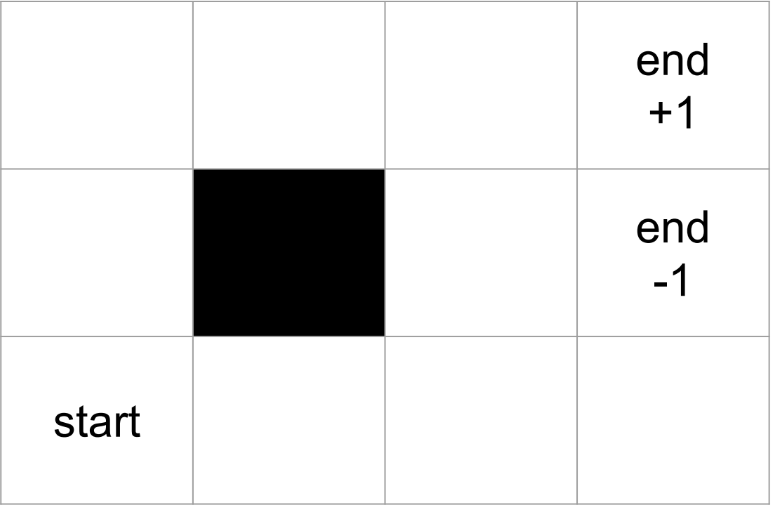
Analysis of States and Actions in the

GridWorld Environment

# Introduction

The document explains all aspects regarding the states and actions included in the GridWorld simulation found in StandardAndNegativeGrids and ValuesAndAPolicy Jupyter Notebooks. This analysis adopts the fundamental notation and concepts defined in the Reinforcement Learning Concepts markdown (MSDS684\_2/DP, 2023) document. Markov Decision Processes (MDPs) become real through the GridWorld environment, which enables an agent to discover optimal sequential decision making. For solving the reinforcement learning problem in this environment, it is essential to grasp the exact definitions of states (s) and actions (a).



# State Space Definition (𝑆)

## General Concept

Reinforcement Learning, alongside MDPs defines states (s) as the complete environmental information needed for agents to select their decisions at a specific time. The state space (S) contains every potential state that the agent might occupy. For any state s the condition 𝑠 ∈ 𝑆 applies formally (Jagtap, 2024).

## State Representation in GridWorld

In the specific GridWorld environment under consideration: A state 𝑠 corresponds to the agent’s physical location on a grid. This location is represented by a coordinate pair (𝑟𝑜𝑤, 𝑐𝑜𝑙𝑢𝑚𝑛), where row is the row index and column is the column index. These are implemented as Python tuples. The grid dimensions are 3 rows (indexed 0, 1, 2) and 4 columns (indexed 0, 1, 2, 3).

## Grid Coordinates and State Types

The grid comprises 3 × 4 = 12 unique coordinate locations:

(0, 0), (0, 1), (0, 2), (0, 3) (1, 0), (1, 1), (1, 2), (1, 3) (2, 0), (2, 1), (2, 2), (2, 3)

Based on the notebook outputs (gw.print\_grid\_state()), certain states have special properties:

* + - **Initial State:** The designated starting location for the agent in every episode.

– 𝑠𝑖𝑛𝑖𝑡𝑖𝑎𝑙 = (0, 0)

* + - **Barrier State:** A location the agent cannot enter or occupy. It acts as an obstacle, influencing possible transitions from adjacent states.
      * Barrier location: (1, 1)
      * This location is identified by is\_barrier: True in the print\_grid\_state() output. Crucially, while (1, 1) exists conceptually on the grid layout (Silver, 2017), it is *not* an element of the set of states 𝑆 that the agent can occupy.
    - **Terminal States:** States that, upon entry, terminate the current episode. No further actions can be initiated from these states. They correspond to the concept of boundary states (𝜕𝑆) mentioned in the RL concepts document.
      * Terminal locations: (1, 3) and (2, 3)
      * These locations are identified by is\_terminal: True in the

print\_grid\_state() output.

## Formal State Space 𝑆

The effective state space 𝑆 for this GridWorld includes all grid coordinates *except* the barrier state. Terminal states are part of the state space, representing achievable end-points of an episode.

Therefore, the state space 𝑆 is formally defined as the set:

𝑆 = {(0, 0), (0, 1), (0, 2), (0, 3), (1, 0), (1, 2), (2, 0), (2, 1), (2, 2), (1, 3), (2, 3)}

This set contains 11 distinct states: 9 non-terminal, non-barrier states and 2 terminal states. The coordinate (1, 1) is excluded.

## Markov Property

The state representation (𝑟𝑜𝑤, 𝑐𝑜𝑙𝑢𝑚𝑛) is sufficient to determine the outcome (next state and reward) of any available action. The history of how the agent arrived at state 𝑠 does not provide additional information relevant to future transitions or rewards. Therefore, this GridWorld environment adheres to the Markov property, a prerequisite for standard MDP solution techniques. (OpenAI, 2021)

# Action Space Definition ( )

𝑠

## General Concept

An **action** (𝑎), often referred to as a decision, is a choice the agent can make when in a particular state 𝑠. The set 𝐴𝑠 denotes the specific **set of actions available** to the agent when it is in state 𝑠. This set can differ between states.

## Available Actions in GridWorld

The fundamental actions the agent can attempt correspond to moving one cell in the four cardinal directions:

1. 'up' : Move to the cell in the row above (decrease row index).
2. 'down' : Move to the cell in the row below (increase row index).
3. 'left' : Move to the cell in the column to the left (decrease column index).
4. 'right' : Move to the cell in the column to the right (increase column index).

## State-Dependent Action Sets ( )

𝑠

A critical feature of this GridWorld is that the set of available actions 𝐴 is

𝑠

contingent upon the current state 𝑠. The constraints are:

* + - **Grid Boundaries:** Actions that would move the agent off the 3x4 grid are disallowed. For instance, from state 𝑠 = (0, 0), the actions 'left' and 'down' are not available.
    - **Barrier State:** Actions that would result in transitioning *into* the barrier state (1, 1) are disallowed. For instance, from state 𝑠 = (1, 0), the action 'right' is unavailable, and from state 𝑠 = (0, 1), the action 'down' is unavailable.
    - **Terminal States:** No actions can be initiated from a terminal state. If 𝑠 is a

terminal state (i.e., 𝑠 ∈ {(1, 3), (2, 3)}), then the action set is empty: 𝐴 = ∅.

𝑠

The unavailability of specific actions from a given state is indicated by a None reward value for that action in the gw.print\_grid\_rewards() output provided in the StandardAndNegativeGrids notebook.

## Explicit Action Sets per State

The action sets 𝐴 for each state 𝑠 ∈ 𝑆 are explicitly defined as follows:

𝑠

* + - For non-terminal states:

– 𝐴 = {’𝑟𝑖𝑔ℎ𝑡’, ’𝑢𝑝’}

(0,0)

– 𝐴 = {’𝑙𝑒𝑓𝑡’, ’𝑟𝑖𝑔ℎ𝑡’}

(0,1)

– 𝐴 = {’𝑙𝑒𝑓𝑡’, ’𝑟𝑖𝑔ℎ𝑡’, ’𝑢𝑝’}

(0,2)

– 𝐴 = {’𝑙𝑒𝑓𝑡’, ’𝑢𝑝’}

(0,3)

– 𝐴 = {’𝑢𝑝’, ’𝑑𝑜𝑤𝑛’}

(1,0)

– 𝐴 = {’𝑟𝑖𝑔ℎ𝑡’, ’𝑑𝑜𝑤𝑛’, ’𝑢𝑝’}

(1,2)

– 𝐴 = {’𝑟𝑖𝑔ℎ𝑡’, ’𝑢𝑝’}

(2,0)

– 𝐴 = {’𝑙𝑒𝑓𝑡’, ’𝑟𝑖𝑔ℎ𝑡’}

(2,1)

– 𝐴 = {’𝑙𝑒𝑓𝑡’, ’𝑟𝑖𝑔ℎ𝑡’, ’𝑢𝑝’}

(2,2)

* + - For terminal states:
      * 𝐴 = ∅ (Empty set)

(1,3)

* + - * 𝐴 = ∅ (Empty set)

(2,3)

Note: The state 𝑠 = (1, 1) is not included as it is a barrier and thus unreachable; therefore, 𝐴 is undefined in the context of reachable states.

(1,1)

## Actions, Transitions, and Rewards

In this GridWorld environment, the transitions are deterministic. Selecting an action 𝑎 ∈ 𝐴 from a non-terminal state 𝑠 uniquely determines the next state 𝑠′.

𝑠

For example, taking 𝑎 = ’𝑟𝑖𝑔ℎ𝑡’ from 𝑠 = (0, 0) always leads to 𝑠′ = (0, 1).

Furthermore, executing action 𝑎 in state 𝑠 yields an immediate **reward**,

denoted 𝑟 or more explicitly 𝑟(𝑠, 𝑎). The specific numerical value of the reward depends on the (𝑠, 𝑎) pair and whether the “Standard Grid” or “Negative Grid” configuration is used, as detailed in the gw.print\_grid\_rewards() outputs.

For instance:

In the Standard Grid,

((2, 2), ’𝑟𝑖𝑔ℎ𝑡’) = 1. 0, leading to terminal state (2, 3).

Most other valid moves yield 𝑟 = 0. 0.

In the Negative Grid,

((2, 2), ’𝑟𝑖𝑔ℎ𝑡’) = 1. 0.

Most other valid moves yield 𝑟 =− 0. 1.

Moving into the other terminal state,

((0, 3), ’𝑢𝑝’) =− 1. 0 and ((1, 2), ’𝑟𝑖𝑔ℎ𝑡’) =− 1. 0, leading to state

(1, 3) in both grid types.

A **policy** (π) is a mapping from states to actions,

π: 𝑆 → 𝐴

The ValuesAndAPolicy notebook demonstrates a policy represented as a Python dictionary where keys are state tuples (𝑟𝑜𝑤, 𝑐𝑜𝑙𝑢𝑚𝑛) and values are action strings (e.g., 'up', 'right'). The objective in this MDP is typically to find an optimal policy

\*

π that maximizes the expected cumulative reward.

# Conclusion

The GridWorld environment provides a concrete example of an MDP. Its **states** (𝑠 ∈ 𝑆) are defined by the agent’s (row, column) coordinates on a 3x4 grid, excluding the barrier location (1, 1) and including two terminal states (1, 3) and

(2, 3). The initial state is (0, 0). The **actions** (𝑎 ∈ 𝐴𝑠) available are movements

('up', 'down', 'left', 'right'), but the permissible set 𝐴𝑠 is specific to each state

𝑠, constrained by grid boundaries and the barrier. No actions are possible from terminal states. This precise definition of 𝑆 and 𝐴𝑠, along with the reward function

(𝑠, 𝑎) and transition dynamics, fully specifies the GridWorld MDP.

**References**

1. MSDS684\_2/DP. (2023).Reinforcement Learning Concepts.md. <https://github.com/doug57/MSDS684_2/blob/main/DP/Reinforcement%20Learning%20Concepts.md>

2. Rohan Jagtap. (2024). Understanding the Markov Decision Process (MDP).

<https://builtin.com/machine-learning/markov-decision-process>

3. David Silver. (2017). RL Course by David Silver - Lecture 1: Introduction to Reinforcement Learning. [Video]

YouTube. <https://www.youtube.com/watch?v=2pWv7GOvuf0&list=PLqYmG7hTraZDM-OYHWgPebj2MfCFzFObQ&index=1>

4. OpenAI. (2021). GPT-3: Language model.

<https://openai.com/chat>