Microsoft

Studentské Trenérské Centrum

Absolventská práce, ročník 2021

AI v teorii her

AI in game theory

Jakub Wodecki

31.12. 2022

Abstrakt

Cílem absolventské práce je si vybrat strategickou deskovou či karetní hru. Následně navrhnout, implementovat a diskutovat algoritmus, který se bude snažit o maximalizaci úspěchu v dané hře.

Abstrakt

Goal of this graduation project is to choose a strategic tabletop or card game. Design, implement and discuss algorithm that will try to maximize success in chosen game.

Osnova

1. Úvod

Cílem absolventské práce je si vybrat strategickou deskovou či karetní hru. Následně navrhnout, implementovat a diskutovat algoritmus, který se bude snažit o maximalizaci úspěchu v dané hře.

Další kapitoly této absolventské práce budou pojednávat o vybrané hře, teoretické části vývoje a následně o implementaci těchto teoretických modelů. V průběhu jsou diskutovány jednotlivé postupy a odůvodněny i vybrané postupy v implementační části.

2. Použité technologie

Tato absolventská není technologicky nijak náročná. Při vývoji tedy byl použit pouze jeden programovací jazyk a vývojářské prostřední, pro jeho použití.

Jako programovací jazyk jsem zvolil jazyk Java verze 17.0.2 LTS. Tento jazyk jsem zvolil z důvodu, že mám s ním největší zkušenosti, a proto je mi nejbližší. Použil jsem verzi 17, protože je to poslední nejnovější LTS[1] verze tohoto jazyka.

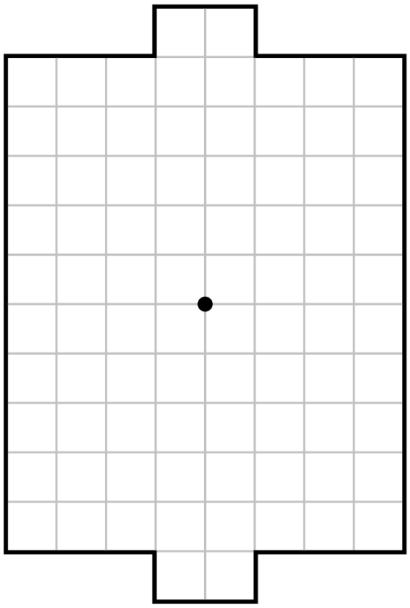
V neposlední řadě jsem používal vývojářské prostředí IntelliJ IDEA Community Edition od firmy JetBrains kvůli její široké škále nástrojů pro správu kódu a příjemnému vzhledu.

3. Vybraná hra

Jako první věc bylo zapotřebí vybrat si logickou nebo karetní hru na které budu pracovat. Ideálně by měla být složitější než piškvorky, ale jednodušší než šachy. Proto jsem si vybral hru s názvem Papírový Fotbal[2]. Kvůli tomu, že bude tato hra hraná v počítači jsem daný projekt pojmenoval jako Code Soccer.

Hra se standardně hraje na kostkovaném papíře, kde se vytyčí hrací pole o lichém počtu sloupců a lichém počtu řádků. Standardně se však hraje na hřišti o rozměrech 9 sloupců a 11 řádcích. Míč se vykopává ze středu těchto souřadnic a následně se může pohybovat po průsečících těchto linek vertikálně, horizontálně i diagonálně.

[1] Long Time Support – Tato verze bude nadále udržována a opravována po dobu 5 let od jejího vydání
[2] V originálním anglickém názvu Paper Soccer



Papírový Fotbal 1 - Vzhled hracího pole

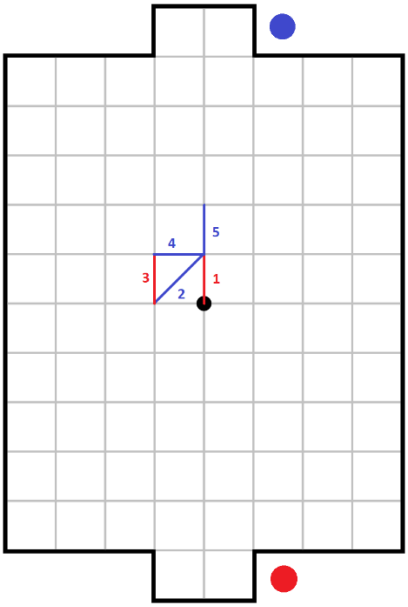
Hráči se střídají ve svých tazích a aby jeden z hráčů vyhrál, musí dostat míč do oponentovy brány, nebo se jeho oponent musí dostat do stavu, kdy nemůže provést žádný legální pohyb[3]. Brány jsou umístěny na dvou protějších stranách hřiště. Zpravidla jsou nahoře a dole, jejich rozměry jsou 2 čtverce na 1 čtverec a jsou zarovnány na střed pole podle sloupců.

Legální pohyby míče po hřišti jsou takové, které neporušují tyto zásady:

- Pohyb ještě nebyl v minulosti dané hry proveden.
- Pohyb není veden mimo hrací plochu.
- Oba body daného pohybu neleží na stejné hranici hracího pole.
- Pohyb musí být právě hloubky 1.

Hra obsahuje jednu speciální mechaniku, která je označována jako odražení. Odražení nastane ve chvíli, kdy se míč dostane do bodu v poli, ve kterém už někdy za dobu hry byl, nebo se dostane na ohraničení hracího pole. Standardně se hráči střídají po jednom pohybu, ale ve chvíli, kdy dojde k odražení tak na tahu zůstává hráč, který odražení uskutečnil. Odražení vypadá následovně.

[3] Pohyb je definován pomocí dvou bodů v hracím poli – od a do, přičemž mimo lokalizaci míče nezáleží na jeho orientaci



Papírový Fotbal 2 - Mechanika odražení

4. Evaluační model

Pro práci na algoritmu protihráče je nejprve důležité vytvořit algoritmus evaluačního systému hry. Tento evaluační systém bude zodpovědný za správné nastavení hry, kontrolování pravidel hry, aby nebyly porušeny, a samotnou hru ukončí, když je naplněna jedna z podmínek výhry nebo prohry.

Střídání hráčů a kontrola pravidel je triviální, tím se zde zabývat nebudu. Bylo však podstatné vymyslet, jakým způsobem budu reprezentovat hrací pole a následně ukládat historii pohybů.

Body herního pole reprezentuji pomocí třídy *FieldPoint*, která obsahuje pořadí sloupce a řádku, které jsou indexované od 0 do počtu řádků/sloupců dané hry – 1. Branky samotné se dále nachází na souřadnicích [střed sloupců; -1] po spodního hráče a [střed sloupců; počet řádků] pro horního hráče. Samotné pohyby jsou dále uloženy v historii hry.

Pro práci s historií jsem přišel na 2 způsoby:

- Sekvence orientovaných pohybů
- HashMap bodů a jejich sousedů

1. Sekvence orientovaných pohybů

Prvním způsobem byla sekvence orientovaných pohybů. Každý pohyb by byl reprezentován číslem o 0 do 7, kdy číslo 0 znamená pohyb nahoru, číslo 1 diagonálně vpravo a nahoru a takto až do čísla 7.

Tento způsob byl použit herní platformou PlayOk.com[4]. Tímto způsobem lze jednoduše zjistit všechny předchozí lokace míče na hřišti, ale hledání v této sekvenci by bylo časově náročné. V historií pohybů však musíme hledat daný pohyb každý tah.

Tento způsob je vhodný pro zapsání průběhu dané hry, ale pro rychlé hledání v historii ideální není. Časová komplexita tohoto způsobu vždy bude *O(n)*.

2. HashMap bodů a jejich sousedů

Druhým způsobem je implementace datové struktury HashMap, která obsahuje všechny body hracího pole jako klíče a k nim asociované listy bodů, se kterými už existuje v historii nějaký pohyb.

Díky vlastnostem datové struktury HashMap dosáhneme při hledání počátečního nebo cílového bodu časové komplexity *O(1)*. Následně získaným listem můžeme jednoduše iterovat, abychom našli zadaný pohyb. Tento algoritmus jsem nakonec implementoval.

5. Algoritmus protihráče

Po vytvoření evaluačního systému je potřeba přijít na možné algoritmy počítačových protihráčů. Napadlo mě několik algoritmů, které by se daly implementovat. Každý z těchto algoritmů je pojmenován podle hlavní mechaniky, kterou používá nebo co připomíná.

1. Naivní protihráč

Naivní protihráč spočívá v tom, že každý jeho pohyb je zcela náhodný na základě možných okolních pohybů. Nezajímá ho, který pohyb může být lepší a ani který ho dostane blíže.

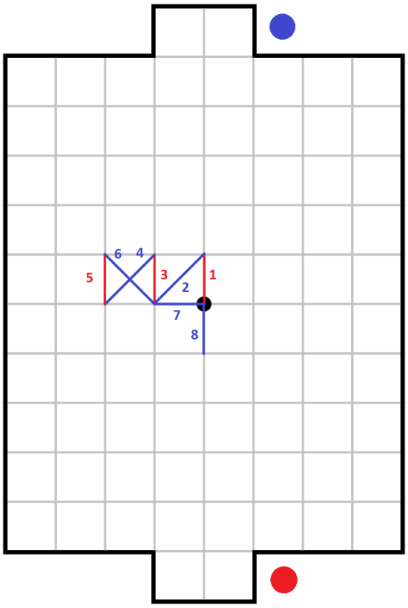
Vhledem k tomu, že nepočítá ani s mechanikou odrazu, je velmi lehké o porazit a spíše vhodný jako testovací subjekt pro otestování správnosti algoritmu hry nebo umělého protihráče. Pro maximalizaci vítězství to tedy není vhodný algoritmus.

[4] Paper soccer. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-12-30]. Dostupné z: https://en.wikipedia.org/wiki/Paper_soccer

2. Nejkratší cesta

Abych maximalizoval možnost výhry umělého protihráče tak je nutné, aby následoval nějakou taktiku. Jedna z nich může být hledání nejkratší cesty v herním poli. Tento algoritmu však skýtá několik problémů, kvůli kterým nestačí najít nejkratší cestu.

Díky mechanice odrazu není vždy ta nejkratší cesta tou nejlepší. Bez použití mechaniky odražení se často hráč nemůže dostat do výhodnější pozice. Příkladem může být následující začátek hry. Červený hráč začíná a reprezentuje nejkratší cestu a modrý hráč reprezentuje lidského hráče.

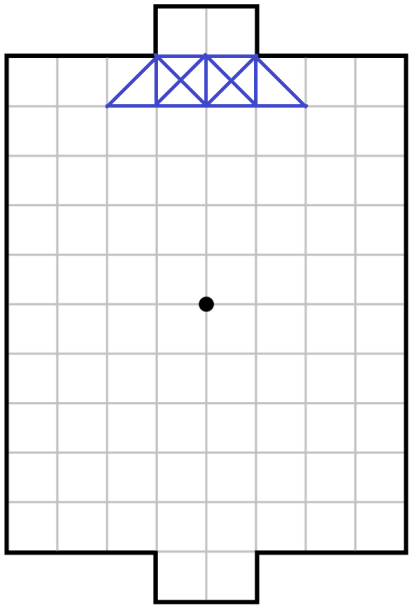


Papírový Fotbal 3 – Ilustrace algoritmu nejkratší cesty

Není proto vhodné spoléhat pouze na hledání nejkratší cesty. Druhým problémem je množství možných kombinací cest v herním poli, které bychom museli projít, abychom našli tu nejkratší cestu. Je to tedy zdlouhavé a časově neefektivní.

3. Defenzivní protihráč

Svou bránu může hráč kompletně zablokovat. Kompletní zablokování brány vypadá následovně.



Papírový Fotbal 4 - Ilustrace algoritmu obrany

V takovém případě nemá protihráč žádnou jinou možnost než se pokusit dostat defenzivního hráče do situace, kde nebude moct udělat žádný jiný pohyb. Z toho vyplývá, že by takový algoritmus musel nejprve svou bránu ubránit a následně Protihráče dostat do pozice s jediným legálním pohybem, který bude ukončující.

Taková obrana brány je velmi náročná a kvůli mechanice odrazu také velmi nebezpečná. Taková hra nejen že bude trvat velmi dlouho, ale také bude nutné následně implementovat jiný algoritmus, který se pokusí zvítězit. Z toho nám vyplývá, že tento algoritmus lze použít spíše jako výpomoc pro minimalizaci prohry. Vítězství to však přinést nemusí.

Algoritmus se spíše snaží neprohrát než vyhrát. Proto je lepší ho použít spíše jako nástroj poslední záchrany.

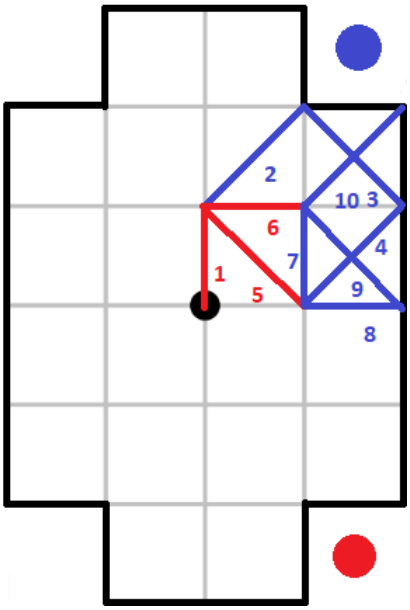
4. Poslední legální pohyb

Další algoritmus již byl nastíněn v tom předchozím. Můžeme se pokusit protihráče dostat do situace, kde bude mít poslední legální pohyb (nebo sekvenci pohybů), které ho na konci dovedou do bodu, kde už nebude mít pohyb žádný.

Takových pohybů však existuje nezměrně moc a hra už musí být do značné míry rozehraná, aby to vůbec bylo možné. Navíc je to způsob, který se v základu nesnaží vyhrát, jako spíše vydržet co nejdéle a najít způsob, jak dostat protihráče do „matu“.

Tento způsob jsem zahodil z důvodu náročné implementace a nízké šance na maximalizaci počtu vítězství.

Pro ilustraci takové hry jsem zvolil mnohem menší pole o rozměrech 5x5, kde modrý hráč udělal pár chyb, kvůli kterým mu došly možné pohyby. Tato hra, ačkoli je kvůli provedeným chybám nepravděpodobná, perfektně ukazuje možnost takové taktiky.



Papírový Fotbal 5 - Ilustrace algoritmu posledního legálního pohybu

5. Nejlepší pohyb hloubky 1

Posledním algoritmem, ke kterému jsem přišel, by byl nejlepší tah hloubky 1 z dané lokace. Pojem nejlepší by však bylo nutné definovat. Nejlepší tah může být takový, který:

- Nás dostane neblíže k bráně protivníka.
- Nás dostane do bodu, který je nejméně výhodný pro protivníka.
- Nám umožní ubránit vlastní bránu.

Tento algoritmus opravdu dokáže maximalizovat šanci výhry relativně k těm předchozím. Aby byl úspěšný, bylo by nutné vytvořit hierarchii mezi těmito možnými tahy, abychom vybrali takový, který opravdu dává největší smysl.

Takový algoritmus by se také dal velmi jednoduše vylepšit tak, že by umělý protivník nahlížel do určité hloubky napřed a hledal, jaký výsledek by pro nás mohl být ten nejlepší.

6. Implementace

Samotný kód hry je umístěn ve veřejném repozitáři organizace Student Trainee Center. Soubor README obsahuje postup, jak implementovat svůj vlastní algoritmus umělého protivníka a Wiki obsahuje dokumentaci ke kódu, která byla vygenerována z komentářů Javadoc v kódu.

Je také nutné podotknout, že součástí kódu není žádná grafická reprezentace hrané hry. Po zadání validního pohybu je cílový bod vypsán do konzole a programátor/hráč si může hru kontrolovat sám svým vlastním způsobem.

Kód je rozdělen do dvou balíčků s názvy *core* a *opponents*, přičemž mimo ně se nachází třída *Main.java* potřebný ke spuštění hry.

1. Core

Tento balíček obsahuje třídy celého životního cyklu hry. Jsou to následující třídy:

- *SoccerGame* – Šablona dané hry, která v sobě uchovává rozměry herního pole, momentální pozici míče, celou historii hry a metody pro kontrolu pravidel hry.
- *GameSetup* – Třída, která zařídí vytvoření hry a spuštění evaluačního systému.
- *GameEvaluation* – Třída, která se stará o pohyby hráčů, zjišťování poslaných pohybů, střídání tahů a ukončení hry.

Dále je zde také balíček *utils* obsahující záznam *FieldPoint,* který slouží k reprezentaci jakéhokoli bodu v hracím poli.

2. Opponents

V tomto balíčku jsou obsaženi všichni protivníci, kteří byli prakticky implementování další pomocné struktury, k jejich vytvoření. Základem je rozhraní *Opponent*, které definuje metody, které musí obsahovat každý protivník. Toto rozhraní dále implementují oba připravení protihráči – *HumanOpponent* a *NaiveOpponent*.

Lidský protivník pouze získává vstup z konzole od reálného hráče a ten naivní vytváří náhodný pohyb, který provede. Zjistí si momentální lokaci míče v poli a relativně k němu si náhodnou generací vytváří možné pohyby, které následně testuje, zda je může provést.

Dále je zde balíček *types* obsahující výčet lokací branek hráčů a druhy oponentů.

Více umělých protihráčů jsem implementovat nestihl.

Závěr

Vymýšlení algoritmů pro tuto hru bylo kvůli možnosti několikanásobného tahu díky odražení náročné. Každý nový pohyb totiž vytvoří novou možnost odražení, ale ne každé odražení je pro hráče přínosné. Při vhodné historii hry lze jednoduše za jeden takový několikanásobný tah překročit celé hrací pole.

Z tohoto důvodů náhodné výběry mají téměř nulovou šanci na výhru a je nutné zvažovat vhodnější postupy, zejména využívání odrazů. Ze všech algoritmů, které jsem zde sepsal má největší šanci na úspěch Nejlepší pohyb hloubky 1 . Ten jsem však implementovat nestihl.

Seznam obrázků

Seznam použité literatury

Paper soccer. In: Wikipedia: the free encyclopedia [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2022-12-30]. Dostupné z: https://en.wikipedia.org/wiki/Paper_soccer