Izmir Institute of Technology

CENG 461 – Artificial Intelligence

Reinforcement Learning

2015-16 Fall Term

Review: MDPs

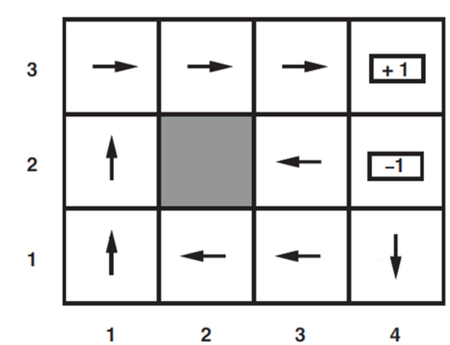
- Fully observable $s_1, \ldots, s_N, a_1, \ldots, a_K$
- Stochastic $P(s'|s, a)$
- Reward $R(s)$
- Objective:

$$\max \; E[\textstyle\sum_{t=0}^{\infty} \gamma^t R(s_t)]$$

- Value iteration:

$$U_{i+1}(s) = R(s) + \gamma \max_a \textstyle\sum_{s'} P(s'|s,a) U_i\,(s')$$

- Policy: Converged value iteration provides us a solution called a policy $\pi(s)$.



Reinforcement Learning

- What if we do not know where the rewards (R) are?
- What if we do not know the transition model (P)?
  - Agents can learn R and P, or the substitutes of these by interacting with the world.
- Agent Types:

	know	learn	use
Utility-based agent	P	$R \rightarrow U$	U
Q-learning agent		Q(s,a)*	Q
Reflex agent		$\pi(s)$	π

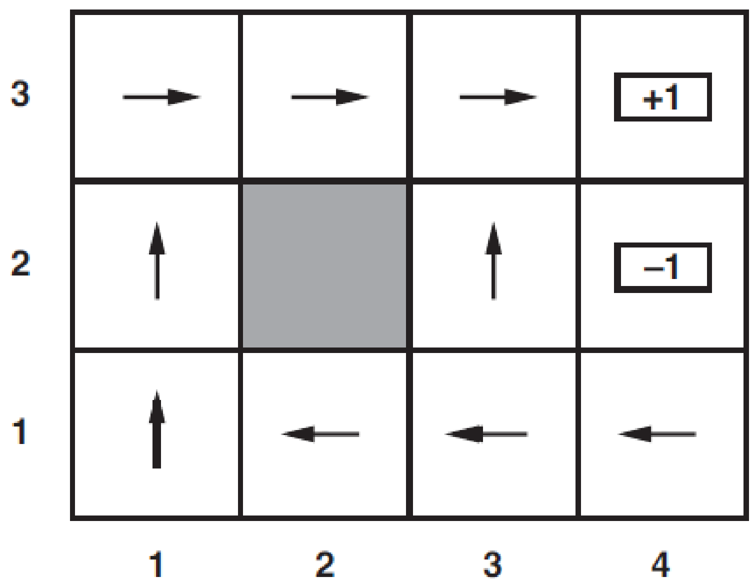
*Q(s,a) is utility over state-action pairs rather than utility over states

Active vs. Passive R.L.

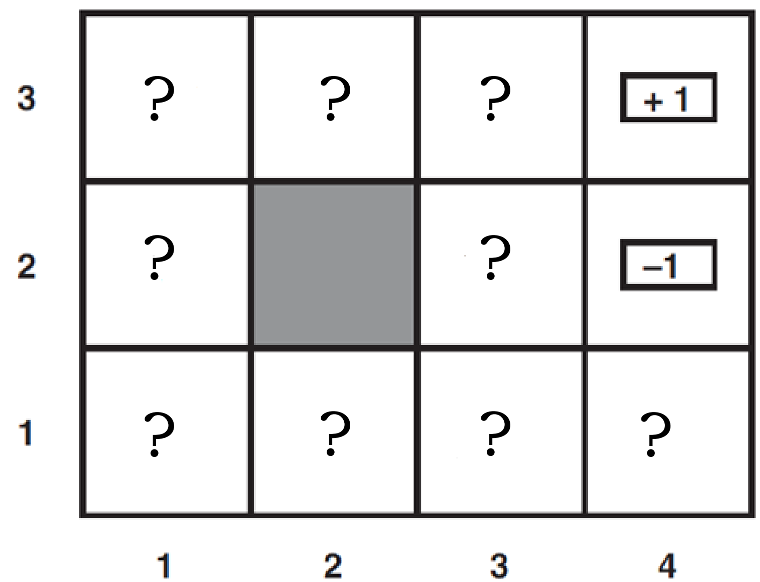
- Passive agent has a fixed policy and learns about rewards (R) and maybe also transitions (P) using that policy.
- Active agent changes the policy as it learns the environment. It can also choose policies to learn more.

Passive Reinforcement Learning

- Execute fixed policy and learn U from the outcomes



Example Fixed Policy



Find out Utilities of the Fixed Policy

Passive Reinforcement Learning

- Execute trials using the fixed policy (e.g. R(s)=-0.04)

$$(1,1)_{-.04} \rightsquigarrow (1,2)_{-.04} \rightsquigarrow (1,3)_{-.04} \rightsquigarrow (1,2)_{-.04} \rightsquigarrow (1,3)_{-.04} \rightsquigarrow (2,3)_{-.04} \rightsquigarrow (3,3)_{-.04} \rightsquigarrow (4,3)_{+1}$$
$$(1,1)_{-.04} \rightsquigarrow (1,2)_{-.04} \rightsquigarrow (1,3)_{-.04} \rightsquigarrow (2,3)_{-.04} \rightsquigarrow (3,3)_{-.04} \rightsquigarrow (3,2)_{-.04} \rightsquigarrow (3,3)_{-.04} \rightsquigarrow (4,3)_{+1}$$
$$(1,1)_{-.04} \rightsquigarrow (2,1)_{-.04} \rightsquigarrow (3,1)_{-.04} \rightsquigarrow (3,2)_{-.04} \rightsquigarrow (4,2)_{-1} \; .$$

- Temporal Difference Learning
  - Start with a blank table of utilities and apply the policy:

If the state s' is new then U(s')=r {assign the reward}

If s is not null, then do

increment N(s)

$$U(s) \leftarrow U(s) + \alpha(N(s)) \cdot (r + \gamma U(s') - U(s))$$

N(s) : the number of times we visit a state

s' : current state, s : previous state (state before the action)

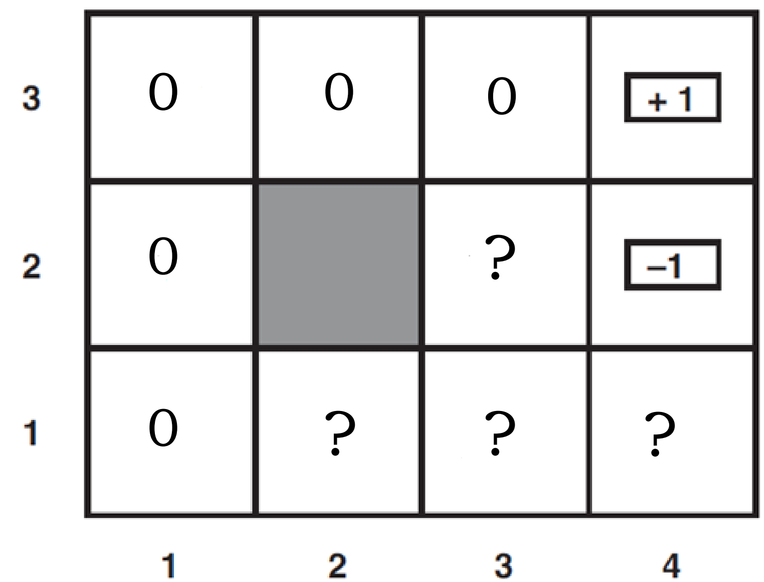
α : learning rate

Passive Reinforcement Learning

- Apply TDL with
  r=0, γ=1 and $\alpha(N)=1/(N+1)$
  $U(s) \leftarrow U(s)+\alpha(N(s))\cdot(r+\gamma U(s')-U(s))$

- Lets assume our policy keeps going through (1,1)-(1,2)-(1,3)-(2,3)-(3,3)-(3,4)

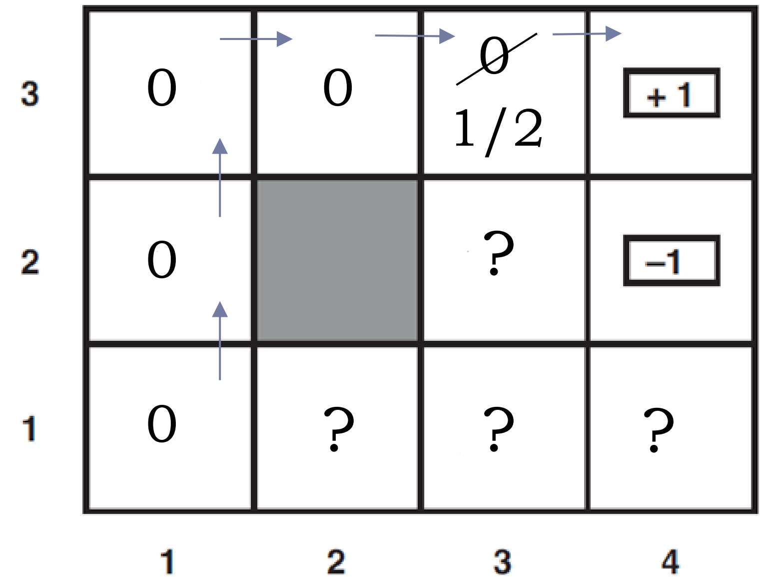
- At first trial, all states are new, U(s) are assigned as zero.



Passive Reinforcement Learning

- At second trial $r=0, \gamma=1$ and $\alpha(N)=1/(N+1)$

$$U(s) \leftarrow U(s)+\alpha(N(s))\cdot(r+\gamma U(s')-U(s))$$



$U(1,1) \leftarrow 0 + 1/2 \cdot (0 + 0 - 0)=0$

$U(1,2) \leftarrow 0 + 1/2 \cdot (0 + 0 - 0)=0$

$U(1,3) \leftarrow 0 + 1/2 \cdot (0 + 0 - 0)=0$

$U(2,3) \leftarrow 0 + 1/2 \cdot (0 + 0 - 0)=0$

$U(3,3) \leftarrow 0 + 1/2 \cdot (0 + 1 - 0)=1/2.$

Passive Reinforcement Learning

- At third trial $r=0, \gamma=1$ and $\alpha(N)=1/(N+1)$

  $U(s) \leftarrow U(s)+\alpha(N(s))\cdot(r+\gamma U(s')-U(s))$

$U(1,1) = 0$

$U(1,2) = 0$

$U(1,3) = 0$

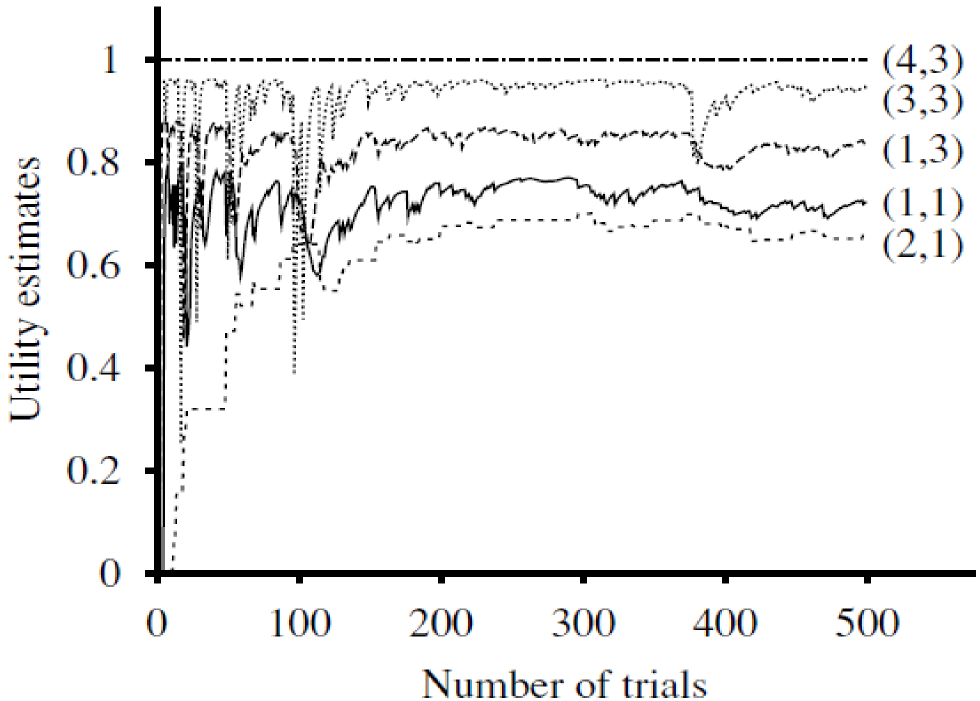
$U(2,3) \leftarrow 0+1/3\cdot(0+1/2-0)=1/6$

$U(3,3) \leftarrow 1/2+1/3\cdot(0+1-1/2)=2/3$

3	0	~~0~~ 1/6	~~1/2~~ 2/3	+1
2	0		?	−1
1	0	?	?	?
	1	2	3	4

Passive Reinforcement Learning

- Convergence passive TDL:

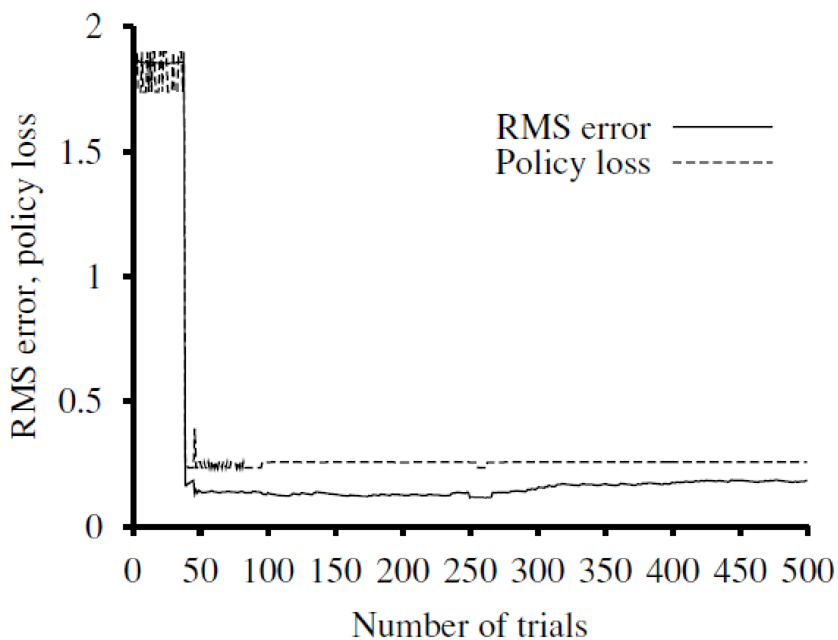


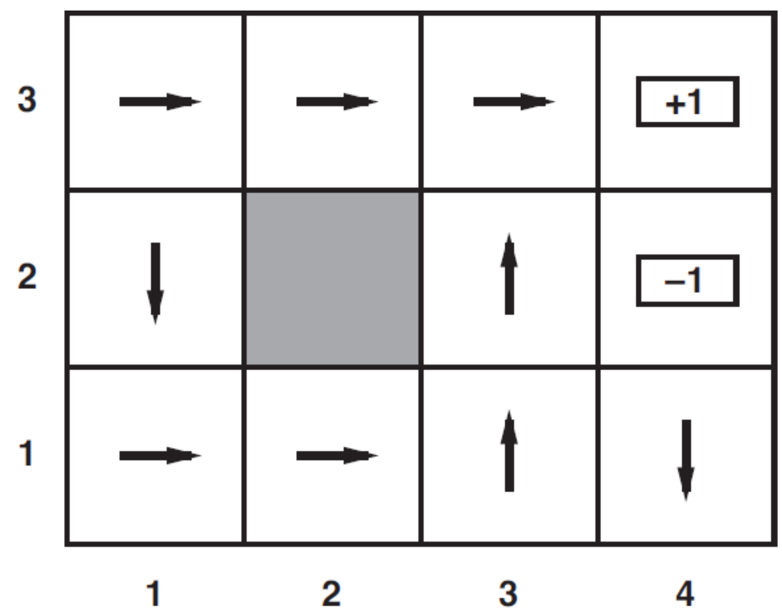
Active Reinforcement Learning

- Passive R.L. has some problems due to fixed π
  - Long convergence
  - Some states may not be discovered
- Active agents change the policy to make use of the gathered information.
- Greedy Active Reinforcement Learning
  - Use the passive temporal-difference learning to gather information.
  - Discard old policy and use learned policy for future trials.

Active Reinforcement Learning

- Performance of greedy active R.L. agent





Policy loss is the difference between the policy the agent learned and the optimal policy

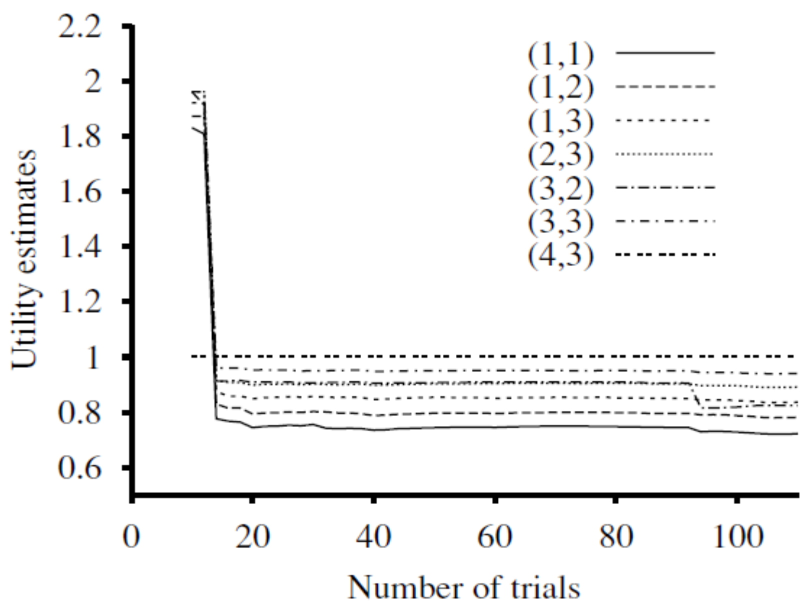
Suboptimal policy that the greedy agent converged

Active Reinforcement Learning

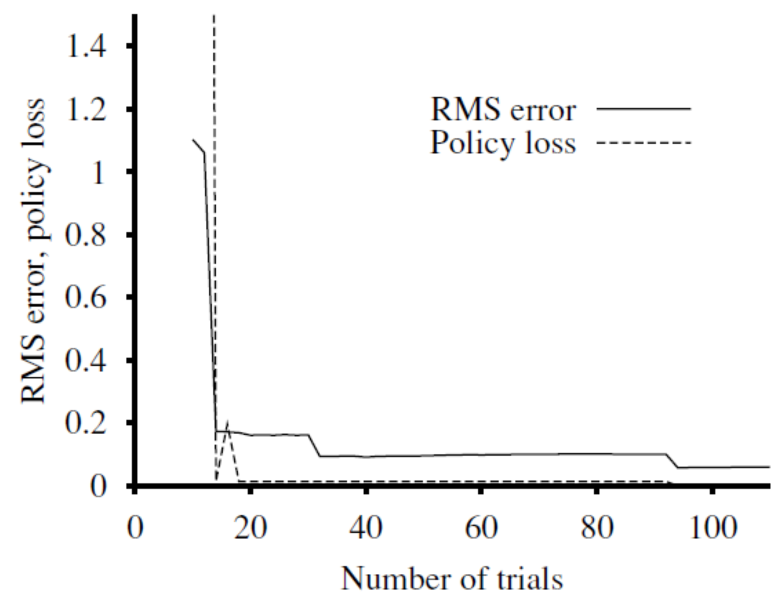
- ***Exploratory agents*** try random policies to discover more about the world
- Randomizing the current policy might get us off the suboptimal local minima, but it is slow.
- Exploration v.s. Exploitation tradeoff

Active Reinforcement Learning

- Performance of exploratory agent
  - Utility estimates converged after 20 trials
  - It converged at a good policy (low utility errors and ~0 policy loss)



Convergence rate of utilities for varying s



Utility error, policy loss

Q-Learning

- Once you learn the utilities you can decide on the optimal policy by

$$\pi^*(s) = \max_a \sum_{s'} P(s'|s,a)U(s')$$

- However, the equation above is solvable only if we know the transition probabilities.
- Instead we can learn $\mathrm{Q(s, a)}$ values that pick the best action for each state directly:
- The update rule for Temp. Diff. Learning becomes

$$\mathrm{Q(s,a)} \leftarrow \mathrm{Q(s,a)} + \alpha(\mathrm{R(s)} + \gamma \mathrm{Q(s',a')} - \mathrm{Q(s,a)})$$

Q-Learning

$$Q(s,a) \leftarrow Q(s,a) + \alpha(R(s) + \gamma Q(s',a') - Q(s,a))$$