IS 7033: Artificial Intelligence and Machine Learning

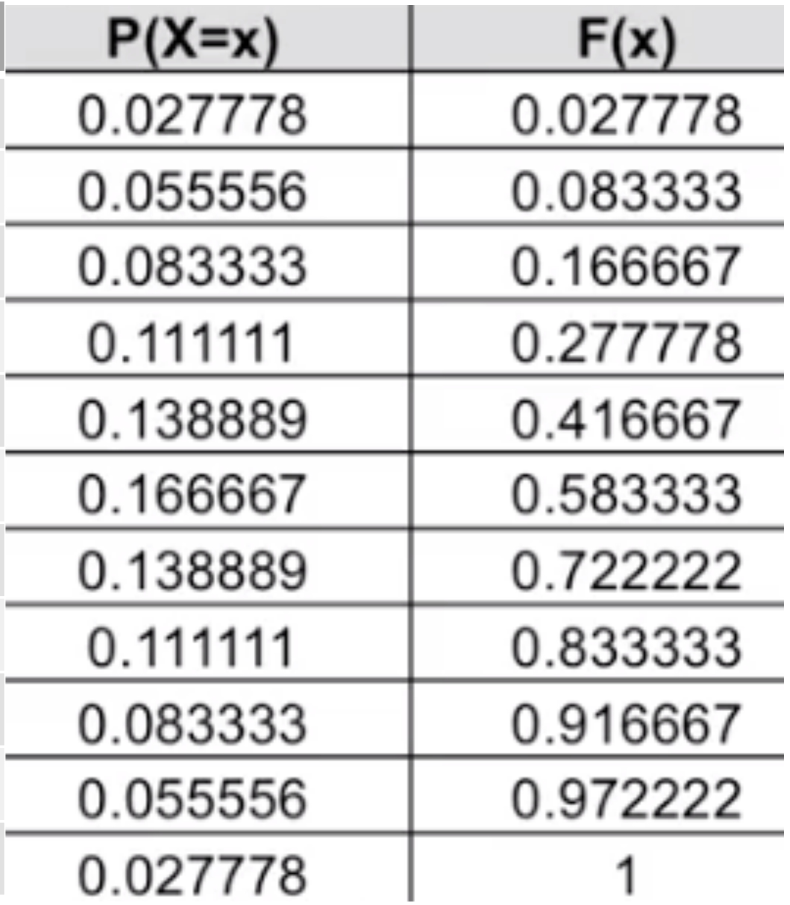
Dr. Peyman Najafirad (Paul Rad)
Associate Professor
Cyber Analytics and AI
210.872.7259

https://github.com/paulNrad/ProbabilisticGraphModels

Random Variable

- A random variable assigns a single numerical value to each basic outcome in the sample space.

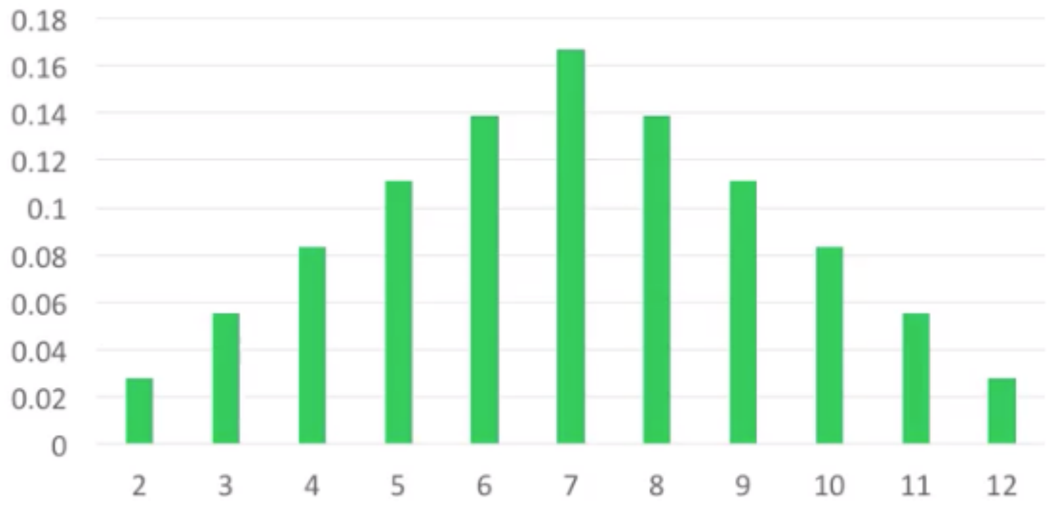
Probability Distributions p(x=X)



x	P(x=X)	P(X=x)	F(x)
2	1/36	0.027778	0.027778
3	1/18	0.055556	0.083333
4	1/12	0.083333	0.166667
5	1/9	0.111111	0.277778
6	5/36	0.138889	0.416667
7	1/6	0.166667	0.583333
8	5/36	0.138889	0.722222
9	1/9	0.111111	0.833333
10	1/12	0.083333	0.916667
11	1/18	0.055556	0.972222
12	1/36	0.027778	1

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

Graphical Representation of P(x=X)



Expected Value

Expected value of a random variable

Mean (or Expected Value) of a discrete variable X is the probability weighted sub of all possible values.

$\mu = E(X) = x_1p(x_1) + x_2p(x_2) + \ldots + x_kp(x_k) = \sum \boldsymbol{x}\, \boldsymbol{p}(\boldsymbol{x})$

$=2\times1/36+3\times2/36+4\times3/36+5\times4/36$

$+6\times5/36+7\times6/36+8\times5/36+9\times4/36$

$+10\times3/36+11\times2/36+12\times1/36$

$= 252/36 = 7$

Var (X)

The variance of a random variable X is denoted by either Var[X] or σ^2_X .
The variance is defined by :

$$\sigma^2_X = E[(x - \mu_X)^2]$$
$$\sigma^2_X = \sum (x - \mu_X)^2 p(x)$$

Bernoulli Random Variable

Bernoulli random variable B allows for only tow outcomes

b=1 and b=0, usually called success and failure

$p(B=1) = p$

$p(B=0) = 1-p$

Mean and Variance

$E(X) = np$

$Var(X)$

Reinforcement Learning

Read Reinforcement Learning by Richard Sutton Chapter 2 and 3

Create Intelligent Behavior?

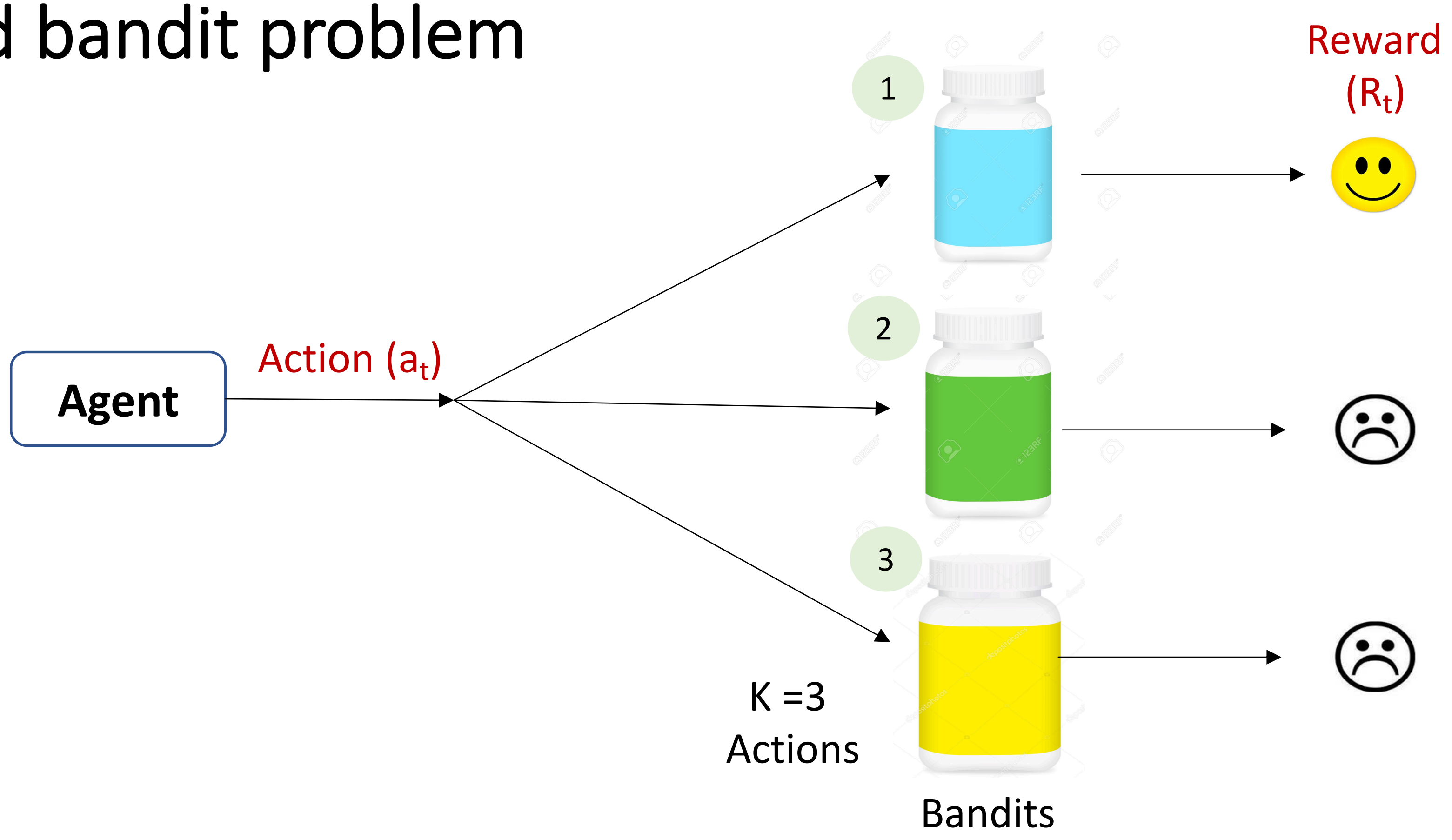
- Give a man a fish and he’ll eat for a day
- Teach a man to fish and he’ll eat for a lifetime
- Give a man a taste for fish and he’ll figure out how to get fish, even if the details change!

In reinforcement learning, the agent generates its own training data by interacting with the world. The agent must learn the consequences of his own actions through trial and error, rather than being told the correct action.

Agenda

- The K-Armed bandit Problem
  - Action Values and Reward
  - Action-value Estimation method
  - Select action greedily using action value function
  - Non-stationary K-Armed bandit
  - Exploration vs. Exploitation Tradeoff
- Markov Decision Processes (MDP)
- Reinforcement Learning
  - State, Action, Reward
- Episodic vs. continuing Tasks

Sequential Decision Making under uncertainty
k-armed bandit problem



An **agent** chooses between “k” actions and receives a reward based on the action it chooses.
The goal of the agent is to maximize the expected reward.

Bandit Algorithm

Repeatedly

Choose Action $\boldsymbol{\alpha} \in \mathbf{A}$

Observe reward $\mathbf{r}$

<u>**Goal**</u>: Maximize reward

Sequential Decision Making with Evolution Feedback
k-armed bandit problem

Action-Value: Value of an action

The value of an action is the **expected reward** when action is taken

$q_*(a)$ is defined as $E(R_t \mid A_t = a)$ $\quad \forall \boldsymbol{a}\ \boldsymbol{\varepsilon}\ \{1, \dots, K\}$

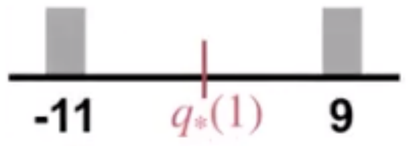
$$= \sum_r p(r|a)\, r$$

The goal of the agent is to maximize the expected reward:

$$\underset{a}{\text{argmax}}\ q_*(a)$$

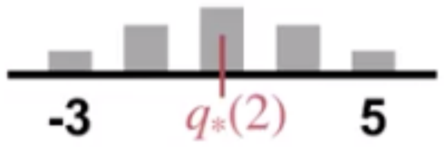
Calculating $q_*(\boldsymbol{a})$

Bernoulli distribution



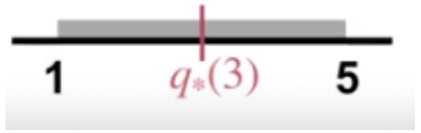
q*(a) = 0.5 x -11 + 0.5 x 9 =

Binomial distribution



q*(a) = 1

Uniform distribution



q*(a) = 3

Action Value Estimation
"Sample Average Method"

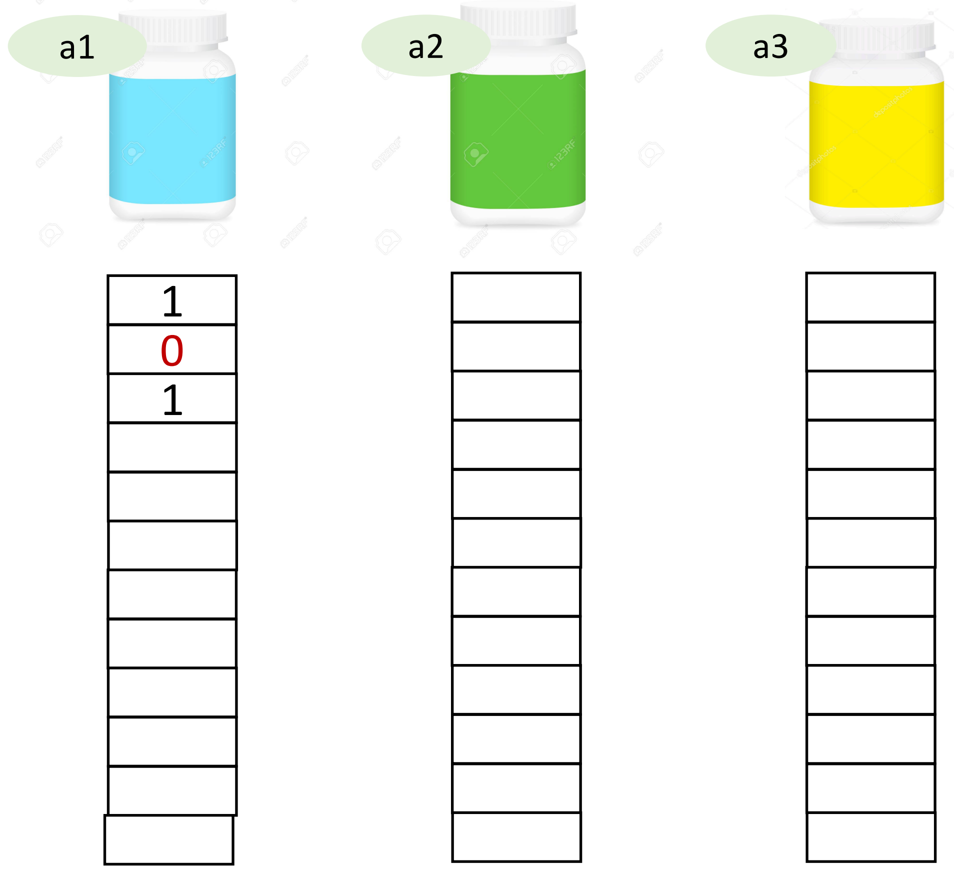
$q_*(a)$ is not known, so we estimate it

$$Q_t(a) = \frac{\textbf{Sum of rewards when "a" taken prior to t}}{\textbf{number of times taken prior to t}}$$

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i}{t-1}$$

Learning Action Value
Sample Average Method

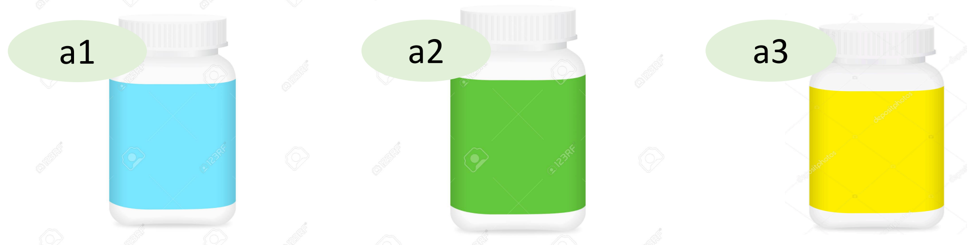
$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i}{t-1}$$



$Q_4(a_1) = 2/3$ $Q_4(a_2) = 0$ $Q_4(a_3) = 0$

Learning Action Value
Sample Average Method

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i}{t-1}$$



1		
0		
1		
	1	
		0
	0	
		1
0		
		1

$Q_{10}(a_1)$ = 2/4 = 0.5

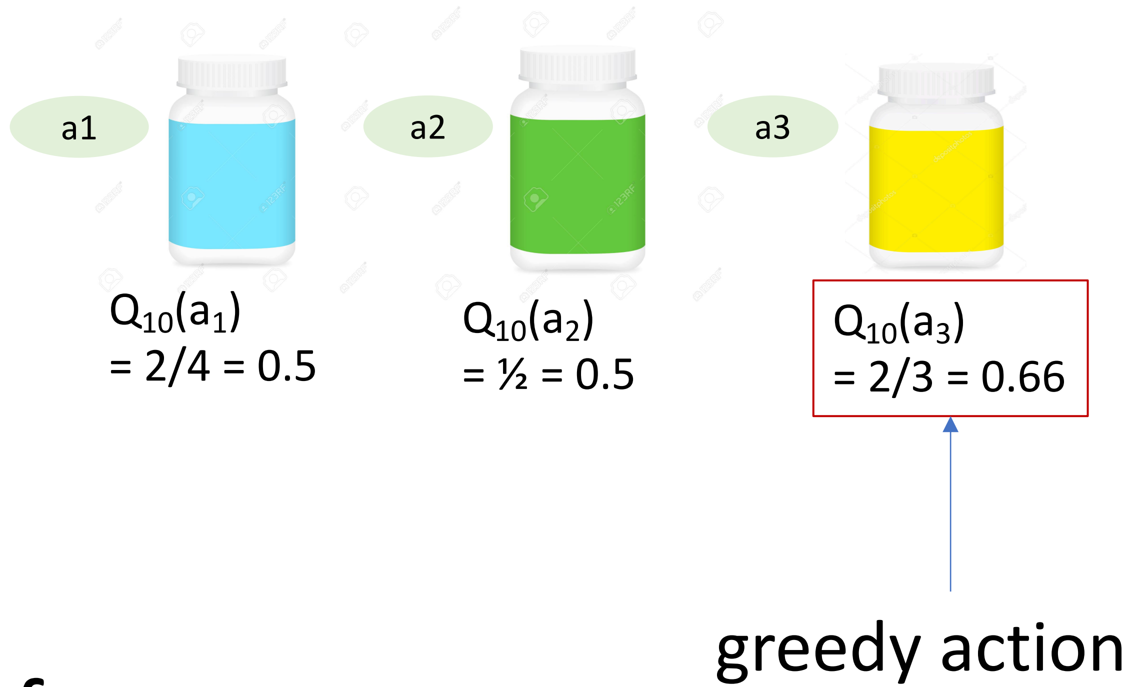
$Q_{10}(a_2)$ = ½ = 0.5

$Q_{10}(a_3)$ = 2/3 = 0.66

Action Selection

Greedy action selection

We can compute the greedy action by taking the argmax of our estimated values



greedy action = argmax Q(a) of a_1, a_2, a_3

exploration-exploitation dilemma in reinforcement learning

The agent may choose to **explore** by choosing a non-greedy action. The agent would sacrifice immediate reward hoping to gain more information about the other actions.

Estimating Action Values Incrementally

- How action values, can be estimated incrementally using the sample-average method
- Identify how the incremental update rule is an instance of a more general update rule
- Describe how the general update rule can be used to solve a non-stationary bandit problem

Estimating Action Values Incrementally

$$Q_{n+1} = \frac{\sum_1^n R_i}{n}$$

$$= \frac{R_n + \sum_1^{n-1} R_i}{n}$$

$$= \frac{R_n + (n-1)/(n-1) \sum_1^{n-1} R_i}{n}$$

$$= \frac{R_n + (n-1) \frac{1}{n-1} \sum_1^{n-1} R_i}{n}$$

$$= \frac{R_n + (n-1)\, Q_n}{n}$$

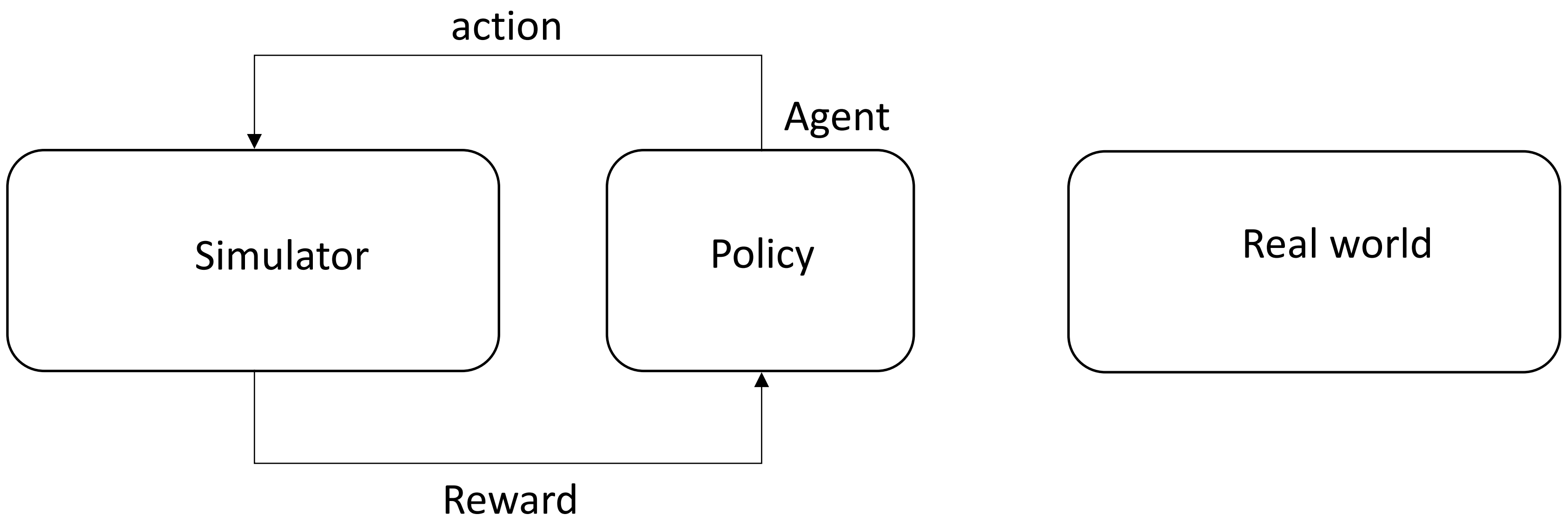
$$= \frac{R_n + nQ_n - Qn}{n}$$

$$= \frac{R_n - Qn}{n} + Q_n$$

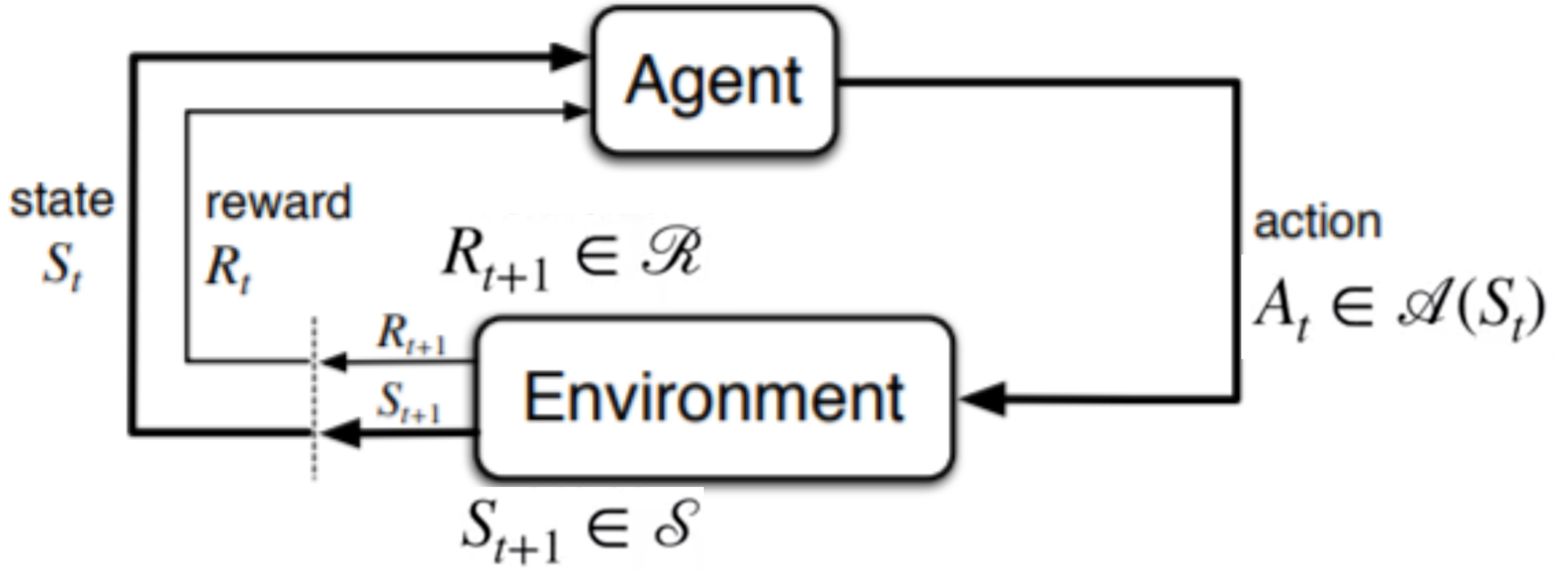
$$\mathbf{Q_{n+1} = Q_n + \frac{1}{n} (R_n - Q_n)}$$

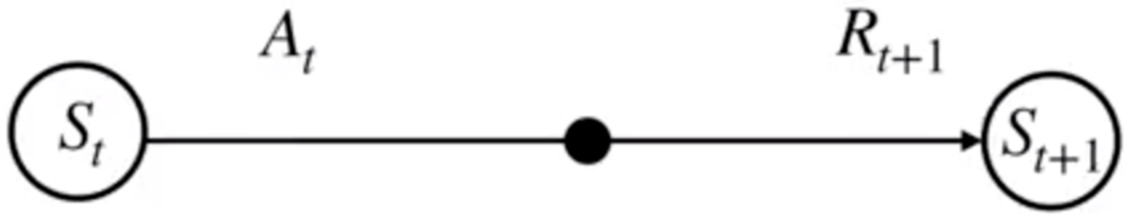
Action Value New estimate = Old estimate + step size $\overbrace{\textbf{(Target - Old Estimate)}}^{\text{Error}}$

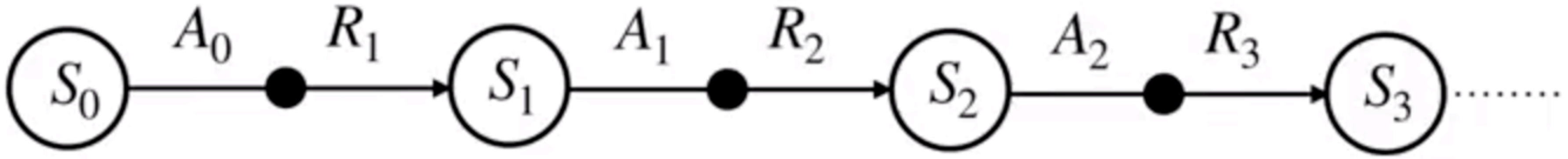
Real World



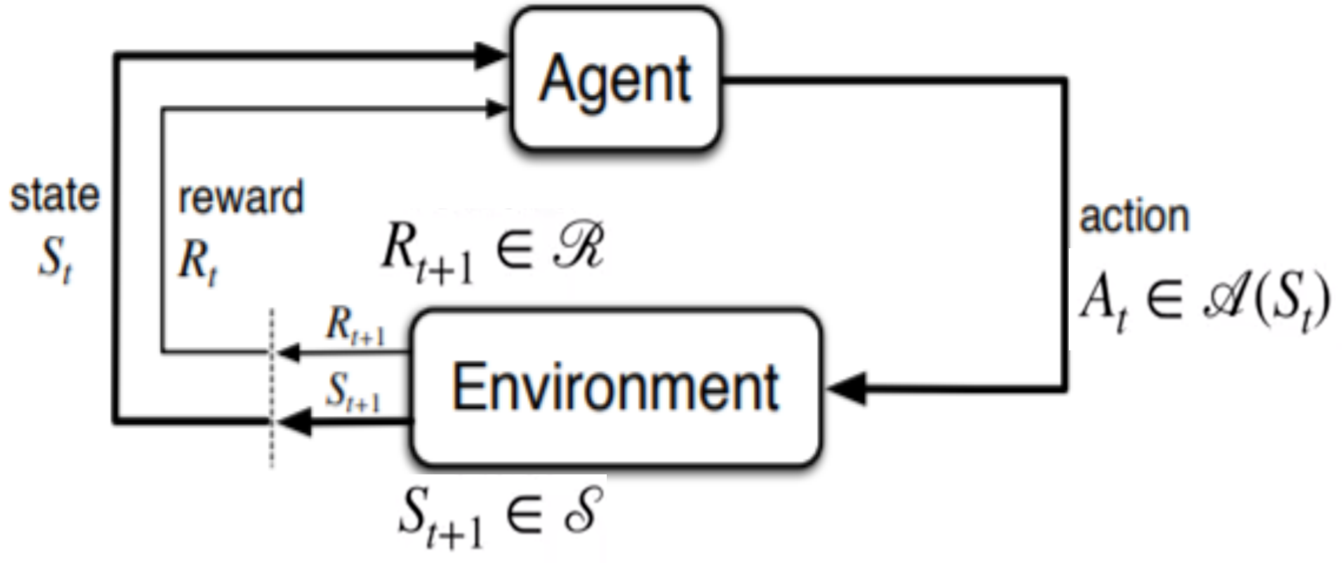
Markov Decision Processes (MDP)

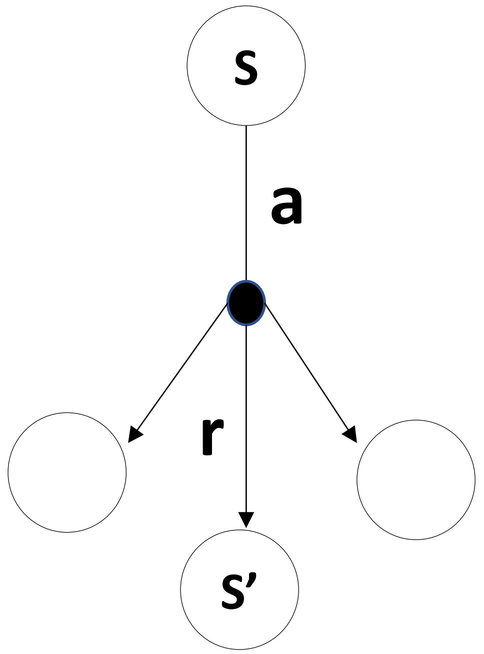

Agent
state S_t
reward R_t
$R_{t+1} \in \mathscr{R}$
action $A_t \in \mathscr{A}(S_t)$
R_{t+1}
S_{t+1}
Environment
$S_{t+1} \in \mathscr{S}$





The Dynamics of an MDP


Agent
Environment
state S_t
reward R_t
action $A_t \in \mathcal{A}(S_t)$
$R_{t+1} \in \mathcal{R}$
R_{t+1}
S_{t+1}
$S_{t+1} \in \mathcal{S}$



Probability distribution

p(s' ,r | s, a)

$$\sum_{s'} \sum_{r} p(s', r \mid s, a) = 1$$

Markov Property:

The present state contains all the information necessary to predict the future

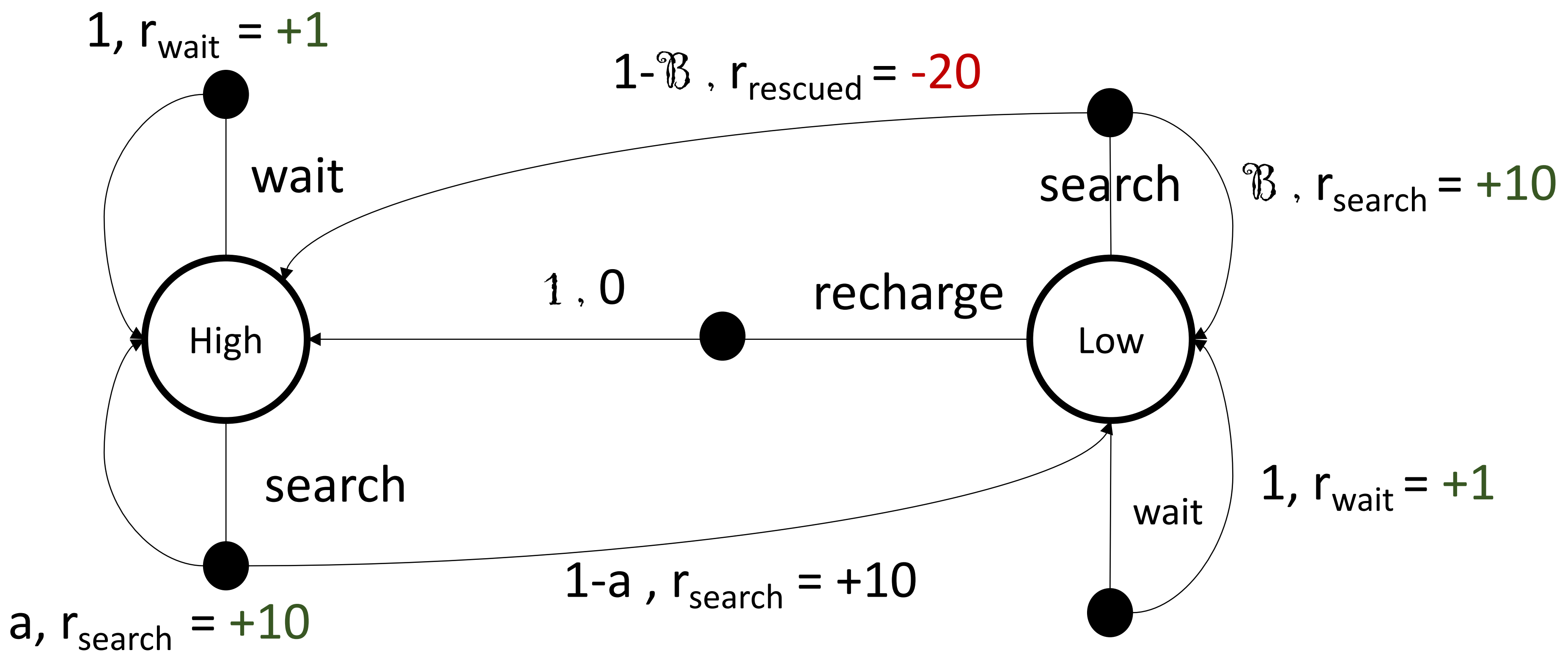
Examples of MDPs
Recycling Robot

S = {low, high} battery

A (low) = {search, wait, recharge}
A (high) = {search, wait}

Reward: when searching for Cans +10, when waiting +1, and -20 is we have to rescue the rebot

Examples of MDPs
Transition Dynamics



s	a	s’	p(s’ | s, a)	r(s, a, s’)
high	Search	high	$\propto$	r_{search}
high	Search	Low	$1-\propto$	r_{search}
Low	Search	high	$1-\beta$	-20
Low	Search	Low	β	r_{search}
high	wait	high	1	r_{wait}
high	wait	Low	0	-
Low	wait	high	0	-
Low	Wait	Low	1	r_{wait}
Low	Recharge	High	1	0
Low	Recharge	Low	0	-

MDP formalism is abstract and flexible and can be used to formalize a wide variety of sequential decision making problems

<u>States:</u>
1) can be low level pixel reading
2) high level object abstraction

<u>Actions:</u>
1) can be low level motor control
2) high level decision

<u>Time Steps</u> can be fixed intervals of time or successive stages of decision making

Robot to pick-and-place objects

State: latest reading of joint angles and velocities

Action: the amount of voltage applied to each motor

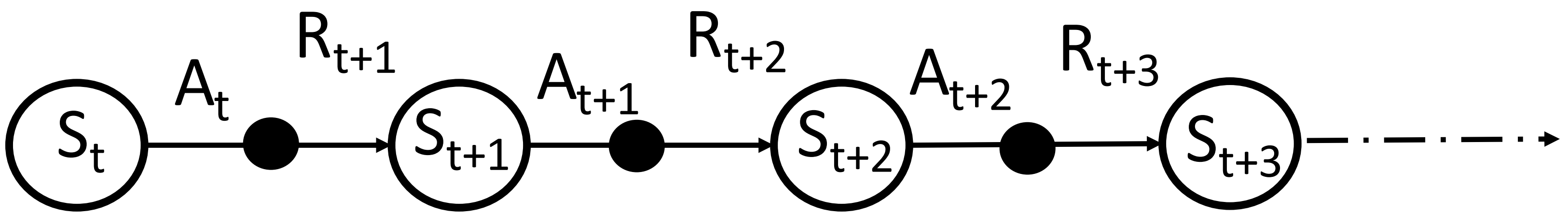
Reward: + 100 when an object successfully placed -1 for each unit of energy consumed.

Goal of an Agent: Formal definition

- In reinforcement learning, the agent's objective is to maximize future reward.
- Agents have long-term goals and what looked good in a short term, might now be the best in the long-term.
- The return is a random variable because the dynamics of the MDP can be stochastic.

Return $E[G_t] = E[R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T]$

Maximize the expected return



Episodic Tasks

- Interaction breaks naturally into **episodes**
- Each episode ends in a **terminal state**
- Episodes are **independent**

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T$$

Goals and Rewards

The maximization of the expected value of the cumulative sum of a received reward

$$E[G_t] = E[\underbrace{R_{t+1} + R_{t+2} + R_{t+3} + \ldots + R_T}_{}]$$

$$\overset{\uparrow}{R_T}\ \text{final step}$$

Continuing Tasks

- Interaction goes on continually
- No terminal state

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \ldots \infty$$

Discounting

How to make sure G_t is finite?

Discount the rewards in the future by γ where $0 \leq \gamma < 1$

Discounting

Discount the rewards in the future by γ where $0 \leq \gamma < 1$

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma k - 1 R_{t+k} \ldots \infty$$

Assuming R_{max} is the maximum rewards the agent can receive

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

$$= R_{max} \sum_{k=0}^{\infty} \gamma^k$$

$$= R_{max} \frac{1}{1 - \gamma}$$

Finite

Effect of γ on agent behavior

$\gamma = 0$

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma k - 1 R_{t+k.} \ldots \infty$$

$$= R_{t+1}$$

Short sided

$\gamma = 1$

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + R_{t+k.} \ldots$$

Long sided