

Zastosowanie algorytmu UCT do stworzenia sztucznej
inteligencji grającej w *Domineering*
Wstępna dokumentacja projektu

Patryk Fijałkowski
Mateusz Burczaniuk

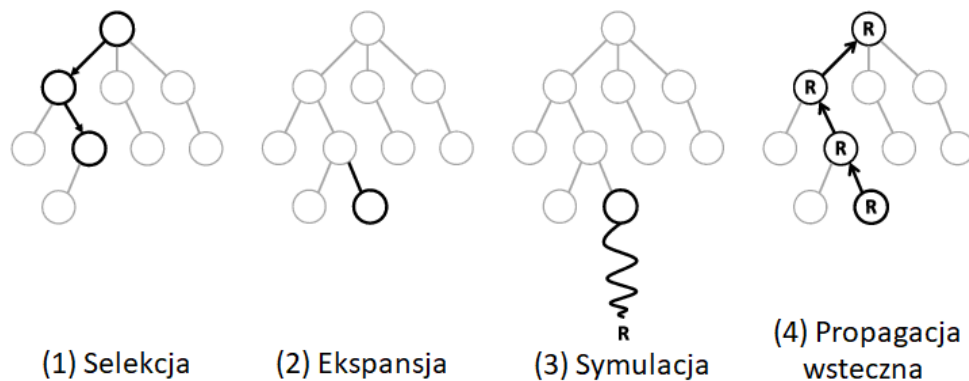
19 kwietnia 2020

1 Opis *Domineering*

TODO

2 Algorytmy MCTS

Monte-Carlo Tree Search to heurystyka, której celem jest podejmowanie decyzji w pewnych zadaniach sztucznej inteligencji, na przykład wybieranie ruchów w grach. Metoda jest oparta na przeszukiwaniu możliwych stanów gry zapisanych w wierzchołkach drzewa i losowym symulowaniu rozgrywek. Algorytmy MCTS opierają się na rozbudowywaniu drzewa ze stanami gry przez iteracyjne wykonywanie czterech faz. Jednym z najpopularniejszych wariantów MCTS jest algorytm UCT. Pseudokod opisany w Listing 1 oraz implementacja MCTS w projekcie bazują na [1]. Przykład działania algorytmu ze szczególnym uwzględnieniem kolejnych faz znajduje się na Rysunku 1.



Rys. 1: Fazy MCTS, źródło: [2]

1. **Faza selekcji** (wiersz 6 w listingu) – wybór pewnego liścia drzewa. Rozdział 2.1 opisuje jeden ze sposobów na wybranie wierzchołka w tej fazie.
2. **Faza ekspansji** (wiersz 7 w listingu) – utworzenie wierzchołków potomnych dla wierzchołka wybranego w fazie selekcji. Tworzone wierzchołki odpowiadają stanom możliwym do uzyskania przez wykonanie jednego ruchu ze stanu rodzica.
3. **Faza symulacji** (wiersz 10 w listingu) – rozegranie partii składającej się z losowych ruchów ze stanu jednego z wierzchołków utworzonych w poprzedniej fazie. Rozgrywana jest ona do końca, czyli do wyłonienia zwycięzcy lub spowodowania remisu, lub jest ucinana po pewnej liczbie ustalonych ruchów i wynik gry jest ewaluowany przez pewną funkcję.
4. **Faza propagacji wstecznej** (wiersz 11 w listingu) – aktualizacja informacji na temat wierzchołków na ścieżce od liścia, z którego rozpoczęto symulację, do korzenia drzewa. Główną przekazywaną wartością jest wynik symulacji.

```

1 def find_next_move(curr_state):
2     iterations_counter = 0
3     tree = initialize_tree(curr_state)
4
5     while iterations_counter < max_iterations_counter:
6         curr_node = select a leaf from tree
7         create child nodes from curr_node
8         if curr_node has children:
9             curr_node = random child of curr_node
10        playout_result = simulate random playout from curr_node
11        update tree according to playout_result
12        iterations_counter++
13
14    best_state = select best child(tree.root)
15    return best_state
16

```

Listing 1: Pseudokod algorytmu Monte Carlo Tree Search

2.1 Algorytm UCT

UCT jest wariantem metody MCTS, który stara się zachować równowagę między eksploatacją bardziej obiecujących ruchów a eksploracją tych rzadko odwiedzonych. Formuła, która odpowiada za wyznaczenie najbardziej obiecującego wierzchołka w fazie wyboru MCTS jest przedstawiona jako wyrażenie (1).

$$\frac{w_i}{n_i} + c\sqrt{\frac{\ln N_i}{n_i}} \quad (1)$$

W wyrażeniu (1), indeks i odnosi się do liczby wykonanych przez algorytm iteracji, czyli czterech faz MCTS. W pierwszym składniku sumy wyrażenia (1), licznik w_i oznacza sumę wszystkich wypłat w danym węźle, a mianownik n_i oznacza liczbę rozegranych symulacji. Zatem ułamek ten przyjmuje wartości większe dla ruchów o większej średniej wygranej, co odpowiada ze eksploatację drzewa. Drugi składnik sumy wyrażenia (1) przyjmuje wartości większe dla wierzchołków, dla których wykonano mniej symulacji i odpowiada eksploracji drzewa. $N_i = \sum_i n_i$, a c jest parametrem eksploracji, który może być dostosowany do badanego problemu.

2.2 Algorytm UCT - wariant 1

TODO

2.3 Algorytm UCT - wariant 2

TODO

3 Algorytm zachłanny

TODO

4 Rozwiązanie

5 Hipotezy badawcze

6 Harmonogram działań

W tabeli 1 przedstawiono planowany harmonogram działań podczas pracy nad projektem.

Tab. 1: Harmonogram pracy

Deadline	Przygotowane zadania
06.05.2020	Stworzenie dokumentacji projektu Zaimplementowanie struktur potrzebnych do operowania na drzewach
13.05.2020	Zaimplementowanie logiki gry Przygotowanie aplikacji okienkowej
20.05.2020	Zaimplementowanie algorytmu UCT
07.05.2020	Przeprowadzenie eksperymentów w celu weryfikacji hipotez
03.06.2020	Wstępna weryfikacja postawionych hipotez
10.06.2020	Pełna weryfikacja postawionych hipotez Stworzenie raportu

7 Projekt techniczny

Projekt zostanie sporządzony przy użyciu języka C#. Nie jest planowane użycie żadnych specjalistycznych bibliotek tego języka.

Literatura

- [1] Levente Kocsis, Csaba Szepesvári, *Bandit based Monte-Carlo Planning*, European Conference on Machine Learning, Berlin, Germany, September 18–22, 2006.
- [2] Steven James, George Konidaris, Benjamin Rosman, *An Analysis of Monte Carlo Tree Search*, University of the Witwatersrand, Johannesburg, South Africa.
- [3] K. Marriott, *NP-Completeness of Minimal Width Unordered Tree Layout*, Journal of Graph Algorithms and Applications, vol. 8, no. 3, pp. 295–312 (2004).
- [4] Christop Buchheim, Michael Jünger, Sebastian Leipert, *Improving Walker’s Algorithm to Run in Linear Time*, Universität zu Köln, Institut für Informatik, 2002.