WSI - ćwiczenie 3

Agnieszka Wójtowicz

8 kwietnia 2022

Spis treści

1 Polecenie oraz elementy języka

1.1 Polecenie Prowadzącego

Program buduje drzewo gry w Czwórki. Wejściem są wymiary planszy NxM (N¿= 5,M ¿=4) oraz maksymalna głębokość drzewa. Zasady: gracze co turę wrzucają 1 kolorowy token (kolor odpowiada kolorowi danego gracza) do jednej z kolumn na planszy, jeżeli w pionie, poziomie lub na ukos znajdą się 4 tokeny gracza to ten gracz wygryw, jeżeli plansza się zapełni to dochodzi do remisu

1.2 Elementy języka

Wykorzystano język programowania Python w wersji 3.8. Skorzystano z pomocniczych bibliotek: *numpy 1.20.1*, *pygame 2.1.2*, *time*, *random*, *pandas 1.3.4*, *anytree 2.8.0*.

2 Cel zadania

Celem zadania jest implementacja algorytmu minimax oraz wykorzystanie go do gry w czwórki.

3 Funkcja heurystyczna

Przyjęto następujące założenia dotyczące funkcji heurystycznej oceny stanu gry. W przypadku pozycji wygranej, ocena wynosi odpowiednio *-inf* dla gracza minimalizującego, i *+inf* dla gracza maksymalizującego.

Gdy na planszy znajdują się jakiekolwiek trzy będące obok siebie takie same tokeny, ich liczba jest zliczona, a ocena wynosi: *liczba* * 1000 (z odpowiednim dla gracza znakiem + i -).

Gdy na planszy znajdują się jakiekolwiek dwa będące obok siebie takie same tokeny, ich liczba jest zliczona, a ocena wynosi: *liczba* * 1000 dzielona przez *rozmiar_planszy* (z odpowiednim dla gracza znakiem + i -).

W każdym innym przypadku, łącznie z remisem w stanie terminalnym, ocena wynosi 0.

4 Wyniki

4.1 Rozmiar przestrzeni poszukiwań

Przy rosnącym rozmiarze planszy, możliwa (osiągalna w sensownym czasie) głębokość będzie malała, gdyż na każdym poziomie, algorytm będzie musiał rozważyć więcej możliwości ruchu.

Zdecydowano się ustalić stały rozmiar planszy (6,5), gdyż przedmiotem zadania nie było badanie jego zmienności na wynik działania algorytmu.

4.2 Czas wykonania kroku

Dla ustalonej wielkości planszy, zmierzono czas znalezienia przez algorytm (dla początkowo pustej planszy) jednego ruchu, dla różnych głębokości. Wyniki znajdują się w Tabeli 1.

depth	time
1.0	0.01
2.0	0.03
3.0	0.18
4.0	1.06
5.0	6.23
6.0	37.34

Tabela 1: Czas wykonania kroku algorytmu o danej głębokości

4.3 Gra różnych przeciwników

4.3.1 Gracze losowi

Dla 20 pełnych gier, sprawdzono, czy przy losowych ruchach wybieranych przez graczy, gra faworyzuje, gracza początkowego. Wyniki zawiera Tabela 2,

początkowy	wygrał min	wygał max	remis
max	10	8	2
min	7	11	2

Tabela 2: Wyniki gry między graczem losowym a losowym

4.3.2 Gracz losowy oraz inteligentny

Dla różnych głębokości drzewa minimax, wykonując 20 powtórzeń dla każdej głębokości zbadano wyniki gry między graczem losowym a graczem działającym według algorytmu minimax, dla różnych graczów początkowych. Wyniki zawiera Tabela 3.

min_depth	max_depth	początkowy	wygrał min	wygał max	remis
1	playing rand	max	5	13	2
2	playing rand	max	8	12	0
3	playing rand	max	14	3	3
4	playing rand	max	19	1	0
5	playing rand	max	16	2	2
1	playing rand	min	7	12	1
2	playing rand	min	9	10	1
3	playing rand	min	17	1	2
4	playing rand	min	17	3	0
5	playing rand	min	20	0	0

Tabela 3: Wyniki gry między graczem losowym a inteligentnym

4.3.3 Dwóch graczy inteligentnych

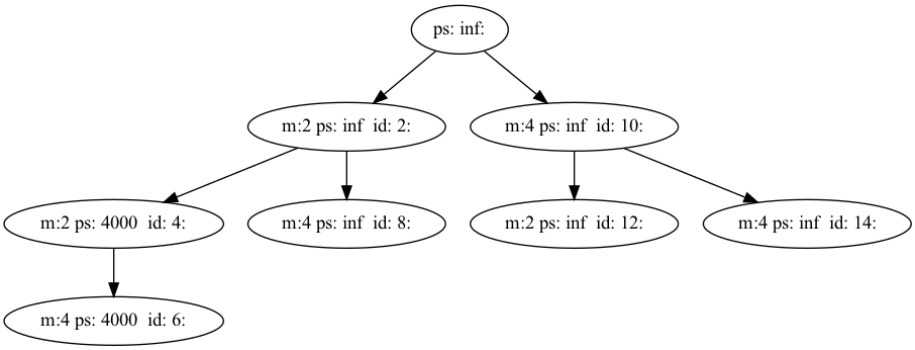
Dla jednego gracza inteligentnego grającego według algorytmu o maksymalnej głębokości drzewa, oraz drugiego gracza przyjmującego kolejne wartości głębokości drzewa, sprawdzono wyniki gry. Zbadano wpływ zmiany gracza początkowego na wyniki. Zawiera je Tabela 4.

min_depth	max_depth	początkowy	wygrał min	wygał max	remis
1	5	max	3	14	3
2	5	max	2	16	2
3	5	max	6	6	8
4	5	max	2	13	5
5	5	max	6	4	10
1	5	min	3	14	3
2	5	min	6	14	0
3	5	min	5	7	8
4	5	min	9	4	7
5	5	min	5	6	9

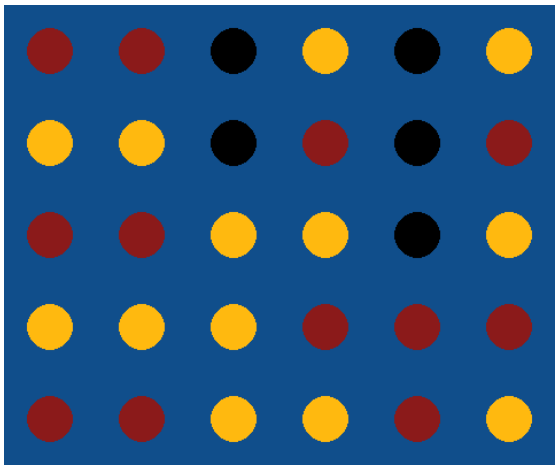
Tabela 4: Wyniki gry między graczem inteligentnym a inteligentnym

4.4 Przebieg gry

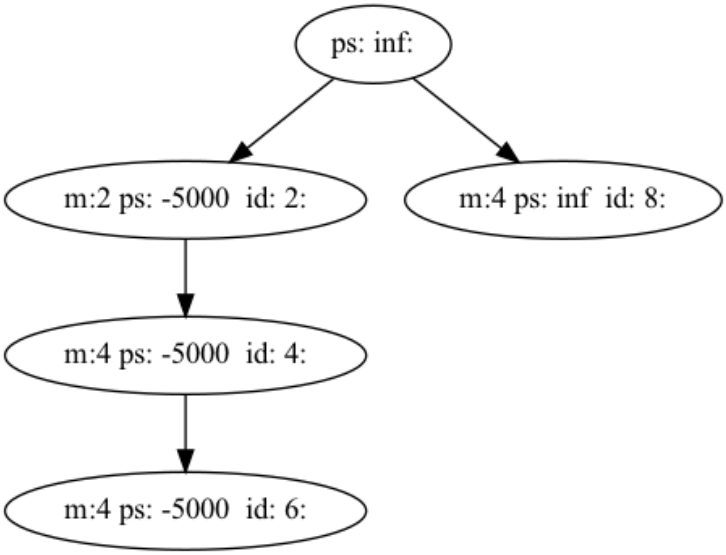
Końcowy etap Ze względu na duży rozmiar drzew gry na wcześniejszych etapach, zdecydowano się zamieścić jednie drzewo gry dla dwóch ostatnich ruchów (wraz z odpowiadającą planszą), dla przykładowej rozgrywki gracza inteligentnego z graczem inteligentnym. Gra rozgrywała się na planszy (5,6), grali ze sobą gracze inteligentni o głębokości drzewa równej 4. Na drzewie gry, oznaszenie "ps" oznacza spropagowaną według algorytmu ocenę danego stanu, "m" oznacza ruch a "id" jest nieistotnym w kontekście zadania numerem węzła wymaganym przez biblioteke.



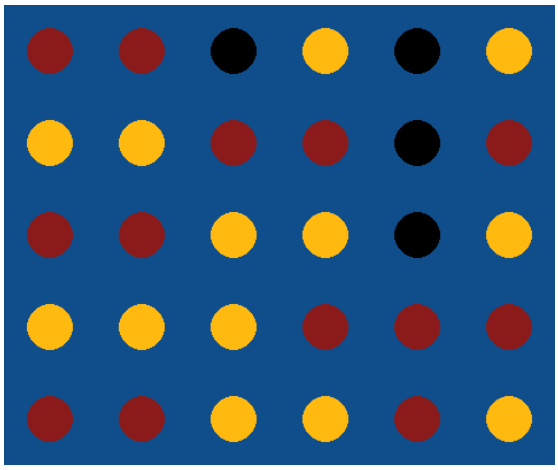
Rysunek 1: Drzewo gry, ruch 26.



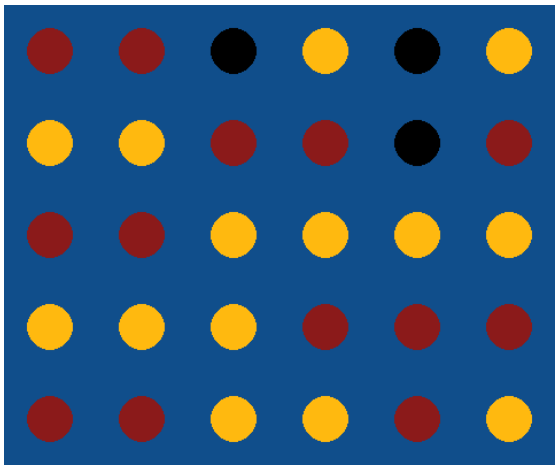
Rysunek 2: Stan gry, ruch 26.



Rysunek 3: Drzewo gry, ruch 27.



Rysunek 4: Stan gry, ruch 28.



Rysunek 5: Stan końcowy gry.

4.5 Pełen przebieg

Dla graczy inteligentnego oraz losowego, głębokości drzewa równej 3, zbadano przebieg gry. Kolejne plansze oraz drzewa gier wyświetlane są poniżej. Zdecydowano się na zmniejszenie rozmiaru planszy do (3,3) oraz wymaganej długości ciągu tych samych tokenów do 3, by ograniczyć rozmiar drzew gier i uprościć poniższą ilustracje. Przebieg gry zawiera dostarczony wraz z dokumentacją folder o nazwie "4.5. Pełny przebieg". Numery w nazwie kolejnych rysunków odpowiadają kolejnym ruchom. Nie zamieszczono ich w raporcie, gdyż wtedy drzewa gry stałyby się nieczytelne.

5 Wnioski

5.1 Czas wykonania kroku

Wraz ze wzrostem głębokości drzewa rośnie przestrzeń poszukiwań, a więc czas wykonania kroku wykładniczo wzrasta. Zdecydowano, że maksymalnym dopuszczalnym czasem wykonania kroku algorytmu jest 10 sekund, a więc przyjęto maksymalną głębokość drzewa minimax jako 5.

5.2 Gra różnych przeciwników

5.2.1 Gracze losowi

Stosunek wygranych do przegranych, niezależnie od gracza początkowego jest zbliżony.

5.2.2 Gracze losowy oraz inteligentny

Dla małej głębokości, stosunek wygarnych i przegranych dla gracza losowego oraz intelignentego jest zbliżony (niezależnie od gracza początkowego). Dla większych wartości głębokości, stosunek wygaranych do przegranych gracza inteligentnego rośnie, osiągając wartość 19 i 20 dla głębokości 5. To, że graczowi losowemu zdarza się wygrać wynika prawdopodobnie z tego, że funkcja heurystyczna opisująca stan gry nie jest dostatecznie dobra.

5.2.3 Dwóch graczy inteligentnych

Dla dużej różnicy głębokości miedzy graczami, zdecydowanie częściej wygrywa gracz o dużej głębokości drzewa gry. Dla zbliżonej głębokości, często zdarzają się remisy. Nie jest to dziwne, gdyż każdy gracz próbuje grać optymalnie, a w przypadku wyboru miedzy przegraną a remisem, to remis ma lepszą wartość funkcji oceny. Gra nie jest ustawiona, tzn. nie ma wpływu początkowy gracz.

5.3 Przebieg gry

Załączone rysunki jasno wskazują, że kolejne ruchy graczy wybierane były na podstawie drzewa decyzyjnego zbudowanego na podstawie algorytmu minimax.

6 Podsumowanie

Udało się zaimplementować działający algorytm minimax, pozwalający na znaczne zwiększenie szansy wygranej w stosunku do gracza losowego. Wyniki wskazują, że zaproponowana funkcja heurystyczna powinna zostać udoskonalona.