VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

STRATEGICKÁ DESKOVÁ HRA S NEURČITOSTÍ

STRATEGIC GAME WITH UNCERTAINITY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **ADRIÁN TULUŠÁK**
AUTHOR

VEDOUCÍ PRÁCE — **doc. Ing. FRANTIŠEK ZBOŘIL, Ph.D.**
SUPERVISOR

BRNO 2020

Vysoké učení technické v Brně
Fakulta informačních technologií

Ústav inteligentních systémů (UITS) Akademický rok 2019/2020

Zadání bakalářské práce

Student: **Tulušák Adrián**
Program: Informační technologie
Název: **Strategická desková hra s neurčitostí**
Strategic Game with Uncertainity
Kategorie: Umělá inteligence

Zadání:

1. Seznamte se s pravidly deskových her, kde aktuální stav hry bývá pro hráče po většinu času utajený. Mezi takové deskové hry patří například hra "Scotland Yard".
2. Určete metody, které by měly být důvodně vhodné pro realizaci systému, který bude hrát tuto hru autonomně. Zvažte vedle klasických metod hraní her i metody pro počítačové učení, jako jsou například metody posilovaného učení.
3. Pro jednotlivé role figur ve hře, to znamená jak pro figuru, která je hledána, tak pro figury, které ji hledají, implementujte algoritmy řízení a ověřte jejich schopnost plnit zadané cíle.
4. Vyhodnoťte úspěšnost obou stran hry pro různé míry zapojení metod strojového učení a diskutujte zjištěné výsledky.

Literatura:

1. Russel, S., Norvig, P.: Artificial Intelligence, A Modern Approach, Pearson, 2009

Pro udělení zápočtu za první semestr je požadováno:

- První dva body zadání.

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce: **Zbořil František, doc. Ing., Ph.D.**
Vedoucí ústavu: Hanáček Petr, doc. Dr. Ing.
Datum zadání: 1. listopadu 2019
Datum odevzdání: 31. července 2020
Datum schválení: 31. října 2019

Abstrakt

Táto práca rieši autonómne fungovanie hry Scotland Yard za využitia metód umelých inteligencií pre hranie hier a strojového učenia. Daný problém je úspešne vyriešený pomocou algoritmu pre hranie hier – Alfa-beta. Strojové učenie bolo riešené, ale nebolo úspešné najmä pre veľkú stavovú expanziu a pre nedostatočné možnosti vlastných zdrojov výpočtového výkonu. Riešenie pomocou algoritmu Alfa-beta bolo testované ľudským protihráčom a výsledok testovania ukázal schopnosť AI plnohodnotne konkurovať ľudskému hráčovi. Výsledkom práce je funkčná verzia autonómneho systému, ktorý hrá hru Scotland Yard v zmenšenej hernej ploche. Na základe experimentov so strojovým učením som navrhol niekoľko vylepšení, ktoré by v budúcnosti mohli viesť k funkčnému riešeniu problému strojovým učením.

Abstract

The thesis focuses on creating an autonomous functional system for the game Scotland Yard by using artificial intelligence methods for game theory and machine learning. The problem is solved by algorithm of game theory – Alpha Beta. There was an attempt to use machine learning, but it proved to be unsuccessful due to the large number of states for expansion and insufficient computational recourses. The solution using Alpha Beta algorithm was tested on human players and it proved the ability of artificial intelligence to fully compete against real players. The resulting system is functional, autonomous and capable of playing the game Scotland Yard on simplified game area. Based on these experiments, the thesis also introduces some improvements that could utilize machine learning and extend the existing solution.

Kľúčové slová

strategické hry, stolné hry, strojové učenie, reinforcement learning, hry s neurčitosťou, q-learning, alfa-beta, Scotland Yard

Keywords

strategic games, board games, machine learning, reinforcement learning, games with uncertainity, q-learning, alpha-beta, Scotland Yard

Citácia

TULUŠÁK, Adrián. *Strategická desková hra s neurčitostí.* Brno, 2020. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. Ing. František Zbořil, Ph.D.

Strategická desková hra s neurčitostí

Prehlásenie

Prehlasujem, že som túto bakalársku prácu vypracoval samostatne pod vedením pána doc. Ing. Františka Zbořila, Ph.D. Uviedol som všetky literárne pramene, publikácie a ďalšie zdroje, z ktorých som čerpal.

.....
Adrián Tulušák
27. júla 2020

Obsah

Kapitola 1

Úvod

Už od roku 1956 je vo svete hier a AI veľkou výzvou snaha o vytvorenie umelej inteligencie, ktorá by bola schopná konkurovať ľuďom. V roku 1997 sa to v prípade známeho šachového súboja aj podarilo, keď Kasparov 2/6 hier prehral a 3 remízoval.

Šach ale nie je jediným bodom záujmu vo svete AI. Mnohé počítačové hry využívajú umelé inteligencie pre vytvorenie čo najzaujímavejšieho prostredia pre hráčov. Do sveta počítačových hier sa prerábajú aj viaceré spoločenské kartové alebo stolné hry, vďaka čomu ich môžu hrať hráči aj online alebo len trénovať sami proti virtuálnym protivníkom.

Cieľom tejto práce je navrhnúť a vypracovať umelú inteligenciu pre hru Scotland Yard, ktorá by bola schopná suplovať hráča a rozhodovať o pohybe figúriek. Jedným z možných a veľmi lákavých riešení je použitie strojového učenia, ktoré zabezpečí, že AI si sama nájde najvhodnejšiu cestu k víťazstvu. Táto možnosť však môže byť náročná na spracovanie a aj na vykonanie počítačom. Okrem tejto možnosti pripadá do úvahy aj možnosť navrhnúť AI, ktorá by sa síce neučila sama, ale podľa zadaných pravidiel pre rozhodovanie, by zvládla hrať tak, aby bola ľudskému hráčovi dôstojným súperom.

Hra Scotland Yard patrí rozhodne do kategórie náročných spoločenských stolných hier a vytvoriť umelú inteligenciu, ktorá môže pomôcť hráčom trénovať túto hru alebo si ju len zahrať aj bez spoluhráčov je veľkou výzvou.

V kapitole 2 sú popísané a vysvetlené základné pojmy súvisiace so spoločenskými stolnými hrami. Kapitola je zameraná na pochopenie fungovania spoločenských hier, v ktorých sa hráči striedajú v ťahoch. Toto pochopenie je základom pre ďalšie kapitoly.

V kapitole 3 je rozoberaná umelá inteligencia. Popisuje princípy umelých inteligencií a prístupy k nim. Táto kapitola vysvetľuje metódy využívajúce v AI, ktoré boli použité v tejto práci. Sú tu dôkladne vysvetlené dve rôzne metódy umelých inteligencií. V ďalších kapitolách je objasnené, prečo bolo vhodné vybrať 2 metódy a akým spôsobom sa využili.

Kapitola 4 obsahuje návrh použitia umelej inteligencie, ktorá má byť schopná vyberať ťahy hráčov tak, aby mohla byť rovnocenným oponentom ľudskému hráčovi. Najprv je vysvetlená spoločná časť pre obe metódy, ktorú tvoria najmä pravidlá pre pohyb hráčov vo virtuálnom zjednodušenom prostredí. Následne kapitola objasňuje, akým spôsobom budú jednotlivé metódy riešené a v čom spočívajú možné nedostatky oboch metód.

Kapitola 5 zahŕňa poznatky a postrehy nadobudnuté v priebehu zhotovovania umelej inteligencie pre hru Scotland Yard. Popisuje, prečo nebolo možné použiť umelú inteligenciu, ktorá by sa učila sama naučila, ako najefektívnejšie vyhrať a tiež nevýhody metódy, ktorá učenie nepotrebovala. V závere kapitoly sú zhrnuté návrhy, ktoré by mohli v budúcnosti zlepšiť fungovanie vytvorenej umelej inteligencie.

Kapitola 2

Spoločenské stolné hry

V tejto kapitole sú rozobrané základné princípy spoločenských stolných hier. V súčasnosti existuje veľké množstvo kategórií stolných hier. V tejto kapitole sú objasnené predovšetkým spoločenské stolné hry hrané ťahmi jednotlivých hráčov a hry s veľkým počtom možných ťahov.

V kapitole je tiež bližšie popísané, čím je špecifická hra Scotland Yard, ktorej sa práca v ďalších kapitolách venuje. V podkapitole o hre Scotland Yard sa rieši aj problematika, veľkého počtu možných ťahov. Veľký počet možných ťahov je jedným z najdôležitejších aspektov hry Scotland Yard. Dôležitosť tohto aspektu je možné pripodobniť k hre *Šach*.

Cieľom tejto kapitoly je oboznámiť čitateľa s princípmi stolných hier všeobecne, ale aj so špecifickejšími stolnými hrami, ktoré sa hrajú ťahmi jednotlivých hráčov a s hrou Scotland Yard. Typy hier a informácie o nich sú prevzaté z Encyklopédie svetových hier L. A W. Pijanowských [12] a z webu cosplaybattle.ru [5].

2.1 Ťahové hry

Účelom spoločenských hier je zabaviť skupinu hráčov hraním jednej spoločnej hry. Mnohé spoločenské hry vyžadujú, aby hráči spolupracovali a tým sa zároveň spoznávali. Ťahové hry väčšinou nevyžadujú rýchly úsudok alebo reflexy (niektoré hry majú obmedzený časový limit pre ťah, ale tým sa táto práca nebude veľmi zaoberať). Je však o to dôležitejšie dôkladne zvážiť každý svoj ťah a vymyslieť vhodnú stratégiu, ktorá by mohla viesť k víťazstvu.

Základné princípy ťahových hier

Základným prvkom ťahových hier je ťah. Jeden ťah vo väčšine hier znamená vykonanie jednej akcie (napr. vyloženie jednej kary, posunutie figúrkou a pod.), ale v konkrétnych hrách sa môže mierne líšiť – v pravidlách hry môže byť ťah definovaný aj iným spôsobom. Napríklad môže hráč v rámci jedného ťahu vykonať 2 akcie (napr. *Mars: Teraformácia*). Alebo môže hráč hrať svoj ťah dovtedy, pokým neminie všetky možnosti, ktoré má, alebo kým nevykoná špecifickú akciu, ktorá znamená ukončenie ťahu (napr. *BANG!* – vyloženie karty *BANG!*)

Keď hráč ukončí svoj ťah, nasleduje ďalší hráč vykonaním svojho vlastného ťahu, ktorý môže uskutočniť podľa pravidiel danej hry. Ak pravidlá neuvádzajú inak, jeden po druhom vykonajú svoj ťah všetci hráči. Vykonanie ťahu všetkými hráčmi sa štandardne nazýva kolo. Keď to pravidlá a tematika hry umožňujú, môže mať kolo aj iný názov.

Ukončenie kola alebo priebeh celého kola vzhľadom na konkrétneho hráča (t.j. odkedy bol hráč naposledy na ťahu, sa vystriedali už všetci hráči a je opäť na ťahu daný hráč) býva zvyčajne dôležitým momentom. Hráči v tejto chvíli môžu zhodnotiť aktuálny priebeh hry a prehodnotiť svoju stratégiu vzhľadom na akcie, ktoré vykonali ostatní hráči v uplynulom kole.

Hry so zložitejšou štruktúrou ťahov

Väčšina hier, predovšetkým staršie dobre známe hry (napr. *Žolík, Človeče, nehnevaj sa...*), majú jednoduchú štruktúru, ktorej najvyšším dielom je kolo. Napríklad v hre *Človeče, nehnevaj sa* sa hrá rovnakým spôsobom počas celej hry – všetky kolá sú si rovnocenné. Bolo by možné oponovať začiatkom hry, kde niektorí hráči hádžu kockou na začiatku hry 3-krát namiesto 1-krát, aby sa hra rýchlejšie rozbehla. Ale pri takto notoricky známych hrách sú pravidlá väčšinou predmetom dohody hráčov. A aj v tomto prípade sa hra po štartovacích kolách už vyvíja spomínaným spôsobom. Nič viac sa v štruktúre hry už nemení, kolá zostávajú najvyšším dielom hry a sú si všetky rovnocenné.

Dôležitou súčasťou tejto práce sú hry, ktorých jedno kolo nie je tým najvyšším prvkom štruktúry hry. Príkladom takejto hry je hra *Mars: Teraformácia* [12], v ktorej kolá tvoria len nižšiu časť štruktúry a samé patria do generácií (pomenovanie tohto prvku štruktúry je tematické). V jednotlivých generáciach hráči odohrajú potrebný počet kôl, v každom kole majú hráči svoje ťahy. Na konci generácie sa vyhodnocuje doterajší priebeh hry, hráči dostávajú odmeny a pod.

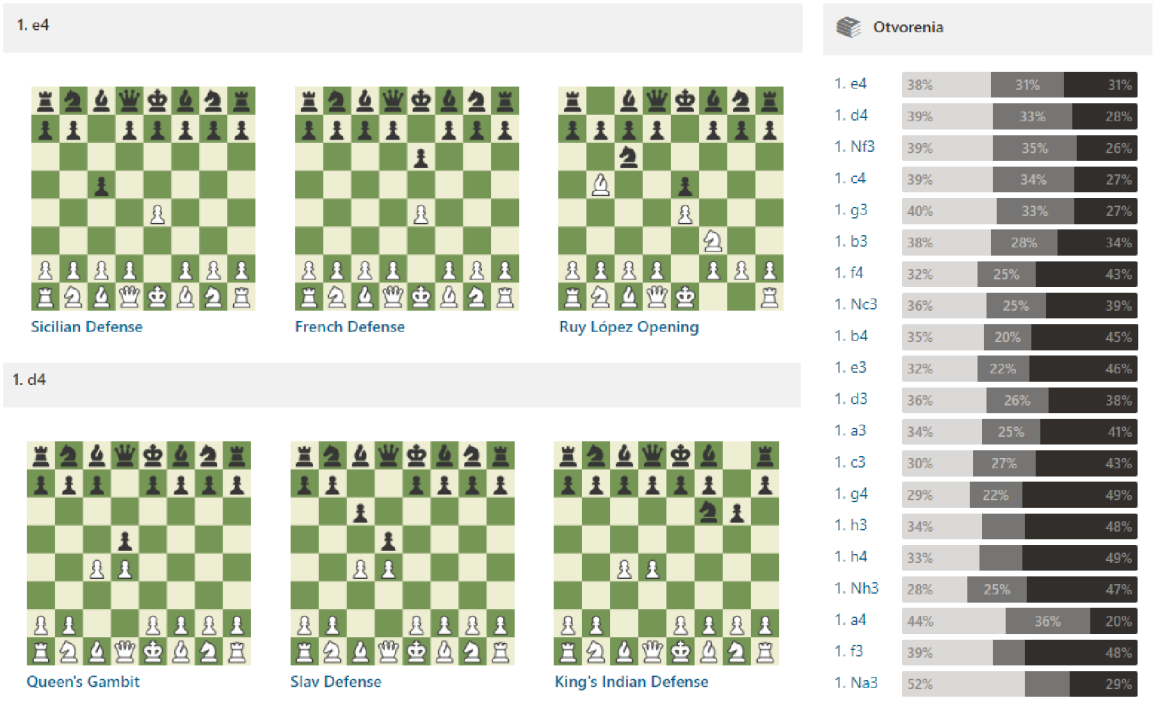
Význam práve takéhoto typu hier je objasnený v kapitole 2.3.

2.2 Hry s veľkým počtom možných ťahov

Najznámejšou hrou, ktorá je významná veľkým počtom možných ťahov je Šach. Pokým sa hráč nedostane do situácie, kedy mu zostal kráľ a jedna figúrka, s ktorými už len uniká pred súperom, tak má vždy veľké množstvo možností, kam potiahnuť ktoroukoľvek figúrkou. A je veľmi dôležité všetky tieto možnosti zvážiť.

Ale rovnako ako hráč na ťahu, aj protihráč má svoje možnosti. A hráč, ktorý zvažuje, aký ťah by mal urobiť, musí zahrnúť do svojich plánov aj ťah protihráča. Týmto sa náročnosť výberu ťahu ešte znásobuje.

Každý ťah, ktorý hráč vykoná, nech by bol v danom momente akokoľvek dobrý, môže nakoniec viesť práve k prehre. Kvôli vysokej náročnosti existujú mnohé teoretické knihy a články o najefektívnejších ťahoch. Príklad takto zhrnutých a vyhodnotených možností je na obrázku 2.1.



Obr. 2.1: Zhodnotenie niekoľkých možných začiatočných ťahov a najpravdepodobnejších reakcií protihráča. V časti *Otvorenia* je vyhodnotená percentuálna výhernosť týchto začiatočných ťahov. Zdroj: https://www.chess.com/sk/openings [3]

Človek vs. AI[1]

Pri hre, akou je Šach, je veľmi dôležité premýšľať dopredu. Hráč musí mať premyslené, čo by protihráč mohol urobiť o niekoľko ťahov. Takýmto spôsobom prehodnotiť všetky možné vlastné ťahy a následne všetky ťahy súpera na vzdialenosť niekoľkých kôl, je pre väčšinu ľudí priveľmi náročné.

Práve pre svoju vysokú náročnosť na výpočty a dôležitosť stratégie je Šach skvelou príležitosťou pre svet IT[2]. Už od roku 1956 prebiehajú pokusy o vytvorenie počítačového súpera, ktorý by bol schopný poraziť človeka [6]. V roku 1988 vyhral prístroj zvaný *HiTech* šachový šampionát v Pennsylvánii. Neskôr, v roku 1996, prebehol známejší zápas – Kasparov vs. Deep Blue. V tomto zápase Kasparov prvú hru prehral, ale počas 6 za sebou idúcich hier sa mu podarilo 3 hry vyhrať a 2 ukončiť remízou [2]. O rok neskôr v odvetnej hre Kasparov prehral zápas tiež tvorený 6 za sebou idúcimi hrami [8]. Podobné súboje prebiehajú dodnes.

Je však dôležité spomenúť, že umelé inteligencie majú pri takto náročných hrách oveľa väčší význam. Nie len pri Šachu sa využívajú umelé inteligencie na trénovanie. Hráči tak môžu skúšať rôzne techniky hrania a vidia ich efektivitu a úspešnosť aj bez toho, aby museli čakať na čas premýšľania protihráča. V špecifických prípadoch môžu vrátiť ťah späť a vyskúšať, ako by sa protihráč zachoval v inom prípade.

[1] AI – anglicky Artificial intelligence – umelá inteligencia
[2] IT – anglicky Information technology – informačné technológie

2.3 Hra Scotland Yard

Významnou hrou medzi spoločenskými stolnými hrami, pri ktorých je potrebné vybrať si jeden z množstva dostupných ťahov, je Scotland Yard. V tejto časti je stručne zhrnutá hlavná myšlienka hry a základné pravidlá. Kompletné pravidlá sú v prílohe A.

Scotland Yard – prehľad

Scotland Yard je spoločenská hra pre 6 hráčov (sú aj varianty pre menej hráčov), rozdelených do 2 tímov – 5 hráčov proti jednému. Každý z 5 hráčov prvého tímu berie na seba úlohu detektíva/agenta. Šiesty hráč hrá za zlodeja, zvaného v tejto hre Mr. X. Hráči sa pohybujú po 200 políčkach rozmiestnených po hracej ploche a prepojených rôznymi spôsobmi.

Hráči sa nepohybujú samostatne, ale ako tímy. Teda keď sú na ťahu detektívi, dohodnú sa, ktorý z nich sa kam posunie a ťah vykonajú naraz. Mr. X tvorí tím samostatne, teda sa pohybuje len podľa vlastného úsudku.

Pohyb po hracej ploche

Hráči môžu využívať 3 základné typy presunu a jeden výnimočný, pričom každý typ presunu má z každého políčka presne určené, kam sa ním môže hráč dostať (nemôže sa presúvať ľubovoľne v určenom smere, ako napríklad kráľovná v Šachu, ale len na vybrané destinácie):

- Taxi – tvorí najviac ciest z každého políčka, umožňuje presun na políčka v tesnej blízkosti
- Autobus – presúva hráča vo väčšej vzdialenosti ako Taxi, možnosť cestovania nie je taká častá ako pri Taxi
- Metro – presúva hráča na značnú vzdialenosť, ale je využiteľné na najmenšom počte políčok
- Trajekt – špeciálny presun na značnú vzdialenosť, ktorý môže použiť len Mr. X, tvorí len 3 prepojenia medzi 4 políčkami

Každý z detektívov má dopredu určený počet *cestovných lístkov* na každý z dopravných prostriedkov. Čiže detektívi majú určený maximálny počet použití každého spôsobu dopravy. Ich úlohou je teda pohybovať sa takým spôsobom, aby využili *cestovné lístky* tak, aby neuviazli, keď už niektorý zo spôsobov nebudú môcť použiť. Musia myslieť na to, aby mali k dispozícii potrebný spôsob presunu pre chytenie Mr. X. Na samotného Mr. X sa toto obmedzenie nevzťahuje. Mr. X sa môže pohybovať každým spôsobom neobmedzene veľakrát. Jedinou výnimkou je pohyb trajektom. Ten môže použiť len Mr. X a aj to len v obmedzenom počte.

Pohyb Mr. X

Mr. X sa môže pohybovať bez obmedzenia používania jednotlivých spôsobov dopravy. Nakoľko hrá proti nemu ďalších 5 hráčov, aby bola hra vyrovnaná, Mr. X je zvýhodnený inými spôsobmi.

Nikto nepozná presnú polohu Mr. X. Keď sa Mr. X presunie, oznámi ostatným hráčom len spôsob presunu. Mr. X svoju aktuálnu polohu oznámi väčšinou raz za 5 ťahov. Počas

týchto 5 ťahov sa detektívi presúvajú a snažia sa chytiť Mr. X bez toho, aby vedeli, kde presne sa nachádza.

Práve táto skutočnosť je najdôležitejším aspektom hry. Dodáva hre nielen na zaujímavosti, ale vzniká v hre neurčitosť. Detektívi tak musia odhadovať a predpokladať, kam by sa s najväčšou pravdepodobnosťou presunul. Detektívi hrajú síce každý za svoju figúrku, ale musia spolupracovať. Sú nútení premýšľať nahlas, aby sa vedeli dohodnúť na najlepšom pohybe každého z nich. Zároveň ale predkladajú svoje plány pohybu zlodejovi – Mr. X.

Stratégia hráčov

Herný plán je rozdelený do 4 hlavných častí. Tieto časti sú oddelené väčšinou mostom alebo nejakým úzkym prechodom. Tieto prechody medzi časťami mesta sú dôležitými strategickými bodmi. Ak sa detektívom podarí zaistiť, aby Mr. X neunikol z oblasti, v ktorej sa práve nachádza, rapídne zvyšujú svoju šancu na výhru.

Ďalšou dôležitou skutočnosťou je, že Mr. X sa na začiatku hry neukáže. Svoju polohu prezradí až po 3. ťahu. Detektívi sa teda na začiatku hry pohybujú úplne naslepo. Musia si do 3 ťahov čo najefektívnejšie rozdeliť priestor, aby mali zaistené všetky oblasti mesta a musia byť pripravení na prvé prezradenie polohy Mr. X. Nie vždy je to možné, ale najvýhodnejšie je presunúť sa do blízkosti metra, ktoré umožní detektívom rýchlo sa presunúť čo najbližšie k Mr. X.

Mr. X musí detektívom čo najviac znemožniť chytiť ho. Musí teda vyhľadávať polohy, ktoré poskytujú čo najväčší počet miest na útek. To spôsobí, že detektívi budú mať viac možností, kam sa Mr. X mohol presunúť a bude pre nich náročnejšie nájsť ho.

Kapitola 3

Umelá inteligencia a metódy hrania hier

V dnešnej dobe je umelá inteligencia využívaná v našej bezprostrednej blízkosti a často o tom ani nevieme. Názory na definíciu umelej inteligencie sa líšia. Spolu s tým sa líšia aj názory na to, akým spôsobom umelú inteligenciu testovať a preukázať tým jej skutočnú inteligenciu. V roku 1967 definoval Marvin Minsky umelú inteligenciu ako vedu o vytváraní strojov alebo systémov, ktoré budú pri riešení určitej úlohy používať také postupy, ktoré – keby ju riešil človek – by sme považovali za prejav jeho inteligencie. V tejto kapitole budú ozrejmené najznámejšie definície umelej inteligencie a tiež jej najznámejšie testy.

Umelá inteligencia sa vo veľkom množstve využíva v hernej oblasti. Herné štúdiá síce nie vždy siahajú po možnosti zakomponovania AI[1] do svojich hier, no aj napriek tomu je táto oblasť pomerne rozvinutá. Ak sa tvorcovia rozhodnú nevyužiť AI, tak je to väčšinou z dôvodu nedostatku času alebo prostriedkov. Hráči v dnešnej dobe vyžadujú viac prepracované grafické spracovanie a príjemnú hrateľnosť než takmer ľudské správanie počítačových protihráčov. Hra proti ľudským hráčom tiež býva väčšou výzvou. Hráči sa tak musia vysporiadať s mnohými nepredvídateľnými okolnosťami, ktoré by sa len ťažko napodobňovali umelou inteligenciou. Nasledujúce časti popisujú významné prístupy a algoritmy umelej inteligencie súvisiace s hrami, ktoré sú dôležité pre túto prácu.

Cieľom tejto kapitoly je oboznámiť čitateľa s umelou inteligenciou všeobecne a predovšetkým s jej využitím v hrách. Táto práca sa zameriava na ťahové hry, preto tu budú ozrejmené predovšetkým tie časti AI, ktorými je možné takéto hry riešiť. Údaje v nasledujúcich kapitolách boli čerpané predovšetkým z publikácie S. Russella a P. Norviga [13], práce V. Neřáda [16] a P. Jurča [9].

3.1 Základné princípy a prístupy k umelej inteligencii

Vo filozofii sa ohľadom inteligencie kládli nasledovné otázky [14]:

- Môžu byť formálne pravidlá použité na vyvodenie správnych záverov?
- Ako vzniká vo fyzickom mozgu vzniká myslenie?
- Kde vznikajú vedomosti?
- Ako vedomosti môžu viesť k akcii?

[1] AI – anglicky Artificial intelligence – umelá inteligencia

Tieto otázky do veľkej miery ovplyvňujú vývoj AI. Keďže hlavným cieľom je vytvoriť inteligenciu schopnú konkurovať ľudskej, samotná ľudská inteligencia musí byť predtým preskúmaná. Na základe postoja k tomu, čo určuje správnosť AI, sa podľa Russela a Norviga [14] prístupy k AI rozdeľujú do 4 skupín podľa zamerania na:

- Ľudské správanie
- Ľudské zmýšľanie
- Rozumné správanie
- Rozumné zmýšľanie

Popisy jednotlivých kategórií sú tiež prevzaté od Russela a Norviga [14].

Ľudské správanie

Tento prístup sa nazýva aj The Turing Test approach – prístup Turingovho testu. Turingov test bol navrhnutý, aby poskytol uspokojivú funkčnú definíciu inteligencie. Počítač (alebo robot) prešiel testom vtedy, ak ľudský rozhodca po položení niekoľkých otázok nedokáže určiť, či odpovede dostal od človeka alebo počítača. Pri takomto chápaní inteligencie by počítač s umelou inteligenciou musel disponovať nasledujúcimi schopnosťami:

- Spracovanie prirodzeného jazyka – pre umožnenie úspešnej komunikácie.
- Reprezentácia vedomostí – pre uloženie vedomostí ktoré má alebo počuje.
- Automatizované uvažovanie – pre využitie uložených informácií v odpovedi na otázku a na vyvodenie nových záverov.
- Strojové učenie – pre adaptáciu na nové okolnosti, detekciu a vyhodnotenie vzorov.

Ľudské zmýšľanie

Anglicky tiež The cognitive modeling approach – prístup kognitívneho formovania. Na to, aby bolo možné povedať, že počítač zmýšľa ako človek, je najprv nutné vedieť, ako vlastne človek zmýšľa. Je potrebné preskúmať spôsob ľudského myslenia. To je možné urobiť 3 cestami [14]:

- Introspekciou – psychologická metóda založená na pozorovaní vlastného vnútorného duševného stavu, teda stavu vlastnej mysle. Pozorovateľ sa snaží zachytiť vlastné myšlienky tak, ako idú jedna za druhou.
- Psychologickými experimentmi – pozorovanie osoby pri vykonávaní nejakej činnosti.
- Zobrazovaním mozgu – pozorovanie mozgu pri vykonávaní nejakej činnosti.

Až keď je známe správanie ľudskej mysle, až vtedy je možné vyjadriť túto teóriu ako počítačový program. Ak sa zhoduje vstup a výstup počítačového programu so vstupom a následnou reakciou človeka, znamená to, že program (alebo jeho časť) môže obdobným spôsobom fungovať aj v ľudskej mysli. Zároveň to znamená, že takáto umelá inteligencia je podľa daných požiadaviek vyhovujúca a dá sa povedať, že zmýšľa ľudsky.

Rozumné zmýšľanie

V angličtine nazývaný The "laws of thought" approach – prístup zákonov myslenia. Základ tohto prístupu je založený na gréckom filozofovi Aristotelovi [14]. Ako prvý sa pokúsil kodifikovať tzv. *správne myslenie* ako nezvratný proces odôvodňovania. Aristoteles založil formu deduktívneho dôkazu nazývanú **sylogizmus**. Sylogizmus poskytuje vzor, ktorý vždy vráti pravdivý záver, ak dostal pravdivé premisy. Tieto zákony myslenia a ich štúdium podporili vznik oblasti zvanej logika.

Logici už v 19. storočí vyvinuli precízny zápis pre vyjadrenia všetkých druhov objektov vo svete a vzťahy medzi nimi. Okolo roku 1965 už existovali programy, ktoré by mohli vyriešiť akýkoľvek problém popísateľný logickým zápisom. Avšak zapísať neformálne vedomosti, ktoré ani nemusia byť absolútne presné, je náročné zapísať pomocou formálnych požiadaviek, ktoré sú potrebné pre logický zápis. Ďalším významným problémom tohto prístupu spočíva v tom, že je rozdiel v riešení **princípu** problému a riešením problému **v praxi**.

Rozumné správanie

The rational agent approach – prístup rozumného agenta. Agent je tu chápaný ako niečo, čo vykonáva nejakú činnosť (z Latinčiny *agere*, čo znamená *robiť*). Keďže ale nehľadáme „nejaké" správanie, ale v tomto prípade správanie, ktoré by sa dalo považovať za rozumné, od agenta sa očakáva viac. Mal by zvládať pracovať samostatne, vnímať svoje okolie, byť schopný prispôsobiť sa zmenám a predovšetkým snažiť sa dosiahnuť daný cieľ.

Správanie rozumného agenta je všeobecnejšie než ostatné zmienené prístupy. Niekedy je potrebné zahrnúť ich do správania agenta, aby sa naozaj mohol správať rozumne. Napríklad *prístup zákonov myslenia* je založený na utváraní správnych záverov. Ak sa má agent považovať za rozumne konajúceho, musí logicky vyvodiť záver, že vykonaná akcia bude smerovať k dosiahnutiu daného cieľa. Následne by mal na základe tohto záveru konať.

Schopnosti potrebné pre zvládnutie Turingovho Testu by tiež agentovi napomohli správať sa rozumne. Schopnosť reprezentácie a rozpoznania reči, či odôvodňovanie na základe nadobudnutých znalostí, umožňuje agentovi robiť dobré rozhodnutia. Schopnosť učiť sa je potrebná na zlepšenie možnosti vytvárania efektívneho správania.

Prístup rozumného agenta má tiež výhodu v tom, že je vedeckému vývoju prístupnejší než prístupy založené na ľudskom správaní alebo myslení. Štandard rozumnosti je matematicky dobre definovaný a všeobecný. Môže byť použitý na vytvorenie agentov, ktorí preukázateľne dosiahnu zadaný cieľ.

3.2 Metódy hrania hier

Pri hraní ťahových hier[2] je najdôležitejšou častou hry výber a uskutočnenie najlepšieho možného ťahu. V prípade jednoduchého ťahu, ktorý je tvorený napríklad presunom figúrky na iné políčko, je možné využiť pre tento výber niektorú z metód hrania hier. Nasledujúce časti kapitoly vychádzajú okrem už spomínaných zdrojov aj zo študijnej opory F. V. Zbořila a F. Zbořila pre predmet IZU [18].

Pre zjednodušenie popisu metód hrania hier je možné uvažovať len hry s 2 hráčmi, ktorí sa pravidelne striedajú vo svojich ťahoch. Obaja hrajú poctivo a snažia sa zvíťaziť. Cieľom je nájsť najvhodnejší ťah pre hráča, ktorý je práve na ťahu – hráč na ťahu bude ďalej nazývaný *hráč A*. Hráč, ktorý nie je na ťahu bude nazývaný *hráč B*. Hľadanie najvýhodnej-

[2] ťahové hry - vysvetlené v kapitole 2.1

šieho ťahu prebieha prehľadávaním stavového priestoru. A to tak, že sa preskúmajú všetky ťahy hráča A a ak nie sú konečné (výherné), je potrebné z týchto ťahov preskúmať možné ťahy hráča B. Týmto spôsobom sa prehľadávanie zanoruje buď pokým sa nenájde výherná možnosť alebo po vopred danú hranicu zanorenia.

Hľadanie ťahu pre hráča A je potrebné riešiť ako *OR problém*. Teda k výhre musí viesť **aspoň jeden** z možných ťahov. Naopak vyhodnocovanie ťahu hráča B je *AND problém*, pretože hráčovi A musia vyhovovať **všetky** možné ťahy hráča B, nakoľko nevie, ktorý z ťahov si on vyberie.

Metódy riešenia zložitejších hier

Rôzne náročné hry si vyžadujú výber vhodnej metódy na ich riešenie. Jednoduchšie hry – také, pri ktorých je možné v reálnom čase vyhodnotiť celý stavový priestor – je možné riešiť AND/OR grafmi, resp. algoritmom AO[3]. Hry, ktorými sa zaoberá táto práca, sa dajú v tomto zmysle považovať za zložitejšie. Je v nich toľko možností, že nie je možné počítať a vyhodnocovať všetky ťahy, ktoré by hráči mohli vykonať. Aspoň nie v reálnom čase. Pre takéto hry sa používajú algoritmy, ktoré preskúmajú len istý počet ťahov od aktuálneho stavu. Určitým spôsobom ohodnotia, ako výhodné sú pozície dosiahnuté v danom počte ťahov a vrátia cestu k najlepšie hodnotenému ťahu.

K takýmto algoritmom patria napríklad MiniMax alebo Alfa-beta rezy (anglicky Alpha-beta pruning). Využitie týchto algoritmov je možné napríklad v hrách ako Šach, Dáma a pod. V prípade, že algoritmus dosiahne maximálnu hĺbku zanorenia (ktorá je dopredu daná) a nebol nájdený koncový stav (výhra/prehra/remíza), algoritmus musí daný stav nejakým spôsobom ohodnotiť.

Pri spomínaných algoritmoch sa na ohodnotenie stavu používajú celočíselné hodnotiace funkcie. Väčšie (kladné) hodnoty znamenajú priaznivé ohodnotenie pre hráča A a zároveň nepriaznivé pre hráča B. Menšie hodnoty sú naopak nepriaznivé pre hráča A, ale priaznivé pre hráča B. Výhra alebo prehra sú reprezentované maximálnou, resp. minimálnou hodnotou uvažovaného číselného intervalu.

3.3 Alfa-beta prerezávanie

Alpha-beta pruning – Alfa-beta prerezávanie je oproti algoritmu MiniMax výhodnejšie v tom, že ak dôjde k odhaleniu nevýhodného ťahu, v danej oblasti už algoritmus nepokračuje, ale vráti sa späť a prehľadáva ostatné cesty. Táto schopnosť môže ušetriť množstvo výpočtového času.

[3] AO – skratka pre AND/OR

Procedúra Alfa-beta:

1. Uzlom X je nazvaný obdržaný vstupný uzol.
2. Ak je uzol X počiatočným (koreňovým) uzlom, nastav $\alpha = -\infty$, $\beta = \infty$ (v praxi sa používa maximálna, resp. minimálna možná hodnota).
3. Ak je uzol X listom (končeným stavom hry alebo uzlom v maximálnej hĺbke prerezávania), ukonči procedúru a vráť ohodnotenie tohto uzlu.
4. Ak je na ťahu hráč A (uzol je typu OR):
   - 4.1. Pokým je $\alpha < \beta$, tak postupne pre prvý, resp. ďalší ťah (bezprostredného nasledovníka uzlu X) volaj procedúru Alfa-beta pre hráča B s aktuálnymi hodnotami premenných α a β. Po každom vyšetrenom ťahu nastav hodnotu premennej α na maximum z aktuálnej a vrátenej hodnoty.
   - 4.2. Ukonči procedúru, vráť aktuálnu hodnotu premennej α a pre koreňový uzol vráť aj ťah, ktorý vedie k najlepšie ohodnotenému bezprostrednému nasledovníkovi.
5. Ak je na ťahu hráč B (uzol je typu AND):
   - 5.1. Pokým je $\alpha < \beta$, tak postupne pre prvý, resp. ďalší ťah (bezprostredného nasledovníka uzlu X) volaj procedúru Alfa-beta pre hráča A s aktuálnymi hodnotami premenných α a β. Po každom vyšetrenom ťahu nastav hodnotu premennej β na minimum z aktuálnej a vrátenej hodnoty.
   - 5.2. Ukonči procedúru a vráť aktuálnu hodnotu premennej β.

Ukážka Alfa-beta prerezávania

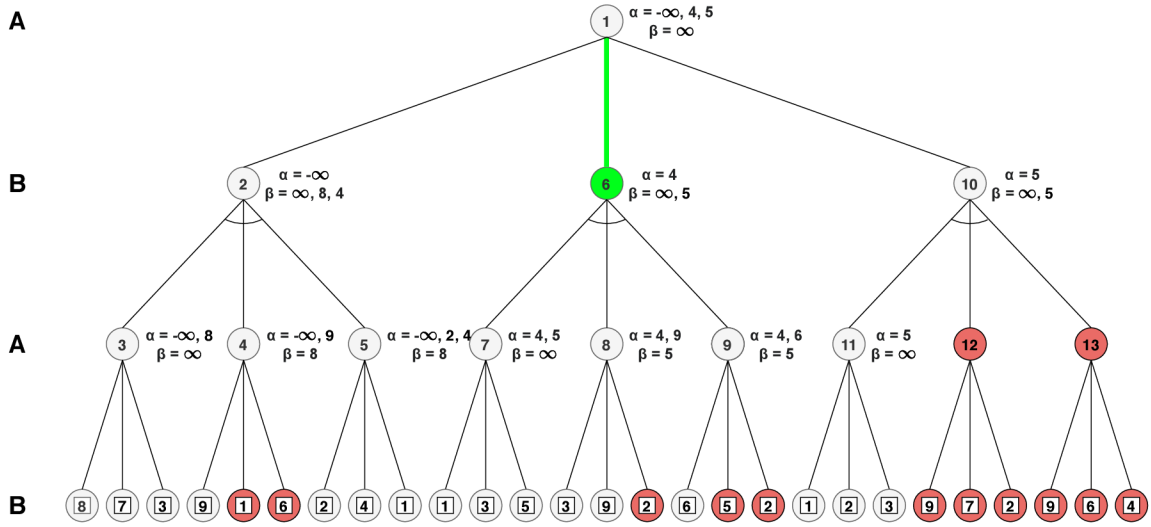
Na obrázku 3.1 je znázornený princíp alfa a beta rezov pri prehľadávaní stavového priestoru. Pre lepšiu identifikáciu pri popisovaní priebehu algoritmu, sú uzly označené číslami 1-13. Hodnoty v štvorčekoch znamenajú ohodnotenie uzlu hodnotiacou funkciou.

Po zavolaní procedúry Alfa-beta na počiatočný uzol (**uzol 1**) sa nastavia hodnoty $\alpha = -\infty$, $\beta = \infty$. Hráč A volá z uzlu 1 procedúru Alfa-Beta pre hráča B na svoj prvý ťah – na uzol 2 – s hodnotami $\alpha = -\infty$, $\beta = \infty$. Hráč B následne volá procedúru Alfa-beta pre hráča A z uzlu 2 na uzol 3 tiež s hodnotami $\alpha = -\infty$, $\beta = \infty$. Hráč A postupne volá procedúru Alfa-beta pre hráča B na všetky svoje možné ťahy. Tu sú všetky volané uzly listovými, procedúra vracia ich ohodnotenie. Hráč A po každom vrátenom ohodnotení listového uzlu nastaví hodnotu α na maximum z vrátenej a aktuálnej hodnoty. V premennej α pribudne hodnota 8. Po každom vyhodnotení vrátenej hodnoty sa skontroluje, či platí $\alpha < \beta$. Po vyhodnotení listových uzlov vráti uzol 3 hodnotu 8.

Hráč B (uzol 2) obdrží od procedúry vrátenú hodnotu 8 a nastaví minimum z aktuálnej a vrátenej hodnoty. Uzol 2 má teraz hodnoty $\alpha = -\infty$, $\beta = 8$. Hráč B volá procedúru Alfa-beta pre hráča A na uzol 4 s hodnotami $\alpha = -\infty$, $\beta = 8$. Hráč A (uzol 4) volá procedúru Alfa-beta pre hráča B na svoj prvý ťah. Dostáva vrátenú hodnotu 9. Vyhodnotí maximum pre novú hodnotu α. Uzol 4 má teraz hodnoty $\alpha = 9$, $\beta = 8$. Neplatí podmienka $\alpha < \beta$ a vyšetrovanie uzlu 4 je ukončené. Uzol 4 vracia hodnotu 9. Uzol 2 obdrží vrátenú hodnotu 9, ktorá však nič nemení. V uzle 2 zostávajú hodnoty $\alpha = -\infty$, $\beta = 8$.

Hráč B (uzol 2) potom volá procedúru Alfa-beta pre hráča A na svoj posledný ťah – na uzol 5 – s hodnotami $\alpha = -\infty$, $\beta = 8$. Hráč A (uzol 5) volá postupne procedúru Alfa-beta

na všetky svoje možné ťahy a postupne upravuje hodnotu premennej $\alpha = -\infty, 2, 4$. Stále musí platiť nerovnosť $\alpha < \beta$. Po vyhodnotení všetkých ťahov vracia uzol 5 hodnotu 4 uzlu 2. Hráč B (uzol 2) vyhodnotí minimum z aktuálnej ($\beta = 8$) a vrátenej hodnoty (4) a prepíše hodnotu $\beta = 4$. Uzol 2 teraz vráti uzlu 1 hodnotu 4. Uzol 1 vyhodnotí maximum z aktuálnej a vrátenej hodnoty a aktualizuje hodnotu $\alpha = 4$. Hráč A (uzol 1) volá procedúru Alfa-beta pre hráča B s aktuálnymi parametrami na uzol 6. Popísaný postup sa uplatňuje postupne na všetky nasledovné uzly.



Obr. 3.1: Znázornenie prechodu algoritmu Alfa-beta stavovým priestorom.

3.4 Reinforcement learning

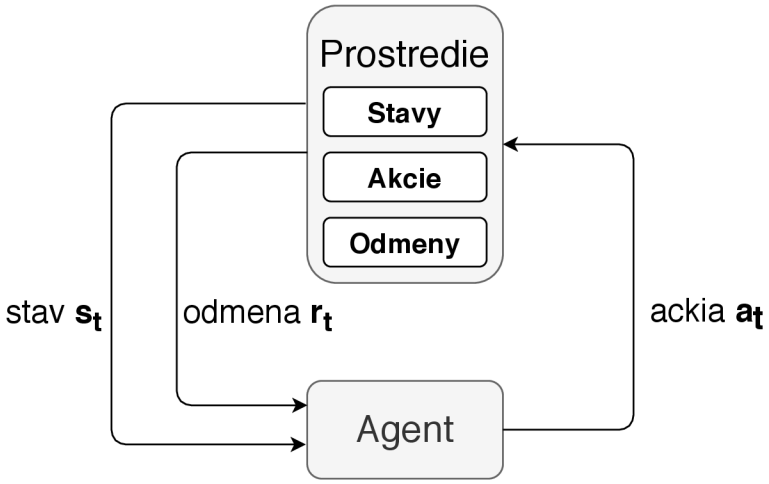
Pri písaní nasledujúcich 2 podkapitol som okrem už uvedených zdrojov čerpal z publikácie S. Russela a P. Norviga [15], z dizertačnej práce C. Watkinsa [17] a práce M. Coggan [4].

Existujú 3 základné typy strojového učenia. Učenie s učiteľom (anglicky supervised learning), učenie bez učiteľa (anglicky unsupervised learning) a učenie zo skúsenosti (česky posilované učení, anglicky reinforcement learning). Posledné z nich – učenie zo skúsenosti – nemá v Slovenčine ustálený preklad, preto budem v tejto práci používať anglický výraz, teda **reinforcement learning**.

Učenie zo skúsenosti je pre ľudí najprirodzenejším spôsobom učenia. Už od detstva sa učíme, ako sa poučiť zo skúseností a ako nerobiť znovu tie isté chyby. Rodičia môžu dieťaťu donekonečna opakovať, že oheň páli, ale dieťa sa bude snažiť dosiahnuť oheň, až kým si to nevyskúša a nezistí, že on nielen páli, ale popálenie aj bolí. Je teda potrebná interakcia s prostredím a možnosť zapamätať si výsledok z danej akcie. A práve týmto spôsobom pracuje reinforcement learning.

Model reinforcement learningu

Na obrázku 3.2 je zobrazený základný model reinforcement learningu. Agent v čase t reaguje na stav prostredia s_t, v závislosti od ktorého vykoná akciu a_t. Vykonaním akcie a_t agent zmení stav prostredia. Táto zmena prostredia je distribuovaná agentovi novým stavom s_t a o hodnote prechodu medzi stavmi je agent oboznámený odmenou r_t [10].



Obr. 3.2: Model reinforcement learningu

Reinforcement learning sa nachádza niekde medzi učením s učiteľom a učením bez učiteľa. Pri RL (reinforcement learning) je vhodné spomenúť, že *greeedy*[4] metódy nemusia vždy fungovať. Sú veci, ktoré je jednoduché urobiť, ak hneď v danom okamihu agent vie, či to bolo správne alebo nie. Ak je ale problém zložitejší a je potrebné brať do úvahy dlhodobé dôsledky, greedy metóda nebude stačiť. Bude nutné agenta trénovať so zámerom maximalizovať dlhodobé odmeny. Druhá vec, ktorú je nutné tu spomenúť je, že v niektorých problémoch závisí na postupnosti prechádzania stavov. Nestačí sa len v každom stave pozrieť, či ide správnym smerom k cieľu. Odmena v takomto prípade závisí od celej histórie agentových prejdených stavov. Tu je na rozdiel od učenia s učiteľom a bez učiteľa dôležitý čas.

Objavovanie vs. využívanie

Pri reinforcement learningu je dôležité vysporiadať sa s problémami, ktoré sa pri ostatných typoch učenia nevyskytujú. Jedným z nich je rozhodovanie, kedy **objavovať** nové akcie a kedy **využívať** už preskúmané (exploration vs. exploitation trade-off).

Agent sa má snažiť dosiahnuť cieľ čo najefektívnejšie. Mal by teda vyberať také akcie, ktoré sú podľa jeho nadobudnutých vedomostí najlepšie. To znamená, že by mal využívať naučené informácie. Môže sa však stať, že naučená cesta ešte stále nebude tou najlepšou možnou. Podstatnou otázkou je, ako a kedy sa rozhodnúť, kedy objavovať a kedy využívať. Bez objavovania by agent vždy použil prvú cestu vedúcu k cieľu a nikdy by nenašiel žiadnu inú – aj keby bola lepšia.

Jeden z najjednoduchších spôsobov vyriešenia tohto problému sú tzv. ϵ-*greedy* metódy. Agent vyberá *greedy* akcie s pravdepodobnosťou $1 - \epsilon$ (ϵ je v rozsahu 0-1). Akcie *nongreedy* sú vyberané s pravdepodobnosťou ϵ. Ak je agent v statickom prostredí, je možné v priebehu učenia hodnotu ϵ postupne znižovať, aby agent viac využíval naučené akcie a objavoval čím ďalej, tým menej.

Ďalšou z možností je využitie faktu, že počiatočná hodnota Q-funkcie nemá vplyv na konvergenciu. Môžeme nastaviť počiatočnú hodnotu tak, aby bolo možné rozlíšiť, ktorá

[4]greedy metódy – doslovný preklad: žravé metódy. Sú to metódy, ktoré vyberajú v danom okamihu najlepšie hodnotenú možnosť.

akcia už bola vykonaná. Je teda možné nastaviť hodnotu Q-funkcie na maximálnu možnú hodnotu a používať greedy stratégiu. Agent si tak vždy vyberie najprv akcie, ktoré ešte nepreskúmal. Po preskúmaní sa každá z hodnôt pre danú akciu zníži (na reálnu hodnotu, ktorou je daná akcia ohodnotená). Týmto spôsobom bude agent pokračovať, pokým si nepreskúma všetky akcie a následne bude vyberať najlepšiu cestu z naučených hodnôt.

3.5 Q-learning

Q-learning bol predstavený C. Watkinsom v roku 1989 v jeho dizertačnej práci [17]. Q-learning je algoritmus učiaci sa pomocou funkcie $Q(s, a)$, ktorá je popísaná rovnicou 3.1. Q-learning nevyžaduje presný matematický model okolia a jeho cieľom je nájsť optimálnu rozhodovaciu politiku agenta. Inými slovami, naučiť agenta rozhodovať sa, akú cestu za daných podmienok zvoliť.

Učenie Q-learningom zahŕňa agenta, množinu stavov S a množinu akcií A, ktoré sú vykonateľné z tohto stavu. Vykonaním akcie a z množiny stavov A ($a \in A$) sa agent presunie do nového stavu. V závislosti od riešeného problému je mu za tento prechod poskytnutá odmena r (numerická hodnota; čím vyššia, tým výhodnejšia). Cieľom agenta je maximalizovať získanú odmenu. Agent si môže získané hodnoty pamätať v tabuľke (tzv. Q-table, čo označuje tabuľku Q-hodnôt) alebo učením neurónovej siete.

Q-hodnoty sa na začiatku inicializujú ľubovoľnými hodnotami. Ako agent prechádza prostredím a získava rôzne odmeny využitím rôznych akcií, aktualizuje Q-hodnoty. Aktualizáciu Q-hodnôt popisuje nasledovná rovnica (zdroj [10]):

$$Q(stav, akcia) \leftarrow (1 - a)Q(stav, akcia) + a(odmena + \gamma \max_{a} Q(novy\ stav, vsetky\ akcie)) \tag{3.1}$$

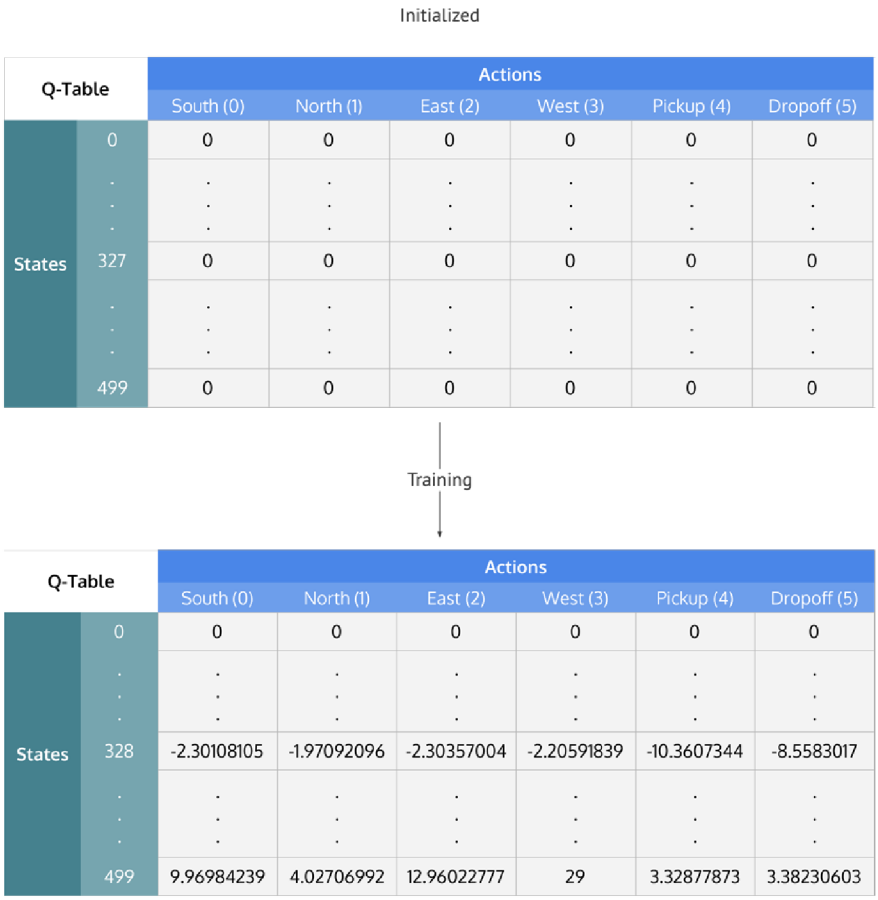
Kde:

- α je konštanta učenia ($0 < \alpha \leq 1$). α podobne ako pri učení s učiteľom vyjadruje rozsah, v ktorom sa budú Q-hodnoty meniť každú iteráciu;
- γ je miera zľavy ($0 \leq \gamma \leq 1$). Udáva váhu (dôležitosť) budúcich odmien. Hodnota blížiaca sa k 1 vyjadruje dlhodobý efekt odmien, zatiaľ čo hodnota 0 núti agenta brať do úvahy len okamžité odmeny, čím z neho robí *greedy* agenta.

Vysvetlenie rovnice 3.1:

Do Q-hodnoty agentovho aktuálneho stavu a zvolenej akcie je priradená (aktualizovaná) najprv váha $(1 - \alpha)$ starej Q-hodnoty. K nej sa pripočítajú naučené hodnoty. Naučené hodnoty sú kombináciou odmeny za zvolenie danej akcie v aktuálnom stave a zľavnenej maximálnej možnej odmeny, ktorú bude môcť agent získať v stave, do ktorého práve ide.

Q-hodnota konkrétneho stavu a akcie je teda súčet okamžitej odmeny a zľavnenej budúcej odmeny. Všetky tieto Q-hodnoty sa ukladajú do Q-tabuľky. V Q-tabuľke je riadok pre každý stav a stĺpec pre každú akciu. Q-tabuľka je veľmi podobná tabuľke odmien, ale jej význam a použitie sú úplne odlišné. Q-tabuľka je znázornená na obrázku 3.3.



Obr. 3.3: Q-tabuľka je najprv inicializovaná nulami a počas trénovania sa maximalizujú hodnoty, ktoré optimalizujú agentove správanie. Zdroj: [10]

Algoritmus Q-learningu

1. Inicializuj Q-tabuľku nulovými hodnotami.
2. Ľubovoľným spôsobom vyber jeden zo stavov a priraď ho do aktuálneho stavu (S).
3. Prehľadávaj stavový priestor:
   - 3.1. Pre aktuálny stav (S) vyber jednu zo všetkých dostupných akcií (a)
   - 3.2. Presuň sa do nového stavu (S'), ktorý je výsledkom vybranej akcie (a).
   - 3.3. Zo všetkých dostupných akcií stavu (S') vyber tú s najvyššou Q-hodnotou.
   - 3.4. Aktualizuj Q-tabuľku podľa rovnice 3.1.
   - 3.5. Nahraď aktuálny stav novým stavom.
4. Ak je dosiahnutý cieľ, opakuj postup (bod 2). Ak nie, vráť sa na bod 3.1..

Kapitola 4

Prístupy k riešeniu hry Scotland Yard pomocou algoritmov pre hranie hier

Cieľom práce je navrhnúť a implementovať umelú inteligenciu, ktorá by bola schopná rozumne hrať hru Scotland Yard. Navrhnutá AI by mala byť schopná hrať za obe strany hráčov. Využitie je možné napríklad za účelom trénovania. Navrhnutá AI by mala byť schopná konkurovať ľudskému hráčovi. Umelá inteligencia by v tomto prípade mala simulovať hráča, a preto som sa rozhodol zvoliť prístup *ľudského správania* (bližší popis v kapitole 3.1). Tento prístup bude hrať úlohu hlavne pri testovaní.

Existuje už niekoľko počítačových aj mobilných spracovaní hry Scotland Yard. Avšak z tých, ktoré som mal možnosť bližšie preskúmať, ani jedna nevyužíva AI pre hranie namiesto hráčov.

Rozhodol som sa vytvoriť 2 návrhy pre spracovanie AI v prípade hrania hry Scotland Yard. Jeden z nich (4.2) bude využívať algoritmus Alfa-beta (popísaný v kapitole 3.3). Cieľom tohto návrhu je nájsť čo najvhodnejšiu hodnotiacu funkciu, ktorá by dokázala pripodobniť správanie agentov tomu ľudskému. Druhý návrh (4.3) je založený na strojovom učení. Konkrétne bude využívať Q-learning (popísaný v kapitole 3.5). V tomto prípade je však nejasné, či bude možné agentov naučiť hľadať Mr. X hlavne kvôli veľkej stavovej expanzii.

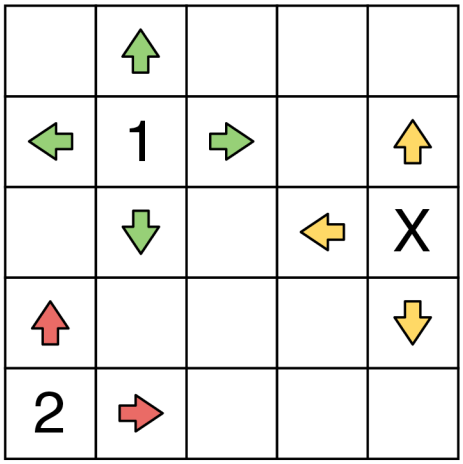
4.1 Stavový priestor a pohyb agentov

Hra Scotland Yard je tvorená hernou plochou o veľkosti 200 políčok (viac v kapitole 2.3). Na takejto hracej ploche by bolo pri 5 agentoch približne 3×10^{11} možných kombinácií rozmiestnenia agentov (stavov). S tak veľkým stavovým priestorom by bolo spracovanie daného problému výpočtovo náročné. Tiež by nebolo možné v prípade potreby zlepšiť správanie agentov, a to hlavne pri použití Alfa-bety (viď kapitola 4.2). V pôvodnej verzii hry Scotland Yard stojí výhra alebo prehra na dôkladnom premyslení ťahu. Nakoľko v prípade riešenia hry pomocou AI nebude možné vidieť do „myslenia“ hráčov, bude o to dôležitejšie sledovať, akým spôsobom sa pohybujú. V pôvodnej hre sa ale pohybujú pomerne málo a každý ťah môže rozhodovať o výhre alebo prehre. Aby bolo možné čo najlepšie skúmať správanie agentov, riešený problém som zjednodušil. Hraciu plochu som obmedzil na štvorec o veľkosti 5×5 políčok. Počet políčok sa tým zníži z 200 na 25.

Ďalšie zjednodušenie, ktoré som navrhol za účelom lepšej práce s AI, je obmedzenie počtu hráčov. Hru hrajú síce len 2 hráči (podľa pravidiel sú to 2 skupiny hráčov, ktoré v konečnom dôsledku hrajú, ako keby hrali 2 hráči), no obmedzil som počet detektívov. Vzhľadom na menšiu hraciu plochu by nemalo význam, aby jedného Mr. X hľadalo 5 detektívov. Detektívi preto budú len 2. Mr. X tiež v takomto prípade bude potrebovať úpravu. Jeho pohyb v zmenšenej hracej ploche bude možné vidieť každé 3 ťahy.

Pohyb hráčov bude možný v 4 smeroch: hore, dolu, vpravo a vľavo. Okrajmi hracej plochy nie je možné prechádzať. Políčka si preto nebudú rovnocenné – z vnútorných políčok je možné vykonať 4 rôzne ťahy, z vonkajších 3 a z rohových len 2.

Spomínané úpravy prostredia sú zobrazené na obrázku 4.1. Čísla (1, 2) zobrazujú polohu 2 detektívov, X zobrazuje aktuálnu polohu Mr. X. Šípky okolo nich znázorňujú možné akcie z daných pozícií, čím poukazujú na rozdielnosť jednotlivých políčok.



Obr. 4.1: Znázornenie stavového priestoru zjednodušenej verzie hry Scotland Yard.

Vytvoril som teda prostredie, v ktorom sú 2 detektívi, ktorí sa môžu nachádzať na jednom z 25 políčok a Mr. X, ktorý sa tiež môže nachádzať na jednom z 25 políčok. Prostredie je tvorené 15 625 možnými stavmi (25^3 – zanedbávam stavy, v ktorých by hráči boli na tom istom políčku pre jednoduchšiu implementáciu).

4.2 Návrh využitia algoritmu Alfa-beta

Algoritmus Alfa-beta je vhodný na hranie hier 2 hráčov. Hra Scotland Yard nie je určená pre 2 hráčov, ale pre 2 tímy. Tím detektívov ale musí úzko spolupracovať (viď kapitola 2.3). Musia spolu komunikovať a spoločne nájsť také ťahy pre každého detektíva, aby to bolo pre všetkých naraz výhodné. Dá sa v takom prípade uvažovať o tíme detektívov ako o jednom hráčovi, ktorý ťahá všetkými figúrkami naraz, čo je možné považovať za jeden ťah.

Preto som sa rozhodol detektívov navrhnúť ako jedného hráča, ktorého ťah je tvorený posunutím každej figúrky. Teda narozdiel od Mr. X, ktorý má v zmenšenej hracej ploche najviac 4 možnosti pohybu, u detektívov to bude najviac 16 možností (4×4).

Pohyb detektívov

Detektívi budú vidieť polohu Mr. X len raz za 3 ťahy. Vždy po prezradení polohy Mr. X detektívi najbližšie 3 ťahy nebudú mať žiadne nové informácie. Okrem prípadu, kedy by

Mr. X chytili. Vtedy už ale nové informácie nie sú potrebné, pretože hra je v danej chvíli ukončená výhrou detektívov. Je preto nevyhnutné navrhnúť ich pohyb tak, aby boli schopní pohybovať sa aj bez nových informácií.

Algoritmus Alfa-beta sa štandardne používa na vypočítanie ťahu, ktorý má agent práve vykonať. Alfa-beta spracuje síce viac ťahov dopredu, ale určuje len prvý ťah, ktorý by mal agent vykonať. Algoritmus totiž nemôže vedieť, aký ťah si protihráč skutočne vyberie, a preto je vhodné v ďalšom ťahu počítať najvýhodnejší ťah znovu vzhľadom na ťah protihráča.

V tomto prípade budú detektívi musieť využívať celý prehľadaný strom Alfa-bety, nakoľko po vykonaní ťahu nebudú mať žiadne nové informácie a budú sa musieť spoliehať len na tie informácie, ktoré mali už aj v minulom ťahu.

Pohyb Mr. X

Pohyb Mr. X je rovnako dôležitý ako pohyb detektívov. Bude tiež využívať Alfa-betu. Mr. X nemá v prípade algoritmu Alfa-beta problém s informáciami o stave hry. Môže si v každom ťahu „premyslieť“ svoj ťah. Za účelom testovania je potrebné navrhnúť pohyb Mr. X tak, aby bolo možné ovládať ho aj manuálne.

Obaja hráči sa snažia zvíťaziť. Mr. X ale má viac informácií, a preto si myslím, že je vhodné, aby sa pohyboval iným spôsobom, než detektívi. To znamená, že budú mať obaja hráči svoje vlastné hodnotiace funkcie.

Heuristická funkcia

Heuristická (hodnotiaca) funkcia je najdôležitejšou časťou algoritmu Alfa-beta. Samotný algoritmus len vytvorí strom zložený zo všetkých možných akcií v daných stavoch. Jej hodnoty budú distribuované až k aktuálnemu stavu, a teda od nej závisí, ktorý ťah bude vybraný.

Keďže cieľom detektívov je chytiť Mr. X, najväčšiu prioritu musia mať tie ťahy, kde je možné chytiť ho. Ďalším vhodným aspektom, ktorý som sa rozhodol zahrnúť do hodnotiacej funkcie, je vzdialenosť od predpokladanej pozície Mr. X. Najväčší význam má vzdialenosť v 3. ťahu, teda keď Mr. X prezradí svoju polohu. Vtedy je potrebné, aby detektívi boli čo najbližšie pri ňom a aby ho podľa novších informácií mohli chytiť.

Návrh heuristickej funkcie pre pohyb detektívov

Heuristická funkcia pre ohodnotenie **koncového uzla** v algoritme Alfa-beta ohodnocuje stavy nasledovne:

$$\sum_{i=0}^{n} f(i) \tag{4.1}$$

$$f_1(i) = (n - i) \times r \tag{4.2}$$

$$f_2(i) = -(i \times d(a_1, x) + i \times d(a_2, x)) \tag{4.3}$$

Kde:

n je maximálna hĺbka zanorenia;

i je aktuálna hĺbka zanorenia;

r je konštanta výhernej odmeny (v tomto prípade 50);

d je funkcia pre výpočet vzdialenosti medzi 2 pozíciami;

a_1, a_2 sú pozície detektívov (agentov);

x je pozícia Mr. X;

f_1 sa použije pri výherných uzloch;

f_2 sa použije pri nevýherných uzloch.

Postup hodnotenia

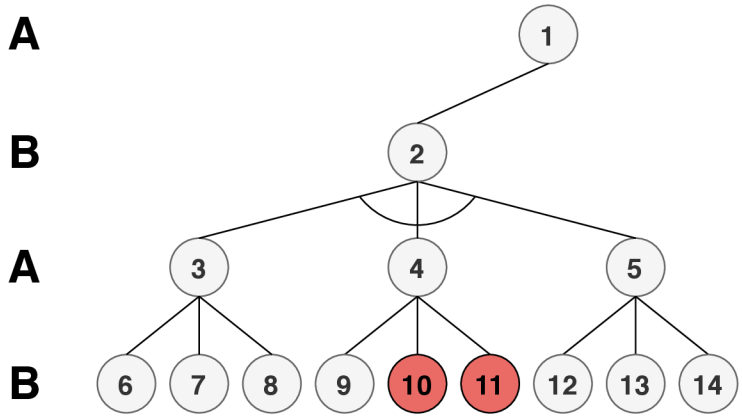
1. Pre každý rodičovský uzol tvoriaci cestu k cieľovému (okrem koreňového) sa vypočíta jeho vlastné ohodnotenie ($f(i)$).
2. Ak **je** aktuálne kontrolovaný uzol uzlom **výherným** (jeden z agentov zvolil ťah, pri ktorom sa jeho pozícia rovná aktuálne predpokladanej pozícii zlodeja), v rovnici 4.1 sa ako $f(i)$ použije rovnica 4.2. K celkovému ohodnoteniu cieľového uzla sa pripočíta $n-i$ (n – maximálne zanorenie; i – aktuálne zanorenie) násobok výherného ohodnotenia r.
3. Ak uzol **nie je výherným**, v rovnici 4.1 sa ako $f(i)$ použije rovnica 4.3. Od celkového ohodnotenia cieľového uzla sa odpočíta n násobok vzdialenosti každého z agentov od predpokladanej pozície zlodeja, kde n je hĺbka zanorenia v stromovej štruktúre.

Postup celkového hodnotenia vysvetlím na príklade, ktorý je na obrázku 4.2. Heuristická funkcia ohodnocuje len **konečné uzly**. Na obrázku 4.2 sú to uzly v poslednom riadku. Pri hodnotení uzlu číslo 6 sa uplatní bod 2 alebo 3 postupu hodnotenia na uzly 6, 3, 2 – uzol číslo 1 je koreňový, čo znamená, že sa v ňom agent práve nachádza; nemá v takom prípade zmysel ohodnocovať ho. Funkcia spočíta hodnoty jednotlivých uzlov a sumu vráti volajúcemu uzlu (koreňovému). Táto suma je ďalej spracovaná algoritmom Alfa-beta, ktorý na základe obdržanej hodnoty vyberie ťah.

V rovnici 4.2 násobenie rozdielom maximálnej hĺbky s aktuálnou hĺbkou zabezpečuje lepšie ohodnotenie cieľového uzla v prípade možnosti skoršieho chytenia.

Podobne v rovnici 4.3 násobenie aktuálnou hĺbkou spôsobuje, že vzdialenosť v neskoršom ťahu je pre detektívov dôležitejšia, pretože po prezradení polohy Mr. X by mali byť pri ňom tak blízko, ako to bude možné.

Návrh hodnotiacej funkcie nie je konečný. Očakávam možné zlepšenie po vykonaní experimentov (kapitola 5.2).



Obr. 4.2: Časť stromu vytvoreného prechodom algoritmu Alfa-beta z ukážky na obrázku 3.1.

4.3 Návrh využitia strojového učenia

Zaujímavejším variantom pre riešenie hry Scotland Yard je strojové učenie. Konkrétne bude použitý Q-learning (viď kapitola 3.5). Využitie Q-learningu je ale možné len na detektívov. Pre učenie je potrebné vytvoriť nemenné prostredie. Mr. X sa teda musí pohybovať stále rovnakým spôsobom. Stavový priestor pre učenie bude rovnaký ako v prípade Alfa-bety (kapitola 4.1).

V popísanom stavovom priestore sa pre pohyb detektívov ponúkajú 2 možnosti. Prvá možnosť je zaviesť systém 2 agentov, ktorí sa budú samostatne pohybovať. Tým pádom sa budú aj samostatne učiť. Znamenalo by to, že bude každý z nich mať maximálne 4 možné ťahy a budú sa musieť naučiť spolupracovať.

Druhá možnosť je vytvoriť jedného agenta. Tento jeden agent by predstavoval tím detektívov. Hýbal by súčasne 2 figúrkami. V takom prípade by mal agent v každom stave najviac 16 možných akcií. Takýto agent by samozrejme pri učení tiež musel zistiť, že sa mu viac oplatí hýbať figúrkami synchronizovane, ale bol by ušetrený možných „nedorozumení“ pri „rozmýšľaní“ dvoch agentov. V prípade nesprávneho ťahu jednej z figúriek, o tom bude vedieť celý tím a hlavne sa z toho poučí celý tím. Táto možnosť, okrem iného, ponúka aj možnosť jednoduchšej implementácie, keďže agent bude len jeden a nebudú musieť spolu nejakým spôsobom komunikovať.

Oba varianty majú svoje klady aj zápory. Najväčšou nevýhodou druhej možnosti je veľký počet akcií. Na základe spomínaných dôvodov som ale usúdil, že druhá možnosť – jeden agent ovládajúci 2 figúrky detektívov – bude v tomto prípade vhodnejšia.

Pohyb agentov Q-learningom

V prípade Q-learningu nebudú agenti brať do úvahy vzdialenosť od polohy Mr. X ani iné údaje, ktoré boli používané pri algoritme Alfa-beta. Hlavným a jediným indikátorom toho, či sa správne rozhodujú, bude výhra. Za výhru obdržia odmenu, ktorá sa predistribuuje do všetkých stavov, ktoré počas danej hry navštívili. Odmena teda bude pridelená ceste, nie stavu.

Detektívi sa zo začiatku budú pohybovať náhodne. Po preskúmaní dostatočného počtu ťahov začnú využívať naučené cesty, čo by malo viesť k výhre.

Aby hra nešla donekonečna (pretože pri nenaučených agentoch to je možné), zaviedol som opatrenie ukončenia hry. Podobné opatrenie je aj v originálnych pravidlách hry Scot-

land Yard. Ak detektívi nechytia hráča do určitého počtu ťahov, vyhráva Mr. X. V tomto prípade bude potrebné chytiť Mr. X do 20 ťahov. Agenti teda môžu získať kladnú odmenu za výhru alebo zápornú odmenu za prehru.

Mr. X sa bude pohybovať podobným spôsobom ako pri variante s algoritmom Alfa-beta. Bude naďalej využívať vzdialenosť od detektívov a pravdepodobnosť, že ho pri danom ťahu chytia (nepočíta pravdepodobnosť v pravom slova zmysle – ťahy, kde nasleduje viac políčok, kde je možné byť chytený, sú považované za pravdepodobnejšie).

Kapitola 5

Hranie hry Scotland Yard s využitím umelej inteligencie

V tejto kapitole budem demonštrovať využitie konkrétnych algoritmov AI v zjednodušenej verzii hry Scotland Yard (popísané v kapitole 4). Rozhodol som sa pre 2 samostatné spracovania riešenia. Jedno algoritmom Alfa-beta (kapitola 4.2), v ktorom sú detektívi aj Mr. X ovládaní práve týmto algoritmom. A druhé riešenie pomocou strojového učenia (kapitola 4.3). Q-learning v tomto prípade ale rozhoduje len o pohybe detektívov. Rozhodovanie Mr. X musí byť nemenné.

V prvej časti tejto kapitoly sa venujem implementácii, experimentom a problémom súvisiacim so spracovaním Alfa-bety. Okrem samotného rozhodovania pre oboch hráčov som implementoval aj možnosť hrať za Mr. X „manuálne“, teda živým hráčom. Táto časť je významná najmä kvôli vylepšeniu rozhodovania.

Druhá časť kapitoly je zameraná na použitie Q-learningu pre rozhodovanie detektívov. Detektívi sa musia naučiť, akým spôsobom chytiť Mr. X, a to len tak, že ho náhodou chytia a poučia sa z toho. Táto časť má tiež viac úprav. Niektoré z úprav boli aj dosť rozsiahle, keďže v pôvodnej implementácii nefungovalo všetko tak, ako by bolo potrebné.

Nakoľko mi išlo hlavne o funkcionalitu a prehľadnosť kódu, na implementáciu som využil jazyk Python. Python má množstvo dostupných knižníc využívaných v AI a v machine learningu, avšak vo výsledku z pádnych dôvodov využité nie sú.

Je dôležité uviesť zmenu terminológie v tejto časti práce. Pri implementácii výraz *agent* označuje detektíva. Pri používaní slova *agent* teda už nie je myslený „niekto“, kto „niečo“ robí v zmysle terminológie AI.

5.1 Použitie algoritmu Alfa-beta

Pred implementáciou samotného algoritmu Alfa-beta bolo potrebné implementovať stavový priestor a pravidlá pre hranie hry Scotland Yard (zjednodušenej verzie). Časť programu, ktorá obsahuje prostredie, je niečo ako pravidlá hry implementované v Pythone. Táto časť nesie údaje o pozíciach hráčov, o stave hry (napríklad počet zahraných kôl) alebo aj informáciu o tom, kedy naposledy bol Mr. X videný. Okrem informácií potrebných pre priebeh hry, sú tu aj metódy zabezpečujúce pohyb hráčov. Keďže v kapitole 4.1 bolo spomínané ignorovanie stavov, v ktorých sa hráči navzájom prekrývajú (takéto stavy v hre nie sú povolené), tento problém je riešený kontrolnými metódami práve v tejto časti. A to tak, že pred

výberom ťahu sa najprv skontroluje, ktoré z ťahov sú uskutočniteľné a len tie postupujú ďalej do výberu.

Pre riešenie algoritmu Alfa-beta pre agentov a pre Mr. X som vytvoril 2 samostatné riešenia. Hlavný rozdiel je v tom, že Mr. X využíva prehľadávanie stromu (teda aj resetovanie koreňového uzla) v každom ťahu, zatiaľ čo agenti si musia strom uložiť na 3 ťahy. Dôvodom vytvorenia samostatných riešení je hlavne prehľadnosť a čitateľnosť zdrojového kódu. Ďalším je, že každý z hráčov má rôzne používanie akcií (Mr. X má k dispozícií maximálne 4 akcie, agenti 16) a tým pádom aj rôzne metódy pre vykonanie pohybu. Obe implementácie algoritmu Alfa-beta obsahujú všetky súčasti potrebné na vykonanie vytvorenia, prehľadania a ohodnotenia stromu stavového priestoru. Každý z hráčov má aj vlastnú heuristickú funkciu.

Základným prvkom algoritmu Alfa-beta je uzol. Obsahuje okrem informácií o rodičovskom uzle, potomkoch a hodnotách *alfa* a *beta* aj dodatočné informácie, ktoré som pokladal za potrebné. Patria tam: poloha agenta 1, agenta 2 (detektívov 1 a 2 – v implementácii je využívaný výraz **agent**), predpokladaná poloha Mr. X, hĺbka zanorenia a ťah (hráč A/B) a uzol, ktorý vedie k najlepšej ceste.

Uzol vedúci k najlepšej ceste sa podľa definície Alfa-beta prerezávania 3.3 používa len v prípade koreňového uzla, pretože sa aj tak vyberie len jeden ťah a v ďalšom sa vykonáva prerezávanie znovu. V tomto prípade je ale potrebné uchovávať si celú cestu k najlepšiemu uzlu. Uchovávanie je potrebné, pretože počas vykonania 3 ťahov nebudú mať agenti žiadne nové informácie (okrem výhry) o stave hry, nakoľko aktuálnu polohu Mr. X nepoznajú. Agenti sa teda budú počas 3 ťahov pohybovať podľa jedného Alfa-beta prerezávania a až po 3 ťahoch sa vykoná nové prerezávanie a ohodnotenie možných stavov.

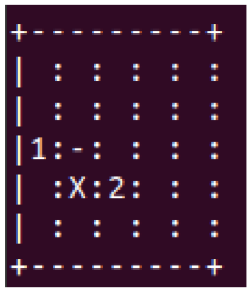
Priebeh hry

Na začiatku hry sa vygenerujú náhodné pozície hráčov tak, aby sa neprekrývali a následne sa herná plocha zobrazí. Ukážka zobrazenia hernej plochy je na obrázku 5.1. Prvý ťah majú agenti, pri ktorom sa vytvorí prázdny koreňový uzol *Node* s údajmi o aktuálnom stave hry. Následne bude prechádzať podľa algoritmu Alfa-beta (kapitola 3.3) všetky uzly, ku ktorým vedú možné ťahy. Možné ťahy sa vyberajú rozdielne pre ťah agentov a ťah Mr. X. Agenti sa môžu pohnúť všade tam, kde to herná plocha umožňuje, ale tak, aby nestáli obaja na tom istom políčku. Mr. X je ale obmedzovaný polohou oboch agentov. V prípade nemožnosti vykonať ťah Mr. X, hra končí výhrou agentov. Po prehľadaní stavového priestoru algoritmom Alfa-beta sa aktuálna pozícia agentov prepíše pozíciou v najlepšie ohodnotenom uzle.

Nasleduje ťah Mr. X, ktorý je implementovaný 2 spôsobmi. Je možné ovládať pohyb Mr. X manuálne alebo taktiež využiť Alfa-beta prerezávanie. Tieto 2 verzie som zaviedol, aby bolo možné testovať „inteligenciu“ agentov aj tým, že postavím proti nim živého hráča. Ak hrá za Mr. X hráč, program len čaká na vstup a prepíše polohu Mr. X. V prípade výberu ťahu Alfa-betou sa pri každom ťahu znovu uplatní algoritmus Alfa-beta a nájde sa nový najvhodnejší ťah.

Agenti musia vo svojom Alfa-beta strome vykonaním svojho ťahu taktiež simulovať ťah Mr. X. Tento proces sa lepšie popíše pomocou obrázku. Na obrázku 3.1 sa agent nachádza v stave 1. Vyberie si ťah, ktorý vedie do stavu 6. Teraz je na ťahu Mr. X. On si vyberie svoj ťah, ale agenti nevedia, kam sa posunul. Preto musia simulovať jeho ťah a posunúť sa v stromovej štruktúre tam, kam by pravdepodobne išiel. V tomto prípade do stavu 7. A práve stav 7 je v tomto prípade stavom, z ktorého agenti budú hľadať cestu ďalej, kým

Mr. X znovu ukáže svoju polohu a oni budú môcť vytvoriť nový Alfa-beta strom s novými informáciami.



Obr. 5.1: Ukážka zobrazenia hernej plochy po zahraní jedného ťahu oboch hráčov. Pomlčka („-“) zobrazuje poslednú známu pozíciu Mr. X.

5.2 Experimenty s algoritmom Alfa-beta

Vychádzajúc z rovníc 4.1, 4.2 a 4.3 vyzerá heuristická funkcia pre ohodnotenie koncového stavu nasledovne:

Pre každý rodičovský uzol v ceste od koreňového ku koncovému (vrátane koncového) pripočítaj hodnotu funkcie $f(i)$.

- Ak **je** uzol výherným (pre ktoréhokoľvek agenta):

$$f(i) = (3 - h) \times 50 \tag{5.1}$$

- Ak uzol **nie je** výherným:

$$f(i) = -(d(x, a_1) \times h + d(x, a_2) \times h) \tag{5.2}$$

Kde:

h je hĺbka zanorenia aktuálne hodnoteného uzlu

$d(x, y)$ je funkcia pre výpočet vzdialenosti medzi pozíciami x a y

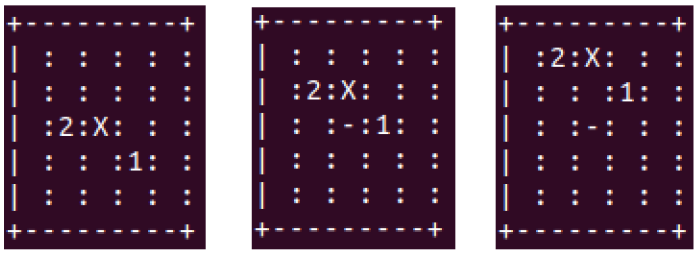
x je pozícia Mr. X

a_1, a_2 sú pozície agentov (detektívov)

Táto verzia heuristickej funkcie bola testovaná najprv na živom hráčovi. Neskôr bola testovaná aj proti Mr. X, ktorého ťahy boli vyberané Alfa-betou s ostatnými verziami heuristickej funkcie. Ale nemalo zmysel snažiť sa potvrdiť rozumné rozhodovanie tak, že by program hral sám proti sebe použitím tej istej heuristiky.

Testovanie ľudskými hráčmi prebiehalo na 4 hráčoch, pričom výsledky boli vo väčšine prípadov zhodné. Hráči v priebehu prvých 10-20 hier prehrávali. Ale približne po tomto počte sa boli schopní naučiť spôsob, ktorým agenti vyberajú ťahy a približne od 20. hry narástla výhernosť hráčov na približne 90%.

Výber ťahov sa stal po pomerne krátkom čase predvídateľným. Agenti sa v niektorých situáciach správali nerozumne. Príkladom takejto situácie je obrázok 5.2. Na obrázku 5.2 sú zobrazené 3 za sebou idúce ťahy oboch hráčov. Podobné „formácie“ agenti vytvárali dosť často a vo väčšine prípadov bolo možné len chodiť pomedzi nich a nikdy Mr. X nechytili.



Obr. 5.2: Ukážka priebehu 3 za sebou idúcich ťahov s použitím 1. verzie heuristickej funkcie.

Úpravy heuristickej funkcie – v2

Nakoľko prvotná heuristická funkcia nebola veľmi účinná, je potrebné upraviť ju tak, aby sa výsledok viditeľne zlepšil. Zaviedol som priebežne 3 úpravy, ktoré zefektívnili správanie agentov.

Prvá modifikácia spočíva v počítaní hodnôt pre každého agenta samostatne (rovnice 5.3, 5.4). Znamená to, že ak aj jeden agent má v danom uzle chytiť Mr. X, ten druhý bude hodnotený bežným spôsobom. Okrem tejto úpravy som zaviedol aj modifikáciu prípadu, keď ani jeden z agentov nie je vo výhernej pozícií (rovnica 5.5). V takom prípade sa k celkovému hodnoteniu pripočíta ohodnotenie toho agenta, ktorý je k predpokladanej pozícií Mr. X bližšie. Hodnotenie bude teda do istej miery prihliadať na to, že aj jeden agent stačí na chytenie Mr. X a nemusia tam byť obaja naraz.

2. verzia heuristickej funkcie vyzerá nasledovne:

Pre každý rodičovský uzol v ceste od koreňového ku koncovému (vrátane koncového) pripočítaj hodnotu funkcie $f_1(i)$ resp. $f_2(i)$ resp. $f_3(i)$ a to nasledovne:

- Ak je uzol výherným pre agenta 1, použi $f_1(i)$.
- Ak je uzol výherný pre agenta 2, použi $f_2(i)$.
- Ak uzol nie je výherným, použi $f_3(i)$.

$$f_1(i) = (3 - h) \times 50 - d(x, a_2) \times h \tag{5.3}$$

$$f_2(i) = (3 - h) \times 50 - d(x, a_1) \times h \tag{5.4}$$

$$f_3(i) = -min(d(x, a_1), d(x, a_2)) \times h \tag{5.5}$$

Kde platí rovnaké značenie ako pri rovniciach 5.1 a 5.2.

Pri pozorovaní hráčov pri riešení hry, bolo správanie agentov viditeľne rozumnejšie. Avšak aj napriek tomu sa hráči po približne 20 hrách naučili, kde má algoritmus slabé miesta a naučili sa ich obchádzať.

Úpravy heuristickej funkcie – v3

Druhá úprava – teda 3. verzia heuristickej funkcie – zaviedla menej zmien, ale o to významnejších. Všimol som si nedostatok agentov, ktorý sa prejavoval v prípade, že nebolo možné v priebehu 3 ťahov Mr. X chytiť. V takom prípade sa agenti nesprávali tak, ako by sa zachoval človek. Chyba spočívala vo veľkej dôležitosti konečných stavov, ktoré vychádzali len

z predpokladu. Túto dôležitosť im dávalo násobenie hĺbkou v každom výpočte. Rozhodol som sa váhu hĺbky ponechať len v prípade víťazných uzlov.

Postup predošlej verzie zostáva nezmenený. Rovnice 5.3-5.5 boli nahradené nasledovnými rovnicami:

$$f_1(i) = (3 - h) \times 50 - d(x, a_2) \tag{5.6}$$

$$f_2(i) = (3 - h) \times 50 - d(x, a_1) \tag{5.7}$$

$$f_3(i) = -min(d(x, a_1), d(x, a_2)) \tag{5.8}$$

Táto úprava pri výslednom testovaní zdvihla výhernosť agentov o takmer 10%.

Úpravy heuristickej funkcie – v4

Tretia úprava sa týkala len prípadu nevýherného uzlu pre oboch agentov. Už v predošlých verziách som experimentoval s významnosťou pozície agenta, ktorý je bližšie k predpokladanej pozícií Mr. X. Tieto experimenty spočívali v tom, že som do celkového ohodnotenia počítal len hodnotenie agenta, ktorý je k cieľu bližšie. To však spôsobilo, že pozícia agenta, ktorý bol momentálne ďalej, nebola vôbec braná do úvahy.

Preto som zaviedol hodnotenie oboch agentov. Význam toho, ktorý je bližšie som ponechal. A to tak, že funkcia pripočíta len polovicu hodnotenia vzdialenejšieho agenta. Hodnotiaca funkcia teda zoberie do úvahy oboch agentov, ale väčšiu váhu bude mať ten, ktorý sa dostane bližšie k Mr. X. Úprava heuristickej funkcie znie nasledovne:

Ak uzol nie je výherným, použi obe funkcie $f_3(i)$ aj $f_4(i)$.

$$f_3(i) = -min(d(x, a_1), d(x, a_2)) \tag{5.9}$$

$$f_3(i) = -max(d(x, a_1), d(x, a_2))/2 \tag{5.10}$$

Výsledky experimentov

Experimenty s riešením hry Scotland Yard pomocou algoritmu Alfa-beta viedli k úpravám heuristickej funkcie, ktoré boli popísané v tejto kapitole. Prvá verzia mala viditeľné nedostatky, ktoré som aj hneď uviedol pri popise danej heuristiky. Pri ostatných verziách neboli nedostatky tak zrejmé. Išlo skôr o snahu pochopiť pri sledovaní hry, prečo agenti v rozhodujúcich momentoch zvolili ťah, ktorý zvolili. Následne bolo potrebné toto pochopenie previesť do heuristickej funkcie.

Po dosiahnutí finálnej verzie – v tejto kapitole popísanej ako verzia 4 – som pre porovnanie uplatnil túto verziu na výber ťahov Mr. X a postupne som spustil hru pre všetky doterajšie verzie použité na výber ťahov agentov. Výsledky boli nazbierané zo 100 za sebou idúcich hier s náhodným počiatočným rozložením figúrok hráčov a sú zapísané v tabuľke 5.1. Výsledky v prípade opakovaného spustenia sa vzhľadom na náhodné rozloženie môžu mierne líšiť.

Údaje v tabuľke 5.1 ukazujú postupné zlepšenie vo verziách 2-4. Čo však stojí za povšimnutie, je verzia jedna, ktorá má najvyššiu úspešnosť. Dôvodom je fakt, že Mr. X využíva heuristiku verzie 4. Verzia 1 zahŕňala potrebné faktory na nájdenie Mr. X. Hráči tiež prvé

hry proti programu využívajúcemu verziu 1 na agentoch prehrali. Ale rýchlo sa naňho dokázali adaptovať, čo Mr. X ovládaný algoritmom Alfa-beta nedokáže. Pozoruhodný je fakt, že v priemere agenti chytili Mr. X okolo druhého prezradenia svojej pozície. Pri sledovaní priebehu hry s použitím verzie 4 som si všimol, že agenti vo väčšine hier zaženú Mr. X „do kúta“ (do rohových políčok). To je pre agentov najlepšia stratégia, pretože v týchto políčkach má Mr. X najmenej možností úniku.

Tabuľka 5.1: Prehľad výhernosti agentov a dĺžky hier v hrách s použitím 4 verzií heuristickej funkcie na agentoch a finálnej verzie funkcie na Mr. X.

	verzia 1	verzia 2	verzia 3	verzia 4
Výhernosť (v %)	92	65	74	89
Priemerná dĺžka výhernej hry	5.8	6.4	6.3	6.7

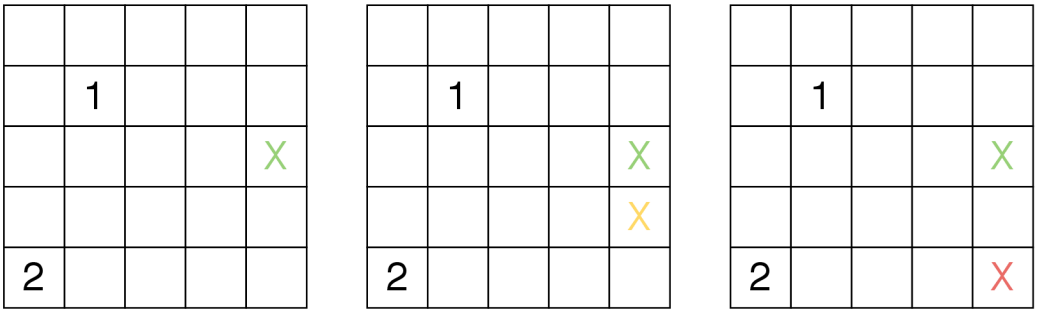
Znamená to, že 4. verzia heuristickej funkcie vyberá ťahy tak, ako by to robil hráč. Nie je schopná adaptovať sa zmenám, ale dá sa povedať, že sa správa rozumne a do istej miery aj ľudsky (prístupy popísané v kapitole 3.1) – čo v konečnom dôsledku bolo cieľom práce.

5.3 Použitie Q-learningu

Podobne ako pri využívaní algoritmu Alfa-beta v kapitole 5.1 je aj tu potrebné najprv vytvoriť stavový priestor. V tomto prípade bude stavový priestor mierne odlišný od toho v kapitole 5.1. Stavový priestor bude tvorený 4 prvkami – pozíciou agenta 1, pozíciou agenta 2, pozíciou Mr. X a **časom, kedy bol Mr. X naposledy videný**.

Naposledy videná pozícia Mr. X tu mení celú situáciu. Vysvetlím to na obrázku 5.3. Čísla 1 a 2 znamenajú pozície agentov. Zelené písmeno X znamená aktuálnu polohu Mr. X. Ak je na hracej ploche len zelené „X“, znamená to, že Mr. X sa momentálne nachádza na pozícií, ktorú je vidieť na hracej ploche. Ak je okrem zeleného „X“ na ploche aj žlté, znamená to, že Mr. X bol videný pred 1 ťahom, pričom žlté „X“ znamená poslednú videnú pozíciu. Agenti poznajú len jeho naposledy videnú pozíciu (v tomto prípade žlté „X“). Podobne aj červené „X“ znamená jeho poslednú známu pozíciu, ale 2 ťahy po tom, ako bol videný.

V tomto chápaní stavového priestoru sú na obrázku 5.3 až 3 rôzne stavy aj napriek tomu, že figúrky sú rozmiestnené rovnako.



Obr. 5.3: Zobrazenie rôznych stavov s rovnakým rozmiestnením figúrok, ktoré má poukázať na dôležitosť času, kedy bol Mr. X naposledy videný.

Vytvorenie prostredia pomocou knižnice OpenAI Gym

Rozhodol som sa použiť nástroj od OpenAI – Gym [1]. Táto knižnica je určená pre prácu s reinforcement learningom a obsahuje množstvo ukážkových prostredí, na ktorých je možné vyvíjať a porovnávať algoritmy reinforcement learningu. Vytvoril som si vlastné prostredie. Toto prostredie je veľmi podobné prostrediu už existujúcemu v knižnici Gym – prostredie Taxi. Obe prostredia sú tvorené stavovým priestorom 5×5 políčok a hráči sa môžu pohybovať 4 smermi.

V priebehu testovania som zistil, že knižnicu Gym nebude možné na riešenie tohto problému použiť. Dôvodom je, že táto knižnica vyžaduje len implementáciu prostredia a následne implementáciu algoritmu učenia. O všetko ostatné sa postarajú súčasti knižnice Gym. Problémom však bolo, že som potreboval sám ovládať zmenu stavov, nakoľko zmena stavu nevychádzala len z akcie, ale aj z pohybu Mr. X. Ale jeho pohyb nebolo možné zahrnúť do akcií agentov, preto nakoniec nebolo možné ani správne narábať s prechodmi medzi stavmi po vykonaní akcie. Pravdepodobne sa následkom toho nesprávne priradzovali odmeny a učenie nebolo úspešné.

Implementácia bez prídavných knižníc pre reinforcement learning

Program bolo potrebné prerobiť tak, aby nevyužíval knižnicu Gym, ale sám spracovával celé učenie. Časť tvoriaca prostredie aj naďalej tvorí hernú plochu, inicializáciu stavov a možnosti pohybu. Ako už bolo spomínané v tejto kapitole, stav je tvorený štvoricou – pozície agenta 1, agenta 2, **posledná známa pozícia** Mr. X a posledné videnie Mr. X. Je vhodné podotknúť, že agenti nemajú žiadnu informáciu o reálnej pozícií Mr. X. Vedia len, kde bol naposledy videný a kedy tam bol videný. Bez knižnice Gym bolo potrebné implementovať aj prácu s Q-tabuľkou a distribúciu odmien podľa vzorca 3.1.

Ako som spomínal vyššie, počas učenia je potrebné manuálne manipulovať so stavmi a meniť ich nie len na základe vykonanej akcie. Priebeh algoritmu učenia bol teda mierne odlišný od toho, ktorý je popísaný v kapitole 3.5.

Po spustení programu prebehnú inicializácie potrebných súčastí. Vytvorila sa tabuľka stavov a akcií a Q-tabuľka s nulovými hodnotami. Následne sa vygenerujú počiatočné pozície hráčov a začne cyklus hry. Z dostupných akcií agentov sa podľa algoritmu Q-learningu vyberie jedna. Následne sa stav ale nemení. Nová pozícia agentov je uložená len v dočasnom stave. Mr. X vyberie ťah na základe dočasného stavu a teda aktuálnej pozície agentov. Až po ťahu Mr. X sa prepočíta stav na základe pohybu Mr. X a zmeny doby od jeho posledného zviditeľnenia. Táto časť je dôvodom nepoužiteľnosti knižnice Gym. V nej nebolo možné dosiahnuť manuálnu zmenu stavov, ktorá by nezávisela len od akcie agentov.

Po vykonaní pohybu a získaní nového stavu sa aktualizuje Q-tabuľka spôsobom popísaným v kapitole 3.5. Implementácia funkcie pre aktualizáciu Q-tabuľky bola inšpirovaná existujúcim riešením učenia Q-learningom od Kyle Kastner [11]. Kvôli dlhému trvaniu výpočtov som prebral zobrazenie priebehu programu – „progress bar“ [7].

Výber ťahu pre Mr. X sa riadi algoritmom Alfa-beta.

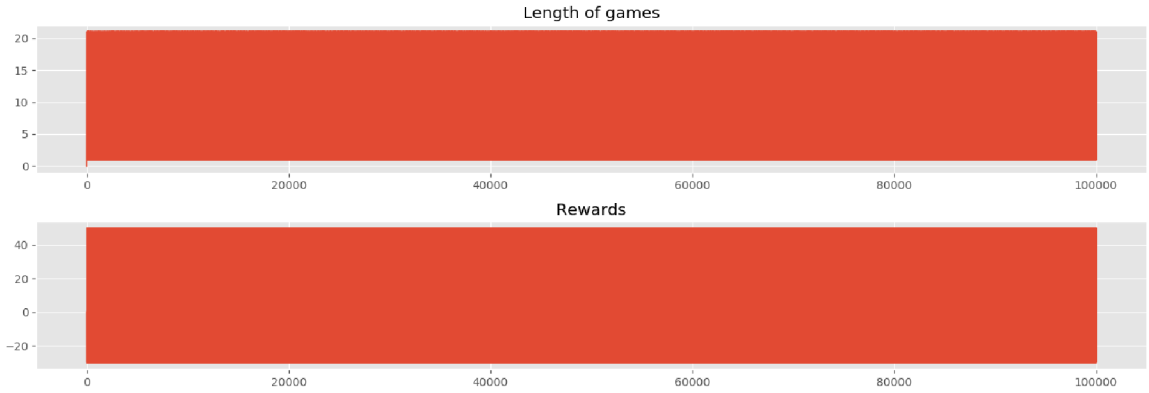
5.4 Experimenty s Q-learningom

Výsledky experimentov s učením agentov pomocou Q-learningu neboli veľmi priaznivé. Po implementácií učenia prebiehalo experimentovanie s odmenami (za výhru aj prehru), s α aj γ. Za ϵ som vymyslel náhradu, aby som si bol istý správnym výberom akcií – ϵ ovplyvňuje,

či bude vybraná akcia z naučených alebo bude preskúmavať nové akcie viď kapitola 3.4. Výsledky boli žiaľ vždy rovnaké.

Po použití algoritmu Alfa-beta, ktorý som používal v minulých kapitolách som sa rozhodol zaviesť pre výber ťahov Mr. X jednoduchšiu metódu hrania hier – mini-max (algoritmus mini-max je veľmi podobný Alfa-bete, ale je zjednodušený). Dôvodom pre túto úpravu bola vysoká výpočtová náročnosť a snaha o zrýchlenie priebehu učenia. Zrýchlenie bolo síce úspešné, ale učenie ešte stále nebolo dosť efektívne na to, aby sa v reálnom čase agenti naučili hrať hru Scotland Yard.

Na obrázku 5.4 je zobrazený jeden z výsledkov pri učení na 100 000 za sebou idúcich hrách. Výsledkom je graf zobrazujúci dĺžku jednotlivých hier a konečné odmeny za hry (graf je pri 100 000 výsledkoch tak hustý, že pri rôznych výsledkoch každú hru vyzerá ako súvislá plocha). Nakoľko výsledky sú počas celého učenia rozptýlené do celého rozsahu grafu, usúdil som, že ide o náhodné výsledky, ktoré vznikajú z náhodných akcií. Správne fungujúce učenie by sa malo v takomto prípade ukázať na grafe tým, že pri dĺžke hier začne graf postupne klesať a hry budú kratšie. Odmeny by mali začať rásť a už nedosahovať minimálne hodnoty, ktoré boli dosahované pri učení.



Obr. 5.4: Výstup učenia Q-learningom na 100 000 hrách zobrazujúci dĺžku každej z hier (v ťahoch) a celkovú odmenu získanú za hru.

Experimentoval som aj s náhradou ϵ, a to tak, že som nahradil výber medzi objavovaním a využívaním pomocou ϵ svojou vlastnou funkciou. Táto funkcia spočítavala celkovú odmenu, ktorá môže byť získaná zo všetkých akcií dostupných z aktuálneho ťahu. Ak suma presiahla stanovenú hodnotu, až vtedy umožnila využívať výber podľa ϵ. Mohol som tak sledovať aj vývoj využívania naučených akcií. No výsledok bol sklamaním. Po asi 100 000 hrách sa využívali naučené akcie len približne 2-4-krát za hru.

Znamená to, že by učenie potrebovalo rádovo viac hier na učenie. Avšak pri snahe vyhovieť tejto požiadavke som narazil na hranice svojich možností. Pri spustení učenia na 500 000-1 000 000 hier, trval priebeh učenia 6-8 hodín (v závislosti od aktuálnych úprav programu) a aj tak to nebolo dostatočne veľa na to, aby sa učenie prejavilo tak, aby výber naučených ťahov viedol k víťazstvu. S mojimi možnosťami som nebol schopný otestovať, či by väčšie množstvo hier skutočne pomohlo, ale som presvedčený o tom, že by to tak bolo.

Snahu riešiť hru Scotland Yard pomocou strojového učenia Q-learningom som musel ukončiť s neúspechom.

5.5 Možné zlepšenie do budúcnosti

V rámci tejto práce sa mi nepodarilo naučiť agentov vyberať ťahy tak, aby dosiahli výhru, avšak prišiel som ešte na zopár možností zefektívnenia učenia, ktoré by v budúcnosti mohli viesť k úspešnému učeniu aj s podobnými možnosťami, ktoré som mal ja.

Do budúcnosti by bolo možné vyskúšať upraviť stavový priestor tak, aby agenti videli hraciu plochu ako „sadu vzdialeností a smerov“. Presnejšie povedané agenti majú v štvorcovej hracej ploche 4 rovnaké rohy a od nich odvíjajúce sa možnosti. Pre lepšie pochopenie je možné predstaviť si, že v každej situácií, sa dá hracia plocha otočiť tak, aby agent bol v ľavej hornej štvrtine hracej plochy. A nech by bol agent v ktorejkoľvek časti, situácia je presne taká istá, ako keby bol vľavo hore. Takýmto spôsobom by bolo možné Q-tabuľku zjednodušiť a vypĺňať tak len $\frac{1}{4}$ z celej doterajšej Q-tabuľky. Na obrázku 5.5 je rovnakými číslami (resp. rovnakými farbami) znázornené, ktoré políčka by sa dali považovať za rovnocenné a tým pádom využívať na ne tú istú položku v Q-tabuľke. Stavový priestor by sa tým mohol pri správnom použití zmenšiť až o približne 70% (presná hodnota závisí od konkrétneho spracovania – čím menej políčok bude učenie brať do úvahy, tým viac sofistikované musí byť priradzovanie reálnej situácie určitému stavu).

1	2	5	3	1
3	4	6	4	2
5	6	7	6	5
2	4	6	4	3
1	3	5	2	1

Obr. 5.5: Návrh úpravy stavového priestoru pre efektívnejšie využívanie Q-tabuľky.

Ďalšia z možných úprav spočíva v oddelení agentov a vytvorenie 2 samostatných agentov rozhodujúcich sa nezávisle od seba. V tomto prípade by bolo vhodné, aby používali tú istú Q-tabuľku (keďže sa pohybujú v tom istom priestore, platia pre nich rovnaké pravidlá a majú rovnaký cieľ. Využívanie zdieľanej Q-tabuľky by zabezpečilo rýchlejšie naplnenie dátami a mohlo by pomôcť rýchlejšiemu učeniu.

Posledným a podľa mňa významným zlepšením je priebežné hodnotenie. V tejto práci sa ohodnocoval celý priebeh hry a to až po skončení hry. Agenti teda hodnotili svoje ťahy len na základe výhry alebo prehry. V prípade, že by hodnotili priebežne, nezáviselo by učenie len od počiatočného rozloženia hráčov, ale v priebehu hry by sa mohli dostať do rovnakých situácií, v ktorých už sa ocitli v minulej hre. Vedeli by tak rýchlejšie zareagovať na rovnakú situáciu aj v prípade, že začali inde. V takomto prípade by bolo vhodné vyhodnocovať úspešnosť vybranej cesty po každom zviditeľnení Mr. X a brať do úvahy jeho vzdialenosť podobne ako pri spracovaní algoritmom Alfa-beta.

Kapitola 6

Záver

Cieľom tejto práce bolo navrhnúť a implementovať umelú inteligenciu, ktorou by bolo možné realizovať systém, ktorý by autonómne hral hru Scotland Yard.

Splniť tento cieľ sa mi podarilo použitím klasických metód hrania hier, konkrétne s využitím algoritmu Alfa-beta. Snahu zapojiť do riešenia hry strojové učenie som ukončil bez úspechu.

V prvom rade som sa oboznámil s pravidlami stolných hier, so štruktúrou a priebehom takýchto hier. Dôkladne som si preštudoval pravidlá hry Scotland Yard a navrhol som spôsob vytvorenia herného priestoru pre počítačovú verziu tejto hry.

Pre realizáciu autonómneho systému, ktorý by hral hru Scotland Yard som navrhol riešenie pomocou algoritmu Alfa-beta, ktorý sa využíva pri hrách s 2 hráčmi, ktorí sa striedajú v ťahoch. V tomto prípade mali byť obe strany ovládané výberom ťahov pomocou algoritmu Alfa-beta. Navrhol som aj riešenie strojovým učením, metódou Q-learning, ktoré by sa v hre Scotland Yard uplatnilo len na stranu detektívov (hľadajúcich). Toto obmedzenie na jednu stranu hráčov bolo nutné pre vytvorenie nemenného prostredia za účelom učenia.

V prípade využitia algoritmu Alfa-beta pre obe hráčske strany sa mi podarilo dosiahnuť také výsledky, pri ktorých po nahradení jednej strany ľudským hráčom bol systém schopný hrať rovnocennú hru s protihráčom. Pri zapojení Q-learningu som narazil na problém veľkej stavovej expanzie aj pri zjednodušenom prostredí a tomu nedostatočnú výpočtovú kapacitu vlastných prostriedkov.

Spracovanie algoritmom Alfa-beta bolo v konečnej verzii na takej úrovni, že hráči pri jednoduchej hracej ploche 5×5 políčok ani po sérií približne 20 hier nedosiahli jednoznačnú výhru. Ťahy vyberané algoritmom Alfa-beta boli hráčmi považované za rozumné.

Práca bola pre mňa prínosom hlavne v pohľade na možnosti riešenia podobných problémov. So skúsenosťami z tejto práce som navrhol niekoľko vylepšení, ktoré by mohli zefektívniť spracovávanie učenia Q-learningom v hre Scotland Yard a pravdepodobne by vo veľkej miere napomohli tomu, aby učenie prebehlo úspešne. Pri správnej „kompresii“ stavového priestoru by bolo možné zmenšiť stavový priestor až o približne 70% a rádovo tak navýšiť rýchlosť učenia. Náročnosť takej úpravy mi neumožnila zahrnúť ju do tejto práce, ale bolo by možné uplatniť tieto zlepšenia v naväzujúcej práci.

Literatúra

[1] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J. et al. *OpenAI Gym*. GitHub, 2016. Dostupné z: https://github.com/openai/gym.

[2] Chess.com. *Kasparov vs. Deep Blue | The Match That Changed History* [online]. 2018 [cit. 2020-06-25]. Dostupné z: https://www.chess.com/article/view/deep-blue-kasparov-chess.

[3] Chess.com. *Openings* [online]. 2020 [cit. 2020-06-25]. Dostupné z: https://www.chess.com/sk/openings.

[4] Coggan, M. *Exploration and Exploitation in Reinforcement Learning*. Montreal: McGill University, 2004. Dostupné z: https://neuro.bstu.by/ai/To-dom/My_research/Papers-2.1-done/RL/0/FinalReport.pdf.

[5] cosplaybattle. *Všetky druhy spoločenských hier. Najzaujímavejšie stolové hry. Každý hrá sám za seba* [online]. 2000 [cit. 2020-06-25]. Dostupné z: https://cosplaybattle.ru/sk/novichkam/vse-vidy-nastolnyh-igr-samye-interesnye-nastolnye-igry/.

[6] Follett, J. *How 22 Years of AI Superiority Changed Chess* [online]. 2019 [cit. 2020-06-25]. Dostupné z: https://towardsdatascience.com/how-22-years-of-ai-superiority-changed-chess-76eddd061cb0.

[7] greenstick. *Python: printProgressBar function with autoresize option*. GitHub, 2019. Dostupné z: https://gist.github.com/greenstick/b23e475d2bfdc3a82e34eaa1f6781ee4.

[8] IBM. *Deep Blue* [online]. 2011 [cit. 2020-06-25]. Dostupné z: https://www.ibm.com/ibm/history/ibm100/us/en/icons/deepblue/.

[9] Jurčo, B. P. *Reinforcement learning v robotike*. Bratislava, 2011. Diplomová práca. Univerzita Komenského v Bratislave Fakulta matematiky, fyziky a informatiky. ISBN 8b03cc49-304b-43f4-9a0d-d1c86895fe95. Dostupné z: http://dai.fmph.uniba.sk/~petrovic/th11/Jurco.pdf.

[10] Kansal, S. a Martin, B. *Reinforcement Q-Learning from Scratch in Python with OpenAI Gym* [online]. 2018 [cit. 2020-06-25]. Dostupné z: https://www.learndatasci.com/tutorials/reinforcement-q-learning-scratch-python-openai-gym/#TheReinforcementLearningProcess.

[11] Kastner, K. *Painless Q-Learning Tutorial implementation in Python*. GitHub, 2019. Dostupné z: https://gist.github.com/kastnerkyle/d127197dcfdd8fb888c2.

[12] Pijanowski, L. a Pijanowski, W. *Encyklopedie světových her*. 1. vyd. Praha: Universum, 2008. ISBN 978-80-242-2225-7.

[13] Russell, S. J. a Norvig, P. *Artificial Intelligence: A Modern Approach*. 3. vyd. New Jersey: Pearson Education, Inc., 2010. ISBN 978-0-13-604259-4.

[14] Russell, S. J. a Norvig, P. Artificial Intelligence. In: Russell, S. J. a Norvig, P., ed. *Artificial Intelligence A Modern Approach*. 3. vyd. Upper Saddle River, New Jersey 07458: Pearson Education, Inc., 2010, s. 1–29. ISBN 978-0-13-604259-4.

[15] Russell, S. J. a Norvig, P. Reinforcement learning. In: Russell, S. J. a Norvig, P., ed. *Artificial Intelligence A Modern Approach*. 3. vyd. Upper Saddle River, New Jersey 07458: Pearson Education, Inc., 2010, s. 830–859. ISBN 978-0-13-604259-4.

[16] Václav, N. *Umělá inteligence pro hraní her*. Brno, 2012. Bakalárska práca. FIT VUT v Brně. Dostupné z: https://dspace.vutbr.cz/bitstream/handle/11012/55269/final-thesis.pdf?sequence=6&isAllowed=y.

[17] Watkins, C. J. C. H. *Learning from Delayed Rewards*. Cambridge, 1989. Dizertačná práca. University of Cambridge, Psychology Department. Dostupné z: https://www.researchgate.net/publication/33784417_Learning_From_Delayed_Rewards.

[18] Zbořil, F. V. a Zbořil, F. *Základy umělé inteligence: Studijní opora*. Brno: Fakulta informačních technologií VUT v Brně, máj 2012.

Príloha A

Scotland Yard – Pravidlá hry

Námet hry

Zlodej zvaný Mr. X sa snaží striasť svojich prenasledovateľov v Londýne. Uteká pred nimi taxíkom, autobusom alebo metrom. Len tým najmúdrejším detektívom sa ho môže podariť chytiť.

Mr. X sa snaží až do konca hry o to, aby svojim prenasledovateľom unikol a skryl miesto svojho pobytu.

Detektívovia sa naopak snažia správne si vyložiť ťahy Mr. X a vystopovať ho.

Cieľ hry:

Mr. X sa snaží uniknúť svojim prenasledovateľom a zostať neobjavený, pokým detektívovia nevyčerpajú všetky svoje ťahy.

Detektívovia sa snažia nájsť Mr. X tým, že sa dostanú na zastávku, na ktorej sa práve nachádza Mr X.

Príprava

Hráči sa dohodnú, kto prevezme rolu **Mr. X**. (Tip: pre túto rolu sú potrebné pevné nervy, preto by ju ma zobrať skúsený hráč). Ostatní hráči prevezmú rolu **detektívov**. Položte lístky na plochu k všeobecnému použitiu.

Mr. X obdrží:

- 1 bielu hraciu figúrku
- šilt pre Mr. X (zabráni, aby detektívi videli, kam sa Mr. X pozerá)
- tabuľku jázd s vloženým papierom
- ceruzku
- lístky:

  5x čierny lístok

  2x dvojitý ťah

Každý detektív obdrží:

- 1 farebnú hernú figúrku podľa vlastnej voľby a príslušnú tabuľku a lístky
- lístky, ktoré položí na svoju tabuľku s lístkami:

  4x metro

  8x autobus

  11x taxi

Štartovné postavenie

Aby ste si určili štartovné pozície, roztrieďte **štartovacie karty** podľa ich zadných strán (D a X). Premiešajte zvlášť karty D a zvlášť karty X a položte ich prikryté na stôl. Každý **detektív** si vytiahne jednu štartovaciu kartu s písmenom D na zadnej strane a postaví svoju hraciu figúrku na odpovedajúcu stanicu. Teraz si **Mr. X** skryte vyberie štartovaciu kartu s písmenom X na zadnej strane a pozrie sa na ňu. Svoju figúrku ale nepostaví na hraciu dosku.

Priebeh hry

Hrá sa až 22 kôl. Jedno kolo sa skladá z toho, že Mr. X najprv vykoná svoj ťah a nakoniec v ľubovoľnom poradí všetci detektívi. K tomu **musí** Mr. X a každý detektív zmeniť svoje stanovisko. Každý detektív použije pre prejdenú trasu lístok, **ktorý pridá zo svojej tabuľky lístkov do všeobecných zásob**. Mr. X si berie svoje lístky **vždy** zo všeobecných zásob.

Ako sa ťahá?

Každé miesto na hracej doske je stanica pre 1 až 3 dopravné prostriedky (taxi, autobus, metro). Farby staníc označujú, aké dopravné prostriedky odtiaľ vychádzajú a kde zastavujú. Aby ste mohli použiť nejaký dopravný prostriedok, musí stáť hracia figúrka na stanici pre tento dopravný prostriedok (farba dopravného prostriedku je zahrnutá vo farbe stanice).

> **Taxíkom** (žltá) sa dostanete na každé miesto na hracej doske. Trasa, ktorú môžete prejsť, je ale krátka: môžete sa posunúť len (pozdĺž žltej linky) k ďalšiemu miestu.
>
> **Autobus** (tyrkysová) ide len z miest s tyrkysovým polkruhom v stanici; s ním môžete prejsť po autobusových linkách o niečo dlhšie trasy.
>
> **Metro** (červená) ide po červených linkách a môže rýchlo prejsť veľké vzdialenosti. Na hracej doske je ale len málo staníc metra (miesta s červeným vnútorným štvorcom v stanici).

Hráč použije lístok odpovedajúci farbou a posunie sa k ďalšej stanici. Prejdenú trasu môžete prejsť znovu späť pri ďalšom ťahu.

Všetky figúrky sa môžu presúvať len na voľné miesta. Ak Mr. X nemôže ťahať na žiadne voľné miesto, prehral. Ak detektív ťahá na miesto, na ktorom sa nachádza Mr. X, vtedy Mr. X tiež prehral. Detektívi nesmú nikdy stáť na rovnakom mieste.

Hracie ťahy Mr. X

Mr. X vykonáva svoje ťahy skryto. Vyberie si pre to tajne novú stanicu, ktorá je nejakou linkou priamo spojená s aktuálnym stanoviskom. Zapíše číslo novej stanice na ďalšie voľné pole svojej tabuľky jázd. Svoj zápis prikryje použitým lístkom. Detektívovia teda vedia, akým dopravným prostriedkom sa Mr. X presúval, ale nevedia, kam. Keď je pri budúcom hracom ťahu opäť na rade, ťahá z tejto zaznamenanej stanice ďalej k ďalšej stanici, ktorú si vyberie atď.

Hracie ťahy detektívov

Po tom, ako Mr. X ukončí svoj ťah, prídu na rad v ľubovoľnom poradí detektívi. Pretože detektívi majú spoločný cieľ, mali by sa hráči dobre dohodnúť na svojich ťahoch pri hre. Každý detektív odovzdá do zásob svoj práve použitý lístok a presunie svoju figúrku na zvolenú stanicu dopravného prostriedku, ktorý si vyberie.

V priebehu hry platí:

- Detektívi majú obmedzené zásoby lístkov. Ak nemá detektív už žiadny lístok na jeden z dopravných prostriedkov, nemôže tento dopravný prostriedok používať.
- Ak už detektív nemá vôbec žiadne lístky alebo už svojimi lístkami nemôže ťahať, musí hru opustiť.
- Detektívi nesmú medzi sebou vymieňať lístky.
- Lístky detektívov ležia vždy viditeľne, aby Mr. X mohol zistiť, aké dopravné prostriedky majú jeho prenasledovatelia ešte k dispozícii.

Špeciálne ťahy Mr. X

Mr. X sa objavuje

Mr. X sa musí ukázať v pravidelných intervaloch, a to po svojom **3., 8., 13., 18. a 24. ťahu**. Ťahy, v ktorých sa musí objaviť sú označené na tabuli jázd zakrúžkovanými číslami a väčším poľom na písanie.

Rovnako ako inokedy vykoná Mr. X svoj zápis na tabuľu jázd a položí na ňu svoj lístok. Potom postaví svoju hraciu figúrku na aktuálne stanovisko. Teraz majú detektívi veľkú šancu Mr. X obkľúčiť alebo chytiť. Nezostáva ale veľa času, pretože už pri ďalšom ťahu si Mr. X vezme svoju hraciu figúrku z dosky a zmizne.

Dvojitý ťah

Ak hrá Mr. X s lístkom na dvojitý ťah, môže v tomto kole navštíviť hneď dve miesta a to v **akejkoľvek povolenej** kombinácií 2 dopravných prostriedkov. Zaznamená obe stanice na tabuľke jázd (do 2 samostatných polí) a položí na ňu 2 lístky použitých dopravných prostriedkov.

Lístok na dvojitý ťah sa dostane z hry. Ak je prvým miestom miesto, na ktorom sa musí Mr. X objaviť, ukáže sa tam, ale zmizne hneď s druhým ťahom. Pretože sa dvojitý ťah hrá priamo za sebou ako 2 normálne hracie ťahy, nesmie Mr. X ani pri prvom ani pri druhom

ťahu vstúpiť na pole, na ktorom stojí detektív. V jednom kole smie Mr. X použiť len 1 lístok na dvojitý ťah.

Čierne lístky

Mr. X môže miesto normálneho lístku použiť čierny lístok, ktorý platí pre **akýkoľvek** dopravný prostriedok. S čiernym lístkom môže Mr. X (a **len** Mr. X) tiež použiť **trajekt** (a tak ísť napr. z bodu 157 do bodu 115). Toto spojenie (čierne linky) nie je možné použiť žiadnym iným lístkom.

Ak Mr. X použije čierny lístok, neobdržia detektívi žiadne upozornenie na to, aký dopravný prostriedok Mr. X mohol použiť. Čierne lístky sa môžu používať aj v rámci dvojitých ťahov. Ako obvykle sa odkladajú na tabuľku jázd.

Koniec hry

Detektívi vyhrávajú, ak:

- sa v akomkoľvek okamihu počas hry ocitnú detektív a Mr. X súčasne na rovnakej stanici. V tomto prípade sa musí Mr. X nechať vidieť.

Mr. X vyhráva, ak:

- stihne do konca 22. kola hry ísť Londýnom bez toho, aby ho detektívi chytili. Kolo je ukončené až vtedy, keď nakoniec ešte raz ťahajú detektívi.
- už žiadny z detektívov nemôže ťahať.