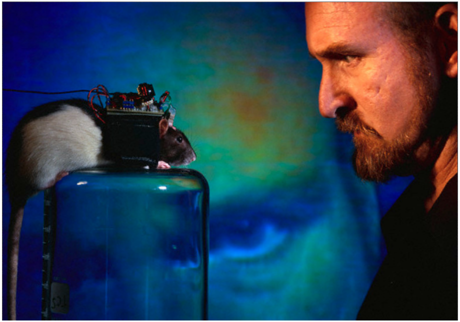
CS325 Artificial Intelligence
Ch. 21 – Reinforcement Learning

Cengiz Günay, Emory Univ.

Spring 2013

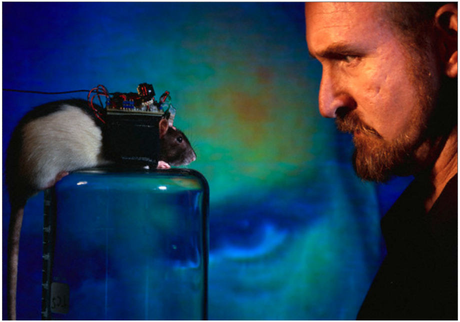
Rats!



Fundooprofessor

- Rat put in a cage with lever.
- Each lever press sends a signal to rat's brain, to the **reward center**.

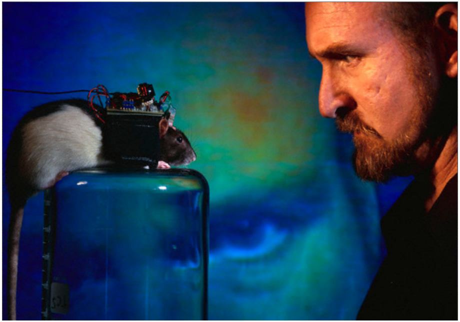
Rats!



Fundooprofessor

- Rat put in a cage with lever.
- Each lever press sends a signal to rat's brain, to the **reward center**.
- Rat presses lever continously until . . .

Rats!

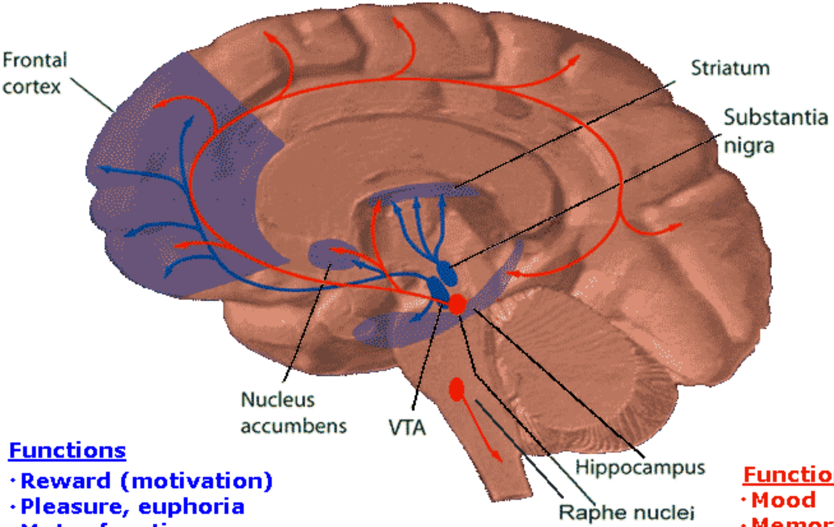


Fundooprofessor

- Rat put in a cage with lever.
- Each lever press sends a signal to rat's brain, to the **reward center**.
- Rat presses lever continously until . . . it dies because it stops eating and drinking.

Dopamine Pathways

Serotonin Pathways



Functions
- Reward (motivation)
- Pleasure, euphoria
- Motor function (fine tuning)
- Compulsion
- Perseveration

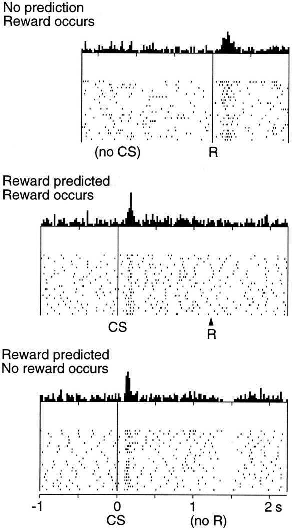
Functions
- Mood
- Memory processing
- Sleep
- Cognition

Wikipedia.org

Dopamine Neurons Respond to Novelty

sciencemuseum.org.uk

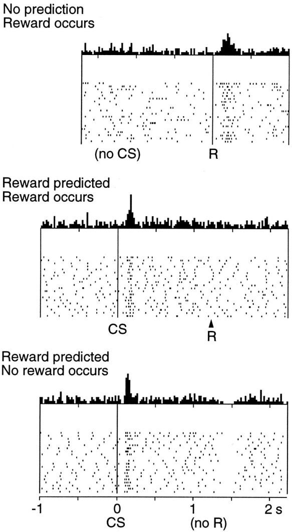
Schultz et al. (1997)



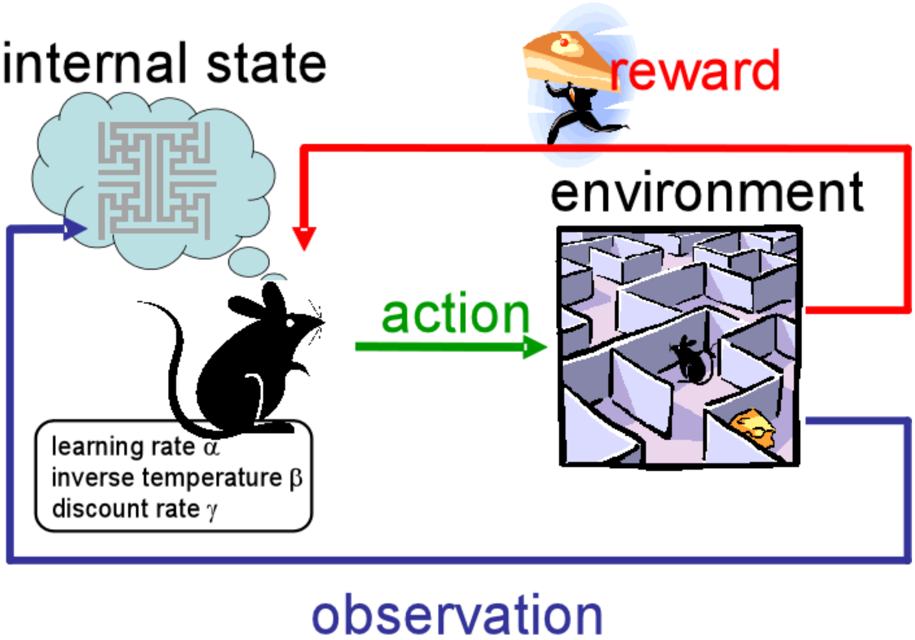
Dopamine Neurons Respond to Novelty

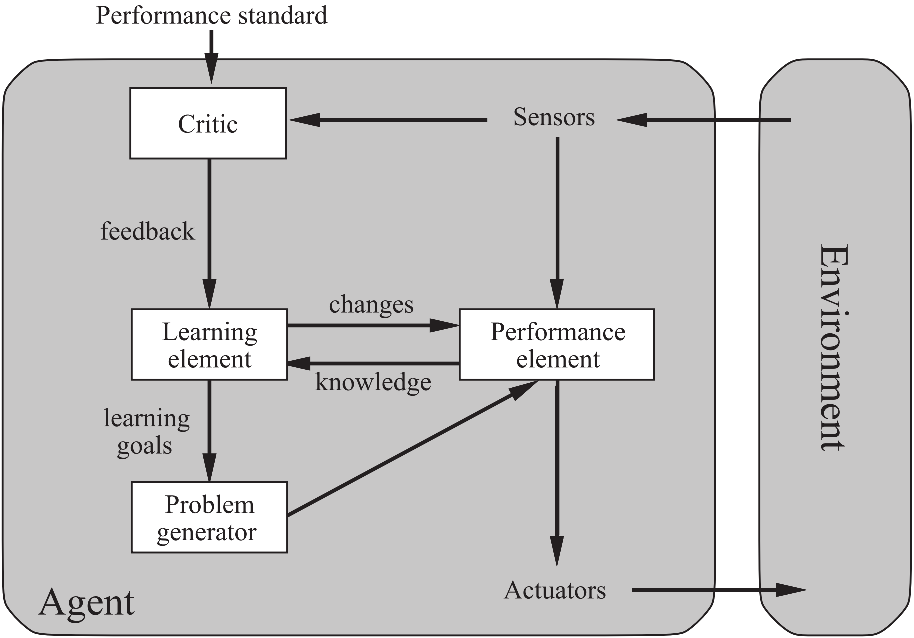
sciencemuseum.org.uk

Schultz et al. (1997)



It turns out:
Novelty detection = **Temporal Difference** rule in Reinforcement Learning
(Sutton and Barto, 1981)


internal state
reward
environment
action
learning rate α
inverse temperature β
discount rate γ
observation


Performance standard
Critic
Sensors
feedback
changes
Learning element
Performance element
knowledge
learning goals
Problem generator
Actuators
Agent
Environment

Entry/Exit Surveys

Exit survey: Planning Under Uncertainty

- Why can't we use a regular MDP for partially-observable situations?
- Give an example where you think MDPs would help you solve a problem in your daily life.

Entry survey: Reinforcement Learning (0.25 points of final grade)

- In a partially-observable scenario, can reinforcement be used to learn MDP rewards?
- How can we improve MDP by using the plan-execute cycle?

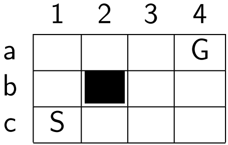
Blindfolded MDPs: Enter Reinforcement Learning

	1	2	3	4
a				G
b				
c	S			

What if the agent does not know anything about:

- where walls are
- where goals/penalties are

Blindfolded MDPs: Enter Reinforcement Learning



What if the agent does not know anything about:

- where walls are
- where goals/penalties are

Can we use the plan-execute cycle?

Blindfolded MDPs: Enter Reinforcement Learning

	1	2	3	4
a				G
b				
c	S			

What if the agent does not know anything about:

- where walls are
- where goals/penalties are

Can we use the plan-execute cycle?

- Explore first
- Update world state based on reward/reinforcement

⇒ **Reinforcement Learning** (see Scholarpedia article)

Where Does Reinforcement Learning Fit?

Machine learning so far:

Where Does Reinforcement Learning Fit?

Machine learning so far:

Unsupervised learning: find regularities in input data, x

Where Does Reinforcement Learning Fit?

Machine learning so far:

Unsupervised learning: find regularities in input data, x

Supervised learning: find mapping between input and output, $f(x) \to y$

Where Does Reinforcement Learning Fit?

Machine learning so far:

Unsupervised learning: find regularities in input data, x

Supervised learning: find mapping between input and output, $f(x) \to y$

Reinforcement learning: find mapping between states and actions, $s \to a$

Where Does Reinforcement Learning Fit?

Machine learning so far:

Unsupervised learning: find regularities in input data, x

Supervised learning: find mapping between input and output, $f(x) \rightarrow y$

Reinforcement learning: find mapping between states and actions, $s \rightarrow a$
(by finding optimal policy, $\pi(s) \rightarrow a$)

Where Does Reinforcement Learning Fit?

Machine learning so far:

Unsupervised learning: find regularities in input data, x

Supervised learning: find mapping between input and output, $f(x) \rightarrow y$

Reinforcement learning: find mapping between states and actions, $s \rightarrow a$
(by finding optimal policy, $\pi(s) \rightarrow a$)

Which is it?

S	U	R	
			Speech recognition: connect sounds to transcripts
			Star data: find groupings from spectral emissions
			Rat presses lever: gets reward based on certain conditions
			Elevator controller: multiple elevators, minimize wait time

Where Does Reinforcement Learning Fit?

Machine learning so far:

Unsupervised learning: find regularities in input data, x

Supervised learning: find mapping between input and output, $f(x) \to y$

Reinforcement learning: find mapping between states and actions, $s \to a$
(by finding optimal policy, $\pi(s) \to a$)

Which is it?

S	U	R	
X			Speech recognition: connect sounds to transcripts
	X		Star data: find groupings from spectral emissions
		X	Rat presses lever: gets reward based on certain conditions
		X	Elevator controller: multiple elevators, minimize wait time

But, Wasn't That What Markov Decision Processes Were?

- Find optimal policy to maximize reward:

$$\pi(s) = \arg\max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t R(s, \pi(s), s')\right],$$

with reward at state: $R(s)$, or from action, $R(s, a, s')$.

But, Wasn't That What Markov Decision Processes Were?

- Find optimal policy to maximize reward:

$$\pi(s) = \arg\max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t R(s, \pi(s), s')\right],$$

with reward at state: $R(s)$, or from action, $R(s, a, s')$.

- By estimating utility values:

$$V(s) \leftarrow \left[\arg\max_{a} \gamma \sum_{s'} P(s'|s, a) V(s')\right] + R(s),$$

with transition probabilities: $P(s'|s, a)$

But, Wasn't That What Markov Decision Processes Were?

- Find optimal policy to maximize reward:

$$\pi(s) = \arg\max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t R(s, \pi(s), s')\right] ,$$

with reward at state: $R(s)$, or from action, $R(s, a, s')$.

- By estimating utility values:

$$V(s) \leftarrow \left[\arg\max_{a} \gamma \sum_{s'} P(s'|s, a) V(s')\right] + R(s) ,$$

with transition probabilities: $P(s'|s, a)$

- **Assumes we know $R(s)$ and $P(s'|s, a)$**

Blindfolded Agent Must Learn From Rewards

Don't know $R(s)$ or $P(s'|s, a)$. What to do?

Blindfolded Agent Must Learn From Rewards

Don't know $R(s)$ or $P(s'|s,a)$. What to do?

- Use Reinforcement Learning (RL) to **explore** and find rewards

Blindfolded Agent Must Learn From Rewards

Don't know $R(s)$ or $P(s'|s, a)$. What to do?

- Use Reinforcement Learning (RL) to **explore** and find rewards

Agent types:

	knows	learns	uses
Utility agent	P	$R \to U$	U
Q-learning (RL)		$Q(s, a)$	Q
Reflex		$\pi(s)$	

Video: Backgammon and Choppers

How Much to Learn?

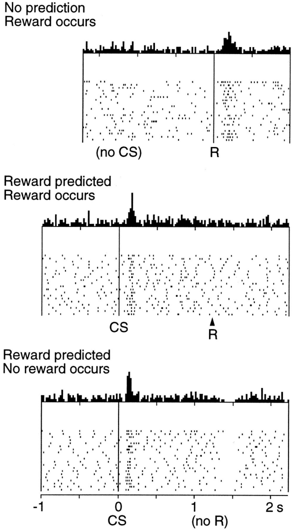
1. Passive RL: Simple Case
   - Keep policy $\pi(s)$ fixed, learn others
   - Always do same actions, and learn utilities
   - Examples:
     - public transit commute
     - learning a difficult game

How Much to Learn?

1. Passive RL: Simple Case
    - Keep policy $\pi(s)$ fixed, learn others
    - Always do same actions, and learn utilities
    - Examples:
        - public transit commute
        - learning a difficult game
2. Active RL
    - Learn policy at the same time
    - Help explore better by changing policy
    - Example: drive own car

RL in Practise: Temporal Difference (TD) Rule

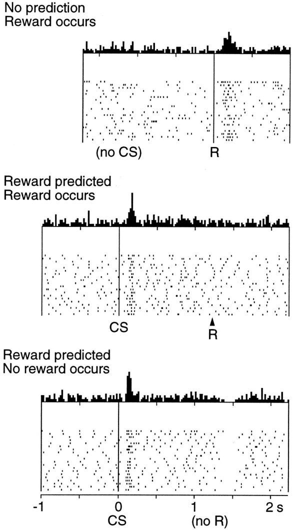
Animals use derivative:



Remember value iteration:

$$V(s) \leftarrow \left[\arg\max_a \gamma \sum_{s'} P(s'|s,a)V(s')\right] + R(s)\,.$$

Animals use derivative:



Remember value iteration:

$$V(s) \leftarrow \left[\arg\max_a \gamma \sum_{s'} P(s'|s,a) V(s') \right] + R(s) \,.$$

TD rule:

Use derivative when going $s \to s'$:

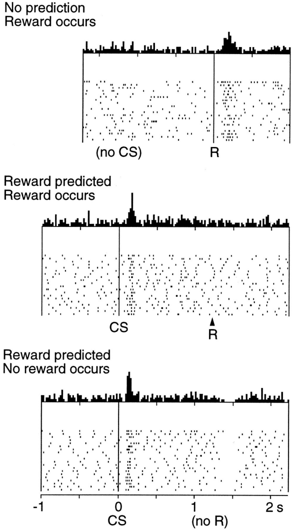
$$V(s) \leftarrow V(s) + \alpha \left(R(s) + \gamma V(s') - V(s) \right)$$

where:

- α is the **learning rate**, and
- γ is the discount factor.

RL in Practise: Temporal Difference (TD) Rule

Animals use derivative:



Remember value iteration:

$$V(s) \leftarrow \left[\arg\max_a \gamma \sum_{s'} P(s'|s,a) V(s') \right] + R(s) \, .$$

TD rule:

Use derivative when going $s \rightarrow s'$:

$$V(s) \leftarrow V(s) + \alpha \left(R(s) + \gamma V(s') - V(s) \right)$$

where:

- α is the **learning rate**, and
- γ is the discount factor.

It's even simpler than before!

Passive RL: Simple Case

	1	2	3	4
a				$+1$
b		■		-1
c	S			

- Keep same policy
- That is, follow same path and update values, $V(s)$

Passive RL: Simple Case

	1	2	3	4
a				$+1$
b				-1
c	S			

- Keep same policy
- That is, follow same path and update values, $V(s)$

To mimic increasing confidence, reduce learning rate with number of visits, $N(s)$:

$$\alpha = \frac{1}{N(s) + 1}$$

like in **simulated annealing**.

Passive RL: Simple Case

	1	2	3	4
a				$+1$
b				-1
c	S			

- Keep same policy
- That is, follow same path and update values, $V(s)$

To mimic increasing confidence, reduce learning rate with number of visits, $N(s)$:

$$\alpha = \frac{1}{N(s)+1}$$

like in **simulated annealing**.
TD rule:

$$V(s) \leftarrow V(s) + \frac{1}{N(s)+1}\left(R(s) + \gamma V(s') - V(s)\right)$$

Passive RL: Simple Case (2)

	1	2	3	4
a	$\rightarrow$	$\rightarrow$	$\rightarrow$	$+1$
b	$\uparrow$	■		-1
c	S			

$$
\begin{aligned}
V(s) &\leftarrow V(s) + \Delta \\
\Delta &= \frac{1}{N(s)+1}\left(R(s) + \gamma V(s') - V(s)\right)
\end{aligned}
$$

For simplicity, $\gamma = 1$.

Passive RL: Simple Case (2)

	1	2	3	4
a	$\rightarrow$	$\rightarrow$	$\rightarrow$	$+1$
b	$\uparrow$	■		-1
c	S			

$$V(s) \leftarrow V(s) + \Delta$$

$$\Delta = \frac{1}{N(s)+1}\left(R(s) + \gamma V(s') - V(s)\right)$$

For simplicity, $\gamma = 1$.

	N	$V(s)$	Δ
a3 $\rightarrow$ a4	1	0	1/2

Passive RL: Simple Case (2)

	1	2	3	4
a	$\rightarrow$	$\rightarrow$	$\rightarrow$	$+1$
b	$\uparrow$	■		-1
c	S			

$$
\begin{aligned}
V(s) &\leftarrow V(s) + \Delta \\
\Delta &= \frac{1}{N(s)+1}\left(R(s) + \gamma V(s') - V(s)\right)
\end{aligned}
$$

For simplicity, $\gamma = 1$.

	N	$V(s)$	Δ
a3 $\rightarrow$ a4	1	0	1/2
a2 $\rightarrow$ a3	2	0	1/6

Passive RL: Simple Case (2)

	1	2	3	4
a	$\rightarrow$	$\rightarrow$	$\rightarrow$	$+1$
b	$\uparrow$			-1
c	S			

$$V(s) \leftarrow V(s) + \Delta$$

$$\Delta = \frac{1}{N(s)+1}\left(R(s) + \gamma V(s') - V(s)\right)$$

For simplicity, $\gamma = 1$.

	N	$V(s)$	Δ
$\mathrm{a3} \rightarrow \mathrm{a4}$	1	0	1/2
$\mathrm{a2} \rightarrow \mathrm{a3}$	2	0	1/6
$\mathrm{a3} \rightarrow \mathrm{a4}$	2	1/2	1/6

Passive RL: Simple Case (2)

	1	2	3	4
a	$\rightarrow$	$\rightarrow$	$\rightarrow$	$+1$
b	$\uparrow$			-1
c	S			

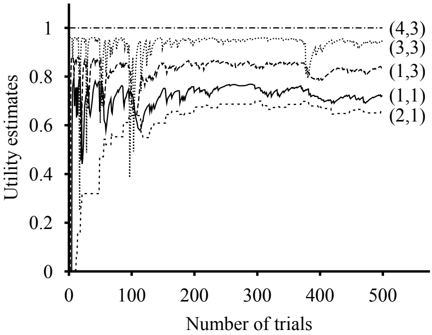
$$\begin{aligned} V(s) &\leftarrow V(s) + \Delta \\ \Delta &= \frac{1}{N(s)+1}\left(R(s) + \gamma V(s') - V(s)\right) \end{aligned}$$

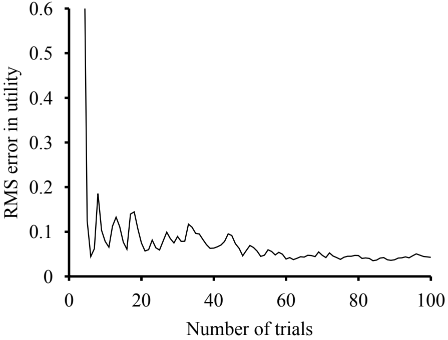
For simplicity, $\gamma = 1$.

	N	$V(s)$	Δ
a3 $\rightarrow$ a4	1	0	1/2
a2 $\rightarrow$ a3	2	0	1/6
a3 $\rightarrow$ a4	2	1/2	1/6

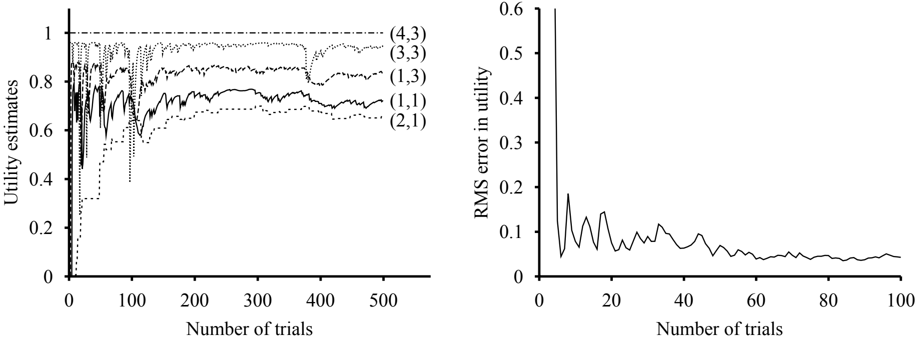
- Convergence time?

Passive RL: Problems?





Passive RL: Problems?



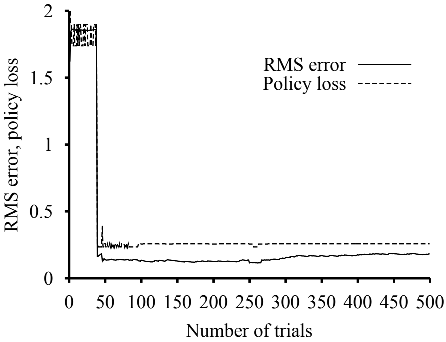
- Limited by constant policy?
- Fewer visited states cause poor estimate?

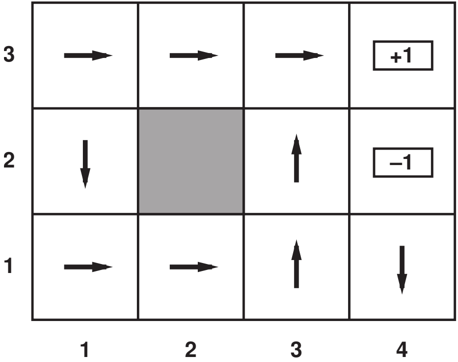
Active RL: Example

- Greedy algorithm
- After updating $V(s)$ and $N(s)$, **recalculate policy $\pi(s)$**

Active RL: Example

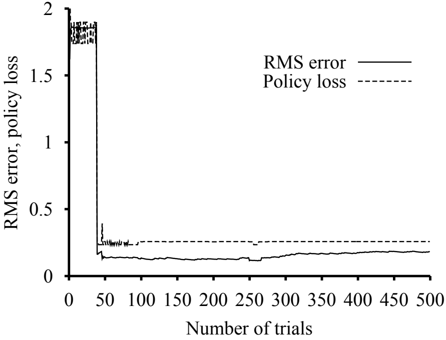
- Greedy algorithm
- After updating $V(s)$ and $N(s)$, **recalculate policy $\pi(s)$**

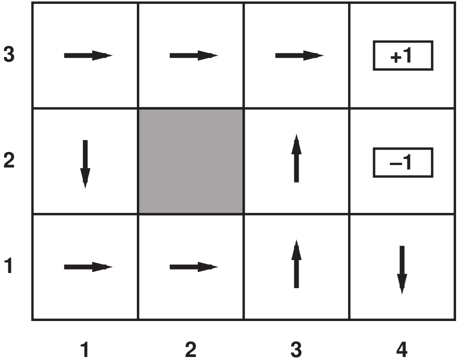




Active RL: Example

- Greedy algorithm
- After updating $V(s)$ and $N(s)$, **recalculate policy $\pi(s)$**





- Greedy algorithm cannot find optimal policy $\Rightarrow$ needs more exploration

How to Improve Active RL?

Source of errors:

Reason for error:	sampling	policy
V too low		
V too high		
increase N helps?		

How to Improve Active RL?

Source of errors:

Reason for error:	sampling	policy
V too low	T	
V too high	T	
increase N helps?	T	

How to Improve Active RL?

Source of errors:

Reason for error:	sampling	policy
V too low	T	T
V too high	T	F
increase N helps?	T	F

How to Improve Active RL?

Source of errors:

Reason for error:	sampling	policy
V too low	T	
V too high	T	
increase N helps?	T	

Exploration vs. Exploitation:

- We can't do without it
- We can't live with too much of it

How to Improve Active RL?

Source of errors:

Reason for error:	sampling	policy
V too low	T	
V too high	T	
increase N helps?	T	

Exploration vs. Exploitation:

- We can't do without it
- We can't live with too much of it

Exploration:

- Minimize it, use random moves?

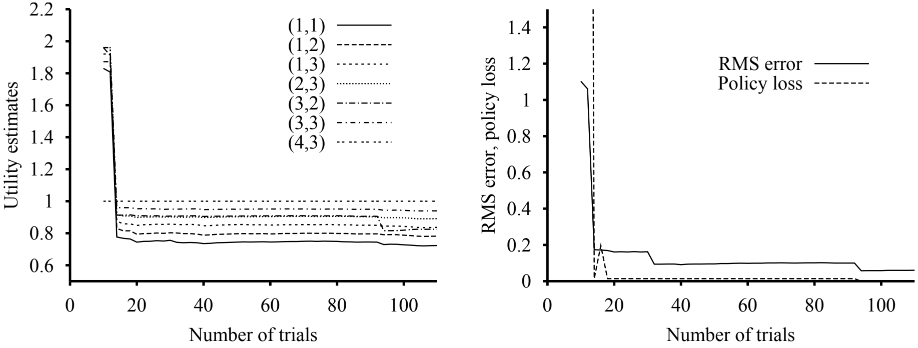
Exploring Agent

	1	2	3	4
a	+1	+1	+1	+1
b	+1		+1	+1
c	S	+1	+1	+1

- Initialize all $V(s) = +R$ (e.g., +1)
- Until $N(s)$>e; exploration threshold
- Then use $V(s)$

- Wait until built confidence

Exploring Agent Does Much Better



Q-Learning

Instead of $V(s)$, use $Q(s, a)$:

$$Q(s, a) = \arg\max_{a} V(s)$$

then the value iteration becomes

$$Q(s, a) = R(s) + \gamma \sum_{s'} P(s'|s, a) \arg\max_{a'} Q(s', a')$$

Q-Learning

Instead of $V(s)$, use $Q(s, a)$:

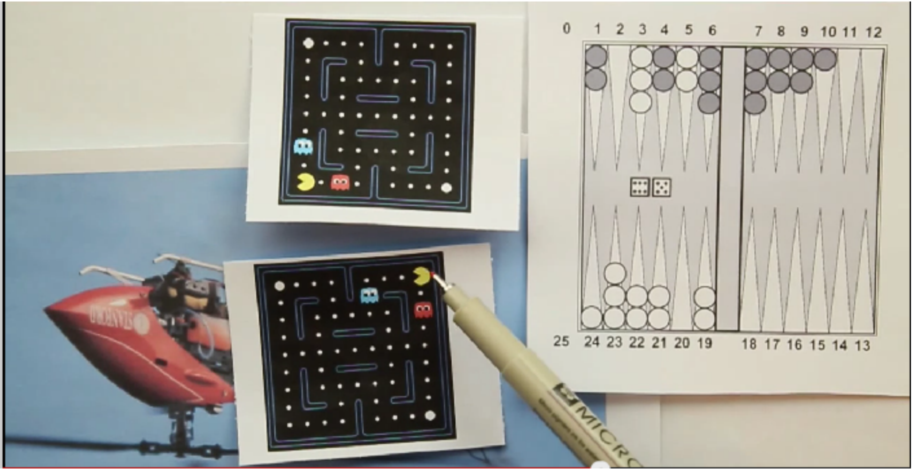
$$Q(s, a) = \arg\max_{a} V(s)$$

then the value iteration becomes

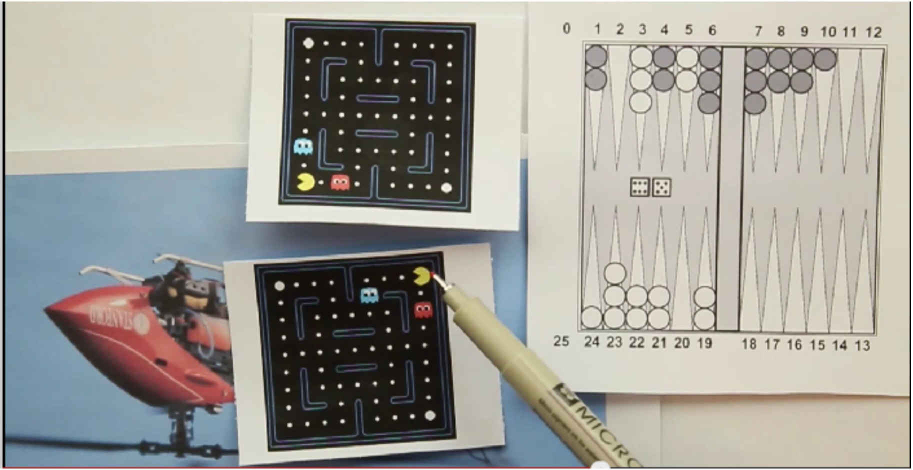
$$Q(s, a) = R(s) + \gamma \sum_{s'} P(s'|s, a) \arg\max_{a'} Q(s', a')$$

- State of the art, but also has problems with dimensionality

Q-Learning in Real World Problems



Q-Learning in Real World Problems



- Translate problem space to feature space: $s = [f_1, \ldots, f_m]$