Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Analýza stratégie pre kartovú hru No Thanks

Diplomová práca

2021
Bc. Filip Husár

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

Analýza stratégie pre kartovú hru No Thanks

Diplomová práca

Študijný program:	Informatika
Študijný odbor:	Informatika
Školiace pracovisko:	Katedra informatiky
Školiteľ:	RNDr. Michal Forišek, PhD.

Bratislava, 2021
Bc. Filip Husár

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Filip Husár
Študijný program: informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: informatika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: slovenský
Sekundárny jazyk: anglický

Názov: Analýza stratégie pre kartovú hru No Thanks
Analysis of Strategies for the Card Game No Thanks

Anotácia: V diplomovej práci autor analyzuje kartovú hru No Thanks. Ide o kombinatorickú hru s neúplnou informáciou. Cieľom analýzy je čo najviac sa priblížiť k optimálnej stratégii pre túto hru. Súčasťou analýzy je riešenie jednoduchších problémov smerujúcich k riešeniu pôvodnej hry, ako napríklad riešenie verzie s úplnou informáciou alebo riešenie menších inštancií pôvodnej hry. Ďalším z aspektov analýzy je rozbor možností pre stratégie protihráčov a zistenie, či a ako od predpokladov o nej závisí naša optimálna stratégia.

Cieľ: Ciele práce sú nasledovné: Formálne definovať kartovú hru No Thanks ako kombinatorickú hru viacerých hráčov. Navrhnúť a implementovať algoritmus pre riešenie upravenej hry s úplnou informáciou o budúcnosti. Navrhnúť a implementovať algoritmus pre riešenie pôvodnej hry pre menší (avšak v rámci možností čo najväčší) počet kariet vhodným prehľadávaním stavového priestoru. Vo vyššie uvedených implementáciách skúšať rôzne možnosti pre stratégie protihráčov a analyzovať ich. Pokúsiť sa zovšeobecniť získané poznatky na implementáciu čo najlepšej stratégie pre pôvodnú hru.

Vedúci: RNDr. Michal Forišek, PhD.
Katedra: FMFI.KI - Katedra informatiky
Vedúci katedry: prof. RNDr. Martin Škoviera, PhD.

Dátum zadania: 03.12.2019

Dátum schválenia: 27.04.2022

prof. RNDr. Rastislav Kráľovič, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

Poďakovanie:
Ďakujem Mišofovi za skvelú spoluprácu a cenné rady v pozícii školiteľa.
Taktiež ďakujem mojej rodine a priateľke Zuzke za poskytnutú podporu pri riešení nemalých problémov ktoré táto práca priniesla.

Abstrakt

V tejto diplomovej práci analyzujeme kartovú hru pre troch hráčov *No Thanks*. K naučeniu počítača hrať hru je možné pristupovať rôznymi spôsobmi ako *mini-max* alebo z oblasti umelej inteligencie *alpha-zero*. Oba spôsoby sú však špecifické pre hry dvoch hráčov.

V tejto práci sa pozrieme na článok popisujúci zovšeobecnený algoritmus *alfa-beta orezávania* pre n hráčov *maxn orezávanie* ktorý predpokladá horné ohraničenie súčtu bodov. Ukážeme, že tento algoritmus nie je použiteľný v hrách kde má pre hráča konečné poradie ako počet získaných bodov. Navrhneme algoritmus, ktorý tento problém rieši a zároveň nepotrebuje predpoklad horného ohraničenia súčtu bodov.

V práci ďalej vyslovíme heuristiku o prehľadávaní nerovnomerných stromových štruktúr s *alfa-n orezávaním* a ukážeme merania hovoriace ktoré túto heuristiku podporujú.

Postupnou konštrukciou stratégií a ich porovnávaní sa budeme snažiť čo najviac priblížiť optimálnemu hraniu.

Kľúčové slová: mini-max, alfa-beta orezávanie, maxn orezávanie, k-d stromy, herná stratégia, ováhovanie parametrov

Abstract

In this master thesis we analyse 3-player card game *No Thanks*. Teaching computer how to play game well can be obtained by several popular methods like *mini-max* or AI algorithm *alpha-zero*. Nevertheless both approaches are typical for 2-player games.

We take a look at article describing generalized algorithm *alpha-beta pruning* for n-player games, *maxn pruning*. This approach assumes upper bound on the sum of the evaluations for each player. We show that this algorithm is not useful when players prefer better position rather than points. We design algorithm which solves this problem and moreover doesn't need any upper bounds.

Then we describe heuristics about searching highly unbalanced trees and show measurements to prove our estimate.

We gradually construct strategies and compare them to make the best approximation of optimal playing.

Keywords: mini-max, alpha-beta pruning, maxn pruning, k-d trees, game strategy, weighting of parameters

Obsah

Úvod **1**

1 O hre No Thanks **3**
1.1 Pravidlá hry . . . 3
1.2 O hre . . . 5
1.3 Zjednodušené verzie hry . . . 8
1.3.1 Verzia hry so známou budúcnosťou . . . 9
1.3.2 Verzia hry s neobmedzeným počtom žetónov . . . 10
1.3.3 Verzia bez vyradených kariet . . . 11

2 Definície a pojmy **13**
2.1 Minimax . . . 15
2.1.1 Aplikácia minimaxu pri 3 a viac hráčoch . . . 16
2.2 Orezávanie . . . 17
2.3 Orezávanie pri hre s 3 a viac hráčmi . . . 18
2.4 k-d stromy . . . 19
2.4.1 Popis k-d stromu . . . 20
2.4.2 Vyhľadávanie v k-d strome . . . 21

3 Náš prínos v oblasti **23**
3.1 Orezávanie v hre s 3 a viac hráčmi bez horného ohraničenia súčtu bodov 23
3.2 Orezávanie v hrách s opakovaným ťahom hráča . . . 26
3.3 Orezávanie v nerovnomernom strome . . . 27
3.4 Ako zmazať rozdiely medzi simuláciou a realitou . . . 29

4 Vytváranie stratégie **31**
4.1 Parametre hry v datasetoch . . . 32
4.2 Generovanie dát pre úseky hry . . . 33
4.2.1 Veľkosti úsekov . . . 34
4.2.2 Počet hier pre generovanie datasetov úsekov . . . 34
4.3 Váhy generovaných stratégií . . . 35

4.4 Porovnanie dvoch stratégií 36
4.4.1 Spôsob porovnania 37
4.4.2 Počet hier na porovnanie stratégií 37
4.4.3 Striktný výber z množiny stratégií 37
4.5 Gradientný zostup 39

5 Dosiahnuté výsledky **41**
5.1 Porovnanie Take Card vs Pay Chip 41
5.2 Porovnanie rôznych stratégií tretieho hráča 42
5.3 Vplyv nezaznamenaných častí stavu hry 43
5.4 Hľadanie najlepšieho ováhovania parametrov 44

Záver **49**

Zoznam obrázkov

1.1 V priebehu hry si hráči rozdeľujú svoje získané karty na súvislé úseky oddelené medzerami. 4
1.2 Ukážka ako môže vyzerať záverečné bodovanie 5
1.3 Simulácia hry s optimálnym hraním všetkých hráčov 9

2.1 Ukážka využitia pravidiel minimaxu v praxi 16
2.2 Ukážka Max^3 17
2.3 Ukážka Max^3 s horným ohraničením 9 (prevzaté z [1]) 18
2.4 Ukážka zlyhania hlbokého orezávania pri hre 3 a viac hráčov (prevzaté z [1]) 19
2.5 Využitie k-d stromu pri dvojrozmerných dátach 20
2.6 Prvky v k-d strome 21
2.7 Vyhľadávanie najbližšieho prvku v k-d strome, detail 22

3.1 Ukážka zlyhania orezávania z článku R.korfa pre potreby hry No Thanks 24
3.2 Orezávanie stromu na základe pozície hráča 25
3.3 Orezávanie stromu pri viacerých ťahoch hráča za sebou 27
3.4 Ukážka orezávania v nevyváženom strome 28
3.5 Porovnanie pôvodného a upraveného vyhodnocovania simulácie 29

4.1 Rozdelenie hry na viacero úsekov 31

5.1 Postupné menenie váh parametrov 46
5.2 Postupné menenie váh parametrov podľa malých datasetov 46
5.3 Postupné menenie váh parametrov podľa malých datasetov s menším ϵ 47

Úvod

Hranie ťahových hier s automatizovaným súperom je problém siahajúci až do 60.rokov, kedy boli prvé pokusy na vývoj počítačových programov na hranie šachu. Počítače boli v tejto dobe hardvérovo obmedzené na riešenie tohto typu problémov, preto sa až v neskorých 80.rokoch podarilo zostrojiť stroje ktoré by boli schopné poraziť profesionálneho hráča šachu. V roku 1987 bol vytvorený superpočítač *ChipTest* na hranie šachu ktorý bol neskôr nahradený vylepšeným nástupcom *Deep Thought*. V roku 1989 sa Deep Thought postavil proti šachovej legende Garrymu Kasparovi, no nevyhral ani jednu z dvojice hier. Práce na superpočítači pokračovali až sa v roku 1996 vďaka tretiemu počítaču zo série *Deep Blue* prvýkrát podarilo vyhrať nad majstrom sveta Garrym Kasparovom. Z nasledovných piatich hier však dve skončili remízou a tri víťazstvom Kasparova. O rok neskôr, v máji 1997 však Deep Blue porazil Kasparova v súčte šiestich zápasov $3^1/_2$: $2^1/_2$.

Deep Blue bol založený na analýze tisícok hier šachových majstrov a ohodnocovaním rôznych typických šachových pozícií. Taktiež obsahoval databázu všetkých konečných pozícii s 5 alebo menej figúrkami a mnoho pozícií so 6 figúrkami. Využíval alfa-beta prehľadávací algoritmus a paralelný výpočet.

Paralelne so šachovým počítačovým svetom boli vyvíjané programy v inej tradičnej hre, Go. Podobne aj tu sa podarilo vytvoriť program ktorý porazil ľudského majstra. V októbri 2015 sa podarilo počítaču AlphaGo poraziť európskeho šampióna Fan Hui 5:0.

Šach ako i Go sú hry pre dvoch hráčov. Výsledok hry sa dá vyjadriť bodom na x-ovej osi. Hodnota 0 znamená remízu, záporné hodnoty znamenajú výhru jedného hráča a kladné hodnoty výhru druhého hráča. Samotná hodnota hovorí aj o tom, či hráč vyhral tesne alebo jednoznačne. To môže byť užitočné v priebehu hry, aby hráč uprednostňoval tie pozície, v ktorých vyhráva s väčším rozdielom.

Hra No Thanks je avšak iná od Šachu alebo Go. V prvom rade ide o hru pre troch a viac hráčov, preto výsledok hry nevieme zaznamenať ako bod na x-ovej osi. Alfa-beta orezávanie použité v Deep Blue je určené pre hru dvoch hráčov, avšak Richard Korf v roku 1989 [1] popísal v článku variant orezávania pre ľubovoľný počet hráčov. V článku však predpokladá horné ohraničenie súčtu počtu bodov a počtu bodov hráča dáva väčšiu prioritu ako konečnému umiestneniu.

V našej práci navrhneme alternatívny algoritmus ktorý nepotrebuje horné ohraničenie bodov a prioritou hráčov je ich konečné poradie. Počet získaných bodov je pre nich až sekundárny atribút.

Špecifikum hry No Thanks z výpočtovej stránky je to, že jedna z dvojice akcií hráča na ťahu výrazne skracuje hru oproti druhej, vďaka čomu je prehľadávaný strom nevyvážený, teda na jednej strane oveľa plytší ako na druhej. To nás priviedlo k otázke, či nemá väčší význam najprv prezerať plytšiu stranu stromu a vzdať sa ideálu prezerať optimálne hranie ako prvé, k čomu je však veľakrát ťažké sa priblížiť.

Pre vytvorenie najlepšej možnej stratégie sme sa inšpirovali technikou použitou v Deep Blue a vytvorili sme si databázu mnohých konečných pozícií. Myšlienku sme ale rozšírili a z databázy konečných pozícii sme postupne vygenerovali niekoľko ďalších databáz prislúchajúcich k rôznym časovým úsekom hry. Rôzne aspekty hry sme zaznamenali do parametrov ktorými popisujeme stav hry a snažili sme sa zistiť dôležitosť každého z parametrov a priradiť mu patričnú váhu.

Následne sme stratégie s rôznymi váhami medzi sebou porovnávali a zisťovali sme tak, ktoré parametre majú akú dôležitosť.

Kapitola 1

O hre No Thanks

No Thanks je kartová hra v ktorej hráči zbierajú karty z verejnej ponuky v strede stola do svojej zbierky. Nazbierané karty znamenajú pre hráčov negatívne body, preto treba dobre odhadnúť ten správny moment kedy kartu odmietnuť a kedy ju prijať. Hra je navrhnutá pre troch až siedmich hráčov, ale pre potreby tejto práce sa budeme zaoberať iba verziou s tromi hráčmi.

Hra pozostáva z dvoch hlavných herných komponentov - kariet a žetónov. Kariet je dokopy 33, každá karta je jedinečná a na každej z nich sa nachádza prirodzené číslo z intervalu $[3, 35]$. Toto číslo udáva počet negatívnych bodov. Pri hre 3 hráčov má každý hráč na začiatku hry 11 žetónov. Žetóny slúžia hráčom na to, aby mohli odmietnuť kartu. Hlavne v prípade, keď dáva veľa negatívnych bodov. Ak hráč nemá žiaden žetón alebo nechce žiaden žetón zaplatiť, kartu si musí zobrať.

1.1 Pravidlá hry

Balíček kariet je na začiatku hry zamiešaný a 9 kariet je vyradených z hry. Ktoré karty sú vyradené nie je známa informácia. Z balíčka budú postupne prichádzať karty do hry. Otočí sa karta a hráči sa striedajú na ťahu. Po tom, čo si kartu niektorý hráč vezme, sa otočí ďalšia karta. V momente keď niektorý z hráčov zoberie kartu a balíček zostane prázdny, hra končí a nasleduje záverečné vyhodnotenie. Pred zahájením hry ešte každý hráč obdrží 11 žetónov. Žetóny sú podľa pôvodných pravidiel hry skryté, avšak všetky ťahy hráčov tejto hre sú verejné (protihráči vedia ako hral hráč na ťahu), takže môžeme predpokladať že každý hráč si vie počítať počet žetónov jednotlivých hráčov. Aj keď ľudský hráč možno nezvládne držať v hlave stav žetónov každého jeho protihráča, je vhodné aby počítač ktorý túto hru bude hrať predpokladal, že jeho súperi hrajú čo najlepšie možne, ako opísal John von Neumann v článku o technike minimax. [2]

Náhodne sa určí začínajúci hráč, otočí vrchnú kartu z balíčka a dostáva sa na ťah.

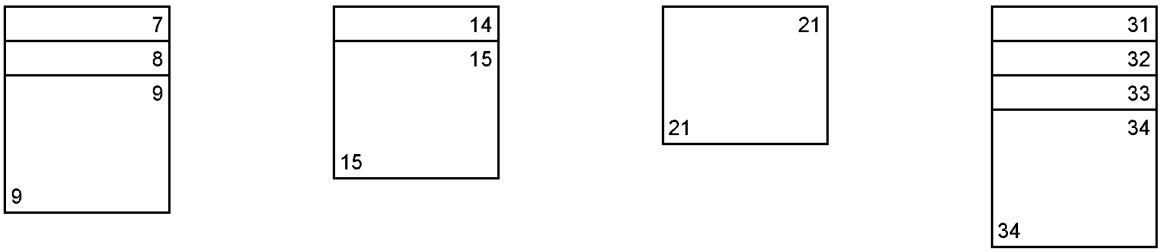
Hráč na ťahu má dve možnosti:

1. *Nie ďakujem*: Hráč odmietne kartu zobrať a k danej karte pridá jeden žetón zo svojej zásoby. Pokiaľ hráč nemá vo svojej zásobe žiadne žetóny, je povinný použiť akciu *Áno prosím* popísanú nižšie. Po tejto akcii sa dostáva na ťah ďalší hráč v poradí.

2. *Áno prosím*: Hráč si zoberie otočenú kartu a zaradí ju medzi svoje získané karty. Spolu s kartou získa hráč aj všetky žetóny ktoré sa odmietaním tejto karty k nej nazbierali. Hráč otočí novú kartu z balíčku a opäť je na ťahu. Získané karty zostávajú otočené a viditeľné pre všetkých.

Odmietnutie karty akciou *Nie ďakujem* spôsobuje, že sa na ťah dostane hráč po ľavici. Karta je otočená až pokým ju niektorý z hráčov nezoberie akciou *Áno prosím*, preto je možné a časté, že karta s vyšším číslom bude odmietnutá tými istými hráčmi veľakrát, až si ju napokon niektorý z tých čo ju predtým odmietli zoberie. Napríklad preto, lebo nazbieraný počet žetónov môže byť pre neho výhodný.

V momente kedy niektorý z hráčov zoberie kartu a už v balíčku žiadna nezostala, končí hra a nasleduje záverečné vyhodnotenie. Hráči si spočítajú žetóny ktoré majú vo svojej zásobe. Svoje získané karty zoradia podľa hodnôt od najmenšej po najväčšiu a dvojice susedných kariet medzi ktorými je rozdiel hodnôt väčší ako jeden oddelia medzerou.



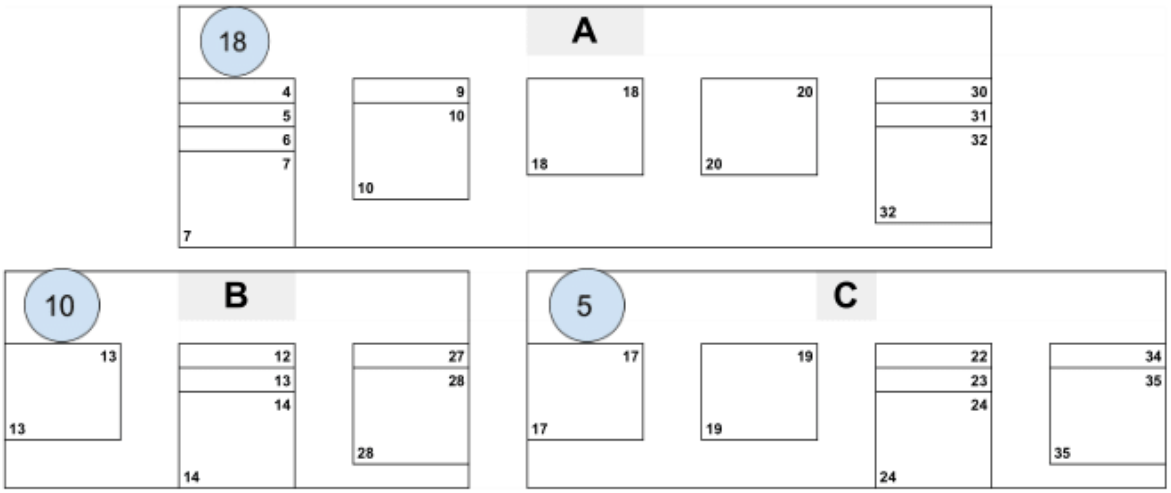
Obr. 1.1: V priebehu hry si hráči rozdeľujú svoje získané karty na súvislé úseky oddelené medzerami.

Každú kartu, ktorá je v niektorom súvislom úseku najmenšia, nazveme vedúcou kartou. Všetky zvyšné karty ktoré nie sú vedúce hráčovi neprinášajú žiadne body. Hráči si teraz spočítajú počet bodov ktoré získali.

Nech $c_1, c_2, ..., c_a$ sú vedúce karty hráča A kde a je ich počet. Počet žetónov hráča A označme ako t_A. Počet bodov P_A hráča A vypočítame ako:

$$P_A = t_A - \sum_{i=1}^{a} c_i$$

Podobne spočítame body aj hráčom B, C s počtom vedúcich kariet b, resp. c.



Obr. 1.2: Ukážka ako môže vyzerať záverečné bodovanie

Príklad: Hráč A má 18 bodov za žetóny, za karty má -81 bodov (karty s hodnotami 5, 6, 7, 10, 31 a 32 nie sú vedúce), teda celkovo má hráč A -63 bodov. Hráč B má 10 žetónov a jeho vedúce karty sú 3, 12 a 27, teda celkový počet bodov hráča B je -32 bodov. Hráč C má celkovo 5 žetónov, z kariet si v tomto prípade nepočíta karty 23, 24 a 35. Celkový počet bodov hráča C je -87 bodov.

Víťazom sa stáva hráč s najväčším skóre. Na ukážke z obr.1.2 je víťazom hráč B, na druhom mieste je hráč A a posledný skončil hráč C. Ako vidíme, vo väčšine prípadov budú hráči získavať negatívne skóre, kde karty budú tvoriť najväčšiu časť bodov. Počet žetónov na konci hry málokedy ovplyvní celkové poradie.

Ako sme si mohli všimnúť, cieľom hráča je zbierať karty tak, aby na konci hry tvorili čo najmenej úsekov. V prípade, že by kartu s hodnotou 18 nezobral hráč A ale hráč C, obaja hráči by si tým polepšili. Hráč A by sa zbavil celého jedného úseku a mal by o 18 bodov viac, zatiaľ čo hráč C by touto kartou spojil svoje dva úseky a tým by väčšiu z vedúcich kariet týchto dvoch úsekov v konečnom bodovaní nepočítal. Získal by tým 19 bodov a určitý obnos žetónov.

1.2 O hre

V tejto časti sa pozrieme na niektoré aspekty hry *No Thanks*. Overíme jej hrateľnosť a pozrieme sa na patové situácie a či sa do nich hráči môžu dostať. Začneme ale tým, že sa pozrieme v akých bodových ziskoch sa môžu hráči nachádzať.

Ukážeme si extrémny prípad, v ktorom všetci hráči v súčte získajú čo najväčší počet bodov. Nech 9 vyradených kariet na začiatku hry má hodnotu $\{27, 28, ..., 35\}$. V takom prípade vidíme, že zvyšné karty tvoria jeden úsek s vedúcou kartou 3. Táto karta musí byť vedúca, pretože sa dostane do hry, získa ju niektorý z hráčov a karta s číslom 2

sa v hre nenachádza. V prípade že by jeden hráč získal všetky karty z balíčku, mal by -3 body, čo je najmenšie možné negatívne skóre zo získaných kariet (v prípade že hráč získal nejakú kartu).

V triviálnom prípade teda všetky karty zoberie prvý hráč a celkové skóre bude (11:11:8). Ak by prví dvaja hráči kartu odmietli a následne tretí hráč by zobral všetky karty, všetci hráči by mali 10 bodov.

Tvrdenie 1: Majme hru *No Thanks* s tromi hráčmi. Najväčší možný súčet bodov všetkých hráčov je 30.

Dôkaz: V hre je 33 žetónov, každý žetón je za 1 bod. V hre sa nachádza 24 kariet. Z nich bude minimálne jedna na konci hry vedúcou kartou. Majme vedúce karty všetkých hráčov $l_1, l_2, ..., l_n$. Označme súčet negatívnych bodov za tieto karty $L = \sum_{i=1}^{n} l_i$. Súčet bodov B všetkých hráčov vypočítame ako:

$$B = 33 - L$$

Keďže vieme, že niektorá karta musí byť na konci hry vedúcou, hodnota L musí byť minimálne 3. Preto:

$$B \leq 30$$

Toto tvrdenie využijeme v neskorších kapitolách.

Menej dôležité, no podobne zaujímavé číslo je pre nás najmenší možný súčet bodov všetkých hráčov. Intuitívne si vymodelujeme nasledovnú situáciu:

- 9 kariet s číslami od 3 do 11 bude vyradených z hry
- každá karta bude na konci hry vedúca
- žetónov je dokopy 33

Najmenší súčet bodov všetkých hráčov B vypočítame ako:

$$B = 33 - \sum_{i=12}^{35} i = -531$$

Príklad ako môže vyzerať nami navrhnutá hra na jej konci:

1. Karty hráča A: $\{14, 17, 20, 23, 26, 29, 32, 35\}$, počet žetónov: 19. Skóre: -177
2. Karty hráča B: $\{13, 16, 19, 22, 25, 28, 31, 34\}$, počet žetónov: 11. Skóre: -177
3. Karty hráča C: $\{12, 15, 18, 21, 24, 27, 30, 33\}$, počet žetónov: 3. Skóre: -177

Teraz vyslovíme dve tvrdenia ktoré hovoria o hrateľnosti tejto hry. V *Tvrdení 2* ukážeme, že jedna inštancia hry sa nemôže v tom istom stave ocitnúť viac ako raz (stavom chápme konfiguráciu hry popisujúcu všetky verejné informácie - počty žetónov hráčov, zoznamy kariet hráčov, vyložená karta...) a keďže hra má konečný počet stavov, tak po konečnom počte krokov skončí. Navyše vieme povedať aj hrubý odhad po koľkých najviac krokoch hra skončí. Toto tvrdenie je pre nás dôležité preto, aby sme neskôr pri implementácii nemali problémy pri úplnom prehľadávaní všetkých možností odohrania hry.

V *Tvrdení 3* popíšeme situáciu v ktorej by bol hráč donútený zahrať dvakrát po sebe akciu *Áno prosím*. Ukážeme, že do tejto situácie sa nemožno pri regulárnom hraní hry dostať. Inými slovami: Ak nemá hráč žiadne žetóny, tak spolu s kartou ktorú bude nútený vziať získa aj nenulový počet žetónov. Tým pádom nebude nútený hrať dvakrát po sebe akciu *Áno prosím*.

Tvrdenie 2: Hra *No Thanks* skončí po konečnom počte ťahov.

Tvrdenie patrí medzi tie triviálnejšie, jeho dôkaz uvádzame mierne neformálne. **Dôkaz:** Pozrime sa na podproblém jednej karty. Ťah *Áno prosím* sa vzhľadom na jednu kartu vyskytuje iba raz - v momente keď niekto tento ťah zahrá, karta je odobratá zo stredu stola a vyloží sa nová karta. Pozrime sa teraz na ťah *Nie ďakujem*. Pri každom takomto zahranom ťahu sa vloží jeden žetón do stredu ku karte. Týchto žetónov je v hre práve 33, teda ťah *Nie ďakujem* môže byť odohratý nanajvýš 33-krát, ťah *Áno prosím* iba raz. Keďže jedna karta sa zo stredu stola odoberie po konečnom počte krokov, aj 24 kariet sa odoberie po konečnom počte krokov. Teda koniec hry nastane po konečnom počte krokov.

Keďže vieme, že v hre sa vyskytne 24 kariet a pre každú kartu sa vykoná nanajvýš 34 ťahov (33-krát akcia *Nie ďakujem* a 1-krát *Áno prosím*), vieme povedať, že v hre *No Thanks* sa nevykoná viac ako $24 \times 34 = 816$ ťahov.

Zároveň je zrejmé, že ak sa vykoná 33-krát za sebou akcia *Nie ďakujem*, jeden z hráčov bude mať všetky žetóny, zatiaľ čo zvyšní hráči nebudú mať žetóny žiadne. Preto hráči bez žetónov budú najbližšie nútení zahrať akciu *Áno prosím*. Horné ohraničenie počtu ťahov je teda zjavne nižšie. Pre nás však stačí, že toto číslo je konečne veľké.

V momente keď je hráč na ťahu a nemá žiaden žetón, je nútený si kartu zo stredu stola zobrať. Problémom hrateľnosti hry *No Thanks* by mohla byť situácia, kedy sa hráč dostane do stavu, že všetky zvyšné karty v balíčku (v ktorom sú aspoň 2 karty) bude nútený si zobrať. V prípade že hráč na ťahu nemá žiadne žetóny a ani v strede pri karte sa nenachádza žiaden žetón, je nútený si kartu zobrať. Žiaden žetón tým ale nezíska a daný hráč zostáva na ťahu - opäť s prázdnou zásobou žetónov. Takto je

nútený si zobrať ďalšiu kartu až kým nezoberie celý balíček. V dôkaze nasledujúceho tvrdenia ukážeme, že do takejto situácie sa hráč nemôže dostať.

Tvrdenie 3: Hráč hry *No Thanks* sa nemôže dostať do stavu, že bude donútený zahrať ťah *Áno prosím* dvakrát po sebe.

Dôkaz: Predpokladajme že hráč H je donútený zahrať dvakrát za sebou ťah *Áno prosím*. Pozrime sa na situáciu pred jeho prvým ťahom z dvojice *Áno prosím*.
Rozlišujeme dve situácie:

1. Hráč H hru začínal a do tohto dvojťahu bol donútený hneď na začiatku hry. Hráč H je však donútený k ťahu *Áno prosím* iba vtedy keď vo svojej zásobe nemá žiadne žetóny. Keďže na začiatku hry má každý hráč 11 žetónov, hráč H nemôže byť donútený k tomuto ťahu na začiatku hry.

2. Pred týmto dvojťahom bol na rade hráč po pravici hráča H. Pozrime sa na to, ako mohol tento hráč hrať. Mohol hrať ťah *Áno prosím*, ktorým by kartu aj žetóny zobral, z balíčka vyložil novú kartu a zostal by na ťahu. Mohol ale hrať aj *Nie ďakujem*, čím by zaplatil jeden žetón zo svojej zásoby do stredu stola a na rad by sa dostal ďalší hráč - hráč H.

Ako vidíme, ťahom *Áno prosím* sa nemení hráč na ťahu, ťahom *Nie ďakujem* však áno. Keďže sa hráč H dostal na ťah, hráč po jeho pravici musel ako posledný ťah zahrať *Nie ďakujem*, čím odovzdal jeden žetón zo svojej zásoby do stredu. Keďže sa v čase prvého ťahu *Áno prosím* hráča H v strede stola nachádzal minimálne jeden žetón, bol po vykonaní tohto ťahu pridaný do zásoby hráča H. Pred druhým ťahom *Áno prosím* má teda hráč H v zásobe aspoň jeden žetón a teda druhá akcia *Áno prosím* za sebou nie je z donútenia.
Týmto je tvrdenie dokázané.

1.3 Zjednodušené verzie hry

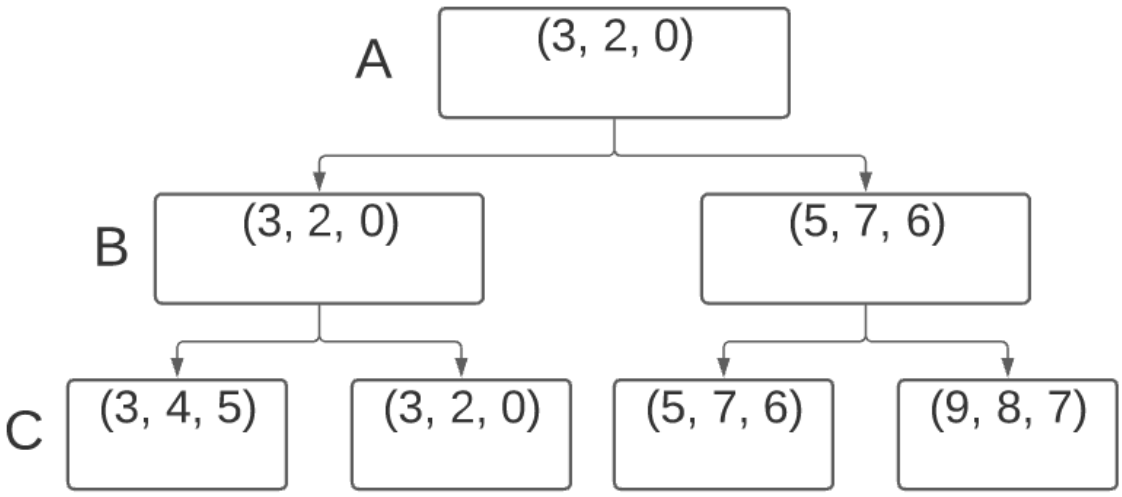
Vhodný spôsob ako spracovať tému väčšieho rozsahu je zjednodušiť si ju, nájsť riešenie na tomto zmenšenom probléme a následne vhodne použiť získané poznatky v pôvodnej verzii problému.
Za účelom zjednodušenia hry sme navrhli niekoľko verzií hry ktoré si popíšeme a vyberieme pre potreby skúmania pôvodnej hry tú najvhodnejšiu.

1.3.1 Verzia hry so známou budúcnosťou

V tejto úprave hry *No Thanks* je balíček upravený tak, že všetci hráči vedia v akom poradí budú karty vstupovať do hry. Taktiež hráči vedia ktoré karty sú na začiatku hry vyradené. Ak teda hráči vedia, že karty 14 a 16 sú vyradené, karta 15 je pre nich nezaujímavá, keďže bude tvoriť sekvenciu dĺžky 1. Môžeme očakávať, že pri optimálnom hraní sa počet žetónov pri karte tvoriacej sekvenciu dĺžky 1 bude blížiť k číslu tejto karty.

Výhodou verzie so známou budúcnosťou je to, že máme úplnú informáciu o hre a na základe toho si vieme odsimulovať hru tak, že každý hráč sa rozhodne optimálne. Pri rozhodovaní pritom očakávame, že všetci hráči sa rozhodujú optimálne. Tento predpoklad môže byť príliš optimistický, avšak ak sa nenaplní a naši súperi nebudú hrať optimálne, nám to vadiť nebude, pretože to by znamenalo, že sami sa dostanú do horšej pozície.

Avšak ukážeme si, že aj keď by sme mali podľa našich predpokladov skončiť na prvom mieste, neoptimálne hranie našich súperov toto môže zmeniť. Preto je možné povedať, že v hre *No Thanks* vo všeobecnosti žiaden hráč nemá istotu toho, že vyhrá.



Obr. 1.3: Simulácia hry s optimálnym hraním všetkých hráčov

Na obrázku vidíme možnú simuláciu hry s optimálnym hraním všetkých hráčov. Koreň stromu je ťah hráča A ktorý sa rozhoduje medzi dvomi možnosťami. Keďže na začiatku simulácie nevie ktorá z týchto dvoch možností je pre hráča A lepšia, odsimuluje najprv jednu, potom druhú a potom ich porovná. Pozrime sa teda na situáciu v ľavom synovi koreňa. V ňom je na ťahu hráč B, ten sa pozrie na svojich dvoch synov - o ťah dopredu. Pre jednoduchosť predpokladajme, že obaja jeho synovia sú listy - koniec hry, a teda vieme povedať ktorý hráč má koľko bodov a na koľkom mieste skončil. Hráč B

dostane od svojich dvoch synov odpovede v tvare (A, B, C), kde príslušné čísla hovoria o bodoch hráčov *A*, *B*, resp. *C*. Hráč *B* si vyberie optimálnu z týchto dvoch možností - hodnotu (3, 2, 0). Podobne by si hráč *B* vybral aj v druhej vetve optimálne - hodnotu (5, 7, 6) a podobne aj jeho rodič, hráč *A*, si vyberie optimálne. Do koreňa sa teda dostalo (3, 2, 0) a hráč *A* vie, že má vykonať akciu ľavého podstromu.

Pozn: Pre jednoduchosť sme si ukázali všeobecný strom simulovania hry s 3 hráčmi. Špecifiká *No Thanks* ako opakovanie ťahu hráča po akcii *Áno prosím* či situácia chýbajúcich žetónov (stav tejto hry bude mať iba jedného syna) sme vynechali.

Hráč *A* teda zahral optimálne a podľa svojich predpokladov skončí pri optimálnom hraní všetkých hráčov na prvom mieste. Tento predpoklad však nemusí stáť na pevných základoch, pretože hráč *B* môže hrať neoptimálne a namiesto (3, 2, 0) vybrať ťah s hodnotou (3, 4, 5). Vtedy zostáva na druhom mieste, no hráč na treťom mieste naňho stráca iba 1 bod.
Skutočnosť je však pre nás ešte pesimistickejšia. Uvažujme situáciu, kedy je namiesto najľavejšieho listu s hodnotou (3, 4, 5) hodnota (0, 2, 3). V tomto prípade sa hráč *B* nemá na základe čoho rozhodnúť, obe situácie sú pre neho identické - v oboch je na rovnakom mieste a rozdiel skóre so súpermi je v oboch prípadoch rovnaký. V takom prípade môže vrátiť náhodnú z týchto dvoch hodnôt. Ak sa však hra naozaj dostane do tohto stavu, hráč *B* sa opäť rozhodne náhodne, čo však môže byť inak ako v simulácii. Preto je možné, že aj keď simulácia hry povie hráčovi *A* že by pri optimálnom hraní všetkých hráčov mal vyhrať, môže sa stať, že skončí posledný napriek tomu že všetci hrali optimálne.

1.3.2 Verzia hry s neobmedzeným počtom žetónov

Ako z názvu vyplýva, hráči majú v tejto verzii hry neobmedzený počet žetónov. Avšak žetóny budú mať na konci hry stále hodnotu plusových bodov. Na konci hry získa každý hráč toľko bodov, koľko žetónov získal akciou *Áno prosím* a odpočíta sa mu bod za každý žetón spotrebovaný akciou *Nie ďakujem*.

Oproti pôvodnej verzii je tu možnosť, že hra bude trvať donekonečna, avšak môžeme predpokladať, že hráči hrajú rozumne a kartu s hodnotou n vezmú akciou *Áno prosím* ak sa pri nej nachádza aspoň n žetónov. Týmto krokom si určite nezhoršia výsledok, naopak, zhoršili by si ho zaplatením žetónu akciou *Nie ďakujem*. Preto aj pri rekurzívnom prechádzaní stromu budeme predpokladať, že hráč v takomto prípade použije akciu *Áno prosím*.

V tejto obmene hry môžeme očakávať vyššie výsledné skóre, pretože hráči nebudú nútení zobrať kartu ktorá by im uškodila, a naopak, inému hráčovi pomohla. Tento zjednodušený variant sa začína podobať na čisto pravdepodobnostnú hru, kedy by nás zaujímala iba hodnota karty a pravdepodobnosť s akou môžeme vytvoriť sekvencie

určitej dĺžky ktorých je táto karta súčasťou. Žetóny ako dôležitý faktor pôvodnej hry by tak stratili svoj pravý význam. Často sa v hrách *No Thanks* stáva, že hráč zoberie kartu nie preto, že sa mu oplatí získať žetóny pre body získané na konci hry, ale preto, že vďaka nim vie lepšie manévrovať počas hry - mať viac žetónov ako súperi je cesta ako sa vyhnúť kartám s vyšším počtom negatívnych bodov.

1.3.3 Verzia bez vyradených kariet

Čo keby sme hrali so všetkými kartami ktoré nám hra ponúka a žiadne pred začatím nevyradili? Táto verzia nám zaručuje istotu, že každá karta sa nachádza v balíčku. Záleží iba na poradí kariet a ťahoch protihráčov.

Môžeme vidieť niekoľko triviálnych priebehov tohto variantu hry:

- Začínajúci hráč A zoberie všetky karty z balíčku. Vytvoria jednu sekvenciu s 3 negatívnymi bodmi a hráč A skončí na poslednom mieste.
- Prví dvaja hráči zahrajú akciu *Nie ďakujem*. Hráč C ktorý je tretí na rade zoberie všetky karty akciou *Áno prosím*. Hra skončí remízou, keďže hráč C bude mať okrem 3 negatívnych bodov za karty o 3 žetóny viac od súperov
- Všetci traja hráči odohrajú ťah *Nie ďakujem* a hráč A zoberie karty akciou *Áno prosím*. Podobne ako v predchádzajúcom prípade hra skončí remízou.

Zaujímavejšia situácia by mohla nastať v prípade, ak by hráč namiesto použitia akcie *Áno prosím* použil akciu *Nie ďakujem* napriek tomu, že sa snaží získať celý balíček kariet.

Teraz si popíšeme jeden z viacerých možných prístupov hráča A:

- Hráč A používa akciu *Áno prosím* na všetky karty okrem poslednej karty z balíčku. Keď tá príde do hry, použije akciu *Nie ďakujem*.

Cieľom hráča A je zobrať všetky karty a navyše získať od každého súpera po jednom žetóne. Tak docieli, že hráči budú zdieľať prvé miesto.

Príklad: Hráč A zobral všetky karty a ako posledná prišla do hry karta 20. Hráč A má momentálne negatívne body za vedúce karty 3 a 21. Zahrá akciu *Nie ďakujem*. Pozrime sa ako teraz budú hrať súperi. Ak si hráč B kartu zoberie, dostane síce hráča A do zlej situácie, ale aj samého seba. Skončil by na druhom mieste čo je podobné ako deliť si prvé miesto s tromi hráčmi, avšak rozdiel v skóre je menej priaznivý. Zahrá preto akciu *Nie ďakujem*. Podobnú situáciu má aj hráč C. Preto sa dostane hráč A opäť na ťah. Ten má teraz možnosť zobrať kartu a skončiť hru na zdieľanom prvom mieste. Otázka je, či by nemohol ešte vyťažiť viac a vyhrať ako jediný.

Ak by hráč A opäť zahral akciu *Nie ďakujem*, hráč B by mal na výber z dvoch možností: buď kartu zoberie a skončí na druhom mieste alebo kartu nezoberie (ak by to bol optimálny ťah z tejto dvojice, rozhodol by sa tak aj hráč C), hráč A by sa dostal opäť na rad, zobral by kartu a skončil by prvý. A hráči B a C na zdieľanom druhom mieste. Hráč B by si preto vybral prvú možnosť. Keďže sa však v tej nachádza hráč A na poslednom mieste, zoberie kartu s tromi žetónmi a všetci hráči skončia na zdieľanom prvom mieste.

Kapitola 2

Definície a pojmy

V tejto kapitole vyslovíme základné definície a pojmy ktoré budeme v tejto práci využívať. Uvedieme si čo je Nashove ekvilibrium a ako súvisí s našou prácou, definujeme si dátovú štruktúru k-d stromy a ukážeme si čo je to alfa-beta orezávanie. Taktiež si zadefinujeme už skôr spomenutý úsek kariet a viacero ďalších vecí.

Definícia 1: Majme zoradenú postupnosť n kariet $c_0, c_1, ..., c_{n-1}$ s hodnotami $c_i \in \{3, 4, ..., 35\}$. Ak platí

$$c_i = c_{i+1} - 1 \text{ pre } i \in \{1, 2, ..., n-1\}$$

voláme túto postupnosť *úsek* a kartu s hodnotou c_0 voláme *vedúcou*.

Príklad: Majme karty s hodnotami $14, 15, 16, 18, 25, 26, 30$. Túto postupnosť vieme rozdeliť na štyri úseky s vedúcimi kartami $14, 18, 25$ a 30.

Úseky hrajú veľkú úlohu v rozhodovaní sa či hráč má záujem o práve vyloženú kartu alebo nie. Zoberme si príklad, kde je vyložená karta s číslom 18, hráč A má kartu 17, hráč B má kartu 19 a obaja hráči majú podobný počet žetónov. Obaja hráči majú o kartu podobný záujem, no zatiaľ čo by hráčovi A priniesla iba body za žetóny, hráčovi B by okrem toho zmenšila vedúcu kartu.

Rozdiel jedného bodu však nie je v tejto hre nič veľké. Záležať bude nakoniec iba na tom, ktorý z hráčov sa dostane skôr na rad - ten si túto kartu vezme. Pretože ak by ju nevzal (napr.preto, že by chcel za ňu získať viac žetónov) vzal by ju jeho súper. Môžeme si všimnúť, že ak majú hráči A a B záujem o podobné karty a hráč B sa dostane na rad po hráčovi A, lepšiu pozíciu pre získanie týchto kariet má hráč A. Ak predpokladáme, že hráč C má dostatok žetónov a nemá o spomenuté karty záujem, tak so šancou 2/3 sa ku karte dostane hráč A (keď je na ťahu hráč A alebo C) a hráč B sa ku karte dostane iba v 1/3 prípadoch (na ťahu hráča B).

Definícia 2: Majme kartu s hodnotou c. Pokiaľ danú kartu zobral hráč ťahom *Áno prosím* a už má vo svojej zásobe kariet kartu s hodnotou $c + 1$, hovoríme že ide o *rozšírenie úseku zdola*. Pokiaľ už má hráč vo svojej zásobe kariet kartu s hodnotou $c - 1$, hovoríme že ide o *rozšírenie úseku zhora*. Pokiaľ ide o *rozšírenie úseku zdola* a zároveň o *rozšírenie úseku zhora*, hovoríme že ide o *spojenie úsekov*.

Na príklade pred touto definíciou môžeme vidieť, že hráč A sa pokúša o rozšírenie úseku zhora a hráč B sa pokúša o rozšírenie úseku zdola.

Pre jednoduchosť, v nasledujúcej definícii myslíme *hernou stratégiou* deterministický algoritmus, podľa ktorého sa hráč rozhoduje na svojom ťahu ktorý krok vykoná. Podobne, *výsledným skóre* myslíme zovšeobecnene výsledok hry pre daného hráča. Pokiaľ má hráč výsledné skóre lepšie ako jeho protihráč, skončil na lepšom mieste. Pokiaľ majú dvaja hráči rovnaké výsledné skóre, skončili na rovnakom mieste.

Definícia 3: [3] Majme hru pre pre $n \geq 2$ hráčov a množinu stratégií $A = \{S_1, S_2, ..., S_n\}$ prislúchajúcich k jednotlivým hráčom. Označme výsledné očakávané skóre hráčov ako $p_1, p_2, ..., p_n$. Pokiaľ platí, že pre ľubovoľnú zmenu stratégie S_i na S_i' je $p_i \geq p_i'$, množina stratégií A je *Nashove ekvilibrium*.

Známym paradoxom a príkladom Nashovho ekvilibria je *Dilema väzňov*. V nej sa 2 hráči nezávisle na sebe musia rozhodnúť či budú spolupracovať s políciou alebo nie, pričom však nevedia ako sa rozhodne ten druhý a ani nemajú žiadnu možnosť medzi sebou komunikovať. Obom väzňom hrozí odsúdenie na rok pokiaľ nebudú s políciou spolupracovať. Pokiaľ však jeden z nich spolupracovať bude a pomôže polícií usvedčiť druhého väzňa z inej trestnej činnosti, spolupracujúci väzeň bude oslobodený a druhý dostane až 20 rokov za mrežami. Pokiaľ budú spolupracovať obaja a navzájom na seba prezradia ďalšie veci, obaja dostanú 10 rokov za mrežami. Otázkou je, ako je ideálne pre hráčov sa rozhodnúť?

Aj keď je tu možnosť, že obaja väzni by mlčali a dostali by 1 rok, nakoľko nevedia ako sa rozhodne ten druhý, paradoxne je pre hráčov optimálne sa rozhodnúť tak, že jeden druhého navzájom udá. Pozrime sa spätne na definíciu Nashovho ekvilibria. Hráči sa majú rozhodnúť tak, aby ich stratégia bola taká, že výsledné skóre je vzhľadom na stratégie všetkých zúčastnených najväčšie možné.

Ak sa väzeň A rozhodne spolupracovať, väzeň B môže mlčať -> dostane 20 rokov, alebo môže spolupracovať a dostane 10 rokov. Ak väzeň A bude mlčať, väzeň B môže tiež mlčať -> dostane 1 rok za mrežami, alebo môže spolupracovať a bude oslobodený. Vidíme, že pre väzňa B je v oboch prípadoch výhodnejšie spolupracovať. Situácia je symetrická, teda platí to aj pre väzňa A.

Poznatky Nashovho ekvilibria vieme aplikovať pre rôzne hry. Hlavne pri hrách so simultánnymi ťahmi nevieme povedať akú stratégiu má súper zvolenú, no pre rôzne

jeho stratégie chcem pre seba vybrať tú najlepšiu možnú - s najlepším výsledným skóre(resp. s najlepším rozdielom skóre oproti tomuto súperovi). V hre *No Thanks* môžeme vidieť podobné črty so spomenutým paradoxom väzňov. Niekedy môže byť pre hráča A výhodné zobrať kartu iba preto, že niektorému zo súperov (hráčovi B) by veľmi pomohla. Aj keď hráčovi A dá zlé body, pohorší mu menej ako by polepšila hráčovi B. Treba však povedať, že týchto situácií je v hrách 3 a viac hráčov málo, pretože tento spôsob hrania je optimálny iba v prípade, že pozícia hráča C nebude vzhľadom na hráča A zmenená. Inými slovami: Nie je dobré ak hráč A pohorší sebe aj hráčovi B za cenu toho, že sa pred neho dostane hráč C.

2.1 Minimax

Cieľom simulovania rôznych priebehov hry je nájsť z nich ten, ktorý popisuje optimálne hranie všetkých zúčastnených strán. V prípade ak protihráči nehrajú optimálne, síce sa náš predpoklad nenaplní, no bude to pre nich znamenať, že ich pozícia v hre sa zhoršila.

Avšak to neznamená priamo, že všetci jeho súperi dosiahnu lepší výsledok. V prípade ak hráč A nehrá optimálne, môže nastať situácia že hráčovi B sa pozícia zlepší, no hráčovi C sa pozícia zhorší. Nakoľko sa však hráčovi A pozícia zhorší, v priemernom prípade sa hráčom B a C pozícia zlepší.

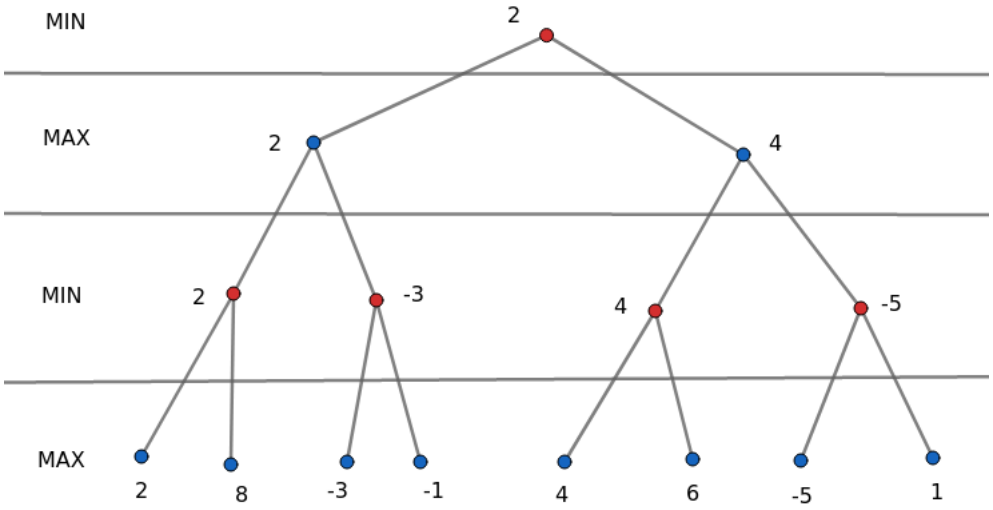
Môžeme teda naďalej brať do úvahy iba optimálne hranie hráčov a v prípade že optimálne hrať nebudú, pozícia súperov bude pravdepodobne lepšia.

Myšlienka minimaxu je založená na hre dvoch hráčov, kde stav hry je popísaný reálnym číslom a hovorí o tom, ktorý hráč má bližšie k víťazstvu (resp. ak je hra na konci, ktorý hráč vyhral). Jeden z hráčov - MAX - vyhrá keď je hodnota stavu hry kladná, druhý - MIN - vyhrá keď je záporná. Zároveň, čím väčšiu hodnotu má stav hry, v tým lepšej pozícii sa nachádza hráč MAX, a naopak.

Pri prehľadávaní rôznych priebehov hry minimax predpokladá, že sa hráči rozhodujú optimálne, teda vyberú si z dvoch možných scenárov ten z tých, v ktorom majú lepšiu pozíciu - pričom zároveň sami predpokladajú, že súper hrá optimálne. Preto musíme tento výpočet robiť od konca hry - od listov stromu, ktorý znázorňuje rôzne priebehy hry.

Na konci hry sme schopní vyčísliť ako hra skončila podľa pravidiel danej hry. Takto získame hodnoty stavov vo všetkých listoch. My sa však potrebujeme rozhodnúť pri koreni stromu ktorý z ťahov vykonať, preto je potrebné hodnoty z listov postupne dostať do koreňa. V našej simulácii sa vieme pozrieť na jeden ťah od konca (na obrázku 2.1 štyri červené vrcholy) a získať hodnoty týchto stavov tak, že v každom z nich sa pozrieme na všetky možné ťahy a vyberieme z nich ten, ktorý je pre daného hráča na

ťahu najvýhodnejší. Na obrázku je to hráč MIN, ktorý z dvojice hodnôt $(2, 8)$ zvolí 2 a podobne ďalej.



Obr. 2.1: Ukážka využitia pravidiel minimaxu v praxi

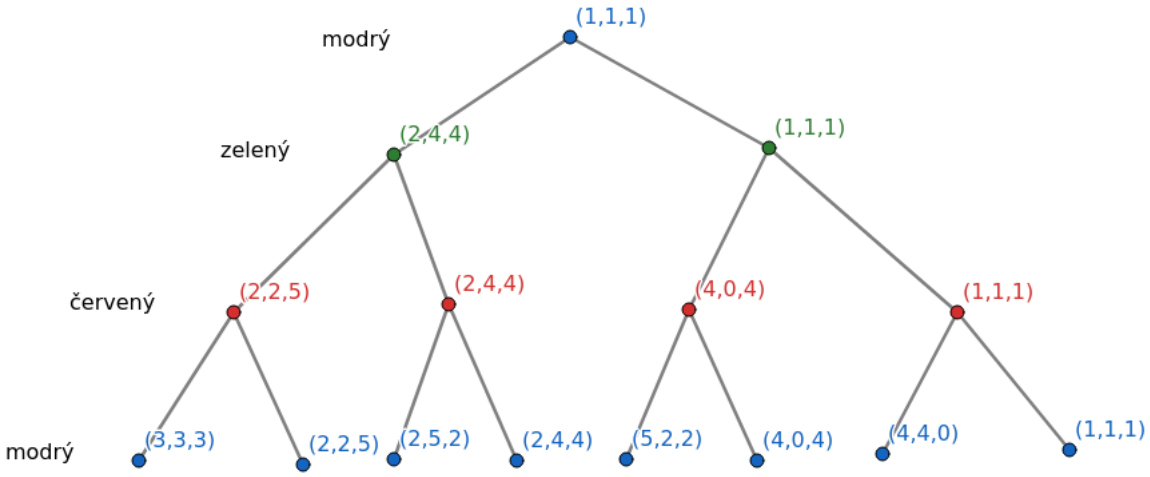
Na vyššej úrovni sa na ťahu nachádza hráč MAX, ktorý z dvojice $(2, -3)$ vyberie 2 a z dvojice $(4, -5)$ vyberie 4. Všimnime si, že pre hráča MIN je teoreticky najvýhodnejšie prísť do listu so stavom -5, no keďže predpokladáme, že hráč MAX hrá optimálne a zo stavov $(4, -5)$ si vyberie stav s hodnotou 4, hráč MIN spraví ťah ktorým sa dostane do ľavého podstromu.

2.1.1 Aplikácia minimaxu pri 3 a viac hráčoch

Pri hrách s 3 a viacerými hráčmi nám už nestačí jedno číslo na popísanie stavu hry. Vhodnú alternatívu popísali páni C.A. Luckhardt and K.B. Irani v [4]. Myšlienka minimaxu, kde jeden z hráčov sa snaží o mimimalizovanie hodnoty a druhý o maximalizovanie hodnoty bola transformovaná na Max^n. Stav hry teda bude popísaný n-ticou hodnôt, kde hráč i sa snaží maximalizovať i-tu dimenziu.

Samotným maximalizovaním nemyslíme to, že hráč i uprednostní stav hry kde má vyššie skóre, ale porovná svoje skóre so súpermi. Hráč bude preferovať ten stav hry, v ktorom má v lepšiu pozíciu. Ak je pozícia hráča v oboch stavoch hry rovnaká, uprednostní tú pozíciu, v ktorej je rozdiel skóre s ostatnými hráčmi pre hráča priaznivejší.

Pozrime sa ako sa na obrázku 2.2 presúvajú hodnoty z listov do koreňa. Stav hry trojčíslie v tvare (modrý, zelený, červený), čiže modrý hráč sa snaží maximalizovať najľavejší parameter, zelený hráč stredný parameter a červený hráč parameter vpravo. Červený hráč preto vyberie z dvojice stavov $(3, 3, 3), (2, 2, 5)$ stav $(2, 2, 5)$. Všimnime si, že modrý hráč uprednostní stav $(1, 1, 1)$ pred stavom $(2, 4, 4)$ napriek tomu, že jeho skóre je nižšie, pretože uprednostňujeme spoločné prvenstvo pred posledným miestom.



Obr. 2.2: Ukážka Max^3

2.2 Orezávanie

Problém, ktorý vzniká pri simulácií, je obrovské množstvo rôznych priebehov hry. Z časového hľadiska je pre nás neprijateľné prehľadávať všetky možnosti ako môže hra prebiehať a z nich vybrať tú najlepšiu. Našťastie existujú algoritmické techniky ktoré nám vedia prehľadávanie stromov urýchliť. Jednou z týchto techník je alfa-beta orezávanie (alpha-beta pruning). Myšlienka orezávania spočíva v tom, že v niektorých prípadoch vieme usúdiť, že niektoré podstromy nemusíme vôbec prehľadávať, lebo nech by v nich bola ľubovoľná hodnota, výsledok nijak neovplyvní.

Stromom, resp. listami stromu prechádzame zľava doprava, t.j. na obrázku 2.1 ako prvú vypočítame hodnotu 2 v najľavejšom liste, následne v ďalšom hodnotu 8 apod. Vďaka tomu vieme do nich posunúť hodnotu 2, keďže v danom bode je na ťahu hráč MIN. Všimnime si na tomto príklade hneď niekoľko situácii kedy vieme niektoré listy, resp. podstromy orezať a tak urýchliť výpočet.

V momente kedy v treťom liste zľava vypočítame hodnotu -3 vieme zaručiť, že v predkovi tohto listu bude hodnota -3 alebo menšia (pretože hráč vyberá minimálnu hodnotu). Zároveň vieme, že vo vzdialenosti 2 od listov bude na ťahu hráč MAX ktorý bude vyberať väčšiu hodnotu, pričom už má vypočítanú hodnotu 4 a o druhej vieme, že bude nanajvýš -3. Preto už v tejto situácii vieme povedať, že z ľavých štyroch listov bude vybraná hodnota 4 pri akejkoľvek hodnote štvrtého listu. Preto si vieme dovoliť tento štvrtý list orezať, príp. ak by tento vrchol mal synov, tak orežeme celý podstrom.

Podobnú situáciu si všimneme aj pri najpravejšom liste. Čo je však zaujímavé, vieme toho spraviť ešte viac a orezať obidva najpravejšie listy. Keď si všimneme, po vypočítaní listu s hodnotou 6 vieme, že hráč MIN si z dvojice $(4, 6)$ vyberie hodnotu 4. Z toho vieme vyvodiť, že hráč MAX vyberie väčšiu hodnotu z dvojice hodnôt kde už jednu má vypočítanú - hodnotu 4. Preto hráč MAX vyberie hodnotu 4 alebo väčšiu.

Keďže však vieme, že hráč MIN v koreni stromu už má vypočítaný ľavý podstrom s hodnotou 2, hodnotu 4 alebo väčšiu z pravého podstromu určite nevyberie. Preto vieme odrezať posledné dva listy (resp. podstromy), pretože výsledok nijako neovplyvnia.

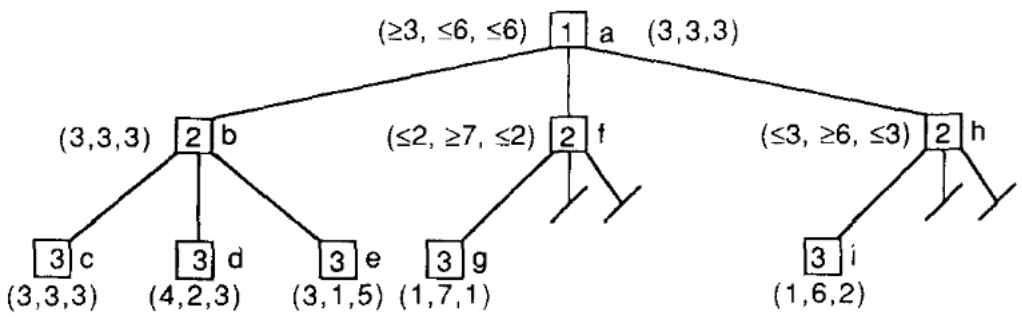
Tejto technike, kedy orezávame na základe informácie o dve generácie v strome vyššie sa v článku [5] hovorí aj hlboké orezávanie (deep pruning). V originálnom názve je toto orezávanie nazývané aj ako alfa-beta orezávanie (alpha-beta pruning), kde parametre alfa a beta udržiavajú najnižšiu hodnotu ktorú hráč MAX vie dosiahnuť, resp. najvyššiu možnú hodnotu ktorú vie hráč MIN dosiahnuť. V prípade, že $\alpha > \beta$ (v príklade v predchádzajúcom odseku bolo alfa=4 a beta=2), môžeme zvyšok podstromu odrezať. V ďalšej podkapitole o aplikácii orezávania v hrách 3 a viac hráčoch bude pre nás toto dôležité.

2.3 Orezávanie pri hre s 3 a viac hráčmi

Techniku orezávania vieme využiť aj pri viacerých hráčoch, i keď obmedzene. V článku [1] kde je popísané využitie orezania v hrách 3 a viacerých hráčov je vyslovený predpoklad horného ohraničenia súčtu bodov všetkých hráčov. Na obrázku 2.3 je súčet horného ohraničenia 9. Zhodou okolností je súčet každého listu rovný 9.

Na ukážke sa nachádzajú hráči $\boxed{1}, \boxed{2}, \boxed{3}$ a stav hry je v tvare $(1, 2, 3)$. Môžeme vidieť že po vyčíslení vrchola $b = (3, 3, 3)$ vieme povedať, že vo vrchole a sa bude na ľavej pozícii nachádzať číslo 3 alebo väčšie. Vďaka hornému ohraničeniu 9 vieme povedať, že na zvyšných dvoch pozíciach sa nebude nachádzať hodnota 6 alebo väčšia. Podobne si môžeme všimnúť situáciu na vrchole f. V ňom je na ťahu hráč $\boxed{2}$, preto na základe hodnoty vrchola $g = (1, 7, 1)$ sa vo vrchole f bude nachádzať na strednej pozícii hodnota 7 alebo väčšia a na zvyšných dvoch hodnota nanajvýš 2.

Vďaka tomuto vieme povedať, že zvyšných synov vrchola f nemusíme prehľadávať, pretože vieme povedať, že hodnota z vrchola f určite nebude vybratá, keďže na ľavej pozícii tam môže byť nanajvýš hodnota 2, pričom už vo vrchole b je nájdená priaznivejšia hodnota pre vrchol a.

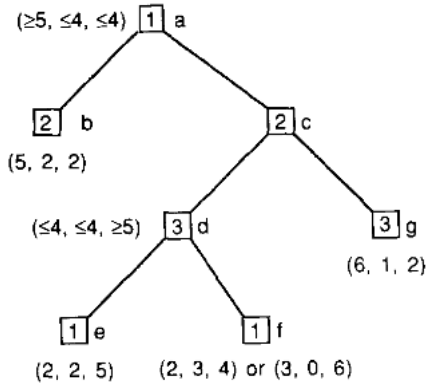


Obr. 2.3: Ukážka Max^3 s horným ohraničením 9 (prevzaté z [1])

Zatiaľ sme videli ukážku plytkého orezávania (shallow pruning), čiže na základe situácie o 1 generáciu vyššie v strome. V článku [1] je ukázané, že toto je najviac čo si môžeme dovoliť. Na obrázku 2.4 si môžeme všimnúť situáciu, kde využitie hlbokého orezania zlyhá.

Všimnime si hodnotu vrcholu e na základe ktorého vieme povedať, že vrchol $d = (\leq 4, \leq 4, \geq 5)$. Zároveň vidíme, že $a = (\geq 5, \leq 4, \leq 4)$, čiže vieme s určitosťou povedať, že ak by sa aj hodnota z vrchola d dostala do vrchola c, do vrchola a sa určite nedostane. Podľa pravidiel hlbokého orezania by sme mohli podľa situácie vo vrchole a (o 2 generácie vyššie) usúdiť, že vrchol f možno orezať.

Tu však nastáva problém. Aj keď je pravda, že hodnota z vrchola f sa do koreňa nedostane, ona sama môže ovplyvniť výpočet. Všimnime si, že pri $f = (2, 3, 4)$ sa do vrchola a dostane $(5, 2, 2)$, no pri $f = (3, 0, 6)$ sa do vrchola a dostane hodnota $(6, 1, 2)$.



Obr. 2.4: Ukážka zlyhania hlbokého orezávania pri hre 3 a viac hráčov (prevzaté z [1])

2.4 k-d stromy

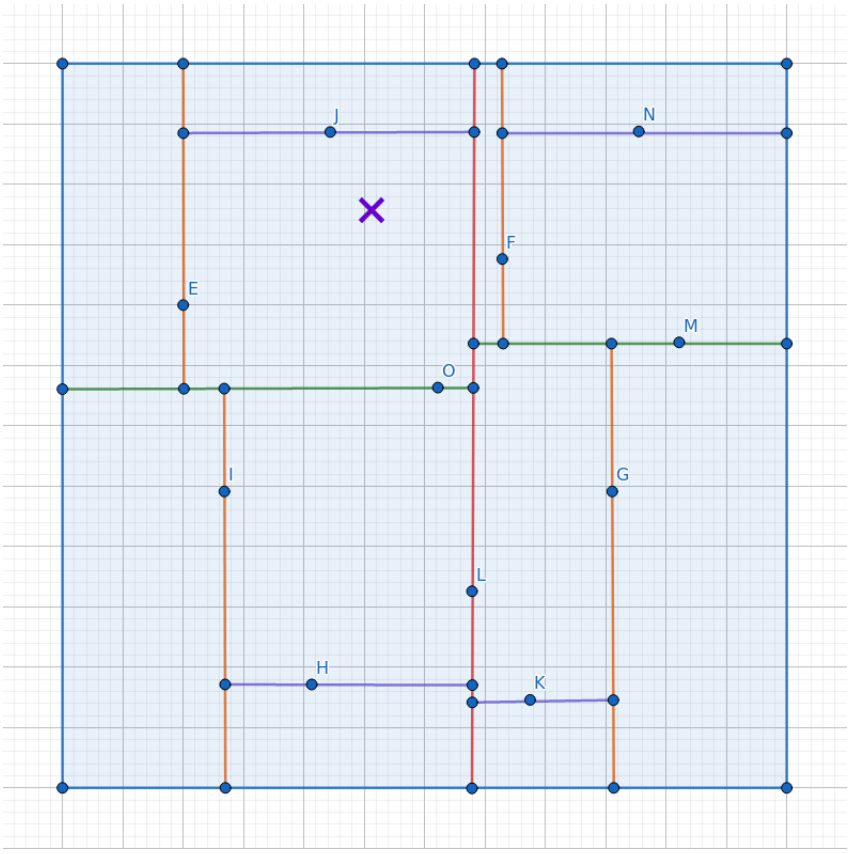
Uvažujme list papiera, na ktorom sú vyznačené body so súradnicami $[x, y]$. Na papieri vyznačíme bod A a chceme nájsť taký bod, ktorý je zo všetkých doterajších bodov k bodu A najbližšie. Môžeme odmerať vzdialenosť od všetkých bodov k bodu A, avšak pri veľkom počte bodov môže byť prechádzanie všetkých bodov zbytočne zdĺhavé.

Pozrime sa na situáciu na obrázku 2.5 kde máme vyznačené body a chceme nájsť najbližší bod k vyznačenému bodu X v ľavom hornom segmente. Okom môžeme vidieť že bod J je k nemu zo všetkých bodov najbližšie. Ako to však zistí počítač? Uloženie bodov do stromovej štruktúry podľa toho aké majú súradnice sa zdá byť dobrým nápadom. Rozdeľme teda plochu na dve polovice podľa súradnice x. Mediánom zo všetkých bodov je bod L. Ten rozdelil body na dve skupiny: tie ktoré sú od neho naľavo (hodnotu x majú menšiu) a tie ktoré sú napravo (hodnotu x majú väčšiu). Každú z týchto skupín opäť rozdelíme na dve časti, no tentokrát sa nám pýta nájsť

medián týchto skupín podľa súradnice y. Týmito mediánmi boli v tomto prípade body O a M. A podobne pokračujeme ďalej až kým sa nedostaneme k oblastiam obsahujúcim iba 1 bod - listom k-d stromu.

Použitie vhodných dátových štruktúr je pre programátora kľúčové aby výkon vyvíjaného softvéru bol na vysokej úrovni, ale aj aby implementácia a narábanie s dátami bolo jednoduché z programátorského hľadiska. Pre rýchle vyhľadávanie hodnôt v zozname je vhodné tento zoznam utriediť a následne hľadanú hodnotu vyhľadať binárne. Časová zložitosť takéhoto hľadania je logaritmická.

V našej práci nás bude zaujímať efektívne nájdenie najbližšej hodnoty, t.j.hodnoty s najmenším rozdielom od hľadanej. Takto sa budeme snažiť nájsť k určitému stavu hry najpodobnejší stav zo širokej databázy stavov. Problém však je, že stav hry nie je reprezentovaný jednou hodnotou, ale veľa rôznymi hodnotami. K-d strom je dátová štruktúra, ktorá slúži na vkladanie k-rozmerných dát a ich rýchle vyhľadávanie.



Obr. 2.5: Využitie k-d stromu pri dvojrozmerných dátach

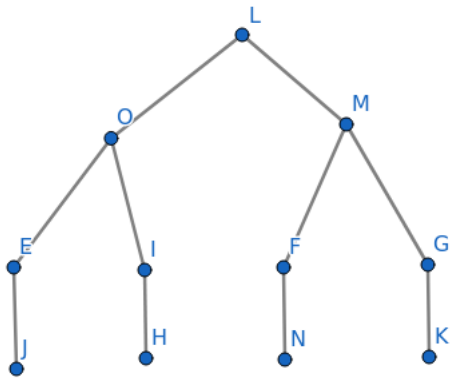
2.4.1 Popis k-d stromu

K-d strom [6] je binárny strom, kde každý vrchol je k-rozmerný bod reprezentujúci jeden prvok databázy - v našom prípade jeden stav hry. Koreňom stromu je prvok, ktorý je mediánom zo všetkých prvkov v závislosti od nultej (najľavejšej) dimenzie.

Všetky prvky v podstrome jeho ľavého syna majú hodnotu nultej dimenzie menšiu a všetky prvky napravo väčšiu.

Podobne, synovia koreňa (prvky so vzdialenosťou 1) sú tiež mediánom, avšak v závislosti od prvej dimenzie a iba spomedzi prvkov v rovnakej skupine podľa rozdelenia koreňom. A podobne ďalej, prvky so vzdialenosťou i od koreňa stromu sú mediány podľa i-tej dimenzie.

V prípade že strom má viac úrovní ako prvky dimznií, na i-tej úrovni (vzdialenosť od koreňa) sa nachádza medián podľa $i(mod)k$-tej dimenzie. Na obrázku 2.5 si môžeme všimnúť že pri dvojrozmerných dátach sme pri nultej iterácii rozdeľovali prvky podľa nultej dimenzie (červená zvislá čiara), následne podľa prvej dimenzie (zelené vodorovné čiary) a prvky vo vzdialenosti 2 od koreňa stromu sú mediánmi podľa nultej dimenzie.



Obr. 2.6: Prvky v k-d strome

Na obrázku 2.6 si môžeme všimnúť rolu prvkov z obrázka 2.5 v k-d strome. Prvky ktoré sú mediánmi nejakej oblasti podľa určitej súradnice sú v štruktúre k-d stromu rodičmi nejakého prvku. Tie prvky ktoré sú v oblasti samé sú v strome listami.

2.4.2 Vyhľadávanie v k-d strome

Vyhľadávanie v k-d strome je podobné binárnemu vyhľadávaniu s tým rozdielom, že musíme uvažovať viacero dimenzií podľa ktorých sú prvky v k-d strome usporiadané.

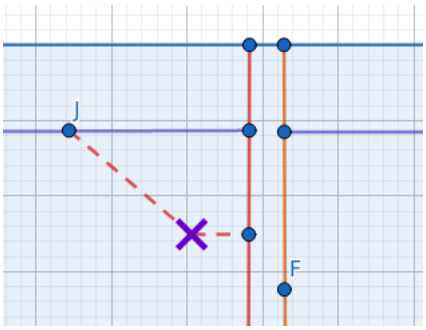
Pozrime sa na obrázok 2.5 kde chceme nájsť prvok s najmenšou vzdialenosťou k bodu X. Vďaka štruktúre ktorú máme popísanú vieme, že tento bod by sa nachádzal v ľavej časti k-d stromu, keďže jeho x-ová súradnica je oproti x-ovej súradnici prvka L menšia. Podobne vieme povedať že by sa nachádzal v ľavom podstrome prvku O podľa porovnania y-ových súradníc. A naozaj, hľadaný bod J sa nachádza v ľavom podstrome prvku O.

Čo ak by sme ale vyznačili bod o niečo viac napravo? Najbližší prvok by bol bod F ktorý je v pravej časti nášho k-d stromu, avšak stále by sme sa dostali do ľavého

podstromu kvôli porovnaniu x-ovej súradnice s prvkom L. Preto si nemôžeme byť istí že keď sa dostaneme k listu, dostali sme odpoveď. Rekurzívne sa musíme presvedčiť, že v oblastiach ktoré sme nepreskúmali sa nenachádza vhodnejší kandidát.

Na obrázku 2.5 môžeme vidieť že najbližší bod k vyznačenému je bod J. Cestou dole pri hľadaní najbližšieho suseda sme mali šťastie a dostali sme sa k nemu ako prvému.

Uvažujme prípad na obrázku 2.7 kedy by bod X bol o niečo viac vpravo. Vidíme, že vzdialenosť medzi bodmi X a J je menšia ako vzdialenosť medzi bodom X a iným segmentom. V tomto segmente sa potenciálne môže nachádzať bod, ktorý je bližšie k bodu X ako bod J. Preto je potrebné pri rekurzívnom vracaní sa späť do koreňa stromu kontrolovať aj tie podstromy, kde je možný výskyt vhodnejšieho kandidáta na najbližšieho suseda.



Obr. 2.7: Vyhľadávanie najbližšieho prvku v k-d strome, detail

Podrobný algoritmus na vyhľadávanie najbližšieho prvku v k-d strome bol popísaný v [7] a v našej práci nebude uvedený.

Kapitola 3

Náš prínos v oblasti

Pri riešení cieľu našej práce budeme využívať niektoré zaužívané techniky v prehľadávaní stromov. Táto kapitola sa bude zaoberať aký prínos sme dosiahli, a to nie len pre potreby našej práce, ale všeobecne pri používaní spomenutých techník.

Pre zvyšok práce budeme pracovať s verziou hry so známou budúcnosťou popísanou v podkapitole 1.3.1.

3.1 Orezávanie v hre s 3 a viac hráčmi bez horného ohraničenia súčtu bodov

V článku Richarda Korfa [1] o orezávaní v hrách 3 a viac hráčoch je dokázané, že je možné aplikovať plytké, avšak nie hlboké orezávanie. Okrem toho článok predpokladá horné ohraničenie súčtu bodov hráčov. Čo ak sa však v hre nenachádza horné ohraničenie, vieme orezávať aj bez toho aby sme mali tento poznatok? A za akú cenu?

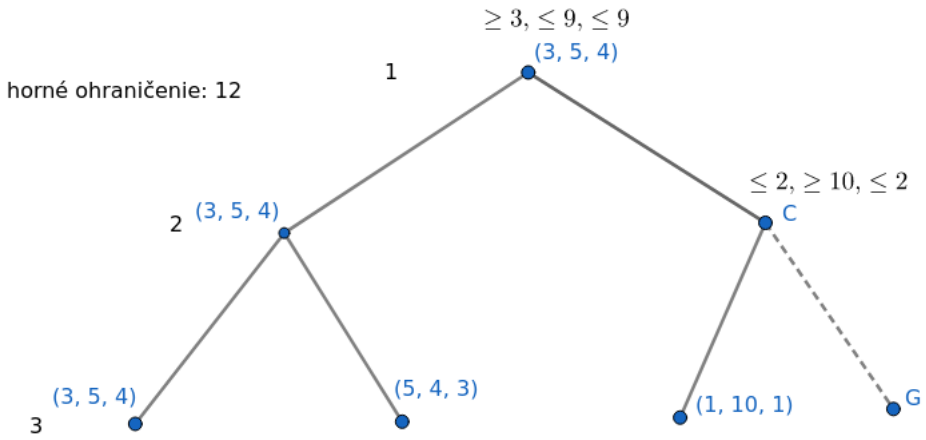
V hre No Thanks sa nachádza horné ohraničenie 30, avšak pravdepodobnosť, že súčet bodov hráčov mu bude blízky, je takmer nulová. Uvažujme situáciu že do hry sa dostanú karty z intervalu $[3, 26]$ a všetky ich zoberie jeden hráč. V takom prípade bude v hre jediná vedúca karta (3) a keďže je v hre 33 žetónov, najvyšší možný súčet bodov všetkých hráčov je 30. Z uvedeného je jasné, že pravdepodobnosť takéhoto zamiešania balíčka spolu pričom všetky karty zoberie jeden hráč je takmer nulová. Navyše, možno predpokladať, že od tohto ohraničenia sa budeme nachádzať veľmi ďaleko.

Všimnime si na obrázku 2.3 akým spôsobom vznikajú nerovnosti vo vnútorných vrcholoch stromu. Vidíme, že po prejdení ľavej vetvy sa do koreňa dostalo $(3, 3, 3)$ a teda na konci výpočtu musí byť hodnota prvého parametru 3 alebo väčšia, keďže v koreni je na ťahu prvý hráč. Keďže predpokladáme horné ohraničenie 9, hodnota zvyšných dvoch parametrov môže byť nanajvýš 6. Pozn: uvažujeme iba kladný počet bodov.

Podľa údajov z www.boardgamearena.com kde môžu hrať užívatelia online aj hru

No Thanks sa priemerné skóre hráčov pohybuje okolo -30. Málokedy sa stane, že niektorý z hráčov má na konci hry nezáporný počet bodov, avšak uvažujme situáciu že v niektorom z listov stromu získa prvý hráč 5 bodov. Vďaka hornému ohraničeniu vieme povedať, že jeho súperi môžu získať nanajvýš 25 bodov. Aby sme na základe tejto už extrémnej situácie mohli niečo orezať, v ďalšom ohodnotenom liste by sme mali nájsť situáciu kedy druhý alebo tretí hráč získa aspoň 26 bodov na konci hry. Tým pádom by prvý hráč získal nanajvýš 4 body a prázdny prienik týchto dvoch nerovností nám umožní orezať zvyšok podstromu.

Ako vidíme, daná situácia je vysoko nepravdepodobná. Avšak orezávanie z tohto článku má ešte ďalšie, závažnejšie úskalie: zohľadňujeme pri ňom iba počet bodov hráča, nie poradie. Ako sme si povedali v úvode práce, prioritou hráčov je čo najlepšie poradie, až sekundárne hľadia na rozdiel bodov voči súperom.



Obr. 3.1: Ukážka zlyhania orezávania z článku R.korfa pre potreby hry No Thanks

Všimnime si situáciu na obrázku 3.1 kde máme ilustrovanú hru s horným ohraničením 12, kde použijeme orezávanie z článku R.Korfa [1] a ukážeme, že takto orežeme vrchol, ktorý by sa inak dostal až do koreňa. Oproti článku predpokladáme, že hráči majú prioritu skončiť na čo najlepšom mieste a počet bodov je až druhoradý.

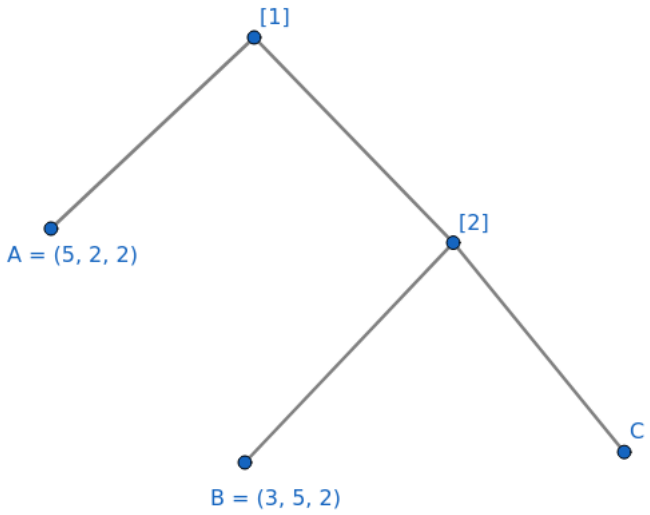
Z ľavého podstromu sa dostala do koreňa hodnota $(3, 5, 4)$ na základe ktorej vieme povedať, že prvý hráč bude mať 3 alebo viac bodov a jeho dvaja súperi budú mať nanajvýš 9 bodov. Po vypočítaní hodnoty $(1, 10, 1)$ ľavého listu pravého podstromu sa dostane do vrchola C informácia, že druhý hráč v ňom získa aspoň 10 bodov a jeho súperi nanajvýš 2 body. Vďaka tejto informácii vieme povedať, že hodnota z vrchola C sa do koreňa nedostane, pretože v koreni hovoríme, že prvý hráč získa aspoň 3 body. Preto si môžeme dovoliť orezať vrchol G a za výsledok prehľadávania stromu prehlásiť $(3, 5, 4)$. Môžeme vidieť, že prvý hráč uprednostnil $(3, 5, 4)$ kde skončil na poslednom mieste pred $(1, 10, 1)$ kde by obsadil delené druhé miesto. Vrchol G mohol navyše mať hodnotu $(1, 11, 0)$ ktorá by sa dostala až do koreňa.

Kvôli neuspokojivým výsledkom použitia orezávania zo spomínaného článku sme pristúpili k problému novou cestou. Budeme porovnávať poradie hráčov. Až v prípade rovnosti budú rozhodovať získané body.

Pozrime sa na obrázok 3.2 kde si môžeme dovoliť orezať vrchol C. Vo vrchole A vyhráva prvý hráč a keďže v koreni je práve prvý hráč na ťahu, do koreňa sa vyberie pravá vetva jedine v prípade ak v nej vyhrá prvý hráč.

Vo vrchole B vyhráva druhý hráč a v rodičovi tohto vrchola je na ťahu práve druhý hráč. Tým pádom vieme povedať, že stav hry z vrchola C bude vybratý jedine vtedy ak v ňom vyhrá druhý hráč. No v takom prípade nebude vybratá do koreňa stromu, preto si môžeme dovoliť tento vrchol orezať.

Môžeme vidieť, že vrchol C môžeme orezať keď vo vrchole B vyhráva druhý hráč a vo vrchole A vyhráva prvý hráč. Navyše, vrchol C vieme orezať aj v prípade že vo vrchole B si druhý hráč prvenstvo delí, resp.ak si vo vrchole A prvý hráč prvenstvo delí.



Obr. 3.2: Orezávanie stromu na základe pozície hráča

Tvrdenie 3.1.1 (Orezávanie v hrách s 3 a viac hráčmi)**.** *Majme vrchol G, ktorého rodič je vrchol B a prarodič vrchol A. Nech je vo vrchole B na ťahu hráč 2 a vo vrchole A na ťahu hráč 1 (obrázok 3.2). Vrchol G môžeme orezať ak platí jedna z nasledovných podmienok:*

1. *Vo vrchole A vyhráva hráč 1 a vo vrchole B vyhráva hráč 2 alebo si prvenstvo delí s jedným z jeho súperov*

2. *Vo vrchole A vyhráva hráč 1 alebo si prvenstvo delí s jedným z jeho súperov a vo vrchole B vyhráva hráč 2*

Dôkaz. Nech v koreni stromu je na ťahu hráč 1 a v jeho pravom synovi hráč 2. Vo vrchole A vyhráva prvý hráč a vo vrchole B vyhráva druhý hráč. Ukážeme, že pre ľubovoľnú hodnotu stavu hry vo vrchole C bude do koreňa vybratý stav hry z vrchola A.

1. Ak bude vo vrchole C vyhrávať druhý hráč a tento stav bude vybratý do jeho rodiča, do koreňa sa dostane hodnota z vrchola A

2. Ak vo vrchole C bude iná hodnota, rodič uprednostní hodnotu z vrchola B, keďže v nej vyhráva druhý hráč. No aj v tomto prípade bude do koreňa vybratá hodnota z vrchola A

Ukázali sme, že nezáleží na hodnote vo vrchole C a preto si môžeme dovoliť celý výpočet tohto vrchola (a príp. jeho podstromu) vynechať. □

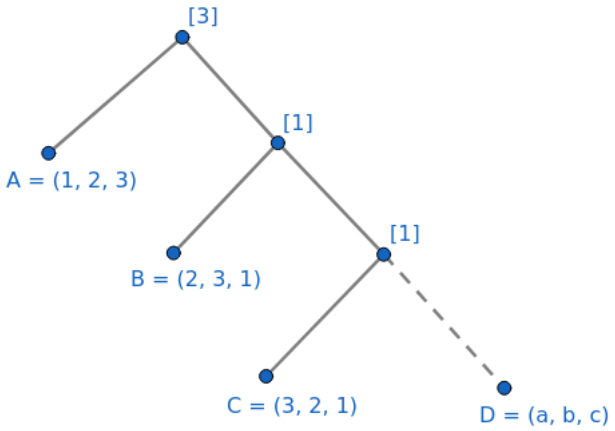
Tento poznatok môžeme využiť v našej práci, avšak javí sa tu jeden možný problém: po akcii *Áno prosím* zostáva ten istý hráč na ťahu, preto ak najprv prehľadáme možnosť *Nie ďakujem*, dostaneme sa do situácie, kedy v pravom synovi koreňa stromu z obrázku 3.2 bude na ťahu ten istý hráč ako v koreni stromu. To ale znamená, že vrchol C nebudeme môcť orezať, pretože môže mať pre tohto hráča lepšiu hodnotu ako vrcholy B alebo A. Všimnime si, že zmena hráča na ťahu je pre techniku orezávania kľúčová. Má preto zmysel vôbec prehľadávať akciu *Nie ďakujem* ako prvú? Pri orezávaní platí zásada, že pre najväčšiu efektivitu treba ako prvé prezrieť optimálne ťahy. Má ale v tomto prípade zmysel pozerať optimálny ťah, ak by išlo o ťah *Nie ďakujem*?

3.2 Orezávanie v hrách s opakovaným ťahom hráča

V hrách kde môže jeden hráč odohrať viac ako jeden ťah za sebou vieme použiť orezávanie podobným spôsobom ako doteraz, ale nestačí sa pozerať na situáciu v rodičovi, keďže v ňom môže byť na ťahu ten istý hráč a orezávanie popísané v predošlej podkapitole bude môcť spôsobiť zmenu výsledku, čo je zakázané. Prišli sme preto s úpravou predchádzajúceho tvrdenia 3.1.1 na všeobecnejšie, ktoré pokryje aj hry kde môže hrať ten istý hráč viac ťahov za sebou. V takom prípade sa budeme v strome pozerať na najbližšieho predka ku ktorému vedie hrana v ktorej sa menil hráč na ťahu.

Všimnime si, že na obrázku 3.3 vieme orezať hranu vedúcu k vrcholu D. Všimnime si, že nakoľko sú vrcholy B, C, D synmi ťahov hráča 1, bude z nich vybratý ten, kde stav hry je najpriaznivejší pre tohto hráča. Keďže ale pred hráčom 1 bol na ťahu hráč 3 ktorý zo svojho ľavého podstromu vypočítal hodnotu v ktorej vyhráva, akonáhle hráč 1 vypočíta v niektorom z jeho synov stav kde vyhráva hráč 1, zvyšok synov môže orezať.

Tvrdenie 3.1.1 vieme zovšeobecniť nasledovne:



Obr. 3.3: Orezávanie stromu pri viacerých ťahoch hráča za sebou

Tvrdenie 3.2.1 (Orezávanie v hrách s 3 a viac hráčmi)**.** *Majme vrchol G, ktorého rodič je vrchol B a vrchol A je jeho predkom. Nech je vo vrchole A na ťahu hráč* 1*, vo vrchole B na ťahu hráč* 2 *a vo všetkých zvyšných vrcholoch na ceste medzi vrcholmi A, B je na ťahu tiež hráč* 2*. Vrchol G môžeme orezať ak platí jedna z nasledovných podmienok:*

1. *Vo vrchole A vyhráva hráč* 1 *a vo vrchole B vyhráva hráč* 2 *alebo si prvenstvo delí s jedným z jeho súperov*

2. *Vo vrchole A vyhráva hráč* 1 *alebo si prvenstvo delí s jedným z jeho súperov a vo vrchole B vyhráva hráč* 2

Dôkaz tohto tvrdenia je takmer identický ako pri pôvodnom tvrdení, takže ho vynecháme.

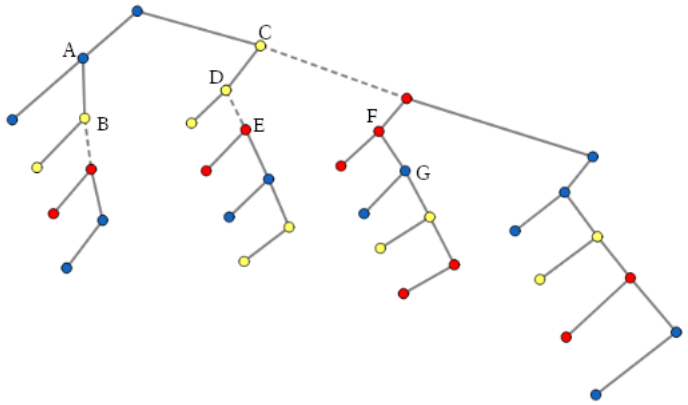
3.3 Orezávanie v nerovnomernom strome

V hrách kde nie je vopred známe koľko ťahov sa vykoná a rôzne ťahy hráčov môžu viesť k skráteniu, resp.predĺženiu hry znamená, že strom ktorým simulujeme rôzne priebehy hry bude nevyvážený. V našej práci sme si všimli, že strom bude v niektorých častiach až 15-násobne hlbší a zrejme nebude úplne efektívne prehľadávať tieto hlboké časti stromu pokiaľ to nie je nutné. Skôr naopak.

Čím viac budú hráči hrať akciu *Áno prosím*, tým viac bude z balíčku kariet ubúdať a teda sa rýchlejšie priblíži koniec hry. A naopak, akciou *Nie ďakujem* sa hra naťahuje.

Ak by bola akcia *Áno prosím* v simulovanom strome vždy ľavý syn vrcholov a pravým synom by bola akcia *Nie ďakujem*, strom bude v ľavej časti najplytší (bude mať najmenšiu vzdialenosť od koreňa k listom) a smerom k pravému kraju stromu sa bude jeho hĺbka zväčšovať.

Všeobecne platí, že pre najefektívnejšie orezávanie je najvýhodnejšie najprv prezerať najlepšie možné ťahy - vtedy sa nám bude dariť orezávať najčastejšie. Avšak vybrať optimálny ťah v danej situácii vôbec nie je ľahká záležitosť a veľakrát sa stane, že sa nám to nepodarí. Ak by sme boli schopní vyhodnotiť ktorá vetva z danej dvojice je lepšia, nemuseli by sme simulovať rôzne priebehy hry a rovno by sme sa rozhodli akú akciu zahrať.



Obr. 3.4: Ukážka orezávania v nevyváženom strome

Porovnajme si potenciál orezávania v nevyváženom strome kedy najprv prezeráme plytšiu časť stromu s prípadom kedy najprv prezeráme hlbšiu časť stromu. Pre jednoduchosť príkladu majme akcie T, P, kde akciou T sa rýchlejšie blížime ku koncu hry. Akciu P ale možno vykonať nanajvýš tri razy po sebe. Týmto sa snažíme napodobniť akciu *Nie ďakujem*, keďže aj tá je obmedzená hráčovými žetónmi, resp.ak je pri karte toľko žetónov aká je hodnota karty, nemá zmysel hrať túto akciu. Na obrázku 3.4 je ukážka hry troch hráčov (modrý, žltý, červený), kedy po dvoch akciách T nastane koniec hry. Po akcii P sa mení hráč na ťahu.

Pokiaľ prechádzame strom zľava doprava, čiže z dvojice akcií T, P prezrieme ako prvú akciu T, orezanie môže v optimistickom prípade nastať až trikrát. Vo vrcholoch B, C môže nastať orezanie v prípade, že v jeho ľavom synovi vyhráva žltý hráč a v ľavom synovi vrchola A vyhráva modrý hráč. Vo vrchole D je obdobná podmienka, avšak jeho najbližší predok kde je na ťahu iný hráč nie je vrchol C, ale koreň stromu. Ak teda vo vrchole A vyhráva modrý hráč a v ľavom synovi vrchola D vyhráva žltý hráč, pravý podstrom vrchola D môžeme orezať.

Všimnime si, že pre splnenie podmienky pre orezanie podstromu museli mať vrcholy B, D, C najprv prezretého svojho ľavého syna a navyše aj rodičia vrcholov B, D, C museli mať prezretý svoj ľavý podstrom. Keby sme strom prechádzali sprava doľava, tak prvé orezanie by mohlo nastať vo vrchole G kedy sme už prehľadali jeho pravý podstrom a aj jeho najbližší predok - červený hráč, prehľadal svoje pravé podstromy. Keďže

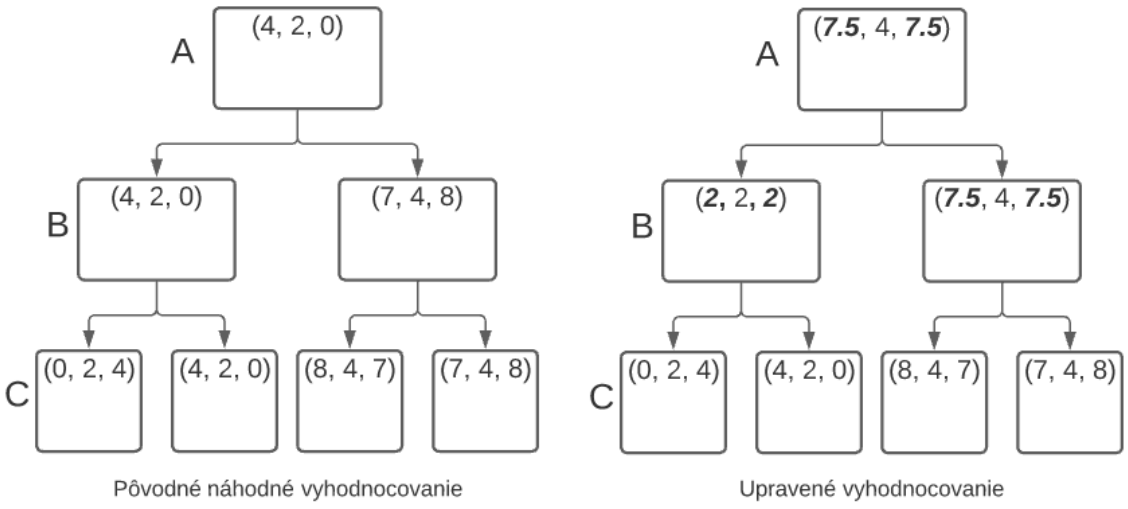
červený hráč bol na ťahu vo vrchole F a aj v jeho rodičovi, vrchol F má informácie z prehľadaného pravého podstromu jeho rodiča. Preto v prípade, že v F je informácia že vyhráva červený a vo vrchole G je informácia že vyhráva modrý, môžeme si dovoliť orezať ľavého syna vrchola G. Podobnú situáciu môžeme nájsť aj v prípade vrcholov D, E či A, B.

Môžeme vidieť, že orezaní je v oboch prípadoch rovnako veľa, no v prvom prípade orezávame veľké podstromy, zatiaľ čo v druhom prípade sú orezané podstromy oveľa menšie.

Na tomto príklade sme si uviedli dva optimistické prípady kde sme zistili, že prezeranie akcie *Áno prosím* ako prvej má oveľa väčšie maximum orezania ako v opačnom prípade. V kapitole 5 sme tieto dva prístupy porovnali a zistili sme, že prezerať akciu *Áno prosím* sa vyplatí aj v priemernom prípade.

3.4 Ako zmazať rozdiely medzi simuláciou a realitou

Tento stav je pre nás neuspokojivý. Nevieme zaručiť aby hráč pri neoptimálnom hraní skončil horšie ako hráč, ktorý hra optimálne, no chceli by sme dosiahnuť aspoň záruku toho, že pri optimálnom hraní všetkých hráčov bude priemerný výsledok lepší.



Obr. 3.5: Porovnanie pôvodného a upraveného vyhodnocovania simulácie

Pozrime sa na situáciu na obrázku vľavo. Najľavejší syn má hodnotu (0, 2, 4), pre hráča A je to najhoršia hodnota zo všetkých listov. Syn druhý zľava má hodnotu (4, 2, 0) a pre hráča A je to najlepšia hodnota zo všetkých listov. Pre hráča B sú tieto dva listy rovnocenné, preto náhodne vyberie jednu z nich. Hráč A preto v polovici prípadoch skončí na prvom mieste a v polovici prípadoch na poslednom mieste. Záležať bude ktorú z dvoch možností hráč B náhodne vyberie.

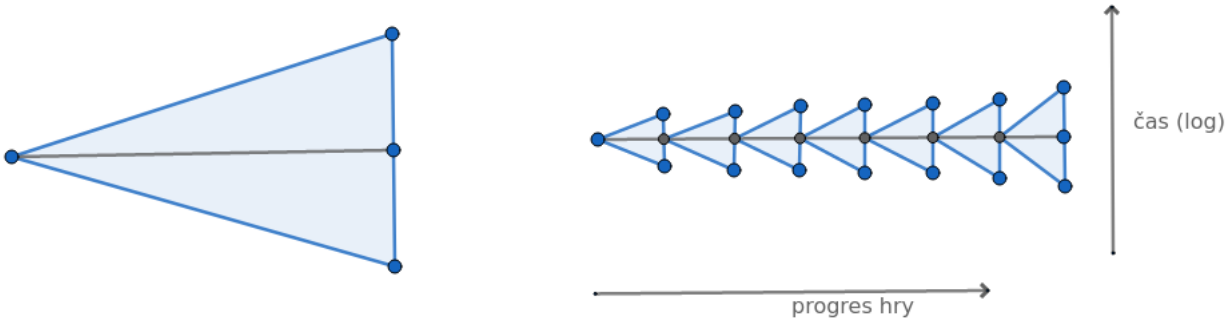
Na obrázku vpravo vidíme ako túto situáciu opraviť. V prípade, že hráč má na výber dve identické situácie, spriemeruje ich a túto hodnotu pošle rodičovi. Hráč A takto vidí, že je pre neho lepšia situácia v pravom podstrome. Keď si vyberie ten, v polovici prípadoch skončí na prvom mieste a v polovici prípadoch na druhom mieste. Úplne prehľadávanie je najsilnejší nástroj ktorý máme, avšak jeho použitie je príliš drahé. V ďalších kapitolách si ukážeme, ako ho vieme urýchliť, avšak stále budeme nútení stlačiť čas aj za cenu nepresností. Cieľom bude docieliť, aby tieto nepresnosti boli čo najmenšie.

Kapitola 4

Vytváranie stratégie

Naším cieľom je čo najviac sa priblížiť optimálnemu hraniu. To by sme vedeli dosiahnuť úplným prehľadávaním, avšak to nám vie zabrať nesmierne veľké množstvo času. V každom ťahu má hráč na výber 2 možnosti (až na okrajové prípady kedy nemá žetóny), takže strom rastie exponenciálne. Na obrázku 4.1 vľavo je zobrazený nárast časovej náročnosti prehľadávania celého stromu.

Pre dosiahnutie rýchleho ťahu počítača sa musíme vzdať úplneho prehľadávania a namiesto toho prehľadávať iba čiastočne, napr.len do určitej hĺbky stromu. Po dosiahnutí danej hĺbky od koreňa by sme sa pokúsili čo najlepšie odhadnúť ako by hra pri optimálnom hraní skončila. Ak by sme v polovici hry (napr.po zobratí dvanástej karty akciou *Áno prosím*) vedeli ohodnotiť stav hry, tak prvú polovicu hry by sme prehľadávali strom iba po dvanástu kartu kde by sme zistili ohodnotenie stavu a v druhej polovici hry by sme prehľadávali strom až po jeho koniec - keďže sme si dovolili na začiatku hry prehľadávať polovicu hry, nebude problém prehľadávať hru od polovice hry až do konca.



Obr. 4.1: Rozdelenie hry na viacero úsekov

Potrebujeme dosiahnuť, aby ťah počítača trval nanajvýš niekoľko sekúnd. Otázkou zostáva, či prehľadávanie polovice hry trvá krátko, alebo to stále prevyšuje naše časové požiadavky. Hru by sme mohli rozdeliť aj na viac úsekov ako dva, avšak každým roz-

delením úsekov sa vzdávame úplného prehľadávania a teda určitej presnosti. Preto by nebolo vhodné hru rozdeliť na viac úsekov ako je potrebné, ale iba na minimum, ktoré spĺňa naše časové požiadavky.

Na obrázku 4.1 vpravo je znázornené rozdelenie na sedem úsekov. Všimnime si že prehľadávanie menších úsekev trvá menej času. Pozn: zvislá os znázorňujúca čas je logaritmická, preto je súčet časov prehľadávania menších úsekov v skutočnosti menší ako čas prehľadávania jedného veľkého úseku.

4.1 Parametre hry v datasetoch

Cieľom tejto podkapitoly je pozrieť sa na konfiguráciu hry a zjednodušiť ju tak, aby sme vedeli rýchlejšie a priamočiarejšie hľadať najbližší stav v datasete, no zároveň aby sme nevynechali podstatný aspekt.

V hre *No Thanks* je počas hry známe nasledovné:

- karty každého hráča
- počet žetónov každého hráča
- karta v strede stola
- žetóny v strede stola
- poradie kariet ktoré ešte prídu do hry (vo verzii so známou budúcnosťou)

Karty

Aby sme vedeli hľadať najbližšieho suseda v datasete, mali by sme mať karty hráča uložené ako konštantný počet hodnôt. Nebudeme si pamätať všetky karty ktoré hráč zobral, namiesto toho si uložíme ako prvý parameter súčet hodnôt jeho vedúcich kariet.

Ak hráč zobral kartu 10, je pre neho lepšie ak karta 11 nie je vyradená z balíčka, pretože táto karta mu nedá žiadne záporné body, naopak, môže ňou získať žetóny. Preto ďalší z parametrov ktorý si zaznamenáme bude počet kariet ktoré do hry prídu a hráčovi rozšíria sekvenciu alebo spoja dve sekvencie dokopy.

Keďže je veľký bodový rozdiel v tom či sa spoja sekvencie vysokých hodnôt alebo nízkych, ďalší parameter ktorý si zapamätáme bude súčet kariet ktoré hráčovi rozšíria sekvenciu.

Ak má hráč vedúce karty $10, 20$, jeho šanca že tieto sekvencie spojí je veľmi malá, zatiaľ čo ak má hráč $14, 16$, stačí aby karta 15 nebola vyradená z balíčka a je veľká šanca že sa tomuto hráčovi podarí spojiť svoje dve sekvencie. Preto ďalší parameter ktorý si zapamätáme bude počet sekvencií ktoré vie hráč spojiť kartami ktoré ešte prídu do hry.

Ktoré karty presne prídu do hry ani ich poradie si nijako nebudeme zaznamenávať, pretože to by značne skomplikovalo operácie v našom datasete.

Žetóny

Ďalší parameter v stave hry budú žetóny jednotlivých hráčov. Datasety vytvárame na hraniciach úsekov - vždy po tom ako je zobratá karta akciou *Áno prosím*, preto v strede stola nebude žiaden žetón a súčet žetónov hráčov bude 33. Takto si vieme zaznamenať rozdiely žetónov a teda budú nám stačiť dve čísla namiesto troch. Príklad: namiesto $(10, 11, 12)$ si zapamätáme iba $(1, 2)$.

Celkovo teda bude mať hra 12 parametrov 5 kategórií

1. vedúce karty
    - rozdiel súčtu vedúcich kariet prvého a druhého hráča
    - rozdiel súčtu vedúcich kariet prvého a tretieho hráča
2. počet rozširujúcich kariet (pre každého hráča zvlášť)
3. hodnoty rozširujúcich kariet
    - rozdiel súčtu rozširujúcich kariet prvého a druhého hráča
    - rozdiel súčtu rozširujúcich kariet prvého a tretieho hráča
4. žetóny
    - rozdiel počtu žetónov prvého a druhého hráča
    - rozdiel počtu žetónov prvého a tretieho hráča
5. počet sekvencií ktoré vie hráč zredukovať, resp. o koľko vie hráč zmenšiť počet svojich vedúcich kariet (pre každého hráča)

4.2 Generovanie dát pre úseky hry

Úsekové dáta hry budeme vytvárať od konca nasledovne: Daný počet ťahov od konca si vygenerujeme náhodnú konfiguráciu hry a z nej odsimulujeme úplným prehľadávaním optimálne hranie. Na konci hry zistíme výsledné skóre a tým pádom získame hodnotu konfigurácie ktorú sme si vygenerovali. Keďže v hre *No Thanks* nie je vopred známe koľko ťahov bude hra trvať ale iba koľko je kariet v balíčku, budeme pre potreby tejto podkapitoly rozumieť dĺžkou úsekov počet kariet ktoré sa z balíčka neodoberú. Súčet dĺžok úsekov bude dokopy 24 keďže toľko kariet sa dostane do hry.

Vygenerovaných hier budeme potrebovať dostatočné množstvo, aby sa najbližší nájdený prvok v tomto datasete čo najviac podobal aktuálnemu stavu hry a vedeli sme podľa neho získať čo najvernejšie ohodnotenie stavu hry.

4.2.1 Veľkosti úsekov

Vyskúšali sme generovať úseky rôznych dĺžok, ale ako najlepšia možnosť sa nám javila začať od konca úsekom dĺžky 6. Odpoveď počítača dostaneme v prijateľnom čase, čo už by sa nedalo povedať pri dĺhších úsekoch. Ak sa hra nachádza v poslednom úseku, počítač pre ohodnotenie stavu nepotrebuje použiť žiaden dataset, ale odsimuluje si hru až do konca a hodnota stavu na konci hry je jej výsledok. Dataset ktorý sa vždy nachádza medzi dvomi úsekmi je využívaný vtedy keď sa hra nachádza v ľavom úseku z týchto dvoch.

Podobným spôsobom ako sme vygenerovali posledný dataset vygenerujeme predposledný dataset, avšak v tomto prípade nebudeme generovať hru až do konca hry, ale v momente keď sa v prehľadávaní hry dostaneme do situácie kedy sa otočí šiesta karta odspodu balíčka sa pozrieme na prislúchajúci dataset a nájdeme v ňom najbližšieho suseda podľa štrukúry kd-stromu.

Ukázalo sa, že vyhľadávanie najbližšieho suseda v kd-strome pri určitej veľkosti datasetu nie je pre naše potreby časovo triviálna operácia a pri prehľadávaní úsekov veľkosti 6 je čas ťahu počítača neuspokojivo veľký. Preto sme zvolili úseky zmenšiť až na veľkosť 3. Pri generovaní rôznych stratégií a ich porovnávaní bude síce generovanie menších úsekov rýchlejšie oproti generovaniu posledného úseku veľkosti 6, avšak pri porovnávaní rôznych stratégií budeme z časových dôvodov používať menší počet hier a z tohto dôvodu bude aj kd-strom menší. Vo finálnej verzii konečnej stratégie bude dataset značne väčší a vtedy prehľadávanie úseku veľkosti 3 sa v kombinácii s vyhľadávaním najbližšieho prvku v kd-strome priblíži času prehľadávania úseku veľkosti 6.

Celkovo budeme mať teda 7 úsekov - prvých šesť veľkosti 3 a posledný veľkosti 6.

4.2.2 Počet hier pre generovanie datasetov úsekov

Aby sme dosiahli čo najpresnejšie ohodnocovanie stavu hry, potrebujeme vygenerovať dostatočne veľké množstvo hier pre vytváranie datasetov úsekov, aby vzdialenosti medzi prvkami boli čo najmenšie a najbližší nájdený prvok bol blízko. Keďže však proces generovania datasetov nevytvára finálny dataset ale slúži iba na určenie smeru ktorým sa posúvať k optimálnej stratégii, nebudeme míňať zbytočne veľké množstvo času na generovanie obrovských datasetov napriek tomu, že sa nemusíme pohnúť najlepším možným smerom. Tento prípad môže nastať, ale keďže tento proces budeme robiť opakovane, je pravdepodobné že po viacej krokoch sa k optimálnej stratégii viac priblížime ako oddialime.

Aby sme sa pozreli aké výsledky nám prinášajú datasety založené na rôznom počte hier, porovnali sme priemernú vzdialenosť náhodného stavu hry k najbližšiemu prvku v kd-strome daného datasetu.

počet hier	100	200	400	800	1600	3200	6400	12800
najbližší sused	48.21	41.31	36.48	31.75	27.86	24.55	21.74	19.30

V tabuľke môžeme vidieť, že s väčšími datasetmi dosahujeme lepšie výsledky v hľadaní najbližšieho suseda. Čísla v druhom riadku hovoria o euklidovskej vzdialenosti najbližšieho suseda k počítanému stavu.

Môžeme si všimnúť, že počet hier z ktorých boli datasety generované stúpa exponenciálne, ale vzdialenosť k najbližšiemu susedovi nie je v dvojnásobne väčšom datasete dvakrát menšia. To je spôsobené tým, že generujeme datasety náhodne, preto diery medzi stavmi častokrát nebudú rozdelené rovnomerne.

V nasledovnej tabuľke si môžeme všimnúť aj porovnanie týchto stratégií priamo v hrách. Spôsobu porovnávania sa budeme venovať viac v ďalšej podkapitole, teraz nám stačí si všimnúť priemerné poradie porovnávaných datasetov. V prvom stĺpci si môžeme všimnúť porovnanie datasetu veľkostí 100 a 200. Vidíme, že väčší dataset dosiahol o niečo lepší priemerný výsledok.

veľkosť datasetu	100	200	400	800	1600	3200	6400
priemerné poradie	2.051	2.043	1.95	2.018	2.1175	2.032	1.961
veľkosť datasetu	200	400	800	1600	3200	6400	12800
priemerné poradie	1.949	1.957	2.05	1.982	1.8825	1.968	2.039

Všimnime si tiež istú nekonzistenciu pri porovnávaní datasetov veľkostí 400 a 800 alebo datasetov veľkostí 6400 a 12800 kedy menší dataset porazil väčší. Tieto zvláštnosti mohli byť spôsobené dvomi faktormi. Prvý z nich je nedostatočný počet hier na ktorých sme ich porovnávali. Druhý z nich je generovanie samotných datasetov. Keďže ich generujeme náhodne, isté nepresnosti tam môžu vzniknúť. Je možné, že pri opätovnom vygenerovaní datasetov týchto veľkostí by sme získali o niečo iné výsledky. Zatiaľ čo pri datasetoch veľkosti 400 a 800 je to veľmi možné, pri datasete veľkosti 12800 už menej. Tu nám však výsledok môže napovedať aj to, že generovanie datasetov takýchto veľkostí nie je pre výsledok už veľmi efektívne.

Z dosiahnutých výsledkov sme sa rozhodli pre fázu generovania používať najprv datasety veľkosti 1000. V neskoršej fáze generovania toto číslo zväčšíme na 2000, resp. 4000 hier.

4.3 Váhy generovaných stratégií

Generovanie náhodných ováhovaní parametrov si vyžaduje určité ohraničenie, aby porovnávané parametre neboli celkom iné. Na to nám bude slúžiť parameter ϵ ktorý nám určí interval z ktorého vygenerujeme náhodný činiteľ pôvodnej hodnoty parametra. Začíname so základnou stratégiou ktorá má nastavené váhy $(1, 1, 1, 1, 1)$. Stratégiam,

ktoré vygenerujeme na porovnanie s našou základnou stratégiou ováhujeme parametre nasledovne: pre každý parameter vygenerujem náhodné číslo z intervalu $[\frac{1}{\epsilon}, \epsilon]$ ktorým vynásobíme daný parameter našej základnej stratégie. Interval sme navrhli takto preto, aby jeho horná a spodná hranica boli inverzným prvkom operácie násobenia. Inými slovami, aby sme v priemernom prípade podobne čísla zväčšovali a zmenšovali.

Na začiatku môžeme zvoliť ϵ relatívne väčšie, až $\epsilon = 2$ a generovaním číslo nanajvýš desaťkrát zmenšíme alebo desaťkrát zväčšíme. Neskôr môžeme ϵ postupne zmenšovať aby sme k hľadanému minimu smerovali postupnými a jemnejšími krokmi.

Na základe porovnávania stratégií ktoré si popíšeme bližšie v ďalšej podkapitole určíme nové ováhovanie našej základnej stratégie a postup opakujeme až pokým sa ováhovanie parametrov neustáli.

Menenie váhy po jednom parametri

K problému ováhovania parametrov sme najprv pristupovali inou cestou. Nemenili sme všetky parametre naraz o náhodné vygenerované číslo, ale menili sme iba jeden a snažili sme sa nájsť ideálnu váhu tohto parametra.

Postup bol nasledovný: Začneme s ováhovaním (1.0, 1.0, 1.0, 1.0, 1.0). Vyberieme si jeden parameter a nastavíme ϵ . Vytvoríme 4 iné stratégie kde má vybraný parameter s pôvodnou hodnotou h upravené hodnoty $h \times \epsilon^2, h \times \epsilon, h \times \epsilon^{-1}, h \times \epsilon^{-2}$. Vytvorené stratégie navzájom porovnáme a tá stratégia, ktorá získala najlepšie skóre nám určí hodnotu parametra. Postup opakujeme s tým, že ováhovanie aktualizujeme a ϵ zmenšíme o polovicu.

Po niekoľkých iteráciách sa môžeme posunúť na ďalší parameter.

Rozhodli sme sa, že od tohto postupu upustíme, keďže sa nám zdal pomalý a okrem toho ťažšie sa v ňom ukážu vzťahy medzi určitými parametrami. Ak existuje súvislosť medzi dvomi parametrami, tento postup nám ju neukáže, pretože vždy z dvojice parametrov je jeden nemenný.

4.4 Porovnanie dvoch stratégií

Aby sme vedeli povedať ktorá zo stratégií S_1, S_2 je lepšia, budeme ich porovnávať v simulovaných hrách. Môže však zavážiť v prospech jednej či druhej stratégie, akú stratégiu S_3 má tretí hráč. Generovanie rôznych stratégií tretieho hráča a porovnávanie s S_1 a S_2 by prinieslo časový nárast porovnávania, preto zvolíme pre tretieho hráča jednu zo stratégií S_1, S_2. Keďže voľba stratégie S_3 môže ovplyvniť výsledok, tú istú hru spustíme dvakrát, pričom raz bude $S_3 = S_1$ a raz $S_3 = S_2$. Takto dosiahneme, že situácia bude pre stratégie S_1, S_2 symetrická.

4.4.1 Spôsob porovnania

Keďže potrebujeme pre hru troch hráčov, aby sme stratégie porovnali symetricky, tú istú inštanciu hry spustíme dvakrát. Prvý raz s dvomi kópiami stratégie A a jednou stratégiou B a druhýkrát opačne.

Vo výsledku nás zaujíma konečné poradie hráčov s danou stratégiou. Keďže v polovici hier sa vyskytnú až dvaja hráči s rovnakou stratégiou, najlepšie priemerné poradie stratégie je 1.333 a najhoršie priemerné poradie stratégie je 2.667.

4.4.2 Počet hier na porovnanie stratégií

počet hier	2	4	8	16	32	64	128	256	512	1024	2048
stratégia A	2.0	2.0	1.917	1.854	1.938	2.0	1.961	2.017	2.023	2.026	2.028
stratégia B	2.0	2.0	2.083	2.146	2.062	2.0	2.039	1.983	1.977	1.974	1.972

Častokrát sa môže stať, že v dôležitom bode hry kedy treba spraviť kľúčové rozhodnutie sa prikloní šťastie k niektorému hráčovi tým, že to ako je zamiešaný balíček, resp. to ako sa vyvíja hra je vygenerované v datasete jeho stratégie. To môže viesť k tomu, že aj horšia stratégia môže poraziť lepšiu. Preto je ideálne náhodu eliminovať tým, že stratégie budeme porovnávať na väčšom počte hier.

V nasledovnej tabuľke sme porovnávali 2 stratégie kde stratégia B sa od stratégie A líšila tým, že váhu posledného parametra popisujúceho redukovanie sekvencií (4.1) má nastavenú na 8. Posledný parameter hovorí o počte sekvencií ktoré ešte hráč môže do konca hry zredukovať. Redukcia sekvencie dokáže dať hráčovi veľa bodov, preto je veľký rozdiel ak hráč vie zredukovať o jednu sekvenciu viac alebo menej. Preto môžeme predpokladať, že tento parameter má pre hodnotu stavu väčšiu váhu ako ostatné. Aké ováhovanie je ideálne bude predmetom ďalšej podkapitoly

V tabuľke môžeme vidieť, že výsledok porovnania týchto dvoch stratégií bol po odohratí 256 hier vždy v prospech stratégie B. Predpokladáme, že so stále zväčšujúcim sa počtom hier by stratégia B stále dosahovala priemernú pozíciu okolo 1.97. Vďaka tomuto výsledku sme sa rozhodli, že stratégie budeme porovnávať na 400 hrách.

4.4.3 Striktný výber z množiny stratégií

Prvý spôsob porovnávania stratégií sme navrhli tzv.*striktný výber*. V tomto prípade porovnáme niekoľko stratégií medzi sebou a tá, ktorá dosiahne najlepšie výsledky sa dostane do ďalšieho kola porovnávania s náhodnou množinou vygenerovaných stratégií.

Predpokladáme, že porovnanie stratégií je vo všeobecnosti tranzitívne. Teda ak stretégia A vyhrá nad stratégiou B a stratégia B vyhrá nad stratégiou C, bez toho aby sme museli robiť ďalšie porovnávanie vieme vyhlásiť, že stratégia A vyhrá nad

stratégiou C. Nemusí to platiť vdžy, ale akceptujeme aj občasné nepresnosti s cieľom skrátiť čas porovnávania.

Ak budeme porovnávať 5 stratégií, nemusíme porovnávať všetky navzájom, ale každú stratégiou porovnáme iba s dvomi inými, prvú s druhou, druhú s treťou, až nakoniec uzavrieme cyklus porovnaním piatej s prvou. Poslednú dvojicu by sme mohli vynechať, ale to by musela platiť tranzitivita porovnávania stopercentne. Preto radšej porovnáme každú stratégiu s dvomi inými, lebo malým nepresnostiam sa nevyhneme a takto o niečo presnejšie určíme ktorá stratégia je z pätice najlepšia.

Zároveň si tak môžeme overiť nakoľko funguje tranzitivita, resp. nakoľko je porovnanie presné. Zoberme si porovnanie stratégií A, B, C s nasledovnými výsledkami:

- A=1.5 vs B=2.5
- B=1.9 vs C=2.1
- C=2.6 vs A=1.4

Vidíme, že v takomto prípade tranzitivita funguje. V prvom prípade stratégia A s priemernou pozíciou 1.5 jednoznačne poráža stratégiu B. Stratégia B vyhráva nad stratégiou C s priemernou pozíciou 1.9. Môžeme tak očakávať, že stratégia A porazí stratégiu C ešte výraznejšie ako porazila stratégiu B, čo sa v tomto prípade aj stalo.

Pre potreby odmerania nepresnosti tranzitivity porovnávania stratégií si zadefinujeme číslo ϕ ktoré bude hovoriť o nepresnosti.

Definícia 4:
Zoraďme si všetky dvojice porovnávaných stratégií: $(S_1, S_2), (S_2, S_3), ..., (S_n, S_1)$. Označme P priemerné umiestnenie v hre. Zoberme prvé členy z každej porovnávanej dvojice stratégií a ich výsledky z porovnania $R_1, R_2, ..., R_n$. Nepresnosť vypočítame ako

$$\phi = |P - \frac{\sum_{i=1}^{n} R_i}{n}|.$$

V prípade hry troch hráčov je priemerné umiestnenie $P = 2$.

Nepresnosť hovorí o tom, aká bola priemerná chyba pri porovnaní dvoch stratégií. Ak by všetky porovnania skončili remízou, čiže $R_1 = R_2 = R_3 = R_4 = R_5 = 2$, znamenalo by to, že stratégie sú vyrovnané a nepresnosť bola nulová.

Pozrime sa ale na prípad, kedy vždy prvá z dvojice stratégií získa skóre 1.9. Pripomína nám to hru *Kameň, papier, nožnice*, kedy nevieme určiť ktorá zo stratégií je najlepšia, pretože všetky dosiahli ten istý výsledok. Preto sú pre nás vyrovnané. Avšak každé porovnanie so skóre 1.9 resp. 2.1 bolo od vyrovnaného porovnania vzdialené 0.1.

Pri predošlom prípade by boli hodnoty $1.5, 1.9, 2.6$. Ich priemerná hodnota 2 a teda nepresnosť je v tomto prípade nulová. Ak by ale stratégia C mala proti stratégii A priemernú pozíciu 2.3, nepresnosť by bola $\phi = |2 - \frac{1.5+1.9+2.3}{3}| = |2 - \frac{5.7}{3}| = |2 - 1.9| = 0.1$

Výber najlepšej stratégie

Každá stratégia získa z porovnávania nejaké skóre R_i. Na ukázanie postupu prideľovania skóre použijeme príklad so stratégiami A, B, C. V prípade že dve stratégie pri porovnaní dosiahnu výsledok 2.0, obe stratégie budú mať rovnaké skóre. Vo zvyšných prípadoch bude lepšia z dvojice stratégií mať o toľko lepšie od porovnávanej, o koľko bola jej pozícia lepšia od priemeru (2.0), a naopak.

Ak stratégia A má základné skóre 0, keďže stratégia B pri porovnaní s A dosiahla výsledok 2.5, skóre stratégie B bude o 0.5 horšie od A, čiže -0.5. Podobne vypočítame skóre stratégie C: keďže v porovnaní so stratégiou B dosiahla výsledok 2.1, bude skóre stratégie C o 0.1 horšie od skóre stratégie B, teda -0.6.

Ďalej sa pozrime na porovnanie stratégií C a A. Stratégia A dosiahla výsledok 1.4, teda od stratégie C bude mať skóre o 0.6 lepšie, čiže 0. Všimnime si, že nám vyšiel náš predpoklad, no v prípade že je nepresnosť nenulová, dostaneme sa do situácie, kedy skóre stratégie z ktorej sme odvodili skóre ostatných stratégií nesedí s predpokladom.

Uvažujme prípad, že porovnanie stratégie A so stratégiou C skončí v prospech C s výsledkom 1.8. To znamená, že skóre stratégie A by malo byť o 0.2 horšie od stratégie C, čiže -0.8. Ktorá stratégia je z tejto trojice teda najlepšia?

Aby sme neprišli k rozporuplným či nepresným výsledkom, posledné porovnanie stratégií A a C do rátania skóre vynecháme. Aby sme spravodlivo vypočítali skóre pre každú z n stratégií, budeme počítať skóre v n iteráciach. V každej z nich začneme od inej stratégie s predpokladom, že táto stratégia má skóre 0 a od tohoto skóre odvodíme skóre pre zvyšok stratégií. Takto dostaneme n rôznych skóre pre každú z n stratégií. Konečné skóre stratégie vypočítame ako priemer z jej n skóre.

	iterácia 1	iterácia 2	iterácia 3	výsledné skóre
stratégia A	0	-0.3	-0.2	-0.167
stratégia B	-0.5	0	-0.7	-0.4
stratégia C	-0.6	-0.1	0	-0.233

V tabuľke vidíme, že stratégia A získala najlepšie skóre.

4.5 Gradientný zostup

Naším cieľom je nájsť čo najlepšie ováhovanie parametrov hry. Za týmto cieľom použijeme myšlienku *gradientného zostupu*[8], aj keď v našom špecifickom prípade budeme potrebovať postupovať mierne inak.

Gradientný zostup porovnáva jednotlivé prvky v datasete a snaží sa nájsť takú parametrizáciu, aby sme dosiahli lokálne minimum diferencovateľnej fukncie.

V našom prípade sa však snažíme nájsť parametrizáciu datasetov, takže prvky použité v gradientnom zostupe budú výsledky porovnávania datasetov.

Budeme postupovať podobne ako pri striktnom výbere najlepšej stratégie z množiny. Majme stratégie S_1, S_2, S_3, S_4, S_5 s ich skóre R_1, R_2, R_3, R_4, R_5. Označme m najmenšie z dosiahnutých skóre. Každé skóre upravíme nasledovne: $R'_i = R_i - m$. Ováhovanie parametrov ktoré vyberieme nebude jedna z pätice stratégií, ale kombináciou váh stratégií a porovnania týchto stratégií. Tie stratégie ktoré v porovnaní vyšli najlepšie budú mať väčší vplyv na výsledné ováhovanie parametrov.

Označme váhy parametrov stratégie S_i ako $(s_{i_p1}, s_{i_p2}, s_{i_p3}, s_{i_p4}, s_{i_p5})$. Výslednú hodnotu parametra pi vypočítame ako $pi = \sum_{j=1}^{5} s_{j_p i} \times R'_j$

stratégie s parametrami	AvB	BvC	CvD	DvE	EvA	upravené skóre R'_i
A=(1.14,0.51,0.9,0.78,1.67)	1.873				2.075	0.183
B=(1.01,0.39,1.03,0.54,2.03)	2.127	2.035				0.047
C=(1.09,0.66,0.79,0.74,1.72)		1.965	1.891			0.1135
D=(1.01,0.36,1.01,0.81,1.81)			2.109	2.185		0
E=(1.48,0.64,0.84,0.6,1.43)				1.815	1.915	0.254

Na ilustračnom príklade v tabuľke si ukážeme ako vypočítať výslednú hodnotu prvého parametra.

$p_1 = 1.14 \times 0.183 + 1.01 \times 0.047 + 1.09 \times 0.1135 + 1.01 \times 0 + 1.48 \times 0.254 = 0.755725$

Podobným spôsobom vypočítame aj ostatné parametre: $p_2 = 0.34913$, $p_3 = 0.516135$, $p_4 = 0.40451$, $p_5 = 0.95946$.

Na konci tohto výpočtu nás ešte čaká normalizácia váh, aby ich súčet bol 5 ako v našej základnej stratégii. $p_1 = 1.27$, $p_2 = 0.58$, $p_3 = 0.86$, $p_4 = 0.68$, $p_5 = 1.61$.

Kapitola 5

Dosiahnuté výsledky

5.1 Porovnanie Take Card vs Pay Chip

V kapitole 3.3 sme ukázali, že na poradí prehľadávaní akcií *Áno prosím* a *Nie ďakujem* záleží. V prílohe k práci sme vytvorili stratégie PayChipStrategy, TakeCardStrategy a NoPruningStrategy. Prvá zo stratégií prezerá najprv možnosť zaplatenia žetónu, druhá zo stratégii prezerá ako prvé zobratie karty a v tretej nepoužívame orezávanie.

Pozreli sme sa na tri rôzne prípady. V prvom prípade je optimálne vždy hrať akciu *Nie ďakujem*. To sme dosiahli tak, že hráčom sme dali neobmedzený počet žetónov a že do hry budú prichádzať iba vedúce karty, teda nebude možné vytvoriť sekvenciu väčšej dĺžky ako 1. Tým pádom každá karta dá hráčovi taký počet záporných bodov aká je jej hodnota a keďže nepríde do hry žiadna susedná karta, najvýhodnejšie je túto kartu nebrať až pokým sa pri nej nevyzbiera toľko žetónov aká je na nej hodnota.

Poznámka: v prípade že sa nachádza $n-1$ žetónov na karte s hodnotou n, použitie oboch akcií *Áno prosím*, *Nie ďakujem* prinesie hráčovi jeden záporný bod.

Môžeme si všimnúť, že stratégia TakeCardStrategy ani raz neorezávala a bola najpomalšia. Čo je však zaujímavé, stratégia PayChipStrategy napriek tomu že spravila až 5322 orezaní, bola pomalšia ako NoPruningStrategy. To je zrejme spôsobené tým, že NoPruningStrategy pri prehľadávaní nepotrebuje overovať žiadne podmienky pre orezávanie. Navyše, môžeme si všimnúť aj na obrázku 3.4 že v prípade prehľadávania stromu sprava doľava, resp. prehľadávanie akcie *Nie ďakujem* ako prvej spôsobuje, že aj v prípade orezania ide iba o jeden vrchol, resp. malý podstrom.

	Nie ďakujem		Áno prosím		náhodná hra	
	čas	orezanie	čas	orezanie	čas	orezanie
PayChipStrategy	6.709s	5322.73	54.043s	73	6.693s	54336.76
TakeCardStrategy	8.115s	0	0.079s	496	2.921s	59370.8
NoPruningStrategy	6.24s	0	52.107s	0	6.573s	0

Pozrime sa na druhý prípad kedy je pre hráčov vždy výhodné hrať akciu *Áno prosím*.

To sme dosiahli tým, že z balíčku sa dostanú do hry také karty, ktoré vedia vytvoriť jednu sekvenciu. Ak upravíme hru tak, že budú v nej aj karty s hodnotami $-1, 0, 1, 2$, hráč získa bod vzatím sekvencie s vedúcou kartou -1.

V tabuľke vidíme, že stratégia *TakeCardStrategy* s výrazným rozdielom porazila zvyšné dve stratégie. Všimnime si, že aj keď je počet orezaní relatívne malý (496), ako bolo ukázané na obrázku 3.4, v prípade *TakeCardStrategy* ide o veľké podstromy čo výrazne urýchli výpočet. Stratégia *PayChipStrategy* aj v tomto druhom prípade o niečo zaostala za *NoPruningStrategy*.

Okrem extrémnych prípadov sme sa pozreli aj na náhodný priemerný prípad. Navrhli sme zmenšenú verziu hry s kartami v intervale $[3, 14]$, z ktorých príde do hry iba 8 kariet. Hráči majú v tejto hre 6 žetónov. Spustili sme 50 náhodných hier a porovnali sme výsledky.

Vidíme, že TakeCardStrategy vychádza aj v tomto prípade z tejto trojice ako jasný víťaz.

5.2 Porovnanie rôznych stratégií tretieho hráča

V podkapitole 4.4 sme uviedli, že stratégia tretieho hráča môže mať vplyv na porovnávanie stratégií zvyšných dvoch hráčov.

Teraz ukážeme merania kde sledujeme či pre stratégiu S má dobrý vplyv ak jeden z jeho protihráčov má takú istú stratégiu, alebo to má zlý vplyv, či žiaden vplyv to nemá.

Stratégia	A_1	B_2	B_3	B_1	A_2	A_3
počet 1.miest	10	18	25	19	10	23
počet 2.miest	20	15	14	17	20	17
počet 3.miest	20	17	11	14	20	10
priemerné skóre	-91	-78	-64	-76	-89	-71

V tabuľke vidíme porovnania dvoch variantov rozdelenia stratégií medzi hráčmi. Dolný index pri type stratégie určuje poradie hráča v hre. Všimnime si, že v prvom prípade keď sú v hre dve stratégie B, dosahujú lepší výsledok než v druhom prípade. Tam, naopak, dosahuje lepší výsledok stratégia A ako v prípade keď nemá spoluhráča s tou istou stratégiou.

Taktiež si môžeme všimnúť, že ak sa vyskytujú dvaja hráči s takou istou stratégiou v hre, lepšie výsledky z dvojice dosahuje ten, ktorý je na ťahu po hráčovi s tou istou stratégiou.

Dôvod týchto výsledkov je pre nás neznámy, no domnievame sa, že to môže byť spôsobené tým, že hráč A_1 resp. B_1 hrajú podľa ich spoluhráčov natoľko nevyspytateľne,

že dôsledky ich hrania pocíti hlavne ten hráč, ktorý ide priamo po nich, teda hráč B_2 resp. A_2.

Pripisujeme to tomu, že predpoklad hráča so stratégiou S o tom ako budú hrať jeho protihráči je silnejší v prípade že niektorý z jeho spoluhráčov hrá podobne ako on.

5.3 Vplyv nezaznamenaných častí stavu hry

Za cieľom zjednodušenia ukladania dát sme niektoré informácie o hre nezaznamenávali. Dobrým príkladom je zoznam kariet hráčov. Namiesto toho si pamätáme súčet hodnôt vedúcich kariet. Ďalšou zaujímavou informáciou je poradie kariet v akom prichádzajú do hry. Žetóny slúžia hráčom ako platidlo a potrebujú ich mať priebežne počas hry. Čo ak by všetky karty ktoré by hráč zobral pri niektorej parmutácii balíčka kariet prišli až úplne na konci hry? Bolo by pravdepodobné, že hráč by bol nútený zobrať si inú kartu z dôvodu minutia žetónov a výsledok hry by zrazu bol iný, pričom podľa našej parametrizácie tieto dva prípady nerozlišujeme.

V prílohe k práci sme v *TestDifferentPermutations* sledovali rozdielnosť výsledkov hry na základe rôznej permutácie kariet ktoré prídu do hry. Naše testovanie sme opäť ukázali na zmenšenej verzii hry. Tentokrát sme použili karty z intervalu $[3, 14]$ z ktorých sa do hry dostalo 8 kariet. Pre rôzne permutácie týchto 8 kariet sme následne odohrali 20 hier. Tento postup sme zopakovali pre 10 rôznych osmíc kariet ktoré boli vybrané do hry.

Pre každú osmicu parmutácií kariet P_i označme výsledky jednotlivých hier ako $r_1 = (p1_1, p2_1, p3_1), ..., r_{20} = p1_{20}, p2_{20}, p3_{20})$. Označme priemerný rozdiel skóre hráča 1 a 2 takto: $d_1 = \frac{1}{20}\sum_{i=1}^{20} p1_i - p2_i$. Podobne vypočítame rozdiel hráča 1 a 3.

Vplyv rôznych permutácií hry zaznamenáme v smerodajnej odchýlke (standard deviation[9]) cez všetky rozdiely v skóre. Čím väčšia je odchýlka, tým nestabilnejšie sú výsledky vzhľadom na permutáciu kariet.

$$SD = \sqrt{\frac{1}{n}(\sum_{i=1}^{n}(p1_i - p2_i)^2 + \sum_{i=1}^{n}(p1_i - p3_i)^2)}$$

Zaujíma nás priemerný rozdiel skóre hráča 1 a hráča 2 a podobne aj priemerný rozdiel hráča 1 a hráča 3. Následne odmeriame smerodajnú odchýlku(standard deviation). Ak je priemerné skóre hráčov $(-5, -4, -2)$, tak výsledok hry $(-8, -7, -5)$ má nulovú odchýlku, keďže rozdiel hráčov je rovnaký.

karty v hre	priemerný výsledok	min D	max D
12, 9, 4, 10, 13, 11, 3, 14	1.55 1.75 1.55	2 2 2(0.2)	-7 -3 -7(3.8)
9, 3, 11, 4, 10, 12, 13, 7	-3.8 -4.05 -1.9	-1 -1 1(0.269)	-15 -15 0(13.102)
10, 6, 8, 14, 4, 3, 9, 13	-6.2 -7.45 -4.95	-4 -4 -4(1.768)	-10 -10 -6(3.021)
12, 3, 14, 7, 8, 5, 10, 4	-12.0 -11.95 -9.5	-12 -12 -9(0.502)	-11 -11 -6(2.5)
8, 3, 4, 6, 5, 10, 11, 12	-5.7 -4.25 -4.3	-7 -6 -6(0.602)	-6 -2 -6(2.909)
7, 3, 5, 12, 8, 11, 14, 6	-10.6 -9.65 -8.75	-13 -12 -11(0.158)	-6 -6 -3(1.492)
5, 8, 7, 13, 10, 12, 6, 9	-3.75 -3.65 -3.05	0 0 1(0.316)	-3 -12 -12(13.3)
6, 5, 10, 13, 12, 3, 11, 9	-1.85 -1.3 -0.45	0 0 1(0.68)	-6 -8 -6(2.909)
7, 9, 8, 11, 6, 13, 5, 4	-7.95 -7.15 -5.75	-6 -5 -4(0.283)	-8 -5 -4(2.843)
3, 13, 6, 5, 8, 11, 14, 10	-8.9 -8.85 -6.65	-8 -8 -5(0.752)	-8 -5 -8(3.71)

V tabuľke vidíme merania rôznych osmíc kariet, ktorých permutácie predstavovali balíček v hrách. Na nenulových hodnotách smerodajnej odchýlky SD vidíme, že na poradí kariet záleží. V stĺpcoch *min D* a *max D* vidíme najmenší, resp. najväčší rozdiel výsledku hry od priemerného výsledku.

Smerodajná odchýlka z týchto meraní vyšla $SD = 2.67969$.

Všimnime si, že v poslednom riadku je víťaz hry s maximálnou odchýlkou druhý hráč, pričom v priemernom prípade vyhráva tretí hráč. Podobnú situáciu môžeme vidieť aj v siedmom riadku. Preto môžeme hovoriť o tom, že poradie kariet má vplyv na výsledok hry. Vo väčšine prípadov však ten vplyv nie je veľký, aspoň nie natoľko aby znamenal zmenu poradia hráčov.

5.4 Hľadanie najlepšieho ováhovania parametrov

V podkapitole 4.4.3 sme ukázali postup ktorým zistíme ktorá z pätice stratégií dosahuje najlepšie výsledky. To nám povedalo v blízkosti akých ováhovaní parametrov budeme generovať nové stratégie.

V nasledovnej tabuľke vidíme v riadkoch päticu stratégií ktoré boli porovnávané. Skóre R_i podľa podkapitoly 4.4.3 je v riadku pod päticou stratégií. Stratégia s najväčším skóre je považovaná za najlepšiu a nachádza sa v ďalšom porovnaní.

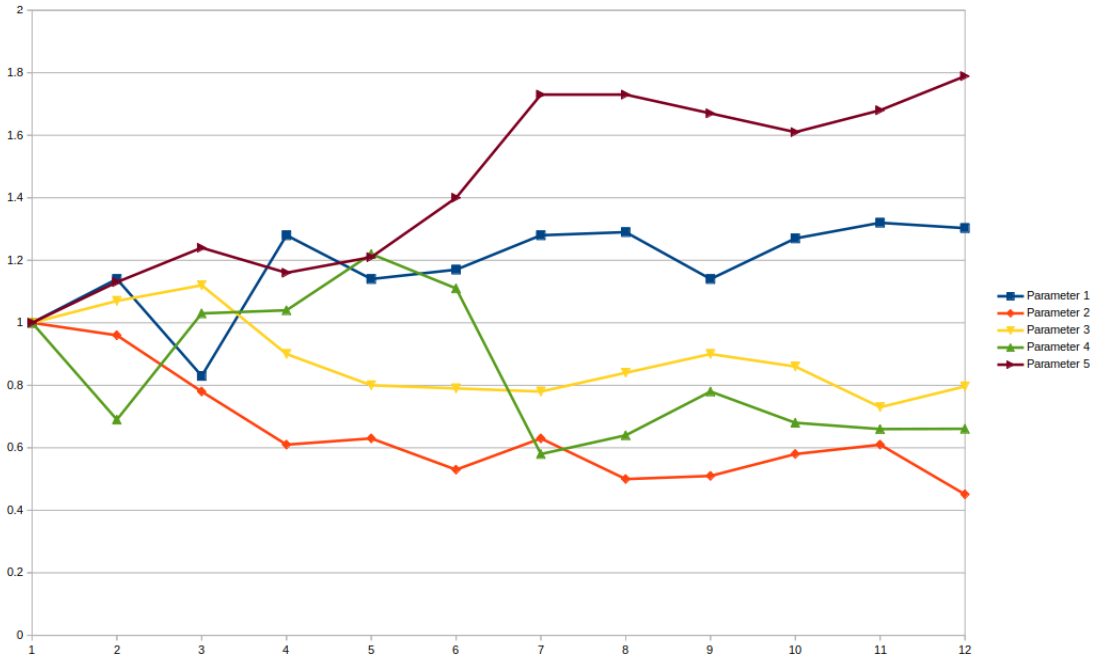
	S1	S2	S3	S4	S5
1	1.0,1.0,1.0,1.0,1.0	0.79,1.13,1.2,0.98,0.89	1.04,1.14,0.64,1.08,1.11	1.12,1.24,0.68,0.86,1.11	1.14,0.96,1.07,0.69,1.13
	0.1821	0.0138	0.0746	0	**0.2962**
2	1.14,0.96,1.07,0.69,1.13	1.22,1.23,1.06,0.68,0.81	0.89,1.08,0.98,0.91,1.14	1.26,0.88,1.02,0.65,1.2	1.12,1.02,0.72,0.78,1.36
	0.2567	0.1375	0.1533	0.2204	0
3	1.14,0.96,1.07,0.69,1.13	1.17,0.8,1.21,0.75,1.07	0.94,1.15,1.0,0.58,1.33	0.83,0.78,1.12,1.03,1.24	0.98,0.85,1.09,0.88,1.21
	0.1413	0.0848	0	**0.1452**	0.0812
4	0.83,0.78,1.12,1.03,1.24	1.04,0.79,1.17,0.76,1.24	1.2,0.78,0.87,1.19,0.95	1.2,0.72,1.05,0.98,1.05	1.28,0.61,0.9,1.04,1.16
	0.1143	0	0.2012	0.0857	**0.2285**
5	1.28,0.61,0.9,1.04,1.16	1.14,0.63,0.8,1.22,1.21	0.95,0.71,0.98,1.23,1.13	0.99,0.82,0.69,1.2,1.3	1.41,0.61,0.81,0.94,1.23
	0.1996	**0.2321**	0	0.0158	0.2088
6	1.14,0.63,0.8,1.22,1.21	0.93,0.63,0.75,1.58,1.11	0.93,0.83,0.83,1.17,1.25	1.29,0.49,0.85,1.24,1.13	0.84,0.89,1.04,1.38,0.86
	0.2247	0	0.0848	0.2172	0.1037
7	1.14,0.63,0.8,1.22,1.21	1.27,0.82,0.85,1.1,0.95	1.17,0.53,0.79,1.11,1.4	0.76,0.95,0.89,1.02,1.37	0.96,0.79,0.7,0.94,1.62
	0.229	0.0538	**0.3049**	0.0239	0
8	1.17,0.53,0.79,1.11,1.4	0.91,0.79,0.48,1.33,1.49	0.95,0.97,1.07,1.09,0.93	1.01,0.24,1.28,0.99,1.48	1.04,0.72,0.78,1.26,1.2
	0.405	0.2271	0.3183	0	0.2496
9	1.17,0.53,0.79,1.11,1.4	1.23,0.34,0.72,1.14,1.57	1.27,0.98,0.69,0.7,1.36	1.37,0.59,1.14,0.91,0.98	1.42,0.87,0.87,0.84,1.0
	0.2674	0.1864	0	0.1086	0.1859
10	1.17,0.53,0.79,1.11,1.4	1.0,0.84,0.87,1.11,1.17	1.34,0.3,0.57,1.33,1.46	1.28,0.91,0.88,1.0,0.93	1.27,0.39,0.65,1.11,1.57
	0.1618	0	0.0028	0.0255	0.0399
11	1.17,0.53,0.79,1.11,1.4	0.72,0.71,0.71,1.42,1.43	1.28,0.63,0.78,0.58,1.73	0.89,0.37,0.9,1.05,1.79	1.12,0.57,0.64,1.15,1.53
	0.1132	0	**0.1568**	0.0028	0.0313
12	1.28,0.63,0.78,0.58,1.73	1.44,0.78,0.6,0.98,1.2	1.08,0.61,0.83,0.31,2.16	0.93,0.78,0.53,0.71,2.05	1.45,1.0,0.63,0.47,1.46
	0.4062	0.0759	0.3157	0	0.1873
13	1.28,0.63,0.78,0.58,1.73	1.19,0.74,0.79,0.48,1.8	1.21,0.55,0.87,0.56,1.82	1.29,0.5,0.84,0.64,1.73	1.31,0.53,0.93,0.67,1.56
	0.0962	0.026	0	**0.1132**	0.0838
14	1.29,0.5,0.84,0.64,1.73	1.26,0.61,0.88,0.73,1.52	1.28,0.44,0.86,0.6,1.82	1.23,0.54,0.74,0.8,1.68	1.14,0.51,0.9,0.78,1.67
	0	0.1638	0.1181	0.1481	**0.2213**
15	1.14,0.51,0.9,0.78,1.67	1.07,0.45,1.07,0.93,1.48	1.19,0.57,1.05,0.7,1.49	0.98,0.43,1.0,0.69,1.9	1.11,0.59,0.79,0.89,1.62
	0.2448	0	0.2228	0.118	0.2283
16	1.14,0.51,0.9,0.78,1.67	1.01,0.39,1.03,0.54,2.03	1.09,0.66,0.79,0.74,1.72	1.01,0.36,1.01,0.81,1.81	1.48,0.64,0.84,0.6,1.43
	0.183	0.047	0.1135	0	**0.254**

Môžeme si všimnúť, že v porovnávaní 3 bola so skóre 0.1452 vybratá stratégia $0.83, 0.78, 1.12, 1.03, 1.24$. Pozrime sa ale na stratégiu $1.14, 0.96, 1.07, 0.69, 1.13$ ktorá dosiahla iba o niečo menšie skóre 0.1413. Môžeme vidieť, že ováhovanie týchto stratégií je v prvom a štvrtom parametri dosť odlišné.

Preto sa núka myšlienka, že aj táto stratégia mohla svojimi výsledkami prispieť do ováhovania do ktorého sa posunieme v ďalšom kroku. Rozhodli sme sa preto, že v tomto bode zmeníme spôsob vyberania najlepšieho ováhovania z pätice stratégií a skonštruovali sme algoritmus inšpirovaný gradientným zostupom popísaný v kapitole 4.5.

	S1	S2	S3	S4	S5
16	1.14,0.51,0.9,0.78,1.67	1.01,0.39,1.03,0.54,2.03	1.09,0.66,0.79,0.74,1.72	1.01,0.36,1.01,0.81,1.81	1.48,0.64,0.84,0.6,1.43
	0.183	0.047	0.1135	0	0.254
17	1.27,0.58,0.86,0.68,1.61	1.36,0.65,0.6,0.63,1.75	1.46,0.42,0.7,0.72,1.7	1,43,0.46,0.75,0.53,1.83	1.21,0.54,0.76,0.93,1.56
	0.037625	0.037125	0.0035	0.012375	0
18	1.32,0.61,0.73,0.66,1.68	1.41,0.46,0.7,0.69,1.73	1.04,0.53,0.92,0.58,1.93	1.42,0.65,0.78,0.35,1.8	1.41,0.32,0.78,0.7,1.79
	0.18375	0	0.14	0.0025	0.31125

Na grafe vidíme grafické zobrazenie ako sa menili váhy parametrov. V tabuľke sú zaznamenané všetky ováhovania v stĺpci S_1(stĺpec obsahujúci dovtedy najlepšie ováhovanie). Vidíme že prvý a piaty parameter sú podľa týchto meraní významnejšie ako zvyšné parametre.

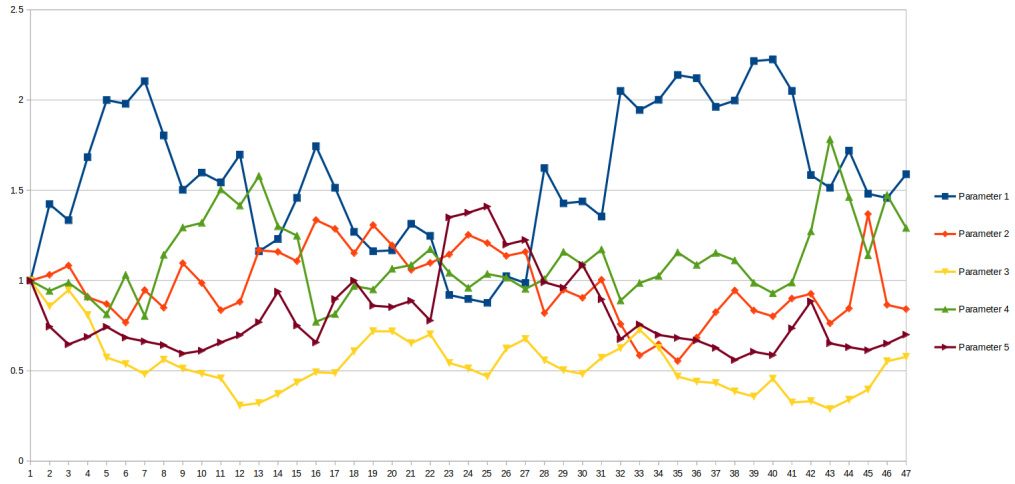


Obr. 5.1: Postupné menenie váh parametrov

Porovnávanie malých datasetov

Podľa odmeraných hodnôt z kapitoly 4 považujeme za štandard vytvárať datasety aspoň z 1000 hier a tieto datasety porovnávať aspoň na 400 hrách. Tento postup je stabilnejší a presnejší, avšak veľmi pomalý.

Preto sme sa skúsili pozrieť na opačný extrém, kedy generujeme datasety iba z 10 hier a porovnávame ich iba na 10 hrách. Okrem toho sme nastavili $\epsilon = 2$ oproti štandardu $\epsilon = 1.3$. Výsledky uvádzame iba v grafickej podobe na obrázku 5.2

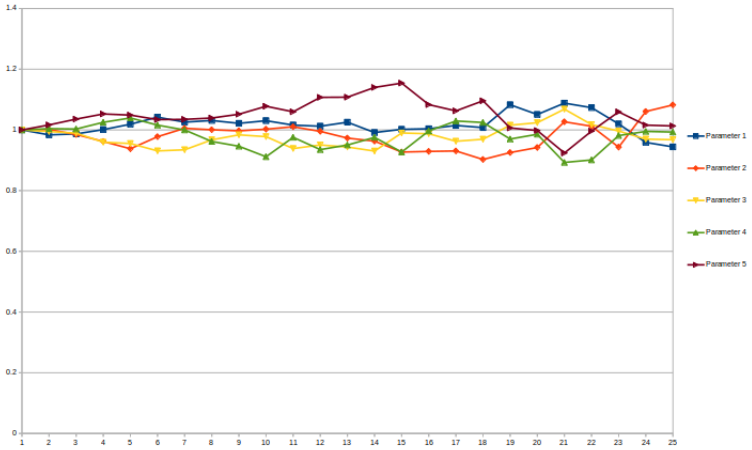


Obr. 5.2: Postupné menenie váh parametrov podľa malých datasetov

V prvom rade si môžeme všimnúť, že hodnoty sú menej stabilné, rýchlo a nečakane menia svoje hodnoty. Vidíme, že aj v tomto grafe je prvý parameter významný, avšak

piaty parameter patrí medzi menej významné, napriek chvíľkovému vyššiemu významu okolo 25.generácie.

Čas vyhodnocovania bol naozaj rýchlejší, preto sme sa pokúsili znížiť nestabilitu a upravili sme výchylku pre generovanie nových ováhovaní na $\epsilon = 1.15$. Počet porovnávaných hier a veľkosť datasetov zostali nezmenené na 10 hrách.



Obr. 5.3: Postupné menenie váh parametrov podľa malých datasetov s menším ϵ

Môžeme si na obrázku 5.3 všimnúť, že podobne ako v predchádzajúcom prípade, váhy parametrov sa neustálili, dokonca si môžeme všimnúť, že od 15.kroku po 20.krok sa hodnoty parametrov otočili. Preto hodnotu tohto ako aj predchádzajúceho merania založeného na malých datasetoch a malom počte porovnávajúcich hier môžeme považovať za nízku.

Porovnanie najlepšieho ováhovania s pôvodným

Doteraz sme porovnávali stratégie iba vrámci jedného generačného kroku, čiže také, ktoré mali približne rovnaké ováhovanie. Z nich sme si vybrali tú najlepšiu stratégiu a pokračovali sme do ďalšieho generačného kroku.

Prepdokladáme teda, že porovnanie pôvodnej stratégie S_i s ováhovaním $(1.0, 1.0, 1.0, 1.0, 1.0)$ so stratégiou S_f ku ktorej sme prišli v podkapitole 5.4 s ováhovaním $(1.3, 0.45, 0.8, 0.66, 1.79)$ z obrázku 5.1 by malo viesť k výhre stratégie S_f.

Tieto dve stratégie sme porovnali na 200 hrách a s nasledovnými výsledkami:

Stratégia	Priemerné skóre	Priemerné poradie
S_i	-79.31	2.022
S_f	-78.58	1.978

Vidíme, že čísla hovoria v prospech stratégie S_f, no rozdiel nie je vôbec veľký. Túto vyrovnanosť môžeme pripísať buď tomu, že porovnávanie stratégií nie je natoľko tranzitívne a teda môže existovať stratégia A ktorej sa darí proti stratégii B, no vo

všeobecnosti stratégia A môže dosahovať horšie výsledky ako stratégia B. V tomto prípade to môže byť prípad našej pôvodnej stratégie s rovnomerným ováhovaním parametrov, no dôvod môže byť aj iný: naše porovnávanie môže byť nepresné kvôli veľkosti stratégií z ktorých sú generované. Koniec koncov nám to aj ukazujú grafy 5.2 a 5.3 kde sa ováhovanie datasetov založených na málo hrách mení až príliš.

Preto predpokladáme, že tento istý postup by bol úspešnejší v prípade väčšieho množstva času alebo výpočtových kapacít, vďaka ktorým by sme mohli porovnávať stratégie založené na rádovo väčšom počte hier.

Implementáciu nebudeme v texte práce bližšie popisovať. Je k nej priložená dokumentácia a samotný kód je prehľadne okomentovaný. Jej aktuálnu verziu možno nájsť na https://github.com/filip12h/MasterNoMerci

Záver

Spoločenské hry zažívajú posledné dekády svoje zlaté časy a postupom času sa mnohé z nich dostávajú do počítačovej formy. Napriek tomu, že hráči sú dnes schopní hrať proti iným ľudským spoluhráčom online, otázka automatizovaných oponentov predstavovaných počítačom je stále aktuálna.

V tejto diplomovej práci sme sa presvedčili, že oblasť hier s 3 a viac hráčmi je stále nerozvinutá a použité prístupy sú pre nás nedostačujúce.

Taktiež sme ukázali, že stále je v tejto oblasti priestor v teoretických posunoch ako napríklad prehľadávanie nevyvážených stromov či orezávanie s prioritou poradia hráčov, resp. bez horného ohraničenia súčtu bodov.

V tejto práci sme sa zamerali na verziu hry so známou budúcnosťou. Pri pokračovaní v tejto práci by mohla slúžiť ako základ pre pôvodnú verziu hry. Pri rozhodovaní ktorú z dvojice akcií hráč vykoná by sme si vygenerovali niekoľko náhodných permutácií zvyšku balíčka, kde na každú z permutácií by sme dostali odpoveď (*Áno prosím* alebo *Nie ďakujem*). Tú z odpovedí, ktorá by sa vyskytla ako častejšia odpoveď, by hráč použil.

Pri ďalšej práci na optimalizácii hrania hry No Thanks by sme mohli uvažovať aj rôzne ováhovania v rôznych časoch hry. Vidieť to môžeme na žetónoch. Žetón dáva na konci hry hráčom jeden bod, avšak v priebehu hry má ešte inú, dôležitú funkciu - je platidlom, vďaka ktorému sa hráč môže vyhnúť veľkému počtu záporných bodov. Preto môžeme predpokladať, že v priebehu hry majú žetóny pre hráčov väčšiu váhu ako na konci hry.

Rôzne ováhovanie hry by bolo možné zisťovať priamo pri generovaní stratégie - podobným spôsobom ako sme porovnávali stratégie na náhodných hrách, by sme porovnávali jednotlivé ováhovania úsekov hry na čiastočných hrách - vygenerovaných konfiguráciách hry prislúchajúcich k danému úseku ktoré by sme odsimulovali od daného úseku do konca.

Literatúra

[1] Korf R.E. 1989. Multi-player alpha-beta pruning. Artificial Intelligence 48, pages 99-111, Elsevier

[2] v. Neumann J., Zur Theorie der Gesellschaftsspiele. Math. Ann. 100, 295–320 (1928)

[3] Nash J.F. 1950. Equilibrium Points in N-Person Games. Proceedings of the National Academy of Sciences 36 (1): 48–49

[4] Luckhardt C.A., Keki B.I. 1986, An algorithmic solution of N-person games, (AAAI'86), p.158-162, AAAI Press

[5] Marsland T.A.. 1987. Computer Chess Methods. Computing Science Department, University of Alberta, EDMONTON, Canada T6G 2H1

[6] Bentley J.L., 1975, Multidimensional binary search trees used for associative searching

[7] Bentley J.L., Friedman J.H., Finkel R.A. 1977. An algorithm for finding best matches in logarithmic expected time

[8] Curry H.B., The Method of Steepest Descent for Non/linear Minimization Problems (1944)

[9] Bland J.M., Altman D.G., Measurement error (1996)