UMA - Dokumentacja wstępna

Kacper Marchlewicz, Przemysław Wyziński

1.Treść projektu

Zastosowanie uczenia ze wzmocnieniem do ustawienia prawdopodobieństwa krzyżowania i sposobu krzyżowania w algorytmie ewolucyjnym. Niech stanem będzie procent sukcesów (ile potomków było lepszych niż rodzice) oraz średnia odległość pomiędzy osobnikami w aktualnej populacji, a akcją wybór wskazanych parametrów algorytmu. Zarówno stany jak i akcje można zdyskretyzować. Funkcje do optymalizacji należy pobrać z benchmarku CEC2017, którego kod da się znaleźć w Pythonie, R i C. Przykład w Pythonie. Przed rozpoczęciem realizacji projektu proszę zapoznać się z zawartością strony.

2. Wykorzystywane algorytmy

Uczenie ze wzmocnieniem

Uczenie ze wzmocnieniem to algorytm, który na podstawie podstawie prób i błędów otrzymuje ocenę. Wraz z działaniem algorytmu uczy się maksymalizować nagrody, przez co coraz lepiej wykonuje powierzone mu zadanie.

Pseudokod

Data: t_{max}, γ, β, e_{max}, ε

Result: Q

begin

$Q_0 \leftarrow$ zainicjuj

$e \leftarrow 0$

while $e < e_{max}$ **do**

$t \leftarrow 0$

$x_t \leftarrow$ zainicjuj

while $t < t_{max}$ & $x_t \notin$ stany absorbujące **do**

$a_t \leftarrow$ wybierz akcję(x_t, Q_t)

r_t, $x_{t+1} \leftarrow$ wykonaj akcję a_t

$\Delta \leftarrow r_t + \gamma \, max_a \, Q_t(x_{t+1}, a) - Q_t(x_t, a_t)$

$Q_{t+1} \leftarrow Q_t + \beta\Delta$

$t \leftarrow t + 1$

end

$e \leftarrow e + 1$

end

end

Działanie algorytmu uczenia się ze wzmocnieniem

Algorytm opiera się na algorytmie Q-learning z epizodami. Algorytm ten wykorzystuje Q-funkcję, którą modyfikuje po kolejnych epizodach oraz na podstawie której wykonuje coraz lepsze akcje.

Początkowo algorytm inicjalizuje macierz Q oraz licznik epok zerami. Następnie w każdej epoce algorytm kolejno:

- Inicjuje pozycję początkową,
- Wybiera akcje,
- Wykonuje akcje,
- Aktualizuje tablicę Q.

W każdym epizodzie algorytm kończy swoje działanie po wykonaniu maksymalnej liczby akcji bądź osiągnięciu stanu absorbującego. Na sam koniec uczenia, algorytm powinien odnajdywać największą nagrodę w jak najmniejszej liczbie akcji.

Algorytm posiada również następujące parametry:

- ε - parametr odpowiadający za wybór akcji wg strategii ε-zachłannej; z prawdopodobieństwem ε wybierana jest akcja losowa, z prawdopodobieństwem 1 - ε akcję na podstawie macierzy Q.
- γ - parametr odpowiadający za wartość nagrody otrzymywanej w obecnej chwili. Zwiększając ten parametr algorytm wyżej wartościuje przyszłe nagrody i mniej zwraca uwagę na obecną nagrodę, co może polepszyć zdolność do szukania innych ekstremów.
- β - parametr odpowiadający za tempo zmian wartości w tablicy Q. Duża wartość tego parametru oznacza, że algorytm będzie mocniej modyfikował swoje informacje, co może zwiększyć tempo osiągania ekstremum.

Algorytm ewolucyjny

Jest to przybliżony algorytm optymalizacyjny w którym stosowane są mechanizmy selekcji, reprodukcji i mutacji inspirowane przez biologiczny proces ewolucji.

Pseudokod

Data: q(x), P_0, μ, σ, p_c, t_{max}

Result: $\hat{x}^*$, $\hat{o}^*$

begin

t ← 0

o ← ocena(q, P0)

$\hat{x}^*$, $\hat{o}^*$ ← znajdź najlepszego(P_0, o)

while *nie spełnione kryterium stopu(t,* t_{max}, P_t, *o)* **do**

R ← reprodukcja(P_t, o, μ)

M ← operacje genetyczne(R, σ, p_c)

o_m ← ocena(q, M)

x ∗ t , o∗ t ← znajdź najlepszego(M, o_m)

if o_t^* ≤ $\hat{o}^*$ **then**

$\hat{o}^*$ ← o_t^*

$\hat{x}^*$ ← x_t^*

end

P_{t+1}, o ← sukcesja(P_t, M, o, o_m)

t ← t + 1

end

end

Działanie algorytmu ewolucyjnego

Algorytm rozpoczyna się od inicjalizacji populacji początkowej - rodziców. Następnie będą z niej tworzone nowe, coraz to lepsze populacje przy pomocy operacji selekcji, krzyżowania i mutacji.

W pętli głównej algorytmu kolejno:

- reprodukcja - wybieramy z populacji poprzedniej iteracji najlepsze punkty, będą one rodzicami obecnej populacji
- operacje genetyczne - dokonujemy krzyżowania rodziców, czyli wygenerowania punktu pośredniego, który następnie jest mutowany - do wartości punktu dodawany jest wektor liczb losowych z rozkładu normalnego przeskalowany przez parametr siły mutacji - powstaje populacja mutantów
- dokonujemy oceny nowej populacji (mutantów) i sprawdzamy czy nie pojawił się nowy najlepszy punkt
- sukcesja - wybór, które punkty z zbioru rodziców i mutantów przejdą do kolejnej iteracji algorytmu

Obliczenia dokonywane są do momentu przekroczenia limitu liczby iteracji, którego wartość zależy od liczby populacji i budżetu obliczeniowego. Należy dokonać takiego doboru parametrów aby znaleźć kompromis pomiędzy eksploracją a eksploatacją algorytmu. Zbyt duża populacja spowalnia proces optymalizacji. W pracy nad projektem zdecydowaliśmy się zastosować reprodukcję turniejową i sukcesję elitarną.

Połączenie uczenia ze wzmocnieniem i algorytmu ewolucyjnego

W zadaniu środowiskiem uczenia ze wzmocnieniem będzie algorytm ewolucyjny, akcjami będzie zwiększanie/zmniejszanie prawdopodobieństwa krzyżowania, a także zmiany sposobu krzyżowania. Stanem będzie procent sukcesów (ile potomków było lepszych niż rodzice) oraz średnia odległość pomiędzy osobnikami w aktualnej populacji.
Jako postać nagrody planujemy przyjąć sumę procentu sukcesów i średniej odległości między osobnikami, gdzie obie te wartości będą przeskalowane przez odpowiednie współczynniki.

3. Plan eksperymentów

W eksperymentach sprawdzimy wpływ sposobu obliczania nagrody, parametrów uczenia ze wzmocnieniem (β, γ, ϵ) na dokładność wyznaczania minimum funkcji, a także parametrów algorytmu ewolucyjnego - rozmiaru populacji, siły mutacji i rozmiaru elity.
Program zostanie uruchomiony 25 razy i zostanie wyliczona średnia wartość końcowa algorytmu, odchylenie standardowe, najmniejsza oraz największa wartość.

Wszystkie prace nad projektem będą wykonane w języku Python z wykorzystaniem bibliotek takich jak numpy, matplotlib, benchmark cec2017.