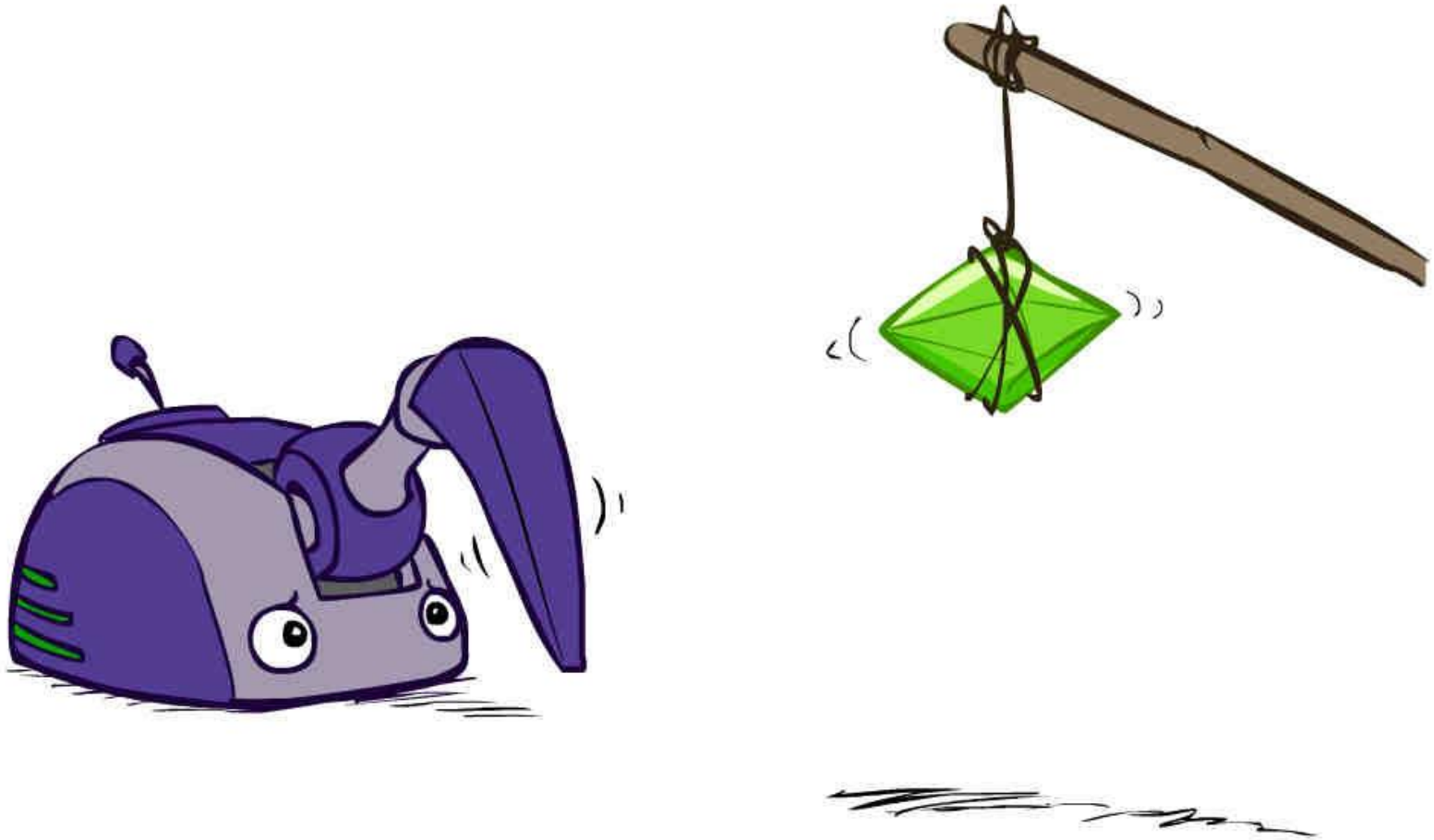
یادگیری تقویتی

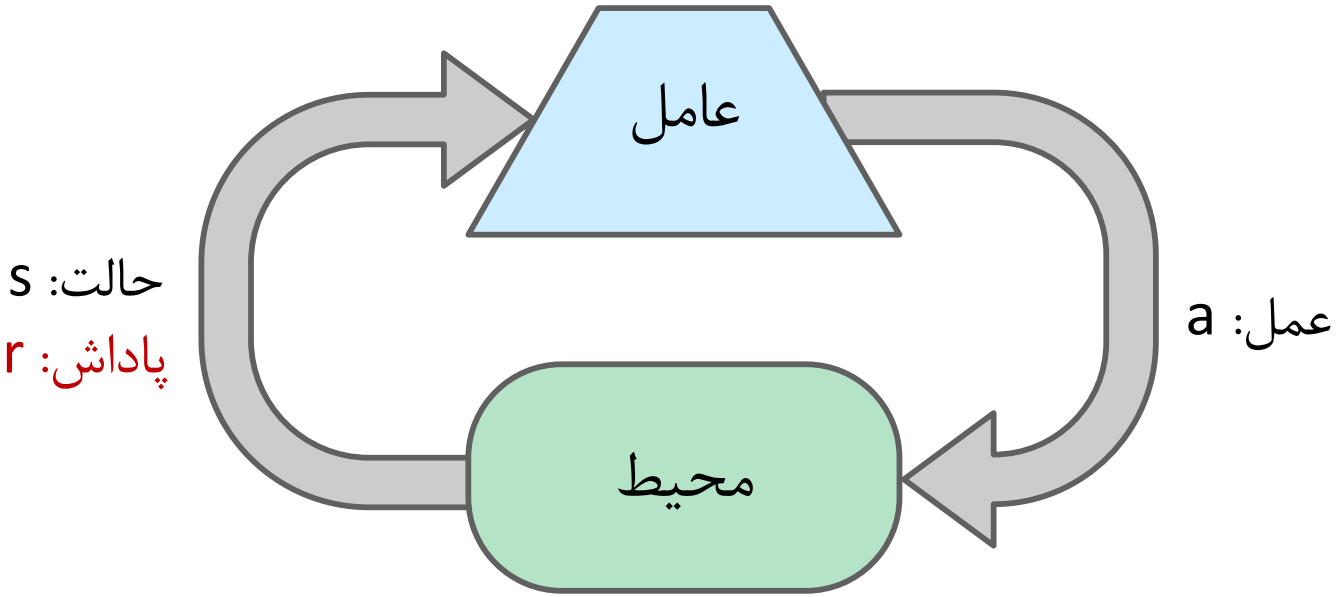
سید ناصر رضوی n.razavi@tabrizu.ac.ir

۱۳۹۵

یادگیری تقویتی



یادگیری تقویتی



- ایده‌ی اصلی.
  - دریافت بازخورد از محیط به شکل پاداش‌ها.
  - سودمندی عامل به وسیله‌ی تابع پاداش تعریف می‌شود.
  - عامل باید به گونه‌ای عمل کند که سودمندی مورد انتظار را بیشینه سازد.
  - یادگیری بر مبنای نمونه‌های مشاهده شده از پیامدها است!

مثال: یادگیری نحوه راه رفتن

پس از یادگیری

در طول یادگیری

پیش از یادگیری

[کوهل و استون؛ ۲۰۰۴]

مثال: یادگیری نحوه‌ی راه رفتن

پیش از یادگیری

مثال: یادگیری نحوه‌ی راه رفتن

در طول یادگیری

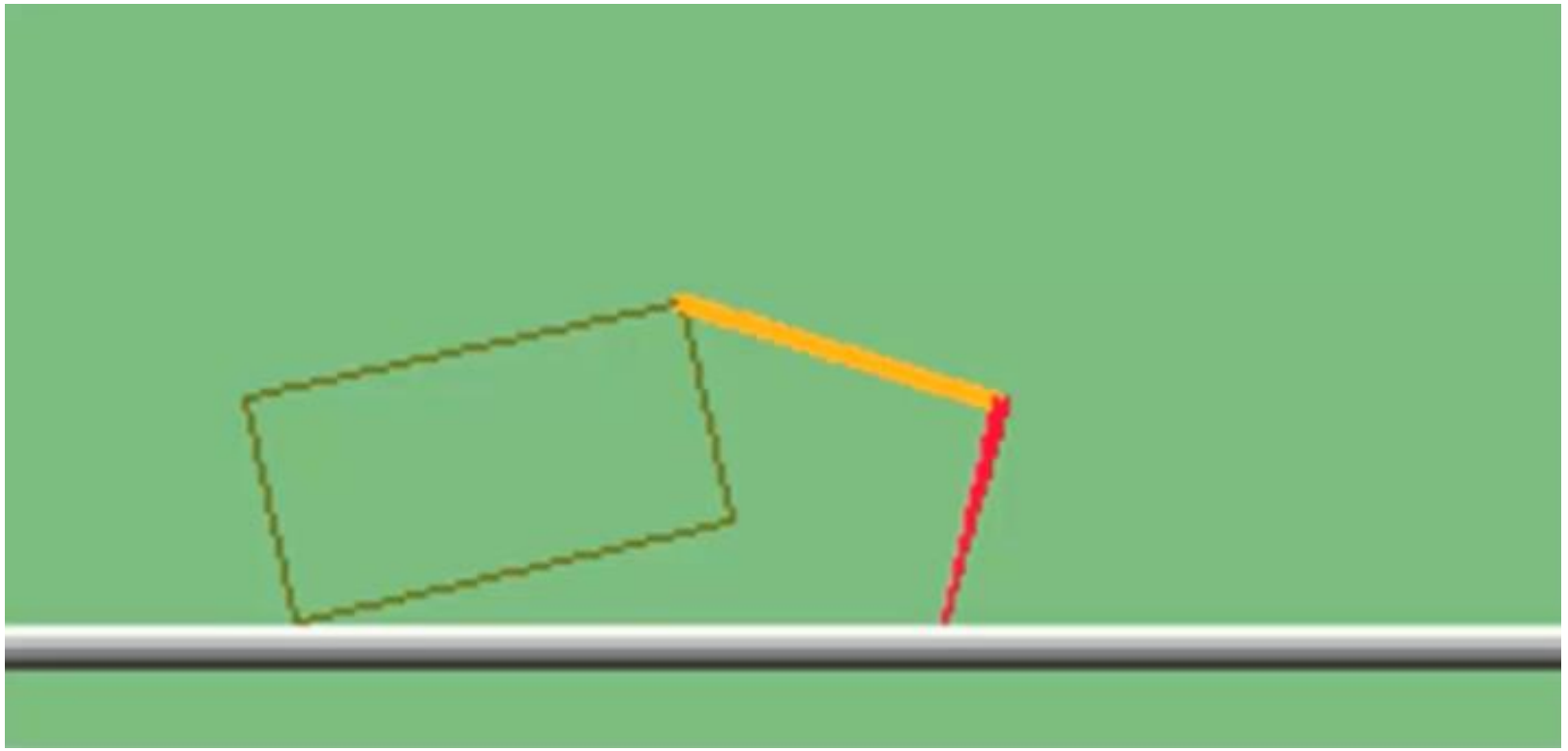
مثال: یادگیری نحوه‌ی راه رفتن

پس از یادگیری

مثال: یادگیری نحوه‌ی راه رفتن مار

[اندرو ان‌جی؛ ۲۰۰۴]

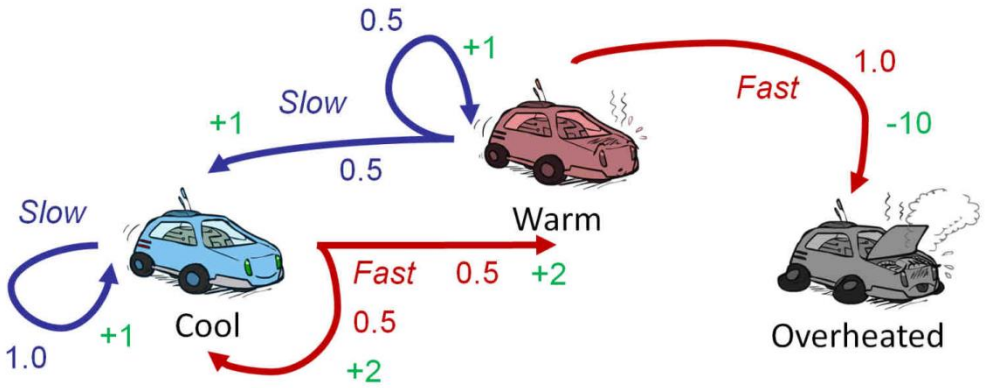
مثال: روبات خزنده



یادگیری تقویتی

- هنوز یک فرآیند تصمیم مارکوف داریم:
  - یک مجموعه از حالت‌ها $s \in S$
  - یک مجموعه از اعمال $a \in A$
  - یک مدل $T(s,a,s')$
  - یک تابع پاداش $R(s,a,s')$

- هنوز به دنبال یک سیاست $\pi(s)$ هستیم.



یادگیری تقویتی

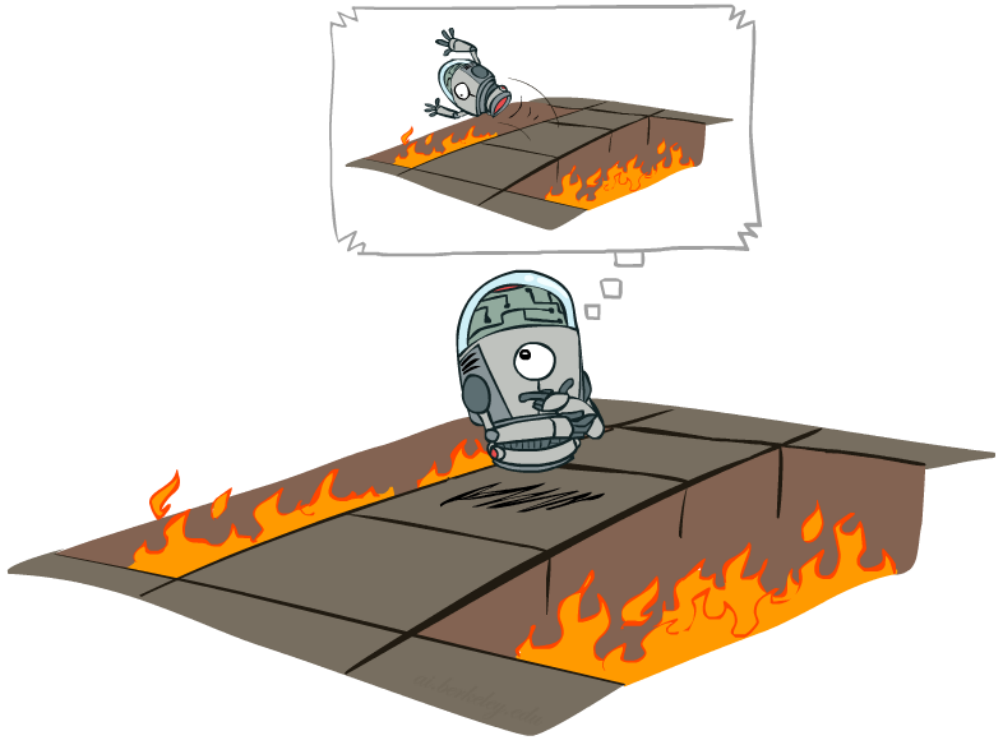
- هنوز یک فرآیند تصمیم مارکوف داریم:
  - یک مجموعه از حالت‌ها $s \in S$
  - یک مجموعه از اعمال $a \in A$
  - یک مدل $T(s,a,s')$
  - یک تابع پاداش $R(s,a,s')$

- هنوز به دنبال یک سیاست $\pi(s)$ هستیم.

- تفاوت. توابع T و R ناشناخته هستند.
  - یعنی، نمی‌دانیم کدام حالت‌ها بهتر هستند و نتیجه‌ی هر عمل چیست؟
  - برای یادگیری باید عمل‌های مختلف و حالت‌های نتیجه شده را آزمایش کنیم.

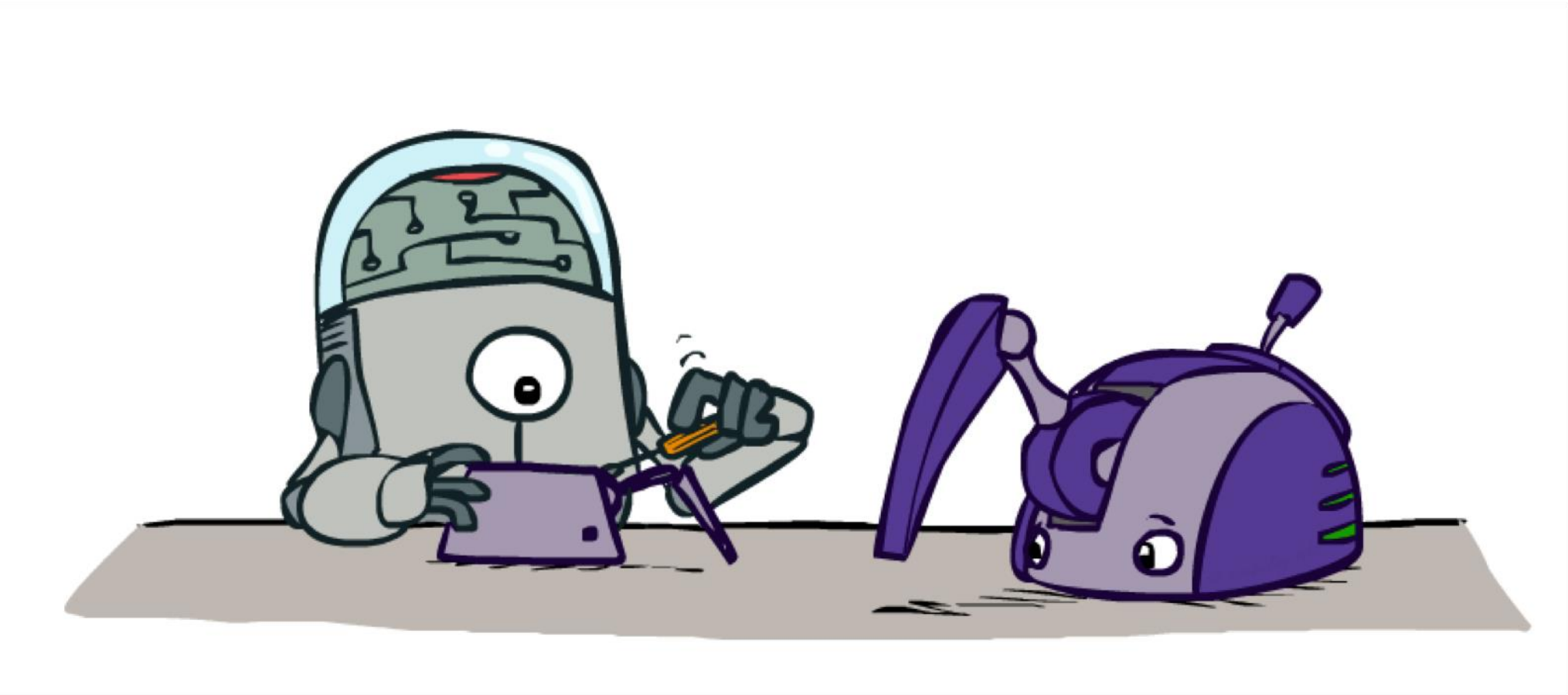
آفلاین (MDP) یا آنلاین (یادگیری تقویتی)؟

یادگیری آنلاین

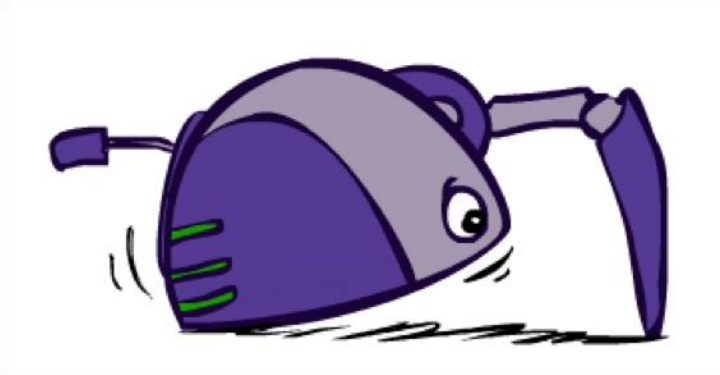


راه‌حل آفلاین

یادگیری مبتنی بر مدل



یادگیری مبتنی بر مدل



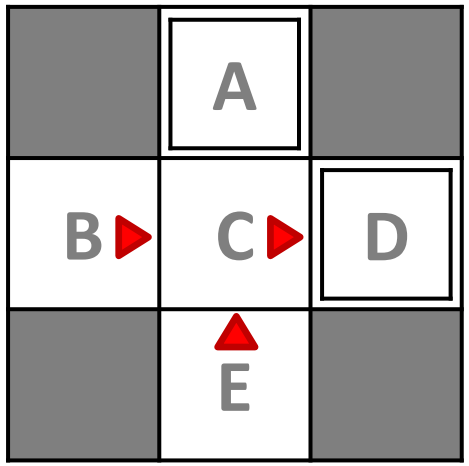
- ایده‌ی یادگیری مبتنی بر مدل.
  - یادگیری یک مدل تقریبی بر مبنای تجربیات
  - حل مدل تقریبی برای به دست آوردن مقادیر

- گام اول: یادگیری یک مدل تجربی از MDP
  - شمارش تعداد s' ها به ازای هر s و a
  - نرمال‌سازی برای به دست آوردن مقدار تقریبی $\hat{T}(s,a,s')$
  - یادگیری مقادیر $\hat{R}(s,a,s')$ با توجه به تجربه‌های (s,a,s')

- گام دوم: حل کردن MDP یاد گرفته شده.
  - مثلاً با استفاده از الگوریتم تکرار مقدار.

مثال: یادگیری مبتنی بر مدل

سیاست ورودی



فرض: $\gamma = 1$

اپیزودهای مشاهده شده (آموزش)

اپیزود ۱

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

اپیزود ۲

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

اپیزود ۳

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

اپیزود ۴

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

مدل تقریبی

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

T(B, east, C)= 1.00
T(C, east, D)= 0.75
T(C, east, A)= 0.25

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

R(B, east, C) = -1
R(C, east, D) = -1
R(D, exit, x) = +10

مثال: سن مورد انتظار دانشجویان کلاس

- هدف. محاسبه‌ی سن مورد انتظار دانشجویان کلاس هوش مصنوعی.

$P(A)$ شناخته شده

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

- بدون داشتن $P(A)$، باید نمونه‌برداری کنیم $[a_1, a_2, \ldots, a_N]$.

$P(A)$ ناشناخته: «مستقل از مدل»

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

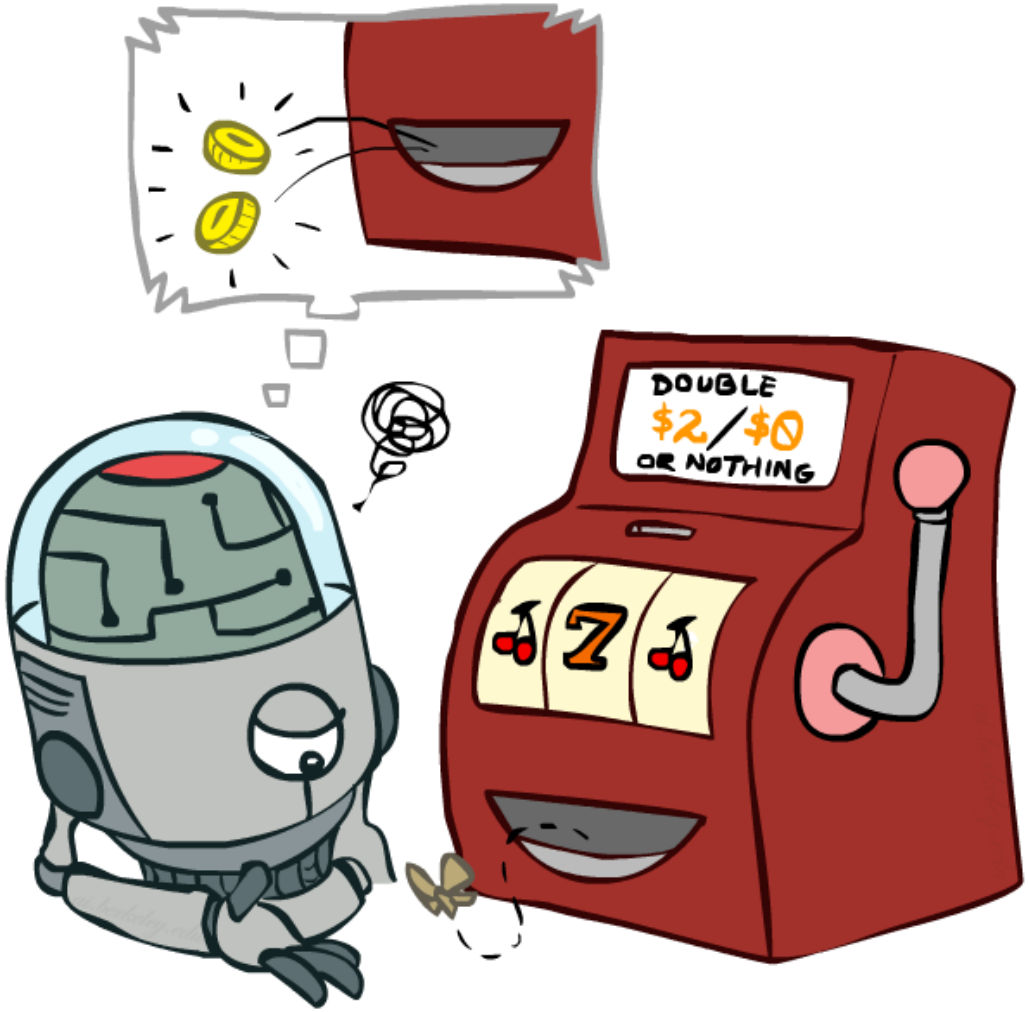
$P(A)$ ناشناخته: «مبتنی بر مدل»

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

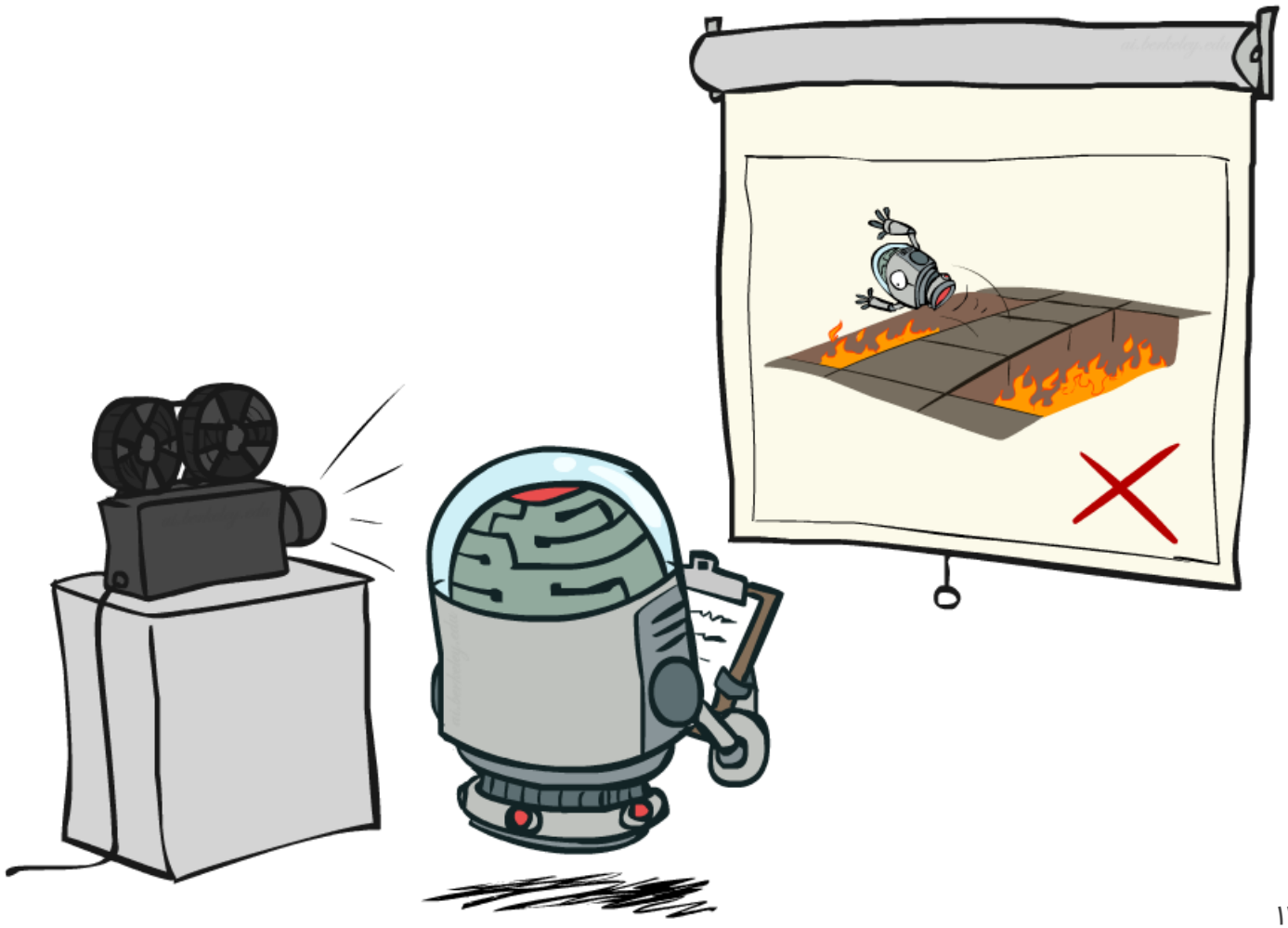
$$E[A] \approx \sum_a \hat{P}(a) \cdot a$$

سن‌های متداول‌تر در نمونه‌های بیشتری ظاهر می‌شوند.

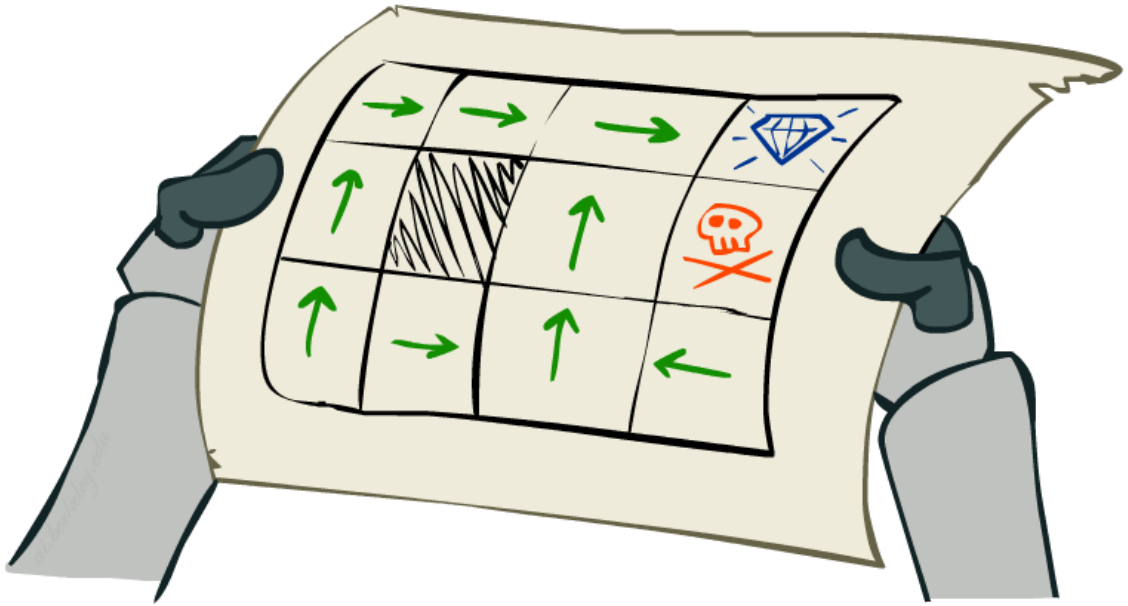
یادگیری مستقل از مدل



یادگیری تقویتی منفعل



یادگیری تقویتی منفعل



- وظیفه‌ی ساده شده. ارزیابی سیاست!
  - ورودی: یک سیاست ثابت $\pi(s)$
  - تابع تغییر حالت $T(s,a,s')$ ناشناخته است.
  - تابع پاداش $R(s,a,s')$ ناشناخته است.
  - هدف: یادگیری ارزش حالت‌ها

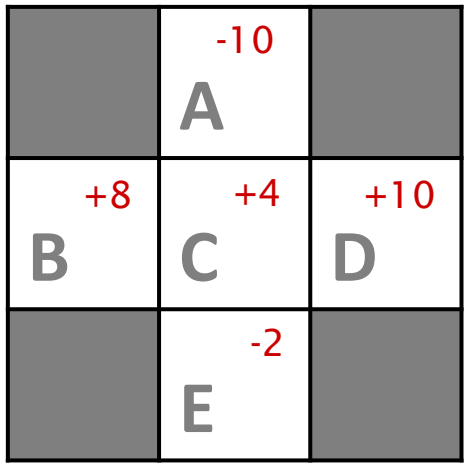
- در این مورد:
  - بر روی اعمالی که باید انجام شوند، هیچ گونه کنترلی نداریم.
  - تنها می‌توانیم سیاست ورودی را دنبال کرده و از تجارب کسب شده یاد بگیریم.
  - این برنامه‌ریزی آفلاین **نیست**! زیرا عامل واقعاً در محیط عمل می‌کند.

ارزیابی مستقیم

- هدف. محاسبه‌ی ارزش هر حالت تحت سیاست ثابت π.
- ایده. میانگین‌گیری از مقادیر نمونه‌ی مشاهده شده.
  - بر طبق π عمل کن.
  - هر بار که با یک حالت روبرو می‌شوی، محاسبه کن که مجموع (کاهش یافته) پاداش‌ها چقدر باید باشد.
  - از نمونه‌های مشاهده شده میانگین بگیر.
- این روش ارزیابی مستقیم نام دارد.

مثال: ارزیابی مستقیم

مقادیر خروجی



اپیزودهای مشاهده شده (آموزش)

اپیزود ۲

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

اپیزود ۱

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

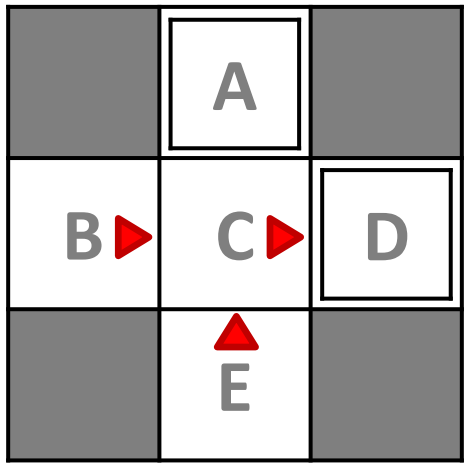
اپیزود ۴

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

اپیزود ۳

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

سیاست ورودی

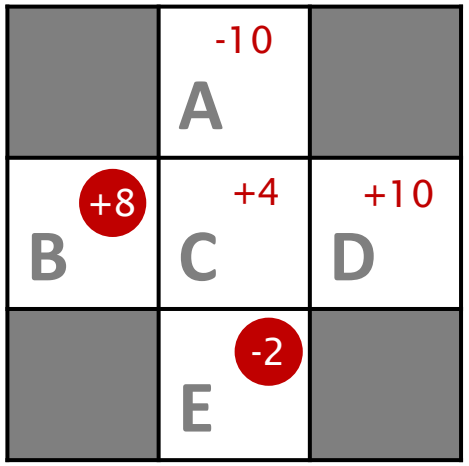


فرض: $\gamma = 1$

مزایا و معایب ارزیابی مستقیم

- مزایا.
  - درک آن ساده است.
  - نیاز به دانش در مورد T یا R ندارد.
  - درنهایت، ارزش درست هر حالت را محاسبه می‌کند.

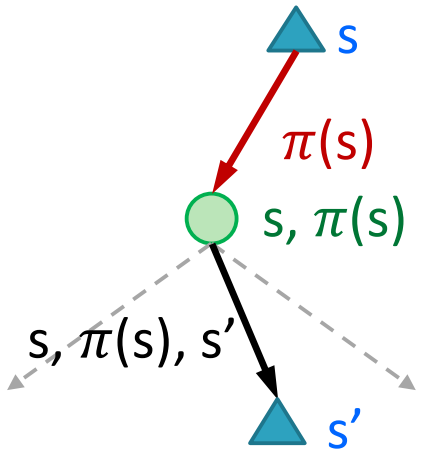
- معایب.
  - اتلاف اطلاعات به دلیل در نظر نگرفتن ارتباط میان حالت‌ها.
  - یادگیری هر حالت به صورت جداگانه.
  - و در نتیجه، نیاز به زمان زیاد برای یادگیری.



با این که حالت‌های B و E هر دو به یک حالت یکسان منجر می‌شوند، اما ارزش محاسبه شده برای آنها بسیار متفاوت است!!!

چرا از روش ارزیابی سیاست استفاده نکنیم؟

- شکل ساده شده‌ی معادله بلمن، ارزش V را برای یک سیاست خاص محاسبه می‌کند.



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

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

  - این روش ارتباط میان حالت‌ها را به طور کامل در نظر می‌گیرد.
  - اما متأسفانه برای استفاده از آن باید توابع T و R را بدانیم!

- سوال کلیدی. چگونه می‌توان ارزش حالت‌ها را بدون نیاز به دانستن T و R محاسبه نمود؟

ارزیابی سیاست مبتنی بر نمونه‌برداری

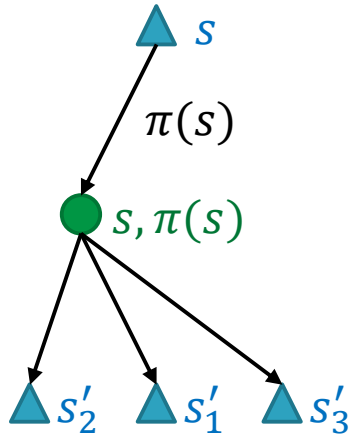
- هدف. می‌خواهیم تخمین خود را از V با استفاده از معادله‌ی زیر بهبود دهیم.

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

- ایده. از حالت‌های نتیجه نمونه‌برداری کن (با انجام عمل!) و سپس از آنها میانگین بگیر.

$$sample_1 = R(s,\pi(s),s_1') + \gamma V_k^{\pi}(s_1')$$
$$sample_2 = R(s,\pi(s),s_2') + \gamma V_k^{\pi}(s_2')$$
$$\dots$$
$$sample_n = R(s,\pi(s),s_n') + \gamma V_k^{\pi}(s_n')$$

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



ارزیابی سیاست مبتنی بر نمونه‌برداری

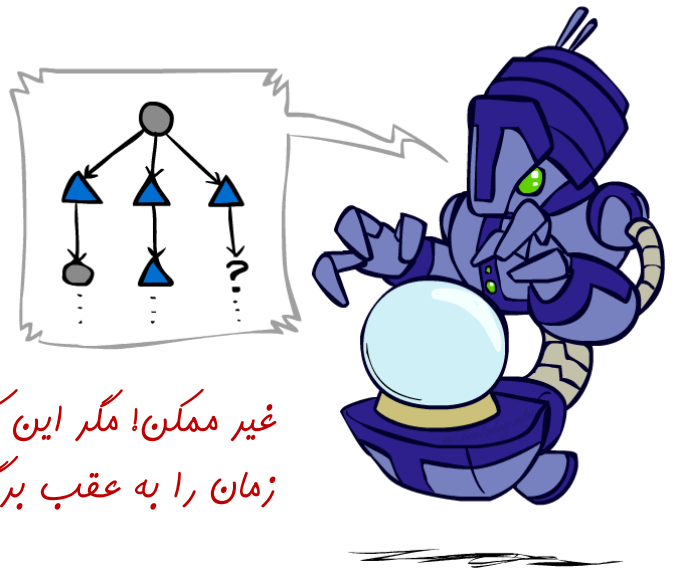
- هدف. می‌خواهیم تخمین خود را از V با استفاده از معادله‌ی زیر بهبود دهیم.

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

- ایده. از حالت‌های نتیجه نمونه‌برداری کن (با انجام عمل!) و سپس از آنها میانگین بگیر.

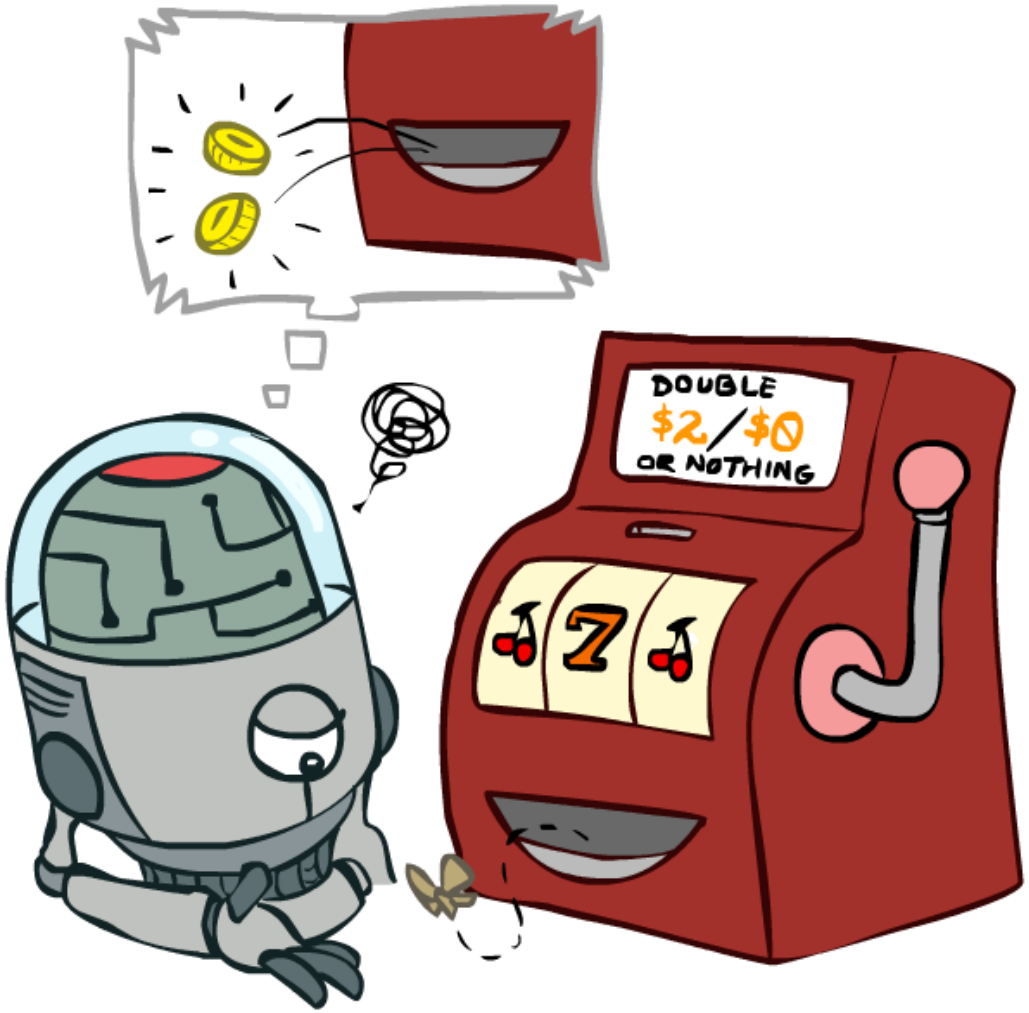
$$sample_1 = R(s,\pi(s),s_1') + \gamma V_k^{\pi}(s_1')$$
$$sample_2 = R(s,\pi(s),s_2') + \gamma V_k^{\pi}(s_2')$$
$$\dots$$
$$sample_n = R(s,\pi(s),s_n') + \gamma V_k^{\pi}(s_n')$$

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

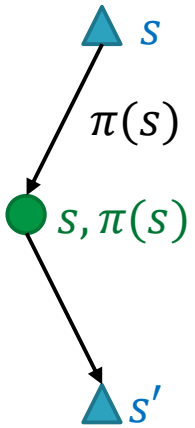


غیر ممکن! مگر این که بتوانیم زمان را به عقب برگردانیم.

یادگیری تفاضل زمانی



یادگیری تفاضل زمانی



- ایده‌ی بزرگ. یادگیری از تک تک نمونه‌ها به صورت جداگانه!
  - به روز رسانی تخمین V(s) پس از هر تجربه (s, a, s′, r)
  - پیامدهایی مانند s′ که احتمال وقوع بیشتری دارند، سهم بیشتری در به روز رسانی مقدار V(s) خواهند داشت.
- یادگیری مقادیر به روش تفاضل زمانی.

نمونه‌برداری از V(s)

$$sample = R(s, \pi(s), s') + \gamma V^{\pi}(s')$$

به روز رسانی مقدار V(s)

نرخ یادگیری

$$V^{\pi}(s) \leftarrow (1 - \alpha)V^{\pi}(s) + (\alpha)\, sample$$

بازنویسی قاعده‌ی به روز رسانی

$$V^{\pi}(s) \leftarrow V^{\pi}(s) + (\alpha)(sample - V^{\pi}(s))$$

خطا

میانگین‌گیری نمایی

- میانگین گیری نمایی.
  - قاعده‌ی به روز رسانی بر اساس درون‌یابی:

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

  - استفاده از این قاعده باعث می‌شود نمونه‌های جدیدتر اهمیت بیشتری داشته باشند:

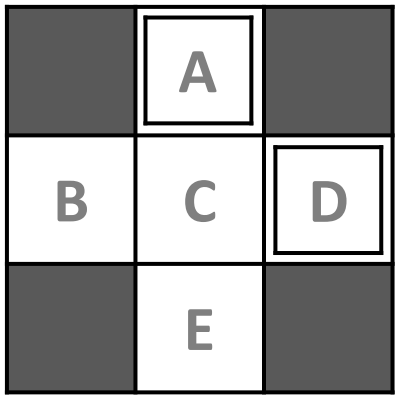
$$\bar{x}_n = \frac{x_n + (1-\alpha)\cdot x_{n-1} + (1-\alpha)^2 \cdot x_{n-2} + \cdots}{1 + (1-\alpha) + (1-\alpha)^2 + \cdots}$$

  - یعنی، فراموش کردن نمونه‌های بسیار قدیمی!
    - به هر حال نمونه‌های مربوط به گذشته‌ی بسیار دور، خیلی درست نیستند.

- توجه. کاهش نرخ یادگیری آلفا می‌تواند باعث پایدارتر شدن الگوریتم یادگیری شود.

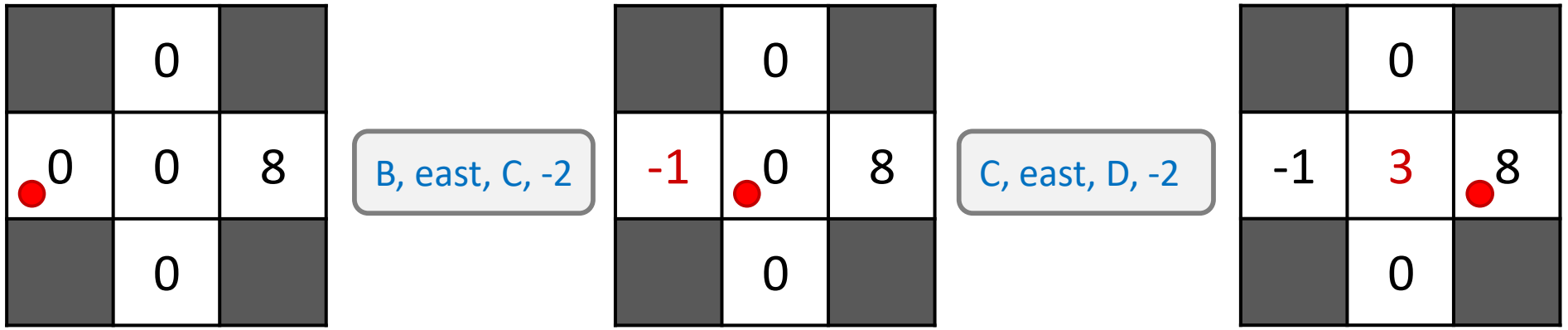
مثال: یادگیری تفاضل زمانی

حالت‌ها



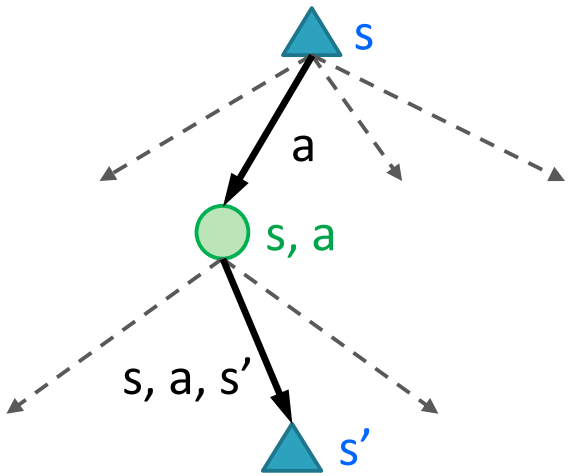
$\gamma = 1, \alpha = 1/2$

تغییر حالت‌های مشاهده شده



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

مشکلات یادگیری تفاضل زمانی



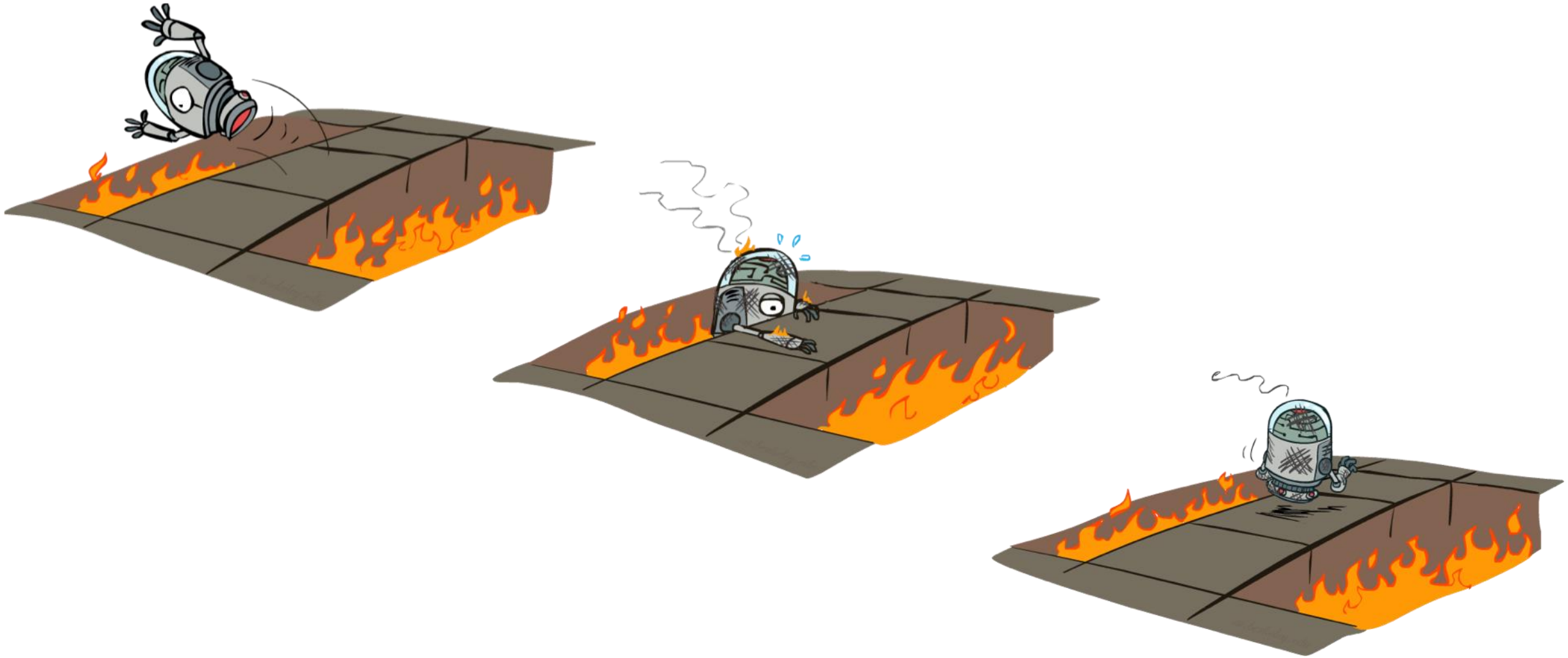
- الگوریتم TD یک روش مستقل از مدل برای ارزیابی سیاست است.
- پس از محاسبه‌ی ارزش حالت‌ها، باید یک عمل انتخاب کنیم.
- اما برای انتخاب عمل نیاز به دانستن مقادیر T و R داریم.

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

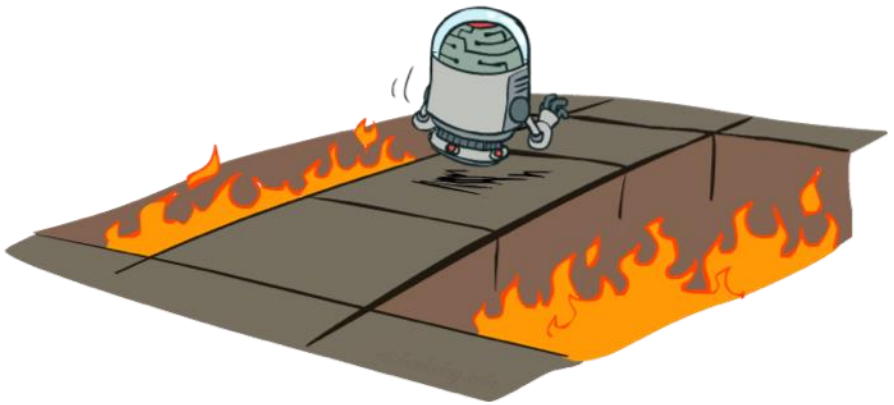
$$Q(s,a) = \sum_{s\prime} T(s,a,s')[R(s,a,s') + \gamma V(s')]$$

- ایده. به جای محاسبه‌ی ارزش حالت‌ها، ارزش حالت‌های q را محاسبه کن.
  - در این صورت، انتخاب عمل به صورت مستقل از مدل قابل انجام خواهد بود!

یادگیری تقویتی فعال



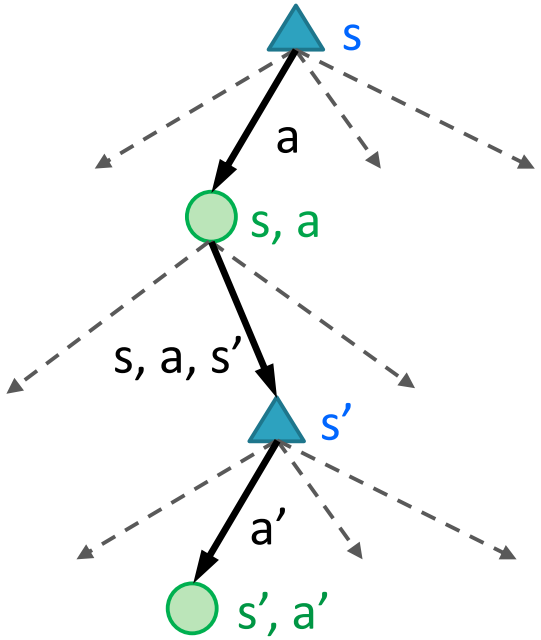
یادگیری تقویتی فعال



- یادگیری تقویتی فعال. محاسبه‌ی سیاست بهینه
  - تابع تغییر حالت $T(s,a,s')$ ناشناخته است.
  - تابع پاداش $R(s,a,s')$ نیز ناشناخته است.
  - اکنون عامل خودش اعمال را انتخاب می‌کند.
  - هدف: یادگیری سیاست بهینه / مقادیر بهینه

- در این مورد:
  - یادگیرنده می‌تواند بین اعمال ممکن انتخاب انجام دهد.
  - مسئله‌ی اساسی: توازن میان میزان کاوش و میزان بهره‌برداری!
  - این برنامه‌ریزی آفلاین **نیست**! عامل واقعاً در محیط عملیاتی را انجام می‌دهد و نتیجه‌ی آن را مشاهده می‌کند.

تکرار مقدار Q



- تکرار مقدار. محاسبه‌ی ارزش حالت‌ها به صورت تکرار شونده
  - با بردار $V_0(s) = 0$ شروع کن (که می‌دانیم درست است).
  - در هر تکرار، با داشتن بردار $V_k(s)$، بردار $V_{k+1}(s)$ را محاسبه کن.

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

- اما مقادیر Q مفیدتر هستند، پس آنها را محاسبه کن.
  - با $Q_0(s,a) = 0$ شروع کن (که می‌دانیم درست است).
  - در هر تکرار، با داشتن بردار $Q_k(s,a)$ بردار $Q_{k+1}(s,a)$ را محاسبه کن.

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

الگوریتم یادگیری Q

- یادگیری Q. الگوریتم تکرار مقدار Q مبتنی بر نمونه‌برداری

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

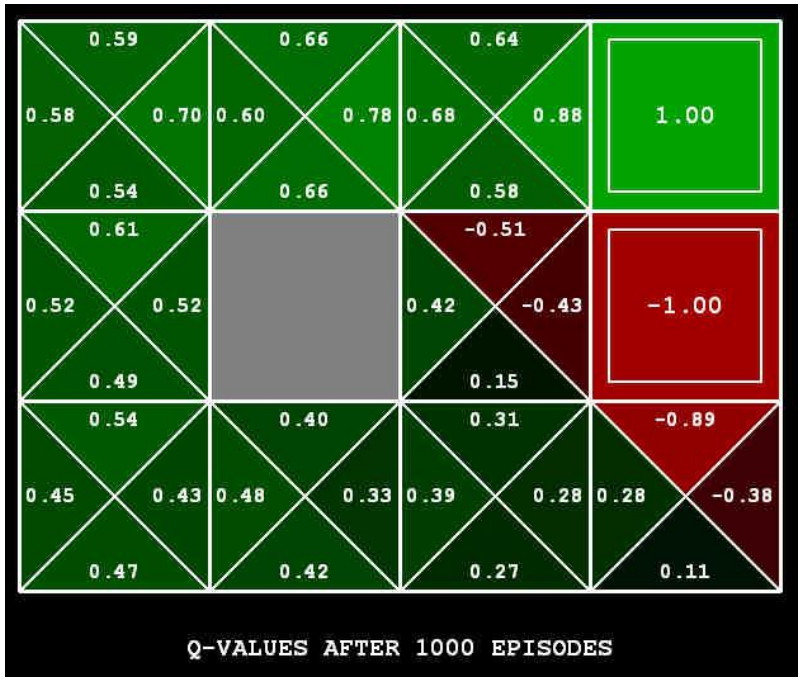
← اما T و R ناشناخته هستند!

- یادگیری مقادیر Q(s, a)
  - دریافت نمونه (s, a, s', r)
  - در نظر گرفتن تخمین قبلی: Q(s, a)
  - در نظر گرفتن تخمین مربوط به نمونه‌ی جدید:

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

  - به روز رسانی تخمین: [میانگین‌گیری]

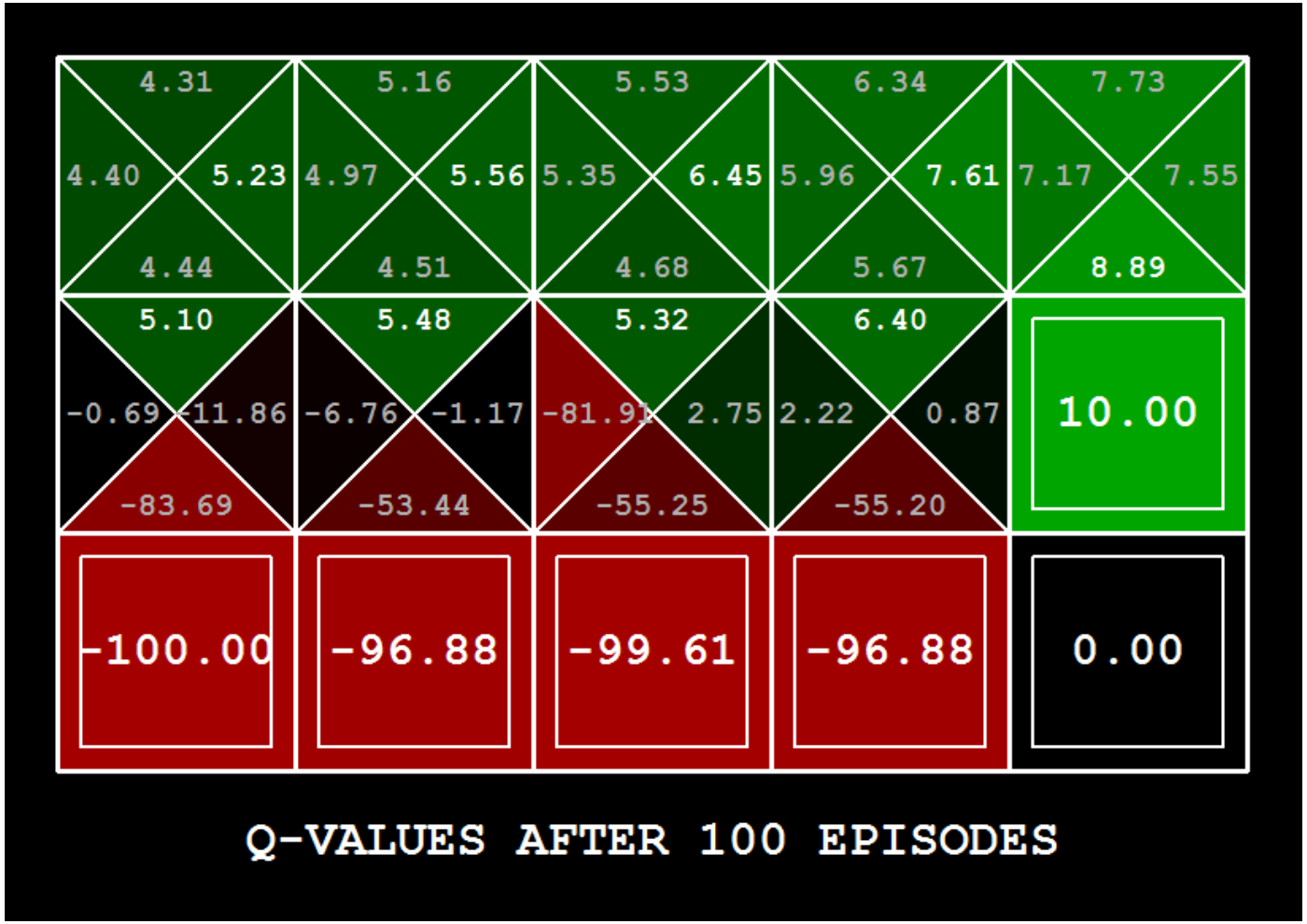
$$Q(s,a) = (1-\alpha)Q(s,a) + (\alpha)[sample]$$




```
% python gridworld.py -a q -k 1000
```

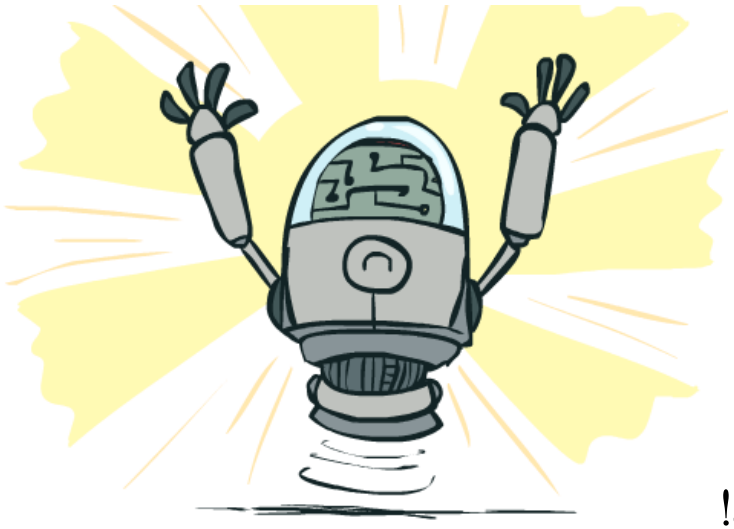
اجرای نمایشی


```
% python gridworld.py -a q -g CliffGrid -k 100 -m
```



ویژگی‌های الگوریتم یادگیری Q

- همگرایی. الگوریتم یادگیری Q در سیاست بهینه همگرا می‌شود - حتی اگر عامل بهینه عمل نکند.



- هشدارها.
  - عامل باید به اندازه‌ی کافی محیط را کاوش کند.
  - نرخ یادگیری باید در نهایت به اندازه‌ی کافی کوچک شود.
  - ... اما مقدار آن نباید خیلی سریع کاهش داده شود.
  - به طور مبنایی، در حد، چگونگی انتخاب عمل به وسیله‌ی عامل اهمیت ندارد!