WSI lab 6

> Proszę zaimplementować algorytm Q-Learning i użyć go do wyznaczenia polityki decyzyjnej dla problemu FrozenLake8x8 (w wersji domyślnej, czyli z włączonym poślizgiem). W problemie chodzi o to, aby agent przedostał się przez zamarznięte jezioro z pozycji 'S' do pozycji 'G' unikając punktów 'H'. Symulator dla tego problemu można pobrać z podanej strony lub napisać własny o takiej samej funkcjonalności.
>
> Oprócz zbadania domyślnego sposobu nagradzania (1 za dojście do celu, 0 w przeciwnym przypadku) proszę zaproponować własny system nagród i kar, po czym porównać osiągane wyniki z wynikami systemu domyślnego.
>
> Za wynik (podczas testowania) uznajemy procent dojść do celu w 1000 prób (10x więcej prób używamy w treningu). W każdej próbie można wykonać maksymalnie 200 akcji.

Liczba epizodów jest stała i wynosi `10 000` w trakcie trenowania. Podczas testowania liczba epizodów wynosiła `1000` . W każdej próbie liczba akcji wynosiła 200 maksymalna. Wszystkie testy odbyły się na domyślnej mapie `8x8` .

```
"8x8": [
        "SFFFFFFF",
        "FFFFFFFF",
        "FFFHFFFF",
        "FFFFFHFF",
        "FFFHFFFF",
        "FHHFFFHF",
        "FHFFHFHF",
        "FFFHFFFG",
    ]
```

Nr	learning rate - beta	epsilon	współczynnik dyskontowania - gamma	wyniki [%] - domyślny system nagród i kar	wyniki [%] - własny system nagród i kar
0	0,9	0,1	0,8	8,1	8,4
1	0,9	0,1	0,7	4,2	10,1
2	0,9	0,1	0,99	15,5	12

Nr	learning rate - beta	epsilon	współczynnik dyskontowania - gamma	wyniki [%] - domyślny system nagród i kar	wyniki [%] - własny system nagród i kar
3	0,5	0,1	0,99	45,6	39,6
4	0,25	0,1	0,99	21,7	37,2
5	0,125	0,1	0,99	61,8	30,4
6	0,0625	0,1	0,99	51,2	50,2
7	0,0625	0,1	0,99	47,3	46,8
8	0,0625	0,25	0,99	59,8	63,9
9	0,0625	0,3	0,99	60,8	40,3

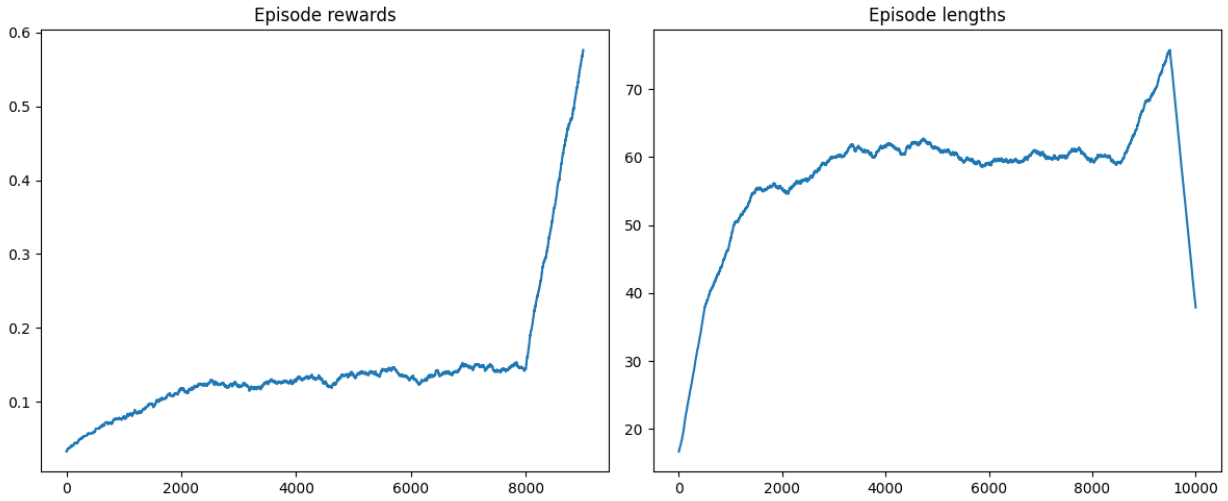
Domyślny własny system nagród i kar

Na podstawie przeprowadzonych testów łatwo zauważyć, że algorytm osiągnął najlepsze wyniki dla $\epsilon = 0,1$ $\gamma = 0,99$ $\beta = 0,125$, gdzie uzyskał 61,8% skuteczności. Łatwo zauważyć, że największy wpływ ma współczynnik dyskontowania w następnej kolejności współczynnik uczenia się. Natomiast dla ustalonych parametrów, wpływ epsilonu - zwanego również czasem współczynnikiem eksploracji, jest stosunkowo niewielki.

Algorytm nie osiąga wyższej skuteczności ze względu na poślizg na lodzie. Jest to symulacja niepewności środowiska, która sprawia, że wybrane akcje w pewnym stanie, mogą powodować przejścia do innych, różnych stanów.

Można to dobrze zaobserwować w tabeli funckji Q dla pola w 7 wierszu i 8 kolumnie, które jest oznaczone kolorem czerownym. Wydaje się, że najlepszym ruchem jest ruch w dół, jednakże przez ów poślizg agent może wpaść w dziurę, czego stara sie unikać. Zatem najbezpieczniejszą opcją jest wybór ruchu w prawo, gdzie niezależnie od następnego stanu, nie może przegrać w jednym ruchu.

```
# Dla pola (w,k) = (7,8)
[0.6    0.611   0.956   0.63] #ruchy kolejno: w lewo, w dół, w prawo, do góry
```



Wykres przedstawia wartość nagrody na epizod oraz długość epizodu z użyciem własnego systemu oceny dla parametrów z wiersza oznaczonego numerem 5

Własny system nagród i kar

Mój system nagród i kar działał następująco:

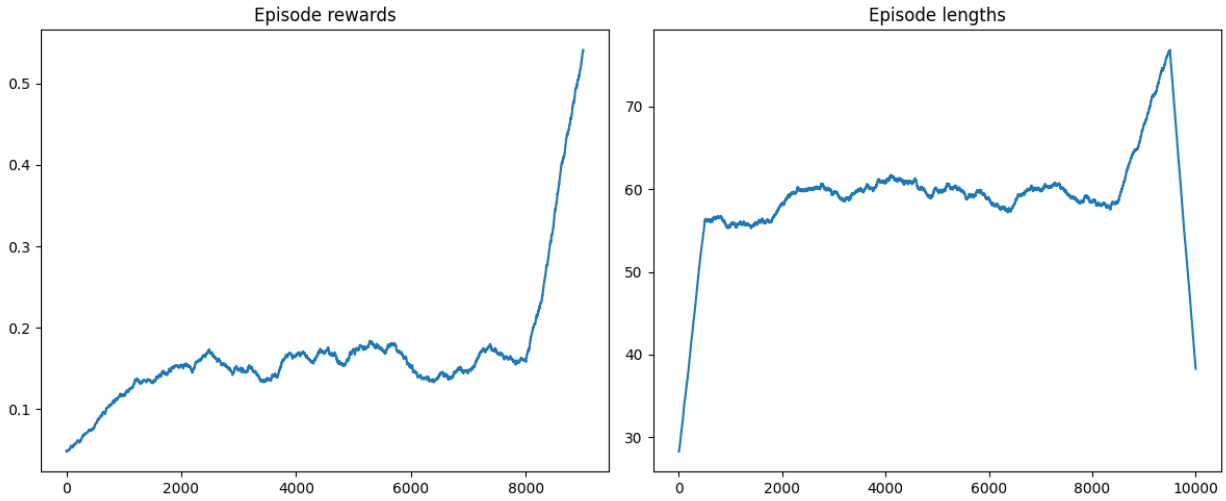
- Gdy agent dotrze do stanu absorbującego i jest to cel, otrzymuje 10 punktów
- Gdy agent dotrze do stanu absorbującego i nie jest to cel, otrzymuje -1 punkt
- Gdy agent po wykonaniu akcji, jest w tym samym stanie, otrzymuje -0,1 punkt
- W przeciwnym przypadku, agent otrzymuje zero punktów

Po sprawdzaniu różnych wielkości nagrody i kar zdecydowałem się na właśnie te wartości aby agent jak najbardziej utrwalił ścieżkę prowadzącą do celu, przy jednoczesnym unikaniu stania w miejscu.

W najlepszym przypadku algorytm z własnym system nagród osiągnął lepsze wyniki w porównaniu do algorytmu z domyślnym systemem oceny.

Przy użyciu własnego systemu nagród występuje analogiczna sytuacja co do wartości funkcji Q, gdzie dla pola (7,8) najbardziej optymalny jest wybór ruchu w prawo

```
# Dla pola (w,k) = (7,8)
[6.868    5.859   8.385   5.405] #ruchy kolejno: w lewo, w dół, w prawo, do góry
```



Wykres przedstawia wartość nagrody na epizod oraz długość epizodu z użyciem własnego systemu oceny dla parametrów z wiersza oznaczonego numerem 9

Wnioski

W przypadku wielu uruchomień algorytm dla tych samych hiperparametrów, zwraca różne wyniki. Jest to spowodowane losowością w wybieraniu akcji podczas eksploracji oraz niepewnością środowiska. Algorytm musi sobie z tym poradzić w trakcie nauki co sprawia, że ciężko jest otrzymać wyższe wyniki. Spory wpływ na wynik ma również rozłożenie dziur na planszy, które mogą dodatkowo utrudniać proces uczenia się.

Pomimo trudności algorytm jest w stanie, osiągać wyniki rzędu 60%. Należy również zauważyć, że przy braku poślizgu, oba algorytmy działają wzorcowo i osiągają 100% dokładność.