

Raport - pracownia nr 5

 Autor:

Szymon Koper (247985)

sz.koper@gmail.com

 Repozytorium:

Kod źródłowy znajduje się [tutaj](#):

<https://github.com/sakydpozrux/Al-course/tree/master/5-markov-decision-processes/mdp-ruby>

Część 1 - Iteracja wartości

Napisz program rozwiązujący dyskretne problemy MDP opisanego typu metodą iteracji wartości lub iteracji polityki.

W tym celu opracuj opis formalny zagadnienia, tak aby specyfikację konkretnej instancji zagadnienia można było oddzielić od programu i umieścić w oddzielnym pliku. Dotyczy to geometrii świata ($N \times M$), modelu niepewności ruchów (a, b), funkcji nagrody (r i indywidualne wartości), i współczynnika dyskontowania (d).

Uwaga: załączony rysunek świata agenta jest jedynie przykładowy. Program powinien być w stanie rozwiązać zadanie dla dowolnego świata, zgodnego z powyższą specyfikacją.

Przez rejestrowanie przez program wyników częściowych, zaimplementuj analizę zbieżności algorytmu tworząc wykres wartości użyteczności stanów programem Gnuplot. Załącz zarówno wynikowy wykres, jak i opracowany komplet poleceń Gnuplota, oraz pliki danych, pozwalające wygenerować ten wykres dla każdej instancji problemu.

1.

Oblicz rozwiązanie zagadnienia 4x3 przedstawionego na wykładzie. Porównaj wyniki. W raporcie podaj komplet otrzymanych użyteczności stanów, politykę, oraz wykres zbieżności. A następnie porzuć ten przykład (był tylko na rozgrzewkę).

Dane wejściowe:

dana	wartość
a	0.8
b	0.1
reward	0.04
discount	1.0
epsilon	0.05

Wejście - pola i nagrody:

```
0 0 0 G
0 F 0 G
S 0 0 0

-0.04 -0.04 -0.04 1.0
-0.04 -0.04 -0.04 -1.0
-0.04 -0.04 -0.04 -0.04
```

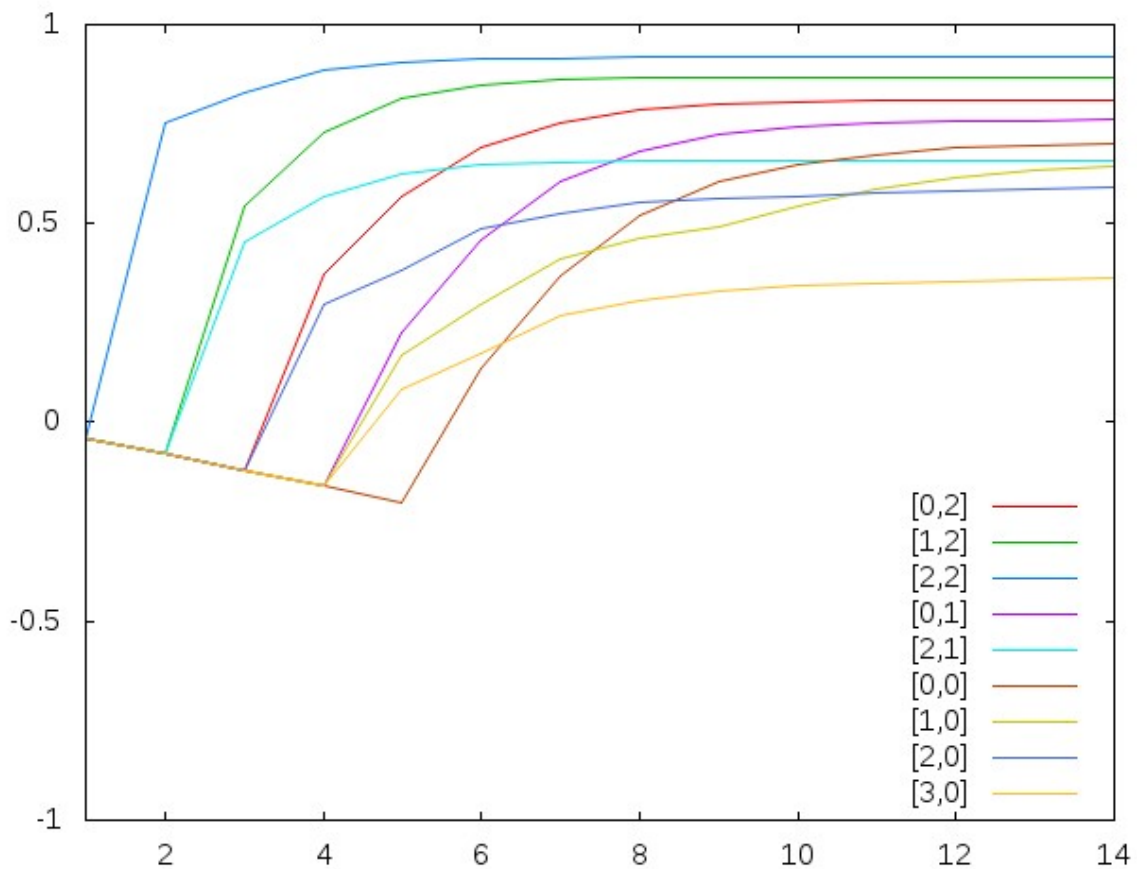
Obliczone użyteczności:

```
0.81 0.87 0.92 1.00
0.76 0.00 0.66 -1.00
0.70 0.64 0.59 0.36
```

Znaleziona polityka:

```
>>>G
^F^G
^<<<
```

Wykres użyteczności:



Wnioski:

Algorytm iteracji wartości zbiega po 14 iteracjach. Wyliczona polityka wygląda bardzo sensownie i jest podobna do polityki przedstawionej na wykładzie.

2.

Znajdź rozwiązanie zagadnienia pokazanego na obrazku ($N=4$, $M=4$), przyjmując model niepewności ruchów dokładnie taki sam jak w przykładach omawianych na wykładzie ($a=0.8$, $b=0.1$), i współczynnika dyskontowania $d=0.99$. Funkcja nagrody wynosi +100 dla stanów terminalnych, -20 dla stanów specjalnych, i $r=-1$ dla zwykłych stanów. Jako kryterium stopu algorytmu możesz przyjąć zejście wszystkich różnic funkcji wartości między kolejnymi iteracjami poniżej 0.01. Podaj komplet wartości użyteczności, politykę optymalną, i wykres zbieżności.

Dane wejściowe:

dana	wartość
a	0.8
b	0.1
reward	1.0
discount	0.99
epsilon	0.05

Wejście - pola i nagrody:

```
0 0 0 0
0 0 0 0
0 0 B 0
S 0 F G

-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -20.0 -1.0
-1.0 -1.0 -1.0 100.0
```

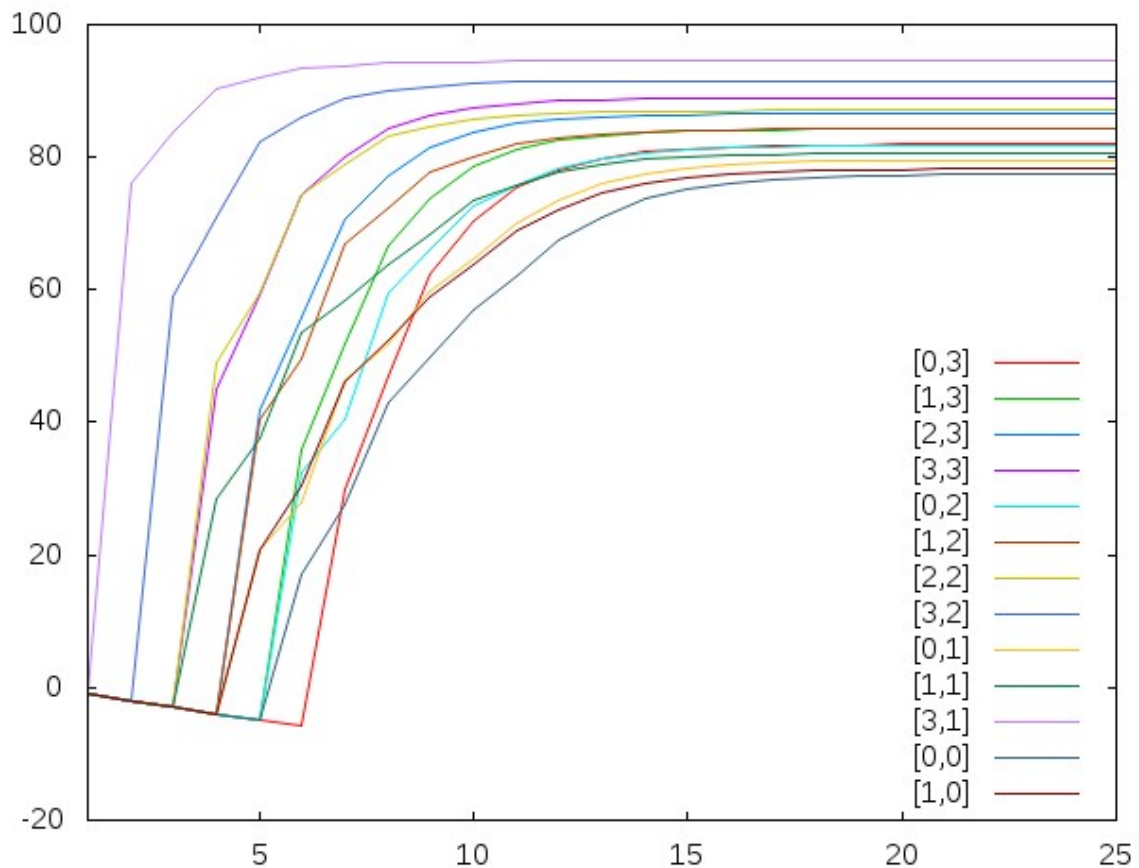
Obliczone użyteczności:

```
81.94 84.26 86.59 88.88
81.73 84.27 87.06 91.55
79.59 80.60 70.47 94.54
77.45 78.25 0.00 100.00
```

Znaleziona polityka:

```
>>>V
>>>V
^^>V
^^FG
```

Wykres użyteczności:



Wnioski:

Algorytm iteracji wartości zbiega po 25 iteracjach. Tutaj również rozwiązanie znalezione przez algorytm iteracji wartości wygląda sensownie.

3.

Zmień funkcję nagrody dla stanów normalnych, specjalnych, i/lub terminalnych (względem podstawowej wersji zadania) w taki sposób, aby skutkowało to istotną zmianą polityki optymalnej. Przedstaw komplet rozwiązań dla zmienionego zadania (użyteczności i politykę). Wyjaśnij uzyskane wyniki.

Dane wejściowe:

dana	wartość
a	0.8

b	0.1
reward	1.0
discount	0.9
epsilon	0.05

Wejście - pola i nagrody:

```

0 0 0 0
0 0 0 0
0 0 B 0
S 0 F G

-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 200.0 -1.0
-1.0 -1.0 -1.0 100.0

```

Obliczone użyteczności:

```

887.12 997.90 1111.33 1002.59
997.80 1137.57 1294.85 1129.65
1110.17 1294.10 1516.40 1201.47
995.86 1121.29 0.00 100.00

```

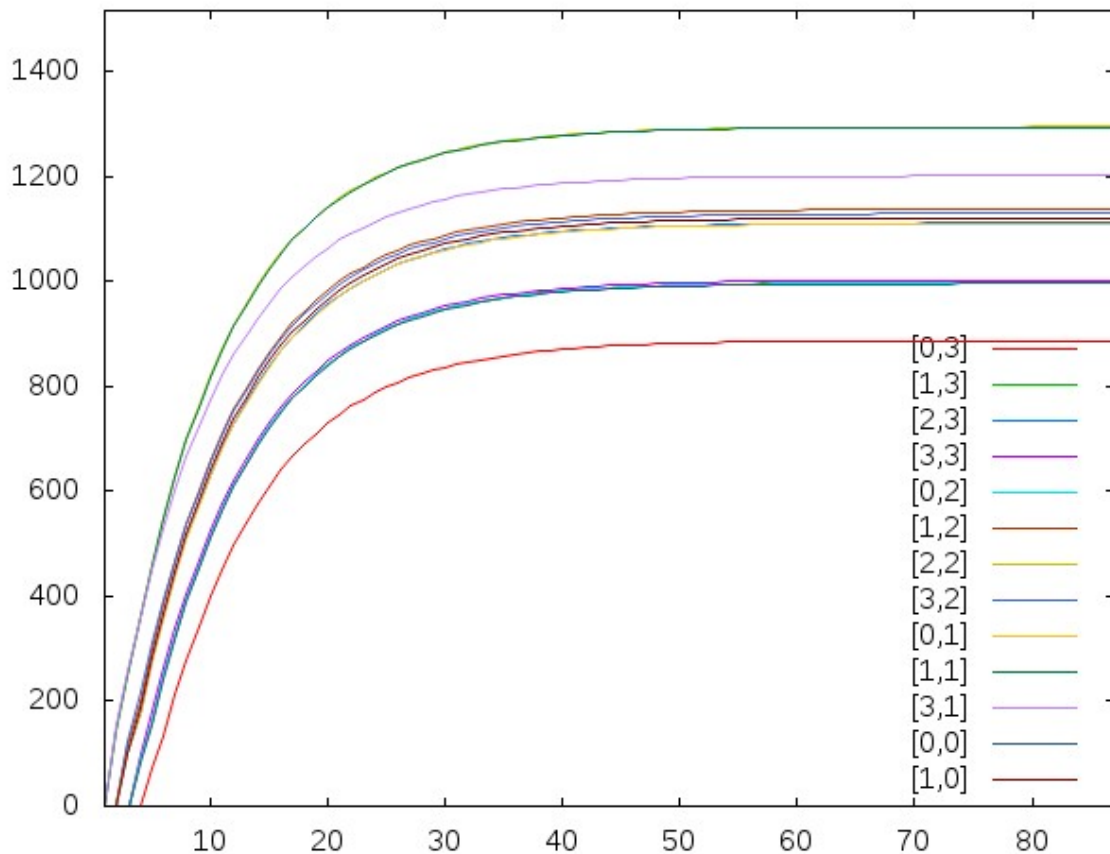
Znaleziona polityka:

```

>vvv
>>v<
>>v<
>^FG

```

Wykres użyteczności:



Wnioski:

Pole specjalne ma nagrodę dużo wyższą (wartość 200) niż stan terminalny (wartość 100), przez co algorytm dąży do jak najdłuższego pozostania na polu specjalnym i kumulację nagrody, poprzez próby wejścia na sąsiednią ścianę. To jest też powodem tak dużej liczby iteracji, w liczbie 87.

4.

Zmień model niepewności ruchów (względem podstawowej wersji zadania) w taki sposób, aby skutkowało to zmianą polityki optymalnej. Przedstaw komplet rozwiązań dla zmienionego zadania (użyteczności i politykę). Wyjaśnij uzyskane wyniki.

Dane wejściowe:

dana	wartość
------	---------

a	0.5
b	0.25
reward	1.0
discount	0.99
epsilon	0.05

Wejście - pola i nagrody:

```

0 0 0 0
0 0 0 0
0 0 B 0
S 0 F G

-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -20.0 -1.0
-1.0 -1.0 -1.0 100.0

```

Obliczone użyteczności:

```

58.10  62.02  66.22  70.04
56.67  60.18  65.33  75.39
53.87  54.52  50.14  83.98
50.96  51.29   0.00 100.00

```

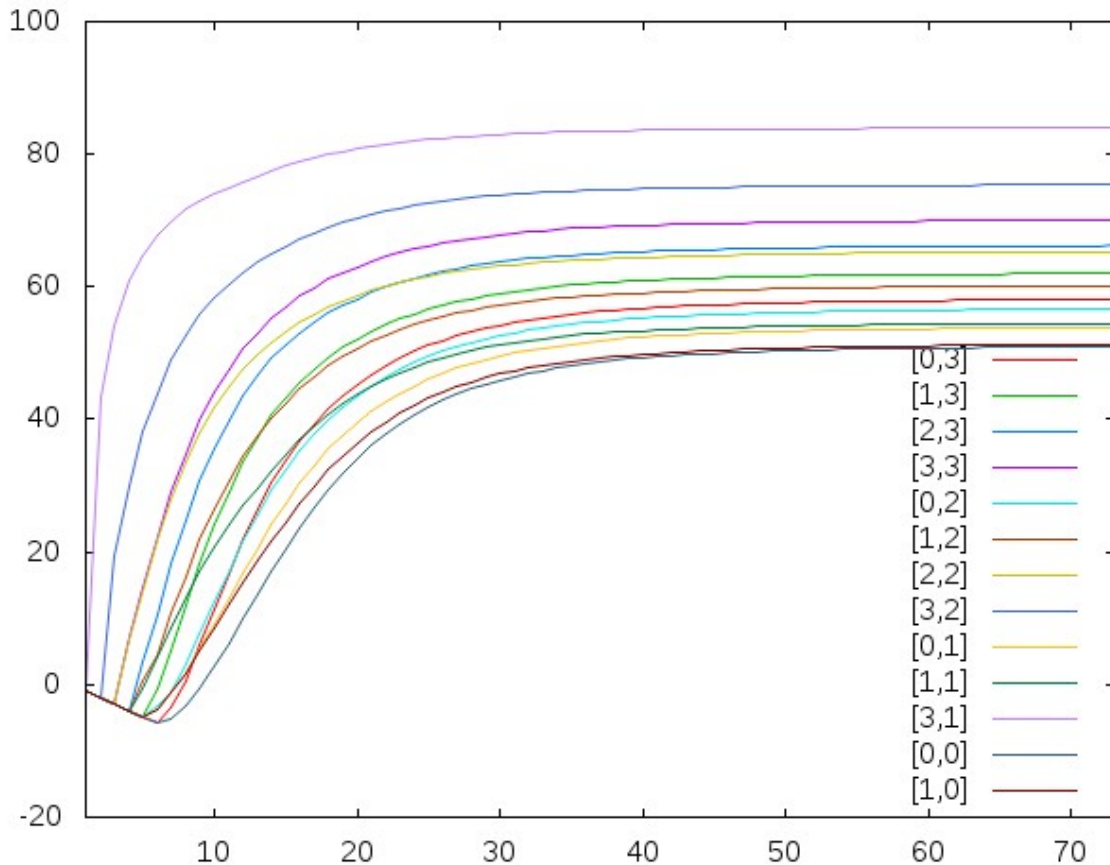
Znaleziona polityka:

```

>>>V
^>^V
^^>>
^^FG

```

Wykres użyteczności:



Wnioski:

Zmiany te skutkują wyraźnie zauważalną zmianą polityki znalezionej przez algorytm iteracji wartości. Algorytm stara się uniknąć przechodzenia przez pole specjalne o wysokiej karze. Ciekawe jest to, że algorytm stara się przechodzić w bezpiecznej odległości od pola specjalnego, czyli nie idzie najkrótszą ścieżką. W polu (4,2) żeby nie ryzykować wejścia na pole specjalne, polityka sugeruje próbę wejścia na ścianę po prawej stronie, co skutkuje szansą 50% na pozostanie w tym samym polu, i po 25% na przejście na pole wyżej lub przejście do stanu terminalnego. Algorytm zbiega po 73 iteracjach.

5.

Zmień współczynnik przeceniania (względem podstawowej wersji zadania) w taki sposób, aby skutkowało to zmianą polityki optymalnej. Przedstaw komplet rozwiązań dla zmienionego zadania (użyteczności i politykę). Wyjaśnij uzyskane wyniki.

Dane wejściowe:

dana	wartość
a	0.8
b	0.1
reward	1.0
discount	0.6
epsilon	0.05

Wejście - pola i nagrody:

```
0 0 0 0
0 0 0 0
0 0 B 0
S 0 F G

-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -20.0 -1.0
-1.0 -1.0 -1.0 100.0
```

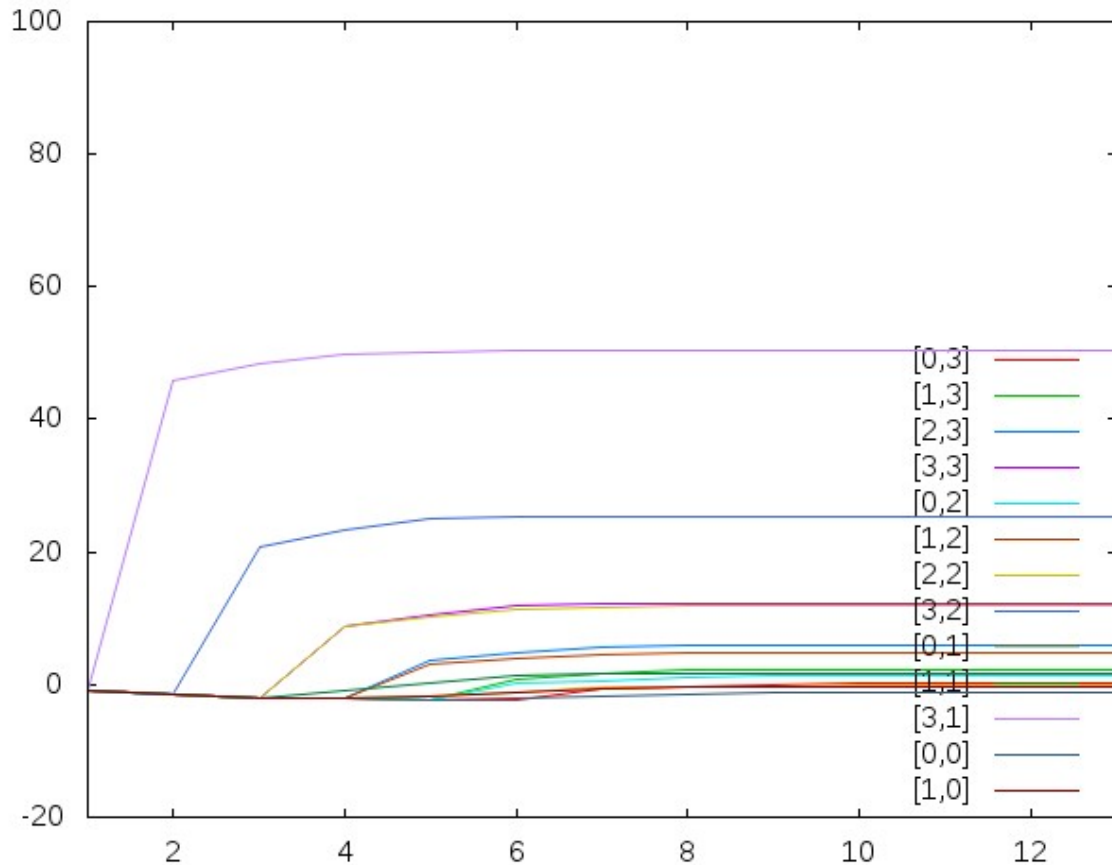
Obliczone użyteczności:

```
0.20    2.30    5.97    12.28
1.37    4.94   11.86   25.39
-0.14    1.77    5.18   50.33
-1.15   -0.23    0.00  100.00
```

Znaleziona polityka:

```
>>>V
>>>V
>>>V
^^FG
```

Wykres użyteczności:



Wnioski:

W stosunku do danych z zadania 3 zmieniłem tutaj współczynnik dyskontowania na 0,6. Jest to wartość na tyle mała, że algorytm nie zdążył wyliczyć poprawnych wartości (zbiegł zbyt szybko, w 13 iteracjach) i sugeruje przejście przez niekorzystny stan specjalny. Wygląda na to, że algorytm przy tych danych sugeruje użycie najkrótszej ścieżki, ale niekoniecznie najtańszej.

Część 2 - Q-Learning

W drugiej części zadania, zaimplementuj obliczanie optymalnej polityki agenta metodą Q-learning z eksploracją. Należy przyjąć te same parametry co w pierwszej części zadania. Jednak model ruchów i wartości nagrody są tu nieznane agentowi. Muszą one być użyte dla wygenerowania przebiegów uczących, zatem program musi składać się z dwóch części: agenta, który uczy się środowiska z przebiegów uczących, i symulatora, który te przebiegi uczące generuje. Za każdym razem wygeneruj 10000 przebiegów uczących

zaczynając ze stanu startowego do osiągnięcia stanu terminalnego. Rozwiąż instancję problemu dla świata na załączonym rysunku z ustawieniami: $a=0.9$, $b=0.05$, $d=0.99$ i współczynnikiem uczenia $\alpha = 1/N(s,a)$ gdzie $N(s,a)$ jest liczbą razy jaką akcja a została wybrana w stanie s w dotychczasowych przebiegach.

Uwzględnij dwie strategie eksploracji, z $\epsilon = 0.05$ i $\epsilon = 0.2$. To znaczy, optymalny ruch (eksploatacja) powinien zostać wybrany z prawdopodobieństwem ϵ , a z prawdopodobieństwem $1 - \epsilon$ ruch wybrany losowo (eksploracja).

Przeanalizuj wpływ wartości ϵ na zbieżność procesu Q-learning, oraz na uzyskaną w jego wyniku politykę.

Oba badane przypadki mają identyczną planszę i nagrody poszczególnych pól:

```
0 0 0 0
0 0 0 0
0 0 B 0
S 0 F G

-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -1.0 -1.0
-1.0 -1.0 -20.0 -1.0
-1.0 -1.0 -1.0 100.0
```

1.

Dane wejściowe:

dana	wartość
a	0.9
b	0.05
reward	1.0
discount	0.99
epsilon	0.05

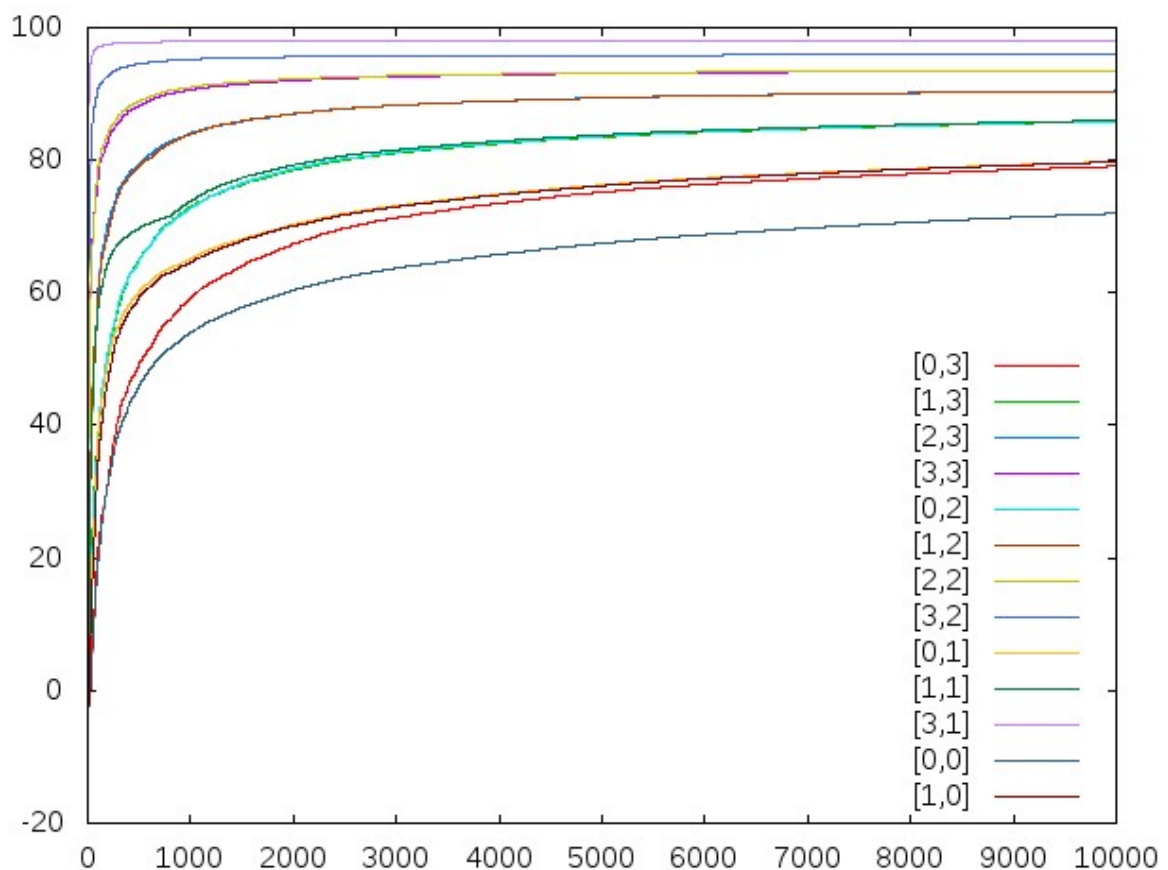
Obliczone użyteczności:

79.10	85.84	90.45	93.51
85.85	90.44	93.56	95.92
79.75	86.00	76.92	97.99
72.00	79.73	0.00	100.00

Znaleziona polityka:

$v > vv$
 $>>> v$
 $>^> v$
 $\wedge FG$

Wykres użyteczności:



 2.

Dane wejściowe:

dana	wartość
a	0.9
b	0.05
reward	1.0
discount	0.99
epsilon	0.2

Obliczone użyteczności:

77.25	85.46	90.40	93.59
85.09	90.45	93.59	95.93
78.47	85.94	76.85	97.99
69.30	78.24	0.00	100.00

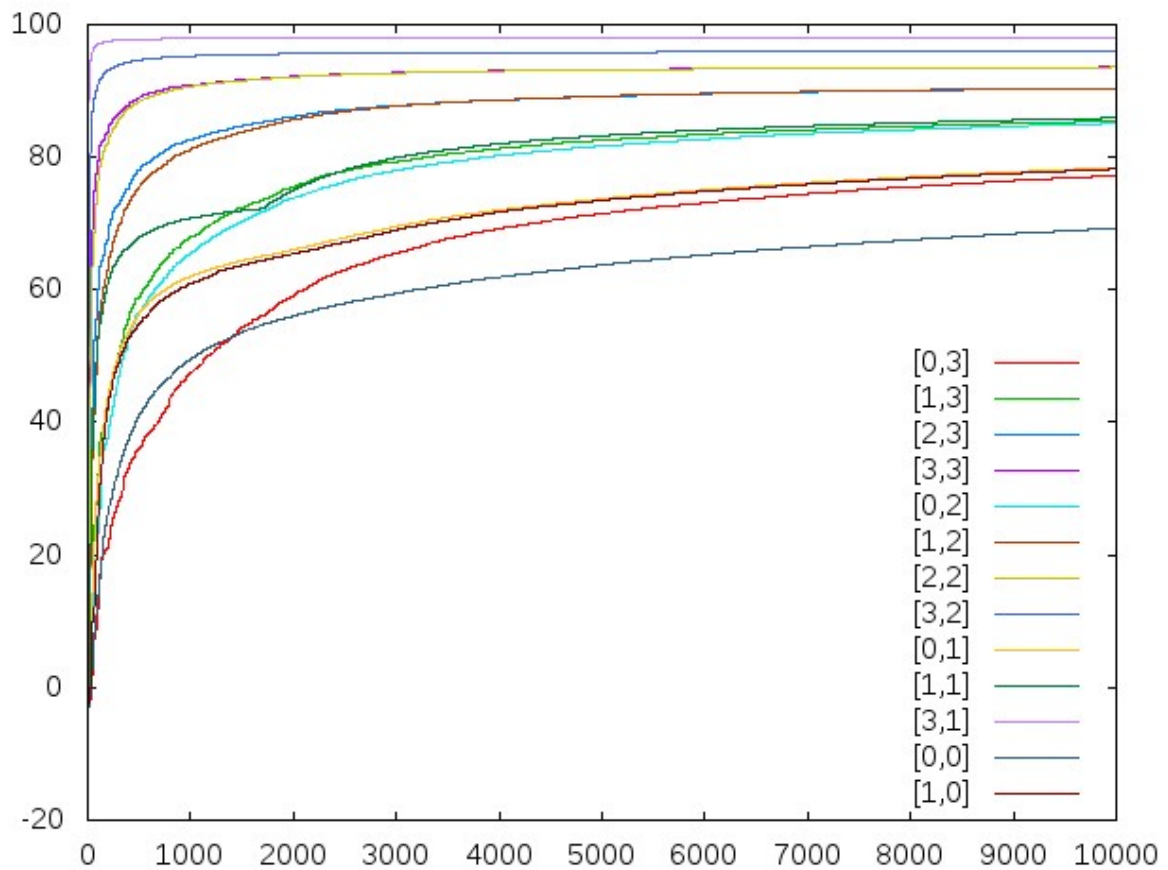
Znaleziona polityka:

```

>>>v
>>>v
>^>v
^^FG

```

Wykres użyteczności:



📁 Wnioski z części 2:

Wartość epsilon ustawiona na 0.05, skutkuje algorytmem odważniej eksplorującym planszę, ale mniej chętnie korzystającym z już posiadanej wiedzy, czyli podążania ścieżką uznawaną za aktualnie optymalną. Widoczna jest mała różnica w polach (0,3) oraz (2,3).