











Навчання з підкріпленням

Лекція 1: Вступ до RL

Кочура Юрій Петрович
iuriy.kochura@gmail.com
[@y_kochura](#)

Сьогодні

Розгляд основних понять навчання з підкріпленням:

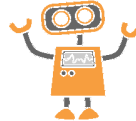
-  Загальна характеристика
-  Цикл взаємодії
-  Гіпотеза винагороди
-  Стан агента
-  Стратегія
-  Функції цінності
-  Класифікація агентів
-  Підзадачі RL

Загальна характеристика

Основним викликом штучного інтелекту та машинного навчання є прийняття правильних рішень в умовах **невизначеності**

Визначення RL

Навчання з підкріпленням (reinforcement learning, RL) — сімейство алгоритмів, які вивчають оптимальну стратегію, метою якої є максимізація загальної винагороди, отриманої агентом при взаємодії з навколишнім середовищем.



- Наприклад, кінцевою винагородою більшості ігор є перемога. Модель навчання з підкріплення може стати експертом у складних іграх, шляхом оцінювання послідовності попередніх ігрових ходів, які в підсумку призвели до перемоги або програшу.

Визначення RL

RL — наука про те, як приймати рішення на основі взаємодій

- Це вимагає від нас задуматися над:
 - часом
 - (довгостроковими) наслідками спричинені діями
 - збором досвіду
 - передбаченням майбутнього
 - боротьбою з невизначеністю

Застосування **RL**

- Ігри ([Atari](#), [AlphaGo](#))
- Робототехніка ([End-to-End Training](#))
- Фінанси
- Взаємодія людини з комп'ютером
- ...

Причини використання **RL**

- Пошук раніше невідомих рішень
 - Приклад, програма, яка може грати в Go краще, ніж будь-яка людина, будь-коли

Причини використання **RL**

- Пошук раніше невідомих рішень
 - Приклад, програма, яка може грати в Go краще, ніж будь-яка людина, будь-коли
- Пошук рішень в режимі реального часу за непередбачених обставин
 - Приклад, робот, який може орієнтуватися на місцевості, яка значно відрізняється від будь-якої очікуваної місцевості

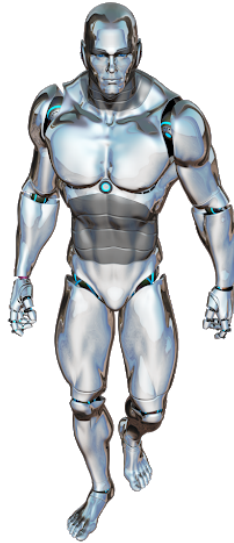
Причини використання **RL**

- Пошук раніше невідомих рішень
 - Приклад, програма, яка може грати в Go краще, ніж будь-яка людина, будь-коли
 - Пошук рішень в режимі реального часу за непередбачених обставин
 - Приклад, робот, який може орієнтуватися на місцевості, яка значно відрізняється від будь-якої очікуваної місцевості
1. Алгоритми навчання з підкріпленням намагаються задовільнити обидва випадки

Причини використання RL

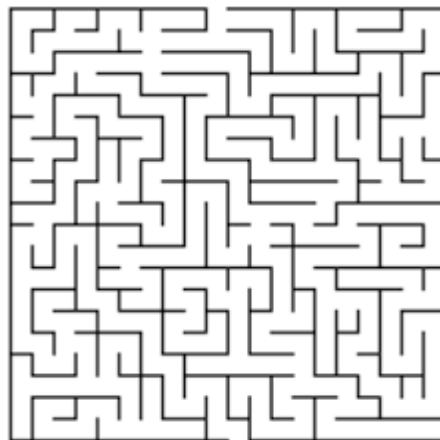
- Пошук раніше невідомих рішень
 - Приклад, програма, яка може грати в Go краще, ніж будь-яка людина, будь-коли
 - Пошук рішень в режимі реального часу за непередбачених обставин
 - Приклад, робот, який може орієнтуватися на місцевості, яка значно відрізняється від будь-якої очікуваної місцевості
1. Алгоритми навчання з підкріпленням намагаються задовільнити обидва випадки
 2. Зверність увагу, що другий пункт стосується не (просто) узагальнення — це більшою мірою про ефективне навчання в режимі реального часу під час взаємодії агента з середовищем

Агент (agent)



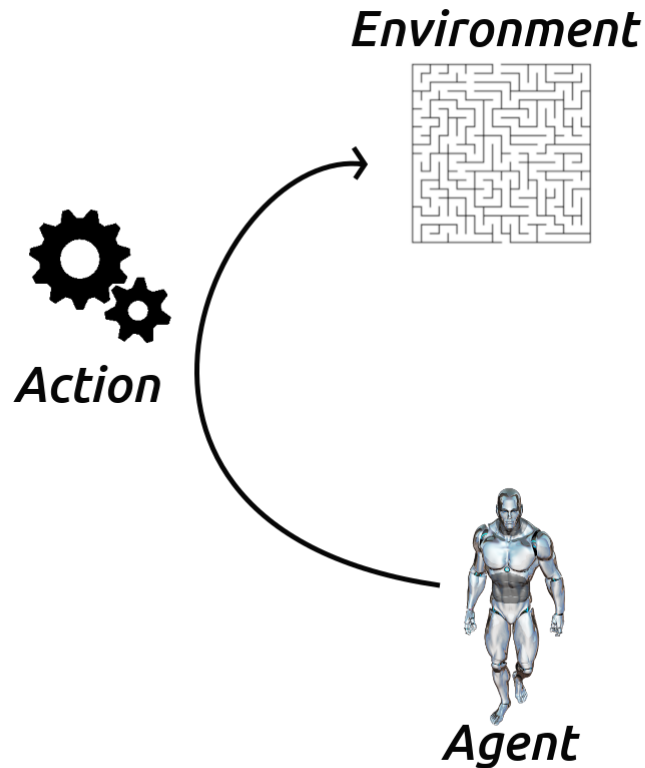
Агент (agent) — це те, що існує окремо від інших речей та використовує певну стратегію (policy) для максимізації очікуваної винагороди (reward), отриманої від переходу між станами середовища (environment).

Середовище (environment)



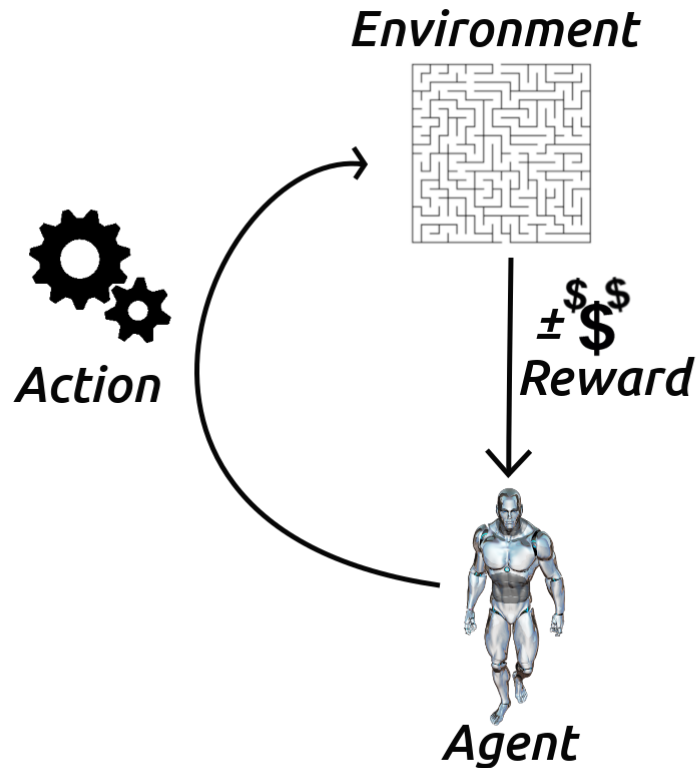
Середовище — це стохастичний та невизначений світ у якому існує та діє агент.

Дія (action)



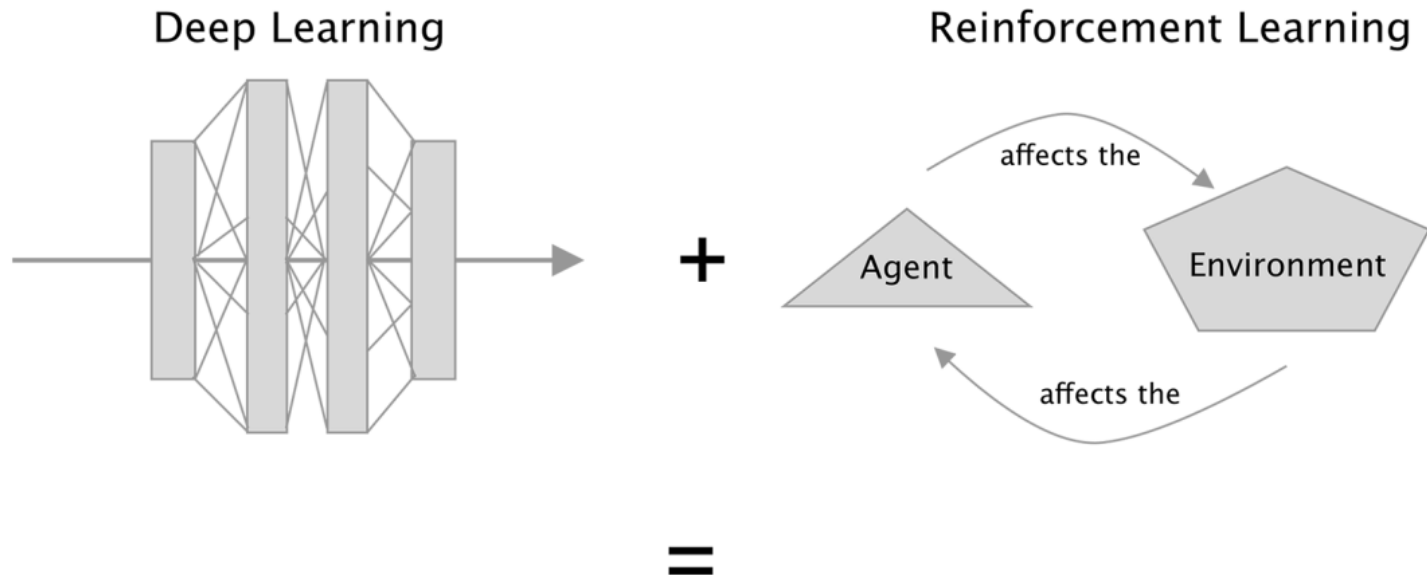
Дія — механізм за допомогою якого агент переходить між дозволеними середовищем станами. Агент обирає дію, використовуючи стратегію.

Винагорода (reward)



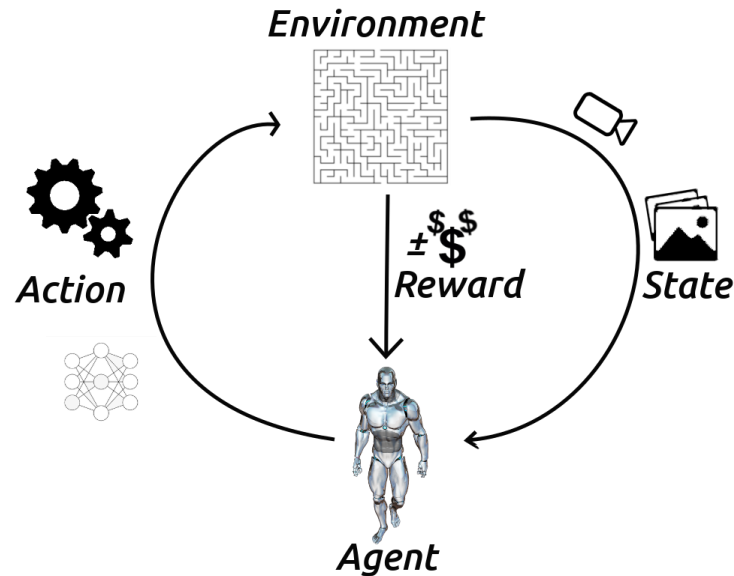
Винагорода — це скалярний сигнал, який отримує агент внаслідок переходу між визначеними станами середовища.

Глибинне RL (Deep RL)



DEEP REINFORCEMENT LEARNING

Глибинне RL (Deep RL)



У глибинному навчанні з підкріпленням агент зазвичай обробляє 2D-зображення згортковими нейронними мережами (CNNs) — це дає йому можливість навчатись "із побаченого" завдяки **наскрізній мережі**, яка перетворює набір пікселів у дії.

Характеристика **RL**

Чим навчання з підкріплення відрізняється від інших парадигм машинного навчання?

- Ніякого контролю, лише сигнал про винагороду

Характеристика **RL**

Чим навчання з підкріплення відрізняється від інших парадигм машинного навчання?

- Ніякого контролю, лише сигнал про винагороду
- Зворотній зв'язок може затримуватися, а не миттєво передаватися

Характеристика **RL**

Чим навчання з підкріплення відрізняється від інших парадигм машинного навчання?

- Ніякого контролю, лише сигнал про винагороду
- Зворотній зв'язок може затримуватися, а не миттєво передаватися
- Час має значення

Характеристика **RL**

Чим навчання з підкріплення відрізняється від інших парадигм машинного навчання?

- Ніякого контролю, лише сигнал про винагороду
- Зворотній зв'язок може затримуватися, а не миттєво передаватися
- Час має значення
- Досвід агента впливає на його наступні дії

Основні поняття **RL**

Формалізм навчання з підкріплення включає у себе такі поняття:

- Середовище (динаміка задачі)

Основні поняття **RL**

Формалізм навчання з підкріплення включає у себе такі поняття:

- Середовище (динаміка задачі)
- Винагорода (визначає мету)

Основні поняття **RL**

Формалізм навчання з підкріплення включає у себе такі поняття:

- Середовище (динаміка задачі)
- Винагорода (визначає мету)
- Агент, який включає:
 - Стан агента (agent state)

Основні поняття RL

Формалізм навчання з підкріплення включає у себе такі поняття:

- Середовище (динаміка задачі)
- Винагорода (визначає мету)
- Агент, який включає:
 - Стан агента (agent state)
 - Стратегію (policy)

Основні поняття RL

Формалізм навчання з підкріплення включає у себе такі поняття:

- Середовище (динаміка задачі)
- Винагорода (визначає мету)
- Агент, який включає:
 - Стан агента (agent state)
 - Стратегію (policy)
 - Q-функцію (state-action value function)

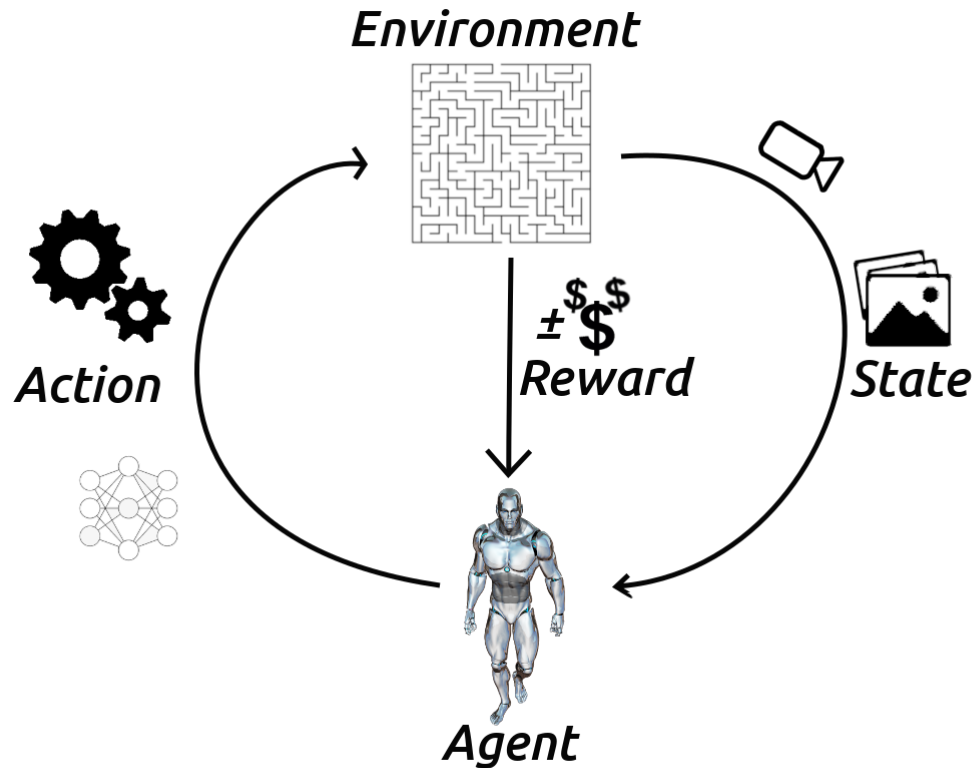
Основні поняття RL

Формалізм навчання з підкріплення включає у себе такі поняття:

- Середовище (динаміка задачі)
- Винагорода (визначає мету)
- Агент, який включає:
 - Стан агента (agent state)
 - Стратегію (policy)
 - Q-функцію (state-action value function)
 - Модель (за бажанням)

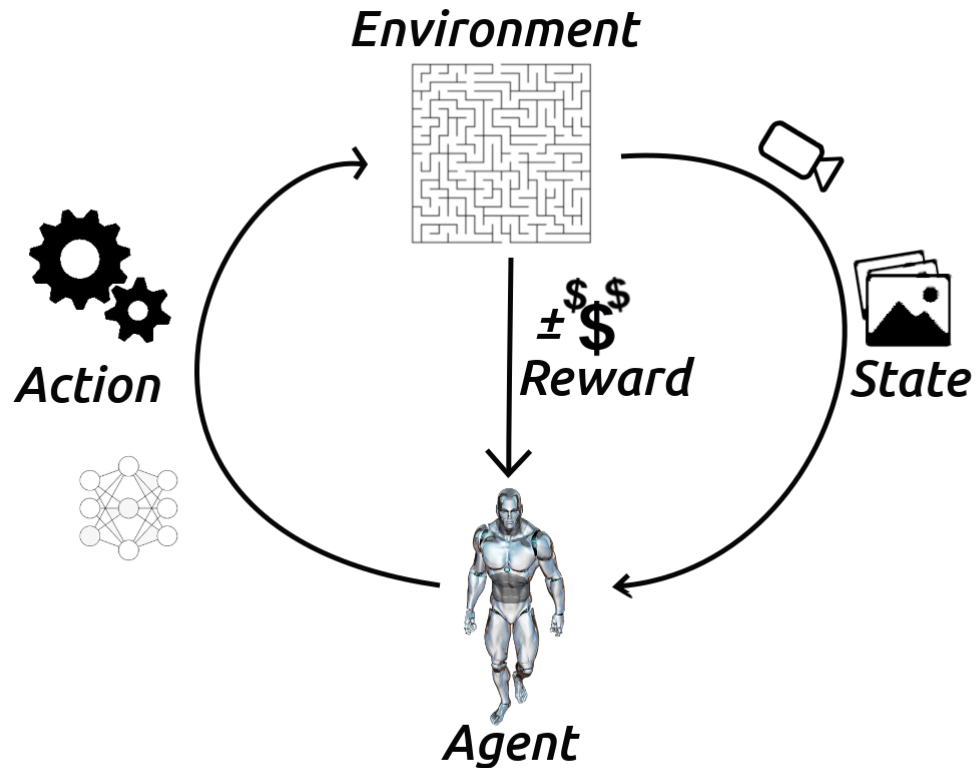
Цикл взаємодії

Цикл взаємодії



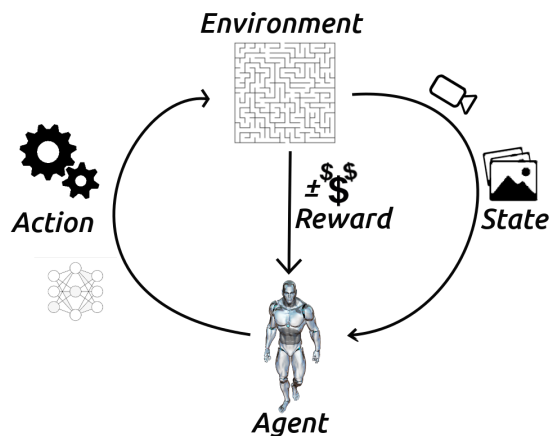
Стан — значення параметрів, що описують поточну конфігурацію середовища. Агент використовує ці параметри для вибору дії.

Цикл взаємодії



Мета — оптимізувати загальну винагороду, отриману агентом при взаємодії з навколишнім середовищем.

Агент та середовище



На кожному кроці в момент часу t агент:

- Отримує спостереження O_t та винагороду R_t
- Виконує дію A_t

Середовище:

- Отримує дію A_t
- Продукує спостереження O_{t+1} та винагороду R_{t+1}

Винагорода

Винагорода R_t — це скалярний сигнал, який отримує агент у якості зворотного зв'язку від середовища.

- Показує, наскільки добре працює агент у момент часу t відповідно до поставленої мети.
- **Завдання агента** — максимізувати кумулятивну винагороду:

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + \dots$$

- G_t називається **загальною винагородою (return)** — сума всіх винагород, які агент розраховує отримати при дотриманні стратегії від певного стану до кінця епізоду.
 - Епізод — кожна спроба агента вивчити середовище.

Гіпотеза винагороди

Навчання з підкріпленням базується на **гіпотезі винагороди**:

"Будь-яка мета може бути формалізована як результат максимізації сукупної винагороди."

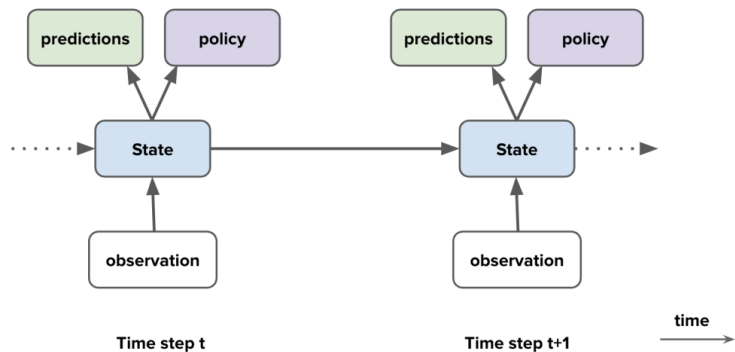
Ключові поняття

Формалізм навчання з підкріплення включає у себе такі поняття:

- Середовище (динаміка задачі)
- Винагорода (визначає мету)
- Агент, який включає:
 - Стан агента
 - Стратегію (policy)
 - Q-функцію, відома також як функція цінності стан-дія (state-action value function)
 - Модель (за бажанням)

Компоненти агента

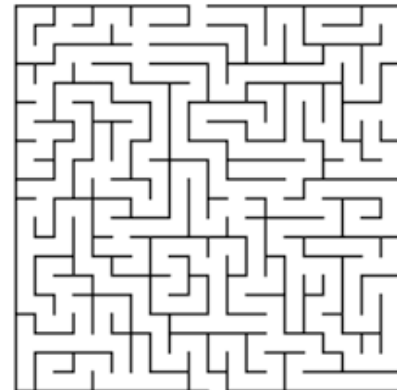
- Стан агента
- Стратегія
- Q-функція
- Модель



Стан середовища

Стан середовища — це внутрішній стан, зазвичай невидимий агенту

- Навіть якщо стан середовища видимий агенту він може містити багато зайвої інформації

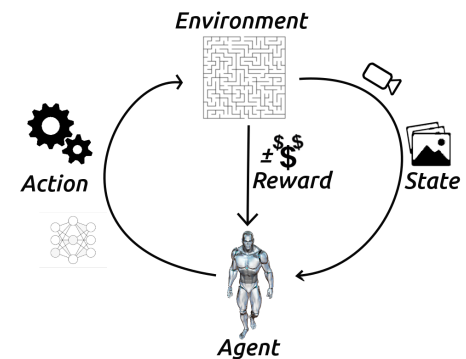


Стан агента

- **Історія** — це послідовність з спостережень O , дій A та винагород R :

$$H_t = O_0, A_0, R_1, O_1, \dots, O_{t-1}, A_{t-1}, R_t, O_t$$

- Історія використовується для побудови **стану агента** S_t



Повністю оглядове середовище

Припустимо, що агент бачить повністю стан середовища. Тоді:

- спостереження = стан середовища
- Стан агента є просто спостереженням:

$$S_t = O_t = \text{стан середовища}$$

У цьому випадку агент бере участь у процесі прийняття рішень Маркова (Markov decision process - MDP). Цей процес названий на честь Андрія Маркова. MDP слугує математичною основою для того, щоб змодельовати прийняття рішення в ситуаціях, де результати є частково випадкові та частково під контролем агента, який приймає рішення.

Марковські процеси прийняття рішень (MDPs)

MDPs надають корисний математичний апарат

Визначення. Процес прийняття рішень є Марковським, якщо

$$p(r, s \mid S_t, A_t) = p(r, s \mid H_t, A_t)$$

- Це означає, що стан містить все, що нам потрібно знати з історії
- Додавання історії не допомагає
- \implies Як тільки стан стане відомим, історію можна буде відкинути
 - Середовище + стан агента — Марковські
 - Історія H_t є Марковською
- Як правило, стан агента S_t є деяким стисненням H_t
- **Примітка:** S_t — стан агента, а не середовища

Частково оглядове середовище

- **Часткова оглядовість**: агент отримує неповну інформацію про стан середовища
 - Камера зору не повідомляє роботу його абсолютне місце розташування
 - Агент, що грає в покер, бачить лише відкриті карти
- Тепер спостереження не є Марковським процесом
- Формально — це **частково оглядовий процес прийняття рішень Маркова** (partially observable Markov decision process, POMDP)
- **Стан середовища** все ще може бути Марковським, але агент цього не знає
- Ми все ще можемо побудувати стан агента, який буде Марковським

Стан агента

- Дії агента залежать від його стану
- **Стан агента** є функцією історії
- Для конкретного стану: $S_t = O_t$
- Більш загально:

$$S_{t+1} = u(S_t, A_t, R_{t+1}, O_{t+1})$$

де u — функція оновлення стану

- Стан агента, як правило, **набагато** менший, ніж стан середовища

Стан агента

Повний стан середовища-лабіринту



Стан агента

Потенційна дальність спостережень агента



Стан агента

Спостереження в іншому місці



Стан агента

Два спостереження неможливо відрізнити



Стан агента

Ці два стани не є Марковськими



Частково оглядове середовище

- Маючи справу з частково оглядовим середовищем, агент може побудувати правильне представлення стану
- Приклади станів агента:
 - Останнє спостереження: $S_t = O_t$ (може бути недостатньо)
 - Уся історія: $S_t = H_t$ (може бути занадто великим)
 - Загальне оновлення: $S_t = u(S_{t-1}, A_{t-1}, R_t, O_t)$ (але як обрати/вивчити u ?)
- Побудувати повністю Марковський стан агента часто є неможливим

Компоненти агента

- Стан агента
- Стратегія (Policy)
- Q-функція
- Модель

Стратегія

- Стратегія визначає поведінку агента
- Стратегія — це план переходу між станом агента до дії
- Детермінована стратегія: $A = \pi(S)$
- Стохастична стратегія: $\pi(A|S) = p(A|S)$

Компоненти агента

- Стан агента
- Стратегія
- Q-функція (функція цінності)
- Модель

Функції цінності

Цінність

Очікувана сукупна винагорода від стану s називається **цінністю (value)**:

$$\begin{aligned} v(s) &= \mathbb{E} [G_t \mid S_t = s] = \\ &= \mathbb{E} [R_{t+1} + R_{t+2} + R_{t+3} + \dots \mid S_t = s] \end{aligned}$$

- Цінність залежить від дій агента
- Метою є **максимізація цінності** $v(s)$ шляхом вибору агентом правильних дій
- Винагороди та цінності визначають **користь** станів та дій (немає контрольованого зворотного зв'язку)
- Зверніть увагу, що загальна винагорода та цінність можуть бути визначені рекурсивно:

$$\begin{aligned} G_t &= R_{t+1} + G_{t+1} \\ v(s) &= \mathbb{E} [R_{t+1} + v(S_{t+1}) \mid S_t = s] \end{aligned}$$

Цінність дій — Q-функція

- 'Q' означає якість (quality)

Q-функція дозволяє оцінити **цінність (якість) дій** :

$$\begin{aligned} q(s, a) &= \mathbb{E} [G_t \mid S_t = s, A_t = a] = \\ &= \mathbb{E} [R_{t+1} + R_{t+2} + R_{t+3} + \dots \mid S_t = s, A_t = a] \end{aligned}$$

Q-функція — функція якості, яка передбачає очікувану загальну винагороду (return) від виконання дій у певному стані та дотриманні заданої стратегії.

- Значення стану та дії буде детальніше розглянуто пізніше

Q-функція

- Фактична функція цінності — це очікувана загальна винагорода:

$$\begin{aligned} v_{\pi}(s) &= \mathbb{E} [G_t \mid S_t = s, \pi] = \\ &= \mathbb{E} [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s, \pi] \end{aligned}$$

- Тут введено фактор знецінювання $\gamma \in [0, 1]$. Чим він менший, тим менше агент замислюється над вигодою від майбутніх своїх дій.
 - Визначає важливість **короткострокових** vs **довгострокових** винагород
- Цінність $v_{\pi}(s)$ залежить від стратегії
- Може використовуватися для оцінки бажаних станів
- Може використовуватися для вибору дій

Функції цінності

- Загальна винагорода має рекурсивну форму: $G_t = R_{t+1} + \gamma G_{t+1}$
- Тому функція цінності може бути записана так:

$$\begin{aligned} v_{\pi}(s) &= \mathbb{E} [R_{t+1} + \gamma G_{t+1} \mid S_t = s, A_t \sim \pi(s)] = \\ &= \mathbb{E} [R_{t+1} + \gamma v_{\pi}(S_{t+1}) \mid S_t = s, A_t \sim \pi(s)] \end{aligned}$$

Тут $A_t = a \sim \pi(s)$ означає, що дія a вибрана на основі стратегії π для стану s (π є детермінованою)

- Це рівняння відоме як **рівняння Беллмана** (Bellman 1957)
- Подібне рівняння можна отримати для оптимальної (= максимально можливої) цінності:

$$v_*(s) = \max_a \mathbb{E}[R_{t+1} + \gamma v_*(S_{t+1}) \mid S_t = s, A_t = a]$$

Апроксимація функції цінності

- Агент постійно апроксимує значення функції цінності
- Для виконання апроксимації існують спеціальні алгоритми
- Завдяки правильній функції цінності агент може поводитися оптимально
- При правильних наближеннях агент може добре поводитися навіть у надзвичайно великих середовищах

Компоненти агента

- Стан агента
- Стратегія
- Q-функція, функція цінності
- Модель

Модель

- **Модель** передбачає поведінку середовища
- Передбачає наступний стан агента \mathcal{P} :

$$\mathcal{P}(s, a, s') \approx p(S_{t+1} = s' \mid S_t = s, A_t = a)$$

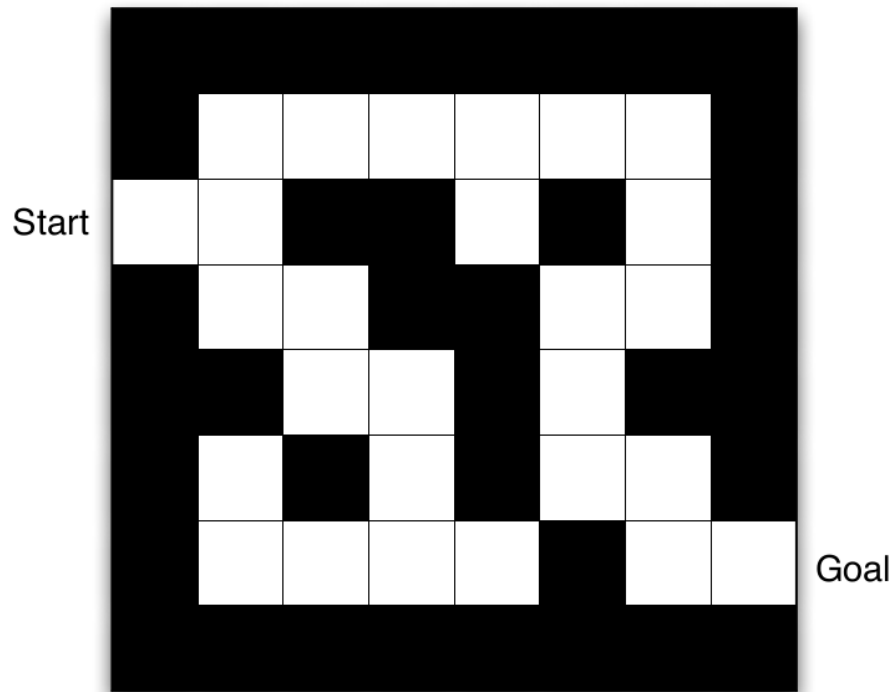
- Або передбачає наступну (миттєву) винагороду \mathcal{R} :

$$\mathcal{R}(s, a) \approx \mathbb{E}[R_{t+1} \mid S_t = s, A_t = a]$$

- Модель не відразу дає нам хорошу стратегію, тому приходится агенту планувати свої дії
- Можуть також розглядатись **стохастичні** (генеративні) моделі

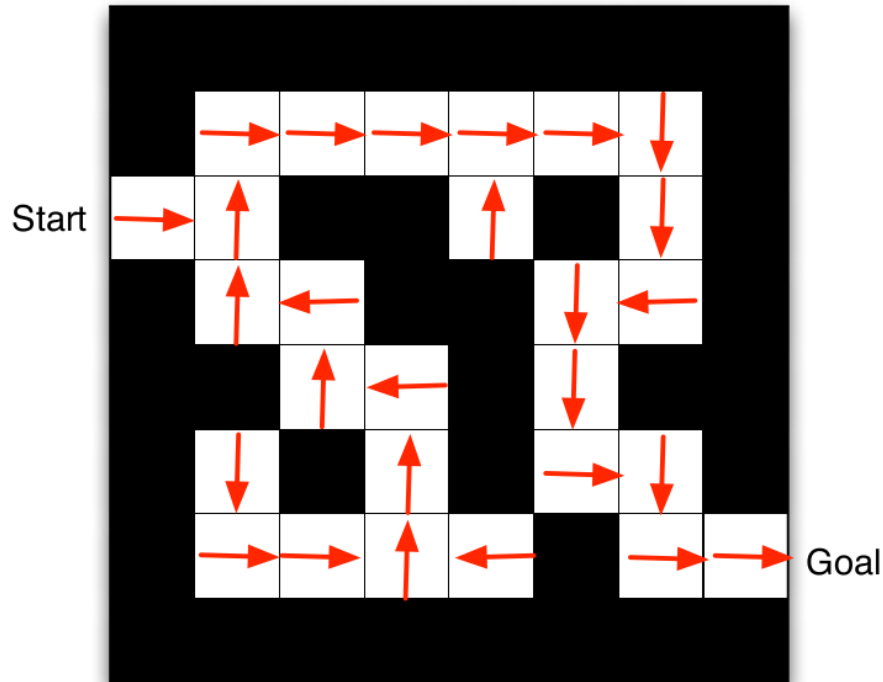
Приклад

Приклад з лабіринтом



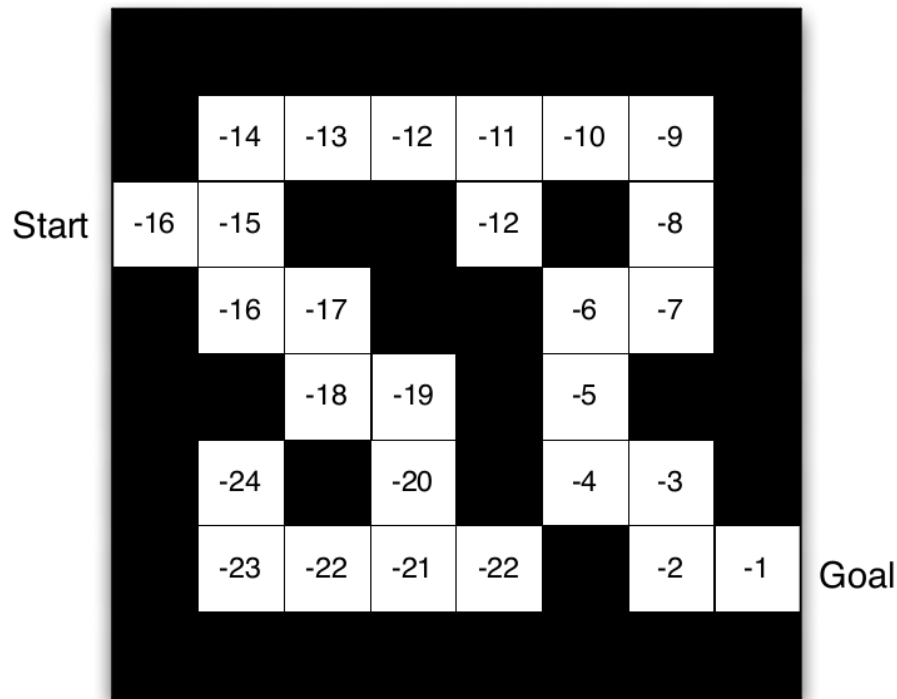
- Винагорода: -1 або 1 за крок
- Дії: N, E, S, W
- Стани: місцезнаходження агента

Приклад з лабіринтом: стратегія



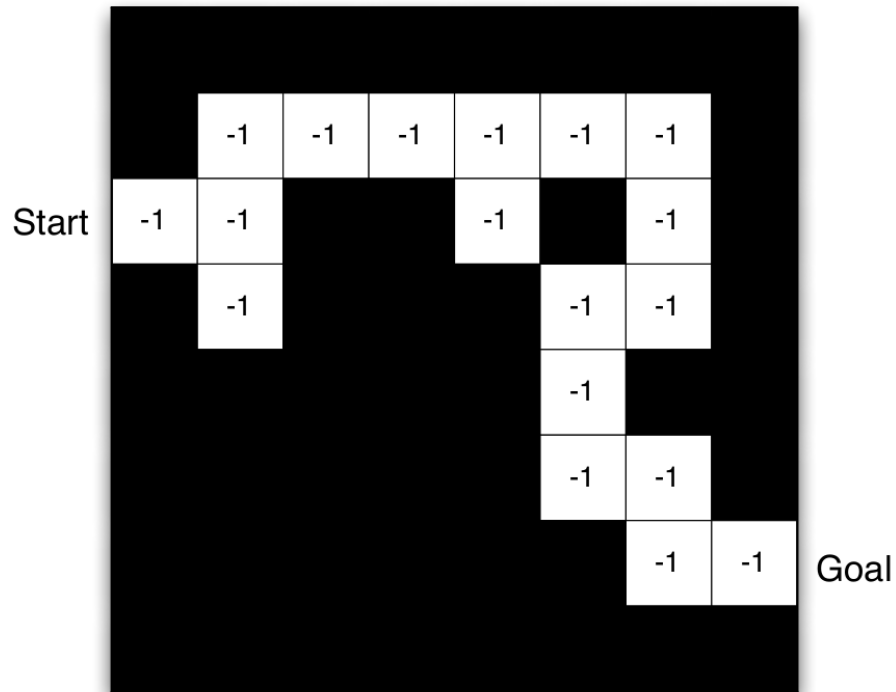
- Стрілки представляють стратегію агента $\pi(s)$ для кожного стану s

Приклад з лабіринтом: функція цінності



- Числа представляють значення $v_{\pi}(s)$ для кожного стану s

Приклад з лабіринтом: модель



- Даний шаблон являє собою модель часткового переходу $\mathcal{P}_{ss'}^a$
- Цифри позначають миттєву винагороду $\mathcal{R}_{ss'}^a$ (у цьому випадку однакова для усіх a та s')

Класифікація агентів

Класифікація агентів

- На основі цінності (Value Based)
 - Функція цінності
 - Відсутня стратегія (неявна)

Класифікація агентів

- На основі цінності (Value Based)
 - Відсутня стратегія (неявна)
 - Функція цінності
- На основі стратегії (Policy Based)
 - Стратегія
 - Відсутня функція цінності

Класифікація агентів

- На основі цінності (Value Based)
 - Відсутня стратегія (неявна)
 - Функція цінності
- На основі стратегії (Policy Based)
 - Стратегія
 - Відсутня функція цінності
- Актор-критик (**Actor Critic**)
 - Стратегія
 - Функція цінності

Класифікація агентів

- Без моделі (Model Free)
 - Стратегія і/або функція цінності
 - Немає моделі

Класифікація агентів

- Без моделі (Model Free)
 - Стратегія і/або функція цінності
 - Немає моделі
- На основі моделі (Model Based)
 - Стратегія і/або функція цінності (за бажанням)
 - Модель

Підзадачі RL

Передбачення та контроль

- **Передбачення:** оцінити майбутнє (для певної стратегії)

Передбачення та контроль

- **Передбачення:** оцінити майбутнє (для певної стратегії)
- **Контроль:** оптимізувати майбутнє (знайти найкращу стратегію)

Передбачення та контроль

- **Передбачення**: оцінити майбутнє (для певної стратегії)
- **Контроль**: оптимізувати майбутнє (знайти найкращу стратегію)

Передбачення та контроль пов'язані між собою:

$$\pi_*(s) = \operatorname{argmax}_a v_\pi(s)$$

Навчання та планування

Два фундаментальні завдання навчання з підкріплення

- Навчання:
 - Середовище спочатку невідоме агенту
 - Агент взаємодіє з середовищем

Навчання та планування

Два фундаментальні завдання навчання з підкріплення

- Навчання:
 - Середовище спочатку невідоме агенту
 - Агент взаємодіє з середовищем
- Планування:
 - Дається (або вивчається) модель середовища
 - Плани агента в цій моделі (без зовнішньої взаємодії)

Навчальні компоненти агента

- Усі компоненти є функціями:
 - Стратегія: $\pi : \mathcal{S} \rightarrow \mathcal{A}$
 - Функція цінності: $v : \mathcal{S} \rightarrow \mathbb{R}$
 - Модель: $m : \mathcal{S} \rightarrow \mathcal{S}$ та/або $r : \mathcal{S} \rightarrow \mathbb{R}$
 - Оновлення стану: $u : \mathcal{S} \times \mathcal{O} \rightarrow \mathcal{S}$
- Наприклад, ми можемо використовувати нейронні мережі та використовувати методи **глибинного навчання** для вивчення цих функцій
- Глибинне навчання — важливий інструмент
- Глибинне навчання з підкріпленням — це багата та активна галузь досліджень

Приклад: Пересування



Emergence of Locomotion Behaviours in Rich Environm...



Share



DeepMind - Emergence of Locomotion Behaviours in Rich Environments

Кінець

Література

- David Silver, [Lecture 1: Introduction to Reinforcement Learning](#)
- Hado van Hasselt, [Lecture Series - Introduction to Reinforcement Learning](#)
- Richard Sutton and Samuel Barto, [Reinforcement Learning: an introduction, second edition](#)
- Richard Sutton [Learning to predict by the methods of temporal differences](#)
- Marco Wiering and Martijn van Otterlo, [Reinforcement Learning](#)
- Watkins Christopher and Peter Dayan, [Q-Learning](#)