Reinforcement Learning with MATLAB

اسپندی

مفاهیم شهودی یادگیری تقویتی (قسمت های اول و دوم دوره)

فهرست مطالب:

1) مفاهیم شهودی
2) مسأله Multi-Armed Bandit
3) فرآیندهای تصمیم‌گیری مارکوف محدود (FMDP)
4) برنامه‌ریزی پویا Dynamic Programming
5) روش‌های Montecarlo
6) یادگیری با تفاضل زمانی Temporal Difference

انواع الگوریتم‌های یادگیری ماشین:

✓ یادگیری نظارت شده supervised learning
✓ یادگیری نظارت نشده (غیر نظارتی) unsupervised learning
✓ یادگیری تقویتی Reinforcement learning

اجزای یادگیری تقویتی:

✓ محیط Environment
✓ عامل Agent
✓ عمل سیاست Policy
✓ پاداش Reward
✓ Eploitation and Exploration

فرآیندهای یادگیری بر اساس اینکه چه تمایلی را در اختیار داریم، دسته‌بندی می‌شوند.

در یادگیری نظارت شده داده‌ها لیبل دارند.
الگوریتم یک معلم درست یا نادرست تصمیمات را تعیین می‌کند.
که کاربرد مانند معلم که می‌دونی.

★ در یادگیری غیرنظارتی لیبل وجود ندارد و خود عامل باید الگوهای موجود بین گروه‌ها را تشخیص دهد.

داده‌ها به گروه‌هایی که بیشترین شباهت را با هم دارند دسته‌بندی می‌شوند.

مثلاً داده‌ی drain می‌آید در صورتی که به چند دسته تقسیم می‌شوند و داده جدید به دسته‌ای می‌رود که بیشترین شباهت را بهشون دارد.

خوشه‌بندی: دسته‌بندی داده‌های بدون لیبل به چند گروه به شکلی که اعضای داخل هر گروه بیشترین شباهت را با هم داشته باشند ولی کمترین شباهت ممکن بین گروه‌های مختلف وجود داشته باشد.
(بیشترین تفاوت)

مثل الگوریتم K-means

حالا! یادگیری تقویتی (Reinforcement Learning): یادگیری از طریق تعامل با محیط

عامل در هر موقعیت (یا حالت) می‌تواند حرکات (یا عمل‌های مختلف) انجام دهد.

برای هر عمل پاداش و یا تنبیه وجود دارد.
Reward Penalty

عامل یاد می‌گیرد که در هر موقعیت، عملی را انجام دهد که میزان پاداش دریافتی ماکزیمم شود یا میزان تنبیه مینیمم شود.

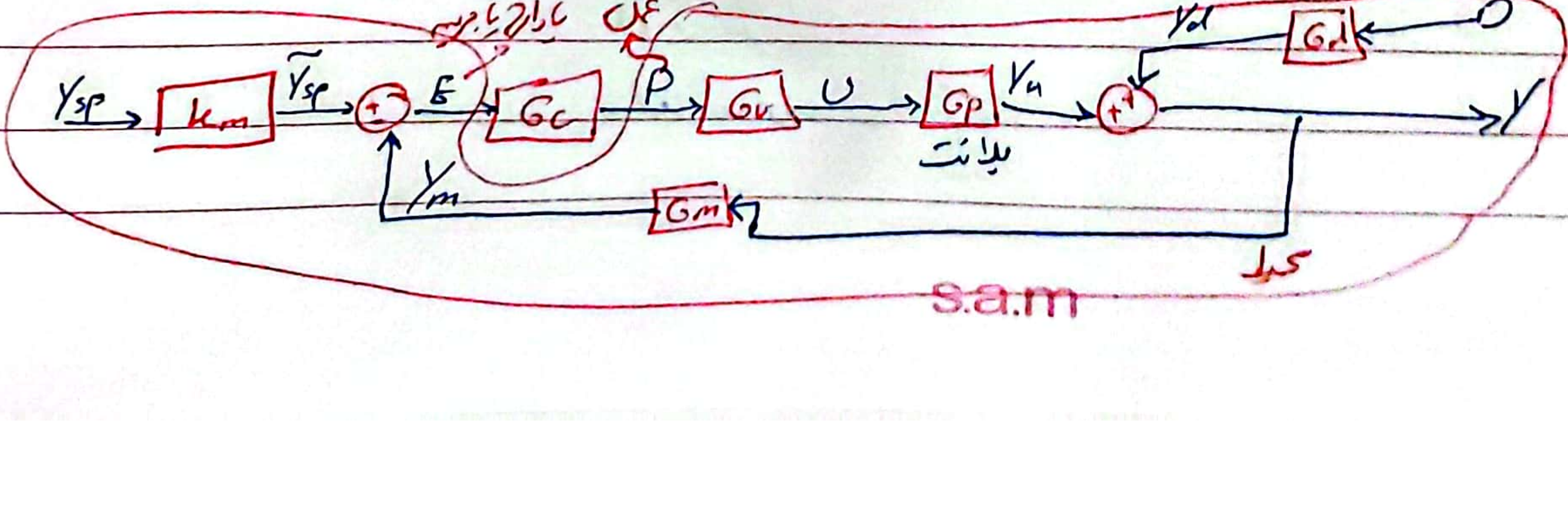
$\max f(x) = \min -f(x)$

مثل یا بگیره وقتی زنگ به صدا درمیاد وقت غذا خوردن است و باید به محل سرو کردن غذا برود.

★ در یادگیری نظارتی و غیرنظارتی با دیتابیس استاتیک کار می‌کنند. اما یادگیری تقویتی با داده‌های دریافت شده از یک محیط دینامیک یاد می‌گیرد.
(محیطی که همواره در حال تغییر است.)

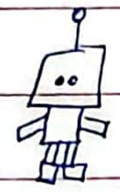
کاربردهای یادگیری تقویتی: کنترل، بازی، رباتیک و بیزینس و ...

خوبی یادگیری تقویتی این است که نیازی به دینامیک محیط ندارد ← مزیت ویژه یادگیری تقویتی در کنترل سیستم‌هایی که دینامیک شخصی از آن در دسترس نیست.



در طراحی کنترل کننده G_c در درس کنترل خطی باید بقیه دینامیک‌ها را بدانیم ولی در یادگیری تقویتی نیازی به دانستن دینامیک‌ها نیست.

و G_c یا کنترل کننده عاملی خواهد شد که باید با تعامل با آزمایش یاد بگیرد.



مثال دیگر: کنترل قدم زدن ربات
- حفظ قدم زدن
- دفع اغتشاشات
- عبور از موانع

در بازی‌ها هم عامل یاد می‌گیرد که با دریافت بیشترین پاداش یا کمترین هزینه بازی را تمام کند.

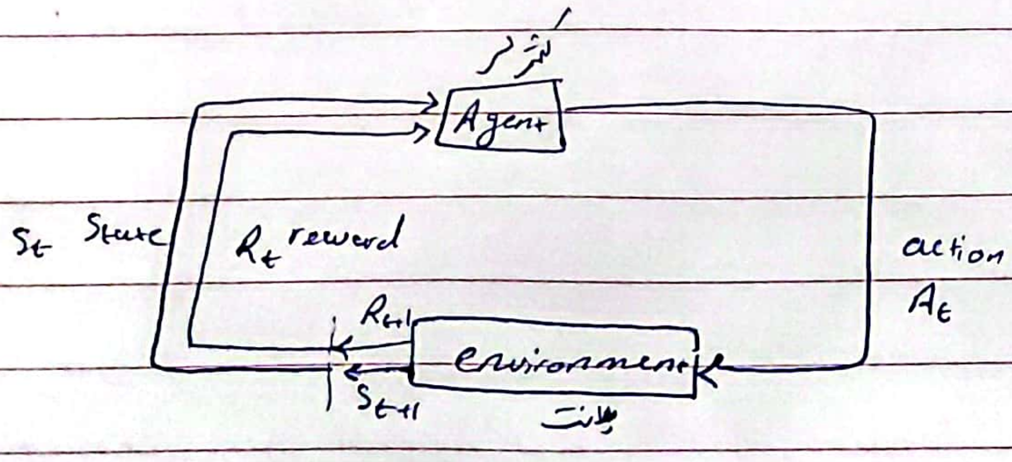
در سال ۱۹۹۷ ابرکامپیوتر Deep Blue در ۱۹ حرکت بهترین شطرنج باز دنیا را شکست بدهد.

اجزای یادگیری تقویتی

اجزای اصلی: عامل و محیط

- عامل ← Agent ← action
- محیط ← environment ← Reward

سیگنال‌های اصلی: حالت: State ، پاداش: Reward ، عمل: Action



مسیر (Trajectory):

$S_0, A_0 \rightarrow S_2, R_1 \rightarrow A_1 \rightarrow R_2, S_2 \rightarrow A_2 \rightarrow R_3, S_3 \rightarrow \dots \rightarrow R_T, S_T$

اصول کار یادگیری تقویتی

عامل از حالت فعلی اطلاع دارد. S_t

عامل در حالت فعلی عملی را انجام می‌دهد. A_t

محیط به حالت جدیدی می‌رود و پاداش را به عامل خواهد داد. R_{t+1} و S_{t+1}

عامل پاداش دریافتی را پردازش می‌کند.

عامل در حالت جدید هم عملی را انجام می‌دهد...

یادگیری تقویتی در حقیقت یک نگاشت از حالت به عمل است.
a s

تابعی که در یک حالت خاص یک عمل خاص را انتخاب می‌کند، سیاست نام دارد.
Policy

هر کاری که انجام می‌دهیم را ارزیابی می‌کنیم Policy evaluation تا به یک سیاست بهینه برسیم.

در رباتی مثل سگ

حالت: زاویه، سرعت و شتاب مفصل‌های مختلف

پاداش: حالت جدید چقدر به حالت مطلوب نزدیک است.

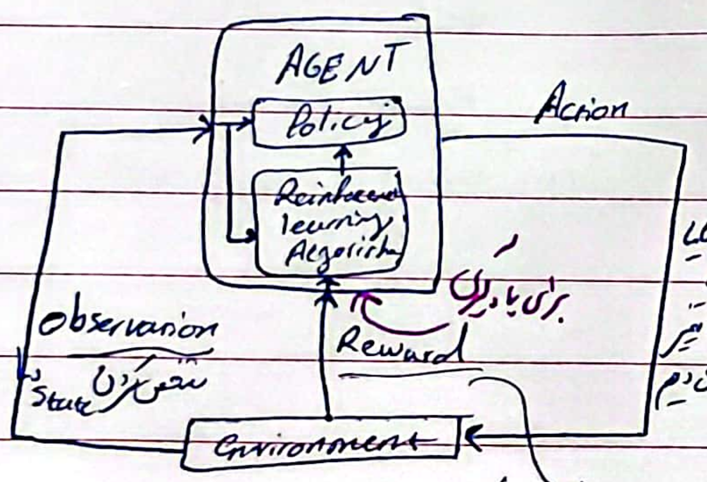
عمل: ولتاژ یا گشتاور اعمالی به موتور مفصل‌ها

سیاست: انتخاب عمل مناسب در حالت‌های مختلف $\theta_1 \in [-90, +90]$

فیدبک حالت $u=-kx$

کار یادگیری تقویتی تعیین بهترین سیاست ممکن است.

تعیین سیاست بهینه از طریق تعامل با محیط



k تعادلی ($u=-kx$)

تغییر بدون دینامیک محیط باعث می‌شود سیاست بهینه از حالت بهینه خارج شود.

الگوریتم‌های یادگیری تقویتی می‌توانند همواره در حال یادگیری باشند.

Action = Function(State Observations)

$u = -kx$

فیدبک حالت

سیاست: نگاشتی از حالت به عمل با پارامترهای قابل تنظیم

یعنی با زبان شما دنبال lookup table هستیم.

یادگیری: الگوریتمی سیستماتیک برای تنظیم پارامترهای سیاست به منظور رسیدن به سیاست بهینه

یادگیری تقویتی: بهبود دادن پارامترهای سیاست از روی پاداش و جریمه

محیط و عامل باتوجه به اهداف تعریف می‌شوند.

در کل هر چیزی به غیر از عامل بخشی از محیط را تشکیل می‌دهد.

محیط می‌تواند هم شبیه‌سازی و هم فیزیکی باشد.

- فیزیکی ← هزینه، دقت بالاتر، سازگاری، ضروری
- شبیه‌سازی ← سرعت یادگیری بالاتر، شبیه‌سازی حالات مختلف، ارزان است

بعد از تعیین مشخصات محیط، باید اهداف تعریف شوند.

با تعیین پاداش مناسب برای عمل‌هایی که عامل انجام می‌دهد اهداف تحقق پیدا خواهند کرد.

پاداش‌ها باید به شکلی باشند که بهبود سیاست و انتخاب عمل‌های بهینه‌تر را نشان می‌دهند.

در یادگیری تقویتی پاداش‌ها اغلب اسکالر و تابعی از حالت (s) و عمل (a) هستند.

Reward = Function (State, Action)

پاداش میزان خوب بودن (goodness) انجام یک عمل در یک حالت خاص را نشان می‌دهد.

پاداش 3- ← جریمه ۳ واحدی

مثال LQR:

$$\dot{x} = Ax + Bu$$

$$J = \frac{1}{2}\int \underbrace{(x^T Q x + u^T R u)}_{\text{پاداش}}$$

(x ← s ، u ← a)

و هدف این است که این پاداش مینیمم شود.

هدف این است که این a ها به گونه‌ای انتخاب شوند که متغیرهای حالت سیستم به صفر همگرا شوند. $x \to 0$

چون می‌خواهیم $x \to 0$ باید عبارت J مینیمم شود. R نشان دهنده میزان مصرف انرژی است و Q هم نشانه سرعت همگرایی به سمت صفر است. می‌خواهیم با کمترین مصرف انرژی به سمت صفر همگرا شویم.

در کل تعیین در فرمولاسیون پاداش وجود ندارد. (مزیت مهم یادگیری تقویتی)

نمایشه sparse تعیین کرد ← نوعی از lookup table

(s) —a, r→ (s')

پاداش می‌تواند برای هر عمل، برای یک انتقال حالت خاص و یا برای هر پایان موفق بازی باشد.

مثال سگوی:

در هر t = 1ms به موتورها دستور می‌دهیم، Action انجام می‌دهیم، برای هر 1ms که آونگ در حالت تعادل باشد، پاداش 1+ اختصاص می‌یابد.

با شکل دهی مناسب پاداش می‌توان به عامل برای سریع شدن یادگیری کمک کرد.

در برخی کاربردها می‌توان اطلاعات و دانش های تجربی را هم در تعریف پاداش لحاظ کرد.

مثل اسلاید 32: Reward: $\alpha v_x - \beta y^2 - \gamma(z - z_r)^2 + 20\,\delta(x-10) - \mu F$

(تشویق: αv_x — نوسان عمودی: βy^2 — ارتفاع مطلوب: z_r — در راستای x، 10 متر حرکت کرد: $20\,\delta(x-10)$ — بیانگر مصرف انرژی: μF)

اگر بیشتر از 10 متر منظور بود بیشتر $u(x-10)$ یا بیشتر چیز دیگه هم تعریف کرد $20(x-10)u(x-10)$

Exploration: جستجو برای یافتن سیاست های بهتر

Exploitation: استفاده از دانش موجود برای تعیین سیاست. اشتباه عمل منجر به سیاست زیر بهینه، عامل به explore نیاز دارد تا کنش های جدید را تست کند و تمام مسیرها را تست کند.

یکی از جنبه های مهم یادگیری تقویتی تعادل بین exploration و exploitation است.

چرا؟ یادگیری آزمایش و خطا با محیط

ما به دنبال اینیم که در طولانی مدت Reward ماکزیمم بشه نه در لحظه.