PŘÍRODOVĚDECKÁ FAKULTA UNIVERZITY PALACKÉHO
KATEDRA INFORMATIKY

BAKALÁŘSKÁ PRÁCE

Texas hold'em limit pokerbot

2013

Jan Vytřísal

Anotace

Texas Hold'em limit je rozsáhlá hra o 10^{18} *uzlech. Není ji možné vyřešit přímo. Rozdělil jsem ji proto na dvě části. Preflop model hledá optimální strategii v prvním sázkovém kole. Postflop model Monte Carlo simulací zjišťuje, která akce má nejlepší očekávanou hodnotu. Výsledná optimální strategie se nedokáže beze ztrát ubránit proti jednoduchému* CallBotovi. *Strategie tak není úplně optimální a je to do jisté míry způsobeno zaokrouhlováním hodnot. Naproti tomu Monte Carlo simulace se dokáže přizpůsobit jednoduchým strategiím druhého hráče. Celý pokerbot si obstojně vede i proti lidskému hráči.*

Děkuji vedoucímu práce Janu Konečnému za cenné rady a mojí matce za neustálou podporu.

Obsah

1. Úvod **7**

2. Pravidla hry **8**
2.1. Hra podrobně 8

3. Přehled přístupů tvorby pokerbotů **10**

4. Tvorba pokerbota **13**
4.1. Preflop model 13
4.1.1. Teorie her 13
4.1.2. Reprezentace hry 14
4.1.3. Lineární programování obecně 16
4.1.4. Lineární programování v pokeru 18
4.1.5. Realizace preflop modelu 19
4.2. Postflop model 22
4.2.1. Monte Carlo simulace 22
4.2.2. Realizace postflop modelu 24

5. Testování a výsledky **26**
5.1. Preflop model 26
5.2. Celý pokerbot 28

6. Programová dokumentace **30**

7. Závěr **31**

Reference **33**

A. Výkladový slovník pojmů **34**

B. Uživatelská příručka **35**

C. Obsah přiloženého CD **37**

Seznam obrázků

1. Herní strom čítající přibližně 10^{18} uzlů. Obrázek vychází z obrázku v [6]. 13
2. Část herního stromu sázkového kola preflop. Stejná barva odpovídá stejné informační množině. 15
3. Pseudokód výpočtu očekávané síly ruky E[HS]. 20
4. Monte Carlo simulace – průchod stromem do hloubky. 23
5. Pseudokód Monte Carlo simulace. 23
6. Pseudokód cyklu hry. 31
7. Okno programu po spuštění. 36
8. Okno programu po showdownu. 36

Seznam tabulek

1. Optimální strategie pro prvního hráče. 27
2. Optimální strategie pro druhého hráče po call. 27
3. Optimální strategie pro druhého hráče po raise. 28
4. Výsledek testování preflop modelu proti jednoduchým botům. . . 28
5. Výsledek testování celého pokerbota proti jednoduchým botům a člověku. 29

1. Úvod

Cílem bakalářské práce je naprogramování umělé inteligence pokerového hráče – pokerbota – varianty Texas Hold'em limit (dále jen poker). Vzhledem k neznalosti karet druhého hráče a budoucích společných karet má herní strom přibližně 10^{18} uzlů a není ho možné přímo vyřešit. Musí být zavedeny abstrakce, které ho zmenší. Čím lépe budou zvolené, tím lépe bude pokerbot hrát.

Problematika pokerbotů se řeší různými způsoby. Mezi nejjednodušší z nich patří pokerboti vytvoření na bázi pravidel, které popisují, jak se má pokerbot rozhodovat za daných podmínek. Monte Carlo simulace se snaží najít, která akce má nejvyšší očekávanou hodnotu náhodným průchodem stromu do hloubky. Herně teoretičtí pokerboti hrají pseudo-optimální strategii, kde z dlouhodobého hlediska není výherce ani poražený. Exploitivní pokerboti analyzují hru soupeře a snaží se na ní vydělat. Další způsoby mimo jiné zahrnují líné a evoluční algoritmy.

Selby [1] jako první vytvořil herně teoretického pokerbota pro první kolo hry – preflop. Hraje podle optimální strategie, takže z dlouhodobého hlediska neprohraje. Pokud ale druhý hráč dělá velké chyby, nedokáže jich patřičně využít. Na druhou stranu, jak ukázali Billings a kol. [2], řízená Monte Carlo simulace[1] dynamicky prohledává herní strom a pokud má model druhého hráče, dokáže takových chyb využít.

Obou těchto přístupů využiji k vytvoření pokerbota, který hraje defenzivně v preflop části hry, kdy ještě nemá mnoho informací. V postflop části jsou otočeny nejméně tři společné karty, takže pokerbot může lépe pracovat se sílou své ruky. Často volí akci, která je více agresivnější, než u optimální strategie.

V práci krátce shrnu způsoby vývoje pokerbotů. Poté popíši dvě stěžejní části pokerbota – preflop a postflop model. V obou částech nejdříve popisuji na jakých principech je model postaven a popíši jeho realizaci. Následuje kapitola o testování, kde otestuji preflop model proti jednoduchým botům. Poté testuji celého pokerbota proti jednoduchým botům a lidskému hráči. Výsledky shrnu v závěru práce a porovnám s očekáváním.

[1] V literatuře nazývaná jako *Selective Sampling*.

2. Pravidla hry

Ve hře jsou dva hráči a dealer, který pouze rozdává karty. Hráči začínají se stejným objemem žetonů (stack). Hraje se s klasickým 52 karetním balíčkem. Karty mají 13 různých hodnot: 2 až A (eso). Každá hodnota je ve čtyřech barvách: ♣♠♡♢.

Na začátku hry každý hráč obdrží dvě *privátní karty* známé pouze jemu. Začíná první ze čtyř *sázkových kol*. V každém z nich se hráči střídají v sázkách. Až jsou jejich sázky srovnány, vloží se do společného banku (pot). Pokračuje se dalším sázkovým kolem. Počínaje druhým sázkovým kolem jsou otáčeny *společné karty*. Na konci posledního sázkového kola ukáží hráči privátní karty a spolu se společnými kartami vytvoří *výherní kombinace*. Kombinace hráčů se porovnají. Silnější vyhraje pot.

2.1. Hra podrobně

Na začátku hry je stanovena výše sázek – *small bet* a *big bet*. Small bet je výše sázky používaná v prvních dvou sázkových kolech, big bet ve druhých dvou. Big bet je dvojnásobek small betu. Určí se začínající hráč. Ten se nazývá *small blind*, druhý hráč je *big blind*.

Začíná první sázkové kolo – *preflop*. Oba hráči vloží do hry povinné sázky. Small blind polovinu small betu, big blind celý small bet. Každému hráči jsou rozdány dvě privátní karty. Na tahu je small blind a má k dispozici tři akce:

Fold Složení ruky a nepokračování ve hře. Hráč ztratí vsazený obnos.

Call Dorovnání sázky soupeře.

Raise Dorovnání sázky soupeře a navýšení. Navyšuje se o small bet.

Zvolí-li small blind call, big blind na to může zareagovat akcemi fold, check, raise.

Check Hráč čeká, nevsází. Je k dispozici, pokud hráč nemusí dorovnat soupeře.

Zvolí-li small blind raise, big blind bude mít k dispozici akce fold, call, raise. Takto se hráči v sázkách střídají. Maximální počet sázek na hráče na sázkové kolo je čtyři. Pokud jsou sázky hráčů srovnány a každý z nich se dostal k tahu, sázky jsou vloženy do potu a začíná další sázkové kolo – *flop*.

Jsou otočeny tři společné karty. Začíná big blind s akcemi fold, check, bet.

Bet První sázka. Je k dispozici, pokud v sázkovém kole ještě nikdo nevsadil.

Zvolí-li check, small blind bude mít k dispozici stejné akce: fold, check, bet. Zvolí-li bet, small blind bude mít k dispozici fold, call, raise. Podmínky pro přechod do dalšího sázkového kola jsou stejné jako v sázkovém kole preflop.

Další následují sázkové kola *turn* a *river*. V obou probíhá sázení stejně jako v sázkovém kole flop s tím rozdílem, že je na začátku otočena jedna společná karta a velikost sázek se rovná big betu.

Na konci sázkového kola river, mají-li oba hráči srovnáno, nastává *showdown*. Sázky se vloží do potu, oba hráči otočí privátní karty a odhalí, jakou mají výherní kombinaci. Je tvořena zkombinováním privátních a společných karet. Výherní kombinace hráčů se porovnají a kdo má silnější vyhrává pot.

Výherní kombinace od nejsilnější:

Royal flush Postupka 10 až A v barvě.

Straight flush Libovolná postupka pěti karet v barvě.

Poker Čtveřice karet stejné hodnoty.

Full House Trojice karet a pár karet stejné hodnoty.

Flush Libovolných pět karet v barvě.

Straight Libovolná postupka pěti karet.

Trojice Trojice karet stejné hodnoty.

Dva páry Dva páry karet stejné hodnoty.

Pár Dvojice karet stejné hodnoty

Nejvyšší karta O výhře rozhoduje hodnota karty: 2 je nejslabší, A je nejsilnější.

Pokud má výherní kombinace méně než pět karet, je doplněna kartami co nejvyšší hodnoty. Říká se jim *kicker* a rozhodují při rovnosti výherních kombinací. Pokud oba hráči mají stále stejně silné kombinace, hra končí remízou. Hráči si pot rozdělí.

Eso se bere jako nejvyšší karta. Při skládání postupek je možné ho použít i jako kartu s hodnotou jedna.

Pokud hráč nemá dostatečný obnos na dorovnání sázky, má možnost zvolit *all-in* – vložit všechny zbylé žetony do potu. Hráč pak může vyhrát pouze tolik žetonů, kolik vsadil celkem.

3. Přehled přístupů tvorby pokerbotů

Kapitola čerpá z přehledového článku [8].

Znalostně orientovaní pokerboti
Pokerboti jsou vytvořeni na základě znalosti pokeru. Spadají sem dvě kategorie.

Expertní pravidlové systémy
Pokerbot je řízen sadou if-then pravidel. Ty specifikují, jakou pravděpodobnostní trojici má vygenerovat za podmínek, ve kterých se nachází. Náhodně z ní pak zvolí akci. Pravděpodobnostní trojice popisuje v jakém procentu případů má pokerbot provést fold, call/check a bet/raise.

Rovnicové metody
Více obecný přístup, kde jsou vstupem zásadní informace o aktuálním stavu hry. Výstup je pravděpodobnostní trojice vypočítána rovnicí. Na jejím základě pokerbot provede náhodnou akci. Typicky je vstupem numerická reprezentace síly ruky a šance na výhru potu – pot odds.

Monte Carlo simulace
Pro akce check/call, bet/raise, Monte Carlo simulace prochází herní strom z aktuálního stavu až do listu a uloží si jeho výplatu. Takových průchodů provede nejméně několik set. Nalezené výplaty zprůměruje a získa tak očekávanou hodnotu obou akcí. Zpravidla pak provede tu s vyšší očekávanou hodnotou (kap. 4.2.1.). Monte Carlo simulace původně volí průchod stromem náhodně. Je možné ho ale řídit a vybírat větve, které se nejpravděpodobněji stanou.

Teorie her – Nashova rovnováha
Cíl teorie her v pokeru je dosáhnutí Nashovy rovnováhy. Strategie jsou v rovnováze, pokud změna strategie znamená zhoršení očekávané hodnoty. Každý hráč tak hraje optimální strategii. Protože je ale výplata jednoho hráče rovna ztrátě druhého hráče, výdělek se při hraní optimálních strategii bude z dlouhodobého hlediska blížit k nule. I když optimální strategie není primárně výdělečná, v pokeru se používá proti neznámému soupeři na snížení ztrát. K nalezení optimální strategie se používají techniky lineárního programování (kap. 4.1.1.).

Exploitivní protistrategie
Zaměřují se na využívání slabin druhého hráče. Pozorují jak hraje a vytváří si model. Exploitují jeho chyby za účelem většího zisku, než u Nashovy rovnováhy. Nevýhodou je, že mohou být samy exploitovány. Rozlišují se postupy adaptivní a statické.

Herní strom s neúplnou informací
Ve hrách s úplnou informací se používá Minimax. Ve hrách, kde je šance na provedení nějaké události (např. rozdání karet, hod kostkou) se používá Expectimax. V pokeru jsou ale zásadní karty, které drží druhý hráč. Určují jeho strategii. Proto je třeba vylepšit Expectimax o model druhého hráče. Efektivita algoritmu

je přímo úměrná přesnosti modelu. Vyvinuté algoritmy se označují jako Miximax, který volí akci s nejvyšším EV nebo Miximix, který někdy volí i akci s nižším EV, aby nebyl tak předvídatelný.

Model obsahuje dvě informace:

- rozložení pravděpodobnosti akcí druhého hráče v každém uzlu stromu
- pravděpodobnost na výhru při showdownu v listových uzlech

Pokud by minimax prohledal hru až k listům, koresponduje s Nashovou rovnováhou.

Nejčastější nejlepší odpověď
Je exploitivní strategie, která vychází z teorie her. Vytvoří offline model druhého hráče pozorováním tréningových her. Předpokládá, že má druhý hráč statickou strategii. Vytvořená strategie je pak exploitivní, ale ne adaptivní.

Skládá se ze třech částí:

Úvod Určí se výchozí chování modelu (standardně vždy call) a zvolí se abstrakce hry.

Tréning K vytvoření přesného modelu druhého hráče je potřeba pozorovat spoustu tréningových her, kde je k dispozici úplná informace – karty druhého hráče. Dochází k mapování zpozorovaných ruk zahraných v reálné hře do frekvencí ve zvolené abstrakci. Čím více je zpozorováno her, tím je model přesnější.

Nejlepší odpověď Jakmile je vytvořen model druhého hráče, je vypočítána nejlepší odpověď. To zahrnuje navštívení každé informační množiny v abstrakci a zvolení strategie, která bude maximalizovat očekávanou hodnotu proti strategii druhého hráče.

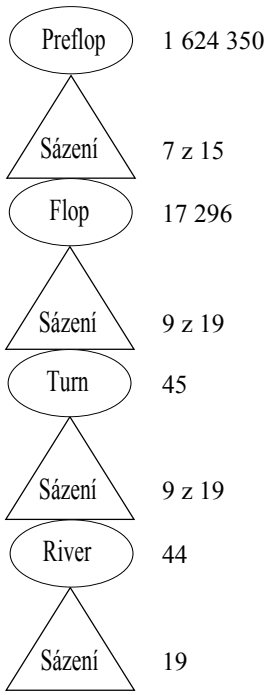
Alternativní Přístupy

Zvažování případů
Jde o algoritmy líného učení. Udržuje se databáze případů, se kterými se pokerbot setkal a jejich řešení. Je-li potřeba vyřešit novou situaci, je vytvořen nový případ, který je porovnán s již vyřešenými. Použije se řešení na bázi nejpodobnějších případů. Případ popisuje dříve zahranou ruku, řešení a výdělek.

Evoluční algoritmy a neuronové sítě
Evoluční algoritmy byly použity k automatickému vývinu pokerbotů. Princip spočívá ve vyvíjení populace z generace na generaci. Členové populace jsou hodnoceni kondiční funkcí. Čím vyšší hodnocení, tím vyšší šance na postup do další generace. V té vyprodukují potomky křížením. Různé populace jsou drženy odděleně. Mohou mezi sebou soutěžit, avšak evoluce je povolena pouze uvnitř jejich vlastní populace.

Bayesian poker

Poslední přístup těží z Bayesianské sítě. Jde o orientovaný acyklický graf, kde uzly reprezentují náhodné proměnné. Hrany mezi nimi představují vztahy závislosti. Ke každému uzlu náleží podmíněná pravděpodobnostní tabulka. Pravděpodobnostní hodnoty jsou vypočítány na základě hodnot v rodičovských uzlích. Přiřažením různých hodnot uzlům sítě může dojít k propagaci pravděpodobnosti, což bude mít za následek rozložení pravděpodobnosti přes náhodné proměnné.



Obrázek 1. Herní strom čítající přibližně 10^{18} uzlů. Obrázek vychází z obrázku v [6].

4. Tvorba pokerbota

Vzhledem k počtu uzlů herního stromu (obr. 1.) jsem problém rozdělil na dvě části. Preflop obsahující první sázkové kolo hry. Postflop obsahující flop, turn a river. Ke každé části jsem vytvořil model, podle kterého pokerbot hraje. Rozlišuji tedy preflop a postflop model.

V preflop modelu se zabývám nalezením optimální statické strategie a dosáhnutí tzv. Nashovy rovnováhy. V postflop modelu dynamicky prohledávám herní strom Monte Carlo simulací. Zjišťuji, jaká akce má nejlepší očekávanou hodnotu.

4.1. Preflop model

Hra v preflop modelu začíná rozdáním karet a vložením povinných sázek. Poté hráči mohou sázet až do limitu čtyři sázky na hráče a hra je utnuta. V kapitole se budu zabývat nejdříve teorii her a lineárním programováním. Následovat bude reprezentace modelu a jeho realizace.

4.1.1. Teorie her

Teorie her obecně zkoumá chování hráčů nějaké hry, kde každý hráč má množinu strategií a snaží se pomocí ní maximalizovat svoji výplatu[3]. Strategie hráči říká, jakou akci má zvolit v každém uzlu hry. Pokud se žádnému hráči nevyplatí změnit aktuální strategii, pak strategie obou hráčů je optimální a hra je v *Nashově rovnováze*. V kapitole se budu zabývat konkrétní problematikou teorie her v pokeru. V preflop části hry začíná small blind, proto prvním hráčem budu myslet jeho. Druhým hráčem bude big blind.

Poker patří do kategorie her, které charakterizují tři hlavní složky:

Nulový součet Pro libovolnou dvojici strategií, kde každá patří jednomu hráči, je součet výplat roven nule. Do výplaty výherce se nezapočítává jeho sázka. Tu pouze dostane zpět. Výplata výherce je rovna sázce poraženého. Poražený svou sázku ztrácí.

Neúplná informace Jeden hráč nezná karty druhého hráče.

Dokonalá paměť Hráč si pamatuje všechny akce, které byly doposud ve hře provedeny.

Strategie jsou rozděleny na dva typy:

Ryzí Říká hráči, jakou akci má udělat v každém uzlu hry.

Smíšená Odpovídá rozdělení pravděpodobnosti mezi množinu všech ryzích strategií.

Cílem je nalézt *optimální strategii*. Nalezení optimální strategie odpovídá optimálnímu rozdělení pravděpodobnosti ve smíšené strategii. Optimální strategie jednoho hráče je nezávislá na stregii druhého hráče. Je nezávislá, protože počítá s nejhorší odpovědí druhého hráče. Jinými slovy, pokud zvolím strategii x, pak počítám s tím, že můj druhý hráč zvolí strategii y tak, aby mi co nejvíce uškodil.

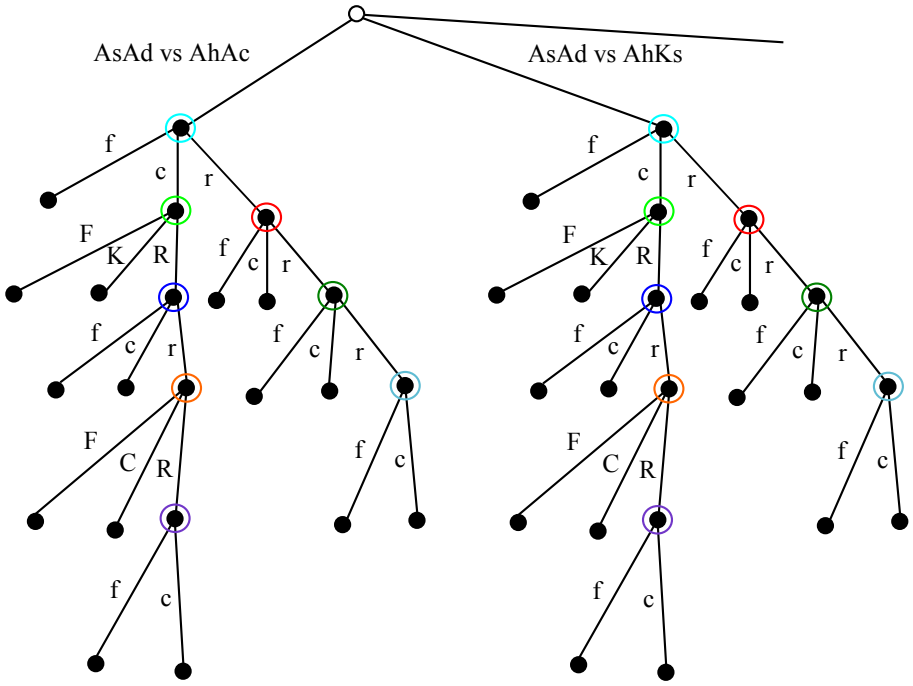
Důsledek hraní optimální strategie je, že pokud druhý hráč také nezačne hrát optimální strategii, pak první hráč může vydělat pouze více, než předpokládal. Protože je poker hra s nulovým součtem, jejich výdělek by se z dlouhodobého hlediska blížil nule. Je ale nutné herní strom zjednodušit, a tak vznikne odchylka.

4.1.2. Reprezentace hry

Základní reprezentace pokeru je pomocí herního stromu. Ten lze převést na normální formu zobrazující hru v matici a nebo sekvenční formu, která je úspornější verzí normální formy.

Herní strom[2] je složený z uzlů, hran, a *informačních množin*. Uzly znázorňují každý možný stav hry a jsou rozděleny na tři typy:

[2]V literatuře často označovaný jako extenzivní forma



Obrázek 2. Část herního stromu sázkového kola preflop. Stejná barva odpovídá stejné informační množině.

Rozhodovací Patří vždy jednomu z hráčů. Hrany vedoucí z rozhodovacího uzlu odpovídají konkrétní akci, mezi kterými hráč volí.

Šance Rozhoduje zde náhoda a patří nehrajícímu účasníkovi hry – dealerovi. Mezi hrany vedoucí z uzlu šance je rozdělena pravděpodobnost s jakou se konkrétní hrana zvolí. Součet pravděpodobností všech hran je roven jedné. V pokeru uzel šance odpovídá rozdání privátních karet hráčům a vygenerování společných karet.

List Nepatří žádnému hráči. Obsahuje výplatu. V pokeru je to výplata hráče small blind[3].

Protože je poker hra neúplných informací, seskupují se rozhodovací uzly do informačních množin. Hráč nedokáže rozlišit mezi uzly v informační množině. Zná pouze své karty a historii hran, které informační množině předcházely. Uzly v informační množině pak obsahují všechny možné karty druhého hráče (obr. 2.)

Normální forma je matice, kde každý řádek odpovídá ryzí strategii jednoho hráče a sloupec ryzí strategii druhého. Položka matice je výplatou prvního hráče při zvolení dvojice strategií na daných indexech.

[3]Výplata hráče big blind je výplata hráče small blind vynásobená (-1).

Nevýhoda normální formy spočívá v exponenciální velikosti matice k velikosti herního stromu. *Velikost stromu* je počet jeho listů. Jinými slovy, je to počet všech možných průchodů stromu, které skončí v listovém uzlu. Z toho důvodu se používá pouze u velmi zjednodušených variant pokeru, jako je například tříkaretní poker.

Nevýhodu normální formy odstraňuje *sekvenční forma*. Její velikost je pouze lineární k velikosti stromu. Hlavní složkou je výplatní matice A z pohledu prvního hráče. Dalšími složkami jsou matice E a vektor e, které reprezentují lineární omezení pro prvního hráče. Podobně F a f pro druhého hráče.

Každý řádek matice A odpovídá *sekvenci* prvního hráče a sloupec sekvenci druhého hráče. Zvolím-li libovolný uzel stromu, pak existuje cesta z kořene, která do něj vede. Sekvence prvního hráče, je posloupnost hran na této cestě. Obdobně pro druhého hráče.

Množinu všech sekvencí prvního a druhého hráče označím Q_1 a Q_2. Pravděpodobnost, s jakou se konkrétní sekvence provede, se označuje jako *váha realizace*. V tomto kontextu je smíšená strategie x prvního hráče určení vah realizací v Q_1. Váhy realizace musí splňovat lineární omezení daná E a e, které hlídají, aby se váhy realizací správně rozdělovaly.

Vektor x reprezentuje smíšenou strategii prvního hráče pouze tehdy a jenom tehdy pokud $E \cdot x = e$. Obdobně y reprezentuje smíšenou strategii pro druhého hráče za podmínek $F \cdot y = f$.

4.1.3. Lineární programování obecně

Lineární programování je matematická disciplína hledající optimální řešení zadané úlohy. Definice v kapitole vycházejí z [4]. V kapitole nejdříve budou definovány základní pojmy z lineárního programování. Poté bude představena Simplexová metoda, která je použita na nalezení optimální strategie pro preflop model.

Definice 1. *Úloha lineárního programování v základním tvaru* je nalezení vektoru x, který úloha minimalizuje nebo maximalizuje ve skalárním součinu $c \cdot x$ za podmínek $A \cdot x \leq b$, $x \geq 0$, kde A je matice úlohy, b je vektor pravých stran a c je účelový vektor.

Úloha vypadá následovně:

$$\begin{array}{ll} \text{Maximalizuj} & c \cdot x \\ x & \\ \text{omezeno} & A \cdot x \leq b \\ & x \geq 0 \end{array} \tag{1}$$

Takto popsaná úloha se také nazývá *primární*.

Definice 2. *Duální úloha* k primání úloze je úloha ve tvaru:

$$\begin{array}{ll} \text{Minimalizuj} & b \cdot y \\ y & \\ \text{omezeno} & y \cdot A \geq c \\ & y \geq 0 \end{array} \tag{2}$$

Složky vektoru y jsou tzv. duální proměnné.

Věta 1. *Silná věta o dualitě* říká, že pokud x je řešením primární úlohy, pak existuje y tak, že $c \cdot x = b \cdot y$.

Věta 1 je důležitá pro sestavení lineární úlohy pro pokerbota.

Definice 3. Úloha lineárního programování je v tzv. *kanonickém tvaru*, pokud:

$$\begin{array}{ll} \text{Maximalizuj} & c \cdot x \\ x & \\ \text{omezeno} & A \cdot x = b \\ & x \geq 0 \end{array} \tag{3}$$

a platí, že všechny řádky matice A jsou lineárně nezávislé.

Přípustné řešení lineární úlohy je vektor (x, z) takový, který splňuje všechna omezení úlohy, přičemž vektor z může obsahovat záporné hodnoty.

Simplexová metoda je algoritmus, který postupnými úpravami pracovní matice hledá optimální řešení. Funguje na následujícím principu:

1. Pokus se převést vstupní úlohu do kanonického tvaru a zapiš ji do pracovní matice S:

$$S = (s_{ij}) = \left[\begin{array}{cc|c} 1 & c & 0 \\ 0 & A & b \end{array}\right]$$

   Pokud je řešení přípustné, pokračuj bodem 2., jinak chyba.

2. Zkontroluj, jestli jsou všechny hodnoty v nultém řádku nekladné. Pokud ano, vrať optimální řešení, jinak pokračuj bodem 3.

3. Najdi index l sloupec s nejvyšší kladnou hodnotou v nultém řádku za podmínky, že l nemůže být index sloupce b.

4. Najdi index k řádku s nejmenším podílem b_k/s_{kl} pro $k > 0$, $s_{k,l} > 0$.

5. Transformuj matici S podle prvku s_{kl} a pokračuj bodem 2.

Po skončení algoritmu je optimální řešení ve sloupci b.

Variace simplexové metody, tzv. *Dvoufázová simplexová metoda*, pak řeší problém možné nepřípustnosti řešení v prvním bodě algoritmu. V první fázi je na základě původní úlohy vytvořena pomocná úloha, kde simplexová metoda nalezne optimální řešení. Tím je získáno základní přípustné řešení pro druhou fázi algoritmu. Simplexová metoda zde nalezne opět optimální řešení a to už je hledané optimum původní úlohy.

4.1.4. Lineární programování v pokeru

V pokeru je úlohou lineárního programování nalézt optimální strategie pro prvního a druhého hráče. V kontextu lineráního programování označím prvního hráče MAX a druhého hráče MIN. Níže uvedené lineární úlohy vycházejí z [5].

MIN chce hrát dokonalou strategii proti fixní smíšené strategii MAXe. Tzn. chce na strategii MAXe co nejvíce vydělat. MAX si to uvědomuje a chce proto minimalizovat hodnotu, kterou MINovi odevzdá. Toho docílí, když místo fixní strategie bude hrát optimální strategii.

Lineární úloha hráče MAX:

$$\begin{array}{ll} \text{Minimalizuj} & t^T \cdot f \\ x, t & \\ \text{omezeno} & x^T \cdot A + t^T \cdot F \geq 0 \\ & x^T \cdot E^T = e^T \\ & x \geq 0 \end{array} \tag{4}$$

Vektor t má velikost rovnou počtu řádků matice F a vznikl převedením primární úlohy na duální. Řádek $t^T f$ reprezentuje účelovou funkci, kterou je potřeba minimalizovat.

MAX chce hrát dokonalou strategii proti fixní smíšené strategii MINa. Tzn. chce na strategii MINa co nejvíce vydělat. MIN si to uvědomuje a chce proto minimalizovat hodnotu, kterou MAXovi odevzdá. Toho docílí, když místo fixní strategie bude hrát optimální strategii.

Lineární úloha hráče MIN:

$$\begin{array}{ll} \text{Minimalizuj} & e^T \cdot r \\ y, r & \\ \text{omezeno} & -A \cdot y + E^T \cdot r \geq 0 \\ & F \cdot y = f \\ & y \geq 0 \end{array} \tag{5}$$

Vektor r má velikost rovnou počtu řádků matice E a vznikl převedením primární úlohy na duální. Řádek $e^T r$ reprezentuje účelovou funkci, kterou je potřeba minimalizovat.

Takto sestavené lineární úlohy jsou vyřešeny pomocí simplexové metody.

4.1.5. Realizace preflop modelu

Vytvoření preflop modelu jsem rozdělil do třech kroků. *Abstrakce* herního stromu, sekvenční forma a lineární programování. Princip je následovný: nad herním stromem zavedu abstrakce a redukuji tak počet uzlů a hran. Poté strom převedu do sekvenční formy, která je vhodná na aplikování simplexové metody. Ta najde optimální strategii.

Abstrakce
Význam abstrakce spočívá ve vypuštění informací, které lze postrádat. Abstrahovaná hra je vyřešena a výsledek použit v původní hře. Abstrakce je dvojího typu. Bezezstrátová a ztrátová.

Bezeztrátová zaručuje, že její aplikace nijak neovlivní původní hru. U *ztrátové* mají vynechané informace dopad na strategii hráče. Čím silnější ztrátová abstrakce, tím neoptimálniji bude pokerbot hrát. Samotné rozdělení hry na dva modely je velmi silná abstrakce.

Abstrakce spočívá v sjednocení všech možných privátních ruk do *tříd ekvivalencí*. Tříd ekvivalencí je 169 a ztotožňují privátní karty hráče podle jejich hodnoty a zda-li jsou v barvě (suited) nebo ne (offsuited).

První hráč může na začátku hry dostat $\binom{52}{2} = 1326$ kombinací privátních karet. Druhý hráč $\binom{50}{2} = 1225$. Celkový počet kombinací privátních karet je tedy $\binom{52}{2}\binom{50}{2} = 1624350$. Abstrakce redukuje výsledný počet privátních karet na $169 \times 169 = 28561$.

Třídy ekvivalencí jsou děleny do třech základních kategorií:

Pár Karty stejné hodnoty. Celkem jich je 13: 22, 33 až AA

Suited Karty v barvě. Konkrétní barva není důležitá. Celkový počet je 78: 23s, 24s až AKs (s jako suited)

Offsuited Karty mimo barvu. Celkem 78: 23o, 24o až AKo (o jako offsuited)

Na základě výše popsané abstrakce rekurzivně vygeneruji herní strom, který bude mít $28561 \times 23 + 1 = 656904$ uzlů, kde 23 je počet uzlů podstromu každé dvojice abstrahovaných privátních ruk. Přičtená 1 znázorňuje kořen hry. Množinu všech informačních množin značím IM_1 pro prvního hráče a IM_2 pro druhého hráče. Každý hráč má ve zmíněném podstrom čtyři informační množiny. Celkem informačních množin na hráče i je $|IM_i| = 169 \times 4 + 1 = 677$.

Označím-li hrany podle akcí, které reprezenují: f fold, c call, k check, b bet, r raise, pak existuje 15 možných cest z kořene podstromu do listu:

```
f, cF, cK, cRf, cRc, cRrF, cRrC, cRrRf, cRrRc,
rF, rC, rRf, rRc, rRrF, rRrC
```

```
foreach board in boards
  e1 = Evaluate(hand1 + board)
  e2 = Evaluate(hand2 + board)
  if(e1 > e2)
    win++
  else if(e1 == e2)
    tie++
  else
    lose++

E[HS] = (win + tie / 2) / (win + tie + lose)
```

Obrázek 3. Pseudokód výpočtu očekávané síly ruky E[HS].

velikost písmena odlišuje hranu prvního a druhého hráče.

Výplatu P listu l vypočítám jako: $P(l) = E[HS] \times pot - bets$. Kde l je list, E[HS] je pravděpodobnost na výhru privátní ruky prvního hráče proti privátní ruce druhého. Pot je velikost potu a $bets$ je výše sázek prvního hráče. Pokud listu l předcházela hrana fold, je výplata $P(l) = -bets$ pokud fold zvolil první hráč nebo $P(l) = pot - bets$ pokud fold zvolil druhý hráč.

E[HS] neboli *očekávaná síla ruky* je procentuální vyčíslení síly ruky. Prochází všechny možné kombinace společných karet a porovnává, jestli by vyhrál první nebo druhý hráč (obr. 3.). Celkový počet kombinací společných karet je $\binom{48}{5} = 1712304$.

Funkce Evaluate přiřadí privátní ruce a společným kartám číslo znázorňující hodnotu ruky. Toto číslo reprezentuje výherní kombinaci spolu s kickery a je ho možné porovnávat s dalšími ohodnocenými rukami. Použil jsem `Hand Evaluator` vytvořený panem Keith Rule a vygeneroval E[HS] pro všechny kombinace abstrahovaných privátních ruk. Výpočet trval přibližně deset hodin.

Sekvenční forma

Nejdříve popíši tvorbu výplatní matice A, následně tvorbu matic a vektorů omezení.

Sekvence hráčů je možné vyčíst z cest podstromu. Pro prvního hráče jsou to:

```
f, c, cf, cc, cr, crf, crc
r, rf, rc, rr
```

Pro druhého hráče pak:

```
c: F, K, R, RF, RC, RR
r: F, C, R, RF, RC
```

Na začátku řádku je uvedena hrana prvního hráče, která předcházela hraně druhého hráče. Je ji potřeba uvést kvůli jednoznačné identifikaci sekvence. Každý hráč má 11 sekvencí. Celkový počet sekvencí hráče i je $|Q_i| = 169 \times 11 + 1 = 1860$.

První je vytvořena výplatní matice A velikosti $Q_1 \times Q_2$, tedy 1860×1860. Vytvořím ji rekurzivním procházením kořenových informačních množin z IM_1. Kořenová informační množina je taková, které nepředchází žádná jiná informační množina vyjma kořenu hry.

Nechť l je libovolný list herního stromu, h_1 je privátní ruka prvního hráče, h_2 je privátní ruka druhého hráče. Dále i_1 je index h_1, i_2 je index h_2, j_1 je index sekvence prvního hráče na cestě k l a j_2 je index sekvence druhého hráče na cestě k l.
Index řádku r do matice A je vypočítán jako $r = 169 \times i_1 + j_1 + 1$.
Index sloupce s do matice A je vypočítán jako $s = 169 \times i_2 + j_2 + 1$.
Konečně položka matice $A_{rs} = c_{h_1 h_2} \times P(l)$, kde $c_{h_1 h_2}$ je pravděpodobnost, že prvnímu hráči bude rozdána h_1 a druhému hráči h_2.
Pokud položka matice neodpovídá listu, je její hodnota rovna 0.

Pravděpodobnost p rozdání konkrétního páru pro jednoho hráče je

$$p = \frac{\binom{4}{2}}{\binom{52}{2}},$$

konkrétní suited ruky

$$p = \frac{\binom{4}{1}}{\binom{52}{2}},$$

a offsuited

$$p = \frac{\binom{4}{1}\binom{3}{1}}{\binom{52}{2}}.$$

Příklad 1. *Jeden hráč má AA, druhý hráč má KK. Pravděpodobnost $c_{h_1 h_2}$ je vypočítána jako:*

$$c_{h_1 h_2} = \frac{\binom{4}{2}\binom{4}{2}}{\binom{52}{2}\binom{50}{2}} = \frac{36}{1624350},$$

protože oba hráči vybírají dvě barvy ze čtyř.

Příklad 2. *Jeden hráč má AA, druhý hráč má AA. Pravděpodobnost $c_{h_1 h_2}$ je vypočítána jako:*

$$c_{h_1 h_2} = \frac{\binom{4}{2}\binom{2}{2}}{\binom{52}{2}\binom{50}{2}} = \frac{6}{1624350},$$

protože jestliže jeden hráč má A♣A♠, druhý hráč vybírá pouze ze dvou dostupných barev - A♡A♢.

Matice omezení E pro prvního hráče má velikost $IM_1 \times Q_1$, konkrétně 677×1860. Vytvořil jsem ji iterativním procházením informačních množin z IM_1.

Položka E_{00} odpovídá kořenu hry a $E_{00} = 1$. Pro všechny ostatní informační množiny $im_1 \in IM_1$ pak do matice přiřadím -1 tam, kde je sekvence předcházející im_1 a 1 tam, kde je sekvence vzniklá zřetězením předcházející sekvence a hranou vedoucí z im_1. Ostatní položky jsou rovny 0.

Vektor e je typu $1 \times IM_1$ a kromě $e_0 = 1$ obsahuje samé nuly. Stejnou technikou vytvořím matici F a vektor f pro druhého hráče.

Výsledné úlohy (4) a (5) převedené do matice, jsou typu 2538×2538. Úlohu jsem uložil do formátu MPS a na výpočet optimální strategie jsem použil program Wolfram Mathematica.

4.2. Postflop model

Hra v postflop modelu začíná flopem a končí showdownem. Pokerbot se nesnaží o optimální strategii. Rozhoduje se pouze podle toho, jaká akce má nejvyšší očekávanou hodnotu. V kapitole popíši teorii okolo Monte Carlo simulace. Následovat bude realizace modelu. Postflop model vychází z [7].

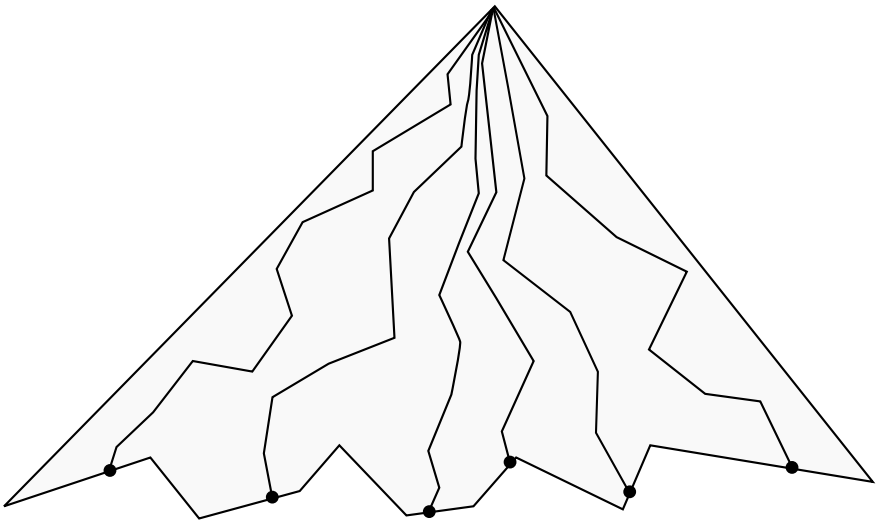
4.2.1. Monte Carlo simulace

Herní strom není možné celý projít do šířky, jako to dělá algoritmus Minimax. Je možné ho ale projít do hloubky (obr. 4.). Monte Carlo simulace (dále jen Monte Carlo) prochází herní strom z aktuální pozice do listu a ukládá si jeho výplatu. Takový průchod se nazývá *trial* a provede se vždy pro hrany check/call a bet/raise. Monte Carlo spustí trialů několik set až tisíc, zprůměruje získané výplaty a získá tak očekávanou hodnotu hrany check/call a bet/raise. Fold má vždy nulovou očekávanou hodnotu, protože nepředpokládá žádné další sázky. Po provedení všech trialů Monte Carlo zvolí hranu s nejvyšší očekávanou hodnotou.

Úmluva. Dále v textu budu označovat hrany check/call a bet/raise jednotně jako call a raise.

Simulace procházení herního stromu může být *řízená* nebo *náhodná*. Řízená simulace přiřadí druhému hráči privátní karty na základě modelu, který o něm má. Když za druhého hráče musí zvolit hranu, vygeneruje pravděpodobnostní trojici {fold, call, raise} a náhodně hranu vybere. Čím přesnější model druhého hráče, tím přesnější bude očekávaná hodnota simulované hrany.
Náhodný přístup model druhého hráče nemá.

Příklad 3. Podle modelu druhý hráč často zvolí fold na riveru, pokud musí dorovnat sázku. Proto Monte Carlo vygeneruje pravděpodobnostní trojici, kde nejvyšší šanci na provedení bude mít hrana fold. Získá tak přesnější očekávanou hodnotu pro call a raise.



Obrázek 4. Monte Carlo simulace – průchod stromem do hloubky.

```
callEV = raiseEV = trial = 0
while trial < MAX_TRIALS
  AssignHand()
  callEV += SimulateHand(Call)
  raiseEV += SimulateHand(Raise)
  trial++

return{ 0, callEV/trial, raiseEV/trial}
```

Obrázek 5. Pseudokód Monte Carlo simulace.

4.2.2. Realizace postflop modelu

Monte Carlo provádí 10 000 trialů. V každém přiřazuje druhému hráči karty a volí za něj i za sebe hrany. Aby byla simulace úspěšná, je tedy potřeba vyřešit tři věci:

1. model druhého hráče,
2. přiřazení privátních karet,
3. rozhodování sama za sebe.

Monte Carlo před prvním trialem generuje náhodné číslo v rozmezí od 0 do 1. Když pak musí volit náhodně akci z pravděpodobnostní trojice, řídí se tímto číslem. Je tak zajištěna konzistentní agresivita druhého hráče.

V Monte Carlo používám jednoduchý *model druhého hráče*. Je to trojrozměrná statistická tabulka, kde první rozměr představuje tři sázková kola: flop, turn, river. Druhý rozměr představuje kolik sázek musí druhý hráč dorovnat: 0, 1. Třetí rozměr odpovídá hranám fold, call, raise.

Statistická tabulka se aktualizuje vždy, když druhý hráč zvolí hranu. Čím více her Monte Carlo s druhým hráčem odehraje, tím přesněji dokáže simulovat jeho hrany.

Příklad 4. Aktuální sázkové kolo: river. Počet sázek k dorovnání: 1.

$$Statistics(river, 1) = \{6000,\ 3000,\ 1000\}$$

Monte Carlo vygeneruje pravděpodobnostní trojici $\{0.6,\ 0.3,\ 0.1\}$ a na jejím zakladě náhodně zvolí hranu.

Na začátku trialu musí Monte Carlo *přiřadit druhému hráči ruku*. Vygeneruje všechny zbývající kombinace privátních ruk. Spolu se společnými kartami je seřadí podle síly výherní kombinace. Zjistí, jak často volí druhý hráč fold. Takové procento nejslabších ruk z nich odstraní. Ze zbývajících ruk přiřazuje ruku druhému hráči náhodně.

Kombinace privátních ruk jsou: na flopu $\binom{47}{2} = 1081$, na turnu $\binom{46}{2} = 1035$, na riveru $\binom{45}{2} = 990$.

Když se Monte Carlo musí rozhodovat *sám za sebe*, zvolí vždy hranu call. Průchod stromem je tak řízený a simulace se vyhne hranám fold.

Simulace v sobě přirozeně zahrnuje informaci o síle i potenciálu ruky.

Síla ruky Jaké procento ze všech možných ruk druhého hráče dokáže ruka porazit.

Potenciál ruky Do jaké míry se může ruka zlepšit nebo zhoršit s dalšími společnými kartami.

Při simulacích jsou náhodně odhalovány zbývající společné karty. Pokud žádný z hráčů nezvolil fold, jsou na konci simulace ruky porovnány.

Výběr hrany s nejlepší očekávanou hodnotou je ovlivněn dvěma vyjímkami.

1. Pokud Monte Carlo určí fold jako hranu s nejlepší očekávanou hodnotou a v kole nebylo navýšeno, pak zvolí check.

2. Pokud jsou očekávané hodnoty hran call a raise kladné a jejích rozdíl je malý, zvolí jednu z hran náhodně.

Druhý bod má za cíl zmást lidského hráče. Řídí se náhodným číslem, které se vygeneruje vždy na začátku hry. Určí míru agresivity Monte Carla.

5. Testování a výsledky

V první části popisuji testování preflop modelu proti jednoduchým pokerbotům. V druhé části otestuji celého pokerbota proti jednoduchým pokerbotům a proti živému hráči.

U pokerbota bude zkoumána výsledná bilance po testování, která odpovídá počtu small betů na odehranou ruku. Je vypočítáná vzorcem

$$b = \frac{v}{h \cdot sb},$$

kde v je celkový výdělek, h je počet odehraných ruk[4] a sb je výše small betu.

Ve všech testováních je $sb = 2$.

5.1. Preflop model

Kapitola obsahuje popis tabulek strategií a testování modelu. Znaky znamenají: f fold, k check, c call, r raise, b bet. Znaky za sebou pak jsou zřetězením akcí ve smyslu: rr raise a pokud se opět dostanu k tahu tak raise, stejně tak rc je raise call, rf je raise fold.

V tabulce 1. je výsledek výpočtu optimální strategie pro prvního hráče. Položky odpovídají sekvencím prvního hráče. Obsahuje 168 ryzích strategií: 13 rr, 66 rc a 89 f. Jediná smíšená strategie je $x1 = 0.64115f, 0.35885rc$.

Tabulky pro druhého hráče jsou dvě kvůli přehlednosti. V první je optimální strategie za předpokladu, že první hráč zvolil hranu call. V druhé je optimální strategie pokud první hráč zvolil hranu raise.

Tabulka 2. má 166 ryzích strategií: 10 f, 130 k, 6 rf, 21 rr a tři smíšené strategie:

$$y1 = \{0.47040rf, 0.52960rr\},$$

$$y2 = \{0.44107k, 0.55893rr\},$$

$$y3 = \{0.44638f, 0.55362k\}.$$

Tabulka 3. má 168 ryzích strategií: 139 c, 29 rc. Jediná smíšená strategie $z1 = 0.74262c, 0.25738rc$.

Pro testování optimální strategie jsem vytvořit tři jednoduché boty. *RaiseBot* vždy zvolí hranu raise nebo bet. *RandomBot* vždy zvolí hranu náhodně podle pravděpodobnosti {1/3, 1/3, 1/3}. *CallBot* vždy zvolí hranu call nebo check.

[4]Jinými slovy, jde o počet odehraných her začínajících rozdáním karet a končící foldem jednoho z hráčů nebo showdownem.

Tabulka 1. Optimální strategie pro prvního hráče.

		Suited												
		A	K	Q	J	T	9	8	7	6	5	4	3	2
Offsuited	A	rr	rr	rr	rr	rc	rc	rc	rc	rc	rc	rc	rc	rc
	K	rr	rr	rc	rc	rc	rc	rc	rc	rc	rc	rc	rc	rc
	Q	rr	rc	rr	rc	rc	rc	rc	rc	rc	rc	rc	f	f
	J	rr	rc	rc	rr	rc	rc	rc	rc	f	f	f	f	f
	T	rc	rc	rc	rc	rr	rc	f	f	f	f	f	f	f
	9	rc	rc	rc	rc	rc	rr	f	f	f	f	f	f	f
	8	rc	rc	rc	rc	f	f	rr	f	f	f	f	f	f
	7	rc	rc	rc	f	f	f	f	rc	f	f	f	f	f
	6	rc	rc	f	f	f	f	f	f	rc	f	f	f	f
	5	rc	rc	f	f	f	f	f	f	f	rc	f	f	f
	4	rc	rc	f	f	f	f	f	f	f	f	rc	f	f
	3	rc	x1	f	f	f	f	f	f	f	f	f	rc	f
	2	rc	f	f	f	f	f	f	f	f	f	f	f	f

Tabulka 2. Optimální strategie pro druhého hráče po call.

		Suited												
		A	K	Q	J	T	9	8	7	6	5	4	3	2
Offsuited	A	rr	k	k	k	k	k	k	k	k	k	k	k	k
	K	rr	rr	k	k	k	k	k	k	k	k	k	k	rf
	Q	rr	rr	rr	k	k	k	k	k	rf	rf	k	k	k
	J	rr	rr	rr	rr	rf	k	rf	k	k	k	k	k	k
	T	rr	rr	rr	k	rr	k	k	k	k	k	k	k	k
	9	rr	k	k	k	k	rr	k	k	k	k	k	f	f
	8	rr	k	k	k	k	k	rr	k	k	k	k	f	f
	7	k	k	k	k	k	k	k	rr	k	k	k	f	f
	6	k	k	k	k	k	k	k	k	y1	k	k	f	f
	5	k	k	k	k	k	k	k	k	k	rr	k	k	k
	4	k	k	k	k	k	k	k	k	k	k	rf	y3	f
	3	y2	k	k	k	k	k	k	k	k	k	k	k	f
	2	k	k	k	k	k	k	k	k	k	k	k	k	k

Tabulka 3. Optimální strategie pro druhého hráče po raise.

		Suited												
		A	K	Q	J	T	9	8	7	6	5	4	3	2
Offsuited	A	rc	rc	rc	rc	rc	rc	rc	rc	rc	rc	c	c	c
	K	rr	rc	rc	rc	rc	c	c	c	c	c	c	c	c
	Q	rr	rc	rc	c	c	c	c	c	c	c	c	c	c
	J	rr	z1	rr	rc	rf	rf	c	c	c	c	c	c	c
	T	rc	c	rr	c	rc	c	c	c	c	c	c	c	c
	9	rc	c	c	c	c	rc	c	c	c	c	c	c	c
	8	rc	c	c	c	c	c	rc	c	c	c	c	c	c
	7	c	c	c	c	c	c	c	rc	c	c	c	c	c
	6	c	c	c	c	c	c	c	c	rc	c	c	c	c
	5	c	c	c	c	c	c	c	c	c	rc	c	c	c
	4	c	c	c	c	c	c	c	c	c	c	c	c	c
	3	c	c	c	c	c	c	c	c	c	c	c	c	c
	2	c	c	c	c	c	c	c	c	c	c	c	c	c

Tabulka 4. Výsledek testování preflop modelu proti jednoduchým botům.

Bot	**Odehranáno**	**Bilance**
RaiseBot	5000000	+0.04051
RandomBot	5000000	+0.13224
CallBot	5000000	−0.04931

Pokud má některý z jednoduchých botů méně než tři hrany na výběr, vybere tu nejblíže původní strategii (e.g. RaiseBot může volit jen mezi fold a call, zvolí call).

Výsledek testovaní v (tab. 4.).

5.2. Celý pokerbot

Stejně jako v předchozí kapitole bude pokerbot testován proti jednoduchým botům. Poté přijde na řadu testování proti živému hráči.

Odehraných ruk je 4000, čas jednoho testu je přibližně 40 minut (tab. 5.).

Pokerbot byl testován proti příležitostnému lidskému hráči. Bylo odehráno 110 ruk. Záznam je uložený v souboru `hvsp.txt`. Kvůli demonstraci soubor obsa-

Tabulka 5. Výsledek testování celého pokerbota proti jednoduchým botům a člověku.

Bot	**Odehranáno**	**Bilance**
RaiseBot	4000	+1.3715
RandomBot	4000	+0.8270
CallBot	4000	+0.7135
Člověk	110	+1.6136

huje očekávané hodnoty vygenerované Monte Carlo simulací a ukazuje zahozené karty. Vybral jsem tři zásadní ruky (značení p pokerbot, h lidský hráč).

12. ruka Pokerbot je small blind. Preflop: p.call, h.check. Flop: 9♣9♡7♠, h.check, p.check. Turn: 2♠, h.bet, p.raise, h.raise, p.call. River: 7♡, h.bet, p.raise, h.raise, p.raise, h.call. Pokerbot ukazuje: 8♢7♣, lidský hráč: 4♣J♡. Pokerbot má fullhouse a vyhrává.

34. ruka Lidský hráč je small blind. Preflop: h.raise, p.call. Flop: Q♣J♡5♢, p.bet, h.raise, p.raise, h.raise, p.call. Turn: A♡, p.bet, h.raise, p.raise, h.call. River: 8♠, p.bet, h.call. Pokerbot ukazuje: K♡T♢, lidský hráč: 9♣A♣. Pokerbotovi došla na turn postupka. Vyhrává.

96. ruka Lidský hráč je small blind. Preflop: h.call, p.check. Flop: A♢A♡K♣, p.check, h.check. Turn: 3♢, p.check, h.check. River: J♡, p.check, h.bet, p.fold. Pokerbot měl: 2♢9♣, lidský hráč: 4♣5♢. Pokerbot zahazuje, lidský hráč vyhrává. Poker bot se nechal zastrašit.

6. Programová dokumentace

Program je vytvořen objektově v jazyce C#. Verze použitého Microsoft .NET framework je 4.0. Program má čtyři hlavní třídy: *Manager.cs*, *NashBot.cs*, *MonteCarlo.cs*, *UI.cs*.

Manager.cs
Řídí hru. Kompletní informace o stavu hry – herní balíček, kdo je na tahu, sázky hráčů, apod. Důležité metody jsou `NewGame()`, `DoAction(Action action)`, `NextRound()`, `NextPlayer()`.

NashBot.cs
Vytvoří celý preflop model (kap. 4.1.5.). Výstupem je dvojice souborů `MaxPreflopStrat.mps` a `MinPreflopStrat.mps` představující úlohy lineárního programování. Vyřeší je program Wolfram Mathematica.

Metoda `GenerateTree()` vygeneruje herní strom. Metoda `GeneratePayoff()` vygeneruje výplatní matici A v sekvenční formě. Metoda `GenerateConstrains()` vytvoří matice omezení pro oba hráče. Nakonec metoda `NashEquilibrium()` vytvoří výstupní úlohy a uloží je do formátu `.MPS`. Metoda `GenNashAction()` vrátí akci na základě optimální strategie.

MonteCarlo.cs
Implementuje algoritmus Monte Carlo simulace (kap. 4.2.1.). Metoda `SimulateActions()` spouští Monte Carlo simulaci a vrací akci s nejvyšší očekávanou hodnotou. V metodě `Simulate()` je algoritmus Monte Carlo simulace. Metoda `SimulateHand(Manager.Action action)` simuluje akci z aktualní pozice až do konce hry.

UI.cs
Uživatelské rozhraní konzolové aplikace. Metoda `GameLoop()` spouští cyklus hry. Pseudokód:

Program zahrnuje pomocné třídy *Node.cs*, *InformationSet.cs*, *Player.cs*, *Loader.cs* a *Program.cs*.

Program také obsahuje projekt *Simplex*, který obsahuje projekt simplexové metody. V pokerbotovy nebyl použit kvůli nedostatečnému výkonu.

```
do
  NewGame()
  do
    do
      if !player[onPlay].IBot
        a = GetAction()
        DoAction(a)
      else
        if round == preflop
          a = GenNashAction()
          DoAction(a)
        else
          a = SimulateActions()
          DoAction(a)
    while NextPlayer()
    NextRound()
  while (round != showdown) AND (won == nobody)
while !GameEnded()
```

Obrázek 6. Pseudokód cyklu hry.

7. Závěr

Výpočet optimální strategie pomocí simplexové metody se úkázal být problémový. Implementovaná dvojfázová metoda byla schopna řešit úlohy jen do několika stovek řádků. Proto byl pro výpočet použit program Wolfram Mathematica a optimalizovaná Revidovaná simplexová metoda. Výpočet jednotlivých strategií trval přibližně čtyři až šest hodin.

Srovnám-li výsledné optimální strategie s optimálními strategiemi Selbyho, nevypadají příliš podle očekávání. Ze strategie prvního hráče (tab. 1.) lze vyčíst, že příliš ruk zahazuje. Na druhou stranu se snaží na silných rukách co nejvíce vydělat, což je správně. U strategie druhého hráče po callu prvního hráče je překvapující, že zahazuje několik ruk, i když nemusí dorovnat sázku (tab. 2.). Naproti tomu strategie druhého hráče po raisu prvního hráče vypadá podle očekávání. Strategie nezahodí ani jednu ruku (tab. 3.).
Jak zmínil Selby ve své práci [1], optimální strategie může být degenerativní. To znamená, že existuje více než jedno optimální řešení. Simplexová metoda hledá řešení pomocí dělění a tak vzniká chyba v zaokrouhlení výsledku. Zaokrouhlení na 8 desetiných míst jsem také musel provést pro uložení úlohy do formátu MPS. Jak mnoho se kvůli tomu odchýlila strategie od optima je těžké určit.
Z výsledku testování proti jednoduchým botům jde vidět, že optimální strategie proti CallBotovi ztráci (tab. 4.).

Celému pokerbotovi nedělalo problém porazit jednoduché boty, protože si o nich dokázal udělat přesný model. Analýzou vybraných ruk z her proti živému hráči (kap. 5.2.) lze učinit tyto závěry:

1. Díky prvku náhody mezi výběrem hran s podobnou očekávanou hodnotou pokerbot zvolil check místo raise. Lidský hráč si myslel, že nic nemá a navýšil. Pokerbot ho však dorovnal. Této technice se v pokeru říká slowplay.

2. Pokerbot klade důraz na budoucí společné karty.

3. I přesto, že nemá silné karty, volí check. Lidský hráč si pak nemůže být jistý, jestli nejde o slowplay.

Vzhledem k variaci v pokeru je 110 odehraných ruk nepatrné množství. Lze však na něm úkázat vlastnosti pokerbota. Díky modelu druhého hráče pokerbot získává přesnější očekávané hodnoty a volí tak odpovídající akce.

Výsledný pokerbot neblafuje se slabou rukou a tak pro silného hráče nebude těžké ho odhadnout. Slabším soupeřům může byt ale zdatným protivníkem.

Reference

[1] A. Selby, *Optimal heads-up preflop poker*, http://www.archduke.demon.co.uk/simplex, 1999.

[2] D. Billings, L. Pena, J. Schaeffer, D. Szafron, *Using Probabilistic Knowledge and Simulation to Play Poker.* AAAI Press / The MIT Press, strany 697-703, 1999.

[3] E. Rasmusen, *Games and Information: An Introduction to Game Theory*, Wiley-Blackwell (4th edition), 2006.

[4] P. Hliněný, *Optimalizační úlohy*, Masarykova univerzita, 2007.

[5] R. Andersson, *Pseudo-Optimal Strategies in No-Limit Poker*, Umea University, 2006.

[6] D. Billings, N. Burch, A. Davidson, R.C. Holte, J. Schaeffer, T. Schauenberg, D. Szafron, *Approximating game-theoretic optimal strategies for full-scale poker*, IJCAI-03, 2003

[7] L.P. Castillo, *Probabilities and simulations in poker*, Master's thesis, University of Alberta, 1999.

[8] J. Rubin, I. Watson, *Computer poker: A review*, University of Auckland, 2011

A. Výkladový slovník pojmů

All-in Vsazení všech žetonů

Big bet Velká sázka. Je rovna dvojnásobku small betu

Big blind Musí na začátku hry vložit povinnou sázku velikosti small betu. V postflopu začíná

Bet První sázka v daném kole

Blaf Sázení se slabou rukou

Call Dorovnání sázky

Flop Druhé sázkové kolo. Jsou otočeny tři společné karty

Fold Zahození ruky. Hráč se vzdává vsazených žetonů

Check Čekání. Hráč nic nedělá

Kicker Karta, která doplní výherní kombinaci. Rozhoduje při rovnosti výherních kombinací

Offsuited Karty mimo barvu

Pot Sázky z minulých sázkových kol

Postflop Označuje sázková kola flop, turn, river

Preflop První sázkové kolo

Raise Navýšení sázek

River Čtvrté, poslední sázkové kolo. Je otočena pátá společná karta

Showdown Odhalení výherních kombinací a určení, který hráč vyhrál pot

Small bet Malá sázka. V této práci je rovna dvěma žetonům

Small blind Musí na začátku hry vložit povinnou sázku velikosti poloviny small betu. V preflopu začíná

Stack Objem žetonů hráče

Suited Karty ve stejné barvě

Turn Třetí sázkové kolo

B. Uživatelská příručka

Okno po spuštění hry (obr. 7.). Vždy po spuštění začíná pokerbot. Popis prostředí:

1. Číslo hrané ruky. Sázkové kolo: preflop. Aktivní hráč je player 1 (pokerbot). Společné karty nejsou otočené žádné a pot je 0.

2. Kdo je small blind / big blind. Stack hráčů. Výše sázek v sázkovém kole. Zobáček ukazuje, kdo je na tahu.

3. Akce, kterou provedl pokerbot.

4. Stejné informace jako v 1), ale je na tahu hráč.

5. Ze stacků ubyly sázky a přidaly se do řádku bet. Zobáček signalizuje změnu hráče na tahu.

6. Ruka hráče. Zkratky pro barvy jsou: s = ♠, c = ♣, h = ♡, d = ♢. Další řádek říká, kolik je potřeba žetonů na call a raise. Volby hráče. Je možné je zapsat buď zvýrazněným písmenem a nebo celým slovem – pro call buď *c* nebo *call*.

Okno po showdownu (obr. 8.).

1. Poslední akce pokerbota. Je vidět otočené společné karty a výši potu.

2. Poslední akce hráče. Zvolil *c* jako check.

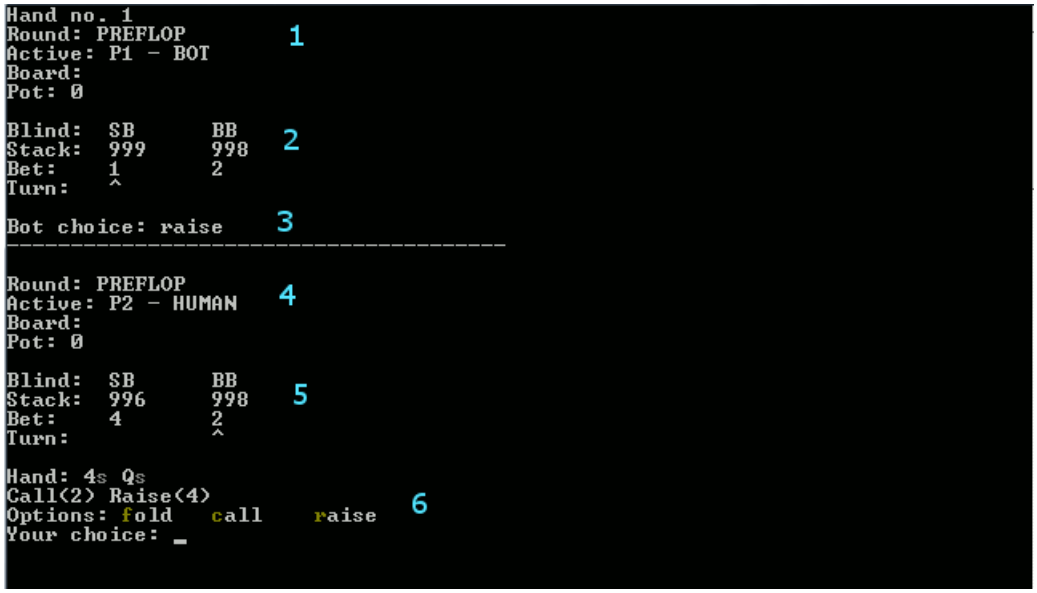
3. Výsledek hry. Jsou otočeny karty pokerbota. Výpis informace o výherních kombinacích. Finální stacky po přidělení potu vítězi.

Soubor `Init.txt` předpokládá dvě hodnoty. Na prvním řádku je to číslo od 100 do 100 000 – reprezentuje startovní stack. Na druhém řádku se předpokládá 0 nebo 1. Při 1 se zaznamená průbeh hry do souboru `GameLog.txt`. Zápis se provede po zavření programu, ale data se ukládají v průběhu hry. S volbou 1 je hra pomalejší. Jiný než uvedený vstup není přípustný. V souborech `HandCount.txt` se předpokládá pouze jedno kladné číslo.

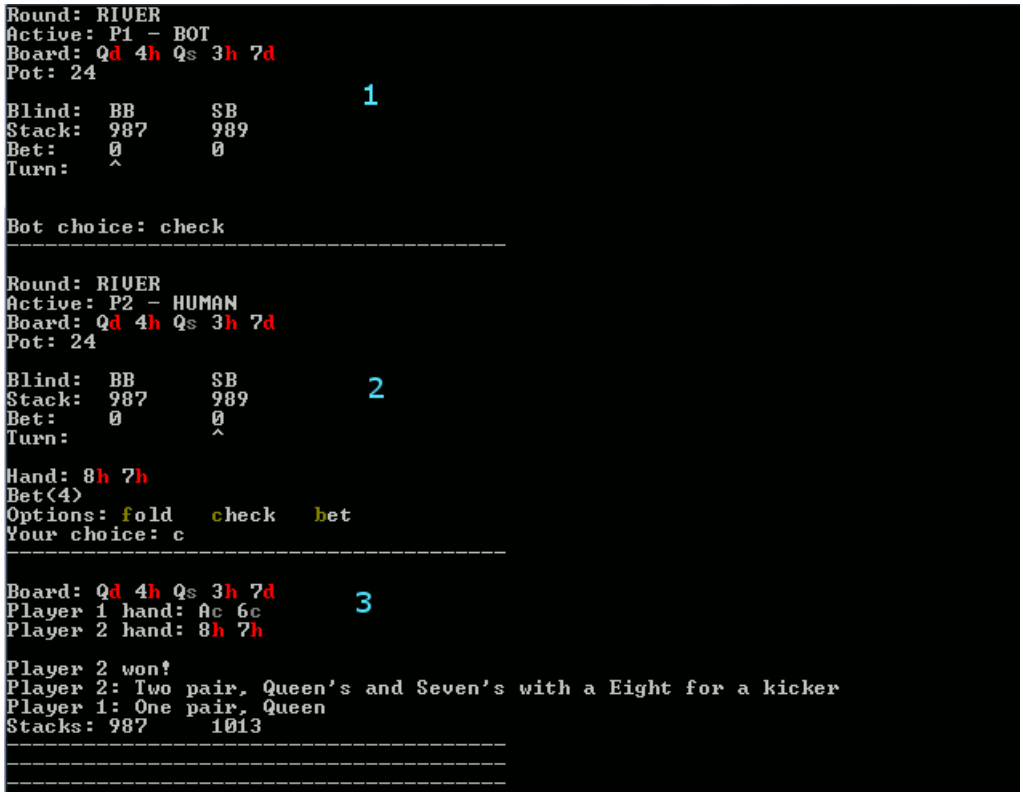
Do `GameLog.txt` se ukládá i vektor očekávaných hodnot, mezi kterými se pokerbot rozhoduje po Monte Carlo simulaci. Je to trojice hodnot před volbou pokerbotova tahu. Ukládá se také pokerbotova privátní ruka po akci fold.

Upozornění: logovani je možné aktivovat jen pokud program může zapisovat na medium kde byl spuštěn.

Hra se sama od sebe zastavuje, aby se zvýšila přehlednost při hraní. Je potřeba vždy zmáčknout klávesu `Enter` pro pokračování.



Obrázek 7. Okno programu po spuštění.



Obrázek 8. Okno programu po showdownu.

C. Obsah přiloženého CD

`bin/`
: Program `TexasHoldemLimit.exe`, strategie `MaxPreflopStrat.txt` a `MinPreflopStrat.txt`, inicializační soubor `Init.txt` a knihovna `HandEval.dll`.

`doc/`
: Soubor `vytrisalbc.pdf` a archiv `vytrisal.zip` se zdrojovými soubory pro sestavení PDF.

`src/`
: Archiv `TexasHoldemLimit.zip` s veškerými zdrojovými soubory.

`readme.txt`
: Informace pro spuštění programu.

Navíc CD obsahuje:

`data/`
: `format.txt` obsahuje informace pro práci s `HandCount.txt` soubory ve složkách Preflop a Fullgame. Složka Preflop obsahuje tři složky jednoduchý botů pro testování preflop modelu. Složka Fullgame obsahuje tři složky jednoduchých botů pro testování celého pokerbota a složku Human se souborem `hvsp.txt`. Na něm je uložen záznam her proti lidskému hráči.

`install/`
: Instalátor .NET Framework 4.0.