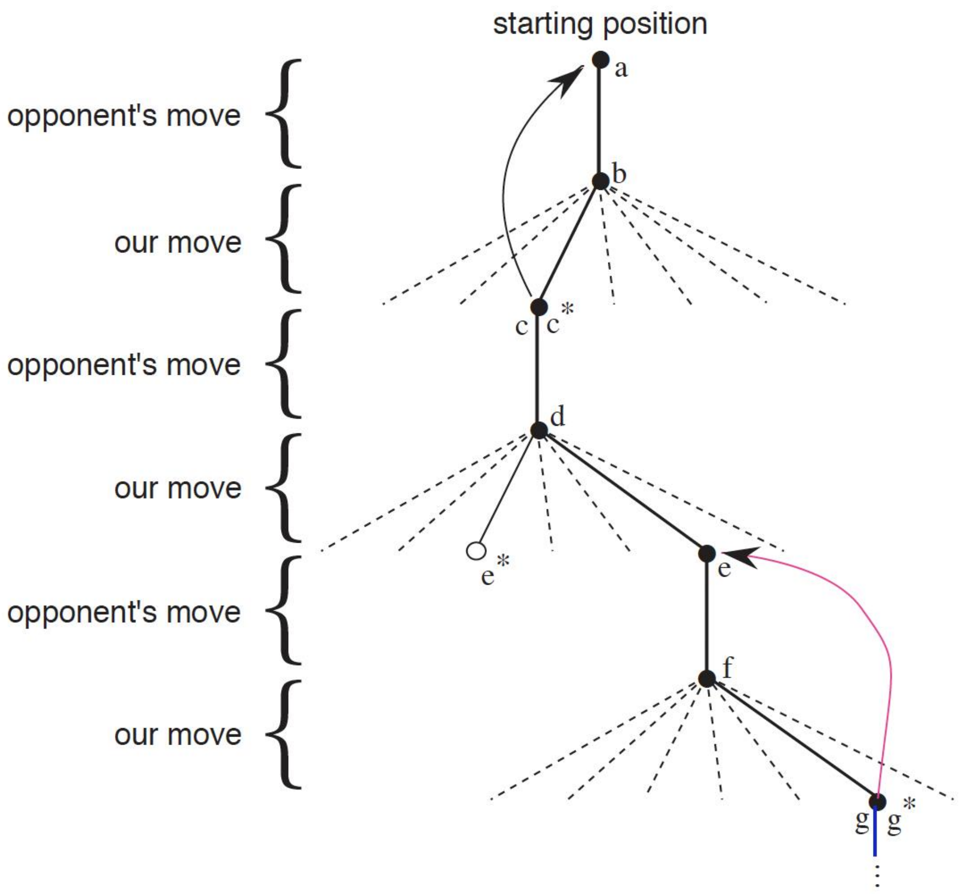
$$V(s) \leftarrow V(s) + \alpha\Big[V(s') - V(s)\Big]$$



McMaster University

Reinforcement Learning: Learning

- The idea is to play many games with the opponent.
- We observe the result from each move and many times, select states that lead to greatest value.
- As we progress, we change the values of our current state.
- To make more accurate estimates of the probability of success (e.g. maximizing the reward), we “back up” the value of state after each greedy move.



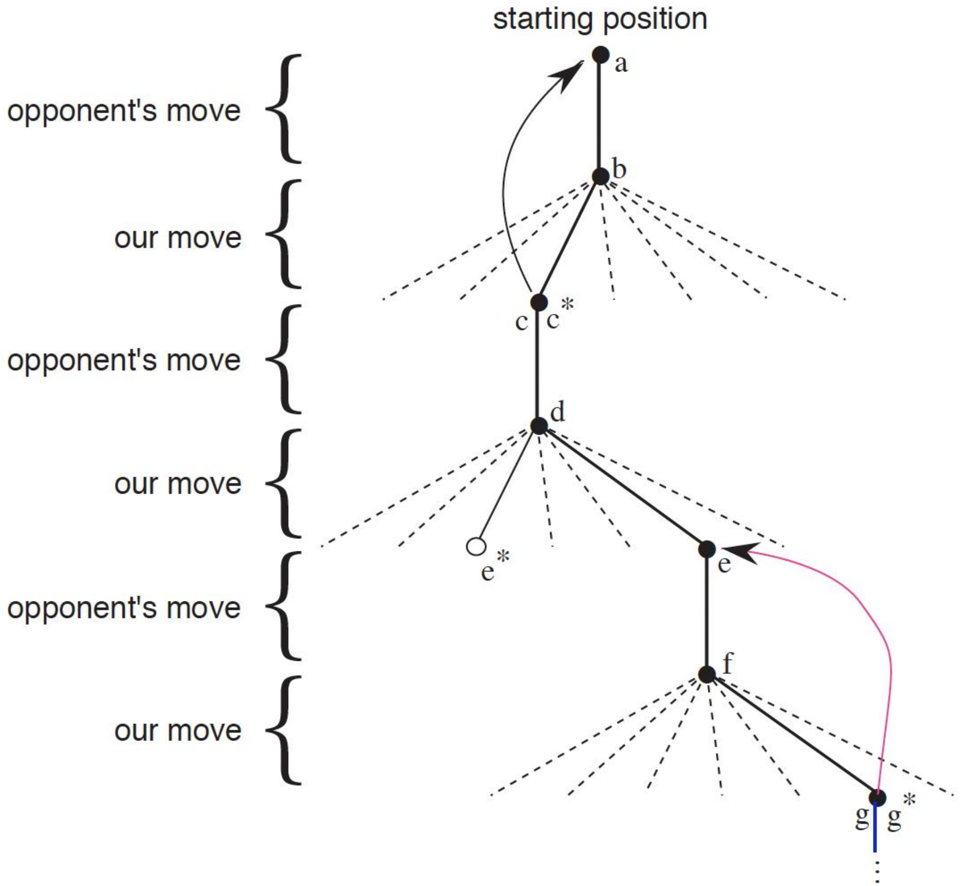
Reinforcement Learning: Learning

- s = state before greedy move,
  s’ = state after the move,
  V = estimated value of s

$$V(s) \leftarrow V(s) + \alpha\Big[V(s') - V(s)\Big]$$

- If the step-size parameter is reduced properly over time, this method converges, for any fixed opponent

- If the step-size parameter is not reduced all the way to zero over time then this player also plays well against opponents that slowly change their way of playing.

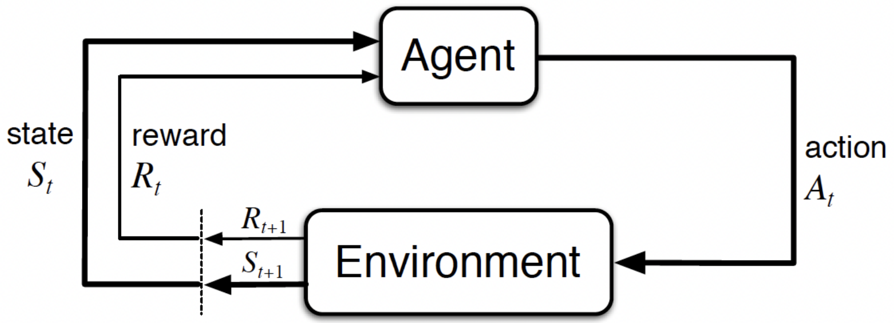


McMaster University

Reinforcement Learning: Policy Evaluation

- How do we compute the state- value of an arbitrary policy?

$$
\begin{aligned}
v_\pi(s) &= \mathbb{E}_\pi\big[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \mid S_t = s\big] \\
&= \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_\pi(s')\Big],
\end{aligned}
$$

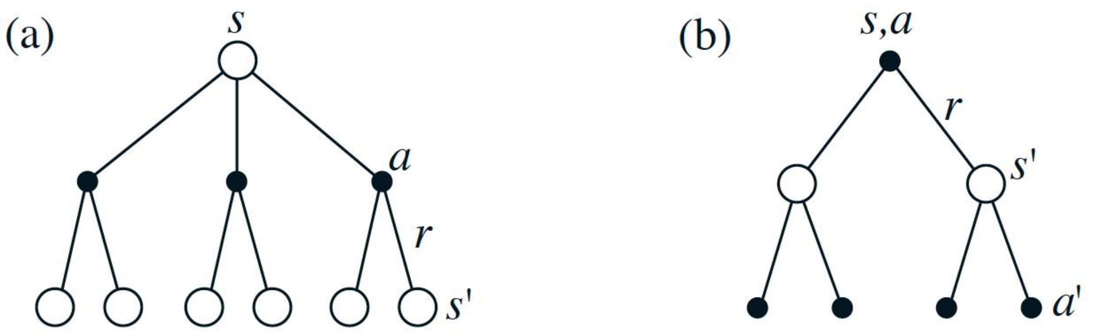


If the environment's dynamics are completely known, then above is a system of |S| simultaneous linear equations.

Reinforcement Learning: Policy Evaluation

- How do we compute the state- value of an arbitrary policy
  - Using the Bellman equation for: $$= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_\pi(s')\Big]$$

The Bellman equation averages over all the possibilities, weighting each by its probability of occurring.



Backup diagrams for (a) value function and (b) back up policy

Reinforcement Learning: Policy Evaluation

- How do we compute the state- value of an arbitrary policy
  - Using the Bellman equation for: $= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_\pi(s')\Big]$

$$
\begin{aligned}
v_{k+1}(s) &= \mathbb{E}_\pi[R_{t+1} + \gamma v_k(S_{t+1}) \mid S_t = s] \\
&= \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\Big[r + \gamma v_k(s')\Big]
\end{aligned}
$$

In principle, iterative solution methods are most suitable.

Readings

Reference Reading:

- Uploaded on Avenue

Thank You