# Introduction

In **reinforcement learning (RL)**there’s no answer key, but your reinforcement learning **agent** still has to decide how to act to perform its task. In the absence of existing training data, the agent learns from experience. It collects the training examples (“this action was good, that action was bad”) through **trial-and-error** as it attempts its task, with the goal of maximizing long-term **reward**.

# Reinforcement Learning Definitions

Reinforcement learning can be understood using the concepts of agents, environments, states, actions and rewards, all of which we’ll explain below. Capital letters tend to denote sets of things, and lower-case letters denote a specific instance of that thing; e.g. A is all possible actions, while a is a specific action contained in the set.

**Agent**: An agent takes actions; for example, your Starcraft II (SCII) AI player, or Super Mario navigating a video game. The algorithm is the agent. In real life, the agent is you.

**Action** (A): A is the set of all possible moves the agent can make. An action is almost self-explanatory, but it should be noted that agents usually choose from a list of discrete, possible actions. In video games, the list might include running right or left, jumping high or low, crouching or standing still. In the stock markets, the list might include buying, selling or holding any one of an array of securities and their derivatives.

**Discount factor**: The discount factor is multiplied by future rewards as discovered by the agent in order to dampen those rewards’ effect on the agent’s choice of action. Why? It is designed to make future rewards worth less than immediate rewards; i.e. it enforces a kind of short-term hedonism in the agent. Often expressed with the lower-case Greek letter gamma: γ. If γ is .8, and there’s a reward of 10 points after 3 time steps, the present value of that reward is 0.8³ x 10. A discount factor of 1 would make future rewards worth just as much as immediate rewards. The discount factor handles issues related to delayed gratification.

**Environment**: The world through which the agent moves, and which responds to the agent. The environment takes the agent’s current state and action as input, and returns as output the agent’s reward and its next state. For our SCII agents, the environment is the game. In real life, you are the agent and the environment could be the laws of physics and the rules of society that process your actions and determine the consequences of them.

**State** (S): A state is a concrete and immediate situation in which the agent finds itself; i.e. a specific place and moment, an instantaneous configuration that puts the agent in relation to other significant things such as tools, obstacles, enemies or prizes. It can the current situation returned by the environment, or any future situation. In SCII, a state is whenever our environment data is returned to our agent – which will happen multiple times a second.

**Reward** (R): A reward is the feedback by which we measure the success or failure of an agent’s actions in a given state. For example, in a video game, when Mario touches a coin, he wins points. From any given state, an agent sends output in the form of actions to the environment, and the environment returns the agent’s new state (which resulted from acting on the previous state) as well as rewards, if there are any. Rewards can be immediate or delayed. They effectively evaluate the agent’s action.

**Policy** (π): The policy is the strategy that the agent employs to determine the next action based on the current state. It maps states to actions, the actions that promise the highest reward.

**Value** (V): The expected long-term return with discount, as opposed to the short-term reward R. Vπ(s) is defined as the expected long-term return of the current state under policy π. We discount rewards, or lower their estimated value, the further into the future they occur.

**Q-value or action-value** (Q): Q-value is similar to Value, except that it takes an extra parameter, the current action a. Q(s, a) refers to the long-term return of an action taking action a under policy π from the current state s. Q maps state-action pairs to rewards. Note the difference between Q and policy.

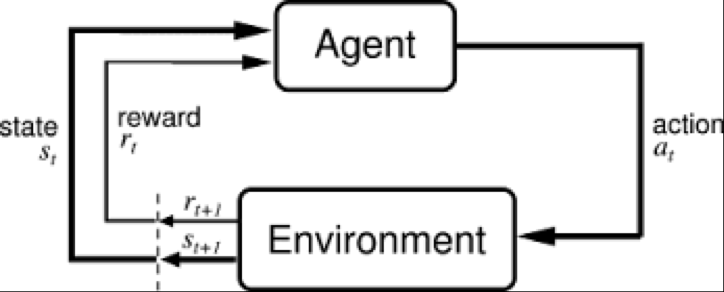
**Trajectory**: A sequence of states and actions that influence those states. From the Latin “to throw across.” The life of an agent is but a ball tossed high and arching through space-time unmoored, much like humans in the modern world.

***Key distinctions:*** *Reward is an immediate signal that is received in a given state, while value is the sum of all rewards you might anticipate from that state. Value is a long-term expectation, while reward is an immediate pleasure. They differ in their time horizons. So you can have states where value and reward diverge: you might receive a low, immediate reward even as you move to position with great potential for long-term value; or you might receive a high immediate reward that leads to diminishing prospects over time. This is why the value function, rather than immediate rewards, is what reinforcement learning seeks to predict and control.*

# Overview

“So *environments* are functions that transform an *action* taken in the current *state* into the next *state* and provide a value for a *reward*; *agents* are functions that transform the new *state* and *reward* into the next *action*.

We can know and set the agent’s function, but in most situations where it is useful and interesting to apply reinforcement learning, we do not know the function of the environment. It is a black box where we only see the inputs and outputs. Reinforcement learning represents an agent’s attempt to approximate the environment’s function, such that we can send actions into the black-box environment that maximize the rewards it spits out.”



*Simple RL Schema \*Credit: Sutton & Barto (incompleteideas.net)*

In the feedback loop above, the subscripts denote the time steps t and t+1, each of which refer to different states: the state at moment t, and the state at moment t+1. Unlike other forms of machine learning – such as supervised and unsupervised learning – reinforcement learning can only be thought about sequentially in terms of state-action pairs that occur one after the other.

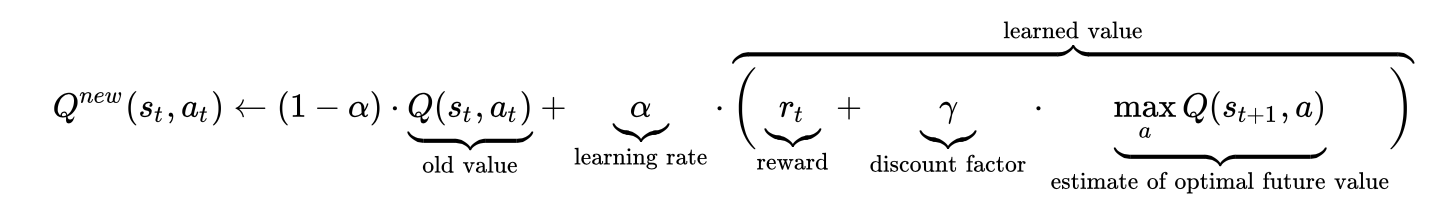
# Exploration / Exploitation Tradeoff

When “learning”, we get better results if we not always take the optimum path/choice (exploitation – exploit the system rewards), but if we occasionally “explore” non-optimum paths/choices. Typically we will use a decreasing probability of exploration, epsilon, with 1-δ as the probability of exploitation.

# Q-Learning

Q-Learning is the process of defining the quality (Q) of an action (a) based on the current state (s) of the environment. Effectively, we are trying to determine the best action for each state. For a simple maze system with a defined maze map and limited actions (up, down, left, or right), with enough exploration and exploitation, we can determine the Q values for each **s** and **a**. For a more complex system, we will need to construct a model that can calculate the Q values for any **s** and **a – at a particular time, t**.

Initially, we know nothing about the Q values. But we can update them using the following equation:



*The Q-Learning Equation Credit: Wikipedia*

By exploring and exploiting our environment, we will update our Q values (or model of Q values) and develop a way to best navigate our environment over time. The Q values can be used as the “known y values” for a supervised learning model for a given state.

The basic learning loop is:

* For a given state, determine an action to take based on Q values and/or model prediction.
* Take the action, with the environment providing the next state and reward.
* Use the state, action, next\_state, and reward to improve the Q values and the model.

Once a model is trained, it can be used to make action predictions for a given state.

# Sources and References:

*https://skymind.ai/wiki/deep-reinforcement-learning*

*https://medium.com/machine-learning-for-humans/reinforcement-learning-6eacf258b265*

*http://incompleteideas.net/book/bookdraft2017nov5.pdf*