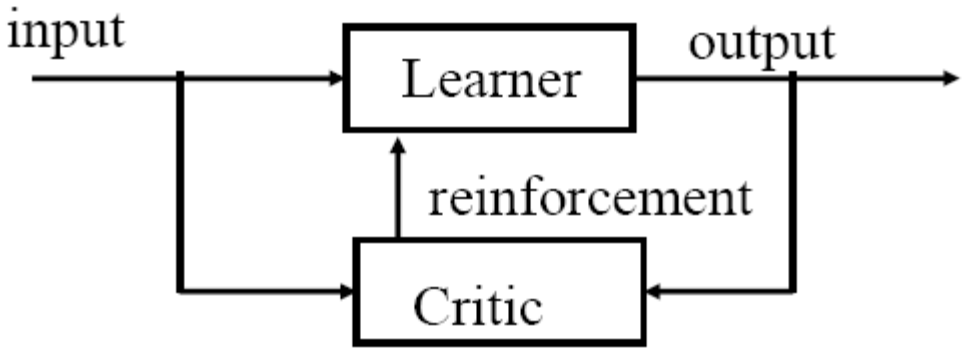
به نام یگانه معبود بخشنده مهربان

یادگیری تقویتی

Reinforcement Learning

یادگیری تقویتی

- میخواهیم سیاست کنترلی مقابل را یاد بگیریم: $\pi : X \to A$
  - نمونه‌هایی از x را مشاهده میکنیم (اما خروجیهای a داده نمیشوند).
  - بجای a یک بازخورد (تقویت یا پاداش) از یک ناقد دریافت میکنیم که مطلوبیت خروجی انتخاب شده را کمّی میکند:



  - سیگنال تقویت ممکن است معین (deterministic) نباشد.
  - هدف یادگیری نگاشت $\pi : X \to A$ است، به نحوی که بهترین تقویت مورد انتظار (expected reinforcement) را بدهد.

مثال: پرتاب سکه‌ها

- **سه سکه (معیوب) برای پرتاب موجود است:**
  - یک سکه از این سه به تصادف برای پرتاب انتخاب میشود (با مشخصه معلوم).
  - یک پیشگویی در مورد شیر یا خط آمدن سکه انتخابی انجام میدهیم.
  - در صورت درست بودن حدس یک واحد پاداش و در غیر اینصورت یک واحد جریمه دریافت میکنیم.
- **مدل یادگیری تقویتی (RL):**
  - ورودی (input): X – سکه‌ای که باید در این مرحله پرتاب شود
  - عمل (action): A – انتخاب شیر یا خط (حدس)
  - تقویت (reinforcement): $\{1, -1\}$
- **سیاست مورد یادگیری:** $\pi : X \to A$ ، مثلاً:

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

- **هدف:** یادگیری سیاست بنحوی که سود مورد انتظار در آینده بیشینه شود:
  - γ : ضریب کاهش ارزش پاداش

$$\pi: \left| \begin{array}{l} \text{Coin1} \longrightarrow ? \\ \text{Coin2} \longrightarrow ? \\ \text{Coin3} \longrightarrow ? \end{array} \right| \quad \textit{s.t. maximize} \; E\left(\sum_{t=0}^{\infty} \gamma^t r_t\right)$$

یادگیری تقویتی

- **هدف:**
  - یافتن نگاشت $\pi^* : X \rightarrow A$ که بیشینه کننده ترکیبی از تقویتهای (پاداشهای) آتی دریافتی در طول زمان است.
- **مدلهای ارزش دهی (کمی کننده میزان مطلوبیت نگاشت):**
  - مدل افق محدود: $E(\sum_{t=0}^{T} r_t)$ افق زمانی : $T > 0$
  - مدل افق نامحدود تنزلی: $E(\sum_{t=0}^{\infty} \gamma^t r_t)$ ضریب تنزل : $0 < \gamma < 1$
  - پاداش متوسط: $\lim_{T \rightarrow \infty} \frac{1}{T} E(\sum_{t=0}^{T} r_t)$
- **مدل یادگیر بطور فعال با محیط تعامل دارد:**
  - در آغاز مدل یادگیری چیزی درباره محیط نمیداند.
  - به تدریج از آزمونها تجربه کسب میکند و یاد میگیرد که چگونه به محیط واکنش نشان دهد.

مسایل مطرح در یادگیری تقویتی

- کاوش (exploration) در مقابل بهره برداری (exploitation):
  - پس از چند مرحله تعامل، آیا سیستم باید بهترین انتخاب جاری را انجام دهد (بهره برداری)، یا سعی در یادگیری بیشتر درباره محیط داشته باشد (کاوش).
  - بهره برداری: ممکن است موجب انتخاب یک عمل شبه بهینه شده و از یادگرفتن پاسخ بهینه پیشگیری کند.
  - کاوش: ممکن است زمان زیادی را برای بررسی عملهای شبه بهینه جاری مصروف سازد.
- تأثیر عملها روی محیط (ورودی x بَعدی):
  - بی‌تأثیر: توزیع x ثابت است. عواقب اعمال (پاداشها) فوری دیده میشوند.
  - مؤثر: توزیع x تغییرپذیر است. عواقب اعمال ممکن است با تأخیر دیده شوند.
- بنابراین از این جهت دو نوع یادگیری تقویتی خواهیم داشت:
  - یادگیری با پاداشهای آنی (مثال پرتاب سکه)
  - یادگیری با پاداشهای تأخیری (مثال ناوبری ربات: انتخاب حرکتها بر وضعیت محیط (مکان ربات) تأثیر میگذارد. یک پاداش بزرگ در مکان هدف داریم که با تأخیر به عملهای قبلی میرسد)

یادگیری تقویتی با پاداش آنی

- **مثال پرتاب سکه:**
  - پاداش مورد انتظار: $E(\sum_{t=0}^{\infty} \gamma^t r_t)$
  - پاداش پس از هر انتخاب مشخص میشود.
  - انتخاب (حدس) ما در محیط و به تبع آن در پاداشهای آتی تأثیری ندارد:

$$E(\sum_{t=0}^{\infty} \gamma^t r_t) = E(r_0) + E(\gamma r_1) + E(\gamma^2 r_2) + \dots$$

$r_0, r_1, r_2 \dots$: پاداشها در هر قدم

  - پاداش مورد انتظار در یک مرحله برای ورودی $\mathbf{x}$ و انتخاب a: $R(\mathbf{x}, a)$

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

ω_j : رخداد مخفی در پرتاب سکه (پاسخ صحیح)

  - پاداش مورد انتظار یک مرحله‌ای برای یک استراتژی $\pi : X \to A$

$$R(\pi) = \sum_x R(\mathbf{x}, \pi(\mathbf{x})) P(\mathbf{x})$$

پاداش مورد انتظار برای $r_0, r_1, r_2 \dots$ است. $R(\pi)$

یادگیری تقویتی با پاداش آنی

- پاداش مورد انتظار: $E(\sum_{t=0}^{\infty} \gamma^t r_t) = E(r_0) + E(\gamma r_1) + E(\gamma^2 r_2) + \ldots$
- بهینه سازی پاداش مورد انتظار:

$$\max_{\pi} E(\sum_{t=0}^{\infty} \gamma^t r_t) = \max_{\pi} \sum_{t=0}^{\infty} \gamma^t E(r_t) = \max_{\pi} \sum_{t=0}^{\infty} \gamma^t R(\pi) = \max_{\pi} R(\pi)(\sum_{t=0}^{\infty} \gamma^t)$$

$$= (\sum_{t=0}^{\infty} \gamma^t) \max_{\pi} R(\pi)$$

$$\max_{\pi} R(\pi) = \max_{\pi} \sum_{x} R(\mathbf{x}, \pi(\mathbf{x})) P(\mathbf{x}) = \sum_{x} P(\mathbf{x})[\max_{\pi(\mathbf{x})} R(\mathbf{x}, \pi(\mathbf{x}))]$$

- استراتژی بهینه: $\pi^* : X \rightarrow A$

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

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

یادگیری تقویتی با پاداش آنی

- **مسأله:** در یادگیری تقویتی $R(\mathbf{x}, a)$ (پاداش مورد انتظار برای انجام عمل a در واکنش به ورودی $\mathbf{x}$) را نمیدانیم.
- **یک راه حل:**
  - برای هر ورودی $\mathbf{x}$، عملهای مختلف a را آزمایش میکنیم.
  - با متوسط گیری روی پاداشهای مشاهده شده، $R(\mathbf{x}, a)$ را تخمین میزنیم:

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

  - انتخاب عمل بهینه: $\pi(\mathbf{x}) = \arg\max_a \tilde{R}(\mathbf{x}, a)$
  - دقت تخمین: (حد هافدینگ (**Hoeffding's bound**))

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

  - تعداد نمونه‌ها: $N_{x,y} \geq \frac{(r_{\max} - r_{\min})^2}{2\varepsilon^2} \ln \frac{1}{\delta}$

یادگیری تقویتی با پاداش آنی

- روش برخط (تقریب تصادفی (stochastic approximation))
  - یک راه دیگر برای تخمین $R(\mathbf{x}, a)$
  - انتخاب عمل a برای ورودی $\mathbf{x}$ و محاسبه پاداش $r^{x,a}$
  - بهنگام سازی یک تخمین بر اساس رابطه زیر:

$$\tilde{R}(\mathbf{x}, a) \leftarrow (1-\alpha)\tilde{R}(\mathbf{x}, a) + \alpha\, r^{x,a}$$

α : یک نرخ یادگیری

  - خواص همگرایی:

– با انتخاب تغییرات نرخ یادگیری بنحو مناسب، تقریب همگرا خواهد شد.

– فرض: $\alpha(n(x,a))$ نرخ یادگیری برای nامین زوج (x,a) آزمایش شونده

– در صورت برقراری دو شرط زیر، همگرایی تضمین میشود:

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

کاوش در مقابل بهره برداری

- در هر لحظه سیستم یک تخمین $\tilde{R}(\mathbf{x}, a)$ برای هر زوج $\mathbf{x}$ و a دارد.
  - آیا سیستم یادگیر باید بهترین انتخابش در لحظه جاری را انجام دهد (بهره برداری):
$$\hat{\pi}(\mathbf{x}) = \arg\max_{a \in A} \tilde{R}(\mathbf{x}, a)$$
  - یا عمل دیگری را انتخاب کرده و تخمینش را بهبود بخشد (کاوش)؟
- استراتژیهای کاوش/بهره برداری مختلفی وجود دارد:
  - کاوش یکنواخت:

– بهترین گزینه جاری با احتمال $1 - \varepsilon$ انتخاب شود: $\hat{\pi}(\mathbf{x}) = \arg\max_{a \in A} \tilde{R}(\mathbf{x}, a)$

سایر گزینه‌ها با احتمال یکنواخت زیر انتخاب شوند: $\frac{\varepsilon}{|A| - 1}$

  - کاوش بولتزمن:

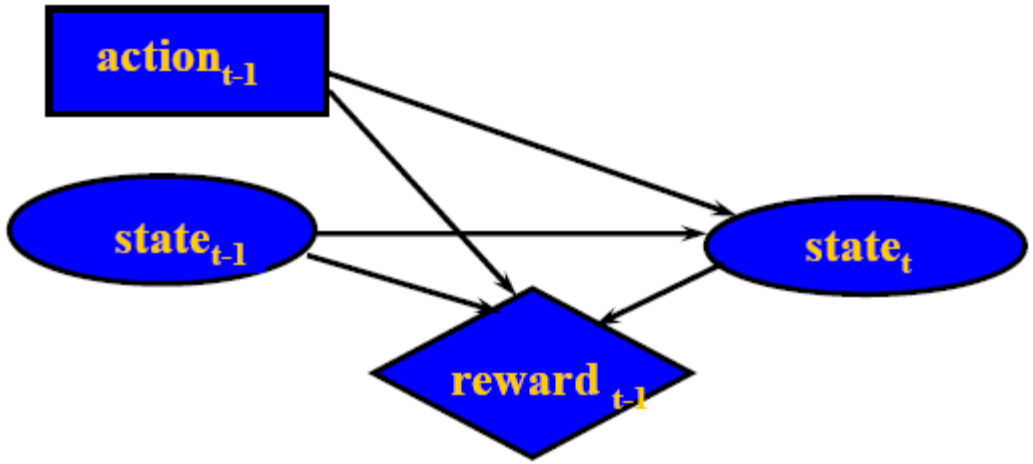
– عمل به تصادف اما به نسبت تخمین پاداش مورد انتظار جاری انتخاب شود:

که $\mathbf{T}$ پارامتر دماست

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

یادگیری تقویتی با پاداش تأخیری

- عملها علاوه بر پاداش آنی، با تأثیر گذاشتن در حالتهای بعدی محیط بطور غیر مستقیم در پاداشهای آتی نیز تأثیر گذار هستند.
- نیاز به مدلی جهت بازنمایی تغییرات حالت محیط داریم.
- در اینجا از مدلی بنام فرآیند تصمیم مارکف (Markov Decision Process یا MDP) استفاده میشود:
  - در زمینه هایی چون AI ، OR، کنترل، و ... کاربرد دارد.
  - فرض مارکف (Markov assumption): حالت بعدی به حالت جاری و عمل جاری بستگی دارد، و نه به حالتها (عملها) در گذشته.



فرآیند تصمیم مارکف

- با یک چهارتایی بصورت مقابل تعریف میشود: (S, A, T, R)

• **A set of states** S (X)	locations of a robot
• **A set of actions** A	move actions
• **Transition model** $S \times A \times S \rightarrow [0,1]$	where can I get with different moves
• **Reward model** $S \times A \times S \rightarrow \Re$	reward/cost for a transition

- هدف یافتن بهترین سیاست $\pi^* : S \rightarrow A$ است.
- تابع ارزش (V) برای یک سیاست، میزان مطلوبیت سیاست را تحت یک مدل، مثلاً مدل افق نامحدود تنزلی $E(\sum_{t=0}^{\infty} \gamma^t r_t)$ ، کمّی میکند، از طریق:
  - ترکیب پاداشهای آتی روی یک مسیر
  - ترکیب پاداشها برای چند مسیر

√450

ارزش یک سیاست برای فرآیند تصمیم مارکف

- یک سیاست ثابت را در نظر گیرید: $\pi : S \to A$
- سئوال: چگونه ارزش این سیاست را با فرض مدل افق نامحدود تنزلی محاسبه کنیم؟
  - معادله نقطه ثابت:

$$V^{\pi}(s) = \underbrace{R(s, \pi(s))}_{\text{پاداش مورد انتظار یک مرحله‌ای برای اولین عمل}} + \underbrace{\gamma \sum_{s' \in S} P(s' | s, \pi(s)) V^{\pi}(s')}_{\text{پاداش تنزل یافته مورد انتظار برای دنبال کردن سیاست در قدمهای بعدی}}$$

$$\mathbf{v} = \mathbf{r} + \mathbf{U}\mathbf{v} \quad \Longrightarrow \quad \mathbf{v} = (\mathbf{I} - \mathbf{U})^{-1}\mathbf{r}$$

  - برای یک فضای حالت محدود، یک مجموعه معادلات خطی بدست می‌آید.

سیاست بهینه

- **ارزش سیاست بهینه:**

$$V^*(s) = \max_{a \in A} \left[\underbrace{R(s,a)}_{\text{پاداش مورد انتظار یک مرحله‌ای برای اولین عمل}} + \underbrace{\gamma \sum_{s' \in S} P(s'|s,a) V^*(s')}_{\text{پاداش تنزل یافته مورد انتظار برای دنبال کردن سیاست بهینه در قدمهای بعدی}} \right]$$

  - عبارت فوق را بصورت مقابل بیان میکنیم: $V^*(s) = (HV^*)(s)$

- **سیاست بهینه:** $\pi^*: S \rightarrow A$

$$\pi^*(s) = \arg\max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V^*(s') \right]$$

محاسبه سیاست بهینه

- برنامه ریزی پویا برای یافتن ارزش بهینه بصورت تکراری:
  - ابتدا تابع ارزش بهینه و سپس سیاست بهینه را محاسبه میکند.
  - تقریب تصادفی همگرا شونده به تابع ارزش بهینه

Value iteration (ε)

initialize $\mathbf{V}$;; V is vector of values for all states

repeat

set $\mathbf{V}' \leftarrow \mathbf{V}$

set $\mathbf{V} \leftarrow \mathbf{HV}$

until $\|\mathbf{V}' - \mathbf{V}\|_\infty \leq \varepsilon$

output $\pi^*(s) = \arg\max_{a \in A} \left[R(s,a) + \gamma \sum_{s' \in S} P(s' \mid s,a) V(s') \right]$

یادگیری تقویتی سیاستهای بهینه

- در یادگیری تقویتی مدل MDP را نمیدانیم.
- هدف: یادگیری سیاست بهینه $\pi^* : S \rightarrow A$
- دو روش اصلی:
  - یادگیری مبتنی بر مدل:
    - ابتدا مدل MDP (شامل احتمالات و پاداشها) یادگیری شود.
    - سپس سیاست بهینه از روی این مدل محاسبه شود.
  - یادگیری بدون مدل:
    - یادگیری مستقیم سیاست بهینه انتخاب عمل.
    - نیاز به یادگیری پارامترهای MDP ندارد.

یادگیری مبتنی بر مدل

- نیاز به یادگیری احتمالات انتقال و پاداشها داریم.
- یادگیری احتمالات:
  - تخمین ML یا تخمین بیزین پارامتر
  - استفاده از شمارش (تعداد):

$$\widetilde{P}(s'|s,a) = \frac{N_{s,a,s'}}{N_{s,a}} \qquad N_{s,a} = \sum_{s' \in S} N_{s,a,s'}$$

- یادگیری پاداشها:
  - مشابه یادگیری با پاداش آنی:

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

- مسأله:
  - در بهنگام سازی برخط سیاست، نیاز به حل MDP پس از هر مرحله بهنگام سازی داریم.

یادگیری بدون مدل

- **ایده:** بهنگام سازی تابع ارزش (بصورت تکراری):

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

- **بانام گذاری مقابل:** $Q(s,a) = R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) V(s')$

- **خواهیم داشت:** $V(s) \leftarrow \max_{a \in A} Q(s,a)$

- **بهنگام سازی می‌تواند تنها برحسب تابع Q تعریف شود:**

$$Q(s,a) \leftarrow R(s,a) + \gamma \sum_{s' \in S} P(s'|s,a) \max_{a'} Q(s',a')$$

Q-learning

- **Q-learning : از ایده بهنگام سازی مقدار Q استفاده می‌کند، تنها بصورت تصادفی (برخط یا نمونه به نمونه) بهنگام سازی را انجام می دهد:**
  - **بجای رابطه قبل:**

$$Q(s,a) \leftarrow R(s,a) + \gamma \sum_{s' \in S} P(s' \mid s,a) \max_{a'} Q(s',a')$$

رابطه زیر را بکار می‌برد:

$$\hat{Q}(s,a) \leftarrow (1-\alpha)\hat{Q}(s,a) + \alpha\left(r(s,a) + \gamma \max_{a'} \hat{Q}(s',a')\right)$$

که در آن:

- $r(s,a)$: **پاداش دریافتی از محیط پس از انجام عمل *a* در حالت *s***
- s' : **حالت جدید که با انجام عمل *a* به آن منتقل می‌شویم.**
- α : **نرخ یادگیری، که تابعی است از** $N_{s,a}$ **(تعداد اجرای *a* در حالت s).**

Q-learning

- الگوریتم زیر بهنگام سازی برخط را بصورت تکراری در حین تعامل مستقیم با محیط انجام میدهد:

Q-learning

initialize $Q(s,a) = 0$ for all s,a pairs

observe current state s

repeat

select action a ; use some exploration/exploitation schedule

receive reward r

observe next state s'

update $Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s',a')\right)$

set s to s'

end repeat

Q-learning

- **همگرایی Q-learning به ارزشهای بهینه Q تحت شرایط زیر تضمین میشود:**
  - **همه حالات ملاقات شوند، و هر عمل در هر حالت به تعداد بینهایت مرتبه آزمایش شود (این شرط از طریق مکانیزم کاوش/بهره برداری تأمین میشود).**
  - **دنباله نرخهای یادگیری برای هر $Q(s,a)$ شرایط زیر را محقق سازد:**

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

که $\alpha(n(s,a))$ نرخ یادگیری برای n امین آزمایش (s,a) می‌باشد.

- **مسأله کاوش در مقابل بهره برداری**
  - **در یادگیری تقویتی با پاداش تأخیری در هر لحظه بر ای هر زوج حالت-عمل تخمین $\hat{Q}(\mathbf{x},a)$ را داریم.**
  - **بهره برداری: انتخاب بهترین عمل در حال حاضر: $\hat{\pi}(\mathbf{x}) = \arg\max_{a \in A} \hat{Q}(\mathbf{x},a)$**
  - **کاوش: انتخاب عمل دیگر a و بهبود دادن تخمین $\hat{Q}(\mathbf{x},a)$**
  - **استراتژی‌ها: کاوش یکنواخت، کاوش بولتزمن**

تسریع Q-learning

- قانون بهنگام سازی Q-learning پایه ممکن است پاداشهای دور (تأخیردار) را بسیار کند انتشار دهد ... .
- مشکل: در هر اجرا، ارزشها را تنها یک قدم به عقب انتشار میدهیم. لذا برای انتشار ارزشها به چند قدم قبل، نیاز به چند آزمایش داریم.
- یک راه حل: نگهداری ارزشها برای تعداد زیادی از قدمها

$$q_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots = \sum_{i=0}^{\infty} \gamma^i r_{t+i}$$

- جایگزین کردن پاداش آنی با پاداش n مرحله‌ای

$$q_t^{\,n} = \sum_{i=0}^{n} \gamma^i r_{t+i} + \gamma^{n+1} \max_{a'} Q_{t+n}(s', a')$$

و به تعویق انداختن بهنگام سازیها به تعداد n قدم:

$$Q_{t+n+1}(s,a) \leftarrow Q_{t+n}(s,a) + \alpha \left(q_t^{\,n} - Q_{t+n}(s,a) \right)$$

تسریع Q-learning

- یک مشکل ایده نگهداری n مرحله‌ای پاداشها، نیاز به انتظار n مرحله‌ای برای بهنگام سازیها است.
- یک راه حل: روش تفاضل زمانی (Temporal difference)
  - نگهداری جزئی پاداشها بعد از هر قدم
  - نسخ مختلفی از این ایده پیاده سازی شده است.
  - ...
- جمع بندی
  - یادگیری تقویتی نسبتاً ساده است.
  - روشهای برخط میتوانند تغییرات در محیطهای غیر ایستا را دنبال کرده و با آنها تطبیق یابند.
  - کاربردهای موفقیت آمیزی داشته است، مثلاً در زمینه های زیر:
    - یادگیری بازیهای فکری دو نفره در سطح قهرمانی
    - تخصیص کانال پویا در شبکه های مخابرات سیار
    - ناوبری ربات در محیطهای دارای مانع