Wydział Elektroniki i Technik Informacyjnych
Politechnika Warszawska

Wprowadzenie do sztucznej inteligencji

Sprawozdanie z ćwiczenia nr 3

Tymon Kobylecki

Warszawa, 2022

Spis treści

1. Wstęp

W niniejszym sprawozdaniu opisane zostało rozwiązanie zadania oraz eksperymenty dotyczące zadania nr 3 polegającego na implementacji algorytmu minimax z obcinaniem $\alpha - \beta$. Całość ćwiczenia została wykonana w języku Python. Wykorzystana została implementacja gry Connect 4 dostępna w repozytorium `https://github.com/lychanl/two-player-games`.

2. Ćwiczenie

2.1. Eksperymenty

Eksperymenty w niniejszym ćwiczeniu laboratoryjnym polegały na wielokrotnym uruchamianiu symulacji rozgrywki Connect 4, w której obydwu graczy zastąpiono botami. Zmienianym parametrem były: głębokość przeszukiwania, manipulowana od 1 do 5 ruchów wprzód, gracz zaczynający grę (min lub max).

2.2. Środowisko - gra Connect 4

Środowiskiem, wewnątrz którego przebiegała symulacja, była gra Connect 4. Jej celem jest połączenie czterech żetonów w nieprzerwany prosty odcinek na planszy, na którą oddziałuje grawitacja. Każdy włożony żeton spada więc na sam dół planszy, lub na ostatnio wrzucony do tej samej kolumny żeton.

2.3. Wyniki

Jako, że symulacja rozgrywki gry planszowej nie dostarcza miarjodajnych danych, które można by przedstawić na wykresie, zaprezentowane zostaną jedynie statystyki zwycięstw graczy. Wyniki te znajdują się w tabeli 2.1 dla wersji algorytmu bez obcinania oraz w tabeli 2.2 dla wersji z obcinaniem.

2.4. Analiza wyników

Przy bardzo głębokim przeszukiwaniu algorytm sprawiał wrażenie „głupszego" - zdarzało mu się popełniać z pozoru bardzo proste błędy. Najprawdopodobniej wynika to jednak z „defetyzmu" (lub nadmiernego optymizmu) algorytmu - w momencie, gdy zauważy, że gra się skończy niezależnie od jego najbliższego ruchu, zacznie losować swój aktualny ruch. Wynika to z faktu, że funkcja heurystyki zwraca wartość ± 5000 niezależnie od głębokości, na której zostanie znalezione

Zwycięzca/głębokość przeszukiwania	1	2	3	4	5
max	25	8	13	11	14
min	0	5	12	7	11
remis	0	12	0	7	0

Tab. 2.1. Tabela wyników dla algorytmu minimax w wersji standardowej

Zwycięzca/głębokość przeszukiwania	1	2	3	4	5
max	25	10	13	10	13
min	0	6	11	6	10
remis	0	9	1	9	2

Tab. 2.2. Tabela wyników dla algorytmu minimax w wersji z obcinaniem $\alpha - \beta$

rozwiązanie. W związku z tym zdarza się, że bot nie wykorzysta swojej okazji do zwycięstwa od razu, jeśli ma wiele opcji wygranej, a głębokość jest większa od 2. Bot „wie" wówczas, że i tak zdąży wygrać, w związku z czym losuje swój następny ruch (z wyłączeniem ruchów, po których przeciwnik może sprzątnąć mu zwycięstwo sprzed nosa). Analogicznie działa to dla bota przegrywającego, któremu zdarza się „pozwolić" przeciwnikowi wygrać w najprostszy sposób, w momencie, gdy głębokość przeszukiwania pozwala mu dostrzec, że przegra niezależnie od swoich działań. Innym łatwo dostrzegalnym aspektem jest zdecydowanie mniejsza liczba remisów przy nieparzystej (3 remisy na przestrzeni wszystkich eksperymentów) niż przy parzystej (średnio 9,25 remisów na eksperyment składający się z 25 uruchomień programu przy określonej głębokości) mierze głębokości przeszukiwania. Wynika to z faktu, że wówczas liść stanowi rezultat gracza aktualnie ruszającego się, zaś przy parzystej liczbie - przeciwnika. Wobec tego, przy parzystej głębokości przeszukiwania algorytm bardziej „skupia się" na blokowaniu przeciwnika, niż na własnym zwycięstwie, co przekłada się na większą liczbę remisów. Większy procent wygranych gracza maksymalizującego wynika z faktu, że ów gracz zawsze zaczynał, co z automatu daje przewagę.

2.5. Wersja bez obcinania α - β

W wersji bez obcinania algorytm bardzo skutecznie radził sobie z rozgrywką, dzięki czemu często zachodziły remisy, a gdy któryś z botów wygrywał, to wynikało to z „przyparcia do muru" drugiego gracza, tj. zwycięzca miał co najmniej dwie opcje wygranej.

2.6. Wersja z obcinaniem α - β

Wersja z obcinaniem α - β działa niemal identycznie jak podstawowa wersja algorytmu minimax, jednak jest szybsza, gdyż nie rozważa gałęzi, na których przeciwnik ma korzystniejszy ruch niż aktualnie ruszający się gracz.

2.7. Wnioski

Algorytm dobrze radzi sobie z grą w Connect 4, jednak w pojedynku dwóch takich samych botów nie zawsze to widać. Eksponuje również wady samej gry, która daje przewagę graczowi zaczynającemu.