Reinforcement Learning

Lecture 2
Markov Decision Processes

Stergios Christodoulidis

MICS Laboratory
CentraleSupélec
Université Paris-Saclay

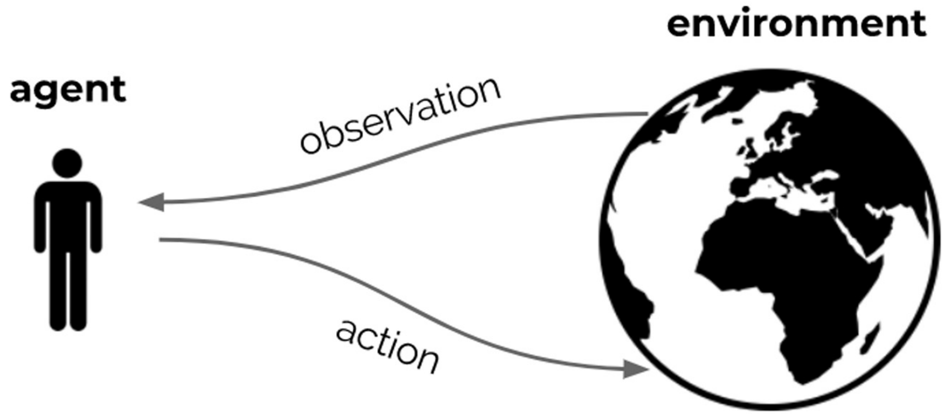
https://stergioc.github.io/

Last Lecture

What is Reinforcement Learning?

- Science and framework of learning to make decisions from interaction
- This requires us to think about
  - ...time
  - ...(long-term) consequences of actions
  - ...actively gathering experience
  - ...predicting the future
  - ...dealing with uncertainty
- Huge potential scope
- A formalization of the AI problem

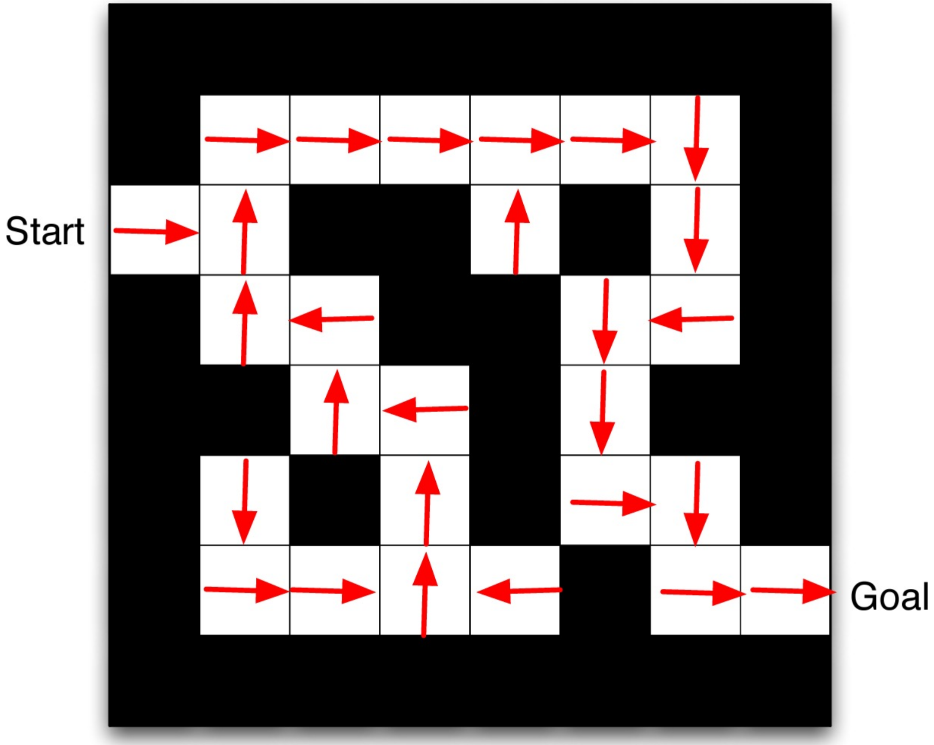
The Agent and the Environment

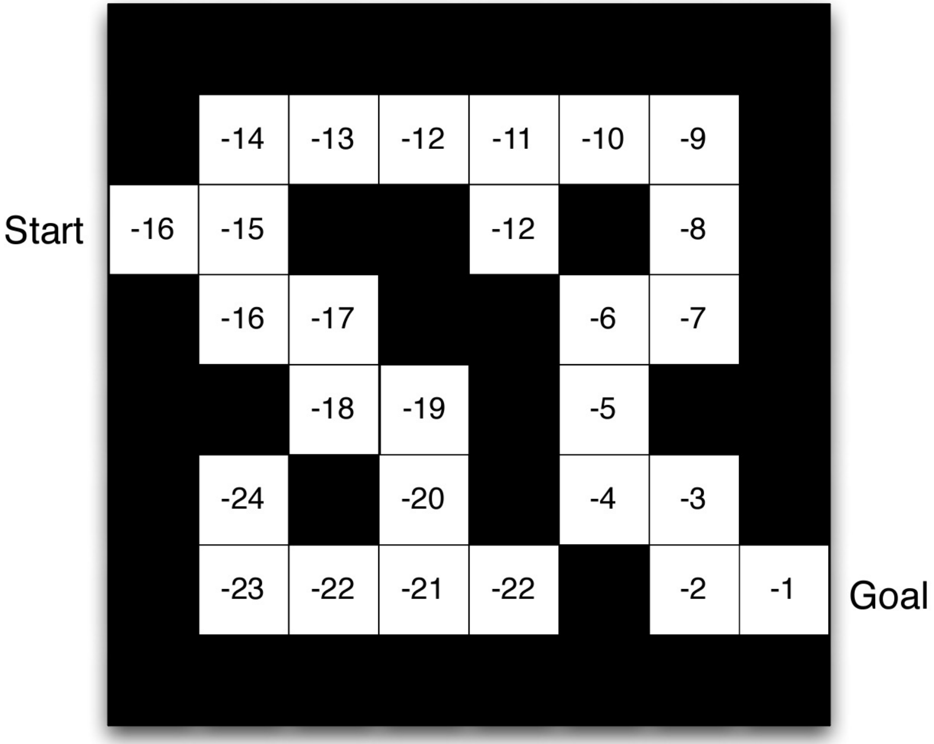


- At each step t the agent:
  - Receives observation O_t (and reward R_t)
  - Executes action A_t
- The environment:
  - Receives action A_t
  - Emits observation O_{t+1} (and reward R_{t+1})

[Hado van Hasselt, 2021]

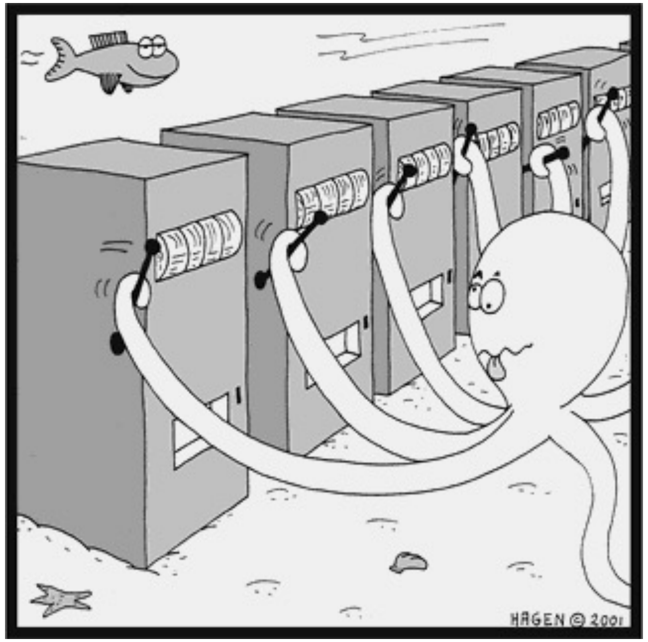
Maze Example

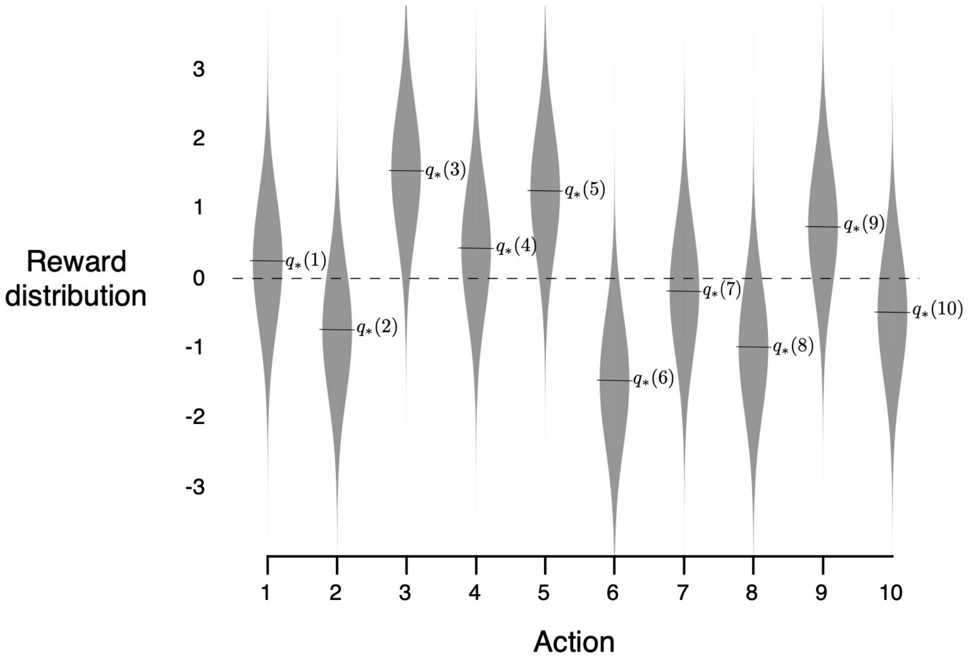




[Hado van Hasselt, 2021]

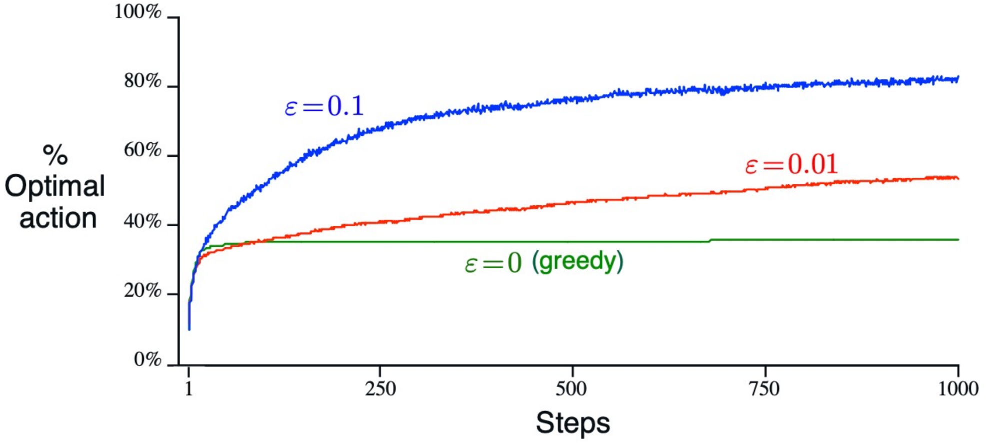
Multi-Armed Bandits

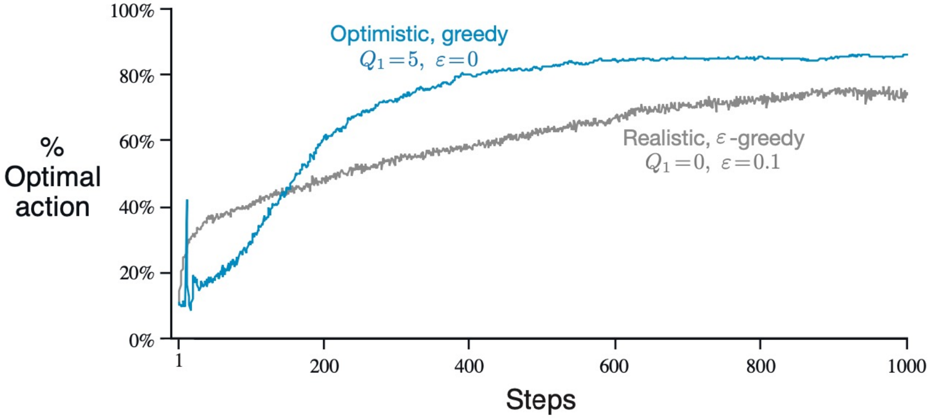


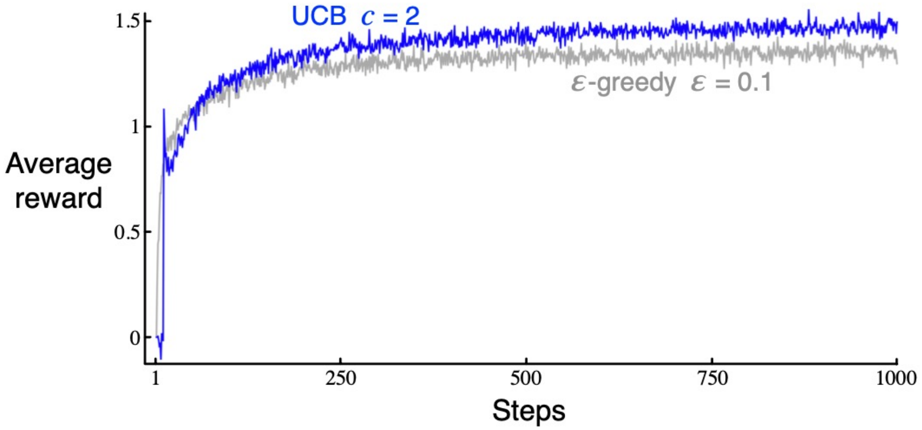


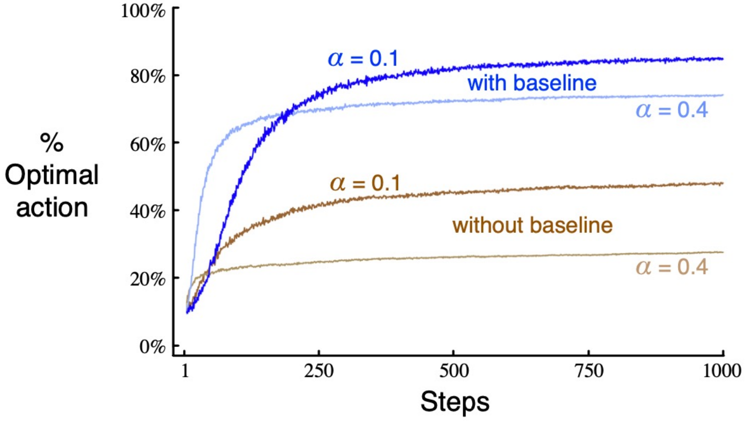
[David Silver, IRL, UCL 2015]

Algorithms









[An Introduction to Reinforcement Learning, Sutton and Barto]

Summary

- Reinforcement learning is the science of learning to make decisions
- Agents can learn a policy, value function and/or a model
- The general problem involves taking into account time and consequences
- Decisions affect the reward, the agent state, and environment state
- Learning is active: decisions impact data
- Have covered several principles for exploration/exploitation
- Each principle was developed in bandit setting
- Same principles can be extended to the MDP setting

Today’s Lecture

Today’s Lecture

- Markov Processes
- Markov Reward Processes
- Markov Decision Processes

Introduction to MDPs

- Markov decision processes formally describe an environment for reinforcement learning
- When the environment is fully observable i.e., The current state completely characterizes the process
- Almost all RL problems can be formalized as MDPs, e.g.
    - Optimal control primarily deals with continuous MDPs
    - Partially observable problems can be converted into MDPs
    - Bandits are MDPs with one state

The Markov Property

Definition

A state S_t is Markov if and only if:

$$\mathbb{P}[S_{t+1}|S_t] = \mathbb{P}[S_{t+1}|S_1 \dots S_t]$$

- "The future is independent of the past given the present"
- The state captures all relevant information from the history
- Once the state is known, the history may be thrown away i.e.
- The state is a sufficient statistic of the future

State Transition Matrix

- For a Markov state s and successor state s′, the state transition probability is defined by

$$\mathcal{P}_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s]$$

- A State transition matrix $\mathcal{P}$ can also be defined that holds the transition probabilities from all states s to all successor states s′,

$$\mathcal{P} = \begin{bmatrix} \mathcal{P}_{11} & \cdots & \mathcal{P}_{1n} \\ \vdots & \ddots & \vdots \\ \mathcal{P}_{n1} & \cdots & \mathcal{P}_{nn} \end{bmatrix}$$

- each row of the matrix sums to 1.

Markov Process

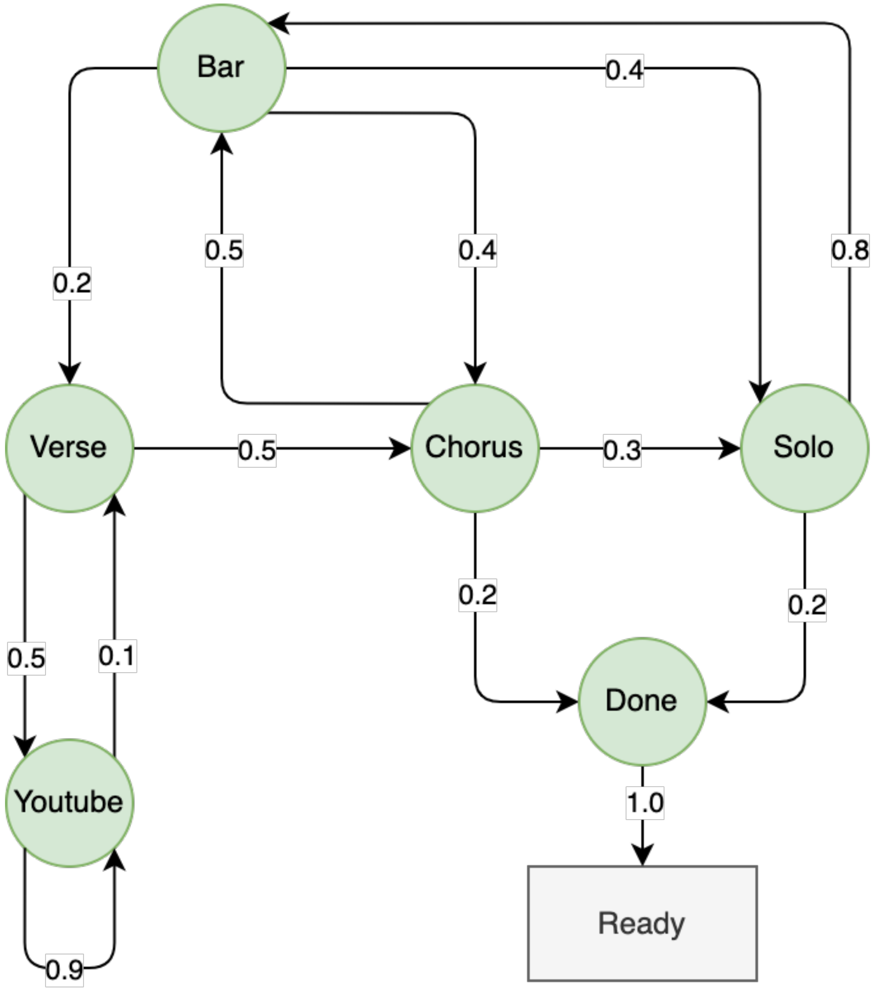
- A Markov process is a memoryless random process
- A sequence of random states $[S_1, S_2, \dots]$ with the Markov property.

Definition

A Markov Process (or Markov Chain) is a tuple $< \mathcal{S}, \mathcal{P} >$

- S is a (finite) set of states
- P is a state transition matrix, i.e., $\mathcal{P}_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s]$

Example: Song Learning Markov Process/Chain

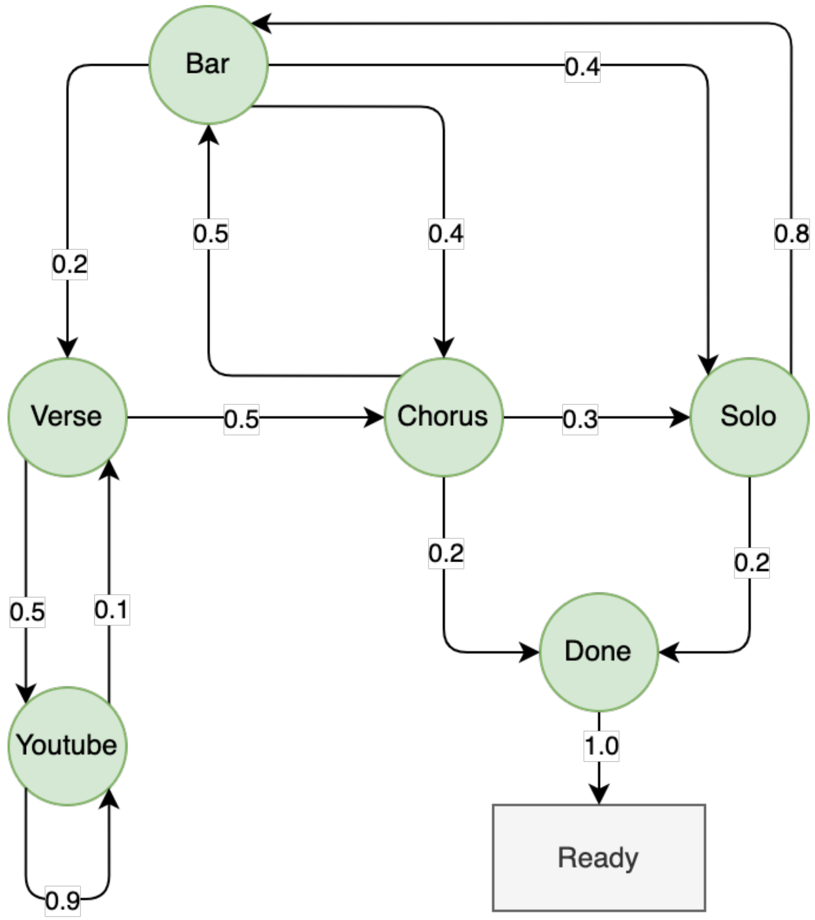


Example: Episodes

- Sample episodes for Markov Chain starting from $S_1 = Verse$

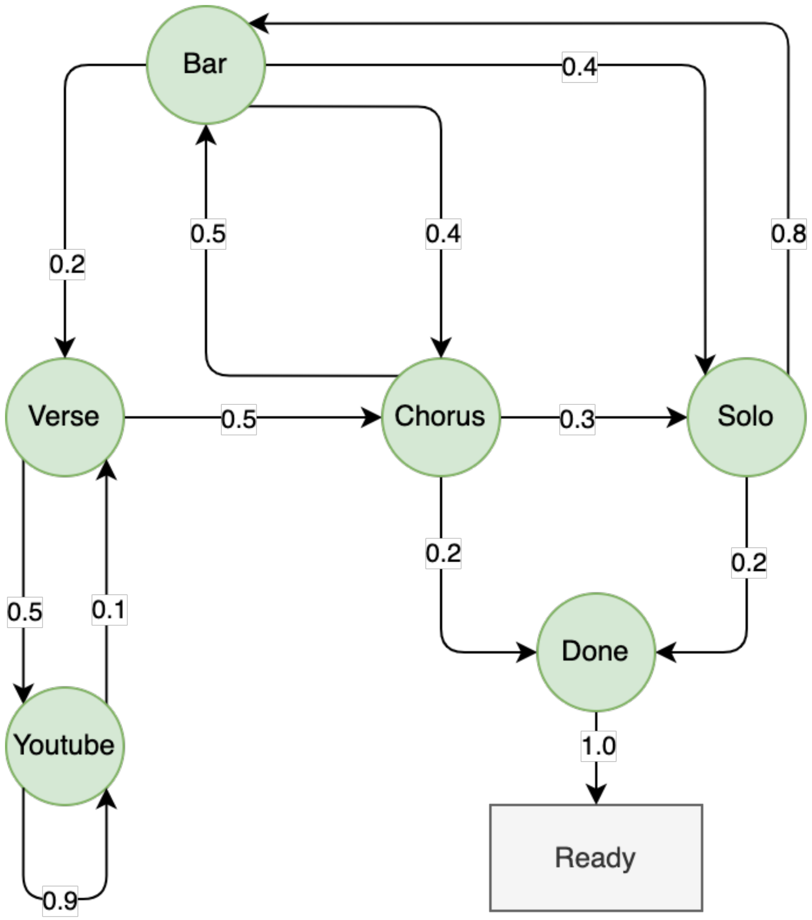
$$S_1, S_2, \ldots, S_T$$

- Verse, Chorus, Solo, Done.
- Verse, Youtube, Youtube, Chorus, Done.
- Verse, Youtube, Verse, Youtube, Chorus, Bar, Chorus, Bar, Chorus, Solo, Done.



Example: Transition Matrix

$$\mathcal{P} = \begin{array}{c} \begin{array}{cccccc} \textbf{V} & \textbf{C} & \textbf{S} & \textbf{Yt} & \textbf{B} & \textbf{D} \end{array} \\ \begin{bmatrix} 0 & 0.5 & 0 & 0.5 & 0 & 0 \\ 0 & 0 & 0.3 & 0 & 0.5 & 0.2 \\ 0 & 0 & 0 & 0 & 0.8 & 0.2 \\ 0.1 & 0 & 0 & 0.9 & 0 & 0 \\ 0.3 & 0.4 & 0.3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{array} \begin{array}{c} \\ \textbf{V} \\ \textbf{C} \\ \textbf{S} \\ \textbf{Yt} \\ \textbf{B} \\ \textbf{D} \end{array}$$



Markov Reward Process

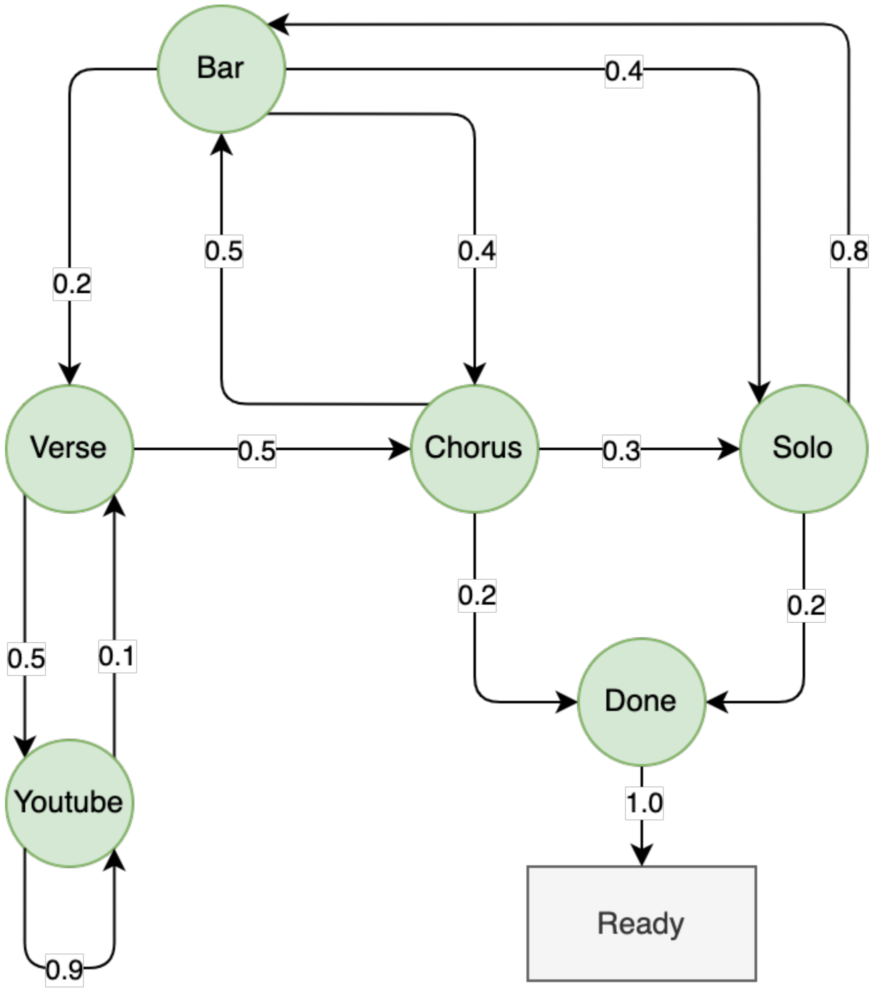
- A Markov reward process is a Markov chain with values.

Definition

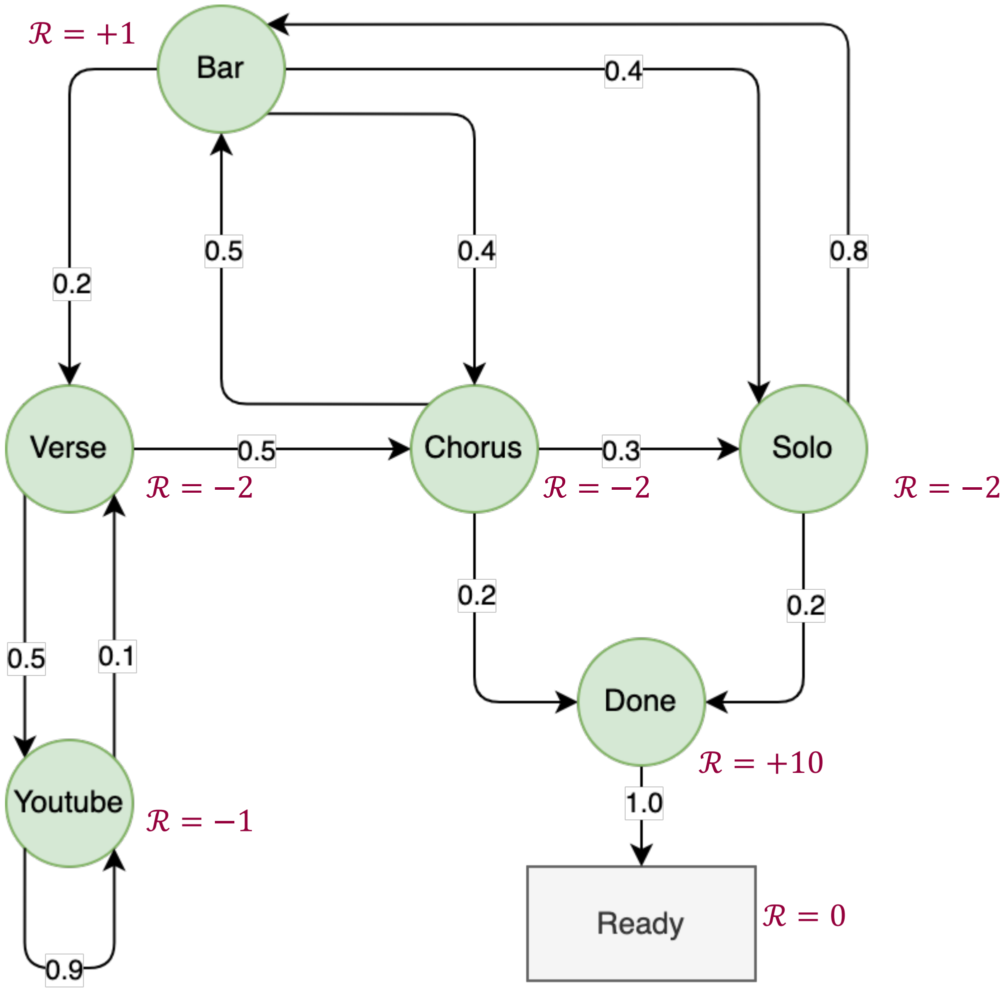
A Markov Process (or Markov Chain) is a 4-tuple $< \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma >$

- S is a (finite) set of states
- P is a state transition matrix, i.e., $\mathcal{P}_{ss'} = \mathbb{P}[S_{t+1} = s' \mid S_t = s]$
- $\mathcal{R}$ is a reward function, $\mathcal{R}_s = \mathbb{E}[R_{t+1} \mid S_t = s]$
- γ is a discount factor, $\gamma \in [0, 1]$

Example: Song Learning Markov Reward Process (MRP)



Example: Song Learning Markov Reward Process (MRP)



Return

Definition

The return G_t is the total discounted reward from time-step t:

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

- The discount $\gamma \in [0, 1]$ is the present value of future rewards
- The value of receiving reward R after $k + 1$ time-steps is $\gamma^k R$.
- This values immediate reward above delayed reward.
  - γ close to 0 leads to ”myopic” evaluation
  - γ close to 1 leads to ”far-sighted” evaluation

Why discount?

Most Markov reward and decision processes are discounted. Why?

- Mathematically convenient to discount rewards
- Avoids infinite returns in cyclic Markov processes
- Uncertainty about the future may not be fully represented
- If the reward is financial, immediate rewards may earn more interest than delayed rewards
- Animal/human behavior shows preference for immediate reward
- It is sometimes possible to use undiscounted Markov reward processes (i.e., $\gamma = 1$), e.g., if all sequences terminate.

State-Value Function

- The value function v(s) gives the long-term value of state s

Definition

The state value function v(s) of an MRP is the expected return starting from state s:

$$\mathrm{v}(s) = \mathbb{E}[G_t \mid S_t = s]$$

Example: MRP Returns

- Sampling Returns from the MRP
- Start from a state ($S_1 = Verse$) and use discount $\gamma = \frac{1}{2}$

$$G_1 = R_2 + \gamma R_3 + \ldots + \gamma^{\mathrm{T}-2} \mathrm{R}_T$$

- E.g.
  - Verse, Chorus, Solo, Done.

$$\mathrm{v}_1 = -2 + \tfrac{1}{2}(-2) + \tfrac{1}{4}(-2) + \tfrac{1}{8}(10) = -2.25$$

  - Verse, Youtube, Youtube, Chorus, Done.

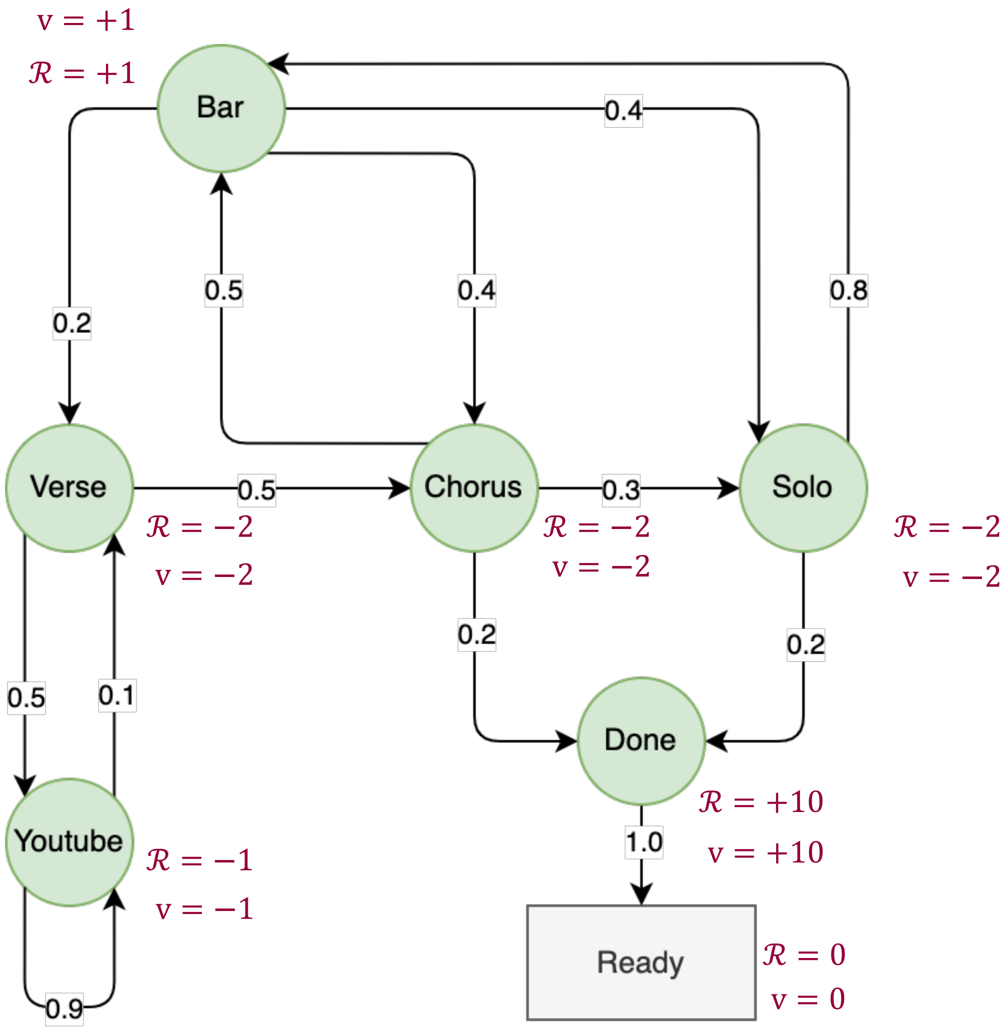
$$\mathrm{v}_1 = -2 + \tfrac{1}{2}(-1) + \tfrac{1}{4}(-1) + \tfrac{1}{8}(-2) + \tfrac{1}{16}(10) = -2.35$$

  - Verse, Youtube, Verse, Youtube, Chorus, Bar, Chorus, Bar, Chorus, Solo, Done.

$$\mathrm{v}_1 = -2 + \tfrac{1}{2}(-1) + \tfrac{1}{4}(-2) + \tfrac{1}{8}(-1) + \tfrac{1}{16}(-2) + \cdots = -3.24$$

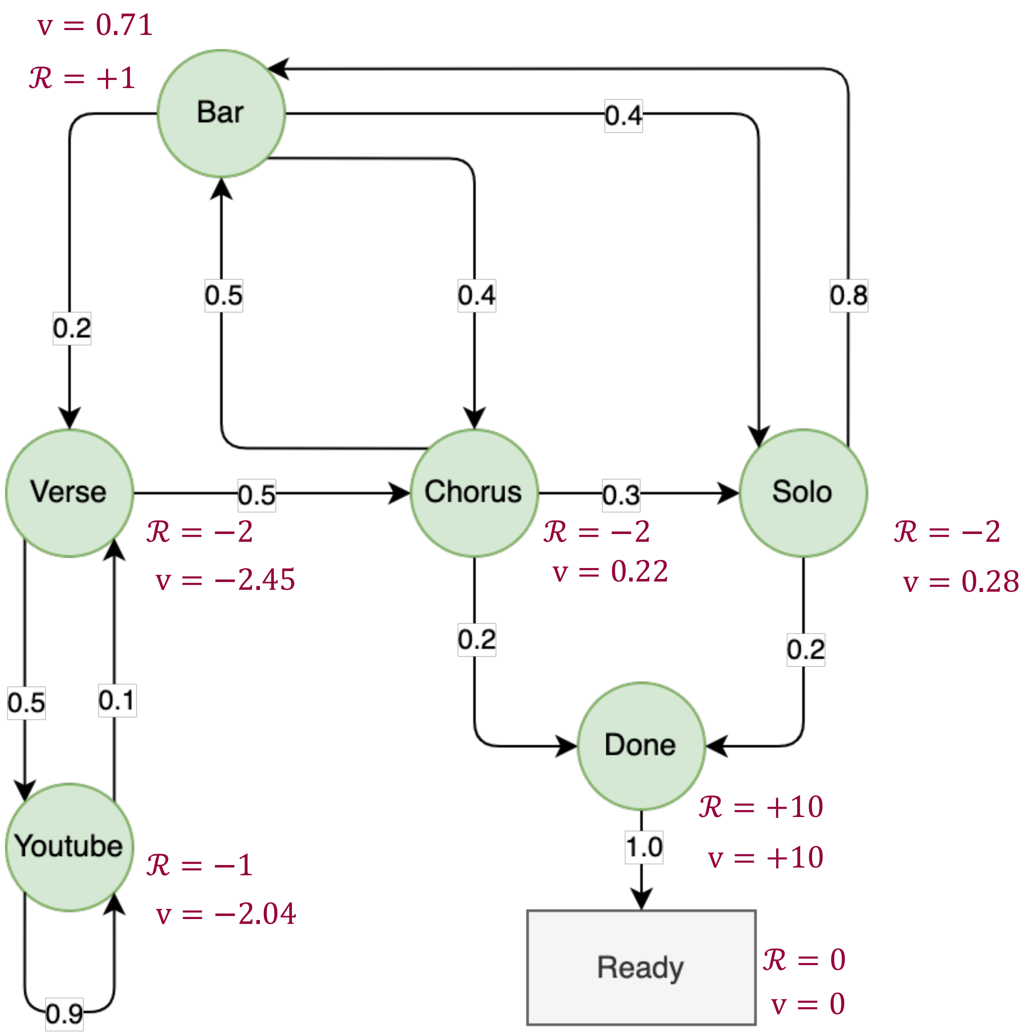
Example: State-Value Function for MRP (1/2)

$\mathrm{v}(s)$ for $\gamma = 0$



Example: State-Value Function for MRP (2/2)

$v(s)$ for $\gamma = 0.5$



Bellman Equation for MRPs (1/2)

- The value function can be decomposed into two parts:
  - Immediate reward R_{t+1}
  - Discounted value of successor state $\gamma \mathrm{v}(S_{t+1})$

$$
\begin{aligned}
\mathrm{v}\,(s) &= \mathbb{E}[G_t \mid S_t = s] \\
&= \mathbb{E}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \mid S_t = s] \\
&= \mathbb{E}[R_{t+1} + \gamma (R_{t+2} + \gamma R_{t+3} + \cdots) \mid S_t = s] \\
&= \mathbb{E}[R_{t+1} + \gamma \mathrm{G}_{t+1} \mid S_t = s] \\
&= \mathbb{E}[R_{t+1} + \gamma \mathrm{v}(S_{t+1}) \mid S_t = s]
\end{aligned}
$$

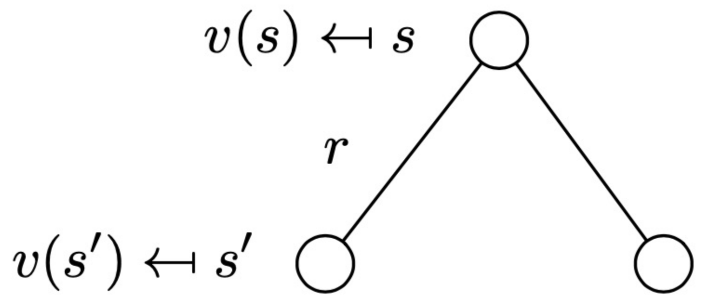
Bellman Equation for MRPs (2/2)

- Essentially, the value of a state is the sum of the immediate reward and the discounted value of the successor state.

$$v\,(s) = \mathbb{E}[R_{t+1} + \gamma v(S_{t+1})|\ S_t = s]$$

- Similarly, we can use the transition matrix and write:

$$v\,(s) = \mathcal{R}_s + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}\, v\,(s')$$

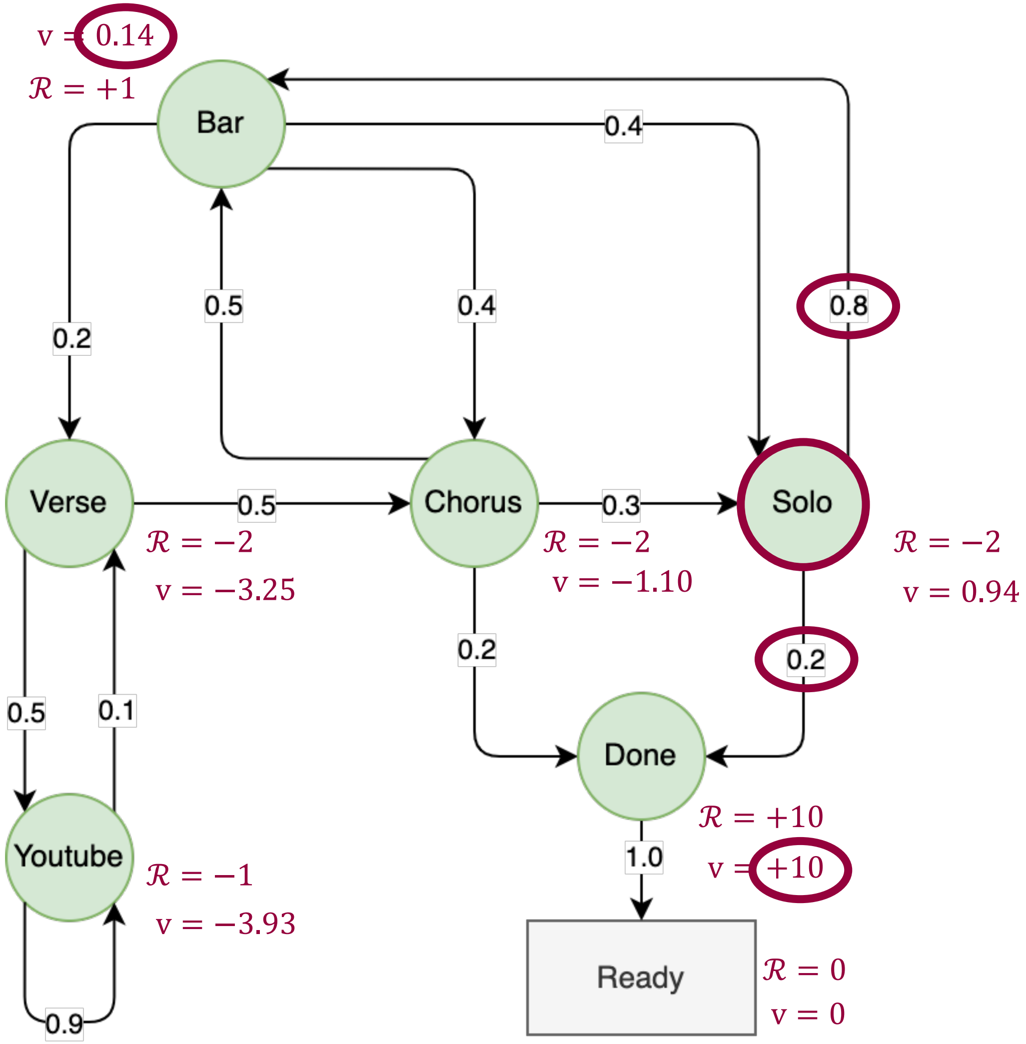


[David Silver, IRL, UCL 2015]

Example: Bellman Expectation Equation in MRP

$\mathrm{v}(s)$ for $\gamma = 0.5$

$0.94 = -2 + 0.5 * (0.8 * 0.14 + 0.2 * 10)$



Bellman Equation in Matrix Form

- The Bellman equation can be expressed using matrices,

$$\mathrm{v} = \mathcal{R} + \gamma \mathcal{P} \mathrm{v}$$

- Where v is a column vector with one entry per state

$$\begin{bmatrix} \mathrm{v}(s_1) \\ \vdots \\ \mathrm{v}(s_n) \end{bmatrix} = \begin{bmatrix} R_1 \\ \vdots \\ R_n \end{bmatrix} + \gamma \begin{bmatrix} \mathcal{P}_{11} & \cdots & \mathcal{P}_{1n} \\ \vdots & \ddots & \vdots \\ \mathcal{P}_{n1} & \cdots & \mathcal{P}_{nn} \end{bmatrix} \begin{bmatrix} \mathrm{v}(s_1) \\ \vdots \\ \mathrm{v}(s_n) \end{bmatrix}$$

Solving the Bellman Equation

- The Bellman equation is a linear equation
- It can be solved directly:

$$\mathrm{v} = (\mathrm{I} - \gamma\mathcal{P})^{-1}\mathcal{R}$$

- Matrix Inversion is computational heavy $\mathcal{O}(n^3)$
- Direct solution only possible for small MRPs
- There are many iterative methods for large MRPs, e.g.
  - Dynamic programming
  - Monte-Carlo evaluation
  - Temporal-Difference learning

Markov Decision Process

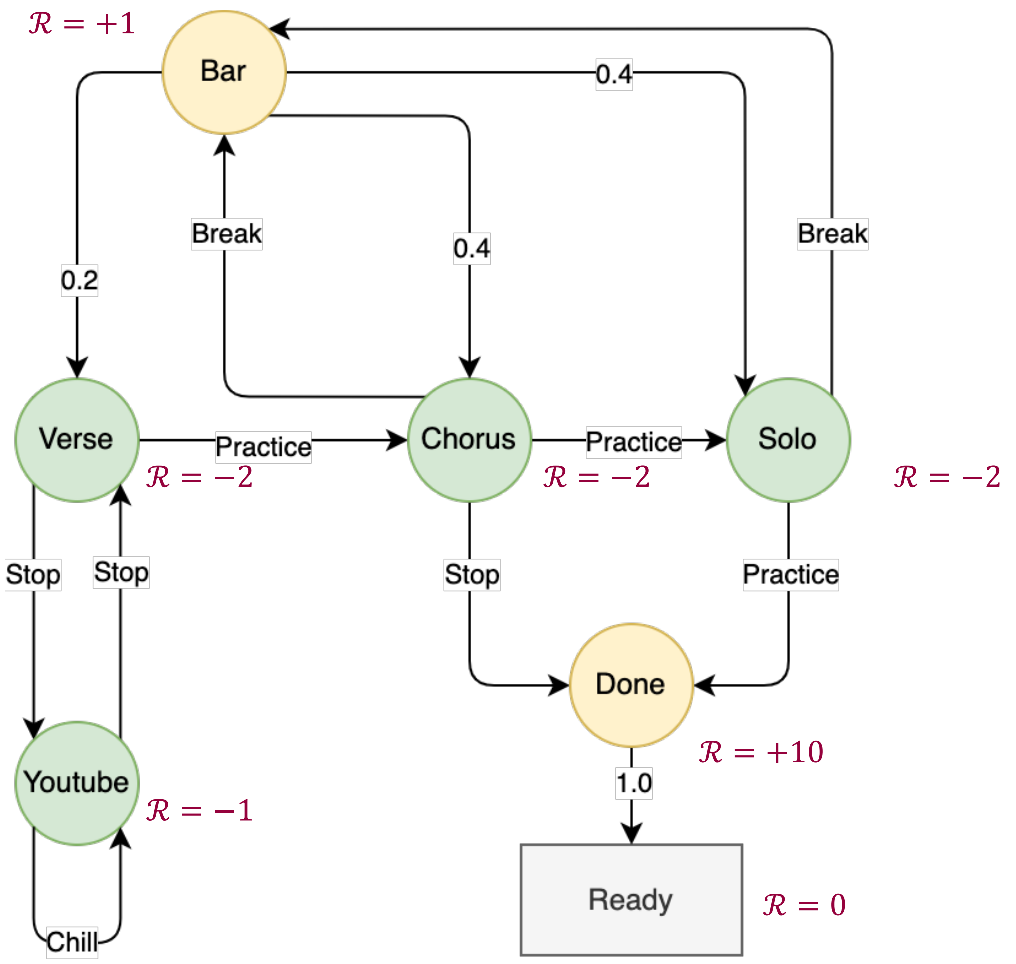
- A Markov decision process (MDP) is a Markov reward process with decisions. It is an environment in which all states are Markov

Definition

A Markov Process (or Markov Chain) is a 4-tuple $< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$

- $\mathcal{S}$ is a (finite) set of states
- $\mathcal{A}$ is a (finite) set of actions
- $\mathcal{P}$ is a state transition matrix, i.e., $\mathcal{P}_{ss'}^{a} = \mathbb{P}[S_{t+1} = s' \mid S_t = s,\ A_t = a]$
- $\mathcal{R}$ is a reward function, $\mathcal{R}_s^a = \mathbb{E}[R_{t+1} \mid S_t = s,\ A_t = a]$
- γ is a discount factor, $\gamma \in [0, 1]$

Example: Song Learning Markov Decision Process (MDP)



Policies (1/2)

Definition

A policy π is a distribution over actions given states:

$$\pi(a|s) = \mathbb{P}[A_t = a \mid S_t = s]$$

- A policy fully defines the behaviour of an agent
- MDP policies depend on the current state (not the history)

Policies (2/2)

- Given and MDP $\mathcal{M} =< \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma >$ and a policy π
- The state sequence $S_1, S_2, \ldots$ is a Markov Chain/Process $< \mathcal{S}, \mathcal{P}^{\pi} >$
- The state and reward sequence $S_1, R_1, S_2, R_2 \ldots$ is a Markov Reward Process $< \mathcal{S}, \mathcal{P}^{\pi}, \mathcal{R}^{\pi}, \gamma >$

$$\mathcal{P}^{\pi}_{ss'} = \sum_{a \in \mathcal{A}} \pi(a|s) \mathcal{P}^{a}_{ss'}$$

$$\mathcal{R}^{\pi}_{s} = \sum_{a \in \mathcal{A}} \pi(a|s) \mathcal{R}^{a}_{s}$$

State-Value and Action-Value Functions

Definition

The state-value function $v_\pi(s)$ of an MDP is the expected return starting from state s, and then following policy π:

$$v_\pi(s) = \mathbb{E}_\pi[G_t \mid S_t = s]$$

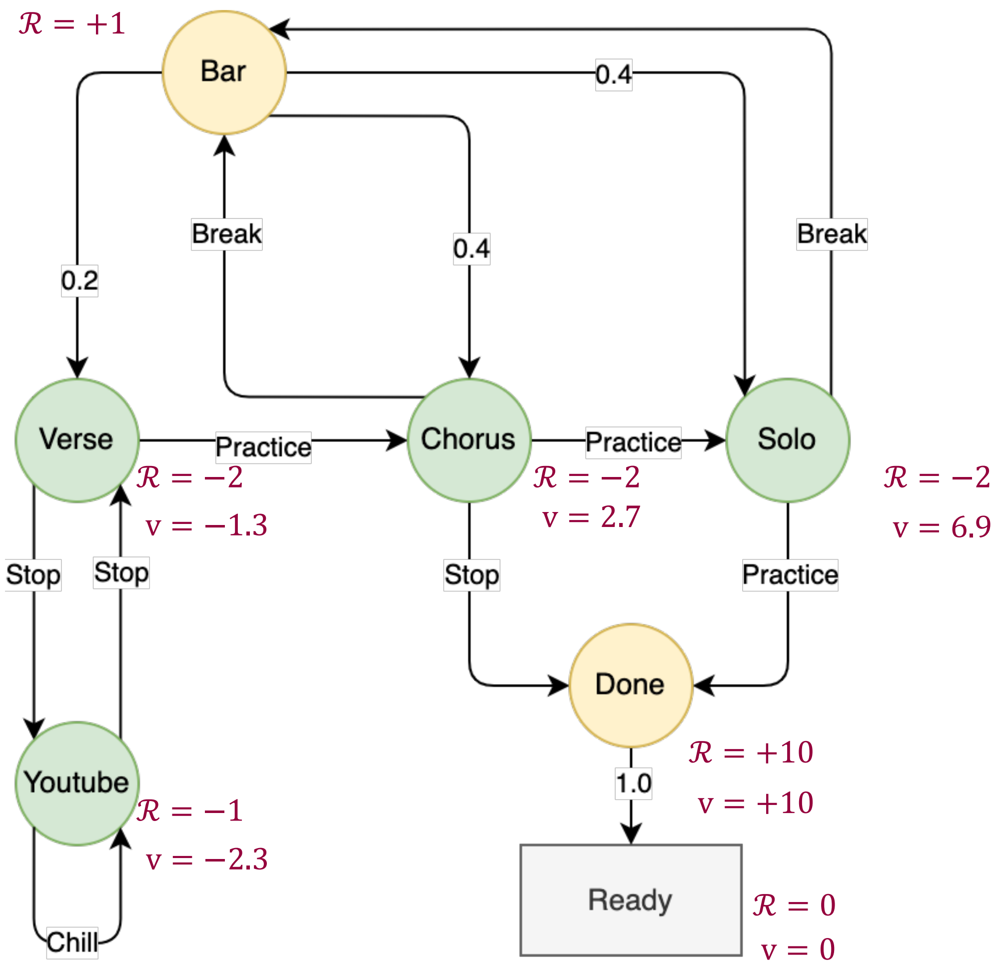
Definition

The action-value function $q_\pi(s,a)$ of an MDP is the expected return starting from state s, taking action a, and then following policy π:

$$q_\pi(s,a) = \mathbb{E}_\pi[G_t \mid S_t = s, \mathrm{A}_t = a]$$

Example: Bellman Expectation Equation in MDP

$v_\pi(s)$ for $\pi(a|s) = 0.5$



Bellman Expectation Equation

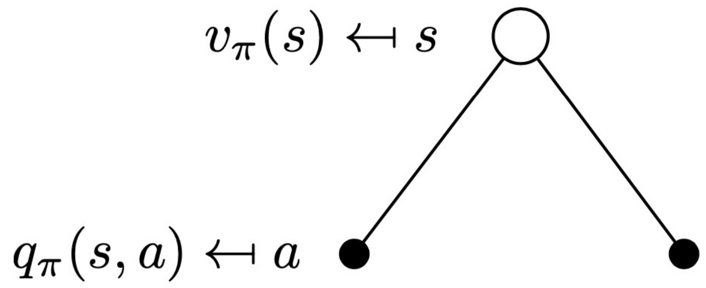
- The state-value function can again be decomposed into immediate reward plus discounted value of successor state,

$$\mathrm{v}_{\pi}(s) = \mathbb{E}_{\pi}[R_{t+1} + \gamma \mathrm{v}_{\pi}(S_{t+1})|\ S_t = s]$$

- Similarly, the action-value can be decomposed as such,

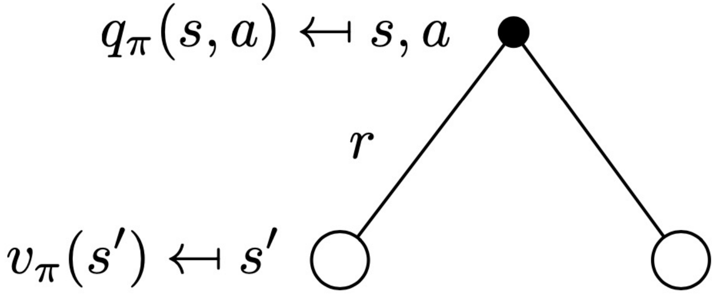
$$q_{\pi}(s, a) = \mathbb{E}_{\pi}[R_{t+1} + \gamma q_{\pi}(S_{t+1}, A_{t+1}) \mid S_t = s, A_t = a]$$

Bellman Expectation Equation for v_π(1/2)



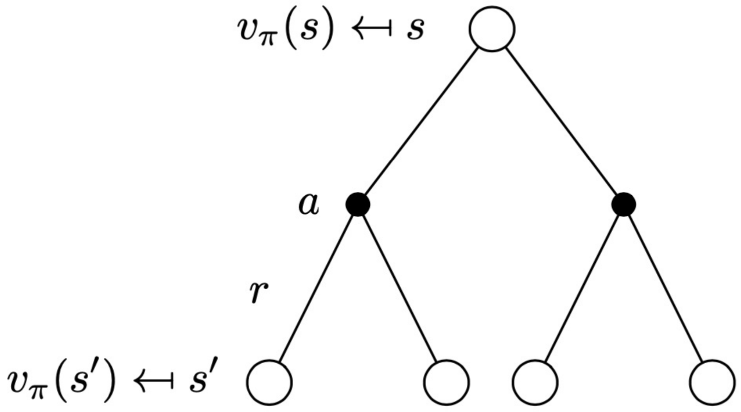
$$v_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a \mid s) q_\pi(s,a)$$

Bellman Expectation Equation for q_π



$$q_\pi(s,a) = \mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \mathrm{v}_\pi(s')$$

Bellman Expectation Equation for $\mathrm{v}_\pi(2/2)$

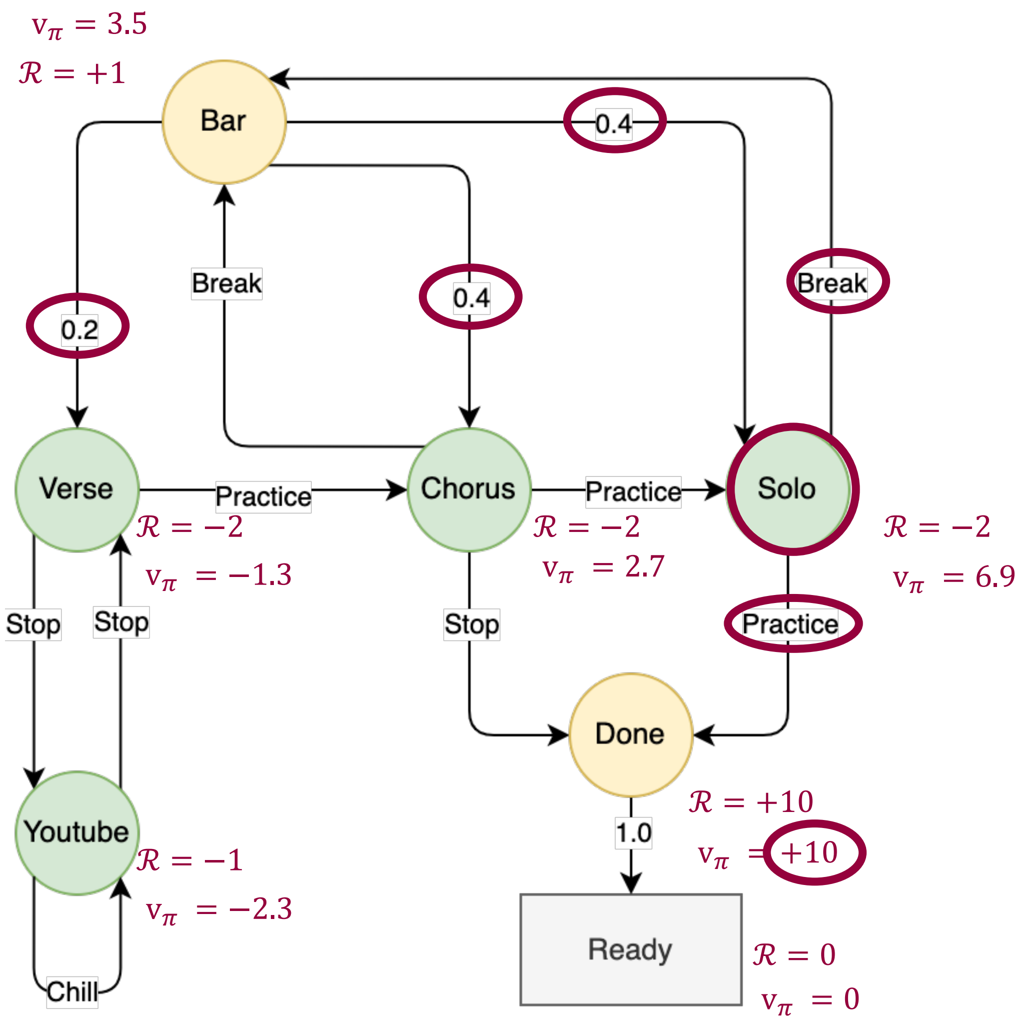


$$\mathrm{v}_\pi(s) = \sum_{a \in \mathcal{A}} \pi(a \mid s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a \mathrm{v}_\pi(s') \right)$$

Example: Bellman Expectation Equation in MDP

$v_\pi(s)$ for $\pi(a|s) = 0.5$

$$7.4 = 0.5 * (1 - 0.2 * 1.3 + 0.4 * 2.7 + 0.4 * 7.4) + 0.5 * 10$$



Bellman Expectation Equation (Matrix Form)

- The Bellman equation is a linear equation

$$\mathrm{v}_{\pi} = \mathcal{R}^{\pi} + \gamma \mathcal{P}^{\pi} \mathrm{v}$$

- It can be solved directly:

$$\mathrm{v}_{\pi} = (\mathrm{I} - \gamma \mathcal{P}^{\pi})^{-1} \mathcal{R}^{\pi}$$

Optimal Value Function

Definition

The optimal state-value function $v_*(s)$ is the maximum value function over all policies

$$v_*(s) = \max_{\pi} v_\pi(s)$$

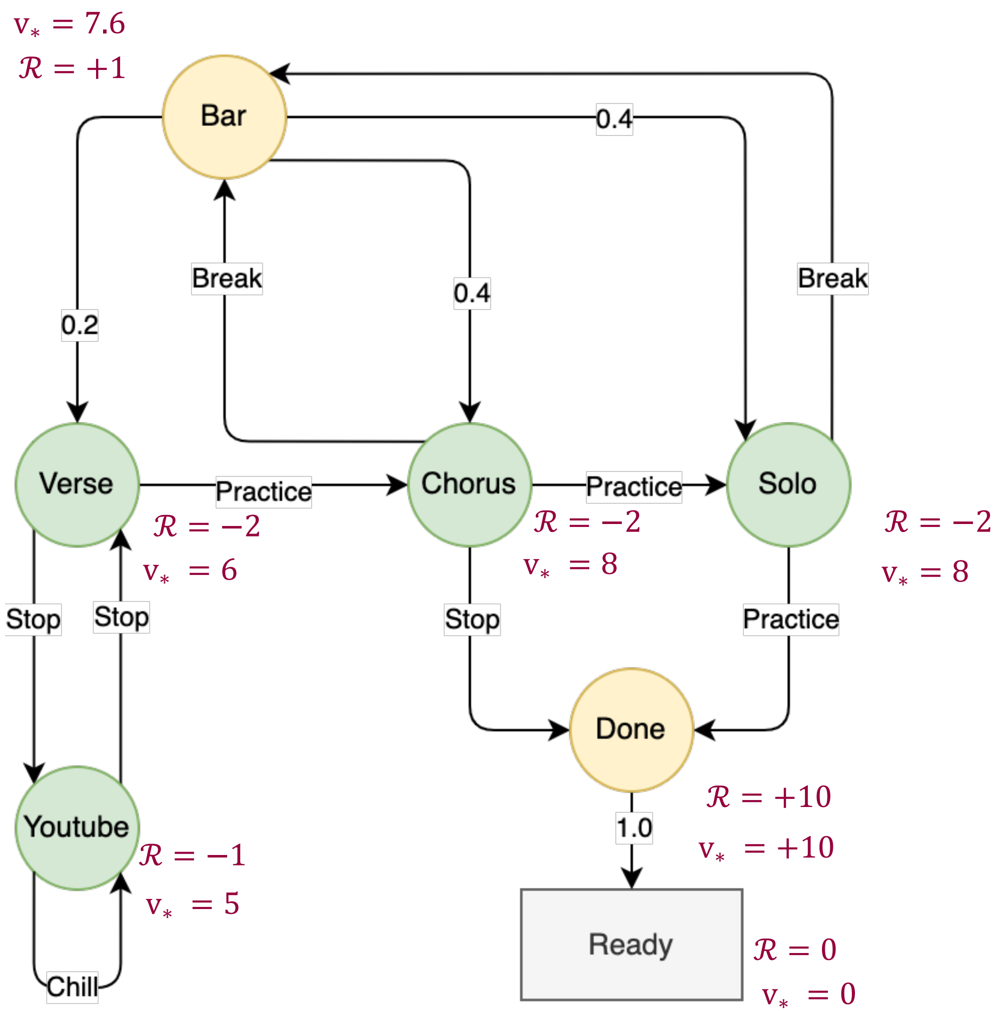
The optimal action-value function $q_*(s,a)$ is the maximum action-value function over all policies

$$q_*(s,a) = \max_{\pi} q_\pi(s,a)$$

- The optimal value function specifies the best possible performance in the MDP.
- An MDP is "solved" when we know the optimal value fn.

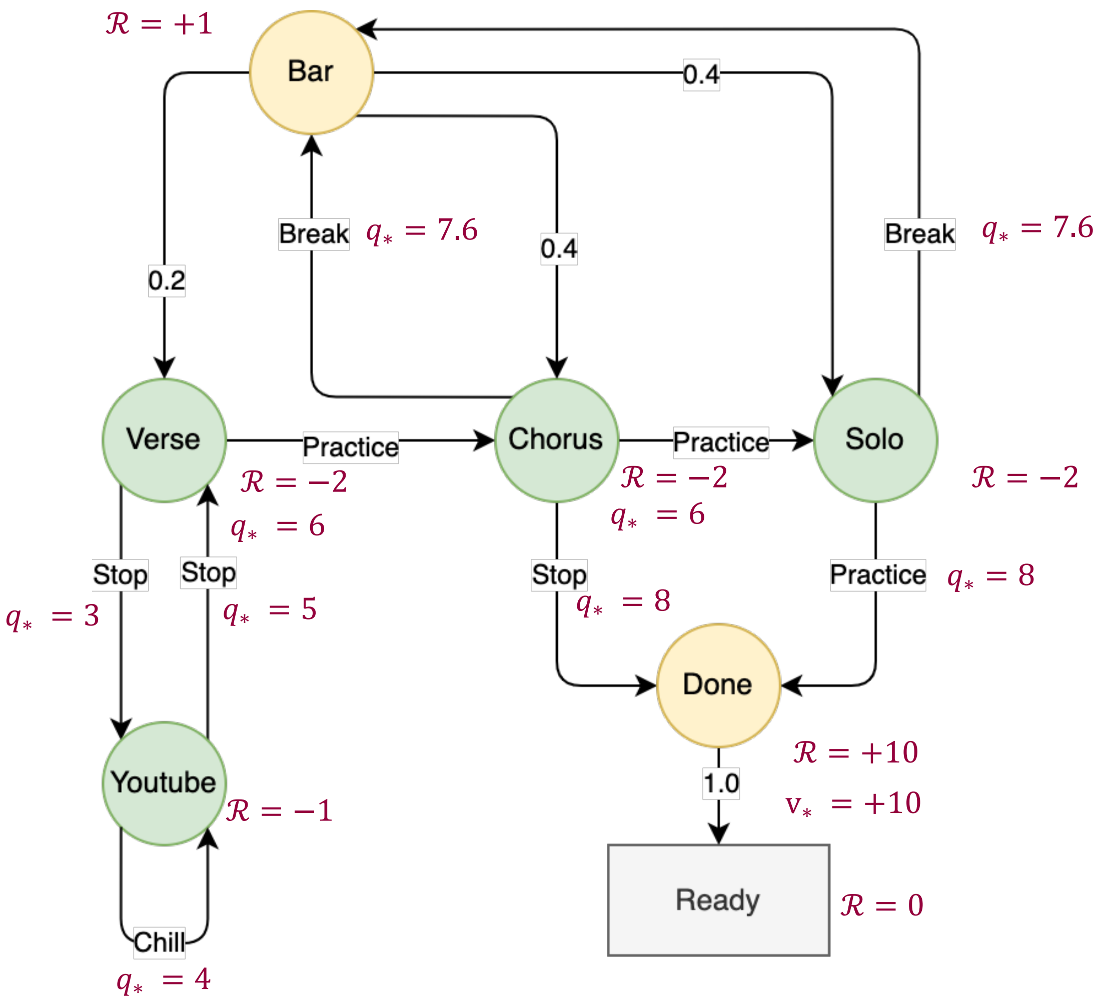
Example: Optimal Value Function for MDP

$v_*(s)$ for $\gamma = 1$



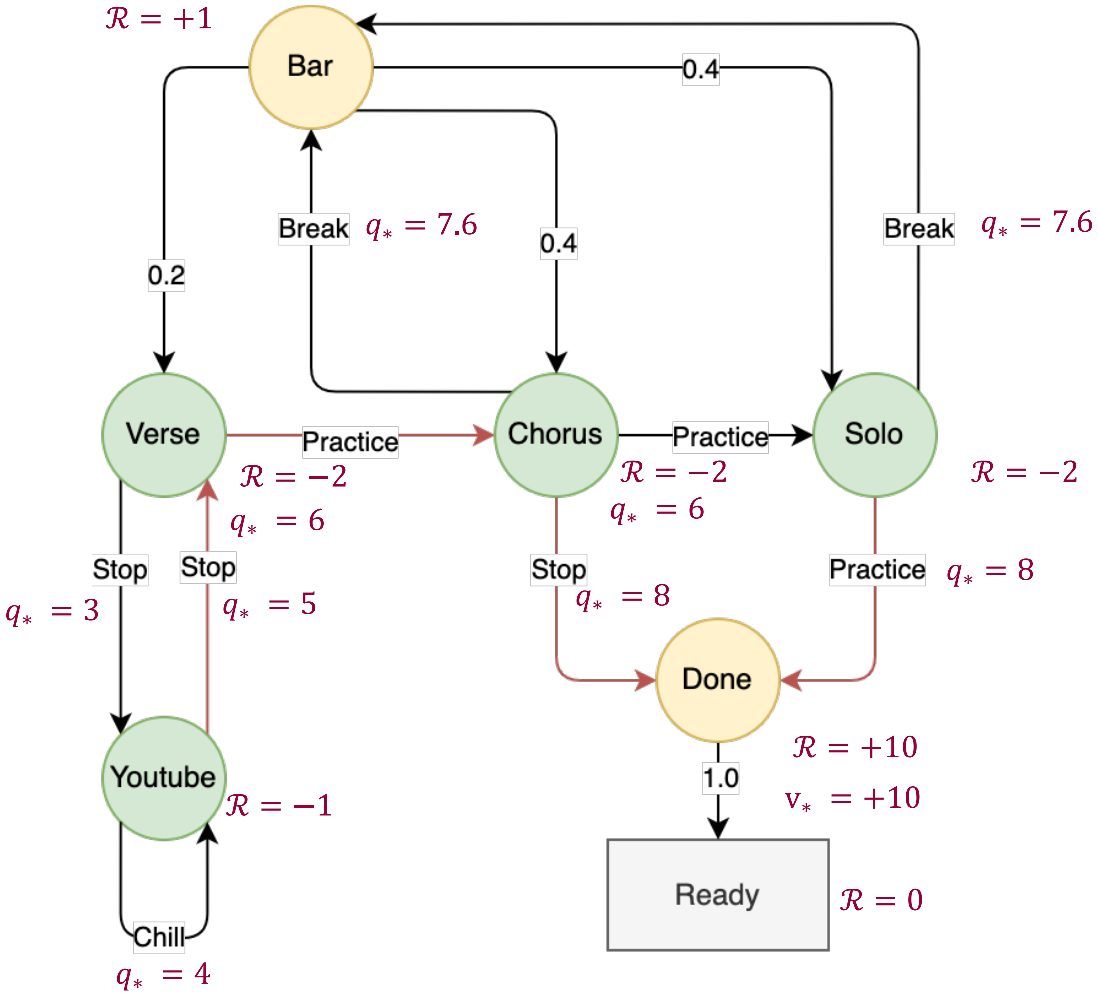
Example: Optimal Action-Value Function for MDP

$q_*(s,a)$ for γ= 1



Example: Optimal Policy for MDP

$\pi_*(a|s)$ for γ= 1



Solving the Bellman Optimality Equation

- Bellman Optimality Equation is non-linear
- No closed form solution (in general)
- Many iterative solution methods
  - Value Iteration
  - Policy Iteration
  - Q-learning

Summary

- Markov (Reward, Decision) Process
- State-Value and Action-Value functions
- Bellman Equations