



هوش مصنوعی

یادگیری تقویتی

Reinforcement Learning

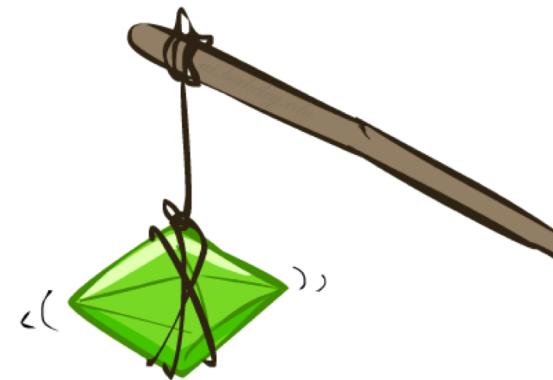
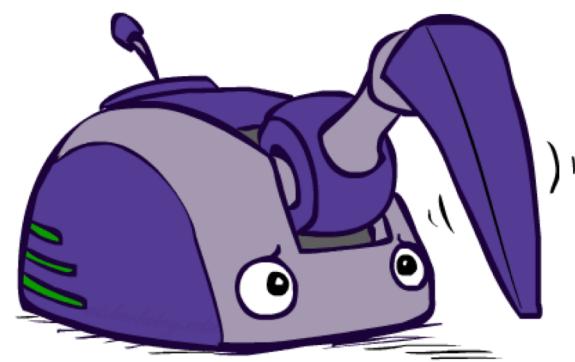
بنام حُسَن راوندجان دُر



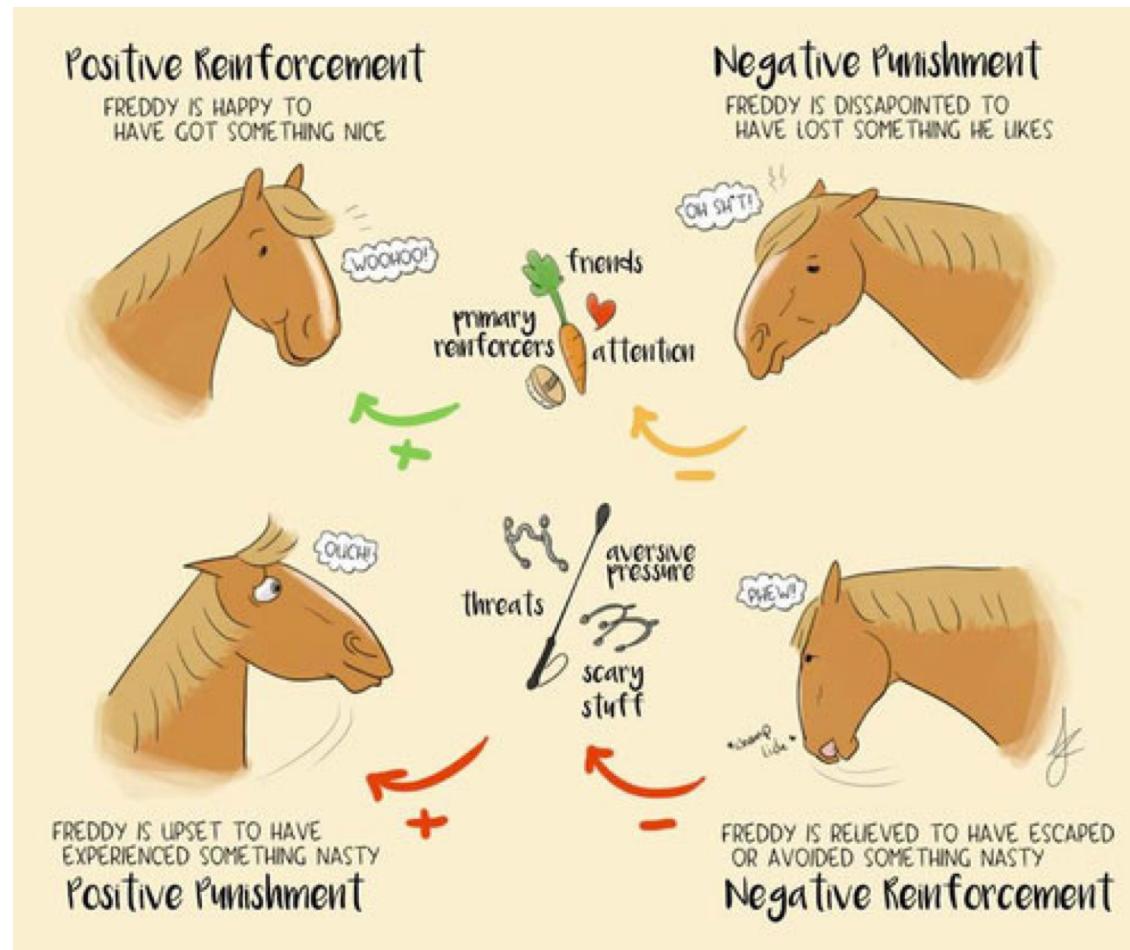
محمد طاهر پیلهور

[Many of the slides were borrowed from Dan Klein and Pieter Abbeel for CS188 Intro to AI at UC Berkeley.]

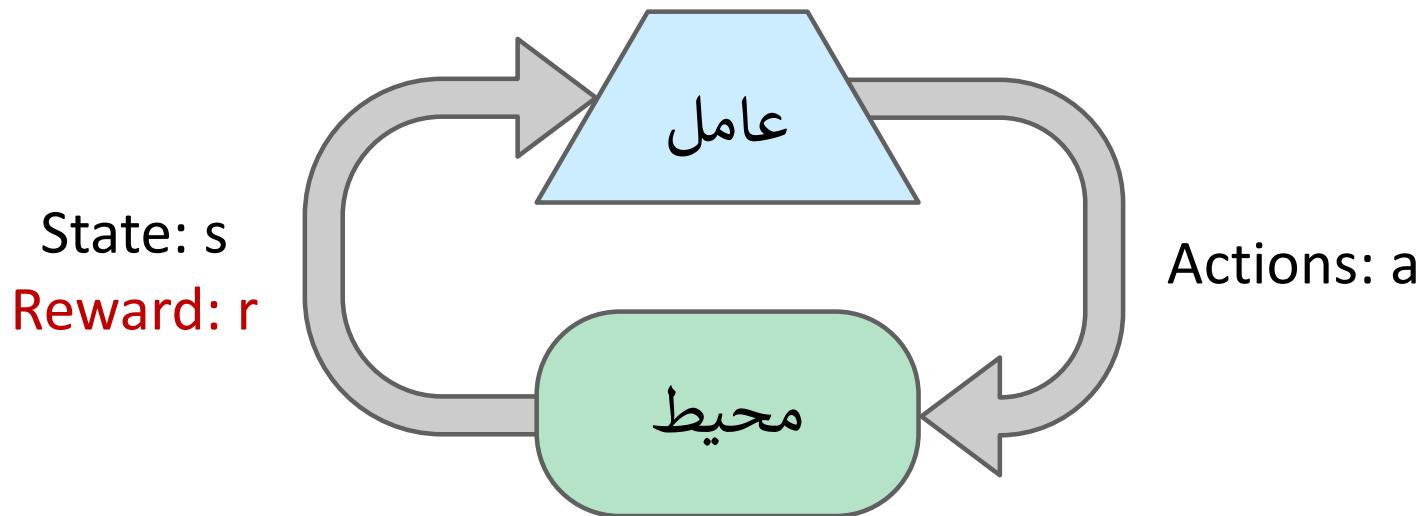
يادگیری تقویتی



یادگیری تقویتی



یادگیری تقویتی



ایده‌ی کلی:

- فیدبک از محیط در قالب پاداش دریافت می‌شود
- سودمندی عامل با توجه به پاداش تعریف می‌شود
- هدف عامل بیشینه کردن پاداش است
- یادگیری بر اساس مشاهدات انجام می‌گیرد!

مثال: یادگیری راه رفتن



ابتدا



در حال یادگیری



بعد از ۱۰۰۰ مرحله (یادگیری)

مثال: یادگیری راه رفتن



Initial

[Kohl and Stone, ICRA 2004]

ویدئو Aibo1.mp4

مثال: یادگیری راه رفتن



Training

[Kohl and Stone, ICRA 2004]

ویدئو Aibo2.mp4

مثال: یادگیری راه رفتن



Finished

[Kohl and Stone, ICRA 2004]

ویدئو Aibo3.mp4

مثال: Sidewinding



[Andrew Ng]

ویدئو [sidewinding.mp4](#)

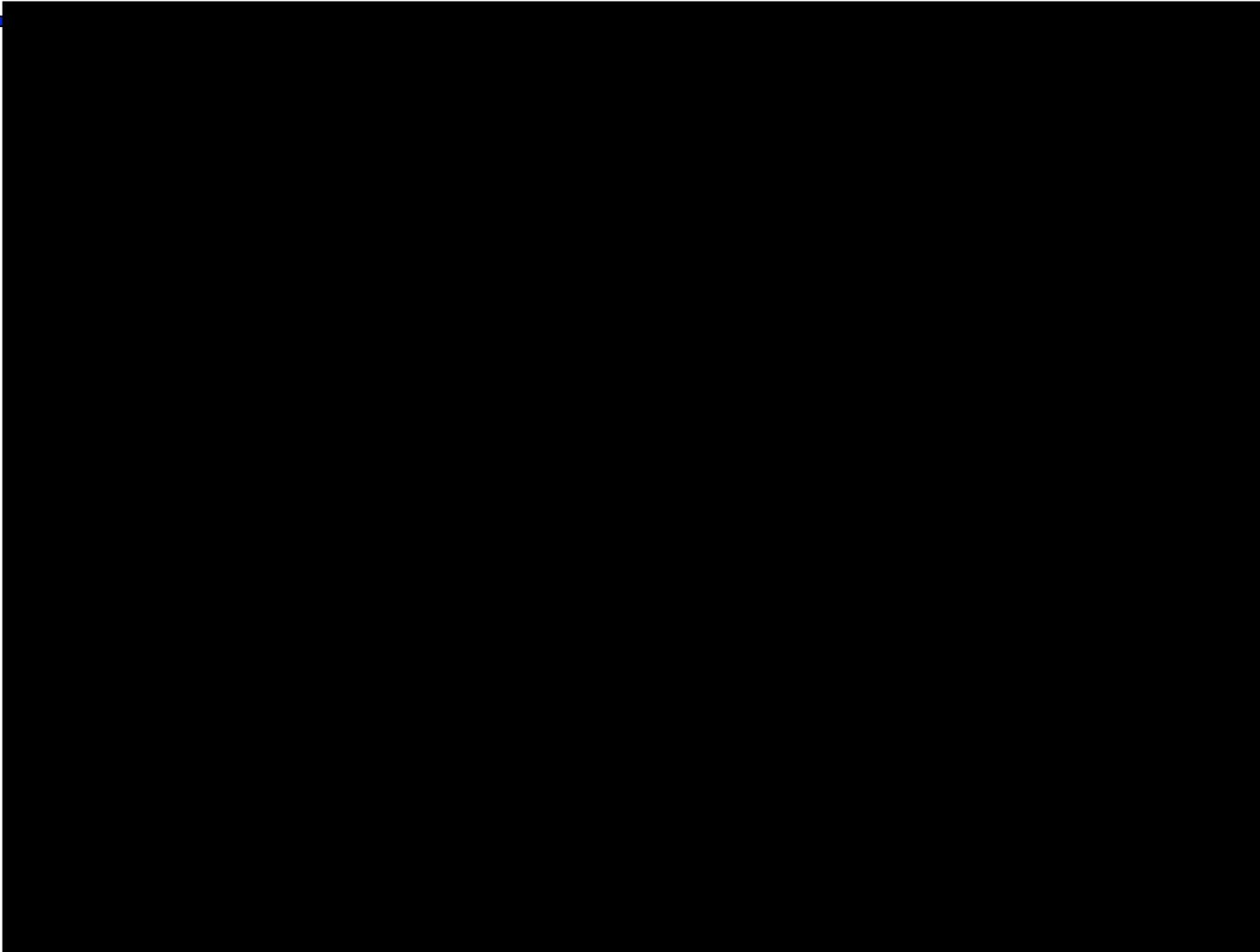
مثال: روبات نوپا



[Tedrake, Zhang and Seung, 2005]

ویدئو 4
[toddler.mp4](#)

مثال : Crawling Robot



یادگیری تقویتی



▪ هنوز فرض بر فرآیندهای مارکوفی است

- A set of states $s \in S$
- A set of actions (per state) A
- A model $T(s,a,s')$
- A reward function $R(s,a,s')$

وهمچنان دنبال راهبرد $\pi(s)$ هستیم

▪ تفاوت: T و R را نداریم!

- ایده‌ای نداریم از اینکه: کدام حالات خوب هستند و نتیجه هر action چیست.
باید اینها را خودمان کشف کنیم و یاد بگیریم!

یادگیری تقویتی

T	٪ ۸۰ اوقات درست می‌رویم ٪ ۱۰ به راست منحرف می‌شویم ٪ ۱۰ به چپ
R	جایزه کوچک: برای هر حرکت (منفی) جایزه بزرگ: برای رسیدن به هدف (+1 یا -1)

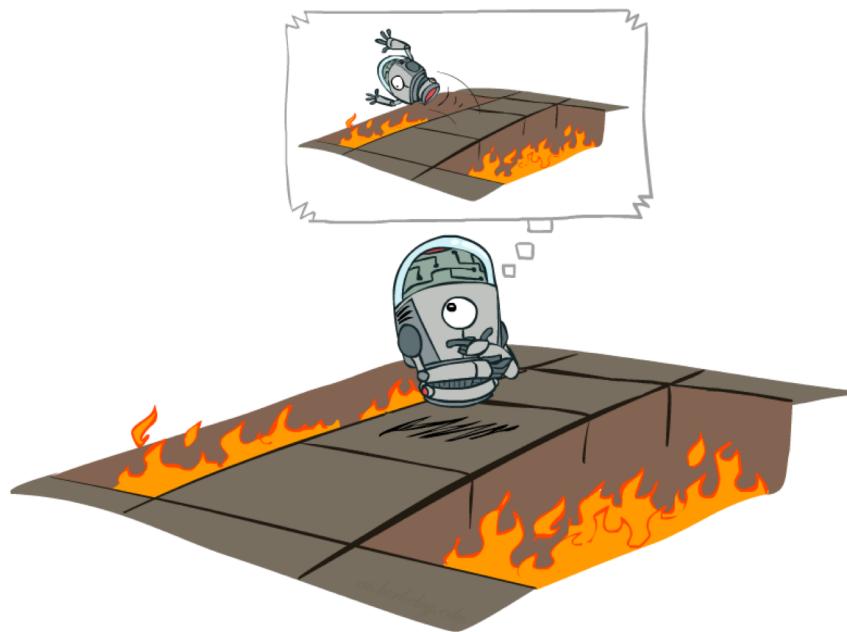
همچنان دنبال راهبرد $\pi(s)$ هستیم

■ T و R را نداریم!

▲ 0.00	▲ 0.00	▲ 0.00	0.00
▲ 0.00		▲ 0.00	0.00
▲ 0.00	▲ 0.00	▲ 0.00	▲ 0.00

0.64 ▶ 0.57	0.74 ▶ 0.49	0.85 ▶ 0.48	1.00 -1.00
0.49 ◀ 0.43	0.48 ◀ 0.28		

Offline (MDPs) vs. Online (RL)

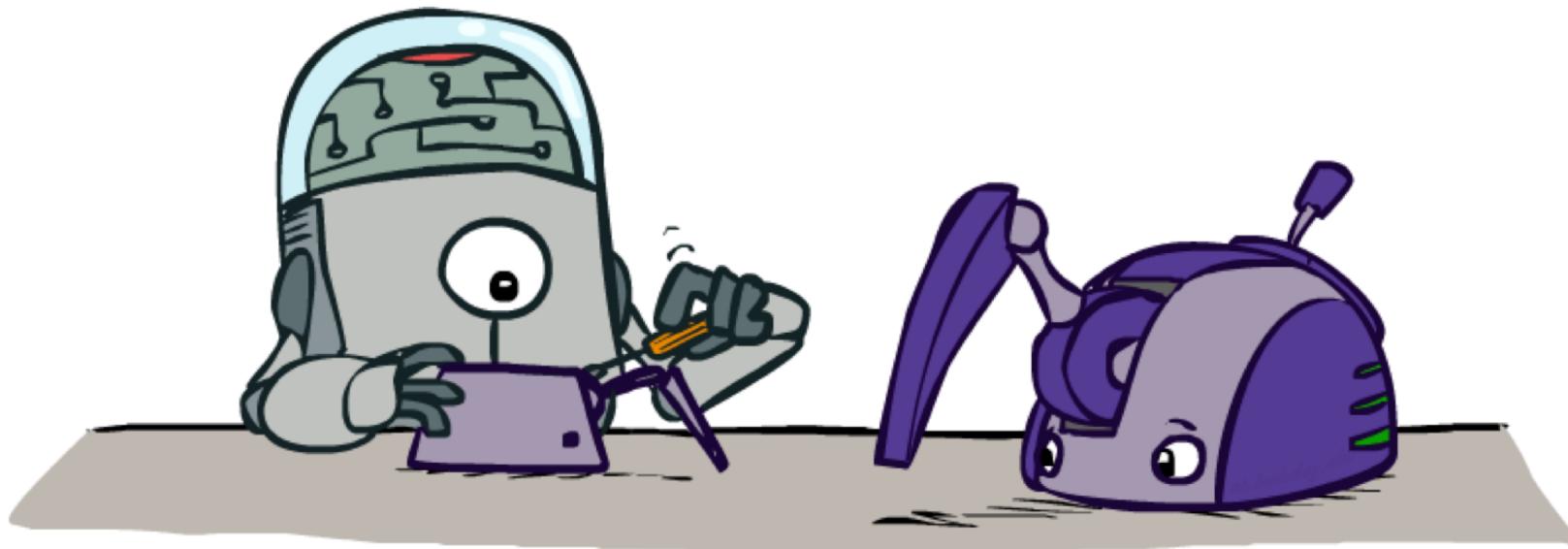


راحل Offline



یادگیری Online

یادگیری Model-Based



یادگیری Model-Based

■ ایده:

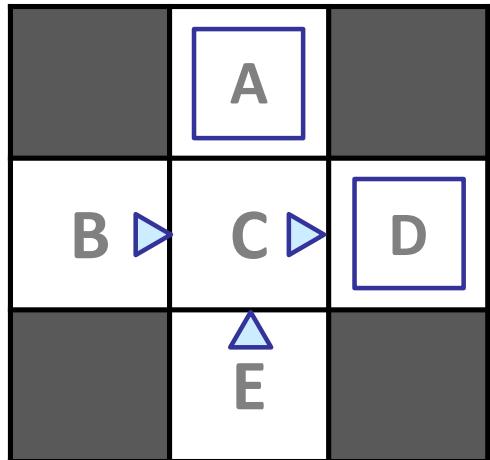
- با استفاده از اطلاعاتی که تا به حال جمع کرده‌ای، یک مدل MDP تخمینی بساز
- این MDP را حل کن و ارزش‌های حالت (در نتیجه راهبرد) را بیاب



- Step 1: Learn empirical MDP model
 - Count outcomes s' for each s, a
 - Normalize to give an estimate of $\hat{T}(s, a, s')$
 - Discover each $\hat{R}(s, a, s')$ when we experience (s, a, s')
- Step 2: Solve the learned MDP
 - For example, use value iteration, as before

مثال: یادگیری Model-Based

Input Policy π



مشاهدات (یادگیری)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

مدل بدست آمده

$\hat{T}(s, a, s')$

$T(B, \text{east}, C) = 1.00$
 $T(C, \text{east}, D) = 0.75$
 $T(C, \text{east}, A) = 0.25$
...

$\hat{R}(s, a, s')$

$R(B, \text{east}, C) = -1$
 $R(C, \text{east}, D) = -1$
 $R(D, \text{exit}, x) = +10$
...

مثال: میانگین سنی

هدف: پیدا کردن میانگین سنی دانشجویان درس هوش مصنوعی

Known $P(A)$

$$E[A] = \sum_a P(a) \cdot a = 0.35 \times 20 + \dots$$

در صورت نداشتن $P(A)$ باید نمونه‌گیری کنیم:

Unknown $P(A)$: “Model Based”

$$\hat{P}(a) = \frac{\text{num}(a)}{N}$$

$$E[A] \approx \sum_a \hat{P}(a) \cdot a$$

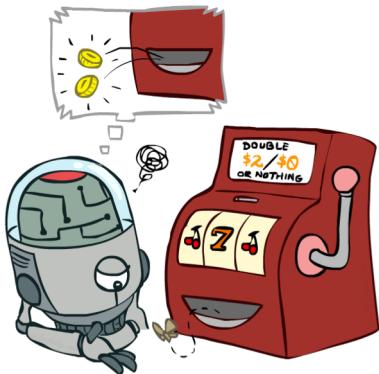
Unknown $P(A)$: “Model Free”

$$E[A] \approx \frac{1}{N} \sum_i a_i$$

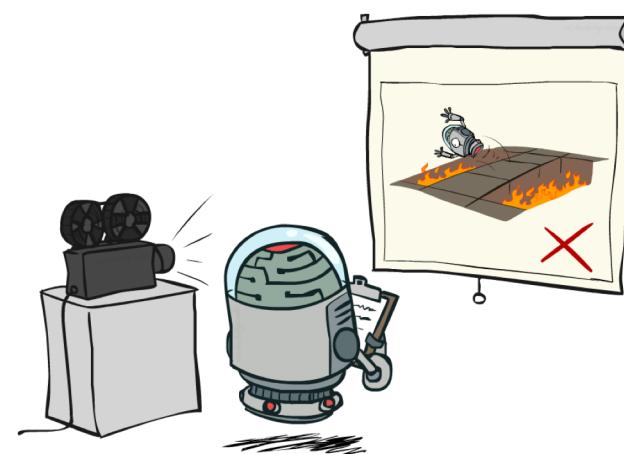
یادگیری بدون مدل – Model-Free



یادگیری بدون مدل – Model-Free



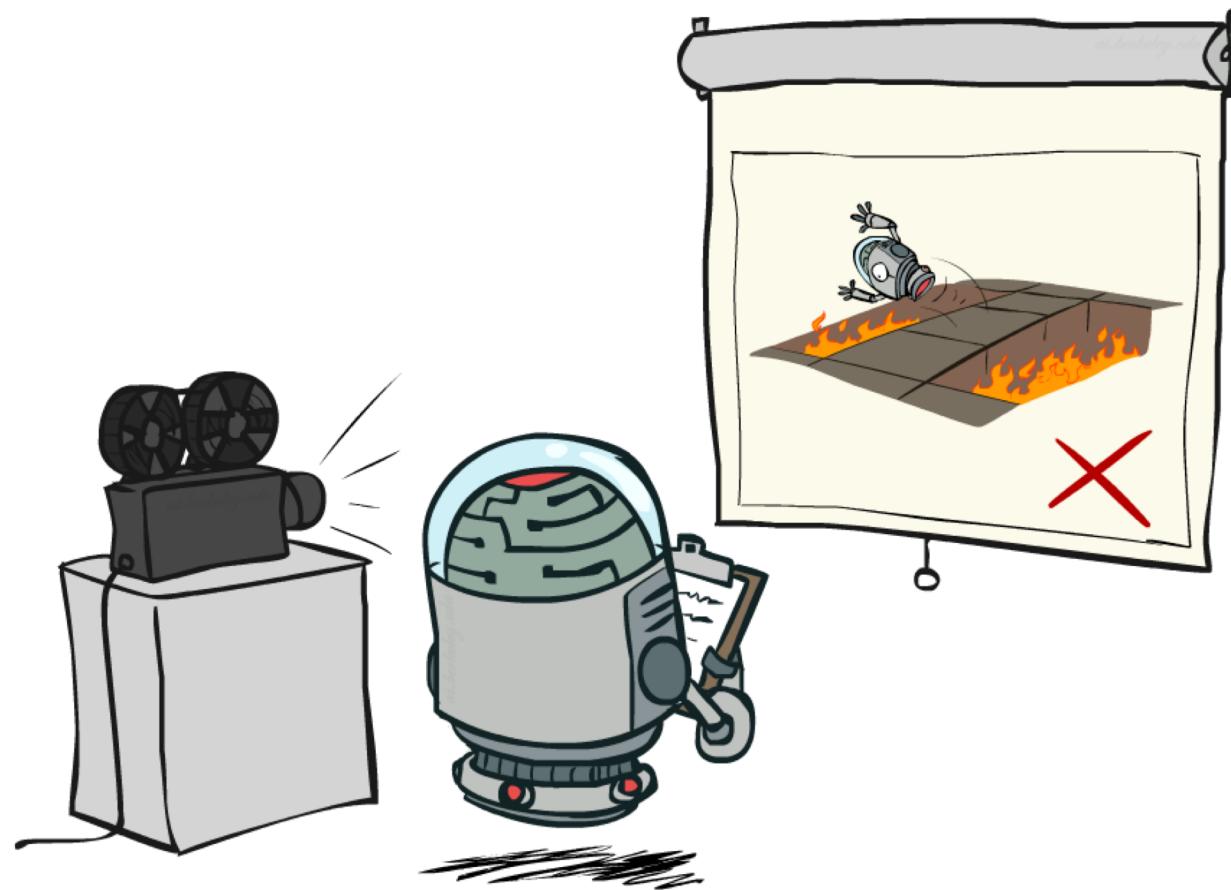
منفعل
Passive



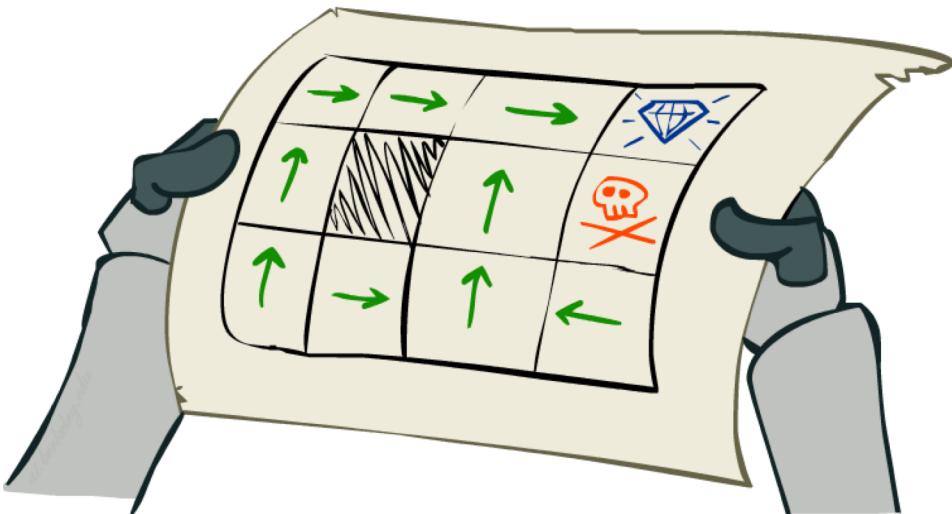
فعال
Active



یادگیری تقویتی منفعل (بدون مدل)



یادگیری تقویتی منفعل



- ورودی: یک راهبرد مشخص ($\pi(s)$)

اطلاعی از:

- احتمالات $T(s,a,s')$

- و پاداش‌های $R(s,a,s')$

نداریم

- هدف: ارزش حالت را محاسبه کنیم

- در این حالت:

- عامل یادگیرنده «هیچ‌کاره» است

- کنترلی روی a ها ندارد، تنها کاری که باید را انجام داده و مشاهده می‌کند

- فقط راهبرد را اجرا کن و از تجربه‌ات بیاموز

- نقشه‌کشی نداریم! a ها را در جهان اعمال می‌کنیم

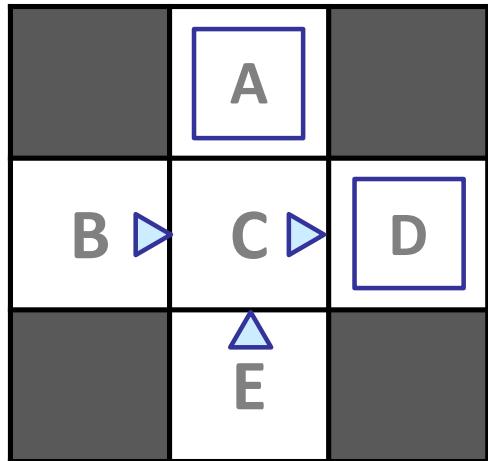
ارزیابی مستقیم



- هدف: محاسبه‌ی ارزش حالات تحت راهبرد π
- ایده: میانگین مشاهدات را محاسبه کن
 - راهبرد π را دنبال کن
 - هر بار که یک حالت را مشاهده می‌کنی، پاداشی را که در نهایت بدست آورده را یادداشت کن
 - میانگین این پاداش‌ها را حساب کن

مثال: ارزیابی مستقیم

Input Policy π



$\gamma = 1$ فرض

مشاهدات (یادگیری)

Episode 1

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 2

B, east, C, -1
C, east, D, -1
D, exit, x, +10

Episode 3

E, north, C, -1
C, east, D, -1
D, exit, x, +10

Episode 4

E, north, C, -1
C, east, A, -1
A, exit, x, -10

ارزش‌های بدست آمده

	-10	
A		
+8	+4	+10
B	C	D
	-2	
E		

مشکلات ارزیابی مستقیم

ارزش‌های محاسبه شده

	-10 A	
+8 B	+4 C	+10 D
	-2 E	

اگر E و B هر دو به C می‌روند
چرا ارزش‌هایشان تا این حد
متفاوت است؟

- مزايا

- ساده و قابل فهم

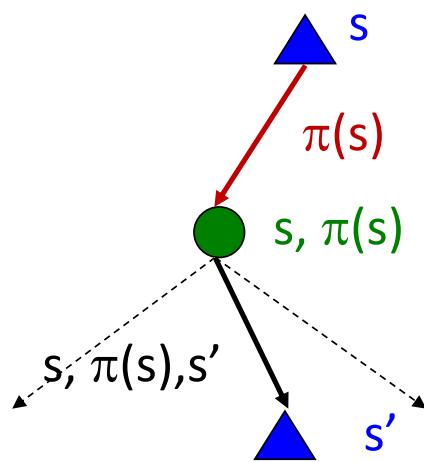
- نیازی به اطلاعات T و R نداریم

- مشکلات

- از اطلاعاتی که راجع به رابطه (اتصال) حالات داریم استفاده نمی‌کند

- زمان زیادی تلف می‌کند تا چیزی یاد بگیرد

چرا از Policy Evaluation استفاده نکنیم؟



به خوبی از ارتباط بین حالت بهره می‌برد!

$$V_0^\pi(s) = 0$$

$$V_{k+1}^\pi(s) \leftarrow \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V_k^\pi(s')]$$

- ولی محاسبه آن نیازمند دانستن مقادیر R و T است! \circledast
- سوال: چطور می‌توانیم بدون داشتن مقادیر T و R ارزش‌های V را آپدیت کنیم؟
- به عبارت دیگر، چطور میانگین وزندار را بدون داشتن وزن‌ها حساب کنیم؟

با استفاده از نمونه‌گیری Policy Evaluation

- هدف: بهبود تخمین ارزش حالت با کمک فرمول زیر:

$$V_{k+1}^{\pi}(s) \leftarrow \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + \gamma V_k^{\pi}(s')]$$

- ایده: action های مختلف را امتحان کن و میانگین بگیر

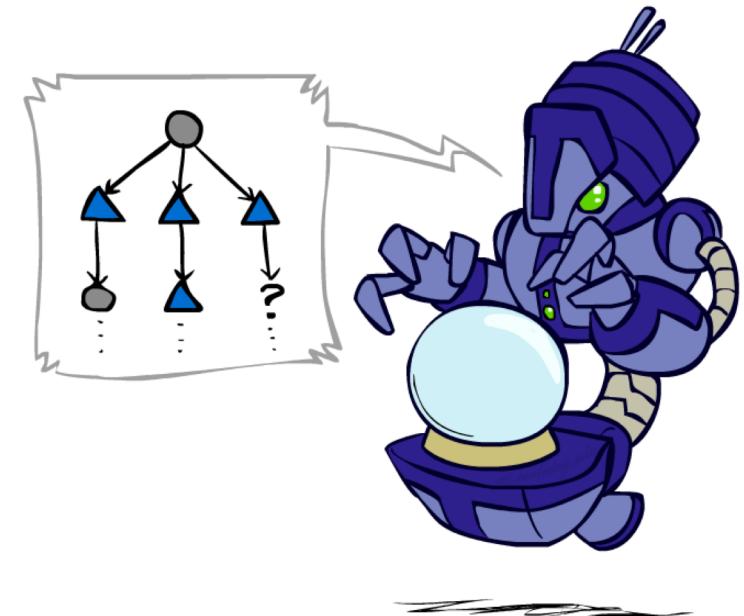
$$sample_1 = R(s, \pi(s), s'_1) + \gamma V_k^{\pi}(s'_1)$$

$$sample_2 = R(s, \pi(s), s'_2) + \gamma V_k^{\pi}(s'_2)$$

...

$$sample_n = R(s, \pi(s), s'_n) + \gamma V_k^{\pi}(s'_n)$$

$$V_{k+1}^{\pi}(s) \leftarrow \frac{1}{n} \sum_i sample_i$$



یادگیری Temporal Difference

- ایده: از هر تک مشاهده درس بگیر
- هر بار که تجربه‌ی جدیدی (s, a, s', r) داشتی، ارزش حالات را آپدیت کن

Sample of $V(s)$: $sample = R(s, \pi(s), s') + \gamma V^\pi(s')$

Update to $V(s)$: $V^\pi(s) \leftarrow (1 - \alpha)V^\pi(s) + (\alpha)sample$

Same update: $V^\pi(s) \leftarrow V^\pi(s) + \alpha(sample - V^\pi(s))$

Exponential Moving Average

فرمول آپدیت:

$$\bar{x}_n = (1 - \alpha) \cdot \bar{x}_{n-1} + \alpha \cdot x_n$$

- نمونه‌های تازه اهمیت بیشتری دارند

$$\bar{x}_n = \frac{x_n + (1 - \alpha) \cdot x_{n-1} + (1 - \alpha)^2 \cdot x_{n-2} + \dots}{1 + (1 - \alpha) + (1 - \alpha)^2 + \dots}$$

- گذشته‌ی دور را فراموش می‌کند

به مرور زمان، learning rate (alpha) را کم می‌کنیم تا به همگرایی برسیم

مثال: یادگیری Temporal Difference

حالات

	A	
B	C	D
E		

مشاهدات

B, east, C, -2

	0	
0	0	8
	0	

C, east, D, -2

	0	
-1	0	8
	0	

	0	
-1	3	8
	0	

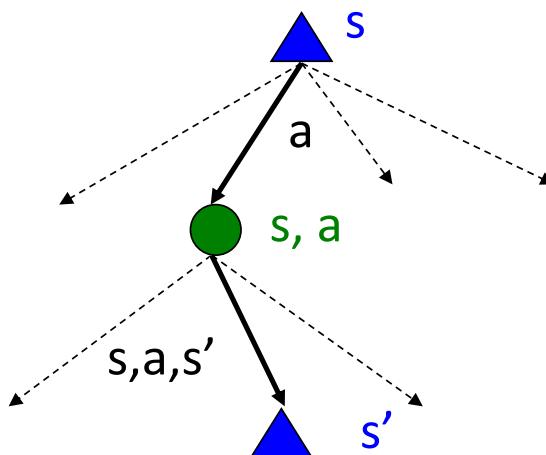
Assume: $\gamma = 1, \alpha = 1/2$

$$V^\pi(s) \leftarrow (1 - \alpha)V^\pi(s) + \alpha [R(s, \pi(s), s') + \gamma V^\pi(s')]$$

مشکل یادگیری TD

با استفاده از TD می‌توان یک راهبرد را ارزیابی کرد (ارزش حالت را برای آن بدست آورد)

- ولی، نمی‌توان با استفاده از ارزش حالت مستقیماً راهبردی بدست آورد!
- نیازمند داشتن مقادیر T و R هستیم

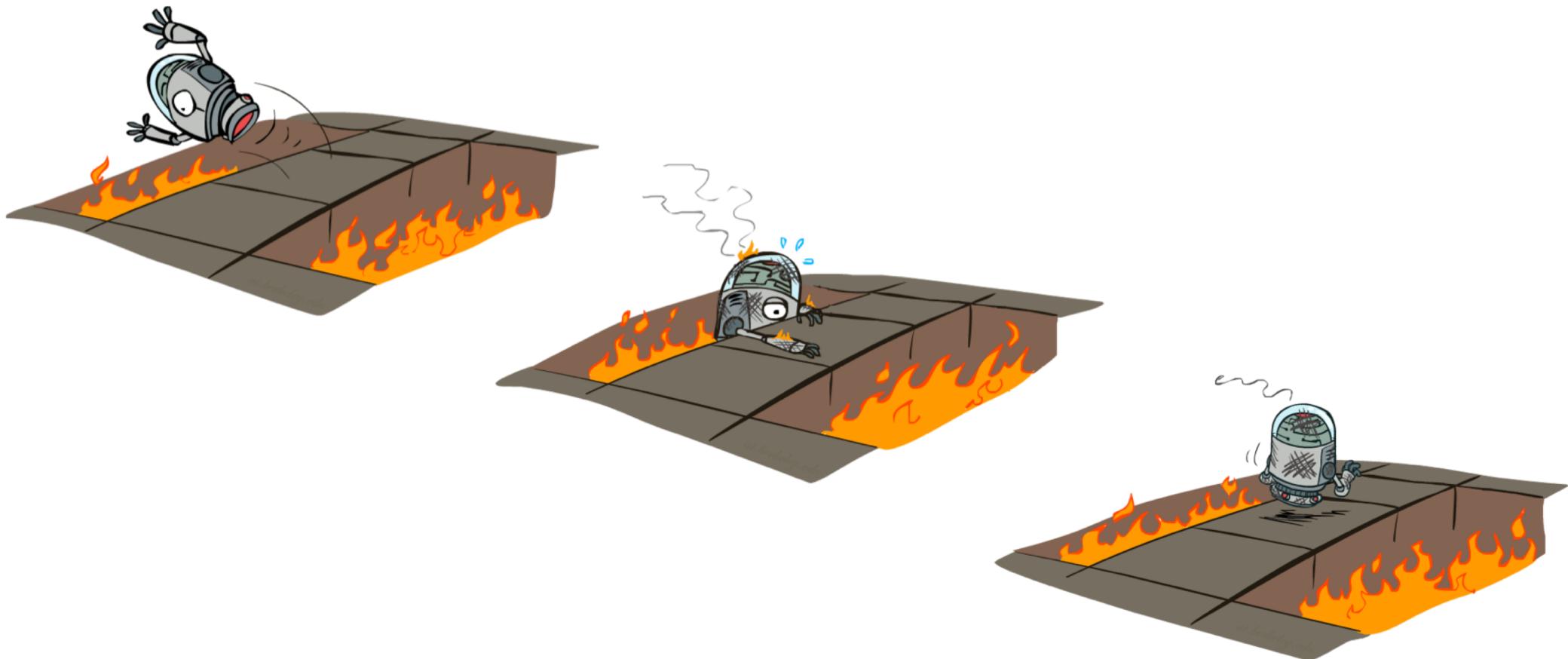


$$\pi(s) = \arg \max_a Q(s, a)$$

$$Q(s, a) = \sum_{s'} T(s, a, s') [R(s, a, s') + \gamma V(s')]$$

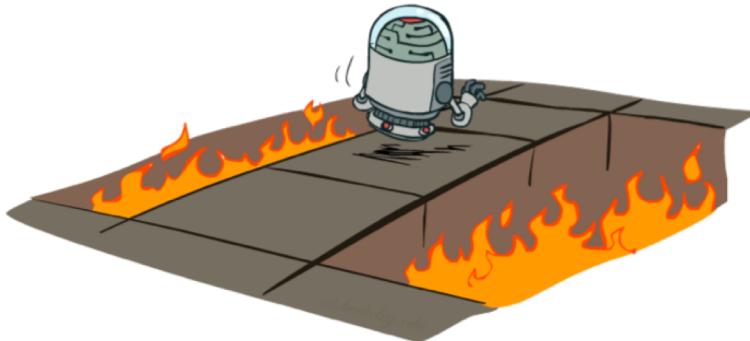
- ایده: به جای ارزش‌ها، Q-value‌ها را حساب کن!

یادگیری تقویتی فعال



یادگیری تقویتی فعال

هدف: ارزش حالات را محاسبه کنیم



اطلاعی از:

- احتمالات $T(s,a,s')$
- و پاداش‌های $R(s,a,s')$

نداریم

ولی، انتخاب **action**ها در کنترل ماست

▪ در این حالت:

- Tradeoff: exploration vs. exploitation
- همچنان نقشه‌کشی نداریم! **action**ها را در جهان اعمال می‌کنیم

Full Reinforcement Learning

Q-Value Iteration

▪ یادآوری Value iteration

▪ با $V_0(s) = 0$ شروع کردیم و با داشتن V_k به محاسبه V_{k+1} پرداختیم

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s, a, s') \left[R(s, a, s') + \gamma V_k(s') \right]$$

▪ ولی Q-value ها بیشتر به کار می‌آیند (مستقیماً می‌توان برای محاسبه راهبرد از آنها استفاده برد)

▪ از $Q_0(s, a) = 0$ شروع کن

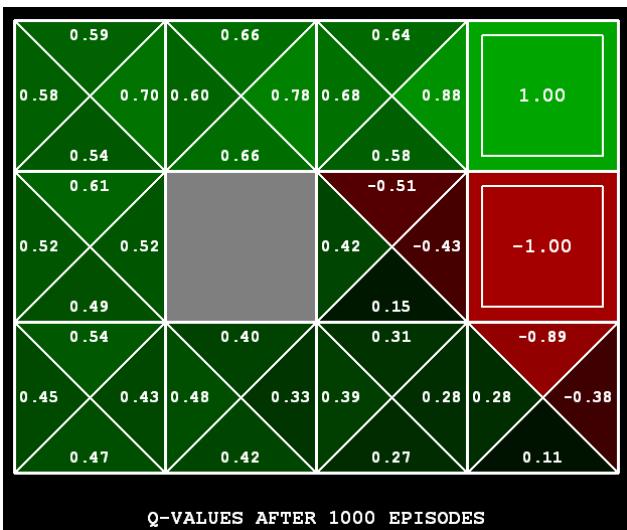
▪ و با داشتن Q_k به محاسبه Q_{k+1} پرداز

$$Q_{k+1}(s, a) \leftarrow \sum_{s'} T(s, a, s') \left[R(s, a, s') + \gamma \max_{a'} Q_k(s', a') \right]$$

الگوريتم Q-Learning

$$Q_{k+1}(s, a) \leftarrow \sum_{s'} T(s, a, s') \left[R(s, a, s') + \gamma \max_{a'} Q_k(s', a') \right]$$

■ مقادير $Q(s,a)$ را با هر مشاهده آپديت کن



■ با داشتن نمونه (s,a,s',r)

■ و تخمین قبلی: $Q(s, a)$

■ و تخمین نمونه جدید:

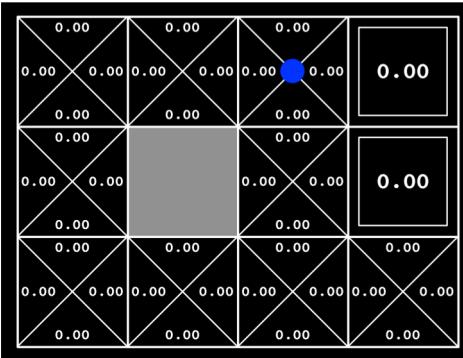
$$\text{sample} = R(s, a, s') + \gamma \max_{a'} Q(s', a')$$

■ مقدار جدید $Q(s,a)$ را محاسبه کن

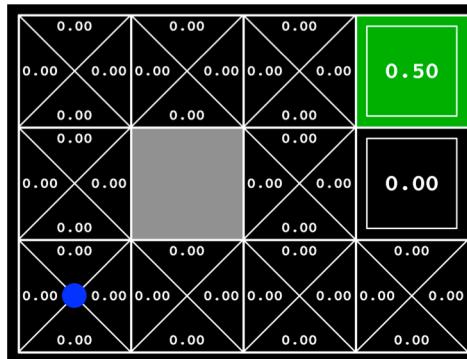
$$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + (\alpha) [\text{sample}]$$

الgoritم - تمرین – Q-Learning

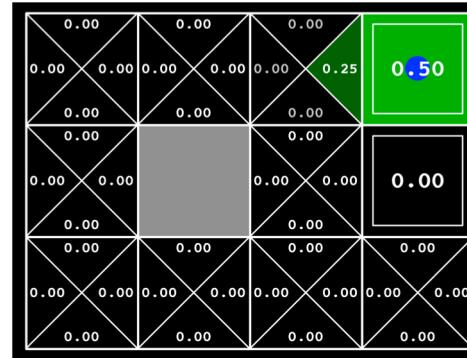
LivingReward = 0



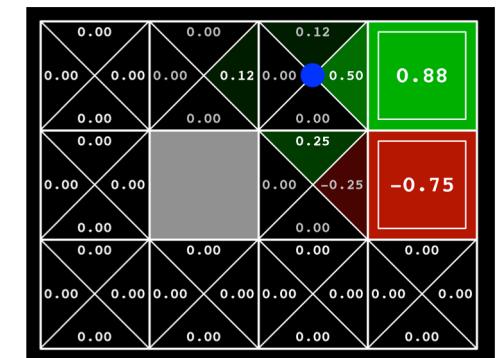
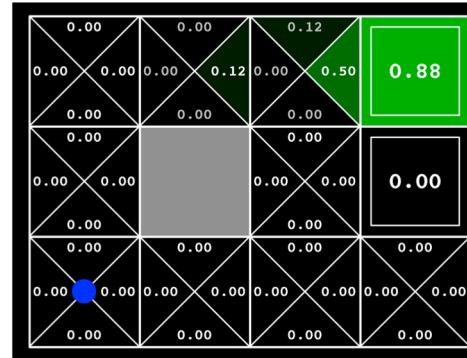
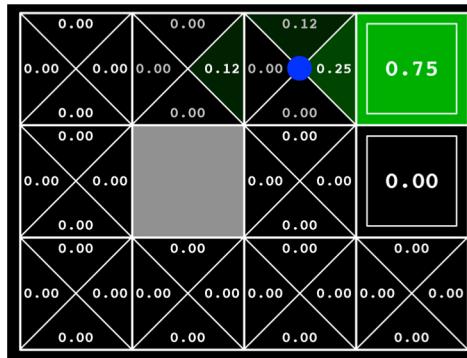
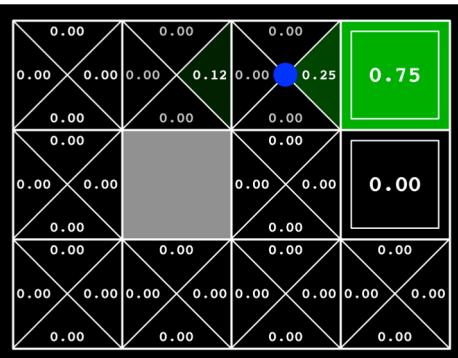
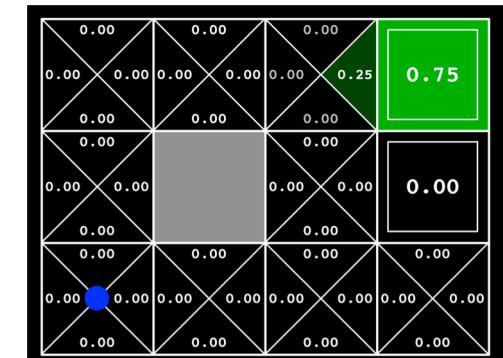
Alpha = 0.5



Noise = 0



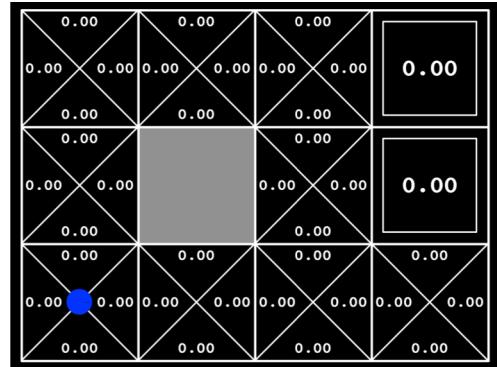
Discount = 1.0



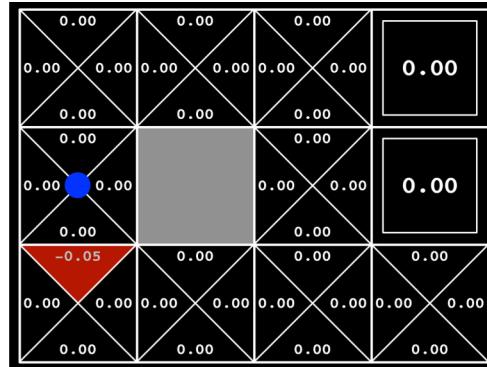
```
python gridworld.py -a q -k 100 -m --noise 0 --discount 1.0 --livingReward 0 -e 0
```

الگوريتم Q-Learning – تمرين

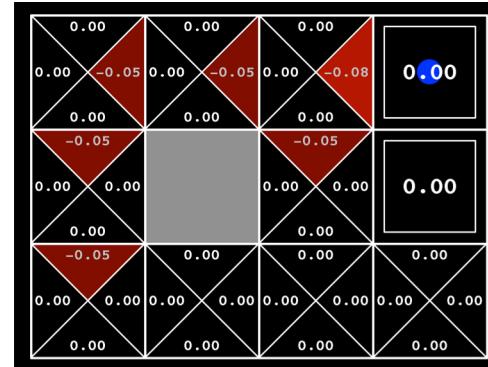
LivingReward = -0.1



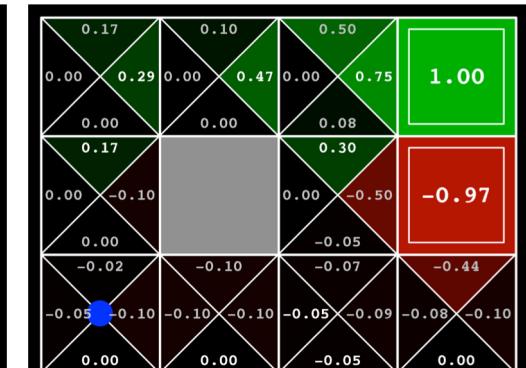
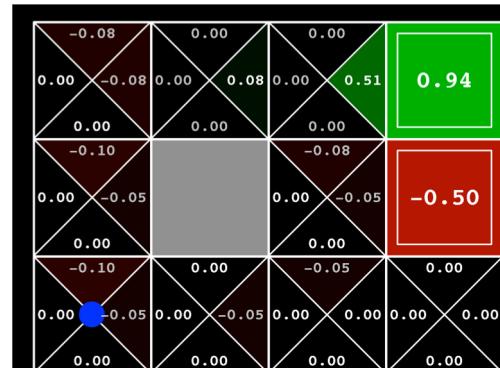
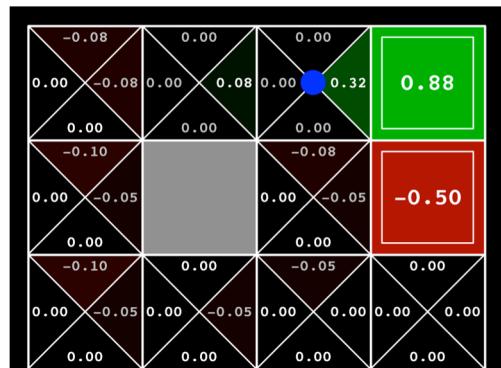
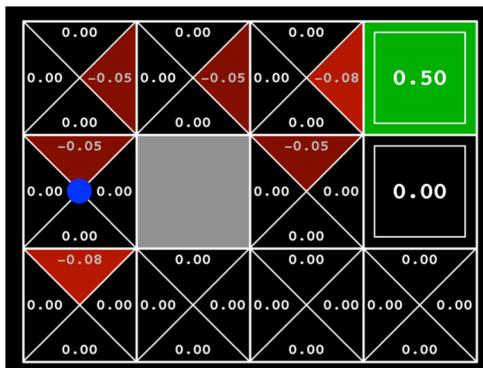
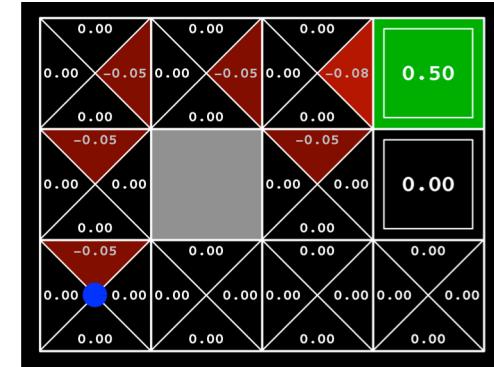
Alpha = 0.5



Noise = 0.2

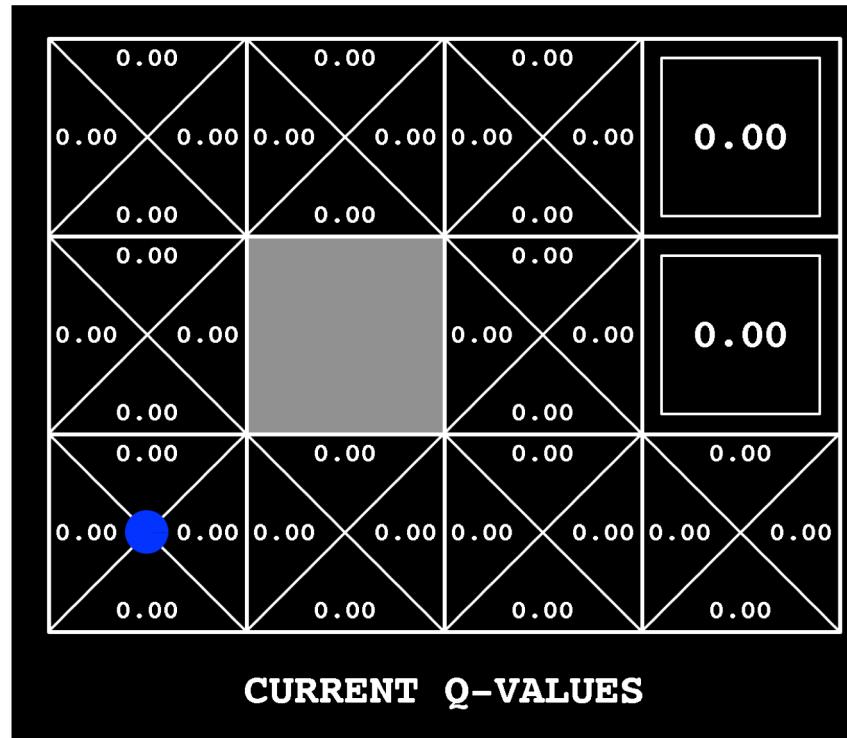


Discount = 0.9



```
python gridworld.py -a q -k 100 -m --noise 0.2 --discount 0.9 --livingReward -0.1 -e 0
```

مثال – Gridworld Learning Q

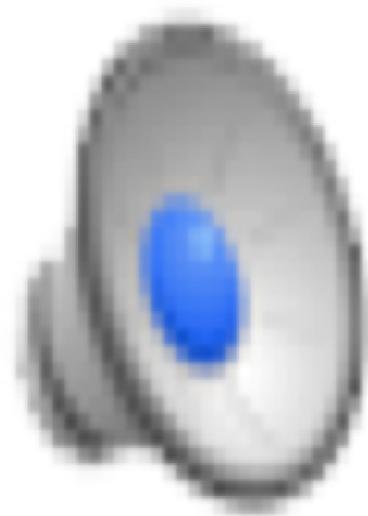


```
python gridworld.py -a q -k 50 --noise 0 -m
```

```
python gridworld.py -a q -i 100 -g BridgeGrid --discount 0.9 --noise 0.0 -m -k 50
```

ویدئو Q-learning.mp4

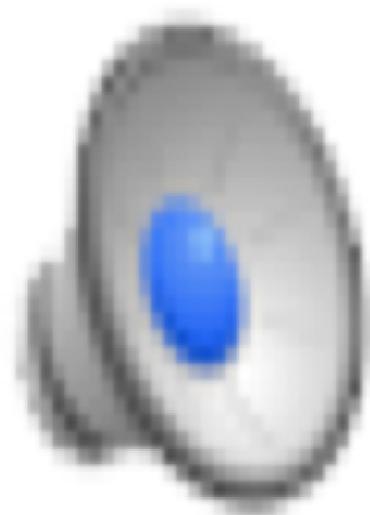
روبات خزنده



ویدئو4 CrawlerBot.mp4

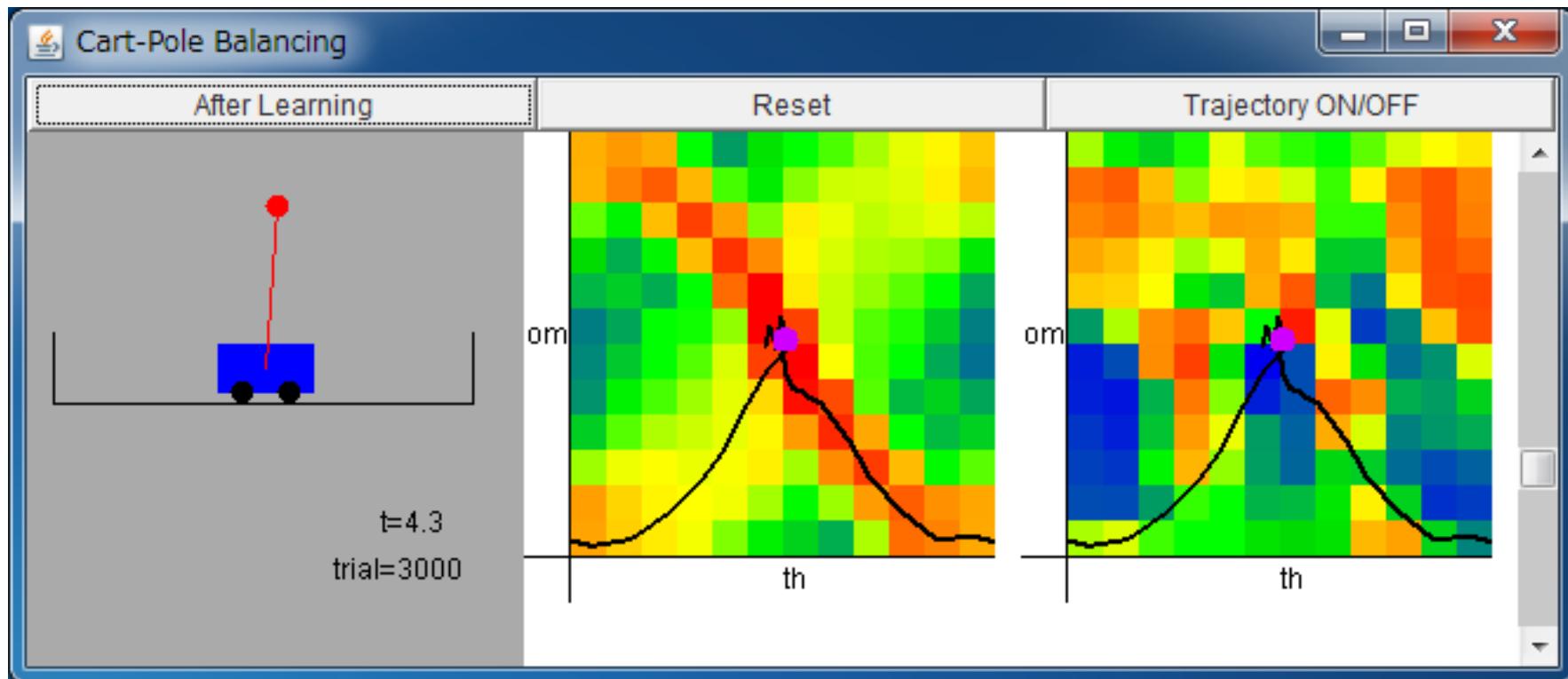
Java -jar BotQLearning.jar

مثال – Crawler Learning Q



ویدئو Q-learningCrawlingRobot.mp4

Pendulum – Q-Learning مثال

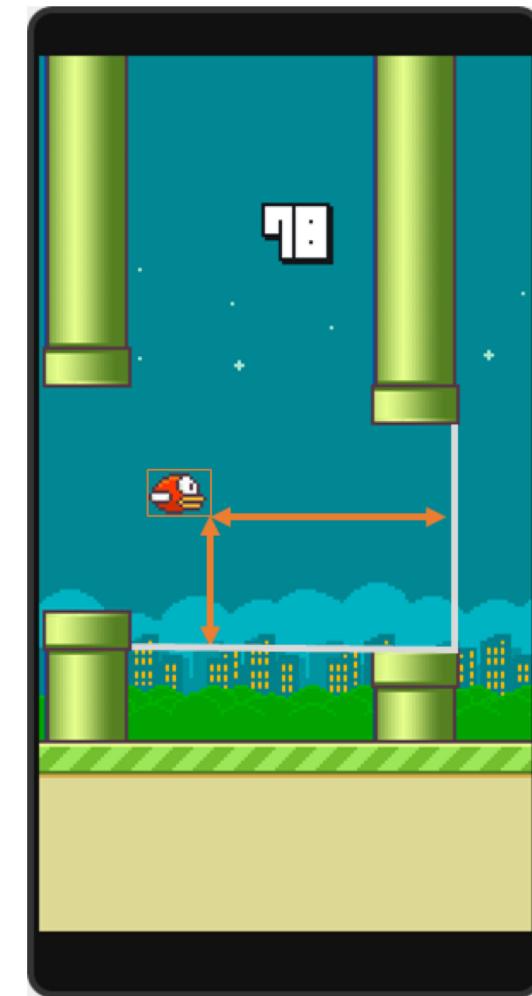


مثال – بازی Q-Learning



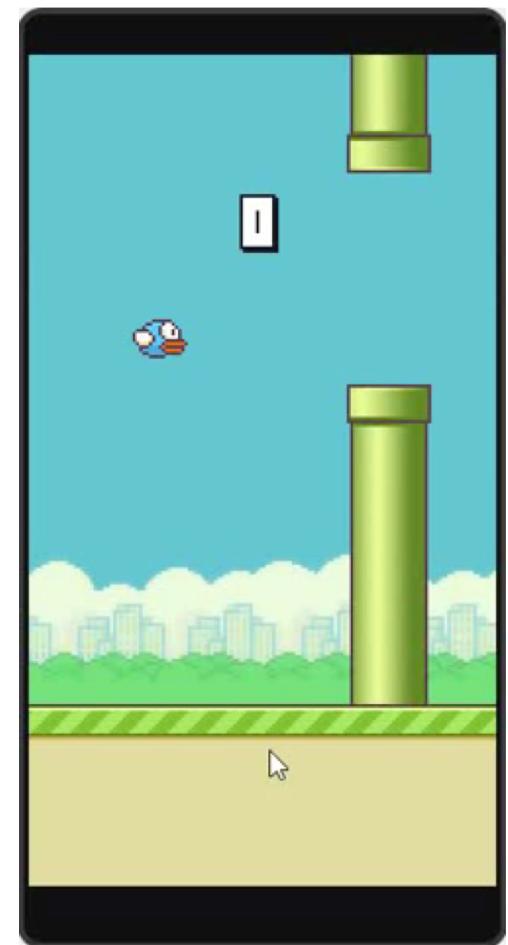
Flappy Bird RL

- State space
 - Discretized vertical distance from lower pipe
 - Discretized horizontal distance from next pair of pipes
 - Life: Dead or Living
- Actions
 - Click
 - Do nothing
- Rewards
 - +1 if Flappy Bird still alive
 - -1000 if Flappy Bird is dead
- 6-7 hours of Q-learning



Flappy Bird RL

- State space
 - Discretized vertical distance from lower pipe
 - Discretized horizontal distance from next pair of pipes
 - Life: Dead or Living
- Actions
 - Click
 - Do nothing
- Rewards
 - +1 if Flappy Bird still alive
 - -1000 if Flappy Bird is dead
- 6-7 hours of Q-learning



حل کردن MDP‌ها

در صورت دانستن مدل MDP - حل آفلاین

Policy Iteration یا Value Iteration روش: هدف: محاسبه V^*, Q^*, π^*

Policy Evaluation روش: هدف: ارزیابی یک راهبرد مشخص

در صورت نداشتن MDP

Model Based

روش: VI یا PI هدف: محاسبه V^*, Q^*, π^*

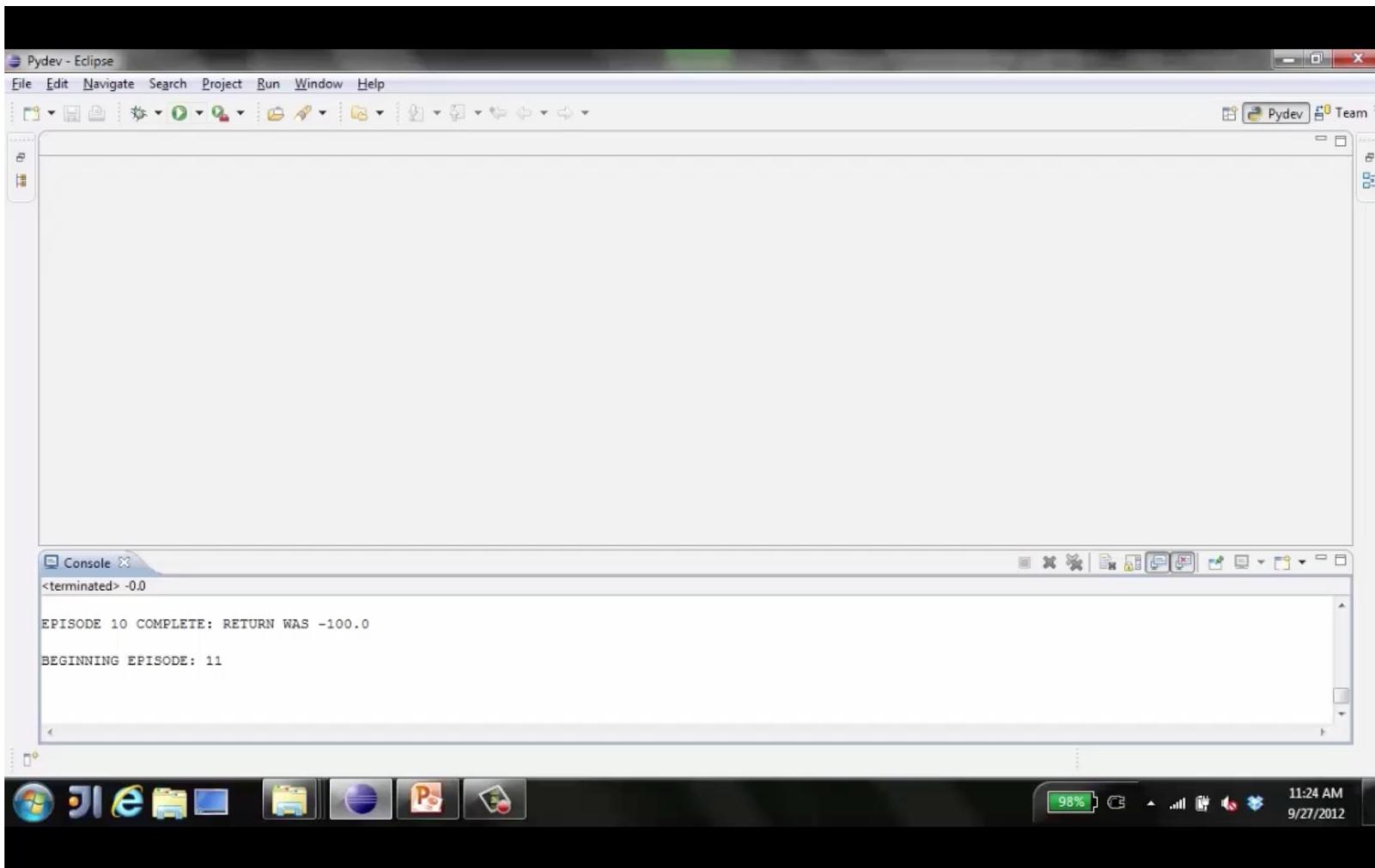
روش: PE هدف: ارزیابی یک راهبرد مشخص

Model Free

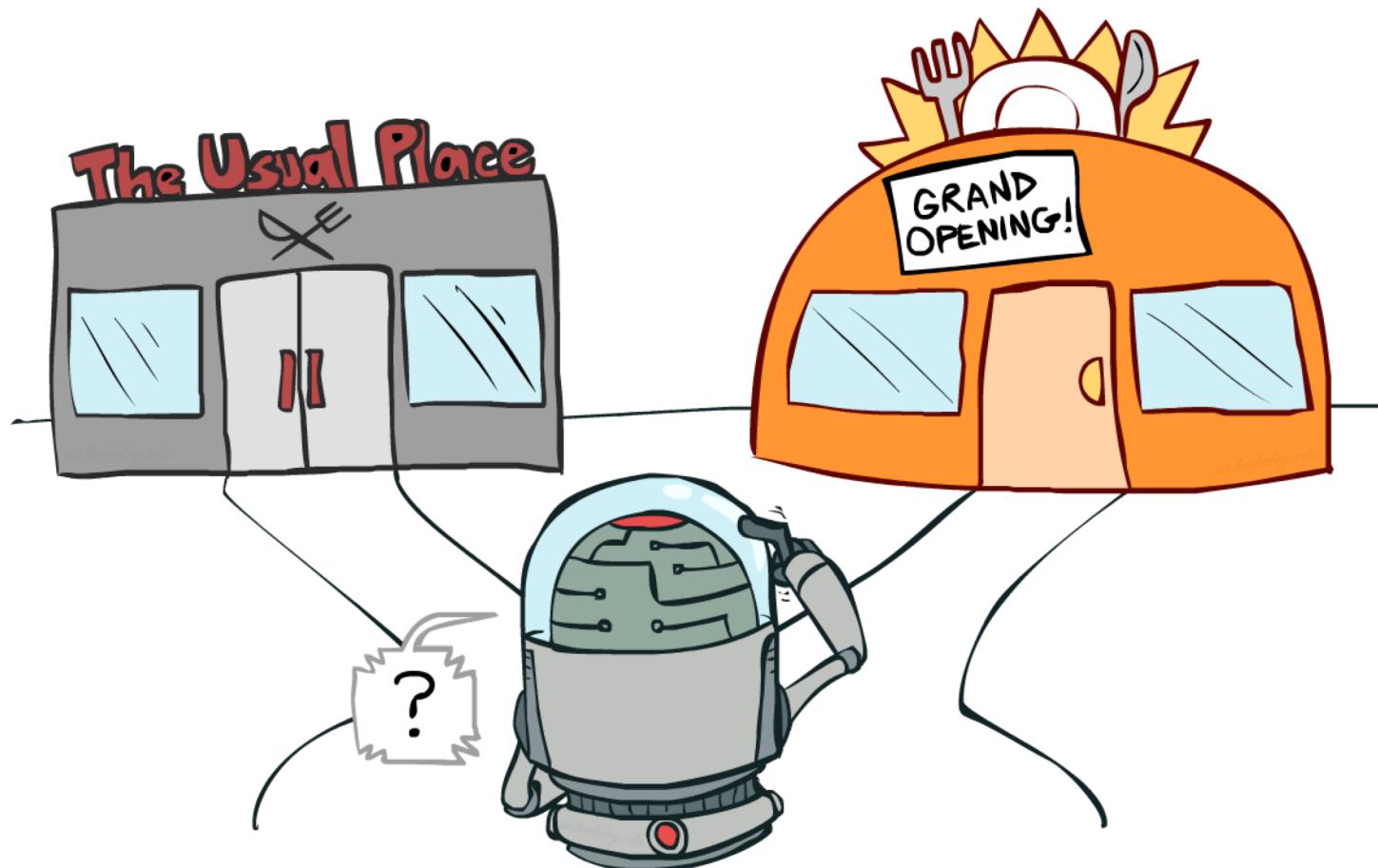
روش: Q-Learning هدف: محاسبه V^*, Q^*, π^*

روش: Value Learning هدف: ارزیابی یک راهبرد مشخص

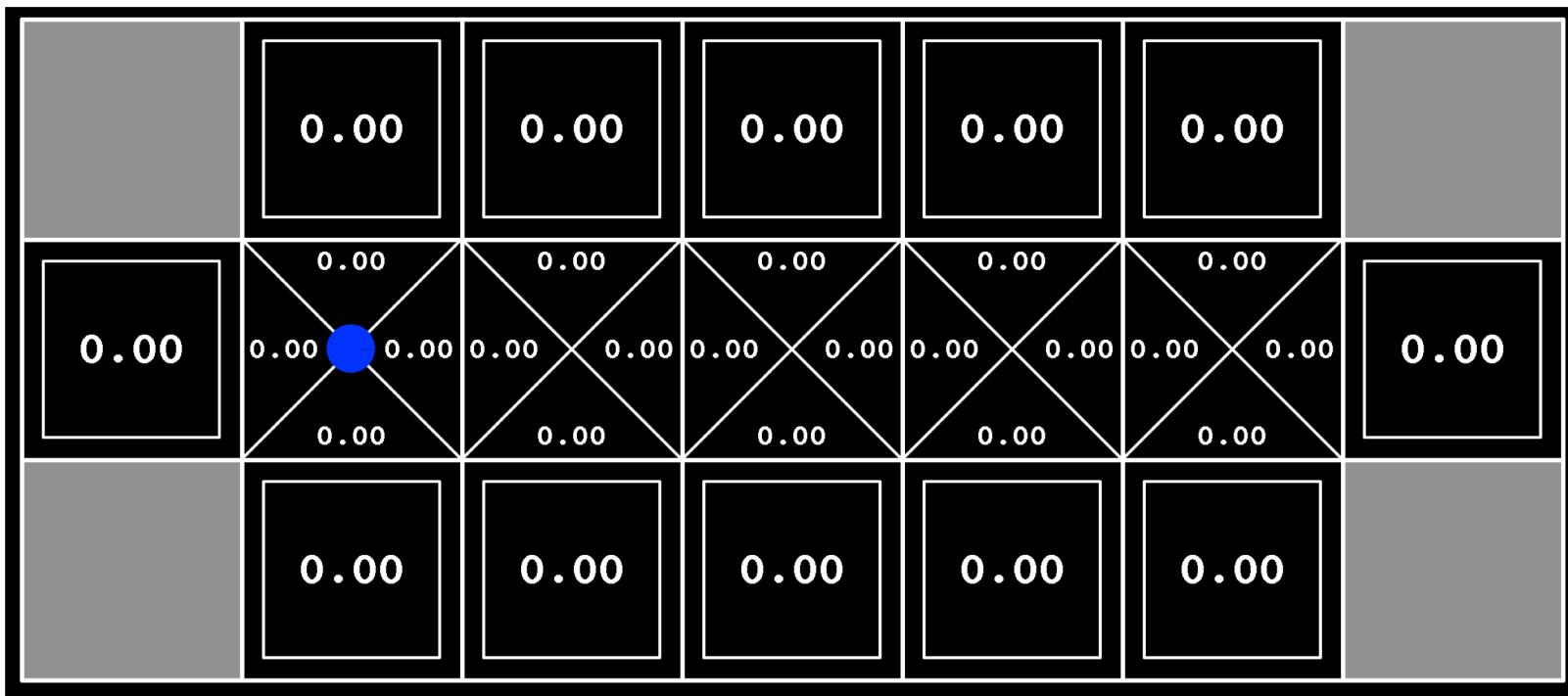
مثال: Auto Cliff Grid Learning Q-



Exploration vs. Exploitation

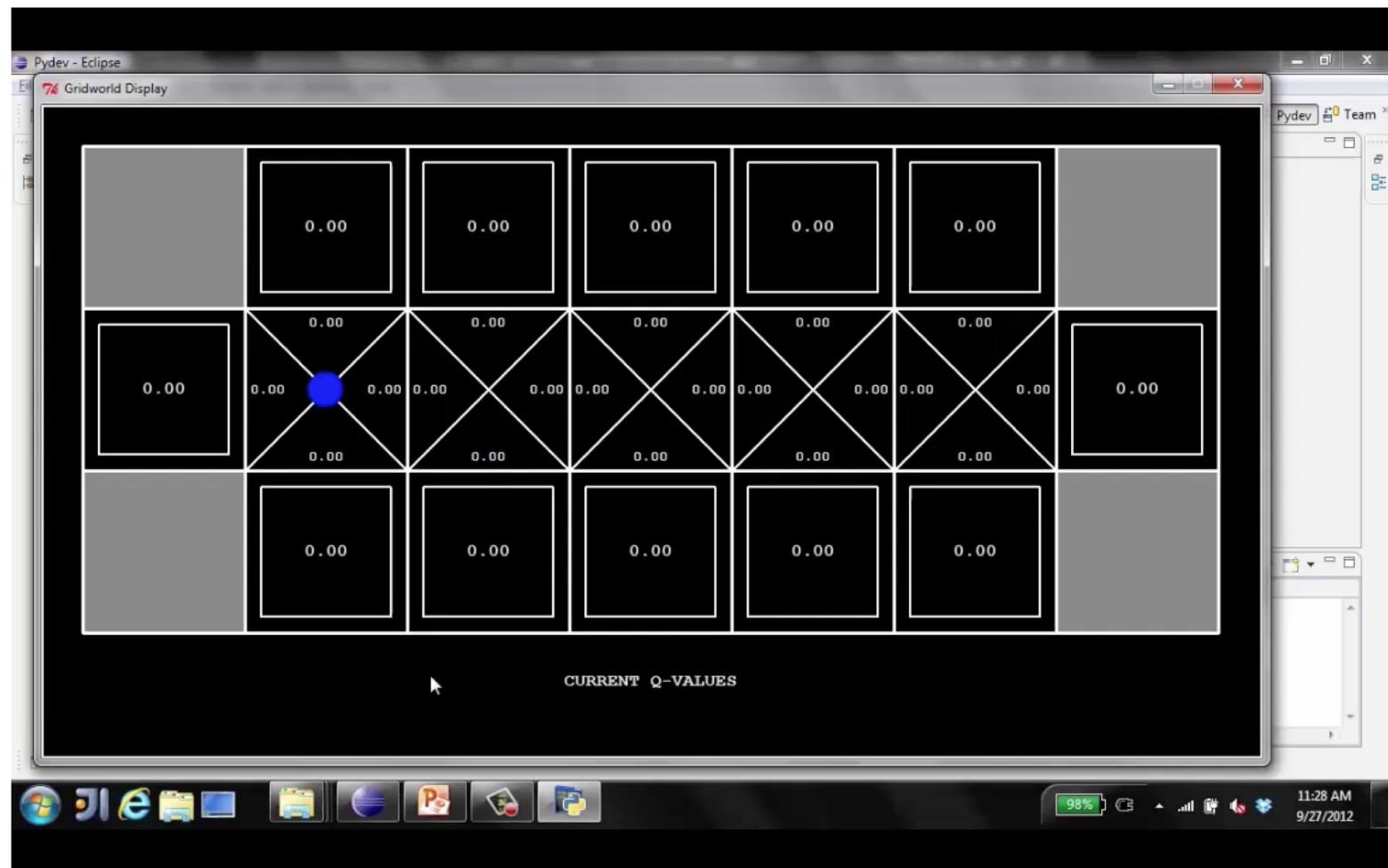


CliffGrid :مثال



```
python gridworld.py -a q -k 50 -n 0 -g BridgeGrid -e 1 -m
```

مثال : CliffGrid



Explore - کشf

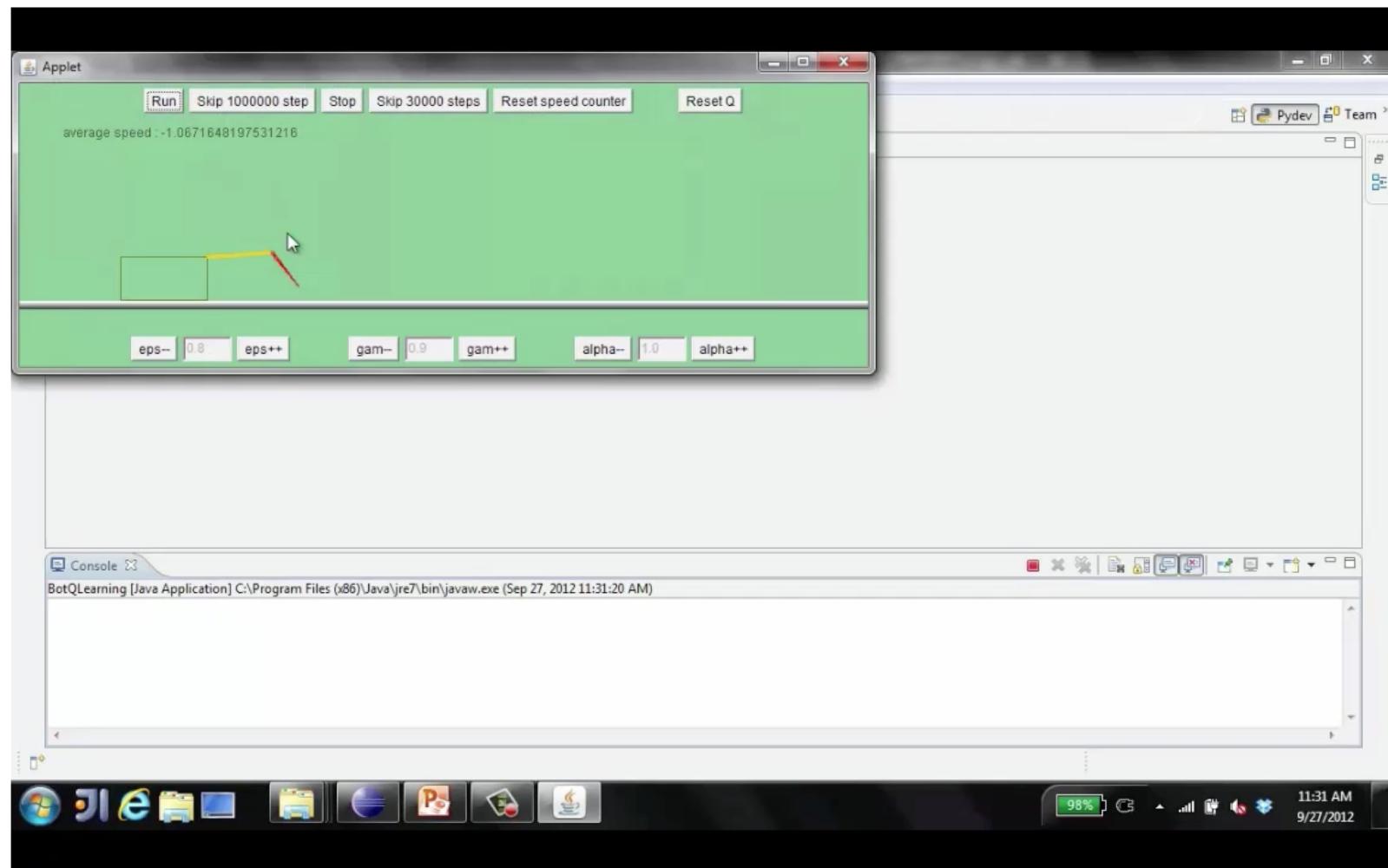


می خواهیم که عامل ما دنبال کشf موقعیت های بهتر برود

- ساده ترین راه حل: ϵ -greedy
- در هر انتخاب یک تاس بنداز
- با یک احتمال کوچک (ϵ)، یک action تصادفی انتخاب کن
- در غیر اینصورت، مطابق راهبرد پیش برو

مشکلات انتخاب تصادفی؟

مثال: Epsilon-Greedy – Crawler



Explore - کشf



می خواهیم که عامل ما دنبال کشf موقعیت های بهتر برود

- ساده ترین راه حل: ϵ -greedy
- در هر انتخاب یک تاس بنداز
- با یک احتمال کوچک (ϵ)، یک action تصادفی انتخاب کن
- در غیر اینصورت، مطابق راهبرد پیش برو

مشکلات انتخاب تصادفی؟

- اگر مقدار ϵ زیاد باشد، حتی اگر عامل تجربه‌ی زیادی داشته باشد، بد عمل می‌کند
- یک راه حل: مقدار ϵ را به مرور زمان کم کنیم
- راه حل دیگر: توابع اکتشاف!

توابع اکتشاف



چه جاهایی را جستجو کنیم؟

- جاهایی که هنوز از بی‌فایده بودن آنها مطمئن نیستیم

تابع اکتشاف

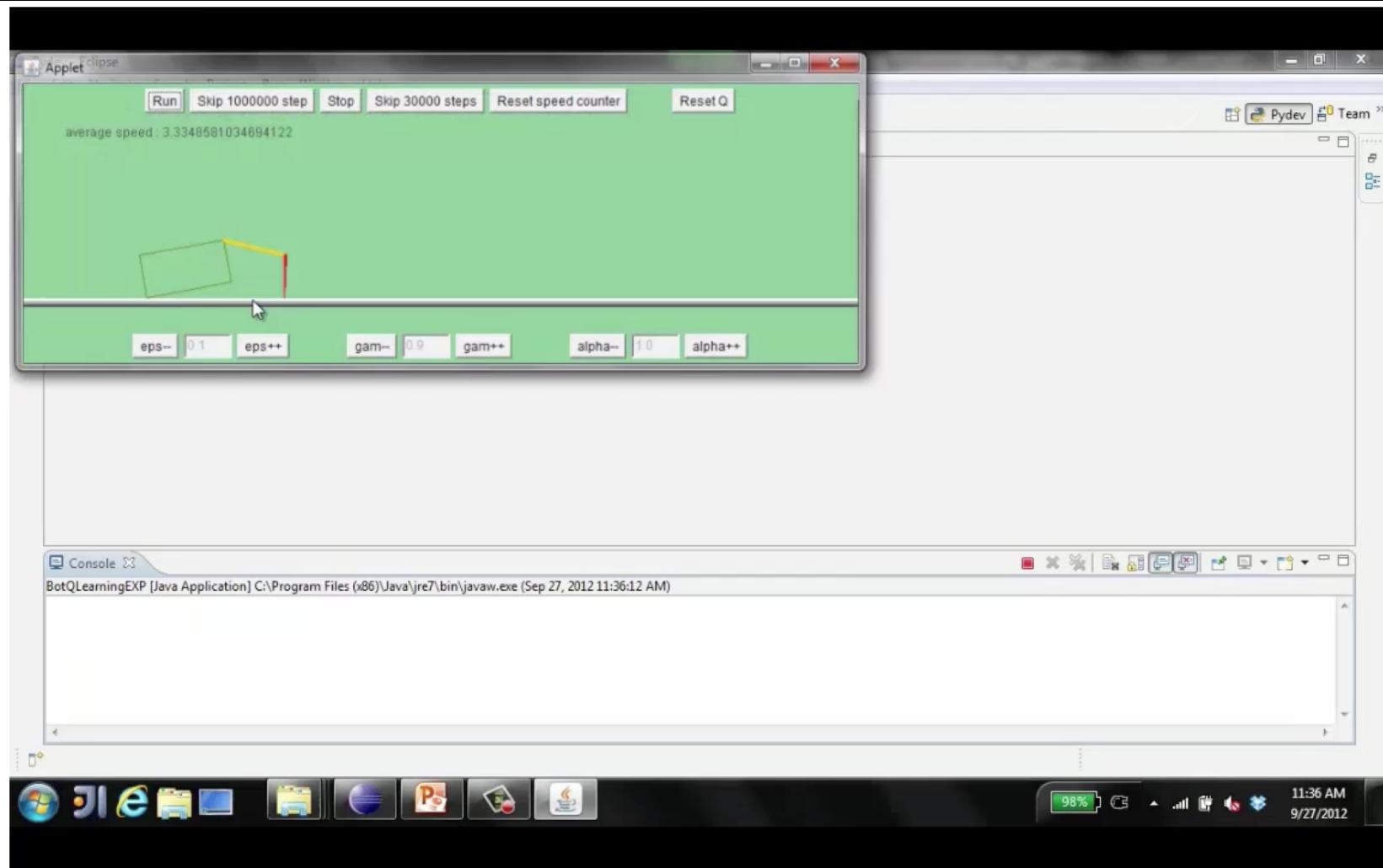
- ورودی: تخمین ارزش حالت s و تعداد دفعاتی که بررسی شده n
- خروجی: احتمال بررسی دوباره

$$f(u, n) = u + k/n$$

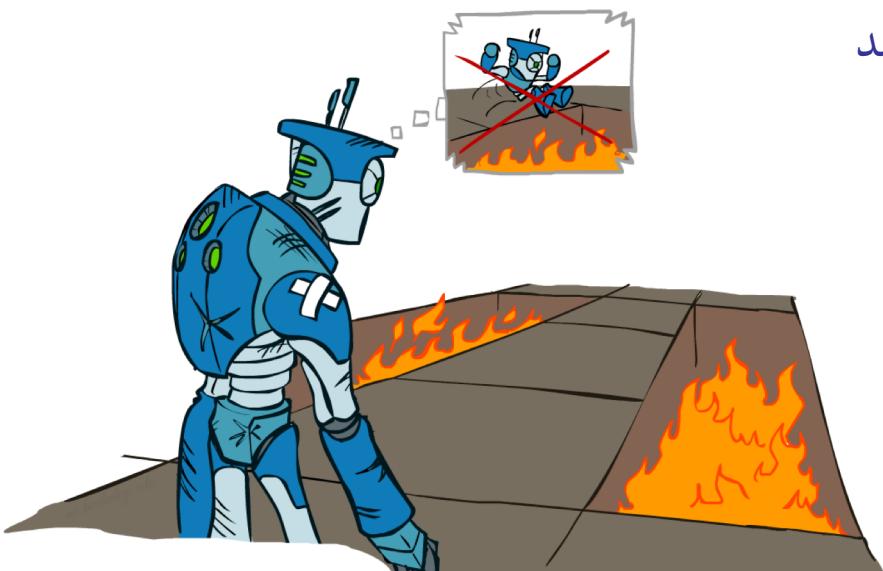
Regular Q-Update:
$$Q(s, a) \leftarrow_{\alpha} R(s, a, s') + \gamma \max_{a'} Q(s', a')$$

Modified Q-Update:
$$Q(s, a) \leftarrow_{\alpha} R(s, a, s') + \gamma \max_{a'} f(Q(s', a'), N(s', a'))$$

مثال: Exploration Function – Crawler



«افسوس» Regret !



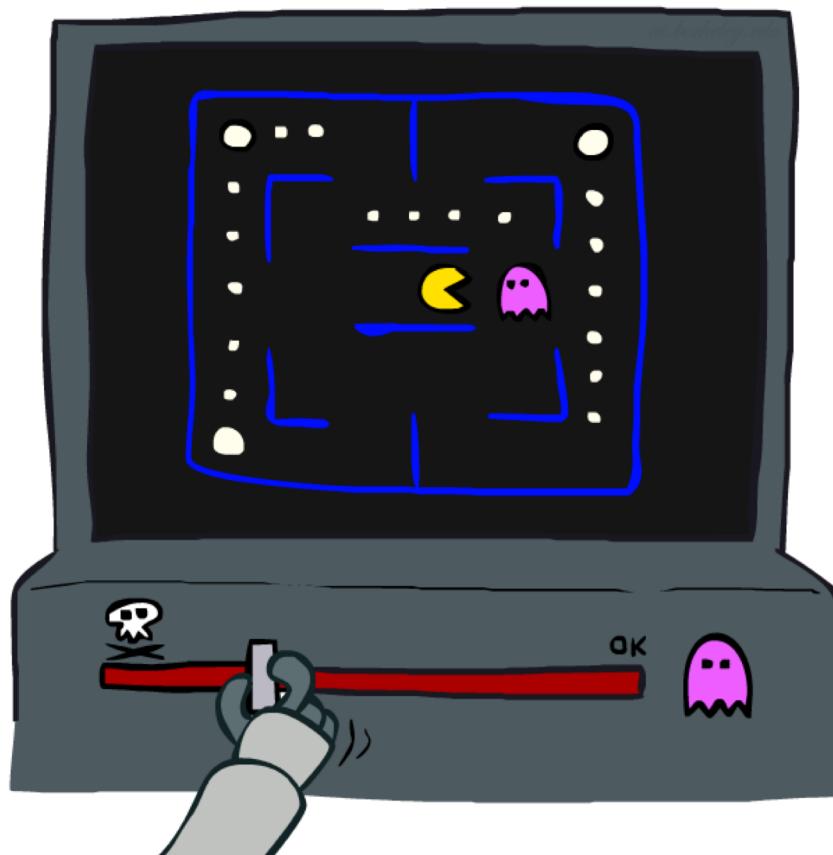
- عامل RL در راه یادگیری اشتباهات زیادی مرتکب خواهد شد

- افسوس مجموع هزینه‌هایی سنت که عامل برای اشتباهاتش داده است.
به عبارت دیگر: میزان پاداش‌هایی که می‌توانست به دست آورد ولی نیاورد!

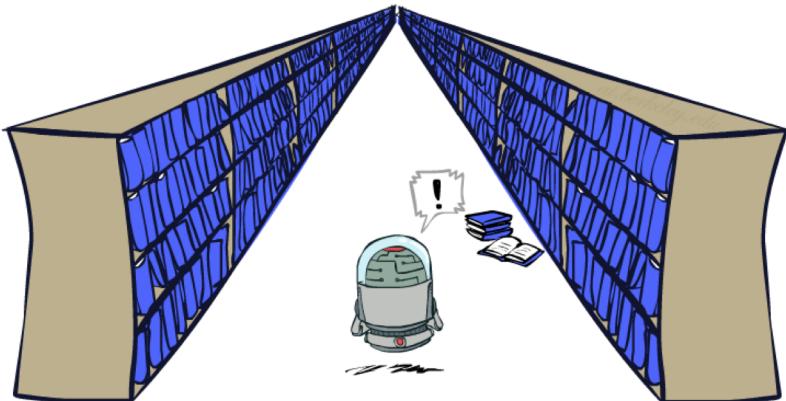
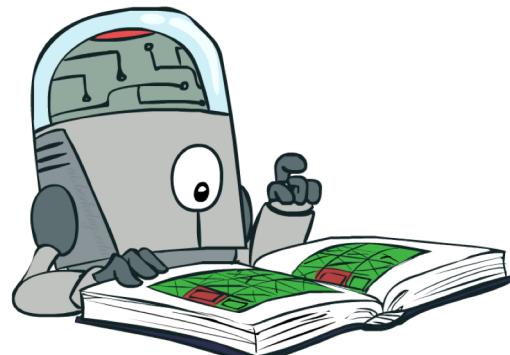
- کاستن مقدار افسوس چیزی فراتر از بهینه بودن راهبرد است و نیازمند
این است که به صورت بهینه به راهبرد بهینه برسیم!

- کشف تصادفی و توابع اکتشاف هر دو راهبرد بهینه را می‌دهند،
ولی رویکرد تصادفی افسوس بیشتری را در بر خواهد داشت

تقریبی Q-Learning



تعمیم دادن بین حالات



تا اینجا یک جدول داشتیم که $Q\text{-value}$ را برای تمامی حالات داشت ولی در مسائل واقعی، نمی‌توانیم برای هر تک حالت یادگیری انجام دهیم!

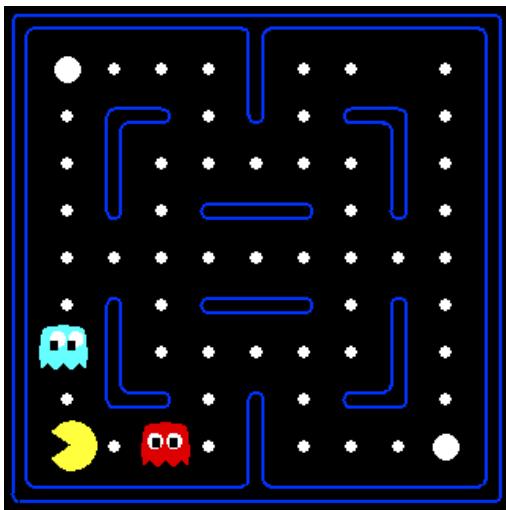
- تعداد بسیار زیادی حالت خواهیم داشت
- ذخیره کردن $Q\text{-value}$ برای تمامی این حالات نیازمند حافظه‌ی زیادی است

بنابراین، نیاز به تعمیم دادن داریم:

- تعداد کمی از حالات را تجربه کن و یاد بگیر
- این تجربه را به حالات مشابه تعمیم بده

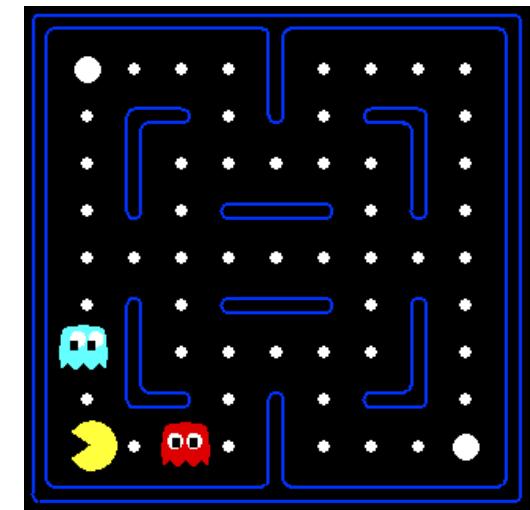
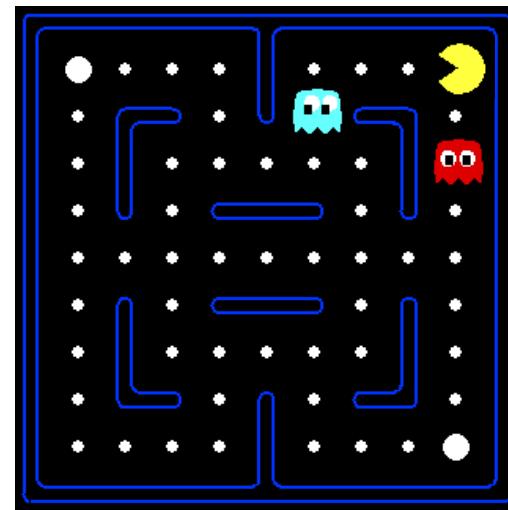
مثال پکمن

و حتی این

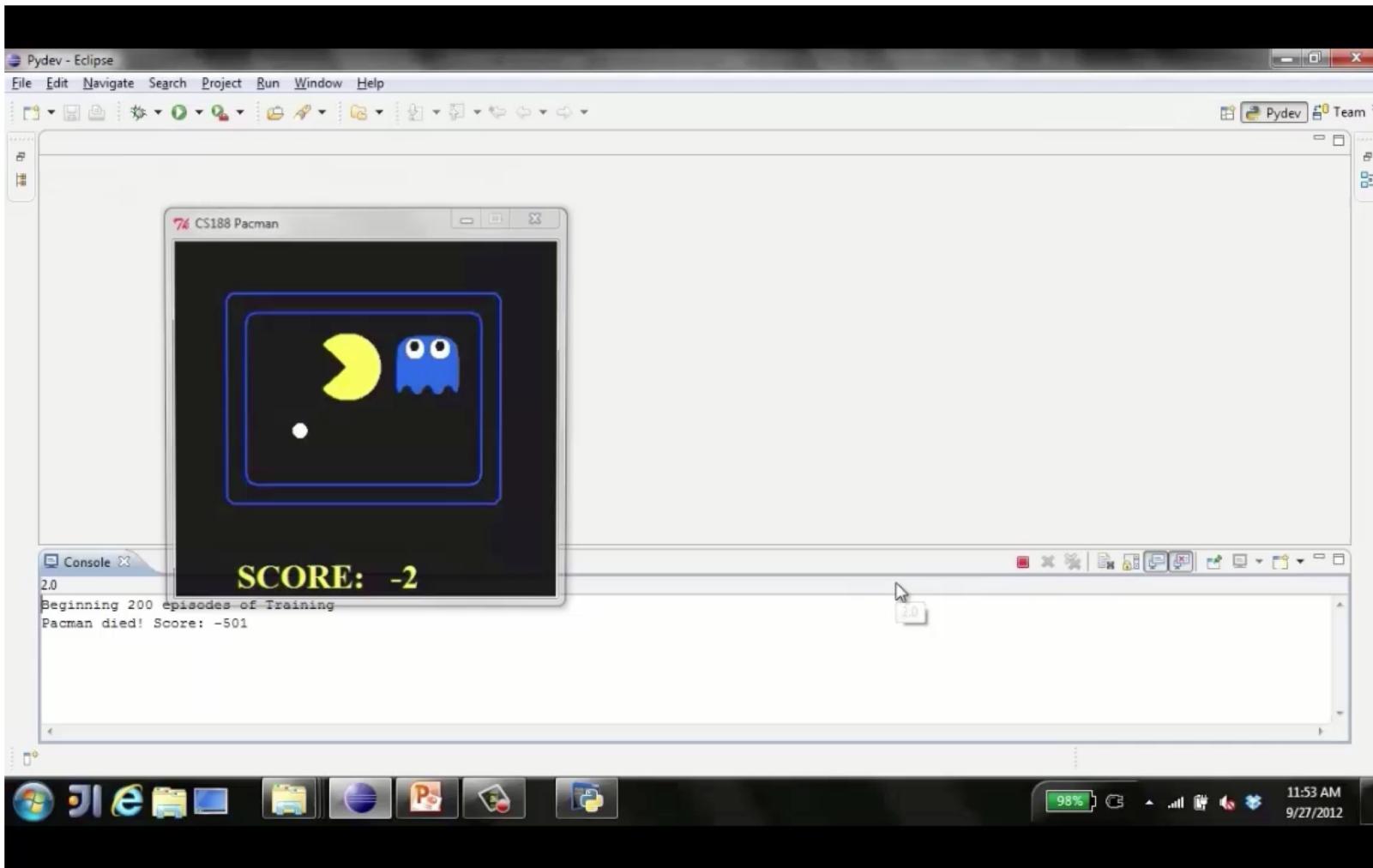


با الگوریتم Q-learning
که تا اینجا داشته ایم، هیچ
ایده‌ای برای این حالت
مشابه نداریم!

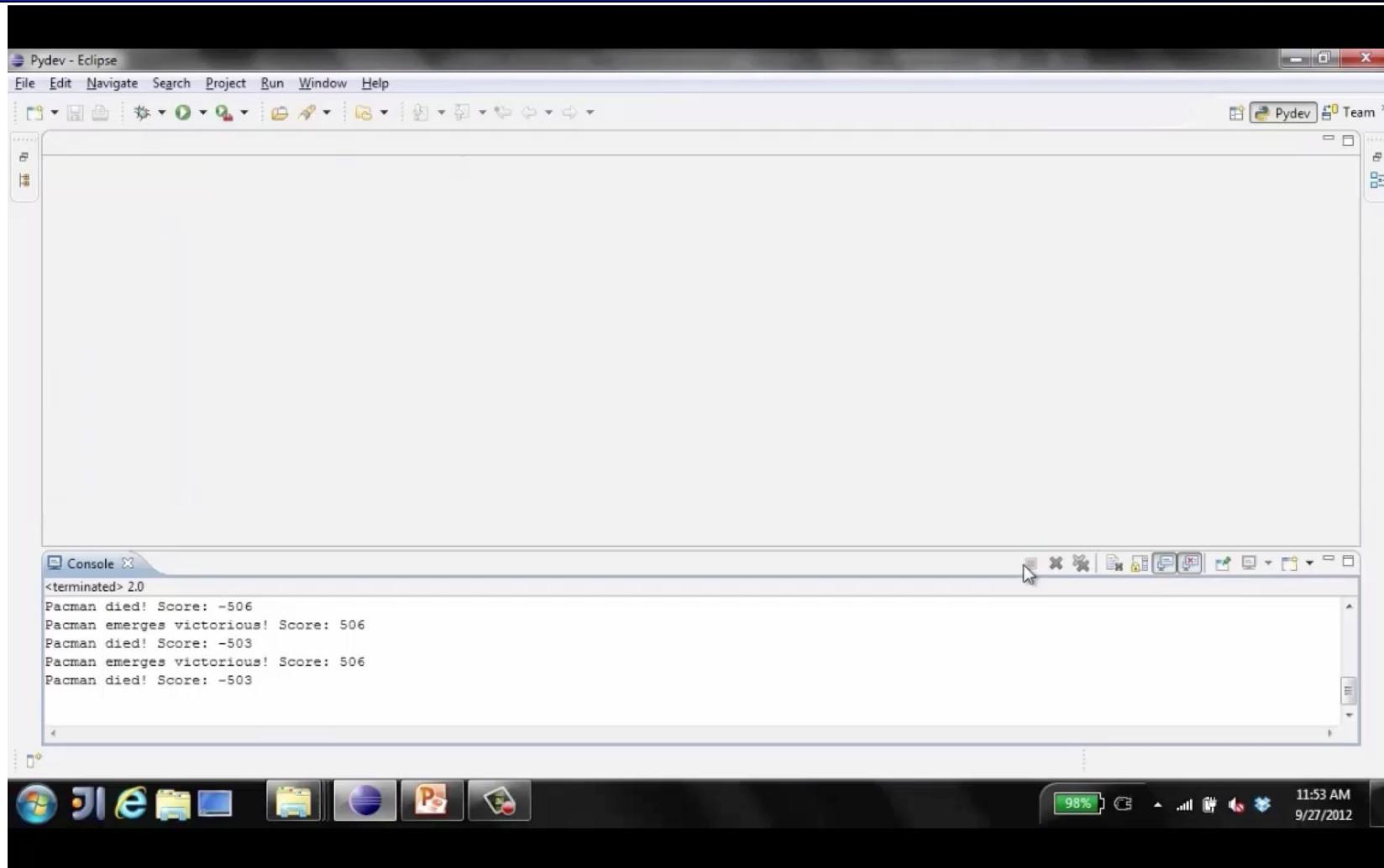
این حالت را تجربه کردیم
(که بد است)



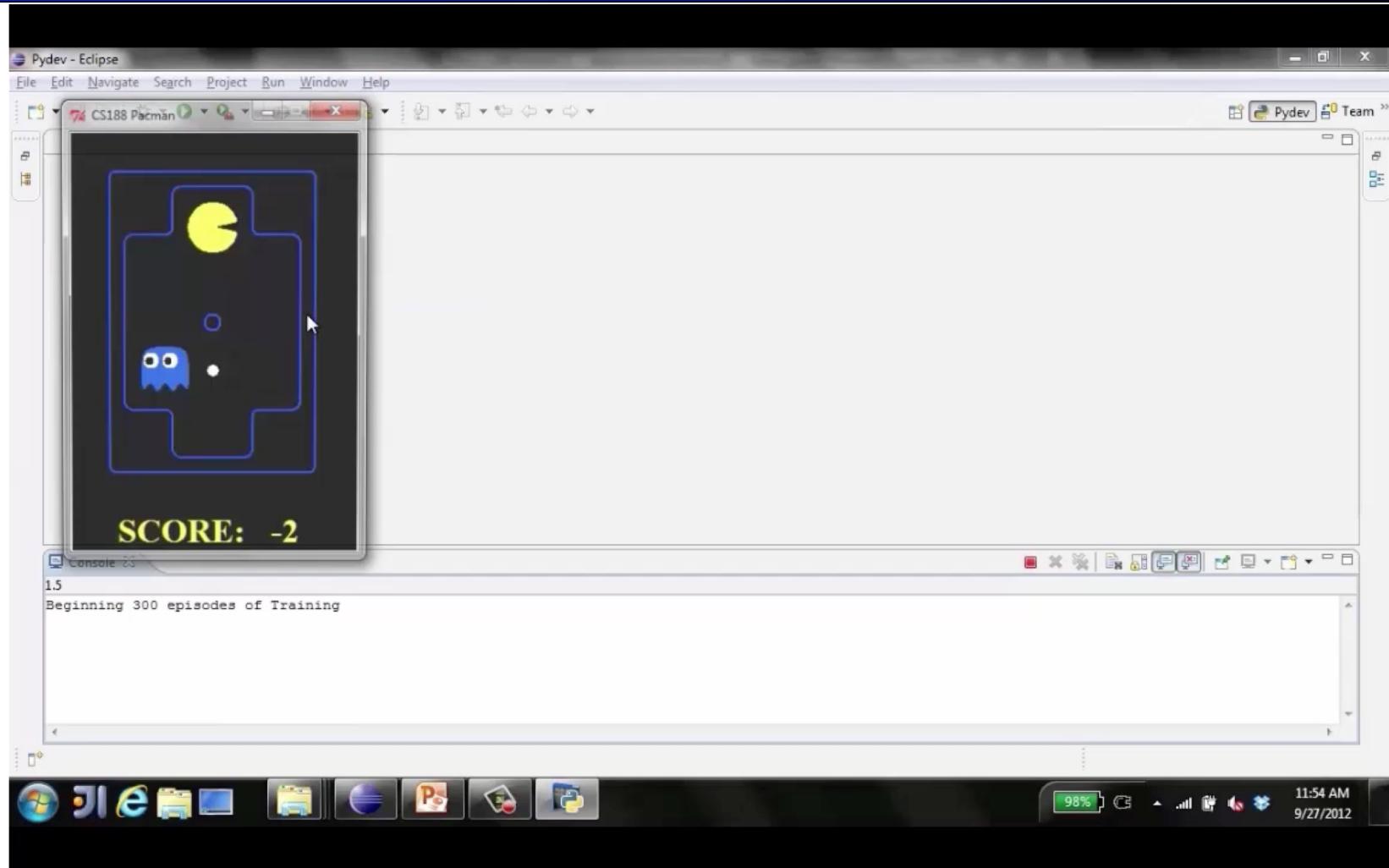
پکمن Q-Learning



پکمن Q-Learning



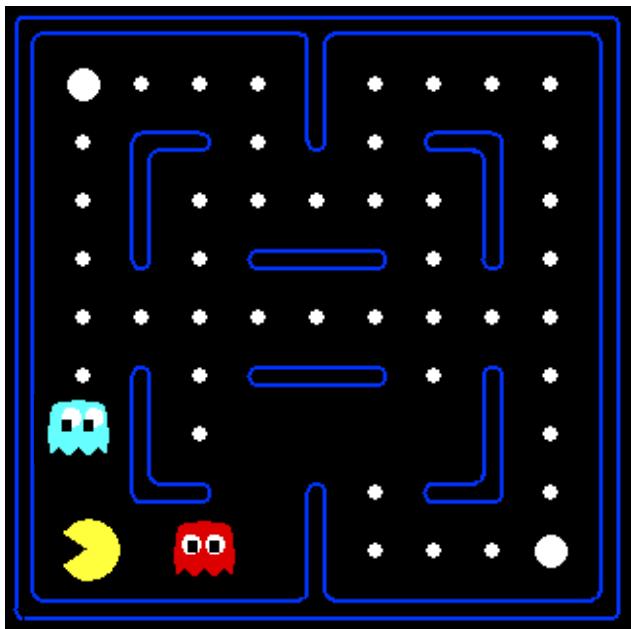
پکمن Q-Learning



نمایش حالات با استفاده از ویژگی‌ها

راه حل: هر حالت را به صورت یک بردار از ویژگی‌ها (**features**) در نظر بگیر

هر ویژگی در حقیقت تابعی است که یک جنبه‌ی مهم از حالت را به صورت کمی (مقدار یا ۰-۱) بیان می‌کند



- مثال:
- فاصله با روح
- فاصله با نقطه
- تعداد روح‌ها
- آیا پکمن در راه را دارد؟
- ...
- همچنین، می‌توانیم (s, a) را با بردار ویژگی نشان دهیم.
- مثال: این action پکمن را به هدف نزدیک تر می‌کند؟

اهمیت ویژگی‌ها

مثال: تخمین قیمت مسکن در تهران

	آدرس	قیمت کل	قیمت واحد	سن بنا	طبقه	زیربنا
220	کامرانیه بن بست دنج	4,000,000,000	20,000,000	2	0	
170	اقدبیه صاحبقرانیه سپند	4,200,000,000	25,000,000	1	0	
170	امیرآباد بالای خیابان ۱۵ بهترین دسترسی	2,260,000,000	13,300,000	2	0	
204	فرمانیه اندرزگو خ سلیمانی شمالی	5,100,000,000	25,000,000	5	0	
230	کردستان بلوار جانبازان	3,750,000,000	16,500,000	4	0	
191	یوسف آباد خیابان اسد آبادی	3,151,000,000	16,500,000	3	0	
54	پیروزی خ پورعبدی خ تاجری	405,000,000	7,500,000	4	3	
145	شهروردی شمالی نرسیده به ک مهاجر	2,610,000,000	18,000,000	1	30	
72	شهران جنوبی خ مخابرات خ رز	828,000,000	11,500,000	1	4	
126	پیروزی چهارصدستگاه خ دریاباری	650,000,000	5,158,000	4	18	
50	هاشمی نرسیده به یادگارکوی	310,000,000	6,200,000	3	4	
81	پیروزی خ شکوفه	450,000,000	5,555,000	2	15	
99	تهران پارس خ عادل	1,039,500,000	10,500,000	5	6	
100	وحیدیه خ نیشان	470,000,000	4,700,000	3	15	
120	دبستان ک شیخ شعبانی	1,140,000,000	9,500,000	3	2	
121	پیروزی بلوار ابوذر خ بوستان	1,028,500,000	8,500,000	5	0	
132	ظفر خ فرید افشار بلوار آرش شرقی	2,112,000,000	16,000,000	5	13	
72	پیروزی خ دهم فروردین	360,000,000	5,000,000	4	9	
107	شریعتی بالاتر از مطهری	1,123,500,000	10,500,000	5	12	
45	نامجو سلمان فارسی کوچه زنجانی	225,000,000	5,000,000	3	14	

توابع ارزش خطی

- ارزش یک حالت را می‌توان از مجموع وزن‌دار ویژگی‌ها بدست آورد:

$$V(s) = w_1 f_1(s) + w_2 f_2(s) + \dots + w_n f_n(s)$$

$$Q(s, a) = w_1 f_1(s, a) + w_2 f_2(s, a) + \dots + w_n f_n(s, a)$$

- خوبی این روش این است که تجربه‌ی عامل در تعدادی مقدار کمی ذخیره می‌شود
- مثال: روح خطرناک است، همیشه از دستش فرار کن

- مشکل: اگر خوب طراحی نشوند، ممکن است دو حالت شباخت زیادی داشته باشند ولی ارزش‌های متفاوتی داشته باشند

- مثال: ویژگی بگوید دو روح نزدیک پکمن هستند، ولی نگوید که آیا پکمن را محاصره کرده‌اند یا در یک طرفش هستند

تقریبی Q-Learning

$$Q(s, a) = w_1 f_1(s, a) + w_2 f_2(s, a) + \dots + w_n f_n(s, a)$$

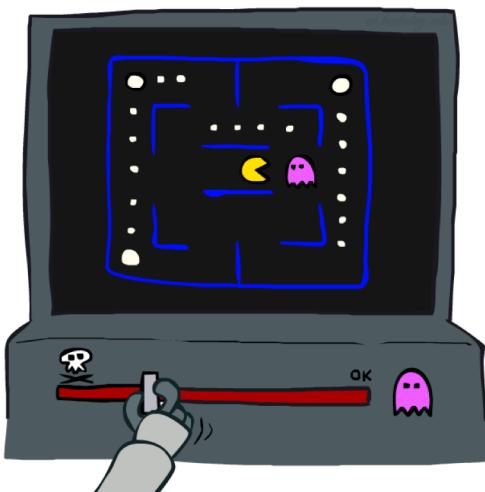
: Q با توابع خطی Q-learning ▪

$$\text{transition} = (s, a, r, s')$$

$$\text{difference} = \left[r + \gamma \max_{a'} Q(s', a') \right] - Q(s, a)$$

$$Q(s, a) \leftarrow Q(s, a) + \alpha \text{ [difference]}$$

$$w_i \leftarrow w_i + \alpha \text{ [difference]} f_i(s, a) \quad \text{های تقریبی Q}$$

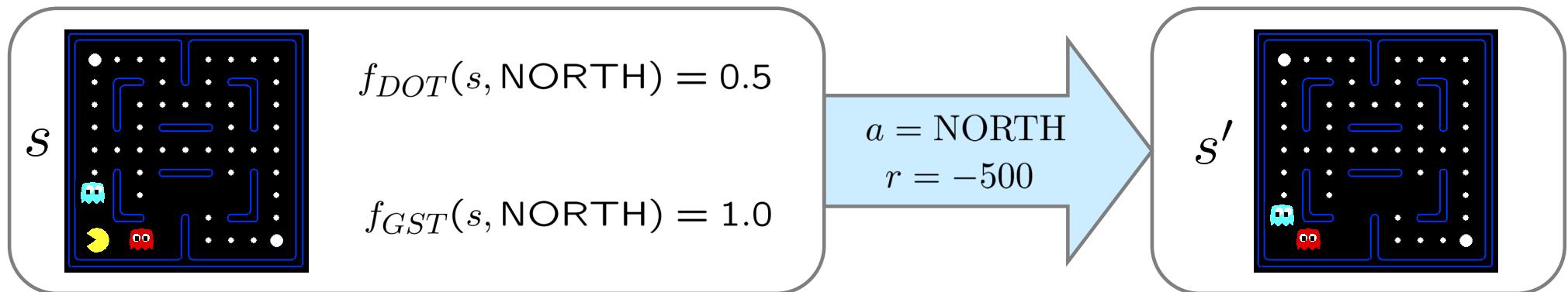


- تخمینی که از Q ها داریم را با تنظیم وزن ویژگی ها بهتر می کنیم.
- مثال: اگر اتفاق بدی افتاد، ویژگی هایی که در این شرایط دخالت داشتند را سرزنش کن.
- با این کار، تمامی حالات مشابه نیز سرزنش می شوند.

- Online least squares

مثال: کیو-پکمن!

$$Q(s, a) = 4.0f_{DOT}(s, a) - 1.0f_{GST}(s, a)$$



$$Q(s, \text{NORTH}) = +1$$

$$Q(s', \cdot) = 0$$

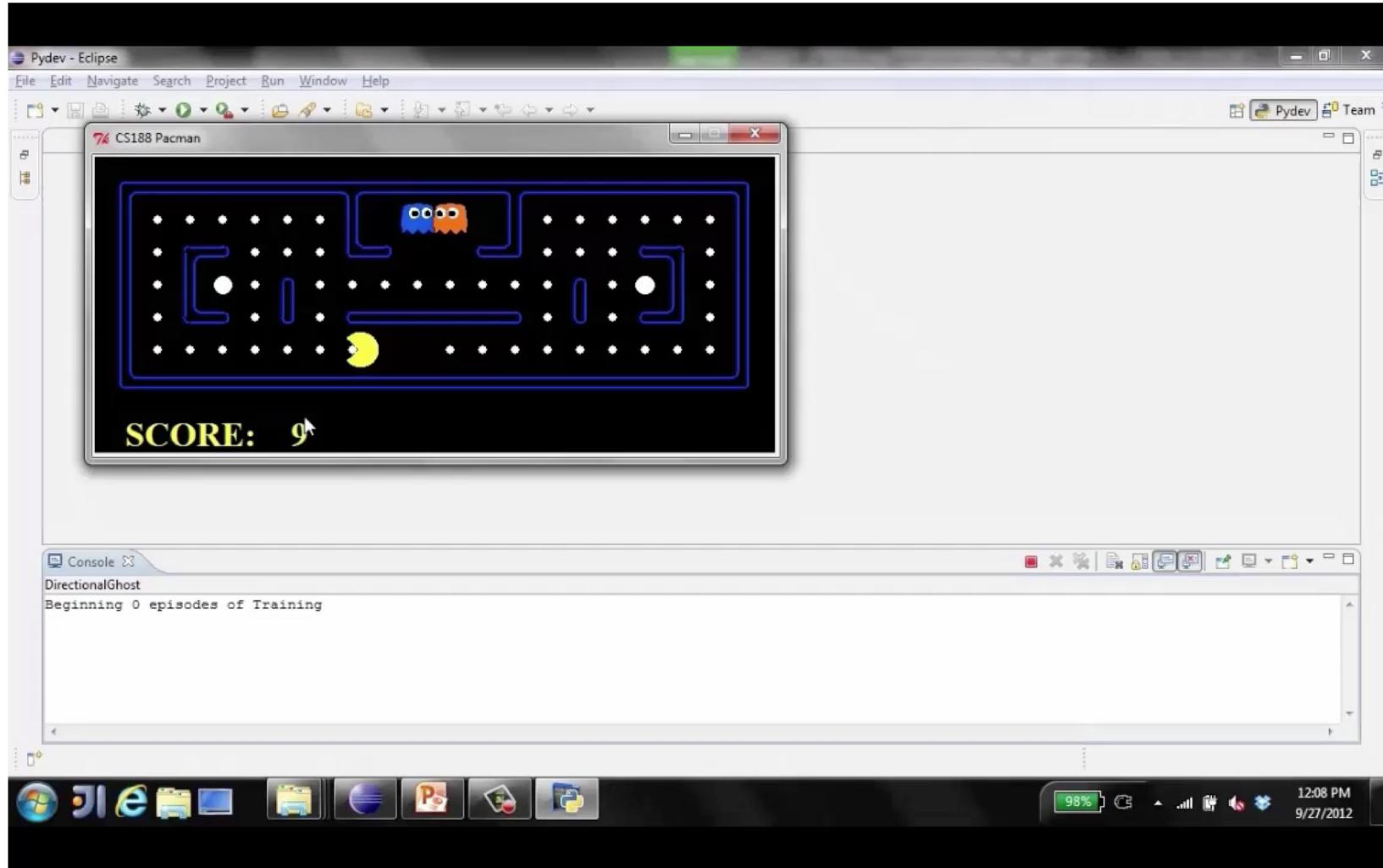
$$r + \gamma \max_{a'} Q(s', a') = -500 + 0$$

$$\text{difference} = -501$$

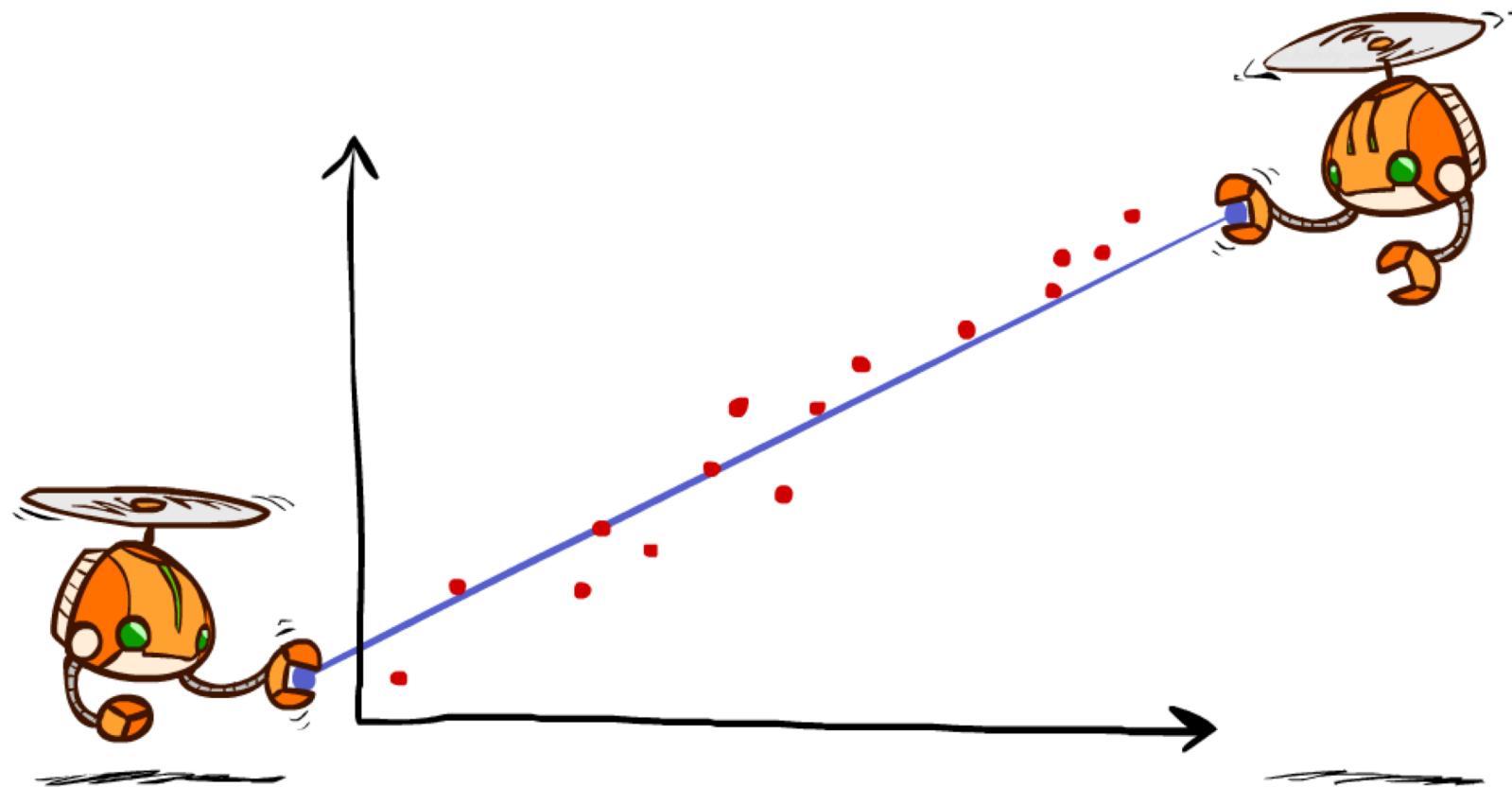
$$w_{DOT} \leftarrow 4.0 + \alpha [-501] 0.5$$
$$w_{GST} \leftarrow -1.0 + \alpha [-501] 1.0$$

$$Q(s, a) = 3.0f_{DOT}(s, a) - 3.0f_{GST}(s, a)$$

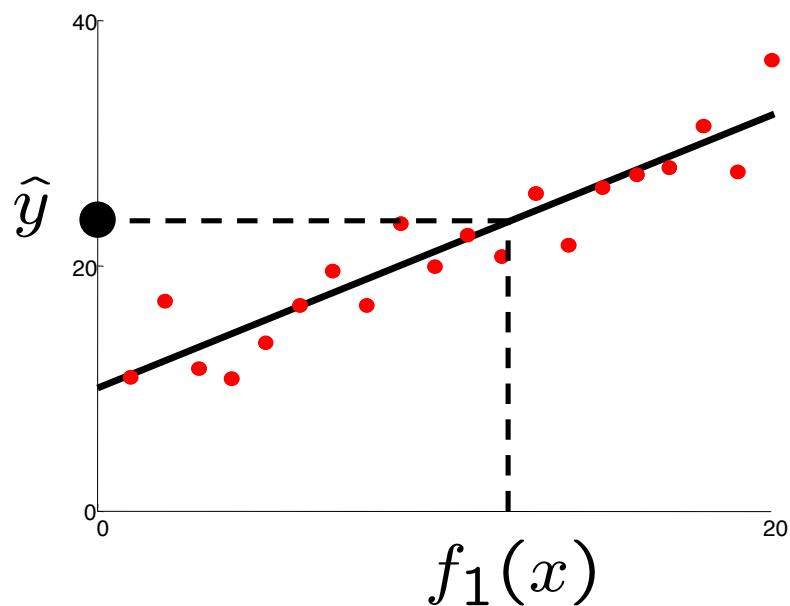
مثال: پکمن با Q-Learning تقریبی



Q-Learning and Least Squares*

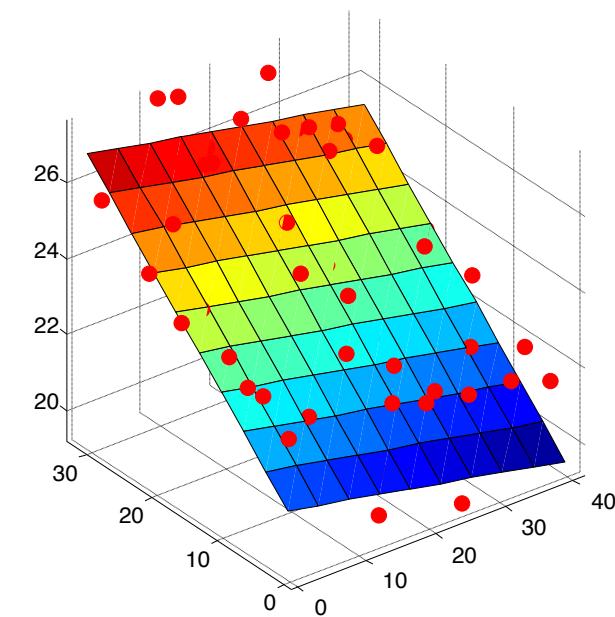


تخمين خطی: رگرسیون*



Prediction:

$$\hat{y} = w_0 + w_1 f_1(x)$$

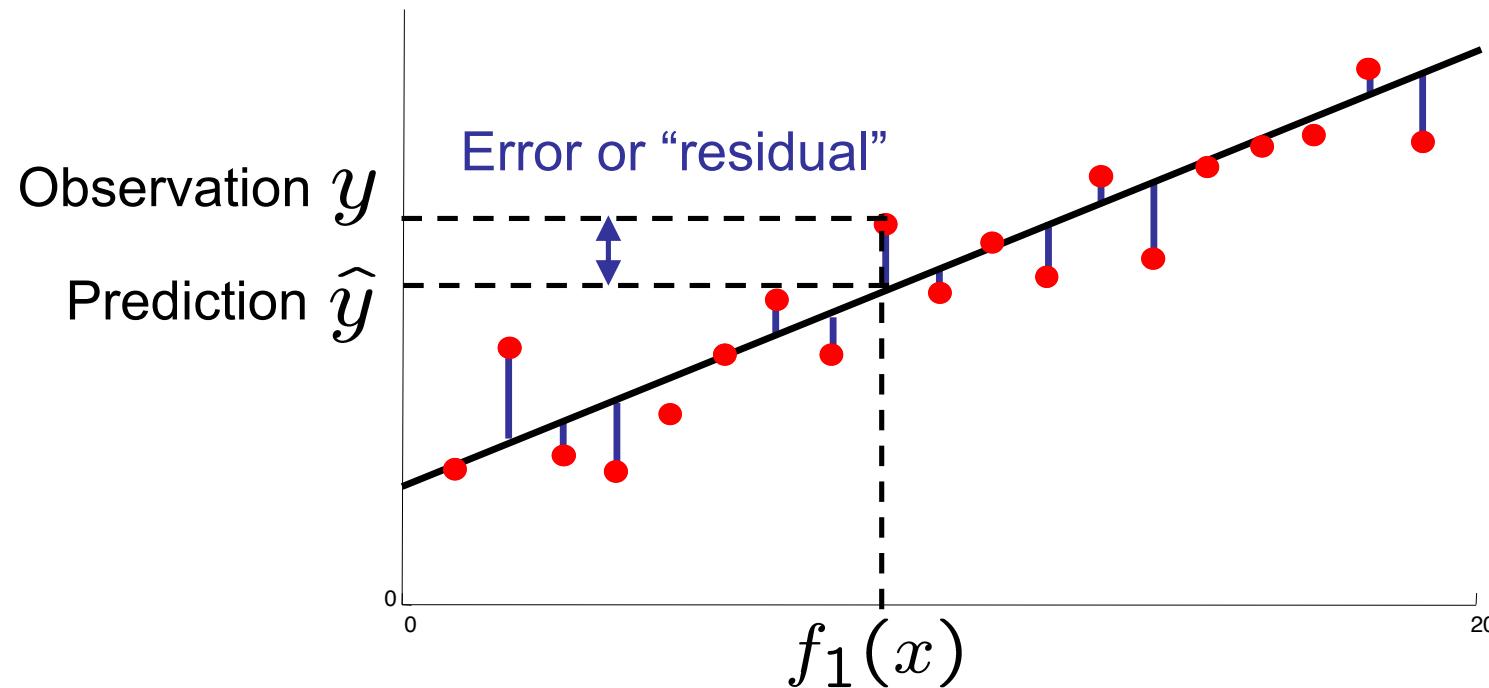


Prediction:

$$\hat{y}_i = w_0 + w_1 f_1(x) + w_2 f_2(x)$$

Least Squares: بهینه سازی*

$$\text{total error} = \sum_i (y_i - \hat{y}_i)^2 = \sum_i \left(y_i - \sum_k w_k f_k(x_i) \right)^2$$



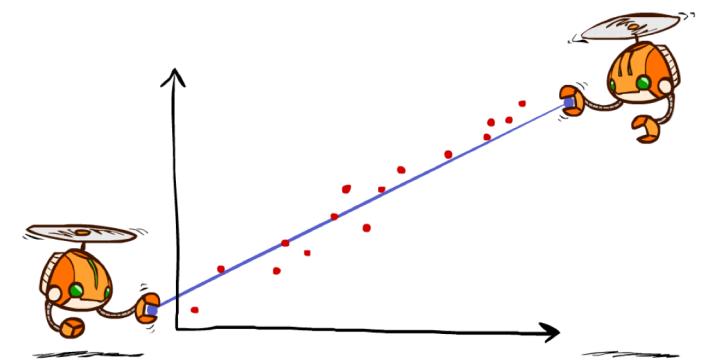
Error کمینه کردن*

با یک نقطه x و ویژگی‌های $f(x)$ و مقدار هدف y و وزنهای w :

$$\text{error}(w) = \frac{1}{2} \left(y - \sum_k w_k f_k(x) \right)^2$$

$$\frac{\partial \text{error}(w)}{\partial w_m} = - \left(y - \sum_k w_k f_k(x) \right) f_m(x)$$

$$w_m \leftarrow w_m + \alpha \left(y - \sum_k w_k f_k(x) \right) f_m(x)$$



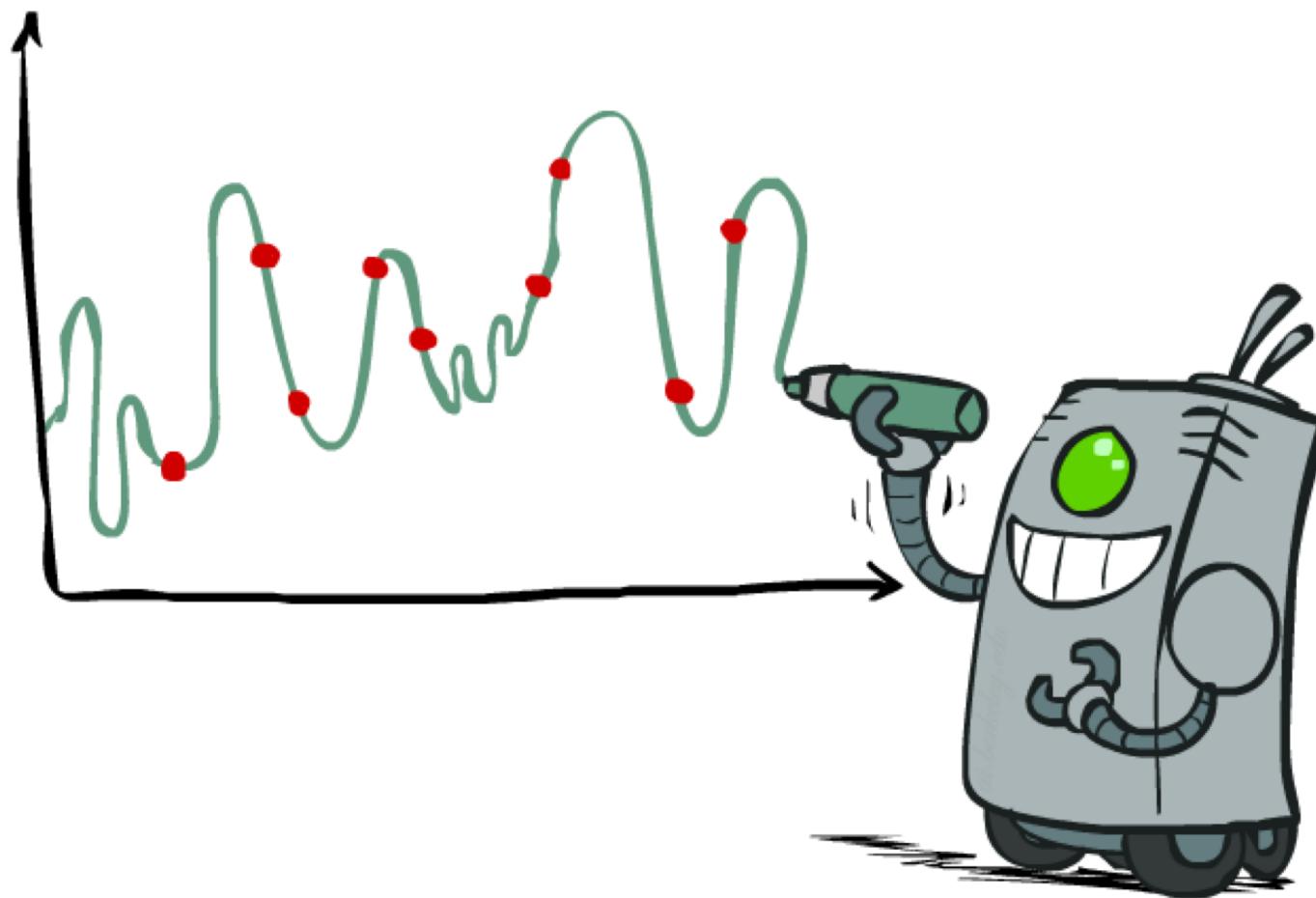
Approximate q update:

$$w_m \leftarrow w_m + \alpha \left[r + \gamma \max_a Q(s', a') - Q(s, a) \right] f_m(s, a)$$

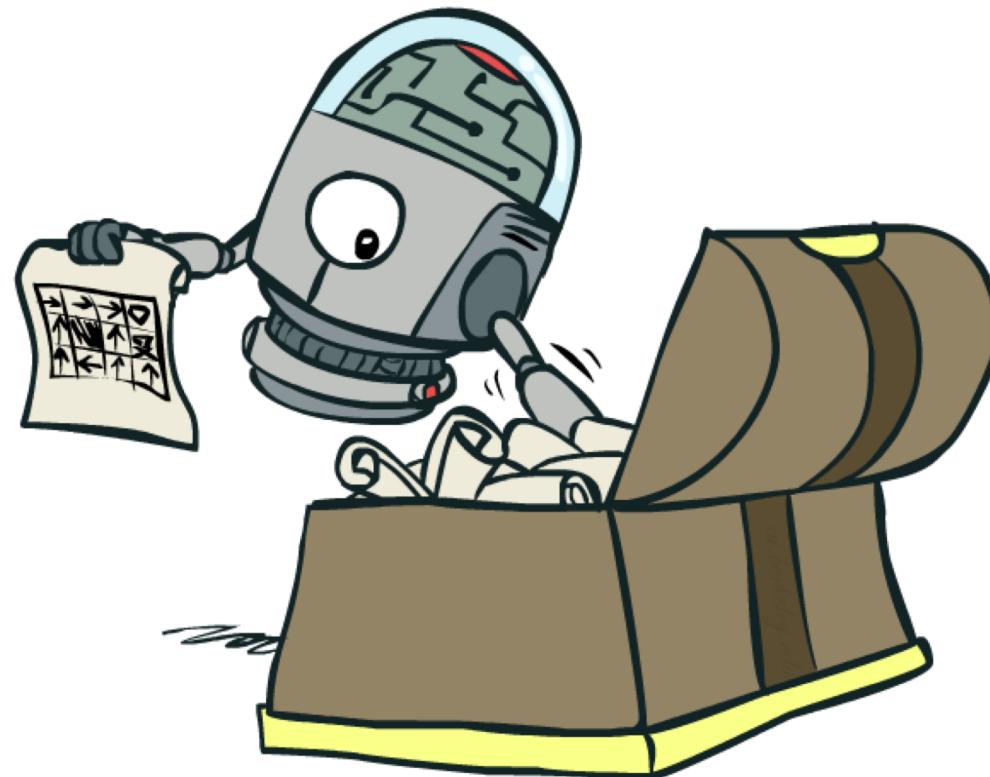
“target”

“prediction”

Overfitting*



جستجوی راهبرد



جستجوی راهبرد

لزوماً راهبردهای موفقی که مبتنی بر ویژگی هستند، آنها بی نیستند که مقادیر Q یا
۷ را خوب تخمین می‌زنند

راهبردی را پیدا کن که پاداش را بیشینه می‌کند و نه ارزش‌هایی که راهبرد را پیش‌بینی می‌کند

جستجوی راهبرد: با یک راه حل عادی (مانند Q-learning) راهبرد را بیاب،
سپس تخمینی که از وزن ویژگی‌ها داریم را با روش‌های دیگر بهبود بده.

جستجوی راهبرد



به پایان آمد این دفتر...



جستجو و نقشه‌کشی

- روش‌های هوش مصنوعی برای حل مسائل زیر را یاد گرفتیم:
 - جستجو
 - مسائل ارضای محدودیت
 - بازی‌ها
 - مسائل تصمیم‌گیری مارکوفی
 - یادگیری تقویتی