Introduction to reinforcement learning : From basic concepts to deep Q-networks:

Urtzi Ayesta and Matthieu Jonckheere

CNRS-IRIT & CNRS-LAAS

Introduction to Reinforcement Learning

- Responsables: Urtzi Ayesta et Matthieu Jonckheere
- Evaluation : TPs et projet final
- Pré-requis : Aucun
- 7 Cours et 7 TP
- 30 eleves max (2 groupes TP)

References

- Puterman., M. L. (1994). Markov Decision Processes. Wiley.
- Richard S. Sutton and Andrew G. Reinforcement Learning: An Introduction, MIT Press, Cambridge, MA, 2018.
- Dimitri P. Bertsekas, Reinforcement learning and optimal control, Athena,
- Courses, David Silver's Reinforcement Learning Course (UCL, 2015), Van Roy (UC Berkeley), Shimkin (Technion), Proutiere (KTH)
- **Probability and Markov chains:** P. Brémaud, Markov chains, Gibbs fields, Monte-Carlo and queues, 1999.

Introduction: different types of learning

- Supervised
- Unsupervised
- Reinforcement learning

Supervised learning

Typical task: We have a set of historical data with labels. We want to be able to guess the labels of new data.

Examples: Neural nets, Boltzmann machines,...

Principle: Basically, we want to learn how a system responds to inputs using its past responses and then generalize to new inputs.

Biggest challenge there:
Tradeoff between overfitting (learn a response working too specifically for the training data set) and efficiency (a response that works well on the training and testing sets).

Unupervised learning

Typical task: We have a data set without labels and we want to retrieve labels or more generally patterns.

Typical example: clustering.

Biggest challenge:
No ground truth (No way to completelty validate the methods)...
One has to find indirect ways to validate methods and results

Reinforcement learning

Here, we have an agent interacting with an unknown environment

Typical task: We want to optimize decisions/actions **by interacting**.

Hence, it is a different situation from both supervized and unsupervized learning.

Biggest challenge: Tradeoff between exploration and exploitation

Demo: You're the reinforcement learner

- Actions in every state, action 1 and action 2
- World formed by two states.

What is the best action in every state?

RL paradigm

Data are observations of a reward which depends on a control (actions).

Learning cycles are:

Initial state + Action $\Rightarrow$ New state + Reward $\Rightarrow$ Action $\Rightarrow$ New state + Reward ...

$\Rightarrow$ means transitions which can be random and depend on the actions taken

To get something mathematically tractable, one needs to suppose that transitions $\Rightarrow$ are going to be Markov, i.e., independent from the past given the present state.

Differences with other learning tasks:

1. You observe a reward signal **sequentially** that depends on your choice of action.
2. Hence, you get out of the i.i.d world and dive into Markov...
3. The global reward depends on the future!! (What you want to optimize depends in each state on non-observed data).

RL within many sciences

RL is a part of “machine learning” deeply connected with many other sciences:

- Neuro-sciences, psychology,...
- It is part of AI and ML (though it s difficult to define these areas those days)
- **Maths**: OR, Statistics, Probability, Optimization, Analysis, ... are all involved
- **Computer science**
- Robotics, Control,...
- ...

Influence from biology/neurosciences

Thorndike (1911) Law of effects: "Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that, when it recurs, they will be less likely to occur. The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond "

Starting points

- Neural nets and RL principles come from computational neuro-sciences, pyschology and for RL also from the study of animal behaviours: models of animal conditionning, behaviour reinforcement in function of satisfaction.
- First theoretical framework : **Bellman equation under Markov models.** (We'll get there quickly)

Artificial intelligence at last?

One of the most promising technique of AI for at least two reasons:

These are very important and central problems as they deal with **taking decisions in real time (taking into account the learning costs).**

It deals with non-stationarity.

Examples of application

- Control resources (e.g., wireless networks)
- Control supply chains,
- Manage an investment portfolio,
- Control a power station,
- Make a humanoid robot walk,
- Play many different games (Atari, chess, Go, Starcraft, ...) better than humans

Very different problems... (WHY?)

same mathematical treatments.

Control and learning

- Indirect. (not always possible nor efficient)

  First we learn a model using typically supervised learning, and then we optimize the control.

- Direct
  Learn a strategy and the model at the same time (or almost).

  Can be a good way if the control is simple enough, even for very complex dynamics.

The simplest model

- One player has two possibilities (two arms to be played).
- Arm k gives an immediate reward when played r_t with distribution ν_k.
- For instance, the player receives 2,0,7,9 from arm 1 and 10,2 from arm 2.
  What do we play next?

Global objective and immediate rewards

- At each time t, one observes (gets) a reward r_t (that depends on noise and on the specific action taken at time $t-1$)
- Typical global objective:

$$\sum_{t=0}^{T} \gamma^t r_t$$

T can be infinite.
Could also be a "regret"

$$\sum_t (r^* - r_t),$$

with r^* the best mean reward possible.

Central difficulty

What is the optimal tradeoff between exploration and exploitation (optimize the objective)?

- For the bandit problem, exploration amounts to estimate the distributions ν_k.
- Many many methods...

Plan of the course

1. Recall basic notions of probability
2. Notions on Markov Chains, convergence to equilibrium,...
3. Notions of MDP (Markov decision process), Theoretical optimal solutions when transitions are known.
   Dynamic programming: iterative solutions.
4. Bandits problems. Unknown model but very simple. First example of tradeoff between exploration and exploitation.
5. Model free for small state spaces. Monte-Carlo vs Temporal differences.
6. Bigger state space: modern computational techniques.

Reminder on Markov chains

Markov chains

Let $\mathcal{S}$ be a countable set, called the state space.

Let the chain $(X_n)_{n\geq 0}$, $X_n \in \mathcal{S}$, be a sequence of random variables in $\mathcal{S}$. and $\mathcal{F}_n = \sigma(X_1, X_2, \ldots, X_n)$ be the σ-*algebra* associated with the chain until instant n. It encodes the history of the process until n.

Definition (Markov chain)

$(X_n)_{n\geq 0}$ is a *Markov chain*(MC) if

$$E[f(X_{n+1})|\mathcal{F}_n] = E[f(X_{n+1})|X_n]$$

for any bounded and measurable function f.

Markov property

The definition implies that a sequence is Markovian if the statistics of the future evolution depends uniquely on the current state and not on the whole history.

In other words, in order to statistically predict the future, the knowledge of the current state is sufficient.

Transitions

Simpler definition (equivalent for countable state space):

Definition

$(X_n)_{n\geq 0}$ is a *Markov chain*(MC) if

$$P(X_{n+1} = x | X_0 = x_0, \dots, X_n = x_n) = P(X_{n+1} = x | X_n = x_n).$$

Even simpler definition for time homogeneous:

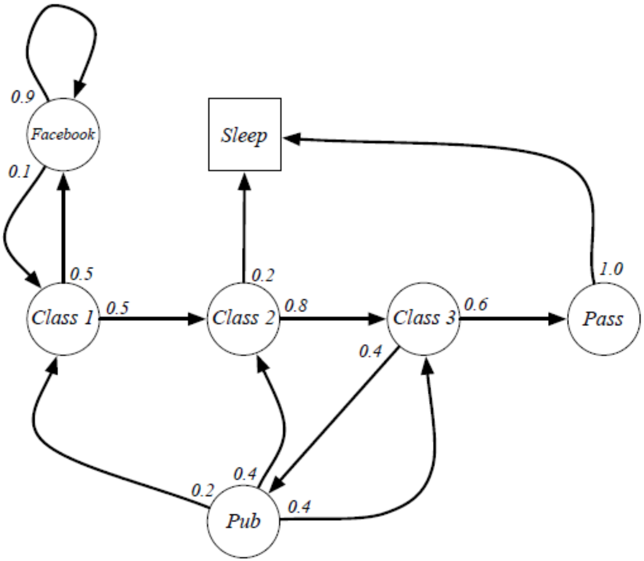
Definition

A MC is *time-homogeneous* when the transition probabilities do not depend on time. Formally,

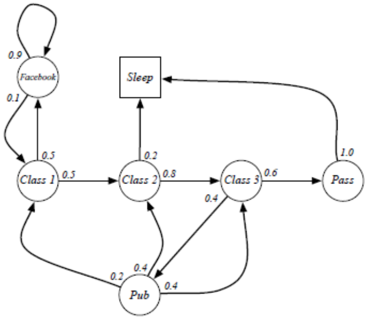
$$\begin{aligned} P(X_{n+1} = x \quad & | X_0 = x_0, X_1 = x_1, \dots, X_n = x_n) && (1) \\ & = P(X_{n+1} = y | X_n = x) = q_{xy}, \ \forall n. && (2) \end{aligned}$$

The matrix Q is called the transition matrix.

Example: Student Markov Chain



Example: Student Markov Chain Transition Matrix



$$\mathcal{P} = \begin{array}{c|ccccccc} & C1 & C2 & C3 & Pass & Pub & FB & Sleep \\ \hline C1 & & 0.5 & & & & 0.5 & \\ C2 & & & 0.8 & & & & 0.2 \\ C3 & & & & 0.6 & 0.4 & & \\ Pass & & & & & & & 1.0 \\ Pub & 0.2 & 0.4 & 0.4 & & & & \\ FB & 0.1 & & & & & 0.9 & \\ Sleep & & & & & & & 1 \end{array}$$

Transitions seen as an operator

In that case, it is convenient to define the operator Q (acting say on bounded functions) as follows

$$Qf(x) = \sum_y q_{xy} f(y)$$

Then, the *Kolomogorov equations* for the transitions of the chain are

$$E[f(X_{n+1})|X_n] = Qf(X_n)$$

Construction of trajectories

A generic construction of a MC is via the recursive sequence

$$X_{n+1} = F(X_n, U_n),$$

where U_n is a sequence of *independent and identically distributed (i.i.d.)* random variables.

Exercise: give an explicit construction of the function F.

Distribution at time n

The transition matrix is instrumental to compute the distribution of the chain at any time:

Proposition

$$P(X_n = y | X_0 = x) = Q^n(x, y) \tag{3}$$

For $n = 2$:

$$\begin{aligned} P(X_2 = y | X_0 = x) &= \sum_z P(X_2 = y, X_1 = z | X_0 = x) \\ &= \sum_z P(X_2 = y | X_1 = z, X_0 = x) P(X_1 = z | X_0 = x) \end{aligned}$$

(conditional proba)

$$= \sum_z P(X_2 = y | X_1 = z) P(X_1 = z | X_0 = x)$$

(Markov)

$$= \sum_z Q(x, z) Q(z, y) = Q^2(x, y)$$

Classes of communication

$i \leftrightarrow j$ if there is a path from i to j with probability > 0.
This generates a class relation.

If all the states communicate, the chain is **irreducible**.

Aperiodicity

We suppose that there is no periodic behaviour.

$$gcd\{n : p_{xx}(n) > 0\} = 1.$$

Invariant measures

Theorem

Consider an aperiodic and irreducible chain on a finite state space $\mathcal{S}$ with transitions Q. Then, there exists a distribution ν on $\mathcal{S}$ such that

$$\lim_n Q^n(x, y) =: \nu(y), \qquad \text{for all states } x, y$$

i.e., X_n converges in distribution to X_∞ with distribution ν.

ν is called an invariant distribution.

ν is a left eigenvector

Writing

$$Q^{n+1}(x,y) = \sum_z Q^n(x,z)Q(z,y),$$

and taking limits:

$$\nu(y) = \sum_z \nu(z)Q(z,y) \qquad \text{for all } y$$

These are known as the **global balance equations.**
$\sum_y Q^n(z,y) = 1$, para todo n, z i.e.,

$$\nu Q = \nu.$$

In general, for all n, $\nu Q^n = \nu$:

Hence, if X_0 has distribution ν, then X_n is stationary. for $n \geq 0$.

Law of large numbers for Markov Chains.

Empirical measures

We fix the initial state $X_0 = x$ and define:

$$N_n(x, y) := \sum_{j=1}^{n} \mathbf{1}\{X_j = y\},$$

the number of visits to y of the chain started with $X_0 = x$ and the *empirical distribution*

$$\hat{Q}_n(x, y) := \frac{N_n(x, y)}{n}$$

Note that $\hat{Q}_n$ are r.v.

LLN computations

Let us first compute the mean of $\hat{Q}_n(x, y)$.

$$E\hat{Q}_n(x, y) = \frac{1}{n}\sum_{j=1}^{n} E(\mathbf{1}\{X_j = y\}|X_0 = x) = \frac{1}{n}\sum_{j=1}^{n} Q^j(x, y) \tag{4}$$

Hence:

$$E\hat{Q}_n(x, y) \to \nu(y). \tag{5}$$

Law of large numbers

Theorem

Let X_n an irreducible an aperiodic Markov chain. Then, for all y,

$$\lim_n \hat{Q}_n(x, y) = \nu(y), \qquad a.s. \tag{6}$$

where ν is the invariant distribution.

This is called Monte-carlo estimation.

There is a Central limit Theorem.

Mean value analysis

Example: Gambler's ruin.
Let us define a random walk absorbed in 0 y c (when the player stops playing).

$$X_{n+1} = X_n + \xi_n,$$

$\xi_n = 1$ with probability p
$\xi_n = -1$ with probability $1 - p$.

Mean value analysis (2)

Define T the (random) stopping time (end of the game).

We wish to compute the probability:

$$U(x) = P(X_T = c | X_0 = x).$$

or the average amount won, i.e.

$$V(x) = E(X_T | X_0 = x).$$

Mean value analysis (3)

Proposition

$$U(c) = 1,$$
$$U(0) = 0.$$
$$U(x) = pU(x+1) + (1-p)U(x-1).$$

$$U(x) = P(X_T = c, X_1 = x+1 | X_0 = x) + P(X_T = c, X_1 = x-1 | X_0 = x),$$

Markov reward processes and Markov decision processes

CHANGE OF NOTATIONS (we follow Barto and Sutton from now on).

Markov Reward Process

A Markov Reward Process is a Markov Chain with values

Definition

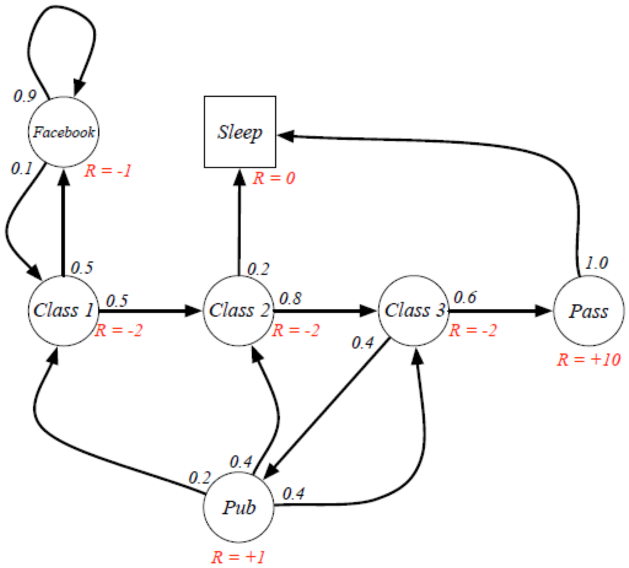
Markov Reward Process is a tuple $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ is a (finite) set of states
- $\mathcal{P}$ is a state transition probability matrix

$$p(s, s') = \mathbb{P}(S_{t+1} = s' | S_t = s)$$

- $\mathcal{R}$ is a reward function $r(s) = \mathbb{E}(R_{t+1} | S_t = s)$
- $\gamma \in [0, 1]$ is a discount factor

Example: Student MRP



Exercises Markov I (to do now !)

Exercise

A miner is at the bottom of a mine and sees three tunnels: 1, 2 and 3. Tunnel 1 leads to the exit in a hour. Tunnel 2 returns to the same crossroads in 2 hours and tunnel 3 returns to the crossroads in 3 hours. Every time the miner is at the crossroads, he chooses one of the tunnels with probability $1/3$*, regardless of what he chose before. Define a Markov reward process describing this situation. Let* T *be the time it takes to leave the mine. Compute* $E(T)$.

Information - Set of policies

Markov Decision Process (MDP)

- observable state and reward
- known reward distribution and transition probabilities
- Action depends on current state and action, $p(s'|s,a)$

Partially Observable Markov Decision Process (POMDP)

- Partially observable state: we know z_t with known $P(s_t = s|z)$
- Observed rewards
- Known reward distribution and transition probabilities

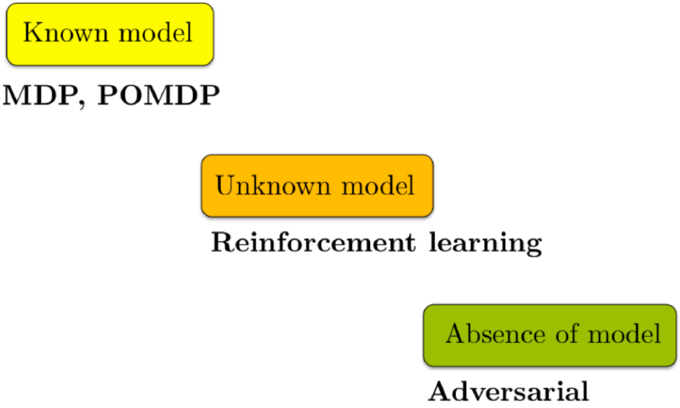
Reinforcement learning

- observable state and reward
- Unknown reward distribution, unknown transition probabilities

Adversarial problems

- observable state and reward
- Arbitrary and time-varying reward function and state transitions

Classification of problems



Major components of an RL Agent

- Policy: A map from states to actions, $\pi(a|s) = \mathbb{P}(A_t = a | S_t = s)$
- Value function: Prediction of future rewards

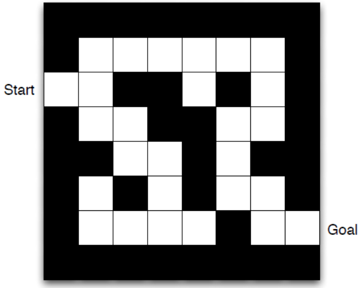
$$v_\pi(s) = \mathbb{E}_\pi(R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | S_t = s)$$

- A model predicts what the environment will do next

$$p(s'|s, a) = \mathbb{P}(S_{t+1} = s' | S_t = s, A_t = a)$$

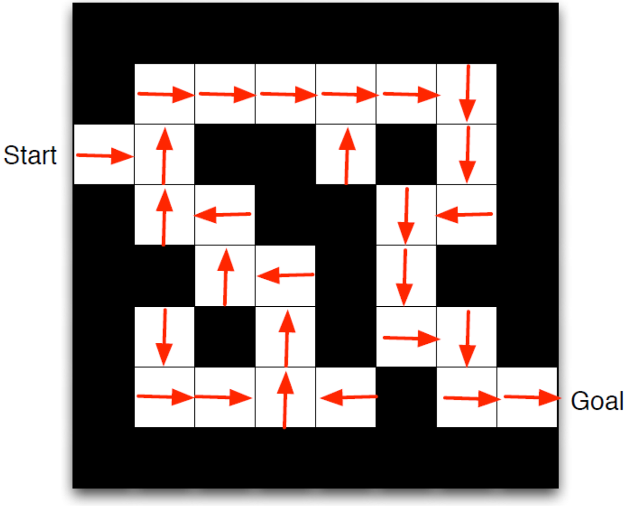
$$r(s, a) = \mathbb{E}(R_{t+1} | S_t = s, A_t = a)$$

Maze Example



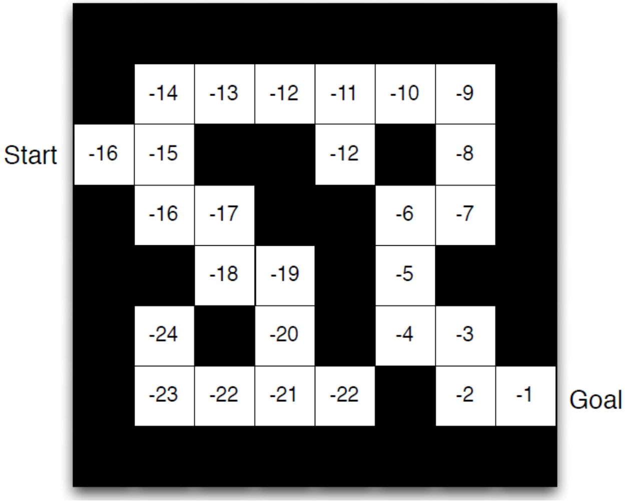
Rewards: -1 per time-step
Actions: N,E,S, W
States: Agent's location

Maze Example: Policy



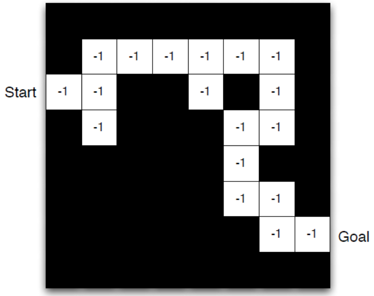
Arrows represent policy $\pi(s)$

Maze Example: Value function



Numbers represent value $v_\pi(s)$ of each state s

Maze Example: Model



Dynamics: how actions change state

Rewards: From visited states

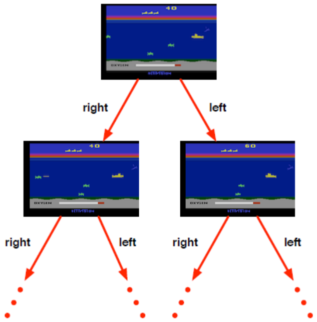
Model may be imperfect

Learning and planning

- Planning. A perfect model of the environment is known
  $\Longrightarrow$ MDP

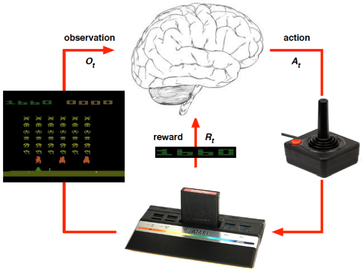
- Reinforcement learning
  - Environment is initially unknown
  - Agent interacts with the environment
  - Agent learns policy

Atari Example: Planning



- Rules known
- Can query emulator
- If I take action a in state s, what would the score and next state be?
- Tree search to find optimal policy

Atari Example: Learning



- Rules unknown
- Learn directly from game-play
- Pick actions on joystic, see pixels and scores

Markov Reward Process

A Markov Reward Process is a Markov Chain with values

Definition

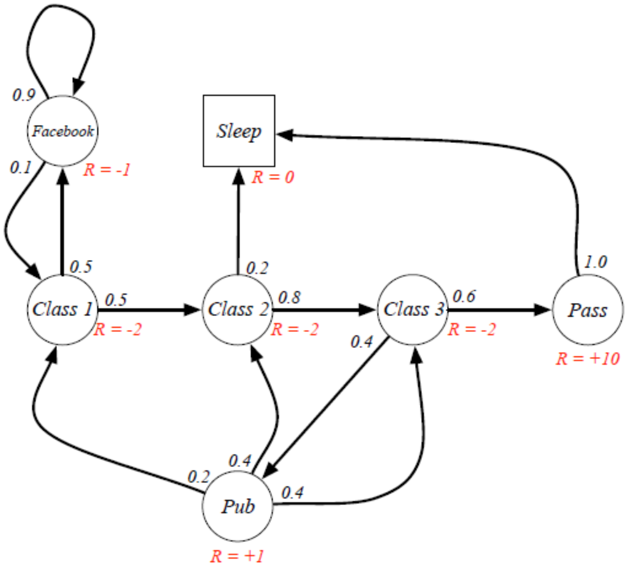
Markov Reward Process is a tuple $\langle \mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ is a (finite) set of states
- $\mathcal{P}$ is a state transition probability matrix

$$p(s, s') = \mathbb{P}(S_{t+1} = s' | S_t = s)$$

- $\mathcal{R}$ is a reward function $r(s) = \mathbb{E}(R_{t+1} | S_t = s)$
- $\gamma \in [0, 1]$ is a discount factor

Example: Student MRP



Return

Definition:

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

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

- The discount γ is the present value of future rewards
- The value of receiving reward R after $k+1$ time-steps is $\gamma^k R$
- Immediate rewards more relevant than future ones
  - γ close to 0 referred as "myopic"
  - γ close to 1 referred as "far-sighted"

Why discount?

Most Markov reward and decision processes are discounted. Why?

- Mathematically convenient to discount rewards
- Avoids infinite returns in non-absorving Markov processes
- If the reward is financial, immediate rewards may earn more interest than delayed rewards
- Animal/human behaviour shows preference for immediate reward

Value function

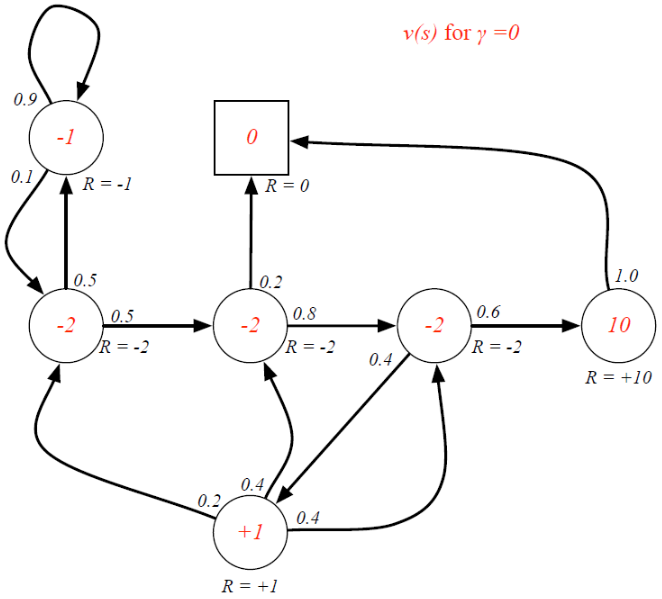
The value function $V(s)$ gives the long-run term value of state s

Definition:

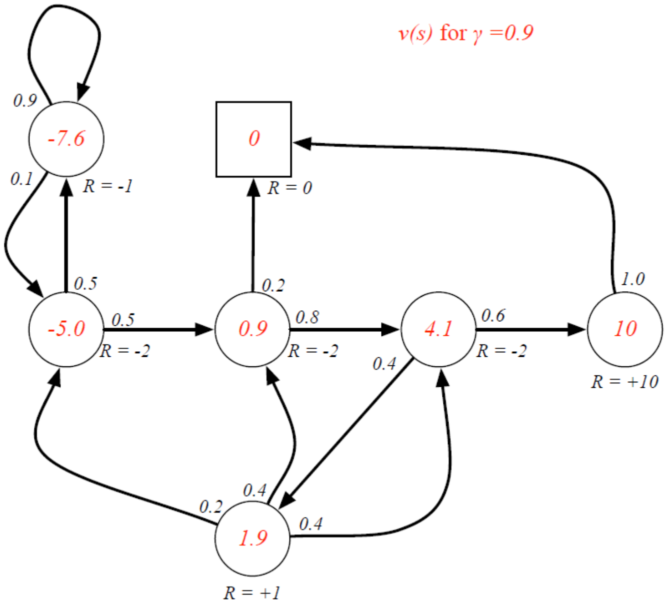
The state value function $V(s)$ of an MRP is the expected return starting from state s:

$$V(s) = \mathbb{E}(G_t | S_t = s)$$

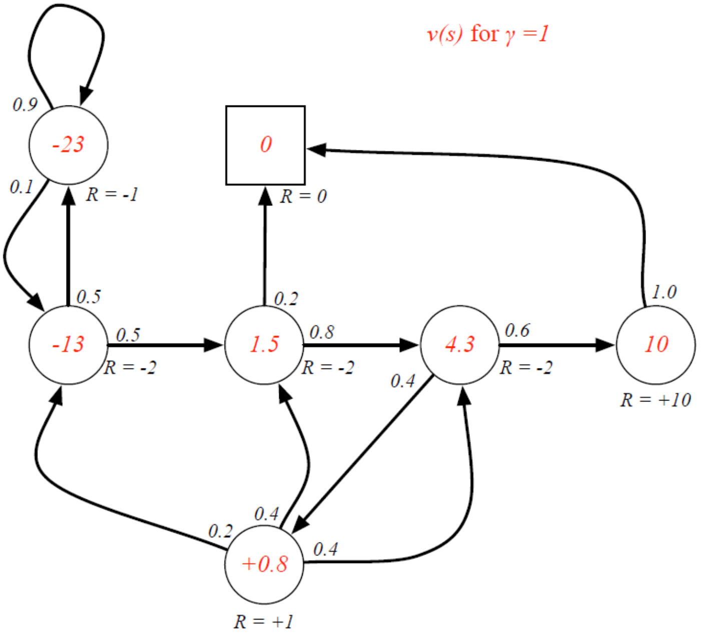
Example: Student MRP



Example: Student MRP



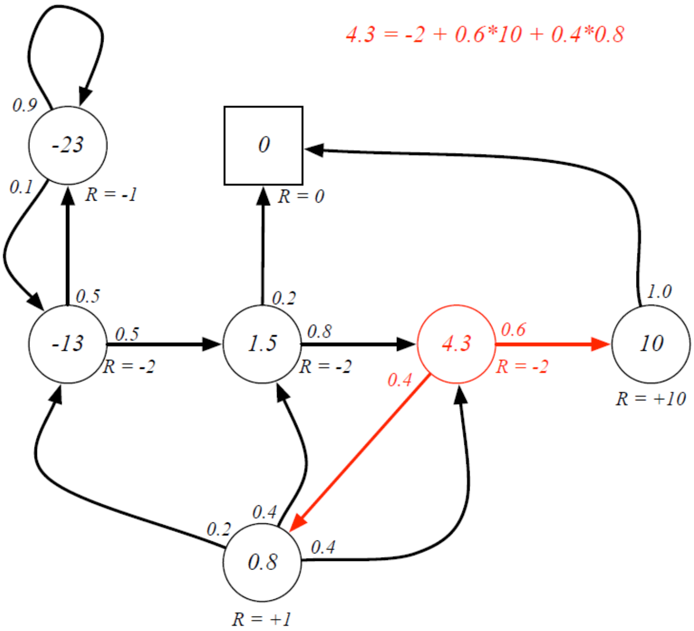
Example: Student MRP



Bellman Equation

$$
\begin{aligned}
V(s) &= \mathbb{E}(G_t | S_t = s) \\
&= \mathbb{E}(R_{t+1} + \gamma R_{t+2} + \ldots | S_t = s) \\
&= r(s) + \gamma \sum_{s'} p(s, s') \mathbb{E}(R_{t+2} + \gamma R_{t+3} + \ldots | S_{t+1} = s') \\
&= r(s) + \gamma \sum_{s'} p(s, s') V(s')
\end{aligned}
$$

Example: Student MRP



Bellman Equation in Matrix Form

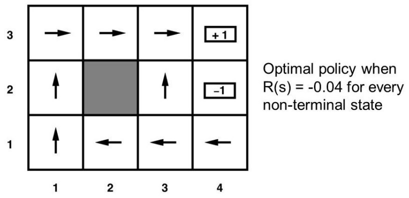
$$\begin{pmatrix} V(1) \\ \vdots \\ V(n) \end{pmatrix} = \begin{pmatrix} r(1) \\ \vdots \\ r(n) \end{pmatrix} + \gamma \begin{pmatrix} p(1,1) & p(1,2) & \cdots & p(1,n) \\ p(2,1) & p(2,2) & \cdots & p(2,n) \\ \vdots & \vdots & \ddots & \vdots \\ p(n,1) & p(n,2) & \cdots & p(n,n) \end{pmatrix} \begin{pmatrix} V(1) \\ \vdots \\ V(n) \end{pmatrix}$$

It can be solved directly: $V = (I - \gamma \mathcal{P})^{-1} \mathcal{R}$

Computational complexity is $O(n^3)$ for n states

Many iterative methods for large MRP: dynamic programming, Monte-Carlo evaluation, Temporal-Difference learning

Example: Grid World



3
2
1
1
2
3
4
+1
–1
Optimal policy when R(s) = -0.04 for every non-terminal state

Policy Matrix: Each action is represented by a number : Action (Up) is represented by 0, (Rigth) by 1, (Down) by 2 and, finally, (Left) by 3

Example: Grid World (cont.)

Transition probabilities: Column 0 represents direction Up, Column 1 represents direction Right, Column 2 represents direction Down and Column 3 represents direction Left.

$$\begin{pmatrix} 0.8 & 0.1 & 0 & 0.1 \\ 0.1 & 0.8 & 0.1 & 0 \\ 0 & 0.1 & 0.8 & 0.1 \\ 0.1 & 0 & 0.1 & 0.8 \end{pmatrix}$$

Exercise: For $\gamma = 0.999$, calculate the value function for all the states

Markov Decision Process (MDP)

An MDP is a Markov Reward process with decisions.

Definition

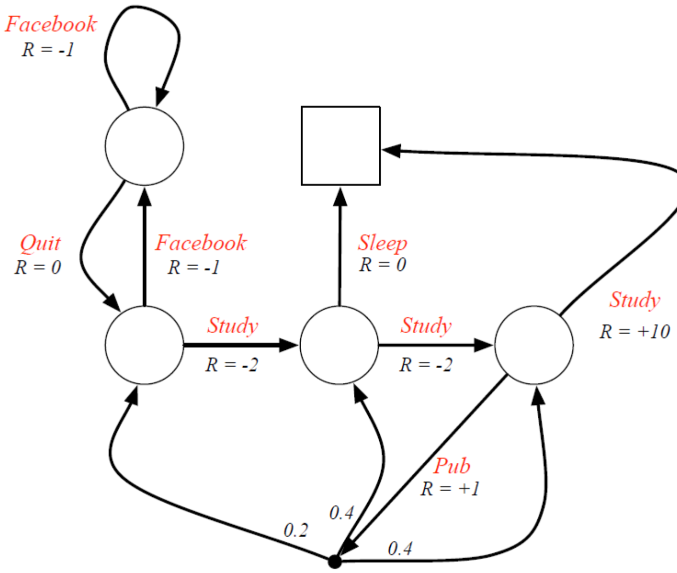
Markov Reward Process is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$

- $\mathcal{S}$ is a (finite) set of states
- $\mathcal{A}$ is a finite set of actions
- $\mathcal{P}$ is a state transition probability matrix

$$p(s'|s,a) = \mathbb{P}(S_{t+1} = s' | S_t = s, \mathcal{A}_t = a)$$

- $\mathcal{R}$ is a reward function $r(s,a) = \mathbb{E}(R_{t+1} | S_t = s, \mathcal{A}_t = a)$
- $\gamma \in [0,1]$ is a discount factor

Example: Student MDP



Policies (1)

Definition:

A policy π is a distribution over actions: :

$$\pi(s) = \mathbb{P}(A_t | S_t = s)$$

- A policy fully defines the behavior of an agent
- Policies are stationary or time-independent

Policies (2)

- Given an MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ and a policy π
- The state sequence $S_1, S_2, \ldots$ is a Markov Process $\langle \mathcal{S}, \mathcal{P}^\pi \rangle$
- The state and reward sequence $S_1, R_2, S_2, R_3 \ldots$ is a Markov reward process $\langle \mathcal{S}, \mathcal{P}^\pi, \mathcal{R}^\pi, \gamma \rangle$

$$p^\pi(s, s') = \sum_{a \in \mathcal{A}} \pi(a|s) p(s'|s, a)$$

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

Value Function

Definition:

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

$$V_\pi(s) = \mathbb{E}(G_t | S_t = s, A_{t:\infty} \sim \pi)$$

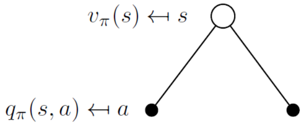
Definition:

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

$$q_\pi(s, a) = \mathbb{E}\left(G_t | S_t = s, A_t = a, A_{t+1:\infty} \sim \pi\right)$$

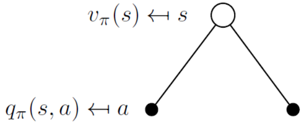
Bellman Expectation Equation

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

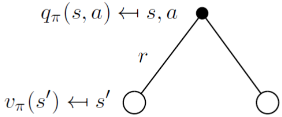


Bellman Expectation Equation

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

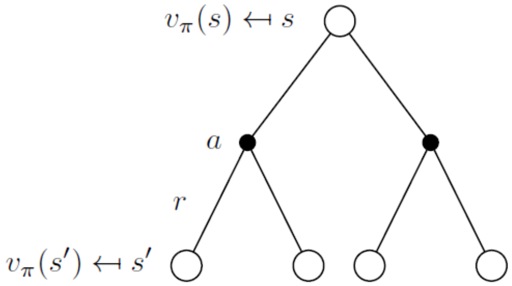


$$q_\pi(s, a) = r(s, a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, a) V_\pi(s')$$



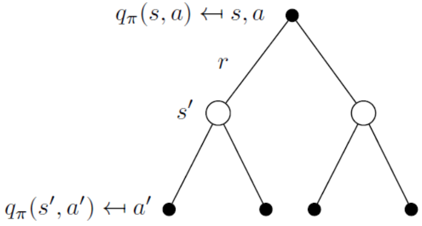
Bellman Expectation Equation for V_π

$$V_\pi(s) = \sum_a \pi(a|s) \left(r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V_\pi(s') \right)$$

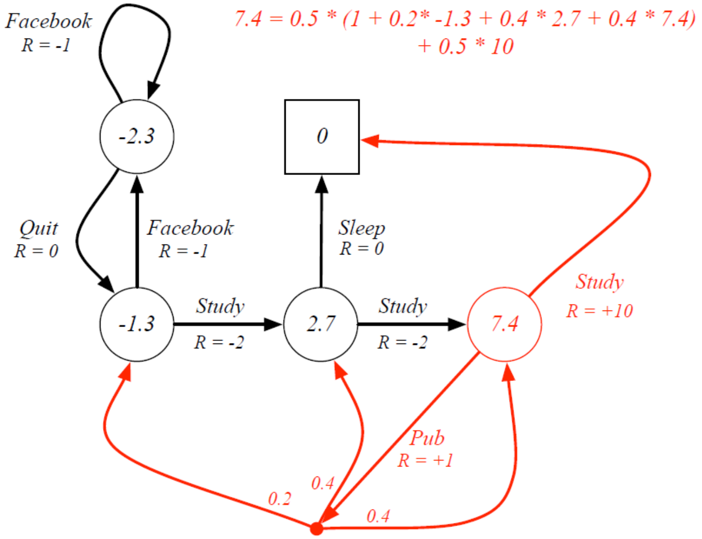


Bellman Expectation Equation for q_π

$$q_\pi(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) \sum_{a' \in \mathcal{A}} \pi(a'|s') q_\pi(s',a')$$


$q_\pi(s,a) \hookleftarrow s,a$
r
s'
$q_\pi(s',a') \hookleftarrow a'$

Bellman Expectation Equation in Student MDP



Optimal Value Function

Definition:

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

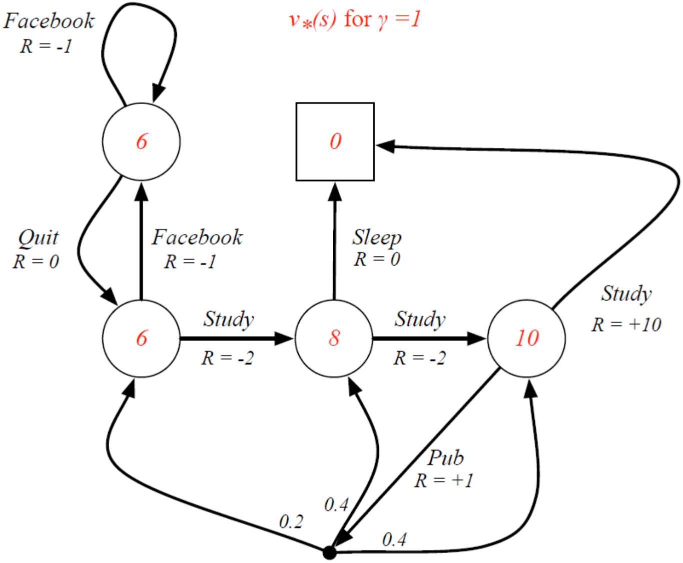
$$V_*(s) = \max_{\pi} V_{\pi}(s)$$

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

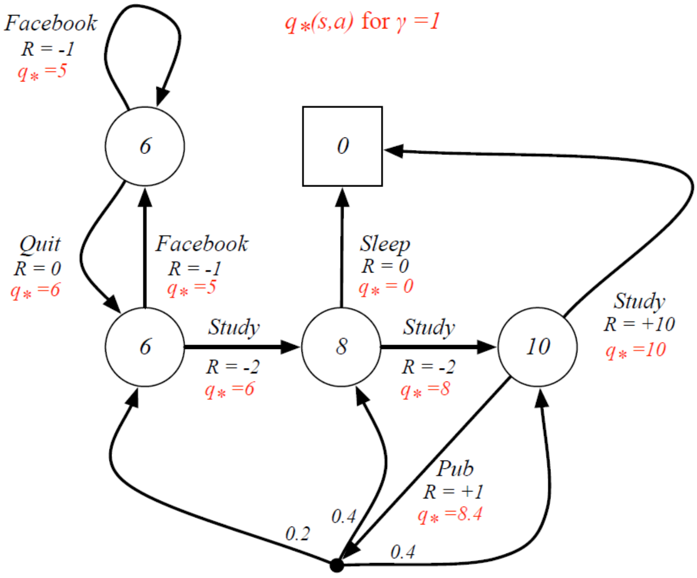
$$q_*(s, a) = \max_{\pi} q_{\pi}(s, a)$$

An MDP is solved if we find either $V_*(s)$ or $q_*(s, a)$

Example: Optimal value function for Student MDP



Example: Optimal action-value function for Student MDP



Optimal Policy

Define a partial ordering over policies

$$\pi \geq \pi' \quad \textit{if} \quad V_\pi(s) \geq V_{\pi'}(s), \forall s$$

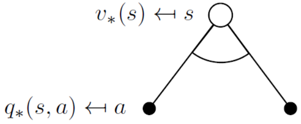
Theorem:

For any Markov Decision Processes

- There exists an optimal policy π_* that is better or equal to all others, i.e., $\pi_* \geq \pi, \forall \pi$
- All optimal policies achieve the optimal value function $V_{\pi_*}(s) = V_*(s)$
- All optimal policies achieve the optimal action-value function $q_{\pi_*}(s, a) = q_*(s, a)$

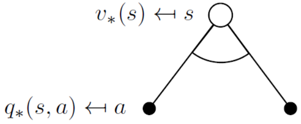
Bellman Optimality Equation

$$V_*(s) = \max_{a \in \mathcal{A}} q_*(s, a)$$

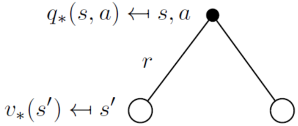


Bellman Optimality Equation

$$V_*(s) = \max_{a \in \mathcal{A}} q_*(s, a)$$

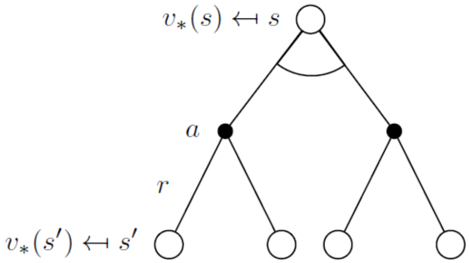


$$q_*(s, a) = r(s, a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s, a) V_*(s')$$



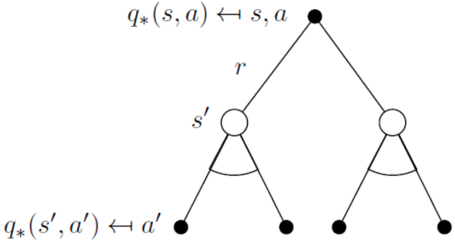
Bellman Optimality Equation for V_*

$$V_*(s) = \max_{a \in \mathcal{A}} \left(r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) V_*(s') \right)$$

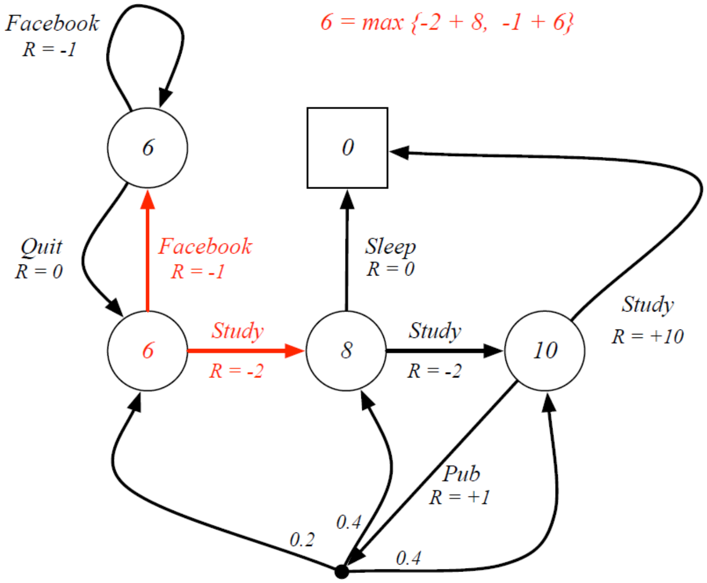


Bellman Optimality Equation for q_*

$$q_*(s,a) = r(s,a) + \gamma \sum_{s' \in \mathcal{S}} p(s'|s,a) \max_{a' \in \mathcal{A}} q_*(s',a')$$



Bellman Optimality Equation in Student MDP

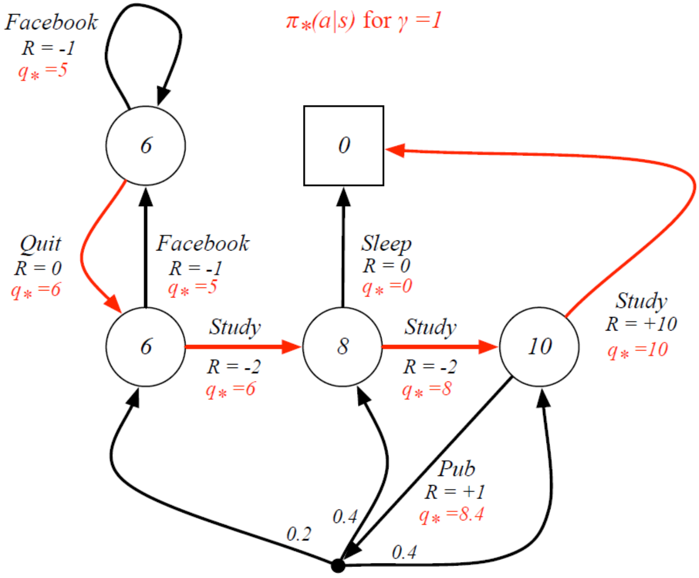


Finding an Optimal policy

An optimal policy can be found by maximizing over $q_*(s, a)$

$$\pi_*(a|s) = \begin{cases} 1 \quad \textit{if} \quad a = \mathrm{argmax}_{a \in \mathcal{A}} q_*(s, a) \\ 0 \qquad\qquad\qquad \textit{otherwise} \end{cases}$$

Bellman Optimality Equation in Student MDP



Finite Markov Decision Process (MDP)

$$\begin{aligned} V_T^{\pi}(i) &= \mathbb{E}\left(R_1 + R_2 + \cdots + R_T | S_0 = i\right) \\ &= \mathbb{E}_{\pi}(\sum_{t=1}^{T} R_t | S_0 = i) \end{aligned}$$

$$V_t(i) = sup_{\pi} V_t^{\pi}(i)$$

Finite Markov Decision Process (MDP)

Imagine action a, which yields reward $r(i,a)$. The best we can do from now on is

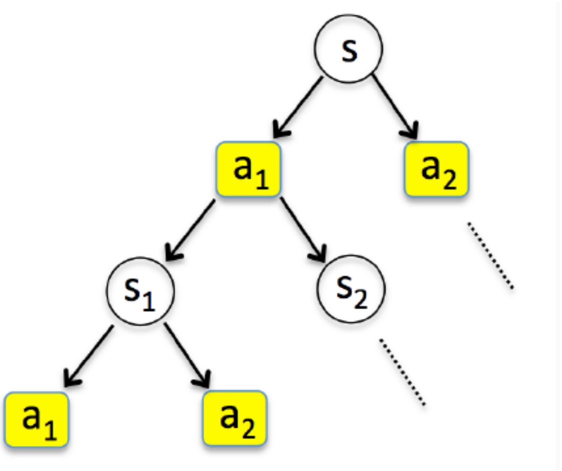
$$r(i,a) + \sum_j p(j|i,a) V^{T-1}(j)$$

Then, the best action is

$$V^T(i) = \max_a \left(r(i,a) + \sum_j p(j|i,a) V^{T-1}(j) \right)$$

Known as Optimality Equation, Dynamic Programming, Howard Equation,...

Bellman's key idea



Decision tree with depth T: $A^T S^{T+1}$ leaves (optimizing over history dependent policies)
Dynamic Programming: S^2AT operations

Richard Bellman

1920 - 1984
American applied mathematician

Introduced **Dynamic Programming** (DP) as a method for solving a complex problem by breaking it down into a collection of simpler subproblems, solving each of those subproblems just once, and storing their solutions.

Extensions to MDPs

- Infinite and continuous MDPs
- Partially observable MDPs
- Undiscounted, average reward MDPs