Wprowadzenie do sztucznej inteligencji - ćwiczenie 6

Kacper Marchlewicz

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.

Z uwagi na długi czas obliczeń dobranie parametrów było obliczane jako średnia z 10 uruchomień programu, wartość startowa epsilon wynosiła 0,1.

Dobranie parametrów beta i gamma dla bazowego systemu nagród:

		beta					
		0,1	0,25	0,5	0,75	0,9	1
gamma	0,1	2,83%	1,84%	2,05%	1,53%	1,86%	3,09%
	0,25	2,44%	2,93%	2,67%	2,96%	2,1%	3,23%
	0,5	6,93%	7,77%	5,47%	7,93%	5,02%	7,09%
	0,75	21,3%	13,0%	10,5%	19,7%	18,69%	5,82%
	0,9	49,56%	39,38	32,28	21,25	28,78%	4,49%
	1	3,9%	2,87%	0,59%	0,38%	2,52%	0,06%

Najlepszy wynik (49,56%) otrzymałem dla bety równej 0,1 i gammy równej 0,9.

Dobranie parametrów beta i gamma dla zmodyfikowanego systemu nagród:

		beta					
		0,1	0,25	0,5	0,75	0,9	1
gamma	0,1	0%	29,98%	0%	0%	0%	0%
	0,25	57,77%	52,62%	43,72%	13,74%	0%	0%
	0,5	68,74%	52,92%	43,43%	33,14%	23,94%	0%
	0,75	65,76%	66,28%	54,02%	40,36%	35,57%	22,5%
	0,9	42,10%	43,43%	29,49%	35,30%	20,23%	4,03%
	1	0,25%	0,26%	0,42%	0,09%	0,60%	0,05%

Moją modyfikacją było dodanie kary (liczba przeciwna do nagrody za skarb) równej -10 za wejście na pole z dziurą. Dzięki temu algorytm powinien trzymać się z dala od pól sąsiadujących z przeręblą, gdzie istnieje ryzyko wpadnięcia.

Najlepszy wynik (68,74%) otrzymałem dla bety równej 0,1 i gammy równej 0,5.

Dobranie parametru epsilon:

Epsilon:	0,1	0,2	0,3	0,4	0,5	0,6	0,7	0,8
Bazowy system nagród	51,47%	46,95%	46,13%	45,18%	46,49	35,64%	50,00%	39,76%
Własny system nagród	68,95%	64,79%	64,85%	59,33%	60,65%	63,41%	61,24%	59,46%

Dla bazowego systemu nagród najlepsze wyniki otrzymałem dla:

beta = 0,1 ; gamma = 0,9 ; epsilon = 0,1

Wyniki:

Średnia: 53,35%

Max: 82,69%

Min: 25,51%

Odchylenie: 16,11%

Dla własnego systemu nagród najlepsze wyniki otrzymałem dla:

beta = 0,1 ; gamma = 0,5 ; epsilon = 0,1

Wyniki:

Średnia: 68.36%

Max:: 89.89%

Min: 54.72%%

Odchylenie: 10.31%

Wnioski:

Dobór bety i gammy jest ważnym elementem algorytmu. Beta jest współczynnikiem uczenia, należy tak dobrać jego wartość, aby znaleźć balans pomiędzy obliczeniami z poprzednich i nadchodzących iteracji. Gamma natomiast im wyższa tym bardziej skupia się na przyszłej nagrodzie. W tym przypadku nagrodę otrzymujemy tylko za dotarcie do skarbu, więc będzie oczekiwana wysoka wartość gammy.

System nagród ma kluczowy wpływ na działanie programu. Pozwala on na nakierowanie algorytmu na odpowiednie zachowanie, swego rodzaju plan. W tym przypadku zwykłe najszybsze znalezienie drogi nie było wystarczające. Potrzebne jest zniwelowanie wpływu poślizgu podczas poruszania się po planszy, odciągnięcie algorytmu od poruszania się blisko dziur.