

# Procesy decyzyjne Markowa (2). Uczenie

Paweł Rychlikowski

Instytut Informatyki UWr

24 maja 2019

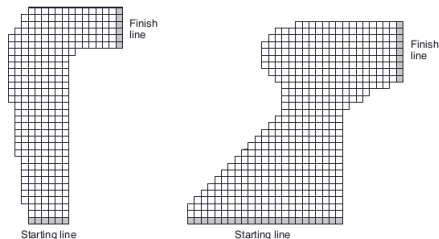
# Podsumowanie algorytmów

- Wartościowanie polityki (policy evaluation):  $(\text{MDP}, \pi) \rightarrow V_\pi$
- Iteracja wartości (value iteration):  $(\text{MDP}) \rightarrow V_{\text{opt}}, \pi_{\text{opt}}$

(chwila opowiadania przy tablicy)

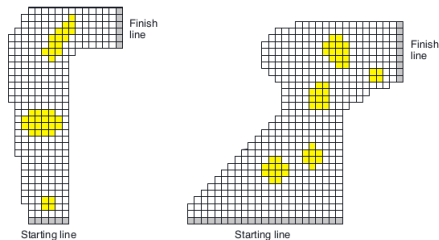
Będą jeszcze inne, w których założymy nieznaną światu...

# Przykład. Wyścigi samochodzików.



- Prędkość autka jest wektorem  
 $(dx, dy) \in \{-3, -2, \dots, 2, 3\} \times \{-3, -2, \dots, 2, 3\}$
- Akcja: zmiana prędkości (każda składowa o co najwyżej 1)
- Celem jest (przejechać) przez metę (możemy to uprościć poszerzając metę i mówiąc, że celem jest dotarcie do piksela mety)
- W pełni deterministyczny świat (BFS, A\*?)

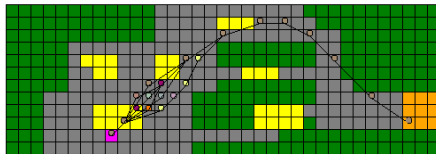
# Wyścigi samochodzików. (2)



- Dodajemy plamy po oleju
- Ruch z pola oleju dodaje dodatkową składową losową do prędkości (znamy rozkład).

W tym momencie klasyczne MDP + algorytm Bellmana (czyli iteracji wartości) powinny dać dobry wynik.

# Wynik algorytmu Value Iteration



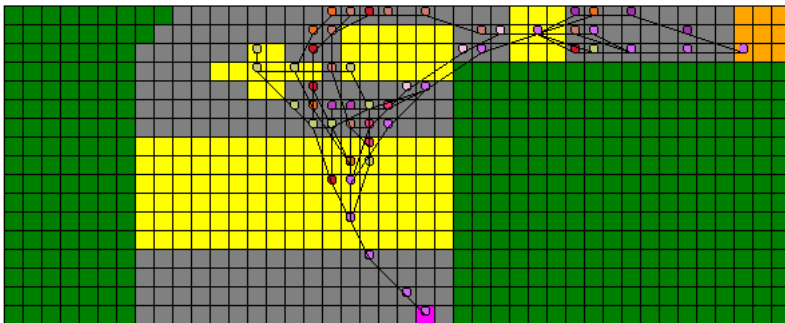
Zwróćmy uwagę, że bez żadnych dodatkowych obliczeń można umieszczać w innych miejscach punkt startowy.

- Fajnie jest dojechać na metę. (+100)
- Ale jeszcze fajniej nie dać się zabić. (-100?)

## Uwaga

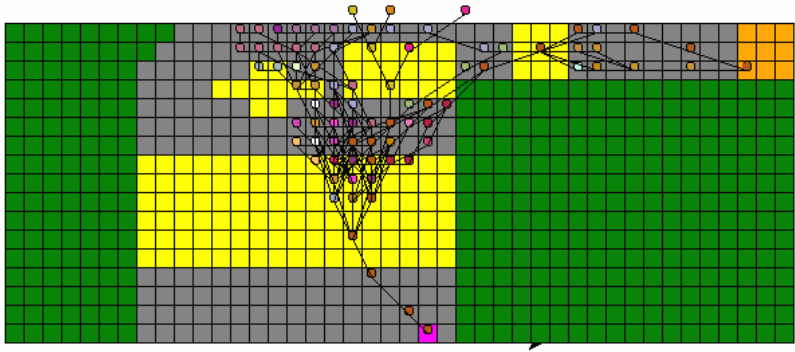
Pamiętamy, że monotoniczna zmiana funkcji wypłaty:

1. nie zmienia wartości MiniMax-owej gry,
2. może zmienić ExpectMinMax



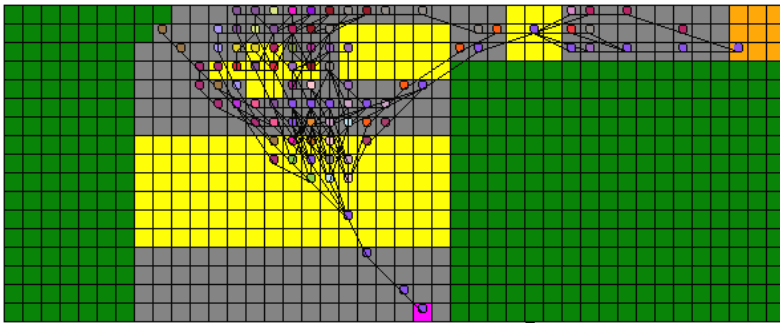
**Pytanie:** Czego spodziewamy się, jeżeli zamienimy karę na wypadek na 10000?

Kara=100





Kara=10000



# Wyścigi samochodzików. (3)

- Problem: dużo większa plansza, dużo większa liczba stanów.
- **Pomysł 1:** położenie „rozmyte”, na przykład w kwadracie  $10 \times 10$  pikseli.
- **Pomysł 2:** dodatkowo informacja, czy jestem 1, 2, czy 3 raz w takim kwadracie ( $3 < 100$ )

**Fundamentalny problem:** nie znamy mechaniki takiego świata (i wielu innych)

# Wyścigi samochodzików. Float

- Prędkość autka jest wektorem  $(v \cos(d), v \sin(d))$ ,
- Możemy zmieniać  $d$  (skręcać), oraz  $v$  (przyśpieszać, hamować)
- Celem jest meta.
- W pełni deterministyczny świat, ale **bardzo duża liczba stanów, zawierających liczby float**)

- Możemy stworzyć **stan abstrakcyjny** i opisać mechanikę świata dla takich stanów
- Oczywiście będzie ona niedeterministyczna, bo nigdy nie będziemy wiedzieć, czy zmiana w świecie float-ów przenosi się na zmianę w świecie int-ów.

## Uwaga

Możemy myśleć o tym, że modelujemy błędy pomiarowe (int zamiast float) za pomocą losowości.

- Zakładamy, że nie dysponujemy modelem (czyli przejściami, prawdopodobieństwami i nagrodami)
- Możemy wszakże wykonywać pewne eksperymenty w naszym systemie, w wyniku których zdobywamy wiedzę **jak nam poszło**

## Uwaga

Zauważmy, że to pasuje do naszych samochodzików z rozmytymi stanami (eksperyment przeprowadzamy na prawdziwym modelu, ale obserwujemy model „rozmyty”)

# Ogólny schemat uczenia ze wzmocnieniem

Dla  $t \in 1, 2, 3, \dots$

- Wybieramy akcję  $a_t = \pi_{act}(s_{t-1})$  (**jak?**)
- Wykonujemy akcję i obserwujemy nowy stan  $s_t$
- Uaktualniamy parametry (**jak?**)

- Estymujemy model podczas eksperymentów
- Rozwiązujemy **wyszacowane** MDP.

## Szacowany MDP

- $\hat{T}(s, a, s') = \frac{\text{cnt}(s, a, s')}{\text{cnt}(s, a)}$
- Nagroda: średnie **r** dla zaobserwowanych **s a r s'**

- Jaką polityką mamy badać świat?
  - a) Wybierającą akcje losowo (strata czasu?)
  - b) Wybierającą akcje prowadzącą do stanu o najlepszej wartości  $V$  (ale możemy się zafiksować na nieoptymalnej ścieżce)
- Wybór między a) i b) to wybór między eksploracją i eksploatacją
  - Pamiętamy: strategia  $\epsilon$ -zachłanna.

Możemy przeplatać etapy wyznaczania modelu i rozwiązywania MDP (bo w kolejnych iteracjach mamy możliwość wykorzystania lepszej strategii eksploatacyjnej).



Dlaczego przeplatanie estymacji modelu i wyznaczania optymalnej polityki może nam pomóc osiągnąć lepszy rezultat?

Mówiliśmy o metodach Monte Carlo, w których przeprowadzamy eksperymenty (losowe przebiegi), żeby estymować (nieznane) parametry MDP.

- Nowy cel: od razu liczyć  $Q(s, a)$ , nie przejmując się tworzeniem modelu.
- Zaczniemy od obliczenia  $Q_{\pi}(s, a)$

## Definicja (przypomnienie)

$Q_\pi(s, a)$  to oczekiwana sumaryczna nagroda, jaką otrzymamy wykonując w stanie  $s$  akcję  $a$ , a następnie postępując zgodnie z polityką  $\pi$

- Użyteczność (dla konkretnego przebiegu):

$$u_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$$

- $\hat{Q}_\pi(s, a) = \text{średnie } u_t$ , gdzie  $s_{t-1} = s$ ,  $a_t = a$

- Zamiast liczyć średnią z całości, można myśleć o uaktualnianiu średniej wraz z pojawieniem się kolejnej informacji.
- Niech:  $\eta = \frac{1}{1+\text{cnt}(s,a)}$
- $\hat{Q}_\pi(s, a) \leftarrow (1 - \eta)\hat{Q}_\pi(s, a) + \eta u$  (gdzie  $u$  jest użytecznością zaobserwowaną w konkretnym przebiegu)

Sprawdźmy, czy to się zgadza.

# Bezmodelowe Monte Carlo – inne sformułowanie (2)

$$\hat{Q}_{\pi}(s, a) \leftarrow (1 - \eta)\hat{Q}_{\pi}(s, a) + \eta u$$

- $u$  jest **zaobserwowaną** użytecznością
- $\hat{Q}_{\pi}(s, a)$  jest naszą predykcją.

Reguła ta minimalizuje odległość między predykcją a obserwacją.

## Uwaga

W informatyce często, rozwiązując jakieś zadanie, korzystamy z niedoskonałego (tymczasowego) rozwiązania, żeby rozwiązać zadanie lepiej.

## Przykład

Szukanie dobrych i złych słów (analizujemy wpisy na jakimś forum), na początku znamy kilka przykładowych dobrych i złych słów.

Będziemy używać  $Q$  (poprzedniej wartości) do obliczenia nowego  $Q$

# Bootstrapping: SARSA

Obserwujemy ciąg akcji i nagród:

$$s_0, a_1, r_1, s_1, a_2, r_2, s_2, \dots$$

- Uaktualnianie Monte Carlo:

$$\hat{Q}_\pi(s, a) \leftarrow (1 - \eta)\hat{Q}_\pi(s, a) + \eta u$$

- SARSA (obserwujemy  $s, a, r, s', a'$ ):

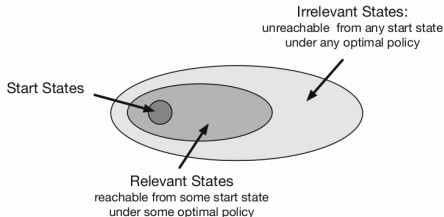
$$\hat{Q}_\pi(s, a) \leftarrow (1 - \eta)\hat{Q}_\pi(s, a) + \eta(r + \gamma\hat{Q}_\pi(s', a'))$$

W algorytmie SARSA zamiast konkretnego (zaobserwowanego)  $u$  bierzemy zaobserwowaną jego  $i$  pierwszą część ( $r$ ) i estymowaną resztę (zielony jest  $cel$ )

## Uwaga

Nie musimy czekać do końca epizodu, żeby uaktualnić wartość  $Q$ !

# Value iteration vs. SARSA



źródło: Sutton, Reinforcement Learning. An introduction

- VI liczy wartości dla stanów „nieoptymalnych”
- VI liczy wartości dla stanów nieosiągalnych (łatwo wymyślić dla autek taką kombinację prędkości i położenia, która jest bezużyteczna)

W momencie, gdy operujemy przebiegami, być może sensownymi, to koncentrujemy się na estymacji rzeczy użytecznych (a na pewno na **osiągalnych!**)



## Uwaga

SARSA estymuje  $Q_{\pi}(s, a)$ . Najbardziej naturalnym celem jest znajomość  $Q_{\text{opt}}$ .

- Algorytm umożliwiający bezpośrednie obliczanie  $Q_{\text{opt}}$  to właśnie **Q-learning**.
- Również radzimy sobie bez modelu.

Standardowy kształt reguły:

$$Q(s, a) \leftarrow (1 - \eta)Q(s, a) + \eta \text{ cel}$$

Celem jest  $r + \gamma V_{\text{opt}}(s')$

Natomiast:

$$V_{\text{opt}}(s') = \max_{a' \in \text{Actions}(s')} Q_{\text{opt}}(s', a')$$

## Algorytm Q-learning

Dla zaobserwowanych  $s, a, r, s'$ :

$$Q(s, a) \leftarrow (1 - \eta)Q(s, a) + \eta(r + \gamma \max_{a' \in \text{Actions}(s')} Q_{\text{opt}}(s', a'))$$

Obok wnioskowania i przeszukiwania jeden z głównych **silników** sztucznej inteligencji.

Użyteczne między innymi w sytuacjach o których ostatnio mówiliśmy, gdy nie chcemy **pamiętać** wartości  $Q(s, a)$  lecz umieć ją **obliczyć**

## Fragment oceny opisowej ze świadectwa dziecka

... umie odróżniać psa od kota.

Dane uczące



## Fragment oceny opisowej ze świadectwa dziecka

... umie odróżniać psa od kota.

Dane uczące i dane testowe



?



?



?



?



?

- Spróbujemy usystematyzować nasze intuicje związane z uczeniem.
- Co wiemy:
  - a. Mamy przykłady, próbujemy je uogólnić.
  - b. Jedno z podstawowych zadań: klasyfikacja, czyli przypisanie **przypadkowi** jego **klasy**.
  - c. Przykłady:
    - Ocena, czy **mail** należy do **spam** czy też **nie-spam**.
    - Wybór **rasy** dla **zdjęcia psa**
    - Czy **napis** jest **adresem e-mail**, **url-em**, **nazwą firmy**, **imieniem i nazwiskiem**, **czymś innym**?

- Oczekiwanym wynikiem może być liczba rzeczywista.
- Przykłady:
  - predycja ceny nieruchomości,
  - ocena masy ciała (gdy znamy płeć i wzrost),
  - przewidywanie zużycia wody (dla MPWiK), gdy znamy temperaturę i dzień tygodnia

## Uwaga

Zadanie rozróżnienia kota i psa byłoby łatwiejsze, gdybyśmy mieli dane nie obrazki, lecz cechy zwierzęcia

Przykłady?

- masa ciała,
- odległość między oczami,
- odległość nosa i oka,
- długość włosów,
- długość wąsów,
- długość ogona

użyteczne mogłyby być też na przykład proporcje różnych cech i inne **wtórne cechy** (wyliczone z podstawowych)



# Cechy (wektor cech)

- Abstrakcyjny **obiekt** możemy zamienić na **wektor cech**.
- Dla (zabawkowego) klasyfikatora **czy-email?**, możemy mieć:
  - Czy długość większa od 10?
  - Jaki procent znaków to znaki alfanumeryczne?
  - Czy zawiera @?
  - Czy kończy się na **.com** (i tak dalej)

Cechami mogą być też na przykład wartości składowych pikseli, kolejne wartości pliku wave, zbiory pomiarów wszystkich wodomierzy z ostatniej doby, itd.

Dla obiektu  $x$  wektor cech oznaczamy często jako  $(\phi_1(x), \dots, \phi_n(x))$ .