Навчання з підкріпленням

Лекція 3: Планування за допомогою динамічного програмування

Кочура Юрій Петрович
iuriy.kochura@gmail.com
@y_kochura

Сьогодні

- Вступ
- Оцінка стратегії
- Ітерація стратегії

Вступ

Що таке динамічне програмування?

Динамічний: послідовний або часовий компонент задачі

Програмування: оптимізація задачі (наприклад, стратегії)

- лінійне програмування
- нелінійне програмування

Що таке динамічне програмування?

Динамічний: послідовний або часовий компонент задачі

Програмування: оптимізація задачі (наприклад, стратегії)

- лінійне програмування
- нелінійне програмування

- Метод вирішення складних задач

Що таке динамічне програмування?

Динамічний: послідовний або часовий компонент задачі

Програмування: оптимізація задачі (наприклад, стратегії)

- лінійне програмування
- нелінійне програмування

- Метод вирішення складних задач
- Шляхом розбиття їх на підзадачі

Що таке динамічне програмування?

Динамічний: послідовний або часовий компонент задачі

Програмування: оптимізація задачі (наприклад, стратегії)

- `лінійне програмування`
- `нелінійне програмування`

- Метод вирішення складних задач
- Шляхом розбиття їх на підзадачі
  - Розв'язати підзадачі

Що таке динамічне програмування?

Динамічний: послідовний або часовий компонент задачі

Програмування: оптимізація задачі (наприклад, стратегії)

```
- лінійне програмування
- нелінійне програмування
```

- Метод вирішення складних задач
- Шляхом розбиття їх на підзадачі
  - Розв’язати підзадачі
  - Об'єднати рішення підзадач

Вимоги до динамічного програмування

Динамічне програмування — це загальний метод вирішення задач, які мають дві властивості:

- Оптимальна підстуктура
  - Виконуєтсья принцип оптимальності

Вимоги до динамічного програмування

Динамічне програмування — це загальний метод вирішення задач, які мають дві властивості:

- Оптимальна підстуктура
  - Виконуєтсья принцип оптимальності
  - Оптимальне рішення можна розкласти на підзадачі

Вимоги до динамічного програмування

Динамічне програмування — це загальний метод вирішення задач, які мають дві властивості:

- Оптимальна підстуктура
  - Виконуєтсья принцип оптимальності
  - Оптимальне рішення можна розкласти на підзадачі
- Перекриття підзадач
  - Підзадачі повторюються багато разів

Вимоги до динамічного програмування

Динамічне програмування — це загальний метод вирішення задач, які мають дві властивості:

- Оптимальна підстуктура
  - Виконуєтсья принцип оптимальності
  - Оптимальне рішення можна розкласти на підзадачі
- Перекриття підзадач
  - Підзадачі повторюються багато разів
  - Розв'язок можна зберегти та повторно використовувати

Вимоги до динамічного програмування

Динамічне програмування — це загальний метод вирішення задач, які мають дві властивості:

- Оптимальна підстуктура
  - Виконуєтсья принцип оптимальності
  - Оптимальне рішення можна розкласти на підзадачі
- Перекриття підзадач
  - Підзадачі повторюються багато разів
  - Розв'язок можна зберегти та повторно використовувати
- Марковський процес прийняття рішень задоволяняє обидві властивості
  - Рівняння Беллмана дає рекурсивне розкладання

Вимоги до динамічного програмування

Динамічне програмування — це загальний метод вирішення задач, які мають дві властивості:

- Оптимальна підстуктура
  - Виконуєтсья принцип оптимальності
  - Оптимальне рішення можна розкласти на підзадачі
- Перекриття підзадач
  - Підзадачі повторюються багато разів
  - Розв'язок можна зберегти та повторно використовувати
- Марковський процес прийняття рішень задоволяняє обидві властивості
  - Рівняння Беллмана дає рекурсивне розкладання
  - Функція цінності зберігає та повторно використовує рішення

Приклад

[ipynb]

Планування за допомогою динамічного програмування

- Динамічне програмування передбачає повне знання MDP
- Використовується для планування в MDP
- Для передбачення:
  - Вхід: MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ та стратегія π
    - або: MRP $\langle \mathcal{S}, \mathcal{P}^\pi, \mathcal{R}^\pi, \gamma \rangle$
  - Вихід: функція цінності v_π
- Або для управління:
  - Вхід: MDP $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$ Вихід: оптимальна функція цінності v_*

Інші застосування динамічного програмування

Динамічне програмування використовується для вирішення багатьох інших проблем, наприклад:

- Алгоритми планування (Scheduling algorithms)
- Алгоритм рядка (String algorithms), напр. вирівнювання послідовності
- Алгоритми на графах (Graph algorithms), напр. алгоритми пошуку найкоротшого шляху
- Графова модель (Graphical models), напр. алгоритм Вітербі
- Біоінформатика, напр. граткові моделі

Оцінка стратегії

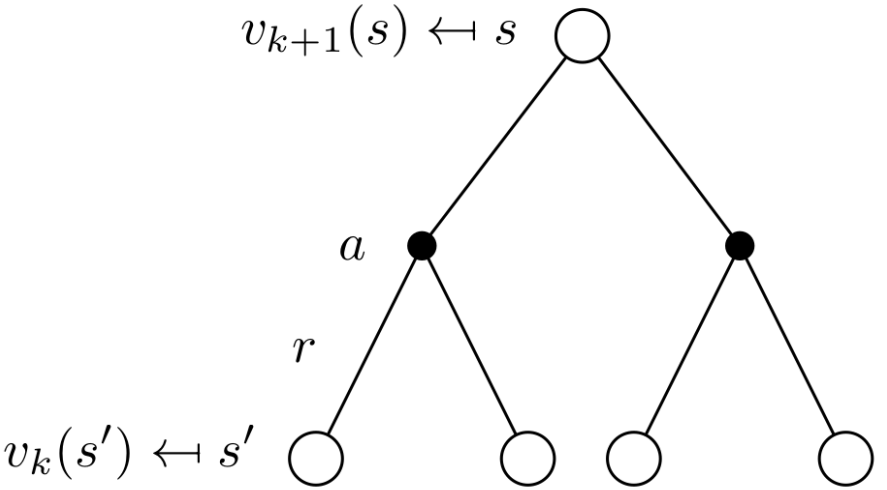
Оцінка стратегії

- Задача: оцінити дану стратегію π
- Розв'язок: ітеративна оцінка рівняння Беллмана
- $v_1 \to v_2 \to \cdots \to v_\pi$

Оцінка стратегії

- Задача: оцінити дану стратегію π
- Розв'язок: ітеративна оцінка рівняння Беллмана
- $v_1 \to v_2 \to \cdots \to v_\pi$
- Синхронна оцінка:
  - На кожній ітерації $k+1$
  - Для всіх станів $s \in \mathcal{S}$
  - Оновити $v_{k+1}(s)$ з $v_k(s')$
- Збіжність до v_π буде доведена пізніше

Оцінка стратегії



$$v_{k+1}(s) = \sum_{a \in \mathcal{A}} \pi(a|s) \left(\mathcal{R}_s^a + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}_{ss'}^a v_k(s') \right)$$

$$\mathbf{v}^{k+1}(s) = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi \mathbf{v}_k$$

Оцінка стратегії

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$ arbitrarily, for $s \in \mathcal{S}$, and $V(terminal)$ to 0

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r \,|\, s, a)\big[r + \gamma V(s')\big]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Джерело: R. S. Sutton and A. G. Barto: Reinforcement Learning: An Introduction

Приклад

actions

	1	2	3
4	5	6	7
8	9	10	11
12	13	14	

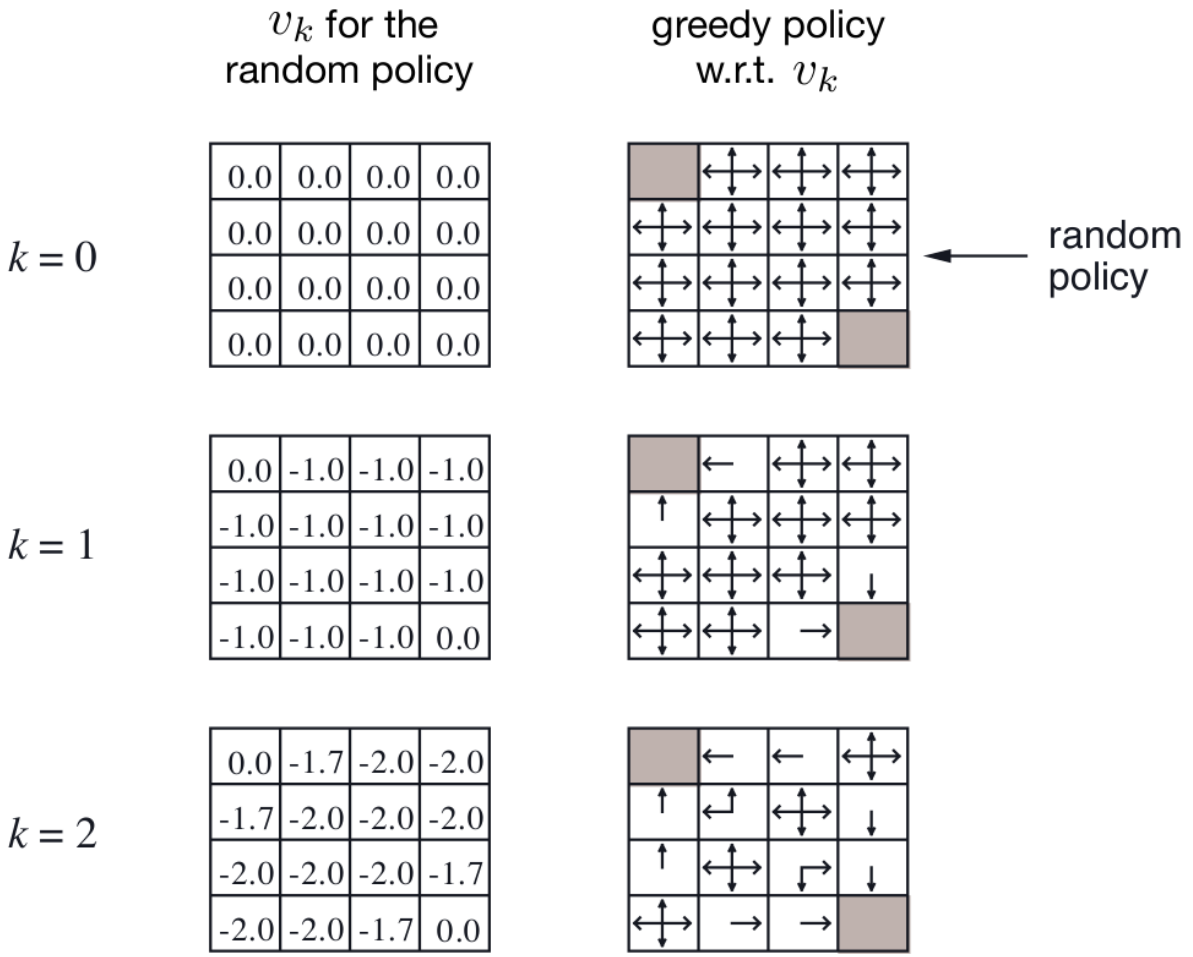
$R_t = -1$
on all transitions

- Епізодичний MDP без знецінювання ($\gamma = 1$)
- Нетермінальні стани: $\mathcal{S} = 1, \cdots, 14$
- Чотири дії можливі для кожного стану: $\mathcal{A} = \text{left}, \text{right}, \text{up}, \text{down}$
- Термінальні стани позначені сірим кольором (два квадрати)
- Дії, що виходять із сітки, залишають стан без змін
- Винагорода становить -1, доки не буде досягнуто термінального стану
- Агент дотримується стратегії:

$$\pi(n|\cdot) = \pi(s|\cdot) = \pi(e|\cdot) = \pi(w|\cdot) = 0.25$$

—
Джерело: R. S. Sutton and A. G. Barto: Reinforcement Learning: An Introduction

Ітеративна оцінка стратегії

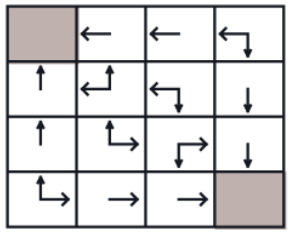


Джерело: R. S. Sutton and A. G. Barto: Reinforcement Learning: An Introduction

Ітеративна оцінка стратегії

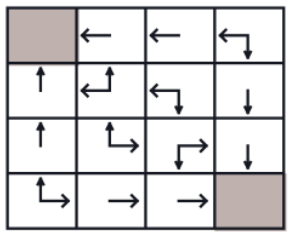
$k = 3$

0.0	-2.4	-2.9	-3.0
-2.4	-2.9	-3.0	-2.9
-2.9	-3.0	-2.9	-2.4
-3.0	-2.9	-2.4	0.0



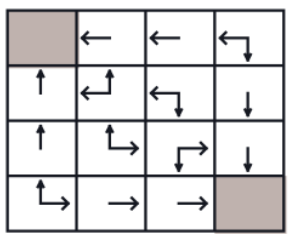
$k = 10$

0.0	-6.1	-8.4	-9.0
-6.1	-7.7	-8.4	-8.4
-8.4	-8.4	-7.7	-6.1
-9.0	-8.4	-6.1	0.0



$k = \infty$

0.0	-14.	-20.	-22.
-14.	-18.	-20.	-20.
-20.	-20.	-18.	-14.
-22.	-20.	-14.	0.0



optimal policy

Джерело: R. S. Sutton and A. G. Barto: Reinforcement Learning: An Introduction

Ітерація стратегії

Як вдосконалити стратегію?

- Дано стратегію π

Як вдосконалити стратегію?

- Дано стратегію π
  - Оцінка стратегії π:

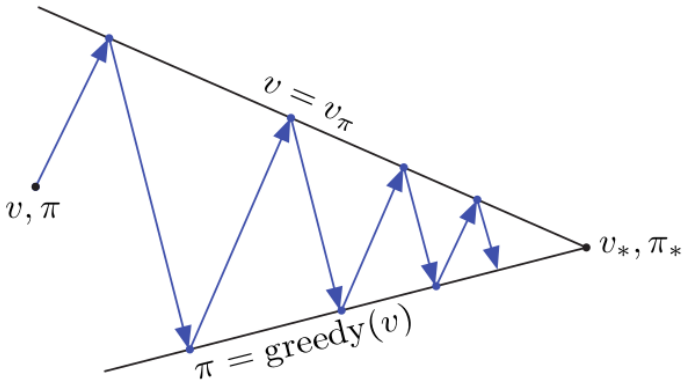
$$v(s) = \mathbb{E}\left[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots \mid S_t = s\right]$$

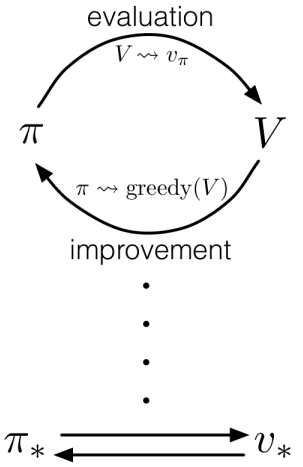
  - Вдосконалення стратегії, діючи жадібно щодо v_π:

$$\pi' = \text{greedy}(v_\pi)$$

- У розглянутому попередньому прикладі вдосконалена стратегія була оптимальною: $\pi' = \pi_*$
- У загальному випадку потрібно виконати більше ітерацій вдосконалення стратегії
- Процес ітеративної оцінки стратегії завжди збігається до π_*

Ітерація стратегії





- Оцінка стратегії: обчислення v_π
  - Ітеративна оцінка стратегії
- Вдосконалення стратегії: отримання $\pi' \geq \pi$
  - Жадівне вдосконалення стратегії

—

Джерело: R. S. Sutton and A. G. Barto: Reinforcement Learning: An Introduction

Кінець

Приклад

[Динамічне програмування vs Монте-Карло]

Література

- David Silver, Lecture 3: Planning by Dynamic Programming
- R. S. Sutton and A. G. Barto: Reinforcement Learning: An Introduction. Ch. 4 - Dynamic Programming