6.7900 Machine Learning (Fall 2023)

Lecture 16: Markov Decision Processes (supporting slides)

Shen Shen

Outline

- Markov Decision Processes
  - Definition
  - Value Functions
  - Policy Evaluation
  - Optimal value functions
  - Value Iteration
  - Q Value Functions and Q-value iteration

- Reinforcement Learning
  - Formulation
  - Q-learning

References

- More RL-flavored presentation:
  - Reinforcement Learning: An Introduction, Sutton and Barton; The MIT Press, 2018. Chapter 3, 4, 6
- More control- and optimization-flavored presentation:
  - Dynamic programming and optimal control; D. P. Bertsekas; Athena Scientific, 2012. Volume 1, Chapter
- More planning and AI-flavored presentation:
  - Artificial Intelligence: A Modern Approach; Russell and Norvig; Pearson, 2021. Chapter 16.1, 16.2,
  - Algorithms for Decision Making; Kochenderfer, Wheeler, and Wray, The MIT Press, 2022. Chapter 7.

- Some slides adapted from: Devavrat Shah, Leslie Kaelbling, Philip Isola, and Pieter Abbeel

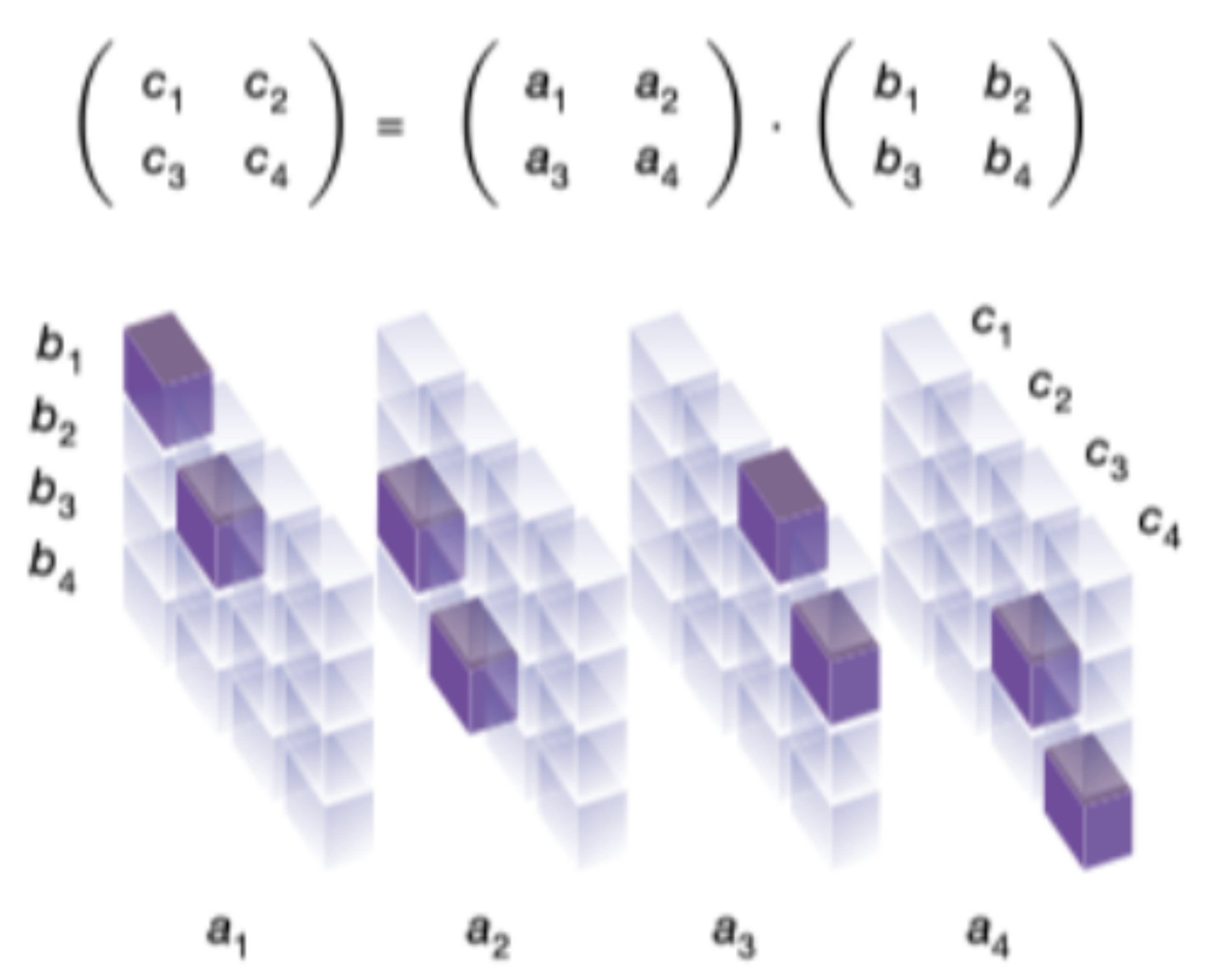
Recent (Deep) RL Highlights

Discovering faster matrix multiplication algorithms with reinforcement learning

https://doi.org/10.1038/s41586-022-05172-4
Received: 2 October 2021
Accepted: 2 August 2022
Published online: 5 October 2022

Alhussein Fawzi[1,2✉], Matej Balog[1,2], Aja Huang[1,2], Thomas Hubert[1,2], Bernardino Romera-Paredes[1,2], Mohammadamin Barekatain[1], Alexande
Francisco J. R. Ruiz[1], Julian Schrittwieser[1], Grzegorz Swirszcz[1], David Si
& Pushmeet Kohli[1]

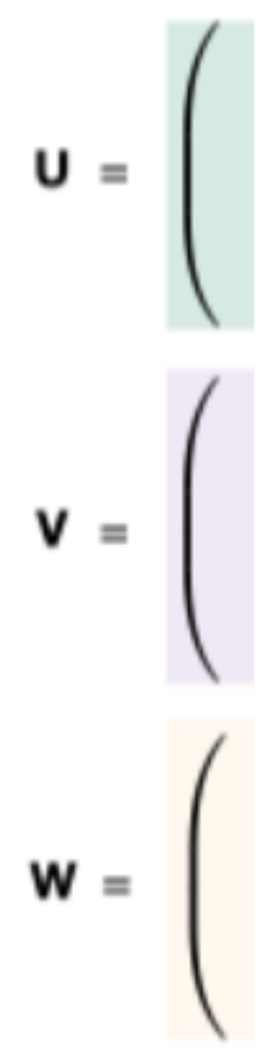
a



b

$$m_1 = (a_1 + a_4)(b_1 + b_4)$$
$$m_2 = (a_3 + a_4)\, b_1$$
$$m_3 = a_1\,(b_2 - b_4)$$
$$m_4 = a_4\,(b_3 - b_1)$$
$$m_5 = (a_1 + a_2)\, b_4$$
$$m_6 = (a_3 - a_1)(b_1 + b_2)$$
$$m_7 = (a_2 - a_4)(b_3 + b_4)$$
$$c_1 = m_1 + m_4 - m_5 + m_7$$
$$c_2 = m_3 + m_5$$
$$c_3 = m_2 + m_4$$
$$c_4 = m_1 - m_2 + m_3 + m_6$$

c



Size (n, m, p)	Best method known	Best rank known	AlphaTensor rank Modular	Standard
(2, 2, 2)	(Strassen, 1969)[2]	7	7	7
(3, 3, 3)	(Laderman, 1976)[15]	23	23	23
(4, 4, 4)	(Strassen, 1969)[2] (2, 2, 2) ⊗ (2, 2, 2)	49	47	49
(5, 5, 5)	(3, 5, 5) + (2, 5, 5)	98	96	98
(2, 2, 3)	(2, 2, 2) + (2, 2, 1)	11	11	11
(2, 2, 4)	(2, 2, 2) + (2, 2, 2)	14	14	14
(2, 2, 5)	(2, 2, 2) + (2, 2, 3)	18	18	18
(2, 3, 3)	(Hopcroft and Kerr, 1971)[16]	15	15	15
(2, 3, 4)	(Hopcroft and Kerr, 1971)[16]	20	20	20
(2, 3, 5)	(Hopcroft and Kerr, 1971)[16]	25	25	25
(2, 4, 4)	(Hopcroft and Kerr, 1971)[16]	26	26	26
(2, 4, 5)	(Hopcroft and Kerr, 1971)[16]	33	33	33
(2, 5, 5)	(Hopcroft and Kerr, 1971)[16]	40	40	40
(3, 3, 4)	(Smirnov, 2013)[18]	29	29	29
(3, 3, 5)	(Smirnov, 2013)[18]	36	36	36
(3, 4, 4)	(Smirnov, 2013)[18]	38	38	38
(3, 4, 5)	(Smirnov, 2013)[18]	48	47	47
(3, 5, 5)	(Sedoglavic and Smirnov, 2021)[19]	58	58	58
(4, 4, 5)	(4, 4, 2) + (4, 4, 3)	64	63	63
(4, 5, 5)	(2, 5, 5) ⊗ (2, 1, 1)	80	76	76

History

- Research area initiated in the 1950s (Bellman), known under various names (in various communities):
  - Reinforcement learning (Artificial Intelligence, Machine Learning)
  - Stochastic optimal control (Control theory)
  - Stochastic shortest path (Operations research)
  - Sequential decision making under uncertainty (Economics)
- Markov decision processes, dynamic programming
- Control of dynamical systems (under uncertainty)
- A rich variety of (accessible & elegant) theory/math, algorithms, and applications/illustrations

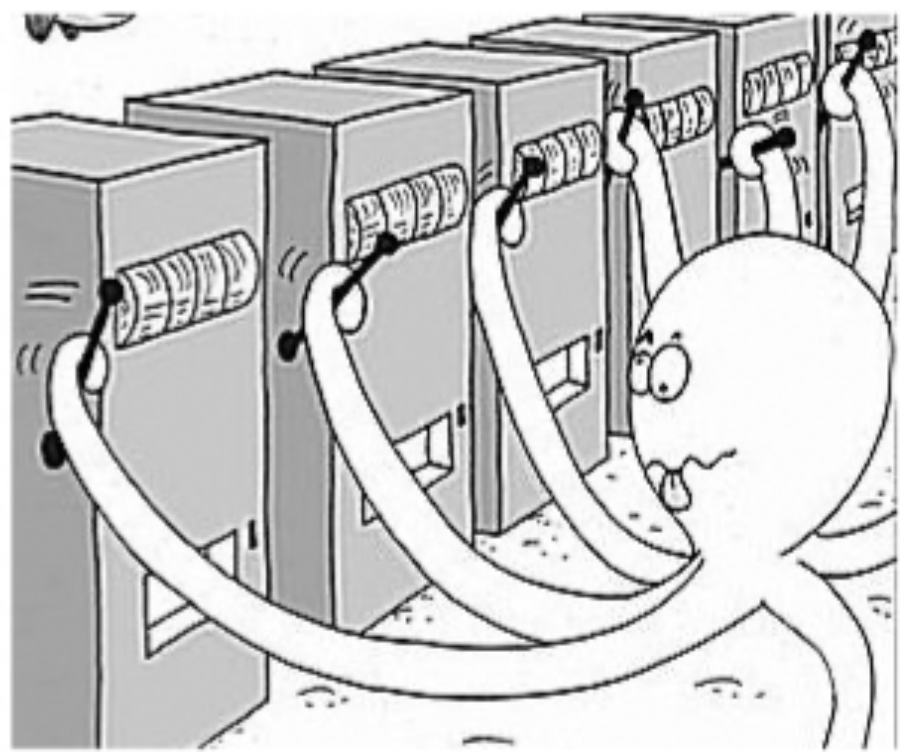
- Diverse community leads diverse notations. We will use the most RL-flavored.
- Many moving pieces, often times, nuanced details. Try draw comparisons.
- Stop by office hours if having any questions.

Recall: Multi-armed Bandits

- k-armed bandit: k, number of action choices
- A_t: the action selected on time step t
- Each action returns a ***reward***
  - R_t: the reward received for selecting a_t
  - *drawn from an* ***unknown*** *reward distribution*
- Goal: pick actions to maximize e.g.

sum of rewards $\max_{A_1,A_2,\ldots,A_T} \sum_{t=1}^{T} \mathbb{E}[R_t | A_t]$

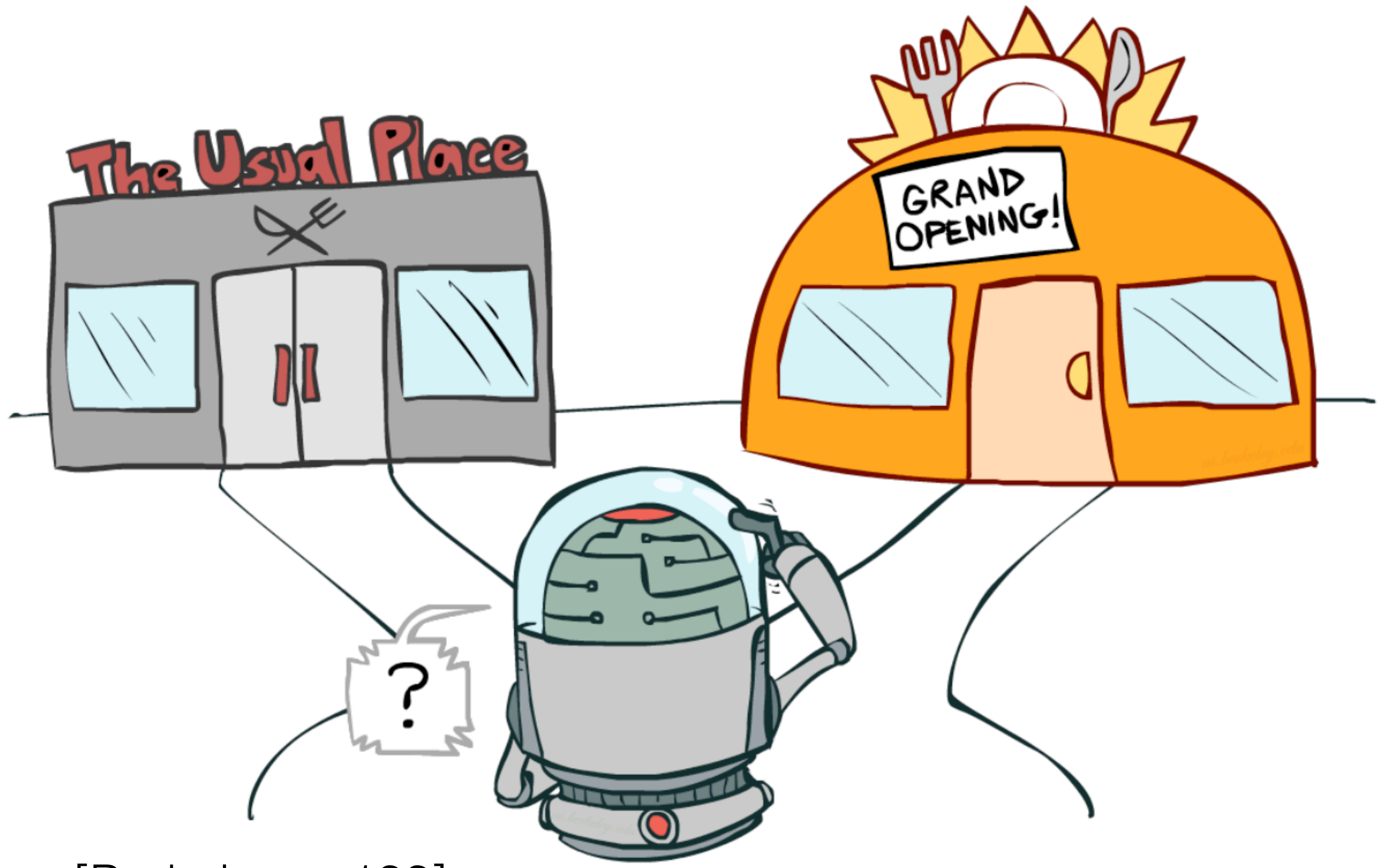
or, average expected rewards $\max_{A_1,A_2,\ldots,A_T} \frac{1}{T} \sum_{t=1}^{T} \mathbb{E}[R_t | A_t]$ with $T \to \infty$



FEYNMAN'S ORIGINAL RESTAURANT PROBLEM
has recently been deciphered from his notes.
READ MORE HERE

[https://www.feynmanlectures.caltech.edu/info/exercises/Feynmans_restaurant_problem.html

[Seinfeld S6E9]



[Berkeley cs188]

Bandits problems:
☹️ Hard: unknown rewards distribution.
(Face “exploration vs. exploitation”.)

😀 Easy: actions don’t “change” the world.

Classification of Decision-Making Scenarios

	Actions Don't Impact State	Actions Change State
Unknown Model	Multi-Armed Bandits	Reinforcement Learning
Known Model	Stochastic Optimization	Markov Decision Process

Here, "model", or sometimes "world model", refers to transition model and/or rewards model.
See next slide for precise definition.

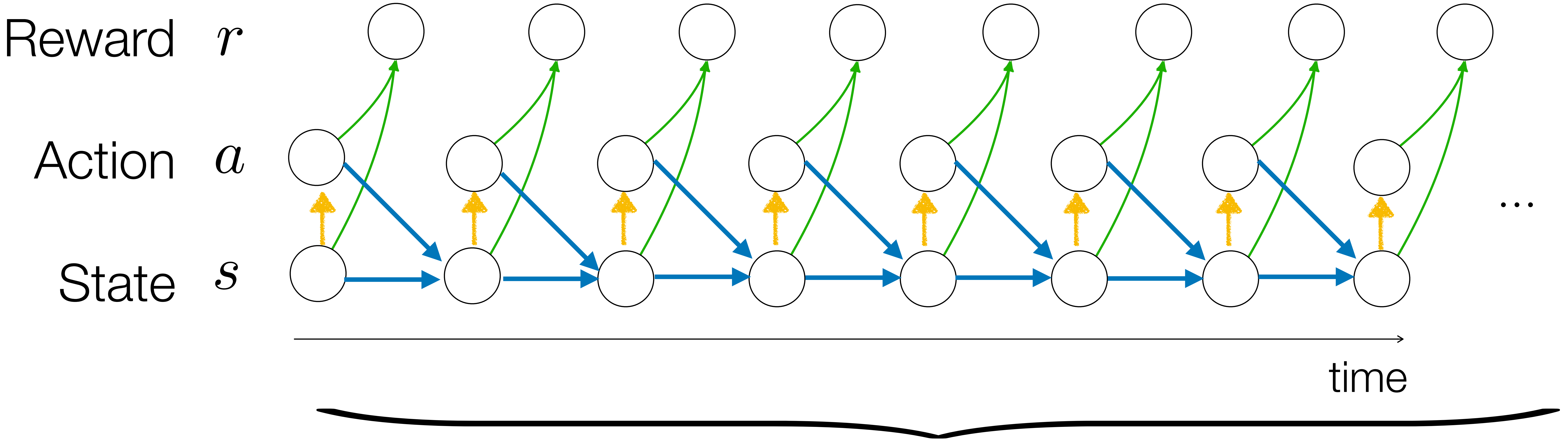
Markov Decision Process

- $\mathcal{S}$: a state space which contains all possible states s of the system.
- $\mathcal{A}$: an action space which contains all possible actions a an agent can take.
- $P(s' \mid s, a)$: the probability of transition from state s to s' if action a is taken.
- $R(s, a)$: a function that takes in the (state and action) and returns a real-valued reward.

Sometimes, also:

- s_0: initial state.
- Objective version (may involve a $\gamma \in [0,1]$: discount factor (details later), and/or T: horizon. Details later).

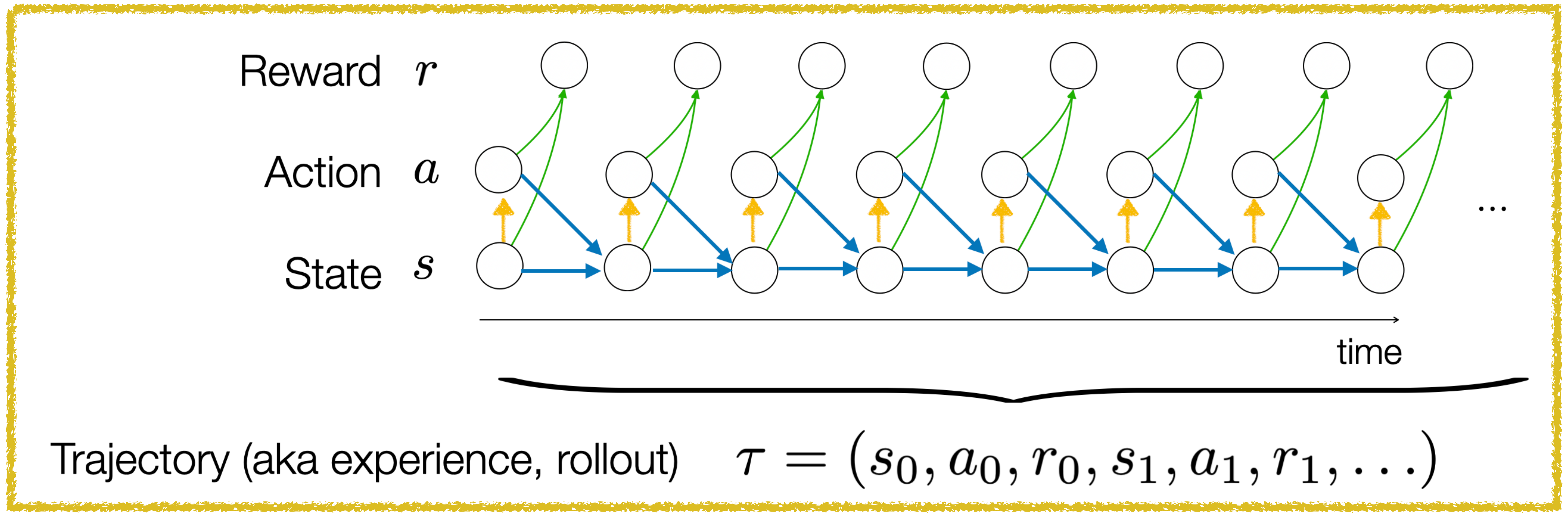
MDP as a graphical model



Trajectory (aka experience, rollout) $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots)$

Policy $\pi(s)$
Transition $P(s' \mid s, a)$
Reward $R(s, a)$

MDP Assumptions and Comments



- Markov Assumption (Memory-less)
  $P\left(S_{t+1} = s' \mid s_t, a_t, s_{t-1}, a_{t-1}, \ldots s_0\right) = P\left(S_{t+1} = s' \mid s_t, a_t\right)$
- Full-observability of states (if this is violated, would become POMDP)
- Policy $\pi(s)$ can be thought of feedback-control law. Policy + MDP = Markov chain.
- Notice the sources of randomness in τ

MDP Goal

- Find a policy $\pi : S \to A$, such that:

  $\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) \mid s_0 = s]$ is maximized for all s_0

- This objective is the "infinite-horizon discounted sum of rewards" setting

- Need $0 \leq \gamma < 1$ to get a finite sum. But discount γ also act to:
  - Tradeoff between near-term versus far-ahead rewards
  - Act like "Compound" interest rate
  - As if the process terminates with probability $1 - \gamma$ after every time-step

DISCOUNTING FOR PUBLIC POLICY:
THEORY AND RECENT EVIDENCE ON THE MERITS OF
UPDATING THE DISCOUNT RATE

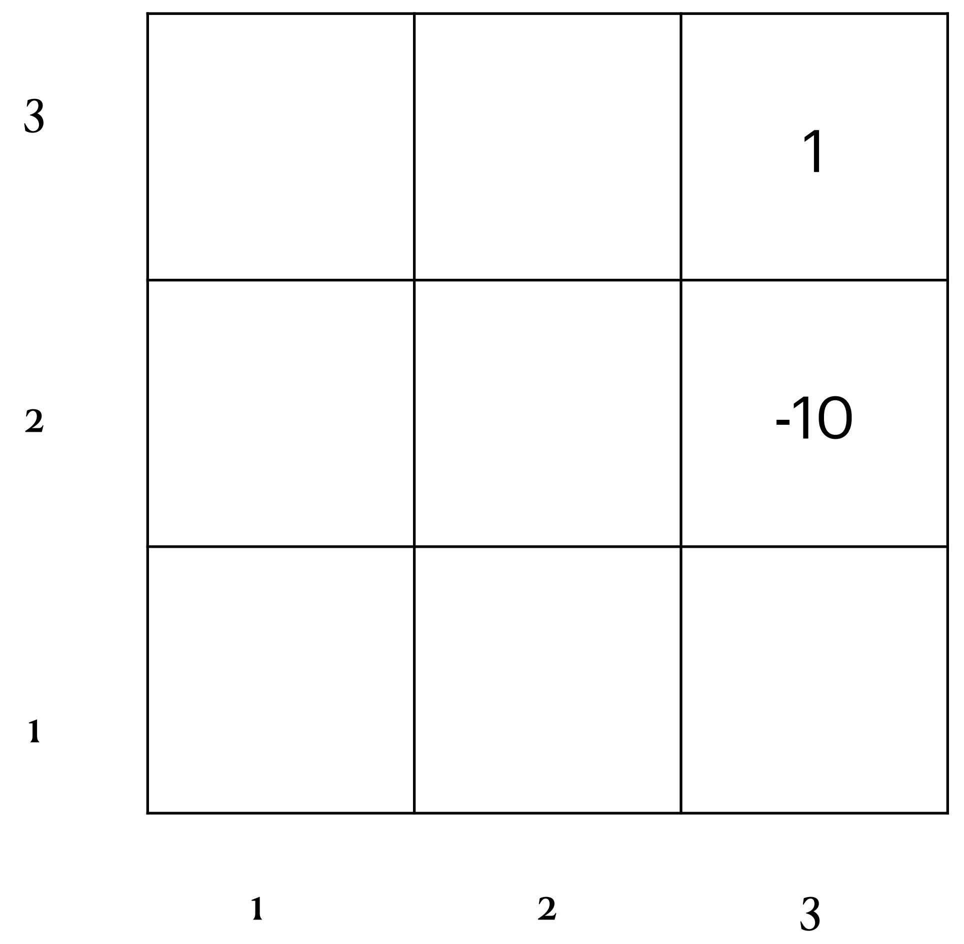
Weighing benefits and costs that take place over time requires discounting those amounts to present value equivalents. This necessitates selecting a discount rate which can adjust for the fact that resources are more valuable today than in the future if consumers prefer to consume today rather than wait, or if firms could be earning a positive return on invested resources. Current guidance from the office of management and budget requires using both a 7 percent and 3 percent real discount rate in regulatory benefit-cost analyses. This issue brief reassesses the current choice of discount rates and methodologies for selecting the 3 percent and 7 percent rates. Empirical evidence suggests that real interest rates around the world have come down since the last evaluation of the rates, and new theoretical advances considering

γ = 0.93 or 0.97

- Many objective variations exist. E.g., infinite-horizon non-discounted average rewards; finite-horizon sum of rewards; finite-horizon un-discounted sum of rewards (this is equivalent to adding an absorbing state in the infinite-horizon counterpart).
- These objective settings differ mostly in theoretical guarantees.
- $\mathcal{S}$, $\mathcal{A}$, and π setup also affect theoretical guarantees, but they also affect computation.

- For the rest of this lecture:
  - $\mathcal{S}$ is discrete and finite
  - $\mathcal{A}$ is discrete and finite
  - π is deterministic

Example: Grid World



(Almost deterministic) Transitions:

- Normally, actions take us deterministically to the "intended" state. E.g., in state (1,1), action "North" gets us to state (1,2)
- If an action would take us out of this world, stay put
- In state (3,2), action "North" leads to two possible next state:
  - 80% chance ends in (3,3)
  - 20% chance ends in (2,3)

State space: 9 cells

Actions space: {North, South, East, West}

Discount $\gamma = 0.9$

Deterministic Rewards:

- State (3,3), any action gets reward +1
- State (3,2), any action gets reward -10
- Any other (state, action) pairs get reward 0

State-Value V Functions

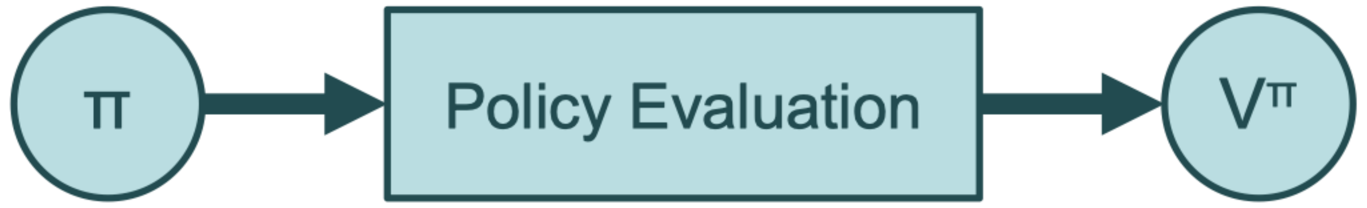
For **any given policy** π, the state-value functions are

$$V^\pi(s) := \mathbb{E}_\tau[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) \,|\, s_0 = s], \forall s$$

Bellman equations

$$V^\pi(s) = \mathbb{E}[R(s, \pi(s))] + \gamma \sum_{s'} p(s' \,|\, s, \pi(s)) V^\pi(s'), \forall s$$

- Policy evaluation:



- For infinite-horizon discounted rewards setup, evaluation amounts to solving a set of $|S|$ **linear** equations

Optimal Policies and Optimal Values

- A policy π^* is optimal if its value functions are no smaller than the value function of any other policy at all states $V^{\pi^*}(s) \geq V^{\pi}(s), \quad \forall s \in S, \forall \pi$

- One way is to enumerate every possible policies, evaluate them and compare. But very inefficient.

- Recall that for any given policy

Bellman equations

$$V^{\pi}(s) = \mathbb{E}[R(s, \pi(s))] + \gamma \sum_{s'} p(s' \mid s, \pi(s)) V^{\pi}(s'), \forall s$$

- We should be convinced of

Bellman optimality equations

$$V^{*}(s) = \max_{a} (\mathbb{E}[R(s, a)] + \gamma \sum_{s'} p(s' \mid s, a) V^{*}(s')), \forall s$$

this would amount to solving a set of $|S|$ **non-linear** equations (how-to later)

Infinite-horizon Summary

- Definition: For **any given policy** π, the state-value functions are

  $V^{\pi}(s) := \mathbb{E}_{\tau}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) \mid s_0 = s], \forall s$

- **Bellman equations**

  $V^{\pi}(s) = \mathbb{E}[R(s, \pi(s))] + \gamma \sum_{s'} p(s' \mid s, \pi(s)) V^{\pi}(s'), \forall s$

- **Bellman optimality equations**

  $V^*(s) = \max_a (\mathbb{E}[R(s, a)] + \gamma \sum_{s'} p(s' \mid s, a) V^*(s')), \forall s$

- There exists a unique V^* satisfying the Bellman optimal equations.
- There exists a stationary deterministic optimal policy π^* (more on how to find this based on V^* later). The optimal policy might not be unique.

Finite-horizon Variant

- Definition: For **any given policy** π, the state-value functions are

  $V_T^{\pi}(s) := \mathbb{E}_{\tau}[\sum_{t=0}^{T-1} \gamma^t R(s_t, \pi(s_t)) \,|\, s_0 = s], \forall s$, where horizon-0 values are all 0.

- **Bellman recursion**

  $V_T^{\pi}(s) = \mathbb{E}[R(s, \pi(s))] + \gamma \sum_{s'} p(s' \,|\, s, \pi(s)) V_{T-1}^{\pi}(s'), \forall s$

- **Bellman optimality recursion**

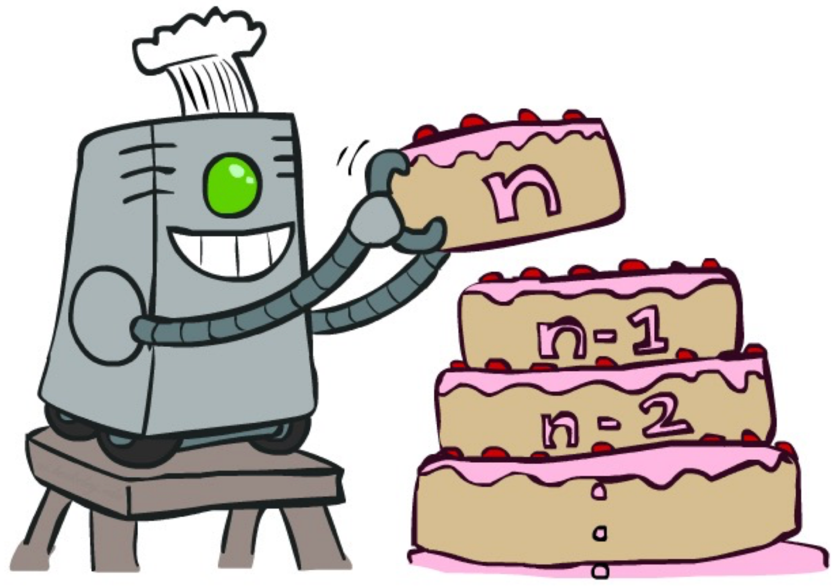
  $V_T^{*}(s) = \max_a (\mathbb{E}[R(s, a)] + \gamma \sum_{s'} p(s' \,|\, s, a) V_{T-1}^{*}(s')), \forall s$

- For a fixed horizon, there exists a unique optimal value function.
- For a fixed horizon, there exists a deterministic optimal policy π^*; the optimal policy may not be unique.
- Optimal value function and optimal policies generally is non-stationary, i.e., depend on horizon.

Value Iteration

Bellman optimality equations

$$V^*(s) = \max_a(\mathbb{E}[R(s,a)] + \gamma \sum_{s'} p(s' \mid s,a) V^*(s')), \forall s$$



$\textsc{ValueIteration}(\mathcal{S}, \mathcal{A}, P, R, \gamma, \epsilon)$

$V(s) = 0$ for $s \in \mathcal{S}$
while True:
    for $s \in \mathcal{S}$:
        $V_{new}(s) \leftarrow \max_a(\mathbb{E}[R(s,a)] + \gamma \sum_{s'} p(s'|s,a) V(s'))$
    if $|V - V_{new}| < \epsilon$:
        return V_{new}
    $V \leftarrow V_{new}$

where $|V_1 - V_2| = \max_s |V_1(s) - V_2(s)|$.

Value Iteration - Comments

- Guaranteed to converge to the infinite-horizon optimal value function V^*.
  (under mild assumptions like rewards bounded in expectation)
- Max-norm error $|V - V^*|$ decreases monotonically per iteration.
- Convergence holds under any initialization.
- When initialized to 0, run line 4 for k iterations, the latest value function would be the optimal horizon-k optimal value function V_k^*.
- Policy extraction: Given optimal $V^*(s)$, how to back out an optimal policy π^*?

$$\pi^*(s) = \arg\max_a \left[\mathbb{E}[R(s,a)] + \gamma \sum_{s'} p(s' | s,a) V^*(s')\right]$$

- E.g., $V^*(s)$ of our grid world:

8.1	9	10
7.29	8.1	-1.18
6.561	7.29	6.561

State-Action-Value Q Functions

- Definition: For any given policy , the state-action-value functions are
  $Q^{\pi}(s, a) := \mathbb{E}[R(s, a)] + \gamma \sum_{s'} p\left(s' \mid s, a\right) V^{\pi}\left(s'\right), \forall s, a$
- Optimal $Q^*(s, a)$ is then expected sum of discounted rewards for being in state s, taking an action a, and act optimally thereafter.
- Optimal quantities must satisfy:
  - $Q^*(s, a) = \mathbb{E}[R(s, a)] + \gamma \sum_{s'} p\left(s' \mid s, a\right) V^*\left(s'\right)$
  - $\mathrm{V}^*(s) = \max_a \mathrm{Q}^*(s, a)$
  - $\boxed{\pi^*(s) = \arg\max_a Q^*(s, a)}$
- Comparing with $\boxed{\pi^*(s) = \arg\max_a \left[\mathbb{E}[R(s, a)] + \gamma \sum_{s'} p(s' \mid s, a) V^*(s')\right]}$ we saw previously, easier to extract optimal policy.

Q Value Iteration

- Recall that optimal quantities must satisfy:
  - $Q^*(s,a) = \mathbb{E}[R(s,a)] + \gamma \sum_{s'} p\left(s' \mid s,a\right) V^*(s')$
  - $V^*(s) = \max_a Q^*(s,a)$
- Similar equations $Q^*(s,a) = \mathbb{E}[R(s,a)] + \gamma \sum_{s'} p\left(s' \mid s,a\right) \max_{a'} Q^*(s',a')$

$\textsc{QValueIteration}(\mathcal{S}, \mathcal{A}, P, R, \gamma, \epsilon)$

$Q(s,a) = 0$ for $s \in \mathcal{S}$, $a \in \mathcal{A}$

while True:

  for $s \in \mathcal{S}$, $a \in \mathcal{A}$:

    $Q_{\text{new}}(s,a) \leftarrow \mathbb{E}[R(s,a)] + \gamma \sum_{s'} p\left(s' \mid s,a\right) \max_{a'} Q(s',a')$

  if $|Q - Q_{\text{new}}| < \epsilon$:

    return Q_{new}

  $Q \leftarrow Q_{\text{new}}$

where $|Q_1 - Q_2| = \max_{s,a} |Q_1(s,a) - Q_2(s,a)|$.

Q Value Iteration - Comments

- Guaranteed to converge to $Q*$ (under mild assumptions like rewards bounded in expectation)
- Can initialize to any value.
- Max-norm error $|Q - Q*|$ decreases monotonically per iteration.
- When initialized to 0, iterations are finite-horizon value functions.
- Can execute "in place" (don't need a separate Q_{new}).
- Can randomly pick (s, a) to update, rather than doing it systematically.
- (If $|Q - Q_{\text{new}}| < \epsilon$ then $|Q - Q*| < \epsilon\gamma/(1 - \gamma)$.)
- (Define greedy policy with respect to value function $\pi_Q(s) = \arg\max_a Q(s, a)$.

  Then if $|Q(s, a) - Q*(s, a)| < \epsilon$, $|V_{\pi_Q} - V*| < 2\epsilon$.)
- Serves as the basis for Q-learning in RL (coming up).

MDP Summary

- Definition
- Policy and V value
- Policy evaluation (via Bellman Linear Equations)
- Finding optimal value functions (requires Bellman non-linear equations)
- Finite-horizon is non-stationary, need to replace equations with recursions
- But both infinite- and finite-horizons can be solved exactly via value iteration
- (V) value iteration cumbersome to extract optimal policy
- Q values
- Q value iteration and policy extraction

Reinforcement Learning

Unknown Model	Multi-Armed Bandits	Reinforcement Learning
Known Model	Stochastic Optimization	Markov Decision Process
	Actions Don't Impact State	Actions Change State

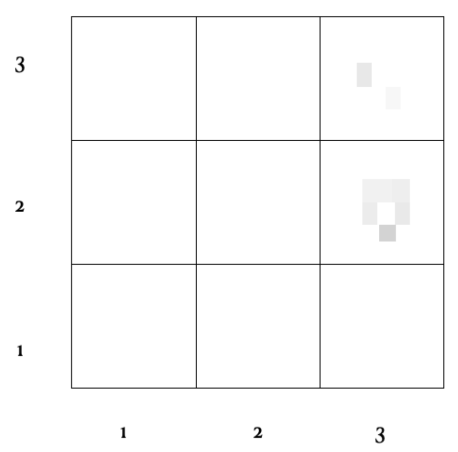
~~Markov Decision Process~~ RL

- $\mathcal{S}$: a state space which contains all possible states s of the system.
- $\mathcal{A}$: an action space which contains all possible actions a an agent can take.
- ~~$P(s' \mid s, a)$: the probability of transition from state s to s' if action a~~ is taken.
- ~~$R(s, a)$: a function that takes in the (state and action) and returns a real-valued~~ reward.

Sometimes, also:

- s_0: initial state.
- Objective version (may involve a $\gamma \in [0,1]$: discount factor (details later), and/or T: horizon. Details later).

Example: Grid World (in RL)



State space: 9 cells

Actions space: {North, South, East, West}

Discount $\gamma = 0.9$

(Almost deterministic) Transitions:

- Normally, actions take us deterministically to the "intended" state. E.g., in state (1,1), action "North" gets us to state (1,2)
- If an action would take us out of this world, stay put
- In state (3,2), action "North" leads to two possible next state:
  - chance ends in (3,3)
  - chance ends in (2,3)

Deterministic Rewards:

- State (3,3), any action gets reward
- State (3,2), any action gets reward
- Any other (state, action) pairs get reward 0

RL ~~MDP~~ Goal

- Find a policy $\pi : S \to A$, such that:

$\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) \mid s_0 = s]$ is maximized for all s_0

Thanks!

Questions?