Vyhľadávanie najlepšieho ťahu v šachu*

Martin Kubiš
Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
xkubis@stuba.sk

5. november 2023

Abstrakt

Šach je strategická stolová hra, ktorá sa skladá z šachovnice o rozmeroch 8x8 a 6 typov figúrok vo dvoch farbách. Aj keď sa na prvý pohľad zdá ako jednoduchá hra, neprestáva trápiť mozgy hráčov jak začiatočníkov, tak profesionálov. Od 50. rokov 20 storočia, kedy sa vyvinul prvý šachový robot, sa vývoj nezastavil a počítače už sú na míle ďaleko pred ľudskými hráčmi. Cielom článku je priblíženie čitateľovi metódy a techniky šachových algoritmov, ich optimalizácia. Taktiež aj porovnanie medzi klasickými šachovými algoritmami a algoritmami obohatené o umelú inteligenciu a akým spôsobom vyhľadávajú najlepší možný ťah. ...

1 Úvod

Hranie šachu si vyžaduje veľa zručností, strategické myslenie, plánovanie určitých ťahov, schopnosť predvídať a byť nepredvídateľný. Aby sa hráči zlepšili, či už na profesionálnej alebo amatérskej úrovni, musia rozvíjať svoje zručnosti a znalosti o šachových algoritmoch a stratégiach, ktoré sú kľúčom k úspechu v tejto komplexnej hre.

V tomto článku sa zameriame na najdôležitejšie algoritmy ako sú minimax algoritmus s alfa-beta pruningom, monte carlo tree search algoritmus, alebo využitie deep learningu, neurálnych sietí a umelej inteligencie a aké sú stratégie na vyhľadanie čo najlepších možných ťahov na prekonanie súpera. a poukážeme na využitie týchto algoritmov v modernom šachovom svete. Ďalej sa budeme venovať technickým výzvam spojeným s vývojom šachových programov a ich schopnosťou hrať na úrovni šachových majstrov.

2 Minimax algoritmus

Minimax algoritmus je rozhodovací algoritmus používaný v hrách na určenie najlepšieho možného ťahu pre hráča v hre pre dvoch hráčov s nulovým súčtom, ako je napríklad šach, piškôrky alebo dáma.

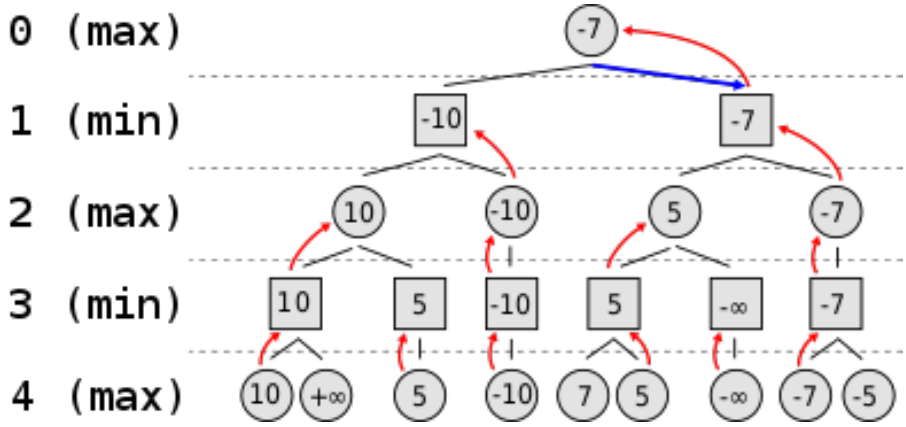
V hre s nulovým súčtom sú zisky jedného hráča presne vyvážené stratami druhého hráča, takže súčet ziskov a strát hráčov je vždy nula.

*Semestrálny projekt v predmete Metódy inžinierskej práce, ak. rok 2023/24, vedenie: Vladimír Mlynarovič

2.1 Princíp fungovania Minimax algoritmu

Minimax algoritmus funguje tak, že predpokladá, že obaja hráči budú hrať optimálne, a potom určí najlepší ťah pre hráča, ktorý práve robí ťah. Hráč 1 sa snaží získať, čo najväčší počet bodov, zatial čo hráč 2 sa snaží získať čo najmenší počet bodov.

Robí to tak, že zvažuje všetky možné ťahy, ktoré by hráč mohol urobiť, a potom hodnotí každý z týchto ťahov simulovaním zvyšku hry za predpokladu, že súper urobí svoj najlepší možný ťah. Algoritmus potom vyberie ťah, ktorý vedie k najlepšiemu výsledku pre hráča, za predpokladu, že súper ako odpoveď urobí svoj najlepší možný ťah. [7]



Obr. 1: Jednoduchá ukážka minimax algoritmu

2.2 Výhody a obmedzenia Minimax algoritmu

Minimax algoritmus je výkonný rozhodovací algoritmus, ktorý poskytuje optimálne riešenia v zmysle teórie hier. Tento algoritmus má však niekoľko obmedzení.

Prvým je jeho výpočtová náročnosť, najmä v hrách s veľkým rozhodovacím priestorom. Preto boli vyvinuté rôzne varianty a optimalizácie, ako napríklad:

- Alpha-Beta pruning
- Otváracie ťahy (Openings) [2]

Vďaka týmto optimalizáciam sa zníži počet vetví, cez ktoré musí algoritmus prejsť, taktiež sa zlepší celkový výkon algoritmu.

3 Alpha-Beta pruning

V praxi väčšina AI šachových robotov používa optimalizovanú verziu minimax algoritmu známeho ako alfa-beta pruning algoritmus. Základnou myšlienkou alfa-beta pruningu je zníženie počtu vetiev, ktoré je potrebné hodnotiť „prerezávaním“(pruningom) konárov, ktoré istotne nepovedú k lepšiemu výsledku ako tie, ktoré už boli hodnotené.

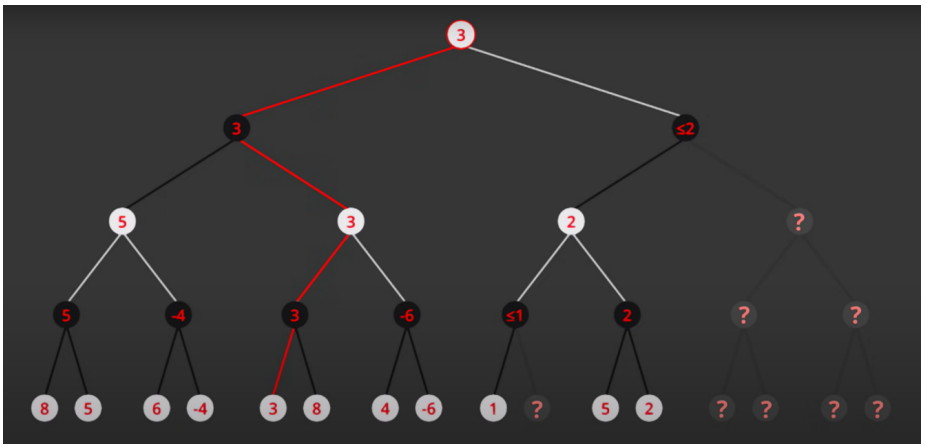
To sa dosiahne použitím dvoch hodnôt, alfa a beta, na sledovanie najlepšieho výsledku, ktorý bol doteraz zistený pre maximalizujúceho hráča (alfa) a najlepšieho výsledku, ktorý bol doteraz nájdený pre minimalizujúceho hráča (beta). [6]

3.1 Princíp fungovania alpha-beta pruningu

Algoritmus funguje tak, že rekurzívne skúma strom hry a vyhodnocuje každý možný pohyb a výsledok pomocou algoritmu minimax. Pri skúmaní každej vetvy stromu aktualizuje hodnoty alfa a beta na základe najlepšieho výsledku, ktorý bol doteraz nájdený.

Ak nájde vetvu, v ktorej je výsledok pre minimalizujúceho hráča (beta) horší ako aktuálny najlepší výsledok pre maximalizujúceho hráča (alfa), prereže túto vetvu a prestane ju vyhodnocovať, pretože minimalizujúci hráč by tento ťah nikdy nezvolil.

Algoritmus postupuje rekurzívne nahor po strome, pričom neustále aktualizuje hodnoty alfa a beta, kým nevyhodnotí všetky možné pohyby a výsledky, alebo kým neoreže všetky vetvy, ktoré sú zaručene horšie ako aktuálny najlepší výsledok.



Obr. 2: ukážka alpha-beta pruningu

4 Monte Carlo Tree Search

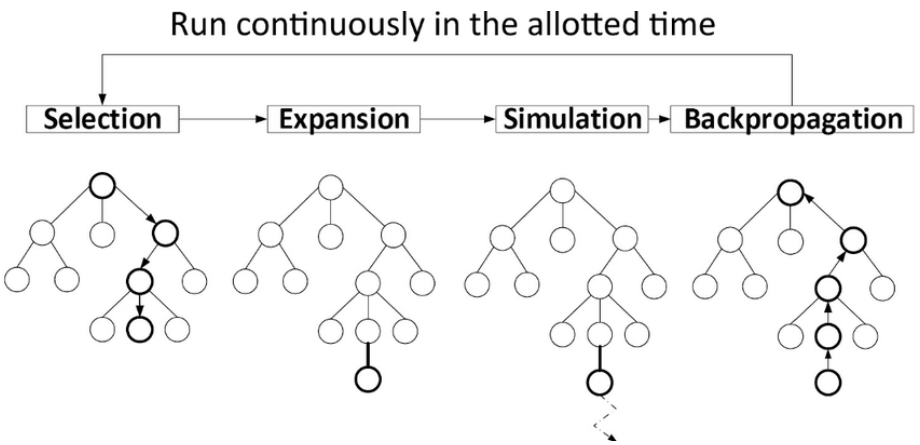
Monte Carlo Tree Search (MCTS) je inovatívny a výkonný algoritmus v oblasti umelej inteligencie a teórie hier, ktorý bol využitý na rôzne stolové hry, ako je napríklad Go a šach. Tento algoritmus výrazne prekonáva tradičné techniky vyhľadávania v stromovom priestore, čím umožňuje získať vysokokvalitné výsledky v hrách s veľkými rozhodovacími priestormi.

4.1 Princíp fungovania MCTS

MCTS sa líši od predchádzajúcich algoritmov Minimax tým, že neprehľadáva celý stromový priestor. Namiesto toho sa používa metóda vzorkovania, pri ktorej

sa ťahy vyberajú a simulujú až do konca hry, aby sa získal odhad aktuálneho stavu hry [8]. Opakuje tento proces niekoľkokrát a na základe získaných hodnôt sa rozhodne, ktorý ťah je najlepší. V MCTS existujú štyri základné fázy:

1. výber (Selection)
2. šírenie (Expansion)
3. simulácia (Simulation)
4. spätné šírenie (Backpropagation)



Obr. 3: ukážka fungovania MCTS

5 Umelá inteligencia a šach

Význam umelej inteligencie pri hraní šachu je obrovský. Moderné šachové programy sú schopné hrať na úrovni a dokonca aj porážať majstrov sveta v šachu a stali sa cenným nástrojom pre hráčov všetkých úrovní.

Umelá inteligencia umožňuje hráčom analyzovať svoje hry, trénovať, zlepšovať sa a objavovať nové stratégie. Vývoj v tejto oblasti pokračuje a umelá inteligencia bude hrať v budúcnosti šachu čoraz dôležitejšiu úlohu.

5.1 Alpha zero

Zlomovým bodom pre AI bolo predstavenie programu AlphaZero od DeepMind v roku 2017. [1] AlphaZero využíva kombináciu Deep Learningu a MCTS (Monte Carlo Tree Search) na samoučenie hier. Program porazil najlepšie šachové programy na svete a dosiahol skvelé výsledky.

AlphaZero prináša nový pohľad na tréning umelej inteligencie, pretože sa učí šach bez ľudského tréningu alebo heuristiky, pričom dosahuje predtým nepredstaviteľnej úrovne. To znamená, že program sa neučil z predrom definovaných pravidiel, alebo ľudských stratégii, ale spoliehal sa na strojové učenie a analýzu veľkého množstva dát, ktoré sám generoval pri hraní šachu.

5.2 význam AI v šachu

Umelá inteligencia prináša radu významných prínosov do sveta šachu

1. výkon a sila

   AI umožnuje vytvárať šachové programy, ktoré sú o dosť výkonnejšie, ako najlepší ľudskí hráči

2. Samo-učenie

   Pomocou AI sa šachové programy vedia učiť samé, čo znamená, že niesú obmedzené ľudskými znalosťami alebo stratégiami [3]. Napr. program AlphaZero, ako bolo už vyššie spomenuté.

3. Analýza a tréning

   Umelá inteligencia je už dnes bežne využívaná hráčmi na analýzu svojich hier, a na tréning. Šachové programy poháňané umelou inteligenciou sú skvelými učitelmi pre hráčov všetkých úrovní, keďže sa dokážu prispôsobovať svojim protivníkom.

6 Záver

Literatúra

[1] Chess.com. Alphazero chess engine. `https://www.chess.com/terms/alphazero-chess-engine`.

[2] Chess.com. Chess openings. `https://www.chess.com/openings/`.

[3] Omid E. David, Nathan S. Netanyahu, and Lior Wolf. Deepchess: End-to-end deep neural network for automatic learning in chess. In Alessandro E.P. Villa, Paolo Masulli, and Antonio Javier Pons Rivero, editors, *Artificial Neural Networks and Machine Learning – ICANN 2016*, pages 88–96, Cham, 2016. Springer International Publishing.

[4] Omid E. David, H. Jaap van den Herik, Moshe Koppel, and Nathan S. Netanyahu. Genetic algorithms for evolving computer chess programs. *IEEE Transactions on Evolutionary Computation*, 18(5):779–789, 2014.

[5] Tiantian Guo, Hongkun Qiu, Bairui Tong, and Yajie Wang. Optimization and comparison of multiple game algorithms in amazon chess. In *2019 Chinese Control And Decision Conference (CCDC)*, pages 6299–6304, 2019.

[6] Donald E. Knuth and Ronald W. Moore. An analysis of alpha-beta pruning. *Artificial Intelligence*, 6(4):293–326, 1975.

[7] Richard E. Korf. Depth-first iterative-deepening: An optimal admissible tree search. *Artificial Intelligence*, 27(1):97–109, 1985.

[8] Świechowski M., K. Godlewski, and B. Sawicki. Monte carlo tree search: a review of recent modifications and applications. `https://link.springer.com/article/10.1007/s10462-022-10228-y`.