Szachy

Daniel Śliwowski

nr ideksu: 241166, Termin: ŚR 11:15, Data: 03.06.2019,
Prowadzący: Dr inż. Łukasz Jeleń

1 Wprowadzanie

Celem niniejszego projektu było stworzenie silnika do gry w szachy. Głównym założeniem była prostota wykonania, z tego powodu zdecydowano się na niektóre rozwiązania. W wnioskach podane są możliwe ulepszenia.

SI oparte jest na algorytmie alfa-beta, zamiast na min-max'ie w celu większej szybkości działania, co zwiększa poziom gry. Dwoma najistotniejszymi elementami algorytmu jest ewaluacja planszy oraz sortowanie wyników. Zagadnienia te zostaną omówione dokładnie w dalszej części sprawozdania.

2 Omówienie silnika gry

Rozdział ten zostanie poświęcony omówieniu implementacji podstawowych elementów silnika gry i zasady jego działania.

2.1 Reprezentacja planszy

Plansza reprezentowana jest jako dwuwymiarowa tablica znaków. Puste pola reprezentowane są jako spacje, figury gracza jako duże litery a figury komputera jako małe. Przykładowo plansza na początku gry:

r	k	b	q	a	b	k	r
p	p	p	p	p	p	p	p

P	P	P	P	P	P	P	P
R	K	B	Q	A	B	K	R

Tabela 1: Reprezentacja planszy

Gdzie: p/P - pionek, r/R - wieża, k/K - skoczek, b/B - goniec, q/Q - hetman, a/A - król.

Zaletami takiego zastosowania są przejrzystość, łatwość debugowania oraz zrozumienia działania. Wadami są zwiększony czas operacji wynikający z potrzeby sprawdzania każdego pola z osobna oraz utrudnienie w implementacji niektórych zasad ruchów.

2.2 Reprezentacja ruchu

Ruchy reprezentowane są w zależności od rodzaju jako ciąg znaków. Dla ruchu normalnych figur oraz pionków bez promocji format jest następujący: x1y1x2y2z, gdzie x1,y1 - położenie startowe, x2,y2 - położenie końcowe, z - zbita figura.

W przypadku promocji pionka sekwencja ma postać: y1y2znP, gdzie y1 - kolumna początkowa, y2 - kolumna końcowa, z - zbita figura, n - jaka jest nowa figura, P - oznaczenie promocji pionka.

Dla gracz zaimplementowano jedynie możliwość wyboru hetmana.

2.3 Algorytm alfa-beta

Algorytm alfa-beta jest ulepszeniem algorytmu min-max. Zakłada on dwie wartości alfa i beta, które pozwalają na odrzucenie części drzewa, w których na pewno nie będzie rozwiązania, a dzięki

temu znaczne przyśpieszenie wykonywania. Alfa jest maksymalną wartością dolnej granicy, a beta minimalną wartością górnej granicy. Zatem wynik N musi spełniać:

$$\alpha \leqslant N \leqslant \beta.$$

W momencie gdy $\alpha \geqslant \beta$ następuje odrzucenie tej gałęzi rozwiązań.

W celu ułatwienia zrozumienia, zaimplementowany algorytm zawsze wykonuje ruchy z perspektywy figur gracza, w wyniku tego potrzebna jest funkcja odwracająca plansze. Skutkiem tego jest również to, że wynik przeciwnika jest zawsze ujemny, zatem należy przemnożyć go razy -1 w celu poprawnego działania algorytmu.

2.4 Ewaluacja planszy

Ewaluacja jest jedną z istotniejszych elementów silnika gry w szachy. Ma ona bezpośredni wpływ na jakość gry SI i poprawnie napisana stanowi kompromis pomiędzy szybkością działania a dokładnością.

Do ocenienia jakości wykonanego ruchu silnik ocenia ilość posiadanych figur, miejsce ich położenia, mobilność oraz czy są atakowane przez figury przeciwnika. Wynik otrzymany z tej oceny dodawany jest w dla swoich bierek, a odejmowany w przypadku bierek przeciwnika.

2.4.1 Ocena posiadanych figur

Wartości punktowe poszczególnych figur zostały zaczerpnięte z oficjalnych zasad szachów i przemnożone przez 100 w celu uzyskania większej dokładności. Zatem:

1. Pionek - 100,
2. Wieża - 500,
3. Skoczek - 300,
4. Hetman - 900.

Wyjątkiem tej reguły stanowią gońce, których wartość jest większa w momencie, gdy na planszy znajduje się para, wówczas każdy goniec warty jest: **300 * ilość posiadanych gońców**. Gdy na planszy jest tylko 1, to jego wartość wynosi **250**.

2.4.2 Ocena położenia

W zależności od figury jej położenie na planszy ma różny wpływ na rozgrywkę, np. skoczek znajdujący się w środku planszy jest groźniejszy niż takie, który stoi w rogu. Celem uzyskania takiej punktacji skorzystano z tablic wartości położenia zaproponowane przez Wikipedię do programowania szachów [**?**].

Szczególnym przypadkiem jest król, który ma dwie tablice. Wynika to z faktu, że na początku gry pozycje za linią pionków są dla króla bezpieczniejsze, jednak sytuacja ta ulega zmienia w przypadku późnej fazy gry, gdzie król jest bezpieczniejszy na środku planszy. Jako końcową fazę gry zdefiniowano moment, gdy wartość pozostałych figur jest mniejsza niż suma wartość wszystkich pionków i hetmana.

2.4.3 Ocena mobilności

Sytuacja, gdy gracz ma większa ilość możliwych ruchów jest bardziej korzystna, zatem za każdy możliwy ruch otrzymuje 5 pkt.

W tym miejscu możliwa jest też ocena, czy mamy sytuacje patową, czy szach mat. Obie są niekorzystne, zatem w momencie, gdy silnik wykryje szach mat odejmuje 200000 punktów, a w sytuacji patowej 150000. Wartości te są skalowane przez głębokość rekurencji, aby zniechęcić silnik do przeszukiwania gałęzi prowadzących do szachu.

2.4.4 Ocena ataku

Sytuacja, gdy figury gracza atakowane są przez przeciwnika zmusza go do ruchu, zatem za każdą atakowaną figurę odejmujemy połowę jej wartość punktowej. Wynika to z tego, że atakowanie nie jest równoznaczne z przejęciem.

2.5 Sortowanie

Aby przyśpieszyć działanie algorytmu zaimplementowano prostą metodę sortowania. Działą ona na zasadzie ułożenia 6 najlepszych ruchów w malejącej kolejności na początku listy ruchów. Nie ma potrzeby sortowania całej listy, ponieważ istnieje duże prawdopodobieństwo, że pozostałe ruchy zostaną odrzucone przez algorytm.

Dzięki zastosowaniu sortowania czas potrzebny na znalezienie optymalnego ruchu zmalał z 16 s na 3-4 s

2.6 GUI

Szata graficzna programu została zaprogramowana za pomocą biblioteki GLFW3, która stanowi API do biblioteki OpenGL. W celu wczytania do pamięci obrazów figur użyto biblioteki stb_image. Rys. 1 przedstawia wygląd programu.

3 Wnioski

W większości udało się stworzyć działający silnik do gry w szachy. W związku z sposobem reprezentacji planszy nie udało się zaimplementować zasad ruchu roszady oraz en passant. Sortowanie listy możliwych ruchów znacząco zwiększyło prędkość działania algorytmu alfa-beta.

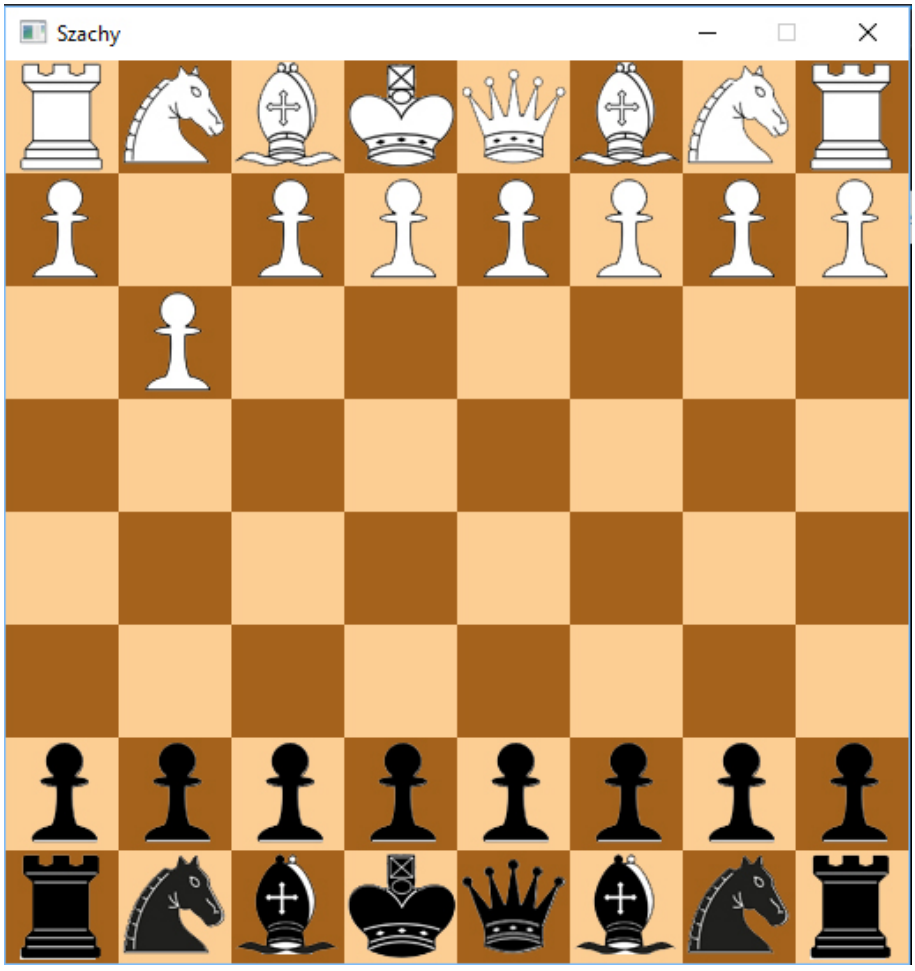
4 Możliwe ulepszenia

1. Zmiana reprezentacji planszy na tablice bitowe spowodowałoby znaczący wzrost szybkości działania oraz ułatwiłoby implementacje niektórych zasad ruchu.

2. Zaimplementowanie protokołu UCI umożliwiłoby wykorzystanie bardziej złożonych opraw graficznych takich jak, np. Chess Arena. Dawało by to możliwość porównania stworzonego silnika do innych.

3. Ewaluacja jest elementem silnika, nad którym można ciągle pracować, by otrzymać co raz to lepsze silniki kosztem czasu działania.

4. Dodajne ruchów z perspektywy przeciwnika zmniejszy czas działania algorytmu, ponieważ nie będzie istniała potrzeba odwracania planszy.



Rysunek 1: Wygląd programu

Literatura

[1] Wartość położenia [04.06.2019] - https://www.chessprogramming.org/Simplified_Evaluation_Function

[2] Poradnik ułatwiający zrozumienie zagadanienia programownaia szachów [04.06.2019] - https://www.youtube.com/watch?v=a-2uSg4Kvb0&list=PLQV5mozTHmaffB0rBsD6m9VN1azgo5wXl&index=1

[3] Opis algorytmu Alfa-Beta [04.06.2019] - http://web.cs.ucla.edu/ rosen/161/notes/alphabeta.html