CS 2750 Machine Learning
Lecture 22

Reinforcement learning

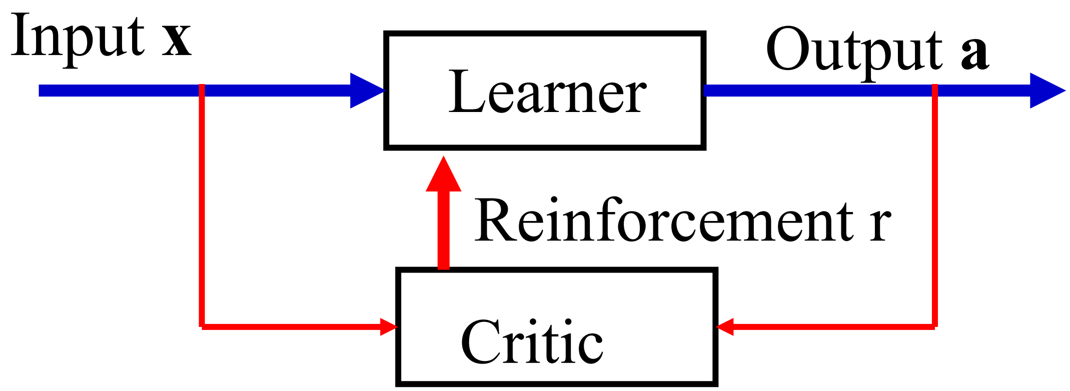
Milos Hauskrecht

milos@cs.pitt.edu

5329 Sennott Square

Reinforcement learning

Basics:

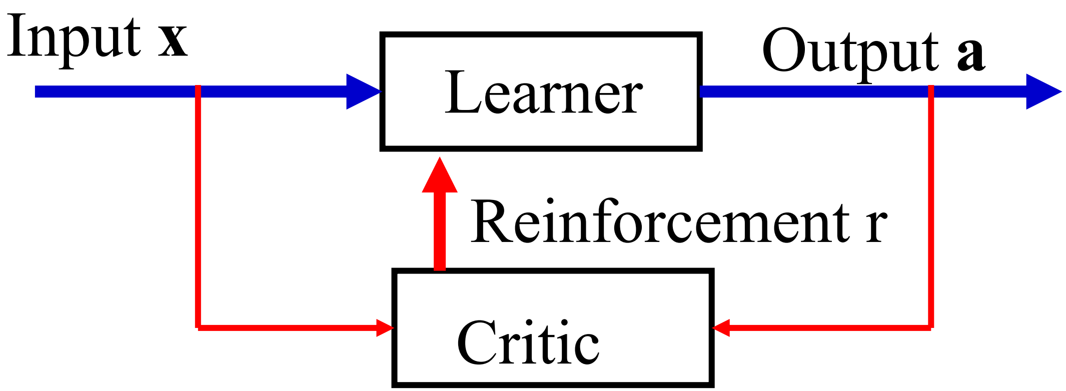


- **Learner interacts with the environment**
  - Receives input with information about the environment (e.g. from sensors)
  - Makes actions that (may) effect the environment
  - Receives a reinforcement signal that provides a feedback on how well it performed

Reinforcement learning

Objective: **Learn how to act in the environment in order to maximize the reinforcement signal**

- The selection of actions should depend on the input
- A policy $\pi : X \rightarrow A$ maps inputs to actions
- **Goal:** find the optimal policy $\pi : X \rightarrow A$ that gives the best expected reinforcements

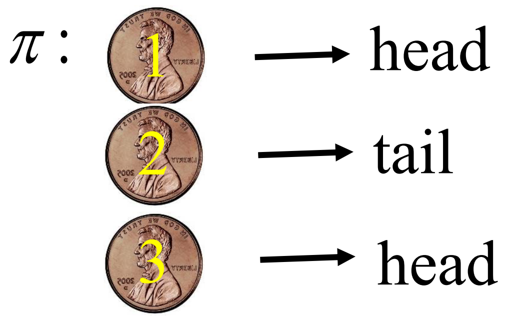


Example: learn how to play games (AlphaGo)

Gambling example

- **Game:** 3 biased coins
  - The coin to be tossed is selected randomly from the three coin options. The agent always sees which coin is going to be played next. The agent makes a bet on either a head or a tail with a wage of $1. If after the coin toss, the outcome agrees with the bet, the agent wins $1, otherwise it looses $1
- **RL model:**
  - **Input:** X – a coin chosen for the next toss,
  - **Action:** A – choice of head or tail the agent bets on,
  - **Reinforcements:** {1, -1}
- **A policy** $\pi : X \rightarrow A$

Example: $\pi :$ | **Coin1** → ***head***
 Coin2 → ***tail***
 Coin3 → ***head*** |



Trajectories

- **Environment + Agent's actions in time generate**

State, action, reward trajectories

Example: **Assume the agent applies the following policy**

$$\pi : \left| \begin{array}{l} \text{Coin1} \longrightarrow \textit{head} \\ \text{Coin2} \longrightarrow \textit{tail} \\ \text{Coin3} \quad\;\; \textit{head} \end{array} \right|$$

One possible SAR trajectory:

	Step0		Step1		Step2		Step k	
state	Coin2		Coin1		Coin2	..	Coin1	
action	Tail	⇨	Head	⇨	Tail	⇨	Head	.. ⇨
reward	-1		1		1		1	

Measuring the quality of the policy

- The quality of the policy can be measured in terms of the total rewards received by following the policy

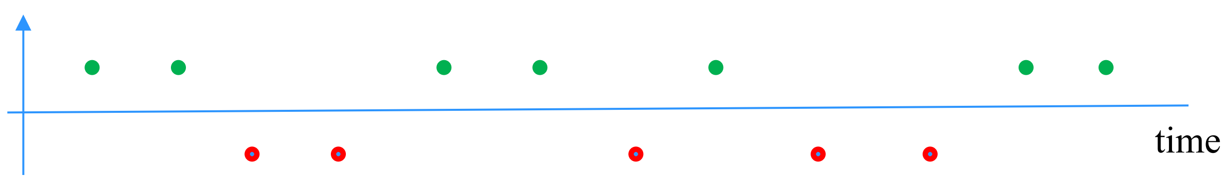
Example: Assume the agent applies the following policy

$$\pi : \left| \begin{array}{ll} \textbf{Coin1} \longrightarrow & \textbf{\textit{head}} \\ \textbf{Coin2} \longrightarrow & \textbf{\textit{tail}} \\ \textbf{Coin3} & \textbf{\textit{head}} \end{array} \right|$$

- The total reward for the policy and one SAR trajectory

  = **sum of rewards for the trajectory**

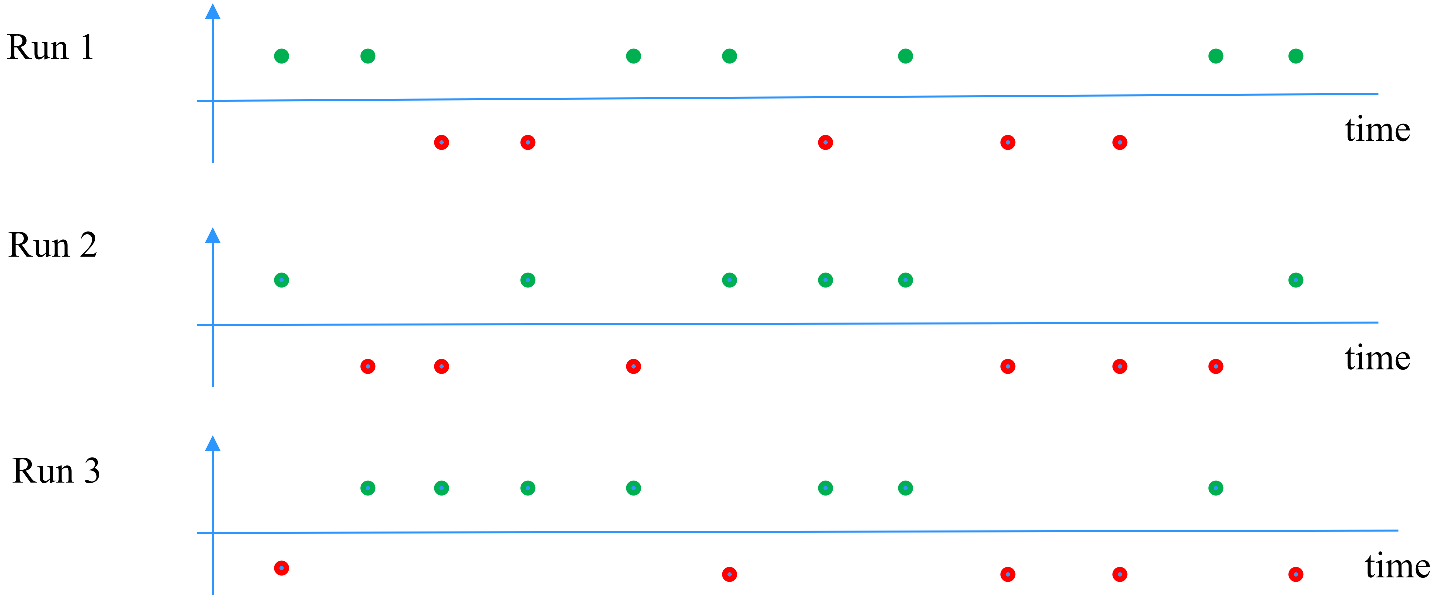
Run 1



- **But there can be multiple different trajectories the agent may face**

Expected rewards

- **Expected rewards for** $\pi : X \rightarrow A$



• • •

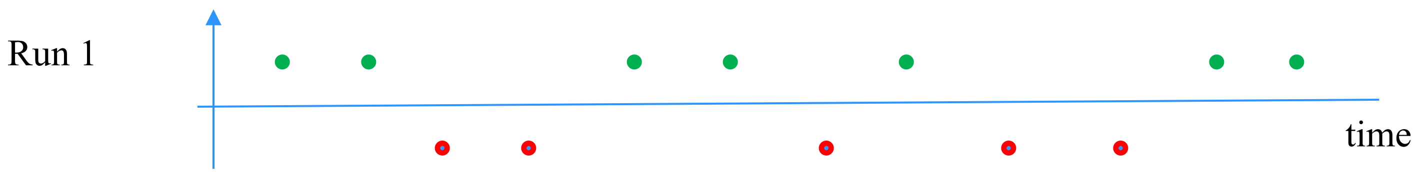
A good measure of the quality of policy $\pi : X \rightarrow A$

$$E(\sum_{t=0}^{T} r_t)$$ Expectation over many possible reward trajectories defined by $\pi : X \rightarrow A$

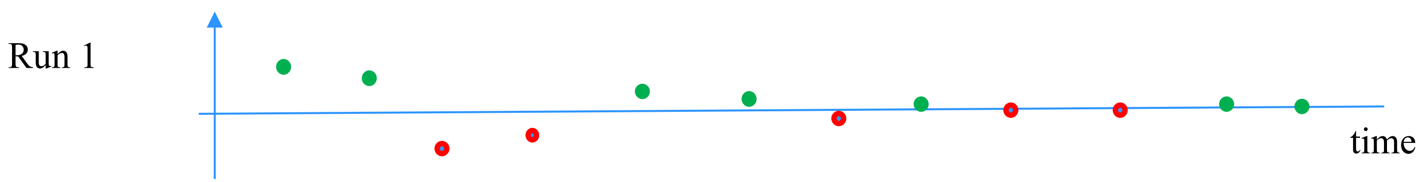
Expected discounted rewards

- **Expected discounting rewards for** $\pi : X \to A$
- **Discounting with** $0 \le \gamma < 1$ (future value of money)

No discounting:



Discounting



Another measure of the quality of policy $\pi : X \to A$

$$E\left(\sum_{t=0}^{T} \gamma^t r_t\right)$$

Expectation over many possible discounted reward trajectories for $\pi : X \to A$

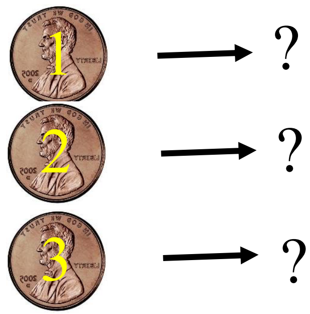
RL learning objective

Learning goal: **find the optimal policy**

$$\pi^*: X \rightarrow A$$

$$\pi^*: \left| \begin{array}{l} \textbf{Coin1} \longrightarrow ? \\ \textbf{Coin2} \longrightarrow ? \\ \textbf{Coin3} \longrightarrow ? \end{array} \right|$$

π^*:



That is, the policy that will maximize the future expected rewards

$$E(\sum_{t=0}^{T} \gamma^t r_t) \qquad 0 \le \gamma < 1$$

a discount factor = present value of money

RL learning: objective functions

- **Objective:**

  Find a policy $\pi^* : X \rightarrow A$

  That maximizes some combination of future reinforcements (rewards) received over time

- **Valuation models (quantify how good the mapping is):**
  - **Finite horizon models**

    $$E(\sum_{t=0}^{T} r_t)$$ Time horizon: $T > 0$

    $$E(\sum_{t=0}^{T} \gamma^t r_t)$$ Discount factor: $0 \leq \gamma < 1$

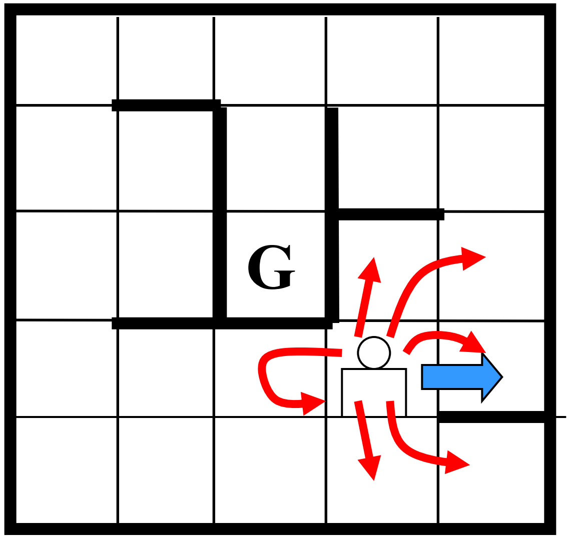
  - **Infinite horizon discounted model**

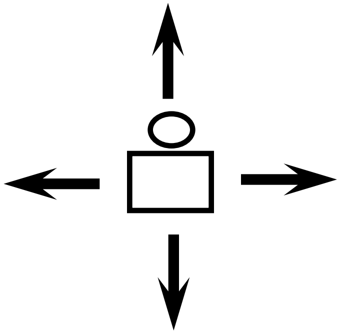
    $$E(\sum_{t=0}^{\infty} \gamma^t r_t)$$ Discount factor: $0 \leq \gamma < 1$

  - **Average reward** $\lim_{T \to \infty} \frac{1}{T} E(\sum_{t=0}^{T} r_t)$

Agent navigation example

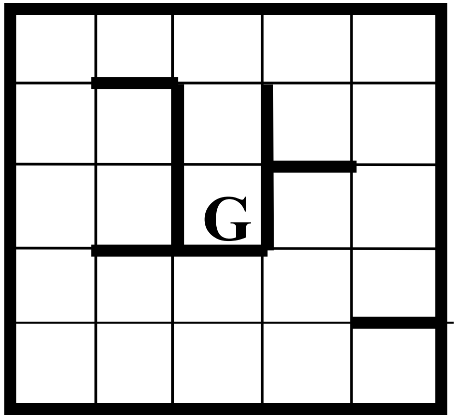
- **Agent navigation in the maze**:
  - 4 moves in compass directions
  - Effects of moves are stochastic – we may wind up in other than intended location with a non-zero probability
  - **Objective:** learn how to reach the goal state in the shortest expected time

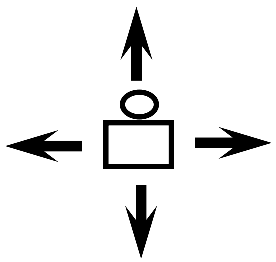




Agent navigation example

- **The RL model:**
  - **Input:** X – a position of an agent
  - **Actions:** A –the next move
  - **Reinforcements:** R
    - -1 for each move
    - +100 for reaching the goal
  - **A policy:** $\pi : X \rightarrow A$





$\pi :$
Position 1 ⟶ ***right***
Position 2 ⟶ ***right***
…
Position 25 ⟶ ***left***

- **Goal:** **find the policy maximizing future expected rewards**

$$E\left(\sum_{t=0}^{\infty} \gamma^t r_t\right) \qquad 0 \le \gamma < 1$$

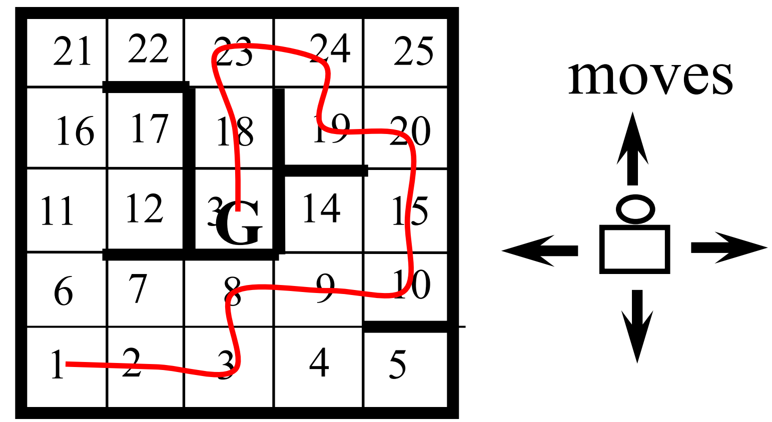
Agent navigation example

State, action reward trajectories

- **policy**

π :

Position 1 → *right*
Position 2 → *right*
…
Position 25 → *left*



	Step0	Step1	Step2	..	Step k	..
state	Pos1	Pos2	Pos3		Pos15	
action	Right →	Right →	Up →		Up	→
reward	-1	-1	-1		-1	

Effects of actions on the environment

Effect of actions on the environment

– More specifically on the next input **x** to be seen

Case 1. No effect The distribution over possible **x** is independent of past actions. The rewards received depend only on the current state **x** and the action *a* chosen.

- **Reinforcement learning with immediate rewards**
  - **3 coin example**

What coin we see next is not affected by our previous action, hence our action does not effect future rewards

	Step0	Step1	Step2	..	Step k	..
state	Coin2	Coin1	Coin2		Coin1	
action	Tail	Head	Tail		Head	
reward	-1	1	1		1	

Effects of actions on the environment

Effect of actions on the environment

– More specifically on the next input **x** to be seen

Case 2. Actions may effect the environment and next inputs **x**. The distribution of **x** can change due to past actions; the rewards related to the action can be seen with some delay.

- **Learning with delayed rewards**

  – **Agent navigation example**; a move action effects next position, and hence more distant future rewards

	Step0	**Step1**	**Step2**	..	**Step k**	..
state	Pos1	Pos2	Pos3		Pos15	
action	Right	Right	Up		Up	
reward	-1	-1	-1		-1	

RL with immediate rewards

- **Game:** 3 biased coins
  - The coin to be tossed is selected randomly from the three coin options. The agent always sees which coin is going to be played next. The agent makes a bet on either a head or a tail with a wage of $1. If after the coin toss, the outcome agrees with the bet, the agent wins $1, otherwise it looses $1
- **RL model:**
  - **Input:** X – a coin chosen for the next toss
  - **Action:** A – head or tail the agent bets on
  - **Reinforcements:** {1, -1} ($1 either won or lost)
- **Learning goal:** **find the optimal policy** $\pi^*: X \rightarrow A$

  maximizing the future expected profits over time

  $E(\sum_{t=0}^{\infty} \gamma^t r_t)$ $0 \leq \gamma < 1$ a discount factor

RL with immediate rewards

- **Expected reward** $E(\sum_{t=0}^{\infty} \gamma^t r_t) \qquad 0 \leq \gamma < 1$
- **Immediate reward case:**
  - Reward depends only on **x** and the action choice
  - The action does not affect the environment and hence future inputs (states) and future rewards:

General Trajectory	Step0	Step1	Step2		Step k	
state	x_0	x_1	x_2		x_k	
action	a_0	a_1	a_2	..	a_k	..
Reward	$r_{x0,a0}$	$r_{x1,a1}$	$r_{x2,a2}$		$r_{xk,ak}$	

$$E(\sum_{t=0}^{\infty} \gamma^t r_t) = E(r_{x0,a0}) + E(\gamma r_{x1,a1}) + E(\gamma^2 r_{x2,a2}) + \ldots E(\gamma^k r_{xk,ak}) + \ldots$$
$$= E(r_{x0,a0}) + \gamma E(r_{x1,a1}) + \gamma^2 E(r_{x2,a2}) + \ldots \gamma^k E(r_{xk,ak}) + \ldots$$

RL with immediate rewards

Immediate reward case:

- Reward for input **x** and the action choice *a* may vary
- **Expected one-step reward for the input x and action *a*:**

$$R(\mathbf{x}, a) = E(r_{\mathbf{x},a})$$

– For the coin bet problem it is:

$$R(\mathbf{x}, a_i) = \sum_j r(\omega_j \mid a_i, \mathbf{x}) P(\omega_j \mid \mathbf{x}, a_i)$$

ω_j : an outcome of the coin toss x

$r(\omega_j \mid a_i, \mathbf{x})$: reward for an outcome and the bet made on **x**

- **Expected one step reward for a policy** $\pi : X \rightarrow A$

$$R(\mathbf{x}, \pi(x)) = E(r_{\mathbf{x},\pi(x)})$$

RL with immediate rewards

- **Expected reward**

$$E(\sum_{t=0}^{\infty}\gamma^t r_t) = E(r_{x0,a0}) + E(\gamma r_{x1,a1}) + E(\gamma^2 r_{x2,a2}) + \ldots E(\gamma^k r_{xk,ak}) + \ldots$$
$$= E(r_{x0,a0}) + \gamma E(r_{x1,a1}) + \gamma^2 E(r_{x2,a2}) + \ldots \gamma^k E(r_{xk,ak}) + \ldots$$

- **Optimizing the expected reward** :

$$\max E(\sum_{t=0}^{\infty}\gamma^t r_t) = \max E(r_{x0,a0}) + \max E(\gamma r_{x1,a1}) + \ldots \max E(\gamma^k r_{xk,ak}) + \ldots$$
$$= \max E(r_{x0,a0}) + \gamma \max E(r_{x1,a1}) + \ldots \gamma^k \max E(r_{xk,ak}) + \ldots$$
$$= \max_{a_0} R(x_0, a_0) + \gamma \max_{a_1} R(x_1, a_1) + \ldots \gamma^k \max_{a_k} R(x_k, a_k) + \ldots$$

- **Optimal strategy:** $\pi^* : X \rightarrow A$

$$\pi^*(\mathbf{x}) = \arg\max_a R(\mathbf{x}, a)$$

RL with immediate rewards

The optimal choice assumes we know the expected reward

$R(\mathbf{x}, a)$

- **Then:** $\pi^*(\mathbf{x}) = \arg\max_a R(\mathbf{x}, a)$

Caveats

- **We do not know the expected reward** $R(\mathbf{x}, a)$
  - We need to estimate it using $\tilde{R}(\mathbf{x}, a)$ from the interactions
- **We cannot determine the optimal policy if the estimate of the expected reward is not good**
  - We need to try also actions that look suboptimal wrt the current estimates of $\tilde{R}(\mathbf{x}, a)$

RL with immediate rewards

- **Problem:** In the RL framework we do not know $R(\mathbf{x}, a)$
  - The expected reward for performing action a at input $\mathbf{x}$
- **Solution:**
  - For each input $\mathbf{x}$ try different actions a
  - Estimate $R(\mathbf{x}, a)$ using the average of observed rewards

$$\widetilde{R}(\mathbf{x}, a) = \frac{1}{N_{x,a}} \sum_{i=1}^{N_{x,a}} r_i^{x,a}$$

  - Action choice $\pi(\mathbf{x}) = \arg\max_a \widetilde{R}(\mathbf{x}, a)$
  - Accuracy of the estimate: statistics (Hoeffding's bound)

$$P\left(\left|\widetilde{R}(\mathbf{x}, a) - R(\mathbf{x}, a)\right| \geq \varepsilon\right) \leq \exp\left[-\frac{2\varepsilon^2 N_{x,a}}{(r_{\max} - r_{\min})^2}\right] \leq \delta$$

  - Number of samples: $N_{x,a} \geq \frac{(r_{\max} - r_{\min})^2}{2\varepsilon^2} \ln \frac{1}{\delta}$

RL with immediate rewards

- **On-line (stochastic approximation)**
  - An alternative way to estimate $R(\mathbf{x}, a)$
- **Idea:**
  - choose action a for input $\mathbf{x}$ and observe a reward $r^{x,a}$
  - Update an estimate in every step i

$$\widetilde{R}(\mathbf{x}, a)^{(i)} \leftarrow (1-\alpha(i))\widetilde{R}(\mathbf{x}, a)^{(i-1)} + \alpha(i)\, r_i^{x,a}$$

$\alpha(i)$ - a learning rate

- **Convergence property:** The approximation converges in the limit for an appropriate learning rate schedule.
- Assume: $\alpha(n(x,a))$ - is a learning rate for nth trial of (x,a) pair
- Then the converge is assured if:

1. $\sum_{i=1}^{\infty} \alpha(i) = \infty$
2. $\sum_{i=1}^{\infty} \alpha(i)^2 < \infty$

RL with immediate rewards

- At any step in time *i* during the experiment we have estimates of expected rewards for each (*coin, action*) pair:

$$\widetilde{R}(coin1, head)^{(i)}$$
$$\widetilde{R}(coin1, tail)^{(i)}$$
$$\widetilde{R}(coin2, head)^{(i)}$$
$$\widetilde{R}(coin2, tail)^{(i)}$$
$$\widetilde{R}(coin3, head)^{(i)}$$
$$\widetilde{R}(coin3, tail)^{(i)}$$

- Assume the next coin to play in step (i+1) is coin 2 and we pick head as our bet. Then we update $\widetilde{R}(coin2, head)^{(i+1)}$ using the observed reward and one of the update strategy above, and keep the reward estimates for the remaining (coin, action) pairs unchanged, e.g. $\widetilde{R}(coin2, tail)^{(i+1)} = \widetilde{R}(coin2, tail)^{(i)}$

Exploration vs. Exploitation in RL

The (learner) actively interacts with the environment via actions:

- At the beginning the learner does not know anything about the environment
- It gradually gains the experience and learns how to react to the environment

Dilemma (exploration-exploitation):

- After some number of steps, should I select the best current choice (**exploitation**) or try to learn more about the environment (**exploration**)?
- **Exploitation** may involve the selection of a sub-optimal action and prevent the learning of the optimal choice
- **Exploration** may spend to much time on trying bad currently suboptimal actions

Exploration vs. Exploitation

- In the RL framework
  - the (learner) actively interacts with the environment and **choses the action** to play for the current input $\mathbf{x}$
  - Also at any point in time it has an estimate of $\tilde{R}(\mathbf{x}, a)$ for any (*input,action*) pair
- **Dilemma for choosing the action to play for x:**
  - Should the learner choose the current best choice of action (exploitation)

$$\hat{\pi}(\mathbf{x}) = \arg\max_{a \in A} \tilde{R}(\mathbf{x}, a)$$

  - Or choose some other action a which may help to improve its $\tilde{R}(\mathbf{x}, a)$ estimate (exploration)

  This dilemma is called **exploration/exploitation dilemma**

- **Different exploration/exploitation strategies exist**

Exploration vs. Exploitation

- **Epsilon greedy exploration:**
  - Uses exploration parameter $0 \le \varepsilon \le 1$
  - Choose the "current" best choice with probability $1-\varepsilon$

$$\hat{\pi}(\mathbf{x}) = \arg\max_{a \in A} \widetilde{R}(\mathbf{x}, a)$$

  - All other choices are selected with a uniform probability $\dfrac{\varepsilon}{|A|-1}$

Advantages:

- Simple, easy to implement

Disadvantages:

- Exploration more appropriate at the beginning when we do not have good estimates of $\widetilde{R}(\mathbf{x}, a)$
- Exploitation more appropriate later when we have good estimates

Exploration vs. Exploitation

- **Boltzman exploration**
  - The action is chosen randomly but proportionally to its current expected reward estimate
  - Can be tuned with a temperature parameter T to promote exploration or exploitation
- Probability of choosing action a

$$p(a \mid \mathbf{x}) = \frac{\exp\left[\widetilde{R}(x,a)/T\right]}{\sum_{a' \in A} \exp\left[\widetilde{R}(x,a')/T\right]}$$

- **Effect of T:**
  - For high values of T, p(a | x) is uniformly distributed for all actions
  - For low values of T, p(a | x) of the action with the highest value of $\widetilde{R}(\mathbf{x},a)$ is approaching 1