# Infinite-Horizon Problems

# Preliminaries

- In a stationary infinite-horizon Markov decision process, each policy $\pi = (d_1, d_2, \ldots)$ induces a bivariate discrete-time reward process $\{r(X_t, d_t(X_t)), \quad t = 1, 2, \ldots\}$.
- The *expected total reward* of policy $\pi$, $V^\pi(s)$ is defined to be

$$
\begin{aligned}
V^\pi(s) &= \lim_{N\to\infty} \mathbb{E}_s^\pi \left\{ \sum_{t=1}^{N} r(X_t, d_t(X_t)) \right\} \\
&= \lim_{N\to\infty} V_{N+1}^\pi(s).
\end{aligned}
$$

- In some models, the limit may not exist. When the limit exists and when interchanging the limit and expectation is valid, we write

$$
V^\pi(s) = \mathbb{E}_s^\pi \left\{ \sum_{t=1}^{\infty} r(X_t, d_t(X_t)) \right\}.
$$

- The *expected total discounted reward* of policy $\pi$, $V_\lambda^\pi(s)$ is defined to be

$$V_\lambda^\pi(s) \quad = \quad \lim_{N\to\infty} \mathbb{E}_s^\pi \left\{ \sum_{t=1}^{N} \lambda^{t-1} r(X_t, d_t(X_t)) \right\},$$

for $0 \le \lambda < 1$. The limit exists when

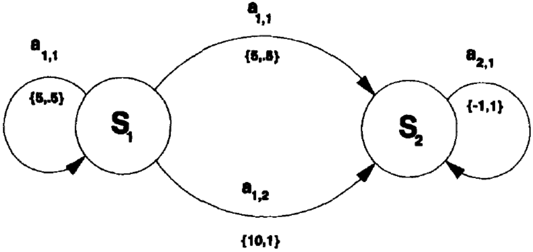
$$\sup_{s\in S} \sup_{a\in A_s} |r(s,a)| = M < \infty.$$

- When the limit exists and interchaning the limit and expectation are valid, we write

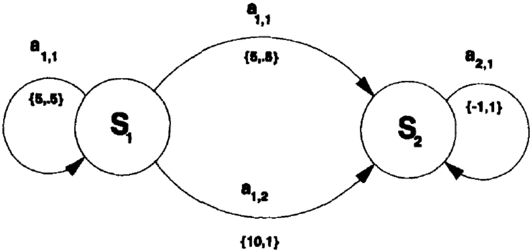
$$V_\lambda^\pi(s) \quad = \quad \mathbb{E}_s^\pi \left\{ \sum_{t=1}^{\infty} \lambda^{t-1} r(X_t, d_t(X_t)) \right\}.$$

# THE VALUE OF A POLICY

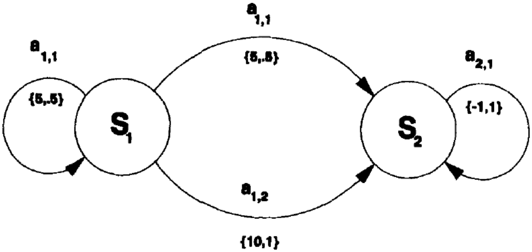
- We use the notation $d^{\infty} = (d, d, \dots)$ to denote the policy that decision rule $d$ is used in each period.




- *Decision Epochs*: $T = \{1, 2, \dots, N\}, \quad N \leq \infty$.
- *States*: $S = \{s_1, s_2\}$
- *Actions*: $A_{s_1} = \{a_{1,1}, a_{1,2}\}$, $A_{s_2} = \{a_{2,1}\}$




- *Rewards*: $r_t(s_1, a_{1,1}) = 5$, $r_t(s_1, a_{1,2}) = 10$, $r_t(s_2, a_{2,1}) = -1$, $r_N(s_1) = r_N(s_2) = 0$ if $N < \infty$
- Transition Probabilities:
  $p_t(s_1 \mid s_1, a_{1,1}) = 0.5$, $p_t(s_2 \mid s_1, a_{1,1}) = 0.5$
  $p_t(s_1 \mid s_1, a_{1,2}) = 0$, $p_t(s_2 \mid s_1, a_{1,2}) = 1$
  $p_t(s_1 \mid s_2, a_{2,1}) = 0$, $p_t(s_2 \mid s_2, a_{2,1}) = 1$




- Suppose there are two decision rules $d$ and $e$, where $d(s_1) = a_{1,1}$, $e(s_1) = a_{1,2}$ and $d(s_2) = e(s_2) = a_{2,1}$. Based on the decision rules, compute $V_N^{e^\infty}(s)$, $V_N^{d^\infty}(s)$, $V_\lambda^{e^\infty}(s)$ and $V_\lambda^{d^\infty}(s)$ for $s = s_1, s_2$.

# Policy Evaluation

- If $\pi = (d, d, \dots)$, then the value function

$$V_\lambda^\pi(s) = r(s,d) + \lambda \sum_{j \in S} p(j \mid s, d) V_\lambda^\pi(j),$$

or

$$V_\lambda^\pi = r_d + \lambda P_d V_\lambda^\pi.$$

- Define the linear transformation $L_d$ by

$$L_d V \equiv r_d + \lambda P_d V.$$

In this notation, the DP equation becomes

$$V_\lambda^\pi = L_d V_\lambda^\pi.$$

This means that $V_\lambda^\pi$ is a fixed point of $L_d$.

# Policy Evaluation

- The above discussion leads to the following important result: $V_\lambda^\pi$ is the unique solution of

$$V = r_d + \lambda P_d V.$$

Further, $V_\lambda^\pi$ may be written as

$$V_\lambda^\pi = (I - \lambda P_d)^{-1} r_d.$$

# Optimality Equation

- We define operator $L$ by

$$LV = \max_d \{r_d + \lambda P_d V\}.$$

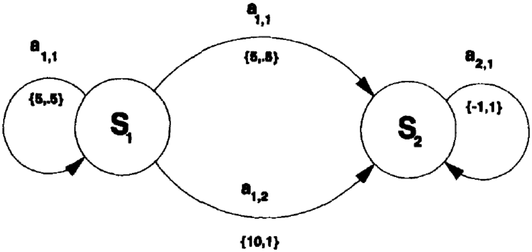
- We now represent the optimality equations in vector notation that

$$V = \max_d \{r_d + \lambda P_d V\} = LV.$$

- In component notation the optimality equations then become

$$V(s) = \max_a \{r(s,a) + \lambda \sum_{j \in S} p(j \mid s,a) V(j)\}.$$

# EXAMPLE




- The optimality equations are

1. Set $V_0 = 0$, specify $\epsilon > 0$, and set $n = 0$ for all $s \in S$
2. For each $s \in S$, compute $V_{n+1}(s)$ by

$$V_{n+1}(s) = \max_{a \in A_s}\{r(s,a) + \lambda \sum_{j \in S} p(j \mid s,a) V_n(j)\}$$

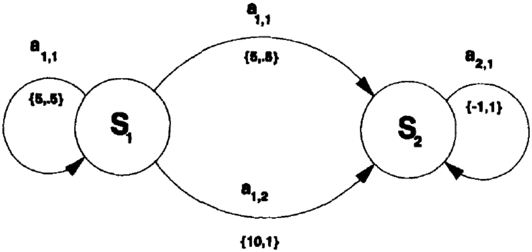
3. If

$$\|V_{n+1} - V_n\| < \epsilon(1-\lambda)/2\lambda,$$

   go to step 4. Otherwise increment $n$ by 1 and return to step 2.
4. For each $s \in S$, choose

$$d(s) = \operatorname*{argmax}_{a \in A_s}\{r(s,a) + \lambda \sum_{j \in S} p(j \mid s,a) V_{n+1}(j)\}$$

   and stop.




- Use value iteration to solve this model. We set $\epsilon = 0.01$, $\lambda = 0.95$, and choose $V(s_1) = V_0(s_2) = 0$. The recursions become

$$
\begin{aligned}
V_{n+1}(s_1) &= \max\{5 + 0.5\lambda V_n(s_1) + 0.5\lambda V_n(s_2), 10 + \lambda V_n(s_2)\} \\
V_{n+1}(s_2) &= -1 + \lambda V_n(s_2).
\end{aligned}
$$

- It requires 162 iterations to satisfy stopping criterion

$$\|V_{n+1} - V_n\| < 0.01 \cdot 0.05/1.9 = 0.00026$$

| $n$ | $v^n(s_1)$ | $v^n(s_2)$ | $\|v^n - v^{n-1}\|$ |
|---|---|---|---|
| 0 | 0 | 0 | |
| 1 | 10.00000 | −1 | 10.0 |
| 2 | 9.27500 | −1.95 | 0.95 |
| 3 | 8.47937 | −2.8525 | 0.9025 |
| 4 | 7.67276 | −3.70988 | 0.857375 |
| 5 | 6.88237 | −4.52438 | 0.814506 |
| 6 | 6.12004 | −5.29816 | 0.773781 |
| 7 | 5.39039 | −6.03325 | 0.735092 |
| 8 | 4.69464 | −6.73159 | 0.698337 |
| 9 | 4.03244 | −7.39501 | 0.66342 |
| 10 | 3.40278 | −8.02526 | 0.630249 |
| 20 | −1.40171 | −12.8303 | 0.377354 |
| 30 | −4.27865 | −15.7072 | 0.225936 |
| 40 | −6.00119 | −17.4298 | 0.135276 |
| 50 | −7.03253 | −18.4611 | 0.080995 |
| 60 | −7.65003 | −19.0786 | 0.048495 |
| 70 | −8.01975 | −19.4483 | 0.029035 |
| 80 | −8.24112 | −19.6697 | 0.017385 |
| 90 | −8.37366 | −19.8022 | 0.010409 |
| 100 | −8.45302 | −19.8816 | 0.006232 |
| 120 | −8.52898 | −19.9576 | 0.002234 |
| 130 | −8.54601 | −19.9746 | 0.001338 |
| 140 | −8.55621 | −19.9848 | 0.000801 |
| 150 | −8.56232 | −19.9909 | 0.000480 |
| 160 | −8.56597 | −19.9945 | 0.000287 |
| 161 | −8.56625 | −19.9948 | 0.000273 |
| 162 | −8.56651 | −19.9951 | 0.000259 |
| 163 | −8.56675 | −19.9953 | 0.000246 |

# Policy Iteration

Policy iteration is another method for solving infinite-horizon Markov decision problems

1. Set $n = 0$, and select an arbitrary decision rule $d_0 \in D$
2. (Policy evaluation) Obtain $V_n$ by solving
$$(I - \lambda P_{d_n})V_n = r_{d_n}$$
3. (Policy improvement) Choose $d_{n+1}$ to satisfy
$$d_{n+1} = \operatorname*{argmax}_{d \in D}\{r_d + \lambda P_d V_n\}$$
setting $d_{n+1} = d_n$ if possible.
4. If $d_{n+1} = d_n$, stop and set $d^* = d_n$. Otherwise increment $n$ by 1 and return to step 2.

# Policy Iteration

- This algorithm yields a sequence of deterministic Markovian decision rules $\{d_n\}$ and value functions $\{V_n\}$.

- Implementation of the maximization in step 3 is *componentwise*. This means that, for each $s \in S$, we choose $d_{n+1}(s)$ so that

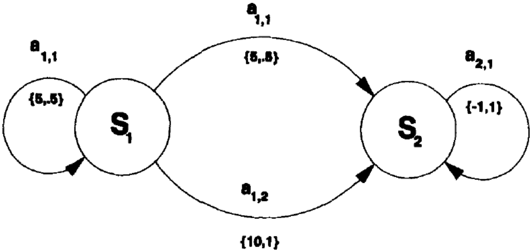
$$d_{n+1}(s) = \underset{a \in A_s}{\operatorname{argmax}} \{r(s,a) + \lambda \sum_{j \in S} p(j \mid s,a) V_n(j)\}.$$

- The following result establishes the monotonicity of the sequence $\{V_n\}$.

**Proposition**

Let $V_n$ and $V_{n+1}$ be successive values generated by the policy iteration algorithm. Then $V_{n+1} \geq V_n$.

# EXAMPLE




- Use policy iteration to solve this model. We set $\lambda = 0.95$, and choose $d_0(s_1) = a_{1,2}$ and $d_0(s_2) = a_{2,1}$.