# Notes on Policy Classes in Reinforcement Learning

based on the discussion in Wouter van Heeswijk’s [Four Policies of Reinforcement Learning](https://github.com/dimitarpg13/reinforcement_learning_and_game_theory/blob/main/articles/ReinforcementLearning/The_Four_Policy_Classes_of_Reinforcement_Learning_Wouter_van_Heeswijk_TDS.pdf), and [Sutton’s book](https://github.com/dimitarpg13/reinforcement_learning_and_game_theory/blob/main/books/ReinforcementLearningSuttonSecondEdition2020.pdf)

(written by D. Gueorguiev, 12/15/23)

Assumption: We formulate the Reinforcement Learning problem as Markov Decision Process (MDP) model.

Sutton (Richard S. Sutton, 2020) defines *Markov Decision Process*, *Learning Policy,* and *Bellman equations* as:

**Markov Decision Process** (abbrev *MDP*): a 4-tuple with

* is a set of states (finite or infinite)
* is a set of actions (finite or infinite)
* is the probability to get from state to state with action and with reward .
* is the discount factor which determines to what extent the focus is on the most recent rewards. with there is no focus on the most recent rewards only.

**Learning Policy** (or just *Policy*): function which represents mapping from states to probabilities of selecting each possible action.

If the agent is following policy at time , then is the probability that if . Note that is an ordinary function which defines a probability distribution over for each .

**Bellman equations**:

for all (1)

(2)

(3)

(4)

(5)

(6)

Eq (1) – (6) represent the Bellman optimality equation for the value function . Notice the use of Sutton’s notation. Those equations come from the discussion in Chapter 3 of his book. For details see the document [“Note on Q functions and V functions in Reinforcement Learning”](https://github.com/dimitarpg13/reinforcement_learning_and_game_theory/blob/main/docs/Note_on_Q_functions_in_Reinforcement_Learning.docx).

The goal is to solve the corresponding system of Bellman equations and thereby find the optimal policy .

Wouter van Heeswijk here references the following variant of the Bellman equation:

(7)

Here with is van Heeswijk’s notation for the value function of the optimal policy and corresponds to Sutton’s . is the reward function for the current state and action values and . Clearly,

(8)

If we set the discount factor to in (6) and expand the brackets pre-multiplying the two terms with the function of the dynamics, , we obtain the following expression for the second term in (6):

(9)

We employ Sutton’s notation for the three-argument function of the dynamics which defined as

(10)

For details, see Eq (3.4) from (Richard S. Sutton, 2020).

Applying (10) to (9) gives us the following expression for the second term in (6):

(11)

This proves that van Heeswijk’s version of Bellman Optimality equation given with (7) is identical to Sutton’s equation (6) with discount factor set to .

Further in his article van Heeswijk stipulates that instead of solving Bellman optimality equation (7) we simply can try to maximize what he calls *cumulative reward* . Note that van Heeswijk *cumulative reward* corresponds to Sutton’s *total return* (see Eq (3.7) from (Richard S. Sutton, 2020)). Van Heeswijk defines *cumulative reward function* over a time horizon given a policy as :

(12)

and stipulates that in order to find the optimal policy we need to find the following maximum

(13)

Here denotes the environment state at moment .

There are two approaches to solve MDP model for optimality.

1. *Policy iteration*
2. *Value iteration*

*Policy iteration* fixes a policy, computes the corresponding policy value ( and/or ), and subsequently updates the policy using the new value. The algorithm iterates between these steps until the policy remains stable.

*Value iteration* relies on similar steps but aims to directly maximize the value functions and/or and only updates the policy afterwards. Finding the optimal value functions ( or ) equates to finding the optimal policy as either suffices to solve the system of Bellman equations.

## Appendix

### The Stochastic Optimization Problem

Let us consider a relatively simple inventory problem. We will formulate deterministic and stochastic versions of the problem. We will introduce the notion of *policy* in this process.

#### The deterministic inventory problem

Inventory Problem: we have to decide how much to order, , at time . We are going to assume that when we order , the items cannot be used until time . Let be the cost of items ordered in period (which can vary from one time period to the next) and assume that we are paid a price when we satisfy the demand given by .

Let be the sales at time , which is limited by the demand , and the available product which is our inventory plus our orders , so we can write

(1)

We assume the unsatisfied demand is lost. The left-over inventory is

(2)

We denote .

We want to maximize :

(3)

Of course, (3) is subject to the constraints (1) and (2). The solution is in the form of the vector of production and sales decisions . The global optimal solution involves finding over the entire horizon, that is for each .

#### The stochastic optimization formulation

Let us make the demands random. Since is random, the inventory is random. This means that the order quantity is random, as is the sales . Given this, the optimization problem (3) does not make sense with random demand and inventories. We “fix” the problem formulation by replacing the decisions with a function, known as policy . Here is a new state variable that captures what we need to know to make a decision. For the inventory problem our state is just the inventory . Thus , we rewrite our objective function as:

(4)

The new formulation (4) replaces (3) and is solved with the set of constraints (1) and (2).

The constraints are applied differently in the stochastic policy formulation. The policy has to obey (1) which is easy to enforce because at time both and are known. Then, after computing , we can compute from (2), which requires obtaining an instance of from the demand distribution (usually Poisson). Finally, the expectation in (4) means sampling over all the different sequences . In practice, we cannot actually do this, so imagine that we create a sample where represents a particular sequence of possible values of . If we assume that each sequence of demands is equally likely, we would approximate (4) with

(5)

Where is the number of sample paths in .

# Bibliography

Heeswijk, W. v. (2021, May 11). *The Four Policy Classes of Reinforcement Learning.* Retrieved from https://github.com/dimitarpg13/reinforcement\_learning\_and\_game\_theory/blob/main/articles/ReinforcementLearning/The\_Four\_Policy\_Classes\_of\_Reinforcement\_Learning\_Wouter\_van\_Heeswijk\_TDS.pdf

Powell, W. B. (2015, Jan 18). *Tutorial on stochastic optimization in energy — Part II: An energy storage illustration.* Retrieved from https://github.com/dimitarpg13/reinforcement\_learning\_and\_game\_theory/blob/main/articles/ReinforcementLearning/Powell-TutorialPartII-StochasticOptJan182015.pdf

Powell, W. B. (2018, Jul 4). *A unified framework for stochastic optimization.* Retrieved from https://github.com/dimitarpg13/reinforcement\_learning\_and\_game\_theory/blob/main/articles/ReinforcementLearning/Powell-A-Unified-Framework-for-Stochastic-Optimization-EJOR.pdf

Powell, W. B. (2019, Sep 3). *Reinforcement Learning and Stochastic Optimization.* Retrieved from https://github.com/dimitarpg13/reinforcement\_learning\_and\_game\_theory/blob/main/books/Powell-Reinforcement-Learning-and-Stochastic-Optimization.pdf

Richard S. Sutton, A. G. (2020, Oct 12). *Reinforcement Learning: An Introduction, second edition.* Retrieved from https://github.com/dimitarpg13/reinforcement\_learning\_and\_game\_theory/blob/main/books/ReinforcementLearningSuttonSecondEdition2020.pdf