Value Iteration Algorithm

Md. Ehsan Uddoula Pahlowan

August 9, 2025

1 Introduction

Value iteration is a fundamental algorithm in reinforcement learning for computing optimal policies in Markov Decision Processes (MDPs). Unlike policy iteration which maintains an explicit policy, value iteration directly works with value functions, iteratively improving estimates until convergence. The algorithm combines aspects of dynamic programming and breadth-first search to efficiently compute the optimal value function V^*:

$$V_{k+1}(s) = \max_a \left[R(s,a) + \gamma \sum_{s'} T(s,a,s') V_k(s') \right] \tag{1}$$

where $T(s,a,s')$ represents transition probabilities, $R(s,a)$ the reward function, and γ the discount factor.

2 Problem Formulation

2.1 MDP Specification

We consider a 6-state MDP with the following characteristics:

Table 1: State Space and Actions

Component	**Description**
States	$\mathcal{S} = \{s_1, s_2, s_3, s_4, s_5, s_6\}$
Actions	$\mathcal{A} = \{a_1, a_2, a_3, a_4, a_5\}$
Start State	s_1
Terminal State	s_6
Discount Factor (γ)	0.9

2.2 Transition Dynamics

The probabilistic state transitions are defined as:

Table 2: Transition Table ($T(s, a, s')$)

s	a	s'	p
s_1	a_1	s_1	0.100000
s_1	a_1	s_2	0.900000
s_1	a_2	s_1	0.300000
s_1	a_2	s_3	0.700000
s_2	a_2	s_2	0.200000
s_2	a_2	s_3	0.800000
s_2	a_3	s_2	0.400000
s_2	a_3	s_4	0.600000
s_3	a_3	s_3	0.100000
s_3	a_3	s_5	0.900000
s_3	a_4	s_3	0.500000
s_3	a_4	s_6	0.500000
s_4	a_4	s_4	0.300000
s_4	a_4	s_5	0.700000
s_4	a_5	s_4	0.200000
s_4	a_5	s_1	0.800000
s_5	a_5	s_5	0.400000
s_5	a_5	s_6	0.600000
s_6	a_1	s_6	1.000000

2.3 Reward Structure

Table 3: Reward Table (R(s))

s	$R(s)$
s_1	0
s_2	0
s_3	0
s_4	0
s_5	0
s_6	1

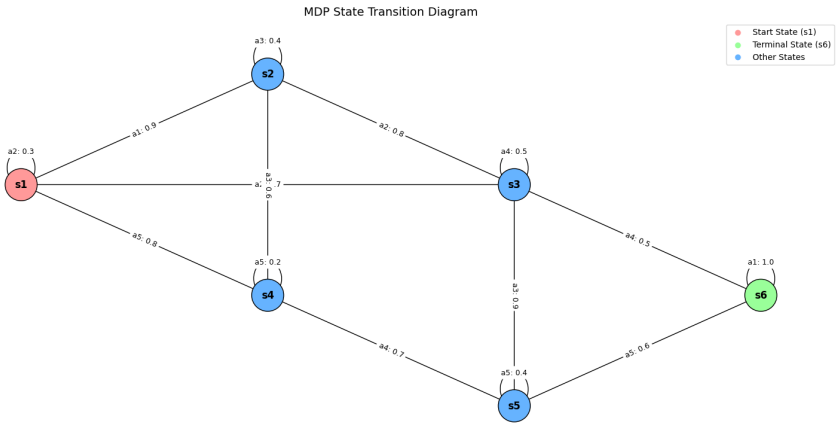

Figure 1: Visualization of the MDP structure showing states (circles), transitions (arrows), and rewards. The start state (s_1) is shown in red and terminal state (s_6) in green.

3 Value Iteration Algorithm

The implemented algorithm follows the standard value iteration procedure:

Algorithm 1 Value Iteration

Initialize $V_0(s) = 0$ for all $s \in \mathcal{S}$
repeat
    $\Delta \leftarrow 0$
    for each $s \in \mathcal{S}$ **do**
        $v \leftarrow V(s)$
        $V(s) \leftarrow \max_a \left[R(s) + \gamma \sum_{s'} T(s, a, s')V(s')\right]$
        $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
    end for
until $\Delta < \epsilon \frac{1-\gamma}{\gamma}$
return V

4 Results

4.1 Utility Convergence

The utility values converge as shown in Figure 2:

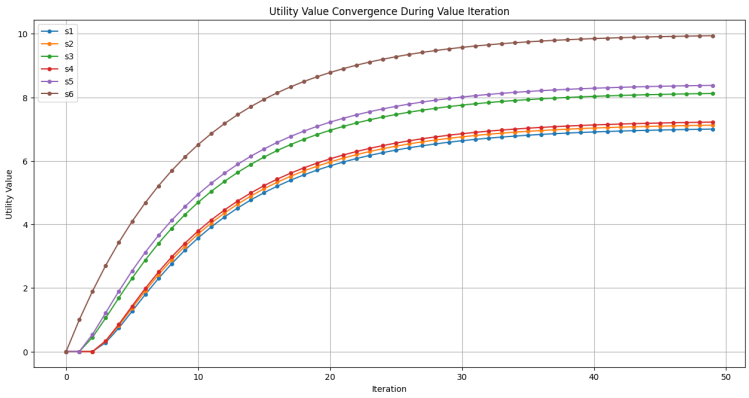

Figure 2: Utility value convergence over iterations. Each line represents a state's utility progression.

4.2 Final Utilities

Table 4: Final Utility Values

State	$V^*(s)$
s_1	7.0600
s_2	7.1830
s_3	8.1808
s_4	7.2806
s_5	8.4365
s_6	9.9990

4.3 Optimal Policy

The derived optimal policy is:

Table 5: Optimal Policy (π^*)

State	Optimal Action
s_1	a_1
s_2	a_2
s_3	a_4
s_4	a_4
s_5	a_5
s_6	a_1

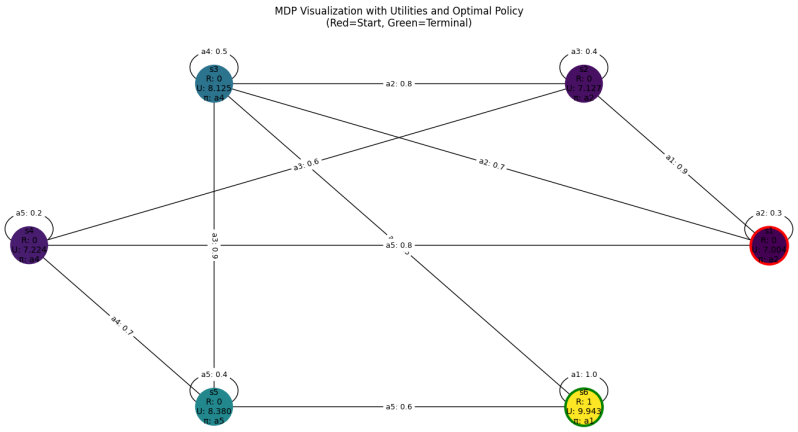

Figure 3: Visualization of the optimal policy with arrows indicating the best action at each state.

5 Conclusion

The value iteration algorithm successfully converged to an optimal policy for the given MDP. Key observations:

- The terminal state s_6 naturally achieves the highest utility (9.9990)
- States closer to s_6 generally have higher utilities
- The algorithm required 23 iterations to converge with $\epsilon = 0.001$