2. Reinforcement Learning: From Bandits to Full RL

March 29, 2025

Contents

1 Introduction **4**

2 Associativity: Reward Dependency on States and Actions **4**
2.1 From Simple Bandits to State-Dependent Rewards 4
2.2 The Agent-Environment Interface 5
2.3 The Markov Property 6

3 Extension to Contextual Bandits **7**
3.1 Definition and Properties 7
3.2 Applications of Contextual Bandits 7
3.3 Action-Value Function for Contextual Bandits 8
3.4 Estimating Action Values in Contextual Bandits 8
3.5 Policy in Contextual Bandits 9
3.6 Example: Contextual Multi-Armed Bandit 10

4 Full RL with State Transitions **11**
4.1 The Critical Addition: Actions Affect Future States 11
4.2 Markov Decision Processes (MDPs) 12
4.3 Episodic vs. Continuing Tasks 13
4.4 The Return and Value Functions in Full RL 13
4.5 State-Value and Action-Value Functions 14
4.6 The Relationship Between State-Value and Action-Value Functions 15
4.7 The Bellman Equations 16
4.8 Optimal Value Functions and Policies 17
4.9 The Bellman Optimality Equations 17

4.10 The Credit Assignment Problem 18
4.11 Comparison: Bandits vs. Contextual Bandits vs. Full RL . . . 19

5 Conclusion 19

6 Summary of Key Equations 20
6.1 Reward Dependencies 20
6.2 MDP Dynamics 21
6.3 Return 21
6.4 Value Functions 21
6.5 Bellman Equations 22
6.6 Optimal Value Functions 22
6.7 Bellman Optimality Equations 22
6.8 Policies 22
6.9 Estimating Action Values 23

1 Introduction

This document covers the transition from the multi-armed bandit problem to full reinforcement learning, exploring how the introduction of states transforms the problem and solution approaches. We follow the notation and framework established in Sutton and Barto's Reinforcement Learning book.

2 Associativity: Reward Dependency on States and Actions

2.1 From Simple Bandits to State-Dependent Rewards

In the multi-armed bandit problem, rewards depend only on the selected action:

$$p(r_t|a_t) \tag{1}$$

Notation Overview

- $p(r_t|a_t)$ - The probability distribution of reward r_t given action a_t
- r_t - The reward received at time step t
- a_t - The action taken at time step t

Associativity introduces the concept of state, where rewards now depend on both the current state and the action taken:

$$p(r_t|s_t, a_t) \tag{2}$$

Notation Overview

- $p(r_t|s_t, a_t)$ - The probability distribution of reward r_t given state s_t and action a_t
- r_t - The reward received at time step t
- s_t - The state at time step t
- a_t - The action taken at time step t

In reinforcement learning with associativity, the agent recognizes that the environment is not stationary but exists in different states. The reward received for a particular action depends not only on which action was chosen but also on the environmental state in which it was chosen.

2.2 The Agent-Environment Interface

The standard agent-environment interface in reinforcement learning extends the bandit formulation to include states:

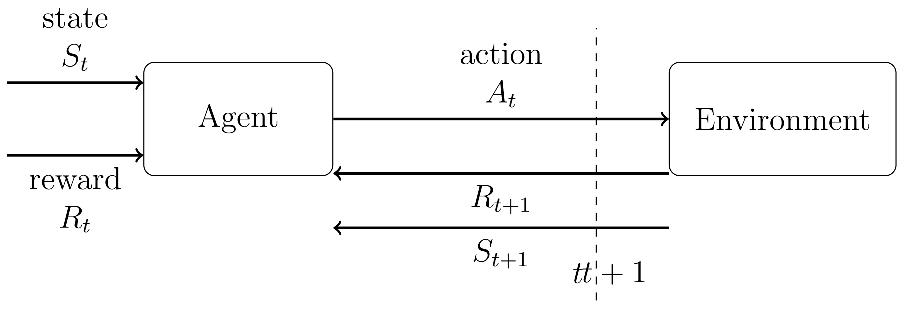

Figure 1: The agent-environment interaction in reinforcement learning (Sutton & Barto)

At each time step t:

- The agent observes the current state $S_t \in \mathcal{S}$
- Based on this state, the agent selects an action $A_t \in \mathcal{A}(S_t)$
- The environment responds with a reward $R_{t+1} \in \mathcal{R}$ and a new state S_{t+1}
- The agent then selects a new action based on the new state, and the process continues

This creates a sequence of states, actions, and rewards:

$$S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \ldots \tag{3}$$

Notation Overview

- S_t - The state at time step t
- A_t - The action taken at time step t
- R_{t+1} - The reward received after taking action A_t in state S_t
- $\mathcal{S}$ - The set of all possible states
- $\mathcal{A}(S_t)$ - The set of actions available in state S_t
- $\mathcal{R}$ - The set of possible rewards

2.3 The Markov Property

A key concept in reinforcement learning is the Markov property, which states that the future is independent of the past given the present. Formally, an environment has the Markov property if:

$$P(S_{t+1} = s', R_{t+1} = r | S_t, A_t, R_t, S_{t-1}, A_{t-1}, \ldots, R_1, S_0, A_0) = P(S_{t+1} = s', R_{t+1} = r | S_t, A_t) \tag{4}$$

Notation Overview

- $P(S_{t+1} = s', R_{t+1} = r | S_t, A_t, \ldots)$ - The probability of transitioning to state s' and receiving reward r given the entire history
- $P(S_{t+1} = s', R_{t+1} = r | S_t, A_t)$ - The probability of transitioning to state s' and receiving reward r given only the current state and action
- S_t - The state at time step t
- A_t - The action taken at time step t
- R_t - The reward received at time step t

The Markov property is crucial because it allows us to make decisions based solely on the current state without needing to remember the entire history of states and actions. This simplifies the problem significantly while still capturing the essential dynamics of many real-world situations.

3 Extension to Contextual Bandits

3.1 Definition and Properties

Contextual bandits represent an intermediate step between multi-armed bandits and full reinforcement learning. In this setting:

- The environment presents a state (or context) S_t to the agent
- The agent chooses an action A_t based on the state
- The environment provides a reward R_{t+1} that depends on both S_t and A_t
- The next state S_{t+1} is **not influenced** by the previous action A_t

The key distinguishing feature of contextual bandits is that while rewards depend on states, actions do not influence future states. The state transitions are independent of the agent's actions:

$$p(s_{t+1}|s_t, a_t) = p(s_{t+1}|s_t) \tag{5}$$

Notation Overview

- $p(s_{t+1}|s_t, a_t)$ - The probability of transitioning to state s_{t+1} given the current state s_t and action a_t
- $p(s_{t+1}|s_t)$ - The probability of transitioning to state s_{t+1} given only the current state s_t, independent of the action taken
- s_t - The state at time step t
- a_t - The action taken at time step t
- s_{t+1} - The next state at time step $t + 1$

3.2 Applications of Contextual Bandits

Contextual bandits are particularly useful in scenarios where:

- The state/context changes independently of the agent's actions
- Decisions only affect immediate rewards but not future states

- The agent needs to adapt to changing contexts

Some practical applications include:

- News article recommendation: The user's interests (state) change over time independently of which articles are recommended, but the reward (click-through rate) depends on both the user's interests and the selected article.
- Dynamic pricing: Market conditions (state) change due to external factors, while the price decision (action) affects immediate revenue but not future market conditions.
- Clinical trials with rotating patients: Each patient represents a new state, and the treatment decision affects the outcome for that patient but not the characteristics of future patients.

3.3 Action-Value Function for Contextual Bandits

In contextual bandits, the action-value function is defined as the expected immediate reward when taking action a in state s:

$$Q(s, a) = \mathbb{E}[R_{t+1}|S_t = s, A_t = a] \tag{6}$$

Notation Overview

- $Q(s, a)$ - The action-value function, representing the expected reward for taking action a in state s
- $\mathbb{E}[\cdot]$ - The expected value
- R_{t+1} - The reward received at time step $t + 1$
- S_t - The state at time step t
- A_t - The action taken at time step t

This differs from the action-value function in full reinforcement learning, which includes future rewards.

3.4 Estimating Action Values in Contextual Bandits

The action-value function in contextual bandits can be estimated from experience:

$$Q_t(s,a) = \frac{1}{n_t(s,a)} \sum_{i=1}^{t} r_i \cdot \mathbb{I}(s_i = s, a_i = a) \tag{7}$$

Notation Overview

- $Q_t(s,a)$ - The estimated action-value after t steps
- $n_t(s,a)$ - The number of times action a has been selected in state s up to time t
- r_i - The reward received at step i
- $\mathbb{I}(s_i = s, a_i = a)$ - An indicator function that equals 1 if $s_i = s$ and $a_i = a$, and 0 otherwise
- s_i - The state at time step i
- a_i - The action taken at time step i

An incremental update rule for the action values can also be derived:

$$Q_{t+1}(s_t, a_t) = Q_t(s_t, a_t) + \alpha[r_{t+1} - Q_t(s_t, a_t)] \tag{8}$$

Notation Overview

- $Q_{t+1}(s_t, a_t)$ - The updated estimate of the action-value
- $Q_t(s_t, a_t)$ - The current estimate of the action-value
- α - The step-size parameter (learning rate), $0 < \alpha \leq 1$
- r_{t+1} - The observed reward
- s_t - The state at time step t
- a_t - The action taken at time step t

3.5 Policy in Contextual Bandits

A policy π in contextual bandits maps states to probabilities of selecting each possible action:

$$\pi(a|s) = P(A_t = a|S_t = s) \tag{9}$$

Notation Overview

- $\pi(a|s)$ - The probability of selecting action a in state s under policy π
- $P(A_t = a|S_t = s)$ - The probability that the agent selects action $A_t = a$ when in state $S_t = s$
- A_t - The action taken at time step t
- S_t - The state at time step t

Various policy types can be used in contextual bandits:

- **Greedy Policy**: Always selects the action with the highest estimated value:
$$\pi(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a'} Q(s,a') \\ 0 & \text{otherwise} \end{cases} \tag{10}$$
- **ϵ-Greedy Policy**: Selects the best action most of the time, but occasionally explores:
$$\pi(a|s) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|\mathcal{A}(s)|} & \text{if } a = \arg\max_{a'} Q(s,a') \\ \frac{\epsilon}{|\mathcal{A}(s)|} & \text{otherwise} \end{cases} \tag{11}$$
- **Softmax Policy**: Selects actions with probabilities proportional to their estimated values:
$$\pi(a|s) = \frac{e^{Q(s,a)/\tau}}{\sum_{a'} e^{Q(s,a')/\tau}} \tag{12}$$
where τ is a temperature parameter controlling exploration.

3.6 Example: Contextual Multi-Armed Bandit

Consider a modified version of the multi-armed bandit problem where there are visible weather conditions (sunny, rainy, cloudy) that affect the rewards for each arm (e.g., different types of investments). The rewards for each arm depend on the weather, but today's action doesn't affect tomorrow's weather.

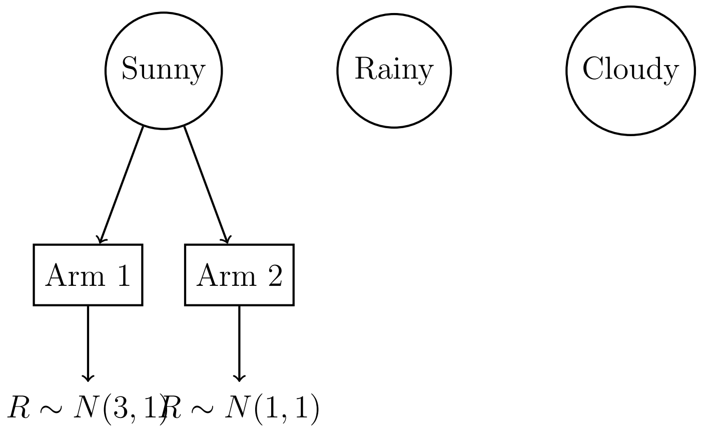

Figure 2: A simplified contextual bandit example with weather states

The key insight is that while the agent's action choice depends on the observed state (weather), and the rewards depend on both state and action, the agent's actions do not influence which state occurs next.

4 Full RL with State Transitions

4.1 The Critical Addition: Actions Affect Future States

The defining characteristic of full reinforcement learning is that actions not only influence immediate rewards but also affect future states, and through them, future rewards:

$$p(s_{t+1}|s_t, a_t) \tag{13}$$

Notation Overview

- $p(s_{t+1}|s_t, a_t)$ - The probability of transitioning to state s_{t+1} given the current state s_t and action a_t
- s_t - The state at time step t
- a_t - The action taken at time step t
- s_{t+1} - The next state at time step $t+1$

This introduces a critical complexity: actions have long-term consequences through their influence on future states. The agent must consider not just immediate rewards but also how actions shape future opportunities.

4.2 Markov Decision Processes (MDPs)

Full reinforcement learning problems are typically formalized as Markov Decision Processes (MDPs), which are defined by:

- A set of states $\mathcal{S}$
- A set of actions $\mathcal{A}$
- A reward function $r(s, a) = \mathbb{E}[R_{t+1}|S_t = s, A_t = a]$
- A state transition probability function $p(s'|s, a) = P(S_{t+1} = s'|S_t = s, A_t = a)$
- A discount factor $\gamma \in [0, 1]$

The dynamics of an MDP are formally defined by:

$$p(s', r|s, a) = P(S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a) \tag{14}$$

Notation Overview

- $p(s', r|s, a)$ - The probability of transitioning to state s' and receiving reward r, given that the agent was in state s and took action a
- $P(S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a)$ - The probability that the state at time t is s' and the reward is r, given that the state at time $t-1$ was s and the action taken was a
- S_t - The state at time step t
- R_t - The reward received at time step t
- A_t - The action taken at time step t

From this joint probability function, we can derive other important quantities:

$$p(s'|s, a) = \sum_{r \in \mathcal{R}} p(s', r|s, a) \tag{15}$$

$$r(s, a) = \mathbb{E}[R_t|S_{t-1} = s, A_{t-1} = a] = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r|scla, a) \tag{16}$$

$$r(s,a,s') = \mathbb{E}[R_t | S_{t-1} = s, A_{t-1} = a, S_t = s'] = \sum_{r \in \mathcal{R}} r \frac{p(s', r|s, a)}{p(s'|s, a)} \tag{17}$$

Notation Overview

- $p(s'|s,a)$ - The probability of transitioning to state s' given state s and action a
- $r(s,a)$ - The expected reward when taking action a in state s
- $r(s,a,s')$ - The expected reward when transitioning from state s to state s' via action a
- $\mathcal{R}$ - The set of possible rewards
- $\mathcal{S}$ - The set of possible states

4.3 Episodic vs. Continuing Tasks

Reinforcement learning tasks can be categorized as either episodic or continuing:

- **Episodic Tasks**: These have a clear endpoint or terminal state. For example, a game of chess ends with either a win, loss, or draw. The sequence of states, actions, and rewards from the start to the terminal state is called an episode.
- **Continuing Tasks**: These go on indefinitely without a natural endpoint. For example, an ongoing process control system or a stock trading agent that operates continuously.

4.4 The Return and Value Functions in Full RL

In full RL, the objective is to maximize the expected return, which is the cumulative discounted reward:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{18}$$

Notation Overview

- G_t - The return at time step t
- R_{t+k} - The reward received k steps after time t
- γ - The discount factor, $0 \leq \gamma \leq 1$

For episodic tasks with a clear terminal state, we can alternatively define the return as:

$$G_t = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1} \tag{19}$$

where T is the final time step of the episode.
To unify the notation for both episodic and continuing tasks, we can use:

$$G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k \tag{20}$$

with the understanding that $T = \infty$ for continuing tasks, and $\gamma < 1$ for continuing tasks to ensure the sum is finite.

4.5 State-Value and Action-Value Functions

The state-value function represents the expected return when starting in state s and following policy π:

$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \middle| S_t = s\right] \tag{21}$$

Notation Overview

- $v_\pi(s)$ - The state-value function for policy π
- $\mathbb{E}_\pi[\cdot]$ - The expected value when following policy π
- G_t - The return at time step t
- S_t - The state at time step t
- R_{t+k+1} - The reward received $k+1$ steps after time t
- γ - The discount factor, $0 \leq \gamma \leq 1$

Similarly, the action-value function represents the expected return when starting in state s, taking action a, and thereafter following policy π:

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}\middle|S_t = s, A_t = a\right] \quad (22)$$

Notation Overview

- $q_\pi(s,a)$ - The action-value function for policy π
- $\mathbb{E}_\pi[\cdot]$ - The expected value when following policy π
- G_t - The return at time step t
- S_t - The state at time step t
- A_t - The action taken at time step t
- R_{t+k+1} - The reward received $k+1$ steps after time t
- γ - The discount factor, $0 \leq \gamma \leq 1$

4.6 The Relationship Between State-Value and Action-Value Functions

The state-value and action-value functions are related by:

$$v_\pi(s) = \sum_a \pi(a|s)q_\pi(s,a) \quad (23)$$

This shows that the value of a state is the expected value of the actions that might be taken in that state, weighted by their probability under policy π.

Conversely, we can express the action-value in terms of the state-value:

$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] \quad (24)$$

This indicates that the value of taking action a in state s is the expected immediate reward plus the discounted value of the next state.

4.7 The Bellman Equations

The recursive relationship between value functions is captured by the Bellman equations. For the state-value function:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')] \tag{25}$$

Notation Overview

- $v_\pi(s)$ - The state-value function for policy π
- $\pi(a|s)$ - The probability of selecting action a in state s under policy π
- $p(s', r|s, a)$ - The probability of transitioning to state s' and receiving reward r, given state s and action a
- r - The reward
- γ - The discount factor, $0 \leq \gamma \leq 1$

And for the action-value function:

$$q_\pi(s, a) = \sum_{s',r} p(s', r|s, a) \left[r + \gamma \sum_{a'} \pi(a'|s') q_\pi(s', a') \right] \tag{26}$$

Notation Overview

- $q_\pi(s, a)$ - The action-value function for policy π
- $p(s', r|s, a)$ - The probability of transitioning to state s' and receiving reward r, given state s and action a
- $\pi(a'|s')$ - The probability of selecting action a' in state s' under policy π
- r - The reward
- γ - The discount factor, $0 \leq \gamma \leq 1$

The Bellman equations express a fundamental property: the value of a state (or state-action pair) equals the expected immediate reward plus the discounted value of the next state (or states).

4.8 Optimal Value Functions and Policies

A policy π is defined as better than or equal to a policy π' if its expected return is greater than or equal to that of π' for all states:

$$\pi \geq \pi' \iff v_\pi(s) \geq v_{\pi'}(s) \text{ for all } s \in \mathcal{S} \tag{27}$$

There is always at least one policy that is better than or equal to all other policies, called an optimal policy, denoted π_*. All optimal policies share the same optimal state-value function, v_*:

$$v_*(s) = \max_\pi v_\pi(s) \text{ for all } s \in \mathcal{S} \tag{28}$$

They also share the same optimal action-value function, q_*:

$$q_*(s,a) = \max_\pi q_\pi(s,a) \text{ for all } s \in \mathcal{S}, a \in \mathcal{A}(s) \tag{29}$$

Notation Overview

- $\pi \geq \pi'$ - Policy π is better than or equal to policy π'
- $v_\pi(s)$ - The state-value function for policy π
- $v_*(s)$ - The optimal state-value function
- $q_*(s,a)$ - The optimal action-value function
- $\max_\pi$ - The maximum over all possible policies
- $\mathcal{S}$ - The set of all possible states
- $\mathcal{A}(s)$ - The set of actions available in state s

4.9 The Bellman Optimality Equations

The Bellman optimality equations characterize the optimal value functions:

$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')] \tag{30}$$

$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma \max_{a'} q_*(s',a')] \tag{31}$$

Notation Overview

- $v_*(s)$ - The optimal state-value function
- $q_*(s, a)$ - The optimal action-value function
- $\max_a$ - The maximum over all actions $a \in \mathcal{A}(s)$
- $\max_{a'}$ - The maximum over all actions $a' \in \mathcal{A}(s')$
- $p(s', r|s, a)$ - The probability of transitioning to state s' and receiving reward r, given state s and action a
- r - The reward
- γ - The discount factor, $0 \leq \gamma \leq 1$

Given the optimal value functions, it is straightforward to determine an optimal policy:

$$\pi_*(s) = \arg\max_a q_*(s, a) \tag{32}$$

4.10 The Credit Assignment Problem

One of the fundamental challenges introduced by state transitions in full RL is the credit assignment problem: determining which actions in a sequence were responsible for a later reward.

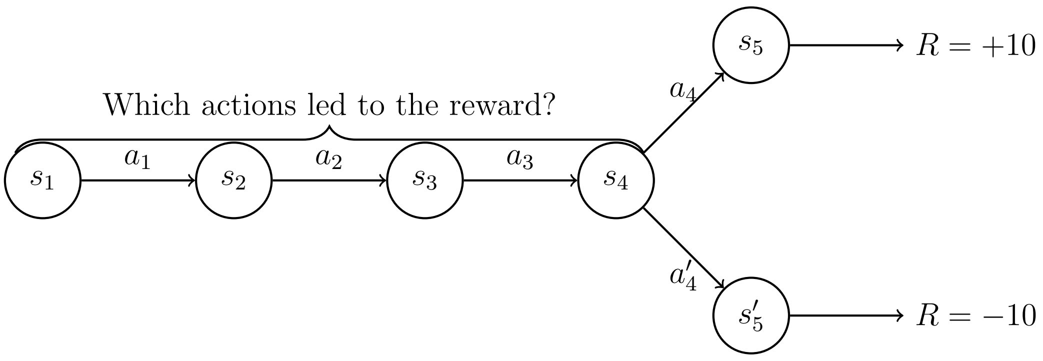

Figure 3: The credit assignment problem: determining which actions in a sequence contributed to the final reward

In the diagram, a positive reward is received after reaching state s_5, and a negative reward after reaching state s'_5. The challenge is to determine which of the earlier actions in the sequence (a, a, a, a/a') were most responsible for the final outcome.

This problem becomes particularly difficult when:

- Rewards are delayed (occur many steps after the critical actions)
- Actions have long-term consequences through complex state transitions
- The environment is noisy or stochastic

Reinforcement learning algorithms address this problem through various methods:

- **Temporal Difference Learning**: Updates value estimates based on the difference between successive predictions
- **Eligibility Traces**: Maintain a record of which states and actions have been visited recently and are thus "eligible" for updates
- **Monte Carlo Methods**: Use complete episode returns to update action values

4.11 Comparison: Bandits vs. Contextual Bandits vs. Full RL

Feature	**Bandits**	**Contextual Bandits**	**Full RL**
States	No	Yes	Yes
Reward depends on state	No	Yes	Yes
Actions affect future states	No	No	Yes
Time horizon	Immediate	Immediate	Multiple steps
Value function	$Q(a)$	$Q(s,a)$	$q_\pi(s,a), v_\pi(s)$
Considers future rewards	No	No	Yes
Credit assignment	Simple	Simple	Complex
Optimal policy computation	Easy	Moderate	Hard
Exploration-exploitation	Simple	Moderate	Complex

Table 1: Comparison of key features across bandit problems, contextual bandits, and full RL

5 Conclusion

The transition from bandits to full reinforcement learning represents a significant increase in complexity and expressive power:

- **Bandits** involve only actions and immediate rewards with no concept of state.
- **Contextual Bandits** introduce states that influence rewards, but actions don't affect future states.
- **Full RL** incorporates state transitions influenced by actions, creating sequential decision problems where actions have long-term consequences.

This progression captures the core challenge of reinforcement learning: making decisions that optimize long-term cumulative reward when actions influence not just immediate rewards but also future states and opportunities.

The key concepts introduced in full reinforcement learning include:

- The Markov property, which allows decisions to be based solely on the current state
- Value functions, which estimate the expected future reward
- The Bellman equations, which express the recursive relationship between values of different states
- Optimal policies, which maximize the expected return
- The credit assignment problem, which involves determining which actions led to observed rewards

Understanding these fundamental concepts provides the foundation for more advanced reinforcement learning methods, including temporal difference learning, Monte Carlo methods, and function approximation techniques.

6 Summary of Key Equations

Here is a summary of all the key equations covered in this document:

6.1 Reward Dependencies

$$\text{Bandits:} \qquad p(r_t|a_t) \tag{33}$$
$$\text{Contextual Bandits:} \qquad p(r_t|s_t, a_t) \tag{34}$$
$$\text{Full RL:} \qquad p(r_t|s_t, a_t) \text{ and } p(s_{t+1}|s_t, a_t) \tag{35}$$

6.2 MDP Dynamics

Joint Probability:
$$p(s', r|s, a) = P(S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a) \tag{36}$$

State Transition:
$$p(s'|s, a) = \sum_{r \in \mathcal{R}} p(s', r|s, a) \tag{37}$$

Expected Reward:
$$r(s, a) = \sum_{r \in \mathcal{R}} r \sum_{s' \in \mathcal{S}} p(s', r|s, a) \tag{38}$$

Expected Reward with Next State:
$$r(s, a, s') = \sum_{r \in \mathcal{R}} r \frac{p(s', r|s, a)}{p(s'|s, a)} \tag{39}$$

6.3 Return

Continuing Tasks:
$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{40}$$

Episodic Tasks:
$$G_t = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1} \tag{41}$$

Unified Notation:
$$G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k \tag{42}$$

6.4 Value Functions

State-Value Function:
$$v_\pi(s) = \mathbb{E}_\pi[G_t|S_t = s] \tag{43}$$

Action-Value Function:
$$q_\pi(s, a) = \mathbb{E}_\pi[G_t|S_t = s, A_t = a] \tag{44}$$

Relationship:
$$v_\pi(s) = \sum_a \pi(a|s) q_\pi(s, a) \tag{45}$$

Inverse Relationship:
$$q_\pi(s, a) = \sum_{s', r} p(s', r|s, a)[r + \gamma v_\pi(s')] \tag{46}$$

6.5 Bellman Equations

For State-Value:
$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)[r + \gamma v_\pi(s')] \tag{47}$$

For Action-Value:
$$q_\pi(s,a) = \sum_{s',r} p(s',r|s,a) \left[r + \gamma \sum_{a'} \pi(a'|s') q_\pi(s',a') \right] \tag{48}$$

6.6 Optimal Value Functions

Optimal State-Value:
$$v_*(s) = \max_\pi v_\pi(s) \tag{49}$$

Optimal Action-Value:
$$q_*(s,a) = \max_\pi q_\pi(s,a) \tag{50}$$

6.7 Bellman Optimality Equations

For State-Value:
$$v_*(s) = \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma v_*(s')] \tag{51}$$

For Action-Value:
$$q_*(s,a) = \sum_{s',r} p(s',r|s,a)[r + \gamma \max_{a'} q_*(s',a')] \tag{52}$$

6.8 Policies

Policy Definition:
$$\pi(a|s) = P(A_t = a|S_t = s) \tag{53}$$

Greedy Policy:
$$\pi(a|s) = \begin{cases} 1 & \text{if } a = \arg\max_{a'} Q(s,a') \\ 0 & \text{otherwise} \end{cases} \tag{54}$$

ϵ-Greedy Policy:
$$\pi(a|s) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|\mathcal{A}(s)|} & \text{if } a = \arg\max_{a'} Q(s,a') \\ \frac{\epsilon}{|\mathcal{A}(s)|} & \text{otherwise} \end{cases} \tag{55}$$

Softmax Policy:
$$\pi(a|s) = \frac{e^{Q(s,a)/\tau}}{\sum_{a'} e^{Q(s,a')/\tau}} \tag{56}$$

Optimal Policy:
$$\pi_*(s) = \arg\max_a q_*(s,a) \tag{57}$$

6.9 Estimating Action Values

$$\text{Averaging Approach:} \quad Q_t(s,a) = \frac{1}{n_t(s,a)} \sum_{i=1}^{t} r_i \cdot \mathbb{I}(s_i = s, a_i = a) \tag{58}$$

$$\text{Incremental Update:} \quad Q_{t+1}(s_t,a_t) = Q_t(s_t,a_t) + \alpha[r_{t+1} - Q_t(s_t,a_t)] \tag{59}$$