RL: Basics

The Markov Reward Process

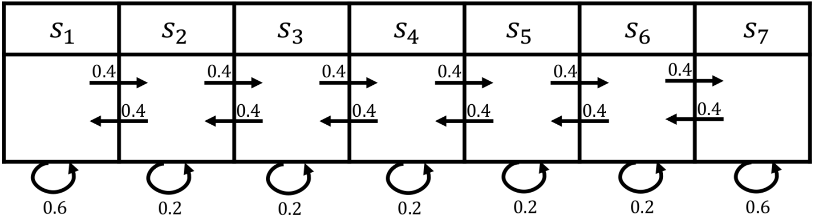
Marius Lindauer

Markov Reward Process (MRP)

- Extend Markov Process by rewards
- Definition of Markov Reward Process (MRP) $M = (S, P, R, \gamma)$
  - S is a (finite)
  - P is dynamics/transition model that specifies $P(s_{t+1} = s' \mid s_t = s)$
  - R is a reward function $R(s_t = s) = \mathbb{E}[r_t \mid s_t = s]$
  - Discount factor $\gamma \in [0, 1]$
- Note: no actions
- If finite number (N) of states, we can express R as a vector

Automated Machine Learning Hannover

Mars Rover as MRP



Rewards:

- $+1$ in s_1,
- $+10$ in s_7,
- 0 in all other states

Return & Value Function

- Definition of Horizon
  - Number of time steps in each episode
  - Can be infinite
  - Otherwise called finite Markov reward process

Return & Value Function

- Definition of Horizon
  - Number of time steps in each episode
  - Can be infinite
  - Otherwise called finite Markov reward process
- Definition of Return: G_t (for a MRP)
  - Discounted sum of rewards from time step t to horizon

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

Return & Value Function

- Definition of Horizon
  - Number of time steps in each episode
  - Can be infinite
  - Otherwise called finite Markov reward process
- Definition of Return: G_t (for a MRP)
  - Discounted sum of rewards from time step t to horizon

$$G_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots$$

- Definition of State Value Function: $V(s)$ for a MRP
  - Expected return from starting in state s

$$V(s) = \mathbb{E}[G_t \mid s_t = s] == \mathbb{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots \mid s_t = s]$$

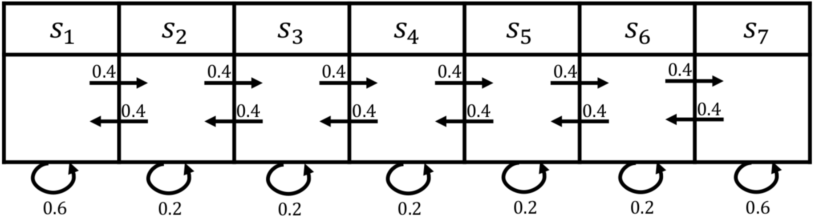
AutoML.org Automated Machine Learning Hannover

Discount Factor

$$V(s) = \mathbb{E}[G_t \mid s_t = s] == \mathbb{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \ldots \mid s_t = s]$$

- Mathematically convenient (avoid infinite returns and values)
- Humans often act as if there's a discount factor $\gamma < 1$
- $\gamma = 0$: Only care about immediate reward
- $\gamma = 1$: Future reward is as beneficial as immediate reward
- If episode lengths are always finite, can use $\gamma = 1$ (but don't have to)

Mars Rover as MRP



Rewards:

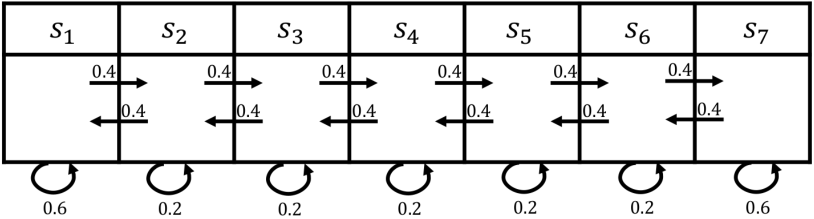
- $+1$ in s_1,
- $+10$ in s_7,
- 0 in all other states

Sample returns for 4-step episodes, $\gamma = 1/2$

- $s_4, s_5, s_6, s_7 : 0 + \frac{1}{2} \cdot 0 + \frac{1}{4} \cdot 0 + \frac{1}{8} \cdot 10 = 1.25$

Automated Machine Learning Hannover

Mars Rover as MRP



Rewards:

- $+1$ in s_1, $+10$ in s_7, 0 in all other states

$$V(s) = \mathbb{E}[G_t \mid s_t = s] == \mathbb{E}[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \gamma^3 r_{t+3} + \dots \mid s_t = s]$$

$\rightsquigarrow$ $V(s_1) = 1.53, V(s_2) = 0.37, \dots, V(s_7) = 15.31$

Automated Machine Learning Hannover

Computing the Value of a Markov Reward Process

- Could estimate by simulation:
    1. Generate a large number of episodes
    2. Average returns
- Requires no assumption of Markov structure

Computing the Value of a Markov Reward Process

- Could estimate by simulation:
- Markov property yields additional structure
- MRP value function satisfies

$$V(s) = R(s) + \gamma \sum_{s' \in S} P(s' \mid s) V(s')$$

Automated Machine Learning Hannover

Matrix Form of Bellman Equation for MRP

$$\begin{pmatrix} V(s_1) \\ \dots \\ V(s_n) \end{pmatrix} = \begin{pmatrix} R(s_1) \\ \dots \\ R(s_n) \end{pmatrix} + \gamma \begin{pmatrix} P(s_1|s_1) & \dots & P(s_n \mid s_1) \\ \dots & & \\ & \dots & \dots \\ P(s_1|s_n) & \dots & P(s_n \mid s_n) \end{pmatrix} \begin{pmatrix} V(s_1) \\ \dots \\ V(s_n) \end{pmatrix}$$

Automated Machine Learning Hannover

Matrix Form of Bellman Equation for MRP

$$\begin{pmatrix} V(s_1) \\ \dots \\ V(s_n) \end{pmatrix} = \begin{pmatrix} R(s_1) \\ \dots \\ R(s_n) \end{pmatrix} + \gamma \begin{pmatrix} P(s_1|s_1) & \dots & P(s_n \mid s_1) \\ \dots & & \\ & \dots & \dots \\ P(s_1|s_n) & \dots & P(s_n \mid s_n) \end{pmatrix} \begin{pmatrix} V(s_1) \\ \dots \\ V(s_n) \end{pmatrix}$$

$$\begin{aligned} V &= R + \gamma P V && (1) \\ V - \gamma P V &= R && (2) \\ (1 - \gamma P) V &= R && (3) \\ V &= (1 - \gamma P)^{-1} R && (4) \end{aligned}$$

Automated Machine Learning Hannover

AutoML.org

Matrix Form of Bellman Equation for MRP

$$\begin{pmatrix} V(s_1) \\ \dots \\ V(s_n) \end{pmatrix} = \begin{pmatrix} R(s_1) \\ \dots \\ R(s_n) \end{pmatrix} + \gamma \begin{pmatrix} P(s_1|s_1) & \dots & P(s_n \mid s_1) \\ \dots & & \\ & \dots & \dots \\ P(s_1|s_n) & \dots & P(s_n \mid s_n) \end{pmatrix} \begin{pmatrix} V(s_1) \\ \dots \\ V(s_n) \end{pmatrix}$$

$$\begin{aligned} V &= R + \gamma P V && (1) \\ V - \gamma P V &= R && (2) \\ (1 - \gamma P) V &= R && (3) \\ V &= (1 - \gamma P)^{-1} R && (4) \end{aligned}$$

$\rightsquigarrow$ Solving directly requires taking a matrix inverse $O(n^3)$

Automated Machine Learning Hannover

Iterative Algorithm for Computing Value of a MRP

- Dynamic Programming :
  - Initialize $V_0(s) = 0$ for all s
  - For $k = 1$ until convergence
    - For all s in S

$$V_k(s) = R(s) + \gamma \sum_{s' \in S} P(s' \mid s) V_{k-1}(s')$$

Automated Machine Learning Hannover

AutoML.org

Iterative Algorithm for Computing Value of a MRP

- Dynamic Programming :
  - Initialize $V_0(s) = 0$ for all s
  - For $k = 1$ until convergence
    - For all s in S

$$V_k(s) = R(s) + \gamma \sum_{s' \in S} P(s' \mid s) V_{k-1}(s')$$

- Computational complexity: $O(|S|^2)$ for each iteration $(|S| = n)$

Automated Machine Learning Hannover