Systemy Agentowe

Projekt 2018 L

Sprawozdanie

Paweł Walczak
Kamil Gabryjelski
Igor Markiewicz

Spis treści

1 Opis zadania

W trakcie realizacji projektu ustalono ostateczny zakres i cel projektu. Stworzony został system niezależnego wyuczania agnetów ścieżki do celu (nagrody) na planszy z przeszkodami algorytmemt uczenia ze wzmocnieniem (krytykiem). Tak uszczegółowiony temat projektu pozwala na założenie o statyczności środowiska i wykorzystanie klasycznego algorytmu typu Q – *learning*.

2 Realizacja zadania

2.1 Założenia

- mapa ma kształt kwadratu wypełnionego polami o liczności n
- istnieje $k < n$ agentów, z których każdy wyuczany jest niezależnie
- na mapie istnieje $l_1 < n$ przeszkód oraz $l_2 < n - l_1$ pól z nagrodami (statycznych stanów absorbujących)
- każdy agent wyucza się niezależnie strategii maksymalizującej wartości funkcji Q wyznaczając swoje ścieżki do nagrody (nagród) i starajac się unikać pól z przeszkodami :

$$Q(s_t, a_t) \longleftarrow (1-\alpha) \cdot Q(s_t, a_t) + \alpha \cdot \left(r_t + \gamma \cdot \max_a \left[Q(s_{t+1}, a) \right] \right)$$

$$\alpha, \gamma \in [0, 1]$$

  - zbiór stanów S zawiera liczby naturalne : $1, 2, \ldots, n$, oznaczające numerowanie kolejnych komórek (wierszami) począwszy od górnego, lewego rogu mapy (Rys. 1)

0	1	$\ldots$	$n-1$
n	$n+1$	$\cdots$	$2n-1$
$\vdots$	$\ddots$	$\ddots$	$\vdots$
$n(n-1)$	$\ldots$	$\ldots$	n^2-1

Rys. 1: Schemat numerowania pól mapy.

– zbiór akcji A jest połączony po przez przypisanie (mapowanie) ze biorem S w następuujący sposób :

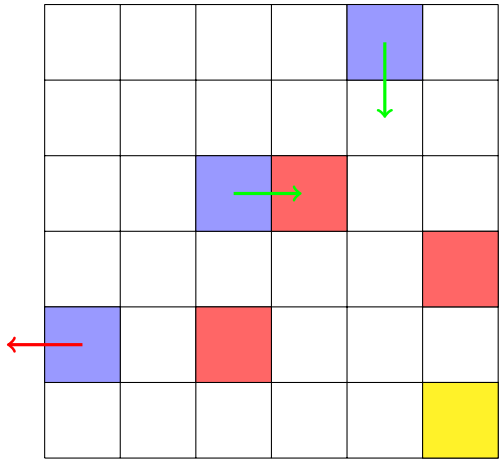
$$f(s_t) = S_{t+1}^{approved}$$

co oznacza że w stanie s_t możemy wykonać prawidłowe przejście do jednego ze stanów w $S_{t+1}^{approved}$. Jako prawidłowe przejście definiuje się przesunięcie do dowolnej komórki przylegającej do danej w schemacie miejskim (prostopadłe drogi), z wyłączeniem sytuacji pozwalających na wyjście poza mapę (Rys. 2)

– pogodzenie sporu eksploracja vs eksploatacja zostało zaproponowane jako wykorzystanie alogrytmu ϵ – zachłannego przy wyborze kierunku ruchu :

1. $\epsilon = const + \frac{\epsilon_{dropoff}}{\text{numer\_iteracji} + \epsilon_{dropoff}}$
2. jeśli wartość rnd z generatora losowego o rozkładzie jednostajnym na przedziale $[0, 1]$ jest mniejsz niż ϵ to :
   (a) wybierz losową akcję
   (b) w przeciwnym razie wybierz losowo akcję ze zbioru akcji o największej wartości funkcji Q

- nad wszystkimi agentami czuwa agent środowiskowy nadający nagrody (kary) za ruch (-1 za wejście na pole z przeszkodą, 2 za wejście na pole z nagrodą, 0 w pozostałych przypadkach)



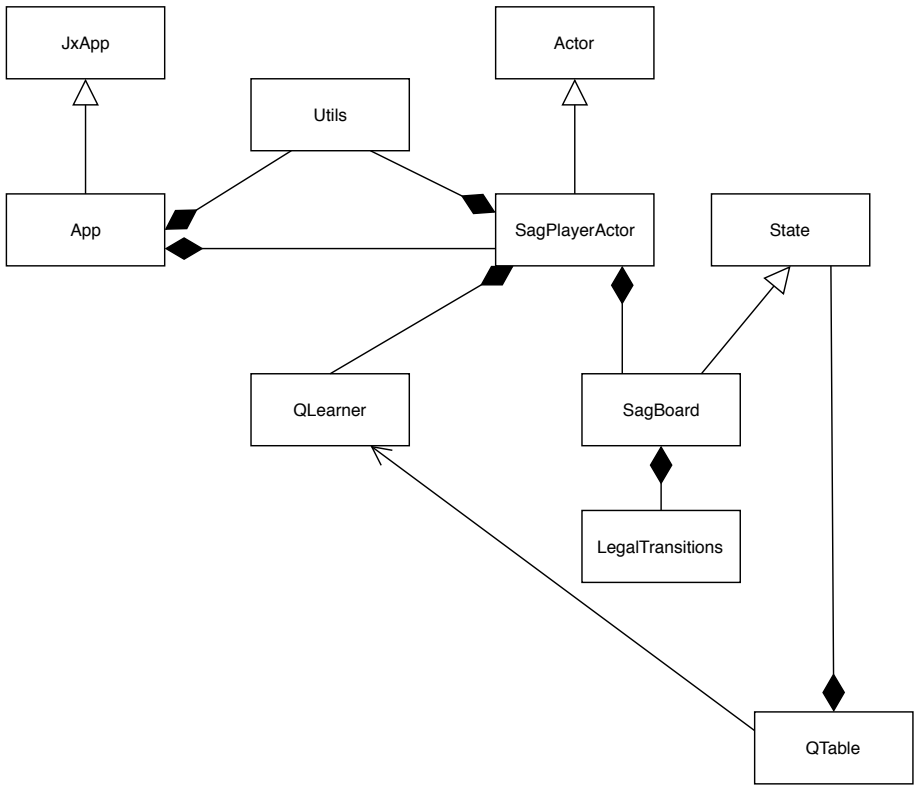
Rys. 2: Na żółto – pole z nagrodą, na niebiesko – agenci, na czerwono – statyczne przeszkody, zielone strzałki – dozwolone ruch, czerwona strzałka – niedozwolone wyjście poza mapę.

2.2 Wybór narzędzi

Jako język programowania została wybrana *Scala* z biblioteką *Akka* służącą do modelowania systemów aktorowych/agentowych. Algorytm Q – learning został zaimplementowany ręcznie. Do wizualizacji poruszających się agentów została użyta biblioteka graficzna *ScalaFX*, a jako system zarządzania projektem został wybrany framework *Maven*.

2.3 Architektura

2.3.1 Diagram UML



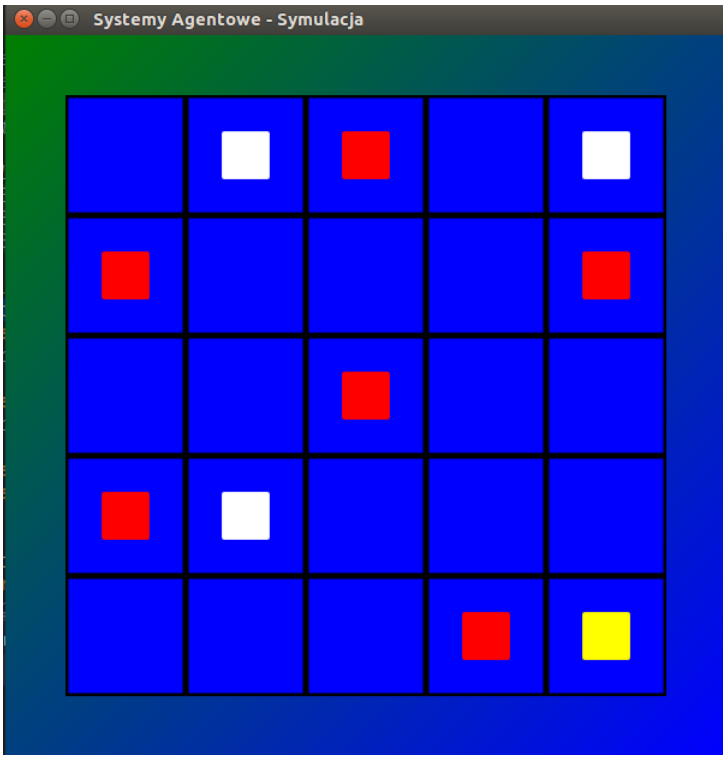
Rys. 3: Diagram UML

2.3.2 Opis plików

- *src.main.scala.qlearning.common*
  - *Evaluator* – trait deklarujący funkcję obliczającą wartości Q
  - *QLearner* – klasa definiująca obiekty służące do uczenia agentów
  - *QTable* – klasa definiująca tabelę wartości Q z metodami do wywoływania na niej
  - *State* – trait deklarujący funkcje dostępne dla stanu
- *src.main.scala.qlearning.actorSystem*
  - *App* – obiekt implementujący trait *JFXApp*, uruchomianie programu oraz reprezentacja graficzna

  - *LegalTransitions* – obiekt zawierający funkcję do generowania stanów i dozwolonych przejść
  - *SagBoard* – klasa implementująca trait *State* i definiująca wymaganie funkcje służące do obsługi agentów i zleceń
  - *SagPlayer* – klasa implementująca trait *Actor* zawierająca definicje funkcji służące do nauki i wykonywania ruchów
  - *Utils* – obiekt zawierający parametry startowe oraz definicje klas będących podstawowymi typami danych
- *src.test.scala.tests* – obiekt z testami jednostkowymi

2.4 GUI



Rys. 4: Wygląd panelu symulacji

3 Badania

Założenia ogólne co do badań :

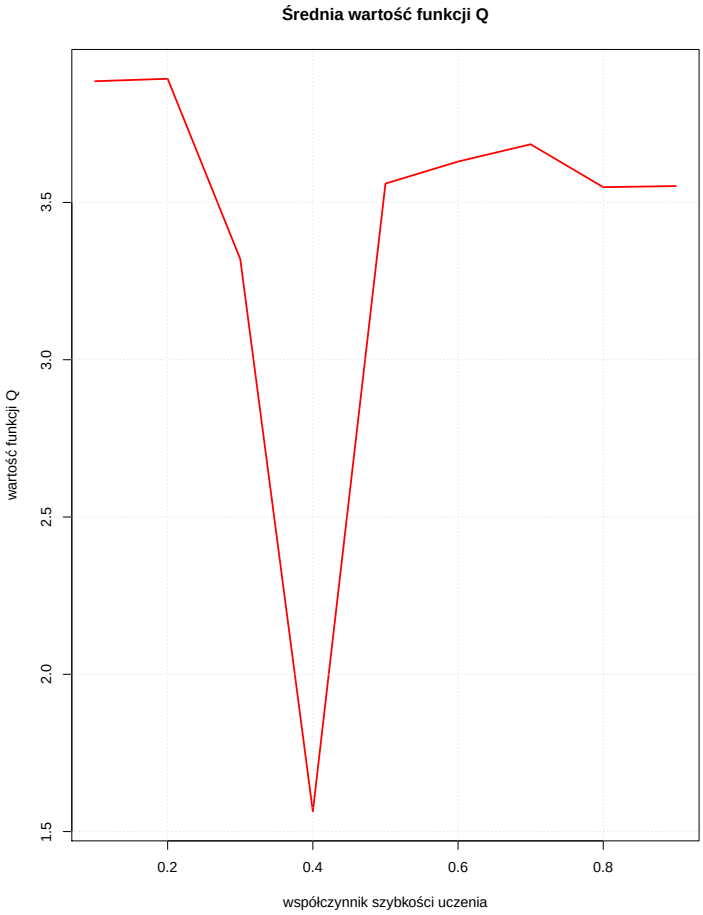
- plansza ma wymiar 5×5
- w badaniu bierze udział trzech agentów (białe pola)

- na planszy istnieje 6 przeszkód (czerwone pola) oraz 1 cel (żółte pole)
- romieszczenie poszczególnych elementów (w tym początkowe agentów) przedstawione jest na Rys. 3
- $const = 0,01$
- $\epsilon_{dropoff} = 5,0$
- liczba iteracji = 1000

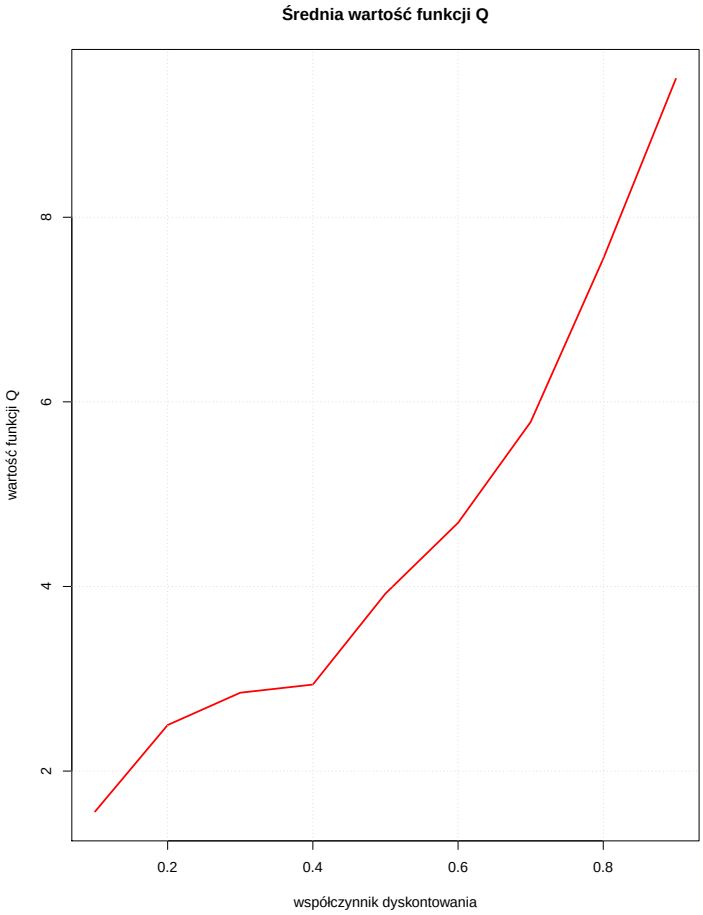
3.1 Badanie wpływu współczynnika szybkości uczenia α i dyskontowania γ

Badania w tej części zostały wykonane przy następujących założeniach :

- badaną cechą jest suma wartość funkcj Q dla wszystkich wykonanych kroków w ostatniej iteracji uczenia
- wynik są zbierane od każdego agenta osobno a następnie uśredniane
- ustalono arbitralnie że pierwszym badaniem będzie testowanie parametru α przy $\gamma = 0,5$
- następnie dla najlepszej wartości parametru α przeprowadzono testowanie parametru γ



Rys. 5: Wyniki dla współczynnika α



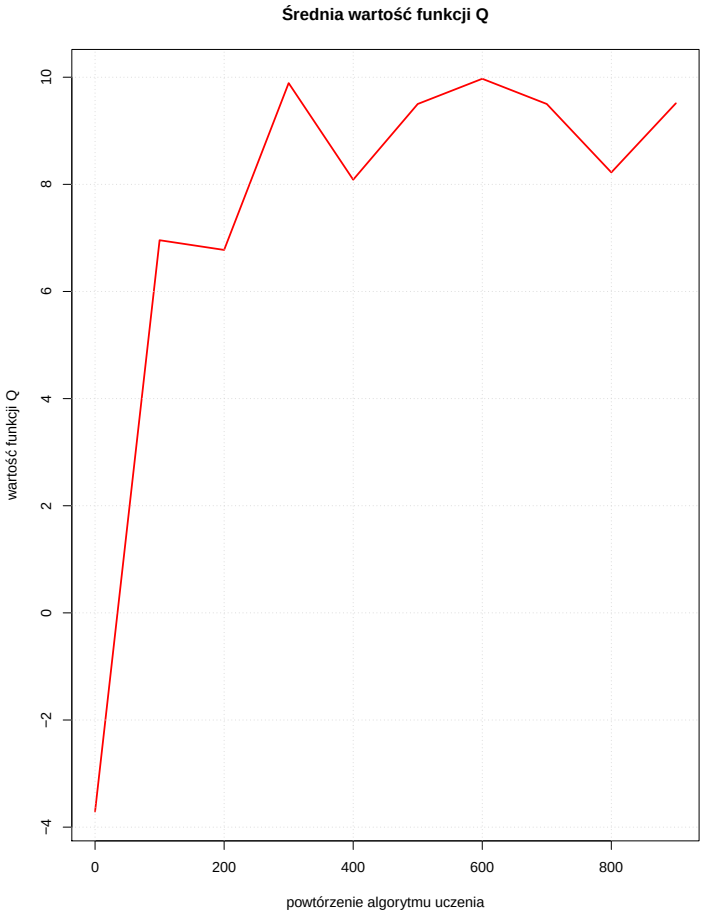
Rys. 6: Wyniki dla współczynnika γ

Możemy zauważyć że najlepsze efekty otrzymujemy dla $\alpha \approx 0,1$ oraz $\gamma \approx 0,9$.

3.2 Badanie wartości Q

W tej części dla ustalonych wartości parametrów α, γ został przeprowadzony test skuteczności :

- badaną cechą jest suma wartość funkcj Q dla wszystkich wykonanych kroków w kolejnych iteracjach uczenia
- wynik są zbierane od każdego agenta osobno a następnie uśredniane



Rys. 7: Wyniki dla $\alpha = 0,1$ oraz $\gamma = 0,9$

3.3 Wnioski

- najlepsze efekty uzyskane zostały dla $\alpha \approx 0,1$ co oznacza że preferowane jest wolniejsze uczenie, zapobiegające zapewne znacznym przeskom
- duża wartość $\gamma = 0,9$ świadczy że zdecydowanie bardziej preferowane są sytuacje w których system może polegać na wartościach wybiegających w "przyszłość"
- wartość funkcji z Rys. 6 nasyca się na poziomie $Q \approx 9$ dla ok 300 iteracji

4 Bibliografia

[1] materiały wykładowe z Systemów Agentowych 2018 L

[2] `https://akka.io/`

[3] "Systemy uczące się" – *P.Cichosz*, WNT Warszawa 2000, 2007 – wyd. drugie

[4] `https://en.wikipedia.org/wiki/Q-learning`