VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

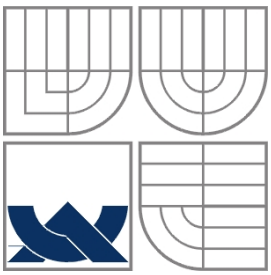
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

UMĚLÁ INTELIGENCE PRO HRANÍ HER

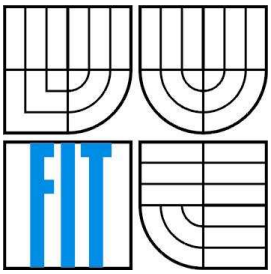
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE Bc. TOMÁŠ KUČÍREK
AUTHOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

UMĚLÁ INTELIGENCE PRO HRANÍ HER

ARTIFICIAL INTELLIGENCE FOR GAME PLAYING

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. TOMÁŠ KUČÍREK

VEDOUCÍ PRÁCE
SUPERVISOR

doc. RNDr. PAVEL SMRŽ, Ph.D.

BRNO 2012

Abstrakt

Arimaa je strategická desková hra pro dva hráče. Byla navržena tak, aby byla jednoduchá pro živé hráče, ale složitá pro počítače. Cílem této diplomové práce je návrh a tvorba programu s prvky umělé inteligence, který by v Arimě dokázal porazit živého hráče. Návrh aplikace probíhal zejména ve třech klíčových částech: hodnocení rozestavení, generování tahů a prohledávání. Program byl spuštěn na herním serveru, kde dokázal porazit řadu botů i živých hráčů.

Abstract

Arimaa is a strategic board game for two players. It was designed to be simple for human players and difficult for computers. The aim of this thesis is to design and implement the program with features of the artificial intelligence, which would be able to defeat human players. The implementation was realized in the three key parts: evaluation position, generation of moves and search. The program was run on the game server and defeated many bots as well as human players.

Klíčová slova

Arimaa, umělá inteligence, hodnotící funkce, generování tahů, ořezávání tahů, deskové hry na počítači, bitboard

Keywords

Arimaa, artificial intelligence, evaluation function, generation moves, pruning move, board games on the computer, bitboard

Citace

Tomáš Kučírek: Umělá inteligence pro hraní her, diplomová práce, Brno, FIT VUT v Brně, 2012

Umělá inteligence pro hraní her

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením doc. RNDr. Pavla Smrže, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Bc. Tomáš Kučírek
22. května 2012

Poděkování

Chtěl bych poděkovat doc. RNDr. Pavlu Smržovi, Ph.D. za vstřícnost při vedení práce.

Obsah

1 Úvod

Tato práce se věnuje deskové hře Arimaa a vývoji programu s prvky umělé inteligence, který v této hře dokáže porazit živé hráče. Arimaa je tahová hra pro dva hráče, vytvořená Omarem Syedem tak, aby nebylo možné vytvořit program, který by byl schopný na běžně používaných počítačích porazit živého hráče pouze hrubou silou.

Druhá kapitola popisuje pravidla pro hraní Arimy, včetně notace zápisu pro uložení průběhu hry nebo aktuální pozice. Správné pochopení všech pravidel je důležité k tvorbě kvalitních hodnotících funkcí.

Ve třetí kapitole jsou vysvětleny důvody pro vznik Arimy a také složitost hry. Dále je zde popsáno dění kolem Arimy, mistrovství, která se v této hře každoročně pořádají a Arimaa Challenge.

Ve čtvrté kapitole jsou popsány samotné hodnotící funkce: základní, které popisují aktuální stav figurek na hrací desce, a pokročilé, které hodnotí budoucí vývoj a nahrazují prohledávání několik tahů dopředu.

Pátá kapitola se zabývá generováním tahů. V rámci ořezávání jsou zde navrženy faktory, které by měl obsahovat každý tah, který bude hodnocen. Tyto faktory jsou testovány, aby se dokázala jejich účinnost.

Šestá kapitola popisuje nezbytnou součást programu, hledání budoucího tahu. Pro tento účel byl zvolen algoritmus Minimax s Alfa-beta ořezáváním, který dominuje u šachových programů. Dále jsou zde popsány metody pro podporu ořezávání.

Sedmá kapitola se zabývá implementací. Je zde uvedena kostra programu a popsány její jednotlivé části. Druhá část kapitoly obsahuje popis datové struktury bitboard, která byla použita pro reprezentaci herní desky.

V osmé kapitole je popsáno učení váhového vektoru z historie her herního serveru pomocí genetických algoritmů a testování těchto vektorů na historii.

V deváté kapitole je představena herní strategie Útok osamoceným slonem a její implementace pomocí změn vah jednotlivých hodnocení. Dále je zde uvedeno několik startovních rozestavení a situace, při kterých přechází program do střední hry.

Poslední kapitola se zabývá testováním programu na herním serveru. Je zde uveden rating, kterého program dosáhl, a další statistiky.

2 Arimaa

Arimaa je desková hra s nulovým součtem, vytvořená Omarem Syedem, odborníkem na umělou inteligenci, v roce 2002. Ve hře proti sobě stojí dva hráči označeni zlatou a stříbrnou barvou, kteří se střídají v tazích. První na tahu je zlatý hráč. Arimu lze hrát se stejnými figurkami a hrací deskou jako šachy. Aby hra byla vhodná i pro hráče, kteří šachy nikdy nehráli, jsou šachové figurky nahrazeny zvířaty. Arimaa je navržena tak, aby měla jednoduchá pravidla, ale byla složitá pro vývoj umělé inteligence.

Tato kapitola pojednává o pravidlech hry. Jsou zde popsány cíle hry a možnosti, jak jich dosáhnout. V závěru kapitoly je popsána notace. Následující informace jsou čerpány z oficiálních WWW stránek [1].

2.1 Figurky a hrací deska

Arimaa tedy obsahuje 32 figurek o 6 typech, kde každý typ je znázorněn jedním zvířetem. Jedná se o slona, velblouda, koně, psa, kočku a králíka. Každý hráč má na začátku hry 8 králíků, 2 kočky, 2 psy, 2 koně a po jednom velbloudovi a slonovi. Jak je vidět na Obrázku 2.1, každé zvíře má jinou sílu, kterou lze určit zcela intuitivně.

Hrací deska má velikost 8x8 polí a na rozdíl od šachů je jednobarevná. Jednotlivá pole hrací desky jsou značena souřadnicemi podle řádku a sloupce. Sloupce jsou značeny písmeny a-f. Řádky jsou značeny číslicemi 1-8. Hrací deska obsahuje 4 speciální pole, takzvané pasti, na pozicích c3, f3, c6, f6. Význam těchto polí bude vysvětlen v kapitole 2.5.

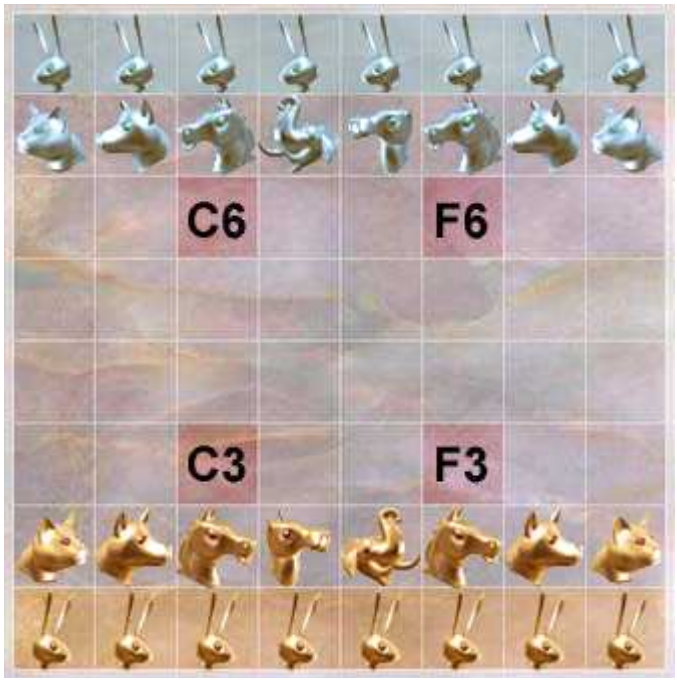
Obrázek 2.1 Síla figurek, citováno z [2]

2.2 Počáteční nastavení

Hra začíná prázdnou hrací deskou. Při prvním tahu si oba hráči postaví své figurky na hrací desku podle svého uvážení. Začíná zlatý hráč, který má pro rozestavení svých figurek k dispozici 1. a 2. řadu. Poté je na tahu stříbrný hráč, který využije 7. a 8. řady. Na Obrázku 2.2 je vidět často používané rozestavení.

Existuje několik doporučení, jak rozestavit své figurky:

- Stavět silné figurky před slabé.
- Nestavět stříbrného slona do stejného sloupce, kde se nachází zlatý slon.
- Stavět silné figurky tak, aby měly rychlý přístup do centra hrací desky.



Obrázek 2.2 Počáteční nastavení

2.3 Cíl

Cílem hry je dostat svého králíka na nejvzdálenější řadu hrací desky, nebo odstranit všechny soupeřovy králíky. Konec hry se určuje podle níže uvedených pravidel po odehrání každého tahu v závislosti na tom, který hráč tento tah právě provedl.

Pokud zlatý hráč právě provedl svůj tah:

1. Pokud se nachází zlatý králík na 8. řadě hrací desky, vyhrál zlatý hráč.
2. Pokud se nachází stříbrný králík na 1. řadě hrací desky, vyhrál stříbrný hráč.
3. Pokud stříbrný hráč nemá na hrací desce žádného svého králíka, vyhrál zlatý hráč.
4. Pokud zlatý hráč nemá na hrací desce žádného svého králíka, vyhrál stříbrný hráč.

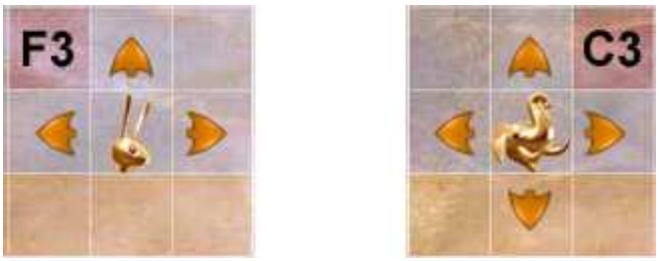
5. Pokud stříbrný hráč nemá žádný možný tah (jeho figurky jsou buď zmraženy, nebo nemají žádné volné pole pro pohyb), vyhrává zlatý hráč.
6. Pokud zlatý hráč nemá žádný možný tah (jeho figurky jsou buď zmraženy, nebo nemají žádné volné pole pro pohyb), vyhrává stříbrný hráč.

Pokud právě provedl tah stříbrný hráč, pravidlům se pouze přehodí pořadí kontroly.

Jestliže se po skončení tahu bude na hrací desce nacházet takové rozestavení figurek, které se již předtím dvakrát na konci tahu objevilo, bude tah prohlášen za neplatný. Hráč musí vybrat jiný tah. Pokud není žádný jiný tah možný, hráč prohrává. Ve hře Arimaa, na rozdíl od šachů, nemůže nastat remíza, pokud chce jeden z hráčů ukončit hru předčasně, musí se vzdát.

2.4 Pohyb

Všechny figurky se pohybují stejně, po horizontální nebo vertikální ose. Po diagonální ose se pohybovat nemohou. Jedná se tedy o pohyb dopředu, dozadu, vlevo a vpravo. Výjimkou je králík, který se nemůže pohybovat dozadu (zpět ke své startovní řadě), viz Obrázek 2.3.



Obrázek 2.3 Pohyb figurek

2.4.1 Tah

V každém tahu musí hráč provést 1 až 4 kroky. Za jeden krok je považován posun jedné figurky o jedno pole v určitém směru. Figurka se může posunout pouze na prázdné pole. Tah nesmí skončit ve stejném rozestavení figurek, ve kterém začal. Tyto kroky mohou být využity jednou figurkou, která může po každém kroku změnit svůj směr. Kroky také mohou být rozděleny mezi více figurek, tudíž se mohou za jeden tah pohnout až 4 figurky.

2.4.2 Tlačení a táhnutí

Silnější figurky mohou pohybovat slabšími soupeřovými figurkami. To znamená, že zlatý pes může pohybovat stříbrnou kočkou a králíkem, ale nemůže pohybovat stříbrným psem, koněm, velbloudem ani slonem. Soupeřova figurka musí sousedit (v ortogonálně přilehlém poli) s figurkou, kterou s ní chceme pohnout. Lze použít dva typy pohybu: tlačení a táhnutí. Oba dva tyto pohyby vyžadují dva

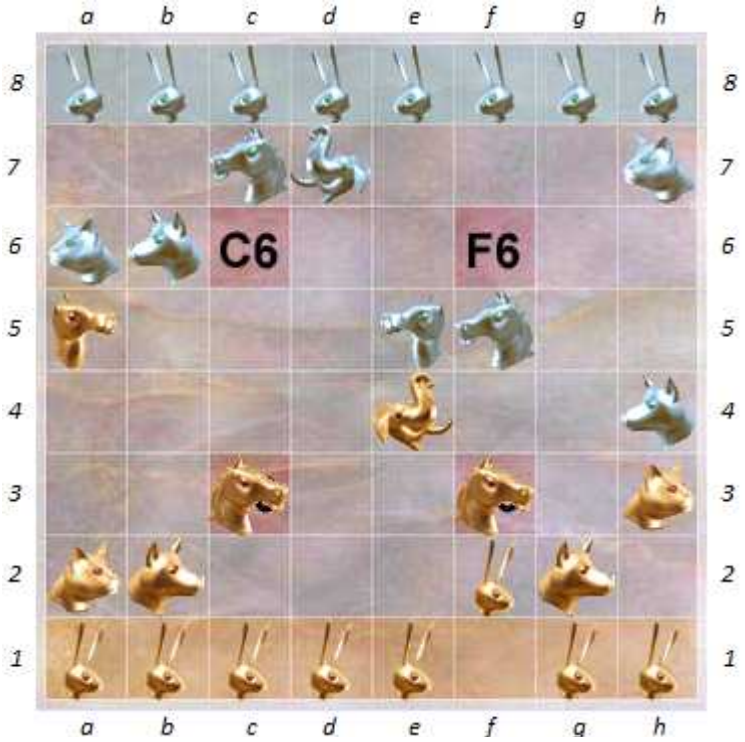
kroky, které musí být oba provedeny v jednom tahu. Během jednoho pohybu lze tedy například použít jen dvoje tlačení.

Při pohybu tažení se v prvním kroku posune silnější figurka na sousedící prázdné pole. V druhém kroku se slabší soupeřova figurka posune na pole, kde dříve stála silnější figurka. Při pohybu tlačení se v prvním kroku posune slabší soupeřova figurka na sousedící prázdné pole. V druhém kroku se posune silnější figurka na pole, kde dříve stála slabší soupeřova figurka. Použitím těchto pohybů lze docílit toho, že se králíci mohou pohybovat zpět. Tyto pohyby nelze kombinovat, oba kroky musejí být provedeny neprodleně za sebou. Také nelze jednou figurkou zároveň tlačit a táhnout.

2.4.3 Zmrazení

Silnější figurkou můžeme zmrazit slabší sousedící soupeřovu figurku. Zmrazená figurka se nemůže sama pohybovat. Kdykoliv zmrazená figurka sousedí s figurkou stejné barvy, stává se tato figurka opět mobilní a může se volně pohybovat. Zmrazenou figurkou lze pohnout pomocí soupeřovy silnější figurky pomocí tlačení nebo táhnutí.

2.4.4 Příklad



Obrázek 2.4 Ukázka pohybů

Rozestavení figurek na Obrázku 2.4 ukazuje některé výše popsané situace:

- Zlatý velbloud na a5 může tlačit stříbrnou kočku na a6. Tlačením ji posune na pole a7 a sám se posune na pole a6.
- Zlatý slon na e4 může táhnout stříbrného velblouda na e5. Tažením se posune na f4 a stříbrný velbloud se posune na původní místo slona, tedy na e4.
- Zlatá kočka na h3 je zmrazena stříbrným psem na h4 a nemůže se tedy sama pohnout.

2.5 Pasti

Pasti jsou jediná místa na hrací desce, kde je možné odebrat figurky ze hry. Pokud se na pasti nalézá figurka, která v sousedství nemá žádnou figurku stejné barvy, je z hrací desky odebrána. Odebrání se provádí okamžitě, nečeká se na konec tahu. Jak je vidět na Obrázku 2.3, zlatý kůň na pozici c3 bude odebrán. Zlatý kůň na pozici f3 odebrán nebude, protože má v sousedství zlatého králíka na pozici f2.

2.6 Notace

Aby bylo možné uložit průběh hry a později si ji přehrát nebo aby bylo možné hrát hru přes internet, byla zavedena přesná notace, jak ukládat jednotlivé tahy a jednotlivé pozice.

Každá figurka je označena prvním písmenem (iniciálou) jejího anglického názvu. Velkými písmeny jsou značeny zlaté figurky a malými písmeny jsou značeny stříbrné figurky. Pouze velbloud je značen písmeny m/M.

Zlaté figurky se značí takto:

- E -slon
- M - velbloud
- H - kůň
- D - pes
- C - kočka
- R – králík

Stříbrné figurky se značí takto:

- e -slon
- m - velbloud
- h - kůň
- d - pes
- c - kočka
- r - králík

Prvními písmeny anglického názvu jsou také označeny směry, kam se má figurka posunout:

- n – sever
- s – jih
- w – západ
- e – východ

- x – odebrání

Záznam hry začíná hlavičkou, která může obsahovat důležité informace o hře: typ hry, místo, kde se hra odehrála, datum, počet kol, jméno zlatého (bílého) hráče, jméno stříbrného (černého) hráče a výsledek. Zde je příklad takovéto hlavičky:

```
Event: Casual Game
Site: Cleveland, OH  USA
Date: 1999.01.15
Round: ?
White: Aamir Syed
Black: Omar Syed
Result: 1-0
```

Následuje seznam tahů, který je uvozen pořadím kola a iniciálami hráče, který tah provedl. Jako iniciály se mohou používat jak dvojice g/s, tak šachové w/b. Během výpisu tahů se mohou objevit 2 klíčová slova. První slovo je „takeback“ a znamená vrácení tahu. Druhé slovo je „resigns“ a znamená, že hráč odstoupil ze hry a prohrál.

Během prvního kola si hráči rozestavují své figurky na hrací desku. Tato akce se značí výpisem každé figurky, jejími iniciálami a pozicí, kam byla umístěna.

Následuje seznam provedených tahů. Každý tah se skládá z několika kroků. Krok je zapsán ve formátu FSRP, kde F znamená iniciálu figurky, SR označuje sloupec a řadu aktuální pozice figurky a P značí směr, kam se figurka posune. V jednom tahu mohou být zapsány i více než 4 kroky, jelikož se jako krok zapisuje i odebrání figurky z hrací desky.

Příklad záznamu hry:

```
1g Ra2 Rb2 Mc2 Dd2 ...
1s ra7 rb7 rc7 rd7 ...
2g Ra2n Ra3e Rb3n Rb4e
2s ra7s ra6s ra5e rb5e
3g Dd2n Dd3n Mc2e Rc4s Rc3x
3s rc7s rc5e rc6x rd5e re5s
4g takeback
3s takeback
3g Rb2n Rb3n Rb4n
3s ...
...
16s resigns
```

Pokud se záznam delší jak jeden řádek, je na začátku i na konci označen textem „-=+=-“. Například konverzace během hry je zaznamenána takto:

```
Chat: -=+=-
2s White: hi, how are u
2s Black: fine, thanks
-=+=-
```

Svou speciální notaci má i grafický záznam pozice figurek na hrací desce. Figurky jsou označeny svými iniciálami, pasti jsou označeny písmenem „x“. Celá hrací deska je ohraničena pomocí značek „|“, „-“ a „+“. Také jsou uvedena označení řádků a sloupců. Hrací deska je pozicována tak, že pole a1 se nachází vlevo dole. Notaci znázorňuje Obrázek 2.5, kde je zaznamenána pozice z Obrázku 2.4.

```
 +-----------------+
8| r r r r r r r r |
7|     h e       c |
6| c d x     x     |
5| M       m h     |
4|         E     d |
3|     H     H   C |
2| C D       R D   |
1| R R R R R   R R |
 +-----------------+
   a b c d e f g h
```

Obrázek 2.5 Notace pozice figurek na hrací desce

2.7 Oficiální pravidla pro partii

Pro hraní Arimy byla ustanovena pravidla, jejichž dodržování je vyžadováno při oficiálních hrách, na všech turnajích či mistrovstvích a také pro všechny hry odehrané boty. Pravidla jsou následující:

- Hra musí být zaznamenána. Všechny pohyby provedené ve hře musí být zaznamenány pomocí výše uvedené notace.
- Musí být použita kontrola času. Ta by měla být zvolena tak, aby hra plynula a byla zajímavá pro diváky, ale zároveň aby hra neztrácela kvalitu a hráči neprohrávali na čas. Jelikož zde neexistuje remíza a zvláště při hře botů by opakování tahů mohlo trvat roky, než by se vyčerpaly všechny možné kombinace tahů, je potřeba zvolit časový limit pro partii. Tento limit by měl být minimálně tak vysoký, aby povoloval 80 kol s využitím veškerého času na tah. Alternativou je zavedení maximálního počtu tahů.

- Pokud uplyne celkový čas na partii, je vítěz určen podle těchto pravidel: Vyhrává hráč, který má na konci partie více kusů figurek. Pokud mají hráči figurek stejně, vyhrává ten, který jich měl po odehrání celého kola naposledy více. V opačném případě vyhrává hráč, který je jako druhý na tahu, standardně označen jako stříbrný.
- Hráč může kdykoliv odstoupit a ukončit partii. Tento typ ukončení se však nedoporučuje (horší rozestavení nebo menší počet figurek ještě nemusí znamenat prohru, zatímco v šachách často ano) a podporuje se ukončení hry standardně na hrací desce.

Kontrola času je stanovena několika údaji ve formátu M/R/P/L/G/T.

- M - Objem času na jeden tah.
- R - Počáteční objem času v rezervě.
- P - Procento, udávající kolik zbývajícího času z tahu se převede do rezervy.
- L - Maximální velikost rezervy.
- G - Objem času na celou hru.
- T – Maximální doba trvání jednoho tahu.

Hráč dostane na každý tah čas M, a pokud tento čas spotřebuje, bude mu čas odebírán z rezervy. Pokud hráči dojde čas v rezervě nebo celkový čas jeho tahu překročí hranici T, hráč prohrává. Pokud tah trvá kratší dobu, než je velikost M, je procento P zbývajícího času převedeno do rezervy. Pokud hra trvá déle, než udává T, hra se ukončí a určí se vítěz.

3 Arimaa bot

Jak píše Omar Syed na webu Arimy [1], nápad vytvořit Arimu dostal v roce 1997, kdy šachový superpočítač vyrobený firmou IBM, Deep Blue, porazil mistra světa Garry Kasparova v šachách. Tato událost je považována za přelomovou událost ve vývoji umělé inteligence pro hraní her. Deep Blue vyhrál především díky principu hrubé síly, tzn. prohledávacíl všechny možné tahy do velké hloubky a hledal nejlépe hodnocené koncové pozice.

Omar Syed věděl, že Deep Blue tímto vítězstvím nedokázal svou „inteligenci“, ale pouze svůj ohromný výpočetní výkon. Vždyť Deep Blue dokázal propočítat 200.000.000 pozic za sekundu a Kasparov pouze 3 [3]. Omar chtěl vytvořit hru, která bude mít jednoduchá pravidla pro živé hráče, ale zároveň bude velmi náročné ji hrát pomocí hrubé síly pro počítače. Tímto chtěl dosáhnout vývoje skutečné umělé inteligence. Jak bylo řečeno, v Arimě se figurka v jednom kroku může posunout pouze o jedno pole, ale tah obsahuje 4 kroky. Tím počet možností jednoho tahu narůstá tak, že i nejvýkonnější počítač dokáže propočítat pozici jen několik tahů dopředu.

Tato kapitola pojednává o složitosti hry, o soutěžích v umělé inteligenci a o jejím vývoji.

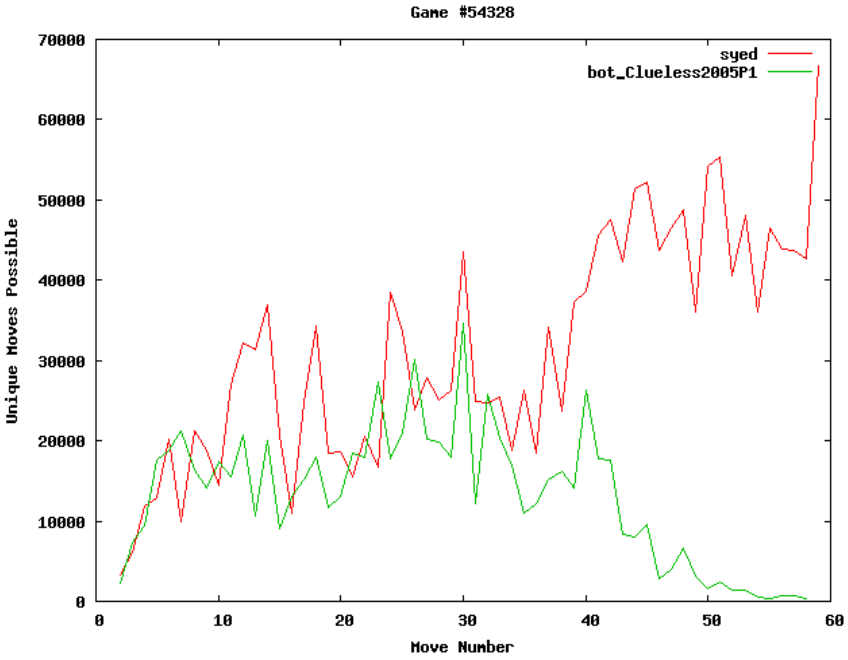
3.1 Složitost

Složitost hry je měřena třemi různými faktory:

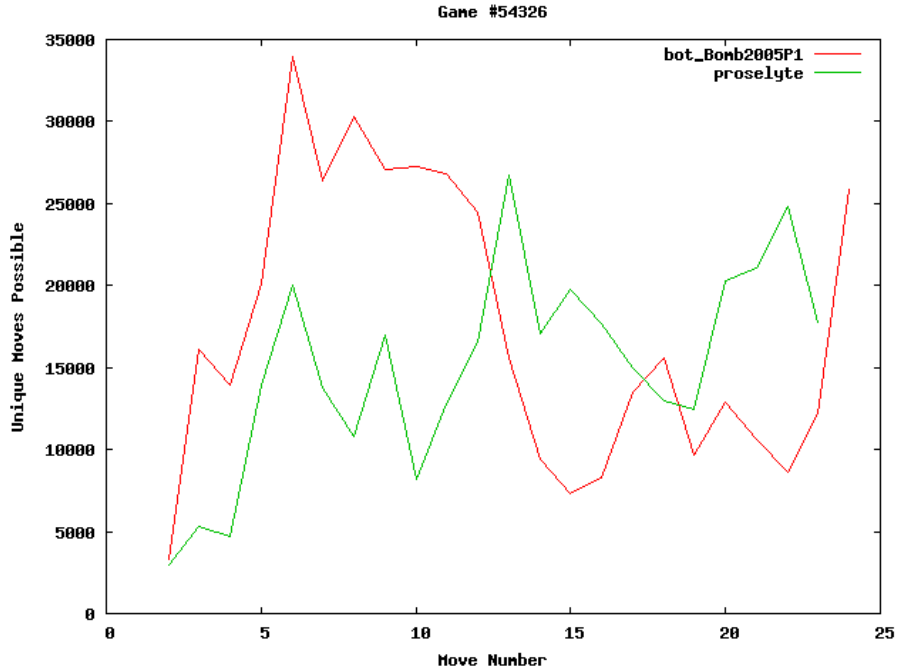
- **Počet přípustných pozic (State-Space Complexity)** – Počet všech možných stavů, do kterých se mohou dostat figurky na hrací desce.
- **Složitost herního stromu (Game-Tree Complexity)** – Počet všech koncových listů prohledávacího stromu před prvním tahem.
- **Počet možných tahů (Branching Factor)** – Průměrný počet všech možných stavů rozestavení figurek na desce po odehrání jednoho tahu.

U šachů se počet přípustných pozic pohybuje od 10^{43} do 10^{50} [4]. Vzhledem k faktu, že Arimaa je podobná šachům, lze očekávat, že počet přípustných pozic bude obdobný. Podle Christ-Jan Coxe byl počet přípustných pozic u Arimy přibližně určen na 10^{43} [5]. Je to o něco méně než u šachů právě proto, že nelze vytvářet nové figurky a hrát s nulovým počtem králíků.

K určení následujících údajů byly využity všechny hry odehrané na herním serveru, které neskončily prohrou na čas nebo rezignací soupeře. Průměrná délka hry je 92 tahů a počet možných tahů je 16064 [6]. Jak je zřejmé z Obrázků 3.1 a 3.2, každá hra může mít zcela jiný průběh.



Obrázek 3.1 Statistika počtu možných tahů u Arimaa hry, vygenerováno pomocí [7]



Obrázek 3.2 Statistika počtu možných tahů u Arimaa hry, vygenerováno pomocí [7]

Každý hráč má před začátkem hry možnost vlastního rozestavení figurek. Počet těchto možností rozestavení pro jednoho hráče je:

$$\binom{16}{8} x \binom{8}{2} x \binom{6}{2} x \binom{4}{2} x \binom{2}{1} x \binom{1}{1} = 64\,864\,800$$

Složitost herního stromu je tedy:

$$\left(\frac{64864800^2}{2}\right) * 16064^{92} \triangleq 10^{402}$$

Šachy mají průměrnou délku hry 80 tahů, počet možných tahů 35 a složitost herního stromu 10^{123} [4]. Ukazuje se tedy, že i když mají počet přípustných pozic podobný, díky volitelné startovní pozici a hlavně velkému počtu možných tahů, je herní strom u Arimy daleko širší.

3.2 Arimaa Challenge

Aby Omar Syed podpořil vývoj umělé inteligence k Arimě, přislíbil odměnu 10 000$ pro programátora, skupinu nebo společnost, která do roku 2020 dokáže vytvořit vítězný program. Tento program musí porazit 3 vybrané živé hráče Arimy v oficiálním zápase o nejméně 3 kolech, takzvaném Arimaa Challenge. Jelikož má jít o souboj s umělou inteligencí a ne s výpočetním výkonem, program poběží na hardware, který lze pořídit do ceny 1000$. Seyd věří, že počítač použitý roku 2004 k prvnímu Arimaa Challenge (Pentium 4 2.4 GHz s 512 MB RAM) měl dostatečný výkon, aby vyhrál, pokud by na něm byl spuštěn ten správný software [8].

Každoročně se odehrávají 2 samostatná mistrovství světa. První je pro počítačové programy, druhé pro živé hráče. Podle výsledků z těchto mistroství jsou vybráni účastníci do Arimaa Challenge.

V Arimaa Challenge od roku 2004 hrály pouze tyto 4 programy:

- 2012 – **bot_Briareus** vytvořil Ricardo Barreira
- 2011 – **bot_marwin** vytvořil Mattias Hultgren
- 2010 – **bot_marwin** vytvořil Mattias Hultgren
- 2009 – **bot_clueless** vytvořil Jeff Bacher
- 2008 – **bot_Bomb** vytvořil David Fotland
- 2007 – **bot_Bomb** vytvořil David Fotland
- 2006 – **bot_Bomb** vytvořil David Fotland
- 2005 – **bot_Bomb** vytvořil David Fotland
- 2004 – **bot_Bomb** vytvořil David Fotland

Do roku 2012 žádný program Arimaa Challenge nevyhrál. Celkové skóre, včetně handicapovaných her, je 76:12 pro živé hráče [8]. Z podrobných výsledků je patrné, že se programy neustále zlepšují a příští roky nás čekají velmi vyrovnané zápasy.

3.3 Arimaa.com

Kolem Arimy se vytvořila rozsáhlá komunita hráčů a programátorů, která Arimu studuje, navrhuje nové strategie a implementace, a tím stále posouvá programy blíže k poražení nejlepších hráčů. Centrem tohoto dění je oficiální stránka Arimaa.com. Zde se nachází velké množství materiálů, od flash tutoriálu, přes herní strategie, až po odborné práce zabývající se Arimou. Hlavním prvkem je zde herní server, kde mohou hrát živí hráči i programy. Podle aktuálního žebříčku má nejlepší živý hráč, Jean Daligault, rating 2440 a nejlepší bot, bot_sharp, rating 2385. Pro srovnání: v roce 2009 bylo jen několik programů, kterým rating přesahoval 1800 [9], což může znamenat, že v Arimě umělá inteligence dohání tu lidskou.

Pro komunikaci mezi Arimaa programy a serverem vytvořil Brian Haskin standardní komunikační protokol, tzv. AEI protokol [10]. AEI pracuje v reálném čase a je plně duplexní. Není nijak závislý na implementaci, protože pracuje se standardním vstupem a výstupem.

4 Hodnotící funkce

Ani nejrychlejší prohledávací funkce nebudou užitečné v tak komplexní hře jako je Arimaa, pokud hra nebude vedena správným směrem. Konečným cílem je najít takové tahy, které by vedly k výhře. Toho obvykle nelze v zadaném časovém limitu dosáhnout. Pokud zkoumané tahy nevedou k výhře, chceme alespoň, aby vedly k co nejlepší situaci na hrací desce. Proto je tak důležitá hodnotící funkce. Její výsledky určují, která pozice je lepší než ostatní a která nejpravděpodobněji povede k vítězství. Tahy vedoucí do těchto pozic se tedy budou hrát.

Tato kapitola pojednává o tvorbě hodnotící funkce a jejích dílčích problémech. Při tvorbě jednotlivých hodnocení jsem vycházel z těchto zdrojů:

- *Analysis and implementation of the Arimaa* [5]
- *Building a World Champion Arimaa Program* [11]
- *Building a Strong Arimaa-playing Program* [12]

Dále jsem na základě studie her, kde jsem toto hodnocení použil, hodnoty v jednotlivých postupech upravil tak, aby nejlépe vyhovovaly zvolené strategii.

4.1 Základní hodnocení

Stejně jako v jiných hrách má Arimaa několik druhů základního hodnocení: podle materiálu, pozice a mobility figurek. Tato jednotlivá hodnocení jsou udávána v bodech. Základ těchto hodnot je převzat z již zmíněné práce [5]. Jak dále popisuje 7. kapitola, každé hodnocení dostane svou jedinečnou váhu, se kterou se přičte k celkovému hodnocení. Například hodnocení materiálu bude mít vyšší váhu než hodnocení pozice. Proto nemůže nastat stav, kdy samotná dobrá pozice figurky bude mít vyšší bodové hodnocení než její existence. Nejdůležitější je tedy rozdíl bodů v rámci jednoho hodnocení.

4.1.1 Hodnocení podle materiálu

Toto hodnocení je založeno na počtu jednotlivých figurek ve hře. Každá figurka je jinak silná a proto má také jiné bodové hodnocení (viz Tabulka 4.1). Hodnocení podle materiálu je nejdůležitějším hodnocením ve většině deskových her a i zde má největší bodový potenciál, protože ztrátu figurek lze jen těžko dohnat poziční výhodou.

Typ figurky	Hodnocení
Slon	16
Velboud	11
Kůň	7
Pes	4
Kočka	3

Tabulka 4.1 Hodnocení jednotlivých figurek

V tabulce chybí hodnota králíka. Tato hodnota nemůže být pevná, protože hodnota králíka v situaci, kdy se jich na hrací desce nachází dalších 7, je menší než hodnota posledního králíka, jehož odebrání by znamenalo konec hry. Jak je vidět v Tabulce 4.2, průměrná hodnota králíka roste s jejich ubývajícím počtem na hrací desce.

Počet králíků ve hře	Hodnocení
8 králíků	69
7 králíků	67
6 králíků	65
5 králíků	62
4 králíci	58
3 králíci	53
2 králíci	47
1 králík	40

Tabulka 4.2 Hodnota králíků podle jejich počtu ve hře

Hodnoty jednotlivých figurek se v průběhu hry živých hráčů mění. Například na začátku se výměna dvou zlatých koček a jednoho zlatého psa za stříbrného velblouda bude zdát jako výhodný tah pro zlatého hráče, ale na konci hry, kdy je málo figurek, bude tato výměna výhodná pro stříbrného hráče. Tento rozdíl v pozdější fázi hry kompenzuje další hodnocení figurek, které v této fázi narůstá.

4.1.2 Hodnocení podle pozice figurek

Další důležitou vlastností figurky je její pozice. Každý typ figurky má ve hře jinou úlohu a měl by se pohybovat v jiné oblasti. Tímto hodnocením docílíme toho, že se figurky přemístí na pozice ideální pro jejich účel. Hodnocení probíhá pomocí tabulek, které jsou souměrné podle osy y a určené pro zlatého hráče. Pro stříbrného hráče je stačí otočit podle osy x. Jediné minusové hodnocení je pozice pastí, protože zde je velká pravděpodobnost ztráty figurky. Také pokud na pasti stojí naše figurka, nelze její pomocí odebrat figurku soupeři. Tyto tabulky tvoří přílohu A.

První tabulka ukazuje hodnocení pozice pro králíka (viz Tabulka A.5). Na jedné straně by měl králík zůstat bránit první řadu, na straně druhé by však měl útočit a dostat se na poslední řadu.

Nejlepší pohyb je po krajích a nejhorší pozice je uprostřed hrací desky, kde může být snadno zmrazen nebo odvlečen do pasti.

Tabulka A.4 ukazuje hodnocení pozice pro kočky a psy. Tyto figurky jsou využívány k bránění pastí, které se nacházejí blíže k základnímu rozestavení. Jak je vidět v tabulce, ideální pozice těchto figurek je na přímo sousedících polích u pasti.

Tabulky A.3 a A2 zobrazuje hodnocení pozice pro velblouda a koně. Tyto figurky by se měly nacházet v centru hrací desky, aby měly největší mobilitu a mohly se rychle dostat ke všem pastím. Centrum hrací desky tvoří 4 pole na souřadnicích d4, d5, e4, a e5. Hlavním účelem těchto figurek je přesouvat soupeřovy figurky do pastí. Zároveň by ale neměly samy útočit a nechat se chytit soupeřovým slonem, proto mají hodnoty pro některé pole na soupeřově straně zmenšené.

Tabulka A.1 zobrazuje hodnocení pozice pro slona. Tato figurka nemůže být zmrazena nebo posunuta nepřítelem. Používá se k útoku nebo ochraně silných figur. Její pozice je ve středu, odkud může rychle zasáhnout, kde je potřeba, nebo zmrazit osamocenou soupeřovu figurku.

4.1.3 Hodnocení podle mobility

Čím více má figurka kolem sebe volných polí pro pohyb, tím je pravděpodobnější, že se rychleji přesune na dobrou pozici. V ideálním případě má každá figurka kolem sebe 4 volná pole, tzn. v každém směru jedno. Pokud v některém směru volné pole není, tudíž se tímto směrem nemůže posunout, je figurka penalizována podle Tabulky 4.3. Může být tedy penalizována až čtyřnásobkem uvedené hodnoty. Pokud je figurka zmrazena, je navíc penalizována podle hodnot v druhém sloupci tabulky. Tímto se snažíme docílit toho, aby figurka měla volné únikové cesty, i když je zmrazena, popřípadě možnosti, jak by ji bylo možné odmrazit. Figurka je penalizována i tehdy, když stojí u hrany hrací desky a nemůže se tedy hýbat všemi směry.

Typ figurky	Penalizace za nemožný směr pohybu	Penalizace za zmražení
Slon	-0,4	0
Velboud	-0,2	-1,2
Kůň	-0,15	-0,8
Pes	-0,1	-0,4
Kočka	-0,1	-0,4
králík	-0,02	-0,08

Tabulka 4.3 Penalizace figurek za omezený pohyb

4.2 Pokročilá hodnocení

Aby byl program silný, musí se kromě základního hodnocení vytvořit také další, rozšířená hodnocení, která nahrazují prohledávání do větší hloubky. Tato hodnocení vycházejí z faktorů, podle nichž se rozhoduje živý hráč. Tato hodnocení ovšem vyžadují další čas, tudíž se může stát, že program zvládne v časovém limitu prohledat možné tahy pouze do menší hloubky. Je tedy potřeba vzít u každého z nich v úvahu jeho kvalitu a rychlost výpočtu. Každé z těchto hodnocení má, stejně jako základní hodnocení, svoji váhu, která bude určena v dalších kapitolách.

4.2.1 Hodnocení podle dosažení cíle

U každého králíka se hledá nejkratší volná cesta, kterou by dosáhl vítězství na poslední řadě hracího pole bez zásahu protihráče. Pro dosažení rychlého výpočtu se nepočítá s pomocí našich dalších figurek. Ze všech nalezených cest se hledá ta s nejmenším počtem kroků. Hodnocení je určeno podle tabulky 4.4. Jak je vidět z hodnocení, je velký rozdíl, zda může králík dosáhnout cíle během jednoho nebo během dvou tahů. Cesty, které trvají 9 a více kroků, jsou hodnoceny 0 body, protože plánování na 3 a více tahů nemá žádný význam.

Počet tahů	Počet kroků	Hodnocení
1	1	5
	2	4,5
	3	4
	4	3,5
2	5	2
	6	1,5
	7	1
	8	0,5
3 a více	9 a více	0

Tabulka 4.4 Hodnocení podle počtu kroků k cíli

4.2.2 Hodnocení podle králičí obrany

Králíci se používají k obraně i k útoku. V obraně se králíci snaží postavit „zeď“ v horizontálním směru, aby mezi nimi nemohl cizí králík projít k první řadě. Do zdi se mohou zapojit i jiné figurky, ty jsou ale hodnoceny jen sekundárně. Při hodnocení se projdou postupně všichni králíci a přičte se 0,15 bodu za každý sloupec (sousedící se sloupcem, v němž se nachází aktuální králík), ve kterém se nachází figurka stejné barvy ve stejné, nebo o 1 vyšší nebo nižší řadě jako aktuální králík. Tato metoda hodnocení má za výsledek snahu králíků stavět jednu společnou zeď místo více kratších.

Snahu stavění zdi z králíků také podporuje fakt, že spojení dvou králíků je tedy dvojnásobně bodově hodnoceno než spojení králíka a jiné figurky.

4.2.3 Hodnocení podle králičího útoku

Jak bylo řečeno, cílem hry je dostat králíka na poslední řadu, proto by bylo dobré králíka, snažícího se tam dostat, podporovat. Na druhou stranu není dobré, aby šel králík naproti cizí figurce, která ho zastaví nebo dokonce zmrazí, a tím pouze umožnil soupeři, aby ho odtlačil do pasti. Proto se hodnotí pouze králíci, kteří před sebou (nebo ve vedlejších sloupcích) nemají žádné cizí figurky. Za každého králíka, který před sebou nemá ve svém sloupci žádnou nepřátelskou figurku, je přičteno hodnocení, které je rovno počtu řad od jeho základní * 0,3 bodu, nebo 0,1 bodu, pokud před sebou má nepřátelskou figurku, ale není to králík. U těchto figurek se dá předpokládat pohyb a uvolnění řady. Dále ještě získávají 0,1 bodu * počet řad od jeho základní řady za každý přilehlý sloupec, ve kterém není před ním žádná nepřátelská figurka. Toto hodnocení má za následek nečinnost králíků, kteří před sebou mají nepřátelské figurky, ale jejich přímý pohyb k cíli, pokud se před nimi uvolní cesta.

4.2.4 Hodnocení podle přístupu k centru

Silné figurky by měly mít rychlý přístup k centru, odkud jsou schopné se rychle přemístit k jakékoliv pasti. Proto se u koně, velblouda a slona hodnotí počet kroků, za které jsou schopni se do něj přemístit. Počítá se cesta, během níž nebude figurka zmražena a kterou je schopna projít během jednoho tahu. Jelikož jsou koně dva, počítá se pouze nejkratší cesta jednoho z nich. Hodnocení je určeno podle Tabulky 4.5.

Typ figurky	Poček kroků				
	0	1	2	3	4
Slon	1	0,8	0,7	0,6	0,5
Velboud	0,9	0,7	0,6	0,5	0,4
Kůň	0,8	0,6	0,5	0,4	0,3

Tabulka 4.5 Hodnocení podle přístupu k centru

5 Generování tahů

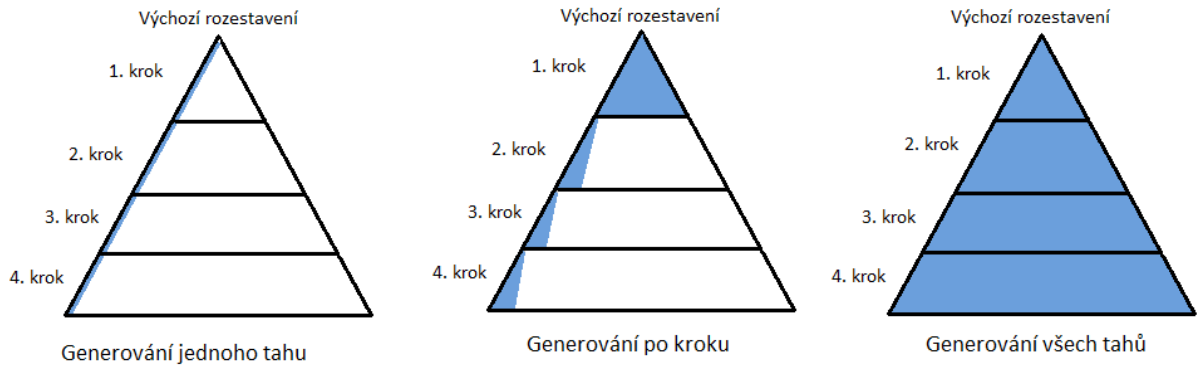
Generování tahů je dalším z klíčových prvků herního programu. Nejdůležitějšími vlastnostmi jsou zde rychlost generování a počet tahů. Samozřejmostí je korektnost vygenerovaných tahů. Tato kapitola popisuje teorii i implementaci generování tahů v programu.

První část kapitoly se věnuje samotnému vytváření tahů pomocí generování jednotlivých možných kroků. Jsou zde popsány možnosti generování, jejich výhody a nevýhody a výběr nejlepší z možností. Dále je zde popsána aplikace tohoto postupu v programu a znázornění její funkce na vývojovém diagramu.

Některé z vygenerovaných tahů vedou do totožného nebo špatného koncového rozestavení a tudíž mají minimální pravděpodobnost, že je program zvolí jako nejlepší tah. Druhá část kapitoly se proto zabývá možnostmi ořezávání množiny vygenerovaných tahů. Jsou navrženy faktory, které by měl splňovat každý tah, který má zůstat v množině pro další zpracování. Tyto faktory jsou testovány, aby se určil jejich přínos pro program.

5.1 Generování tahů

Jelikož se každý tah skládá až ze 4 samostatných kroků a následné zpracování se provádí po jednom tahu, naskýtají se různé způsoby generování tahů. Hlavní postupy jsou zobrazeny na Obrázku 5.1. Každý trojúhelník značí vytváření tahů z výchozí pozice pomocí generování jednotlivých kroků. Spodní hrana trojúhelníku znázorňuje všechny možné tahy. Modré oblasti značí kroky, které jsou vygenerovány během generování prvního tahu. V případě generování jednoho tahu probíhá generování takto: z výchozího rozestavení se vygeneruje jeden možný krok. Z nového rozestavení se vygeneruje opět jeden krok, atd. V případě generování po kroku se z výchozího rozestavení vygenerují všechny možné kroky. Při generování druhého kroku se vybere první nové rozestavení a z něj se opět vygenerují všechny možné kroky, atd. Při generování všech tahů se z výchozího rozestavení vygenerují všechny možné kroky a z každého z nich se potom dále generují všechny možné kroky.



Obrázek 5.1 Hlavní postupy generování tahů

Mohlo by se zdát, že generování po jednom tahu je nejlepší varianta. Má nejmenší paměťovou náročnost a v případě použití prohledávacího algoritmu Alfa-Beta (viz Kapitola 6.2) také vygeneruje pouze ty tahy, které se dále použijí. Problém je ale v tom, jak algoritmus pozná, kde má navázat na minulé generování. Řešením je ukládání aktuální pozice do mnoha proměnných nebo opětovné generování toho, co již jednou vygenerováno bylo. Proto má tato varianta největší průměrný čas generování jednoho tahu.

Naopak generování všech tahů tento problém řešit nemusí, protože vygeneruje všechny tahy zároveň. Jeho průměrný čas generování jednoho tahu je tedy nejmenší. Jelikož počet vygenerovaných tahů dosahuje desítek tisíc, potýká se tento postup s velkou paměťovou náročností a také vygeneruje nejvíce tahů, které nebudou nikdy použity.

Generování po kroku má všechny výhody předešlých řešení a minimalizuje jejich nevýhody. Při generování nového tahu se vezme nepoužité rozestavení z minulého generování, které vzniklo z méně než 4 kroků, a vygenerují se pro něj všechny možné kroky. Generování z každého rozestavení se tedy provádí najednou a není potřeba ukládat aktuální stav. Tento algoritmus má malý průměrný čas na generování jednoho tahu a jeho paměťová náročnost je také malá, protože pouze ukládá tahy vygenerované jedním krokem, jejichž průměrný počet je 11, a tudíž i počet vygenerovaných kroků, které nebudou použity, je zanedbatelný. Tento postup je proto vybrán pro implementaci.

5.1.1 Implementace

Program má generování rozvržené do 4 vrstev. Jednotlivé vrstvy provádí generování jednoho kroku ze zadané pozice. Každá vrstva je určena pro krok zadaného pořadí. To znamená, že první vrstva generuje jednoduché kroky, tlačení a tažení vycházející z výchozí pozice. Další vrstvy generují další kroky pro již vygenerovaný tah, respektive z pozice, do které tento tah vede. Každá vrstva obsahuje svou množinu tahů, kam se ukládají všechny vygenerované tahy a zároveň odtud další vrstva získává vstupní tahy. Například 2. vrstva má množinu, kam ukládá tahy vytvořené ze

vstupního tahu zvětšeného o vygenerovaný jednoduchý krok. Pokud vygeneruje pohyb tlačení nebo tažení, uloží vzniklý tah do množiny 3. vrstvy, protože tento tah bude mít velikost 3 kroků.

Každá vrstva vygeneruje všechny možné kroky ze zadaného rozestavení. Pokud není poslední a pokud vygenerovala některé nové kroky, spustí generování prvního nového tahu na další vrstvě. Algoritmus uvnitř vrstvy je znázorněn na Diagramu 5.1. Je zde uveden způsob generování nových pohybů i spuštění generování na další vrstvě. Během generování je tedy na každé vrstvě vytvořena množina tahů z jedné pozice. Maximální velikost jedné této množiny byla experimentálně určena na 70 tahů. Maximální počet zároveň uložených tahů ve všech vrstvách je tedy pouze 280.

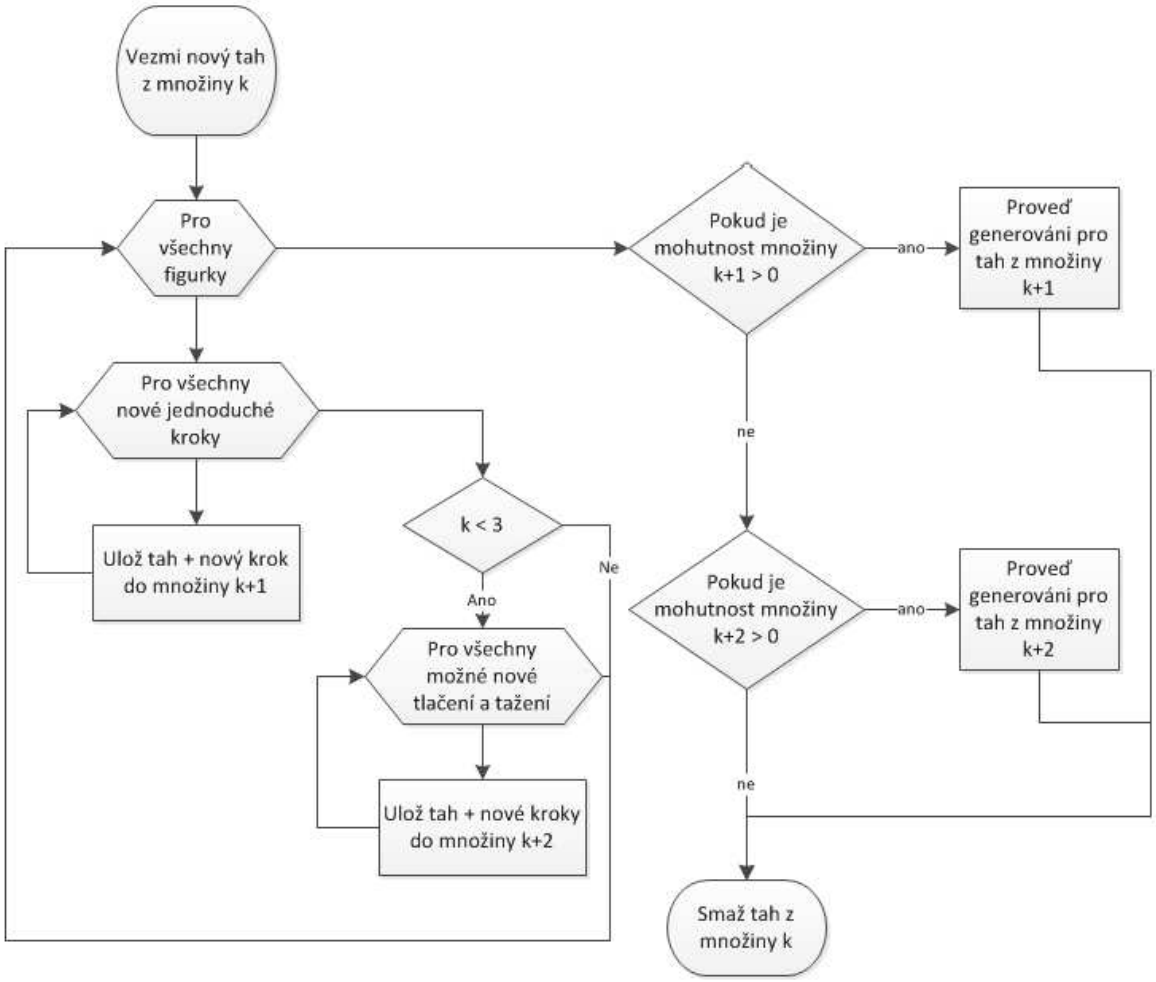


Diagram 5.1 Generování tahů v jedné vrstvě

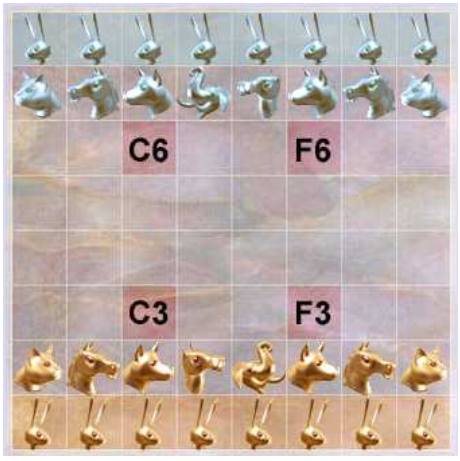
5.2 Ořezávání tahů

Ořezávání tahů má za úkol zmenšit množinu vygenerovaných tahů. Čím menší množina bude, tím užší bude vyhledávající strom a tím více času zbude programu na hodnotící algoritmy. Největší množinu tahů, které jsou odstraněny, tvoří tahy, které vedou do totožných koncových rozestavení a jejichž odstranění nebude mít žádný negativní vliv na program. Pro odstranění dalších tahů jsem navrhl tyto faktory: zachování vlastních figurek, navazující kroky a maximální velikost tahu. Pokud některý tah nebude obsahovat všechny tyto faktory, bude odstraněn. Odstranění nevyhovujících tahů značně zrychlí chod programu, ale zároveň již může mít na program negativní vliv. Pokud by nebyly nevyhovující tahy odstraněny, mohly by být vybrány za nejlepší. Cílem testování je určit, jak velká množina tahů bude odstraněna a zároveň, jak často by tyto tahy mohly být určeny za nejlepší.

5.2.1 Data pro testy

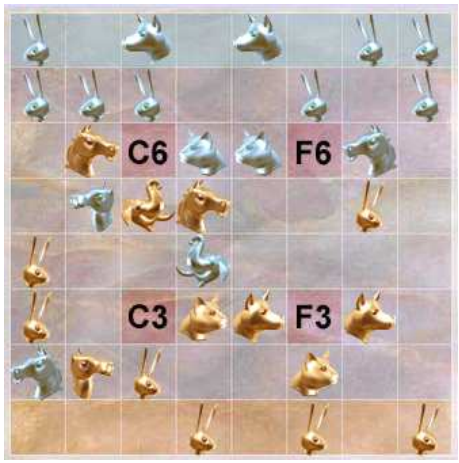
Testování faktorů probíhá ve dvou různých krocích. Nejprve proběhne testování, které zjišťuje, kolik procent vygenerovaných tahů se odstraní kvůli nepřítomnosti daného faktoru. Výsledky těchto testů jsou zaznamenány přímo u popisu jednotlivých faktorů. Nakonec je testován výskyt daného faktoru v tazích z historie odehraných her na herním serveru. Tyto výsledky jsou společně s vyhodnocením uvedeny v poslední části této kapitoly.

Odstraňování tahů tedy může ovlivnit průběh hry, protože se změní množina rozestavení figurek během hry. Testování, které bude měřit počet všech vygenerovaných možných tahů během celé hry, bude mít chybné výsledky. Z tohoto důvodu se poměr odstraněných tahů určuje na jednotlivých zvolených rozestaveních. Tato rozestavení reprezentují průběh celé hry a jsou zobrazena na Obrázcích 5.1 až 5.6. První 2 rozestavení jsou vybrána ze zahájení, další 3 znázorňují střední hru a poslední je z koncovky. Tahy se generují vždy za toho hráče, který je na tahu.



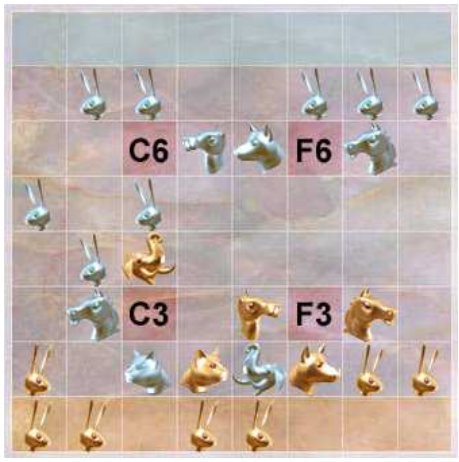
Obrázek 5.1 1. rozestavení (pozice 1s)

Obrázek 5.2 2. rozestavení (pozice 7g)



Obrázek 5.3 3. rozestavení (pozice 24s)

Obrázek 5.4 4. rozestavení (pozice 23g)



Obrázek 5.5 5. rozestavení (pozice 32g)

Obrázek 5.6 6. rozestavení (pozice 84s)

Druhé testování je zaměřeno na určení výskytu jednotlivých faktorů u tahů, které jsou vybrány z množiny vygenerovaných tahů jako nejlepší a následně jsou zahrány. Protože testování neprobíhá na množině všech vygenerovaných tahů, ale pouze na jediném tahu z této množiny, je potřeba zvolit daleko větší počet herních rozestavení než u prvního testování, abychom dosáhli stejného počtu testovaných tahů. Z tohoto důvodu je pro druhé testování použita historie všech her odehraných na herním serveru z období od ledna 2010 do března 2012. Aby testování neprobíhalo na tazích, které jsou špatné nebo provedené v časové tísni, musí každá hra splňovat tyto podmínky:

- Oba hráči mají rating přes 1800
- Čas na tah je minimálně 2 minuty
- Hra je hrána na rating

Tato historie obsahuje 92071 her, z toho jich 1672 splňuje dané podmínky. Z těchto her je dále odstraněn poslední tah obou hráčů: Tah prohrávajícího, protože ví, že prohraje a jeho tah již nemusí být tím nejlepším možným. Tah vítěze, protože pokud je více tahů, které by ukončily danou hru, hráč

již nevybírá ten nejlepší. Takto je vytvořena množina o 70381 tazích, která je použita pro druhé testování.

5.2.2 Unikátní tahy

Generovat více tahů vedoucích do totožného rozestavení nemůže přinést programu žádné zlepšení, a proto jsou odstraněny. Tyto tahy lze rozeznat již během generování. Pokud tah, který se neskládá z maximálního počtu kroků, vede do rozestavení, do kterého se dá dojít tahem složeným z méně kroků, nebo tahem složeným ze stejného počtu kroků, který byl již dřív vygenerován, nemůže se stát, že přidáním dalších kroků do maximálního počtu dojde tento tah do nového unikátního rozestavení. Odstraněny jsou také všechny tahy, které vedou do výchozího rozestavení, jelikož tyto tahy jsou podle pravidel nelegální.

Pro rozpoznání jednotlivých rozestavení se používá hash hrací desky, který je popsán v kapitole 7.2. Program se snaží odstranit zbytečné tahy co nejdříve a tím ušetřit výpočetní čas. Při generování tahů z výchozího rozestavení si program při generování každého kroku ověřuje hash nově vzniklého rozestavení. Nejprve zkontroluje, zda se tento hash již nevyskytnul u tahu s menším počtem kroků. Neověřuje všechny hashe, ale pouze ty, které vytvořil tah, který má o 2 kroky méně než ten současný. Poté zkontroluje hashe vzniklé z tahů, které mají stejný počet kroků. Pokud ani tam hash nenašel, zapíše jej jako nový, a tah i s novým krokem předá k dalšímu zpracování.

Pravidla povolují pouze 2 totožná rozestavení figurek během celé hry, proto si program uchovává i hash již odehraných rozestavení a jejich počet výskytu. Každý vygenerovaný tah je ještě porovnán s historií, než je použit jako možný tah.

V Tabulce 5.1 je zobrazeno testování počtu unikátních tahů v množině všech vygenerovaných tahů z jednoho rozestavení. V prvním sloupci jsou uvedena jednotlivá rozestavení, v druhém je počet všech vygenerovaných tahů, v třetím se nachází počet tahů, které vedou do unikátního rozestavení (dále jen unikátní tahy) a v posledním je výsledná procentuální úspěšnost, nebo také procento zachovaných tahů. Výsledky ukázaly, že úspěšnost se velmi liší. Je to dáno typem pozice a tím, jak na sebe jednotlivé kroky navazují. Průměrná úspěšnost unikátnosti tahu je okolo 9,5%.

	Vygenerované tahy	Unikátní tahy	Úspěšnost
1. rozestavení	23 886	3 412	14,28%
2. rozestavení	396 580	24 092	6,07%
3. rozestavení	635 459	40 833	6,43%
4. rozestavení	121 474	10 657	8,77%
5. rozestavení	386 431	25 393	6,57%
6. rozestavení	2 064	325	15,75%

Tabulka 5.1 Míra unikátních tahů ve všech vygenerovaných tazích

5.2.3 Zachování vlastních figurek

Prvním faktorem, který by měl obsahovat každý tah, je zachování vlastních figurek. To znamená, že by během tahu neměla být z hrací desky odstraněna žádná naše figurka (ať už vlastním přesunem figurky do nechráněné pasti nebo přesunem figurky ze sousedství pasti, na které se nachází naše figurka, která není chráněna žádnou jinou figurkou).

Program tento faktor kontroluje při aplikaci každého nového kroku na aktuální rozestavení. Jsou ale zde definovány 3 výjimky, při jejichž splnění se tah neodstraní:

- Pokud lze ve zbylých krocích dojít králíkem na cílovou řadu.
- Pokud byla naše figurka odstraněna při pohybu tlačení nebo tažení a zároveň můžeme ve zbývajících krocích odstranit i soupeřovu figurku.
- Pokud se figurka odstraní následkem přesunu figurky ze sousedství pasti v prvním kroku.

Třetí výjimkou se zabrání tomu, aby byla silná figurka uvězněna u slabé figurky, která je na pasti. Tato výjimka je definována proto, aby nenastaly situace, ve kterých se slon nemůže pohnout, protože chrání králíka na pasti, nebo v koncovkách, kdy je na hrací desce jen málo figurek a pohyblivost jedné je důležitější než existence obou. Nikdy se tedy neodstraní tah, který by měl zásadní vliv na další pokračování hry v podobě dosažení cíle nebo odebrání nepřátelské figurky. Tabulka 5.2 zobrazuje míru přítomnosti tohoto faktoru ve všech unikátních tazích vygenerovaných pro zadané rozestavení.

	Unikátní tahy	Počet tahů s faktorem	Úspěšnost
1. rozestavení	3 412	2 190	64,19%
2. rozestavení	24 092	23 131	96,01%
3. rozestavení	40 833	38 693	94,76%
4. rozestavení	10 657	8 478	79,55%
5. rozestavení	25 393	23 929	94,23%
6. rozestavení	325	279	85,85%

Tabulka 5.2 Míra tahů zachovávající vlastní figurky ve všech unikátních tazích

5.2.4 Navazující kroky

Tento faktor poprvé popsal Haizhi Zhong ve své práci *Building a Strong Arimaa-playing Program* [12]. Jednotlivé kroky v tahu na sebe mohou nebo nemusí navazovat. Dva kroky na sebe navazují, pokud jejich provedení v obráceném pořadí povede buď do jiné pozice, nebo bude neproveditelné. Z tohoto pohledu existuje 5 typů tahů, které obsahují:

- 4 navazující kroky
- 3 navazující kroky + 1 samostatný krok

- 2 navazující kroky + 2 samostatné kroky
- 2 navazující kroky + 2 navazující kroky
- 4 samostatné kroky

Tahy s menším počtem kroků jsou podmnožinou z jednoho z výše uvedených typů. Zhong analyzoval všechny hry odehrané ve finále World Arimaa championship roku 2005 a zjistil, že poslední typ se v těchto hrách objevil pouze v méně než 2% a nahrazení těchto tahů jiným typem by v žádném z případů nebylo kritické. Z těchto důvodů jsem jako další faktor zvolil povinnost návaznosti prvního a druhého kroku ve všech vygenerovaných tazích.

Program přítomnost tohoto faktoru nekontroluje, ale přímo generuje pouze ty tahy, které jej budou splňovat. Z tohoto důvodu má zcela změněnou druhou fázi generování. Dále jsou uvedeny typy 2 prvních kroků, které program generuje, a jejich příklady z rozestavení zobrazeném na Obrázku 2.4.

- **Tlačení** Tah zlatého hráče: me5n Ee4n
- **Tažení** Tah zlatého hráče: Ma5s ca6s
- **Kroky stejnou figurkou** Tah zlatého hráče: Ca2n Ca3n
- **Pohyb figurkou k zmražené figurce a její následný krok** Tah zlatého hráče: Dg2e Ch3w
- **Pohyb figurkou v druhém kroku na pole, které uvolnila figurka v prvním kroku** Tah zlatého hráče: Db2n Rb1n
- **Tlačení figurky na uvolněné pole**
- **Pohyb figurkou v druhém kroku na past, která se díky prvnímu kroku stala bezpečnou** Tah stříbrného hráče: rf8s hf5n
- **Pohyb figurkou k figurce, která díky prvnímu kroku zamrzla** Tah stříbrného hráče: hf5s dh4w
- **Pohyb z pasti, na kterou může jiná figurka odtlačit nepřátelskou figurku**

Tabulka 5.3 zobrazuje míru přítomnosti 2 nebo více navazujících kroků ve všech unikátních tazích vygenerovaných pro zadané rozestavení.

	Unikátní tahy	Počet tahů s faktorem	Úspěšnost
1. rozestavení	3 412	3 150	92,32%
2. rozestavení	24 092	15 455	64,15%
3. rozestavení	40 833	22 946	56,19%
4. rozestavení	10 657	7 499	70,37%
5. rozestavení	25 393	16 224	63,89%
6. rozestavení	325	306	94,15%

Tabulka 5.3 Míra tahů obsahující navazující kroky ve všech unikátních tazích

5.2.5 Maximální velikost tahu

Posledním faktorem je velikost tahu. Tento faktor udává, že by každý tah měl obsahovat maximální počet kroků. Domnívám se, že vzhledem k povaze cíle hry přesunout figurku na opačný konec hrací desky je jen málo situací, při kterých nevyužití celého počtu kroků povede k nejlepšímu rozestavení. Tyto tahy nanejvýš přinesou malou poziční výhodu, ale nikdy jejich nepřítomnost nerozhodne hru.

Program během generování kontroluje, zda tahy, které neobsahují 4 kroky, mají ze svého koncového rozestavení další možný krok. Pokud ho nemají, ať už kvůli tomu, že nejsou generovány, nebo kvůli absenci některého z výše uvedených faktorů, je tento tah považován za možný. Jinak se provede jeho další generování. Tabulka 5.4 zobrazuje míru přítomnosti tohoto faktoru ve všech unikátních tazích vygenerovaných pro zadané rozestavení.

	Unikátní tahy	Počet tahů s faktorem	Úspěšnost
1. rozestavení	3 412	2 872	84,17%
2. rozestavení	24 092	21 041	87,34%
3. rozestavení	40 833	35 975	88,10%
4. rozestavení	10 657	9 087	85,27%
5. rozestavení	25 393	22 072	86,92%
6. rozestavení	325	238	73,23%

Tabulka 5.4 Míra tahů obsahující maximální počet kroků ve všech unikátních tazích

5.2.6 Vyhodnocení

Testy ukázaly, že díky absenci každého z faktorů lze odebrat značné procento z možných tahů. Také bylo zjištěno, že žádný z odebraných tahů nemá rozhodující vliv na vývoj hry. To znamená, že díky žádnému z odebraných tahů nelze značně vylepšit svou pozici odebráním soupeřovy figurky, ztrátou vlastní figurky, či dokonce vyhrát. Tabulka 5.5 udává kolik tahů se z jednotlivých rozestavení vygeneruje, pokud se odstraní všechny tahy, kterým chybí některý z definovaných faktorů.

	Unikátní tahy	Počet tahů se všemy faktory	Úspěšnost
1. rozestavení	3 412	1 776	52,05%
2. rozestavení	24 092	13 296	55,19%
3. rozestavení	40 833	20 353	49,84%
4. rozestavení	10 657	4 774	44,80%
5. rozestavení	25 393	14 209	55,96%
6. rozestavení	325	198	60,92%

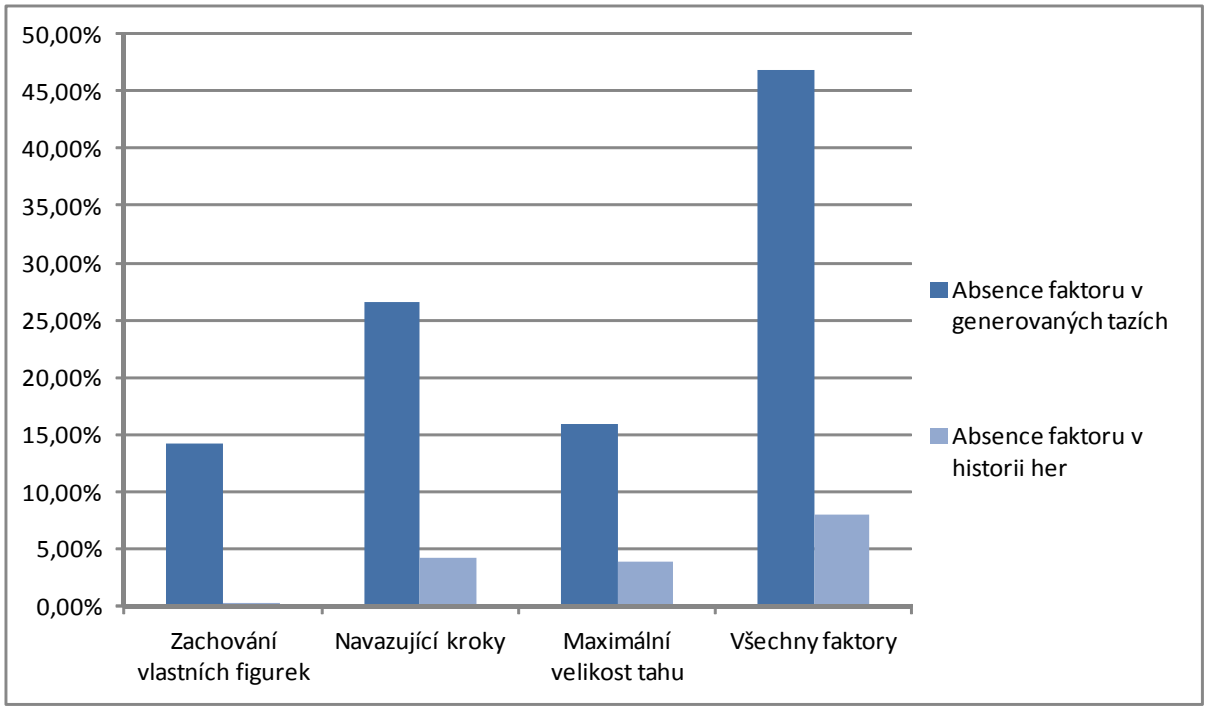
Tabulka 5.5 Míra tahů obsahující všechny faktory

Druhé testování ukázalo, kolik tahů zahraných hráči s ratingem nad 1800 neobsahuje některý z faktorů. Tyto výsledky jsou zobrazeny v Tabulce 5.7. Dle očekávání tahy bez faktoru zachovávající vlastní figurky se objevují pouze v 0,1% tazích. Také tahy bez dalších faktorů se objevují v několikanásobně menším procentuálním zastoupení než v množině vygenerovaných tahů.

Faktory	Absence faktoru	
	v generovaných tazích	v historii her
Zachování vlastních figurek	14,24%	0,10%
Navazující kroky	26,49%	4,11%
Maximální velikost tahu	15,83%	3,84%
Všechny faktory	46,87%	7,91%

Tabulka 5.6 Souhrné výsledky testování

Testování ukázalo, že by se odstranilo 46,78% tahů díky zavedení povinnosti faktorů. Déle bylo určeno, že tyto tahy se objevují pouze v 7,91% v množině odehraných tahů v historii her. Rozdíl těchto hodnot názorně zobrazuje Graf 5.1, kde je vidět značný rozdíl mezi výsledky obou testů. Jelikož tyto tahy nejsou pro hru klíčové, je zavedením povinnosti těchto faktorů dobrým krokem k výraznému zrychlení programu.



Graf 5.1 Porovnání absence faktorů v generovaných tazích a v historii her

6 Prohledávání

Třetím z klíčových problémů umělé inteligence pro hraní her je určení nejlepšího tahu v zadaném čase. Pro jeho nalezení se používají prohledávací stromy, které se větví podle všech možných tahů do co největší hloubky, a podle koncových stavů se hledá nejlepší aktuální tah. V Arimě, stejně jako v šachách, se používá prohledávací strom Minimax s Alfa-beta ořezáváním. Tento program jsem specializoval na nejčastější časovou variantu používanou také při Arimaa Challenge: 2/6/100/0/8/6. Z tohoto důvodu, rozepsaném podrobněji v následující části této kapitoly, jsem stanovil pevnou hloubku prohledávání na 2 tahy. Jako časově výhodnou kompenzaci jsem navrhl dvě nové náročné hodnotící funkce, popsané na konci kapitoly. Tyto funkce částečně nahradí prohledávání o další 2 tahy a najdou klíčové možnosti těchto tahů.

6.1 Parametry prohledávání

Pomocí AEI protokolu lze nastavit různá omezení hry. Boti, kteří jsou dostupní na herním serveru, využívají omezení hloubky prohledávání na 2 tahy (8 kroků) a velikost hash tabulky na 2 MB. Jelikož mé testování je založené na těchto botech, přizpůsobil jsem program tak, aby vyhovoval těmto omezením.

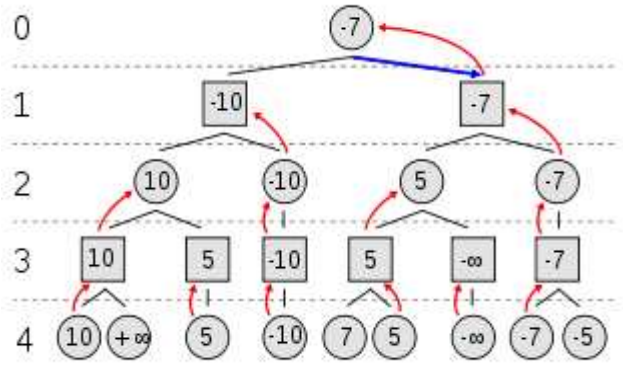
Prohledávání do hloubky 2 tahů trvá od několik sekund do několika minut Při průměrném počtu možných tahů 16 tisíc a při prohledávání do hloubky 3 tahů by byl tento čas daleko za hranicí 2 minuty na tah. Toto omezení se projeví až na konci partie v koncovkách, kdy je již na hrací desce málo figurek, počet možných tahů se radikálně zmenší a možná dosažitelná hloubka prohledávání je až 4 tahy.

Pro zrychlení prohledávání, při němž dochází k častému výskytu stejných uzlů, se využívá Transposition table. Každý záznam v Transposition table by měl obsahovat hash daného rozestavení, nejlepší možný tah pro oba hráče a hodnocení dané pozice. Minimální velikost tohoto záznamu by byla 158 bitů. Do tabulky o 2MB by se tedy vešlo 100 tisíc záznamů. Průměrný vyhledávající strom s hloubkou 2 tahů má 256 milionů koncových uzlů. Z důvodu tohoto omezení jsem tedy Transposition table neimplementoval.

6.2 Prohledávací metoda

6.2.1 Minimax

Minimax je algoritmus pro nalezení nejlepšího tahu v hrách pro dva hráče. Algoritmus vychází ze zadané pozice a pomocí souboru všech možných tahů se rozvětví na všechny možné budoucí stavy. Protože ani velký výkon dnešních počítačů nedokáže v dostupném čase vytvořit celý strom, větvení se zastaví v určité, předem dané hloubce. Větvení se také zastaví, pokud aktuální pozice znamená vítězství pro jednoho z hráčů. Poté se všechny koncové stavy hodnotí funkcí popsanou v kapitole 4. Algoritmus předpokládá, že oba hráči (označíme je A a B) hrají svůj nejlepší možný tah, tudíž se snaží dosáhnout stavu, který bude mít pro ně nejlepší hodnocení. Jelikož hráči hrají proti sobě, hráč A hledá stav s největším hodnocením a hráč B s nejmenším hodnocením. Po vytvoření stromu a hodnocení pozic se algoritmus vrací zpět a každému nadřazenému uzlu určí jeho hodnotu. Pokud je v uzlu na tahu hráč A, dostane uzel nejvyšší hodnotu ze svých potomků; pokud je v uzlu na tahu hráč B, dostane uzel nejmenší hodnotu ze svých potomků. Výsledný tah je ten, jímž se dostaneme do potomka kořene, jenž má jeho hodnotu. Průběh algoritmu je znázorněn na Obrázku 5.1.



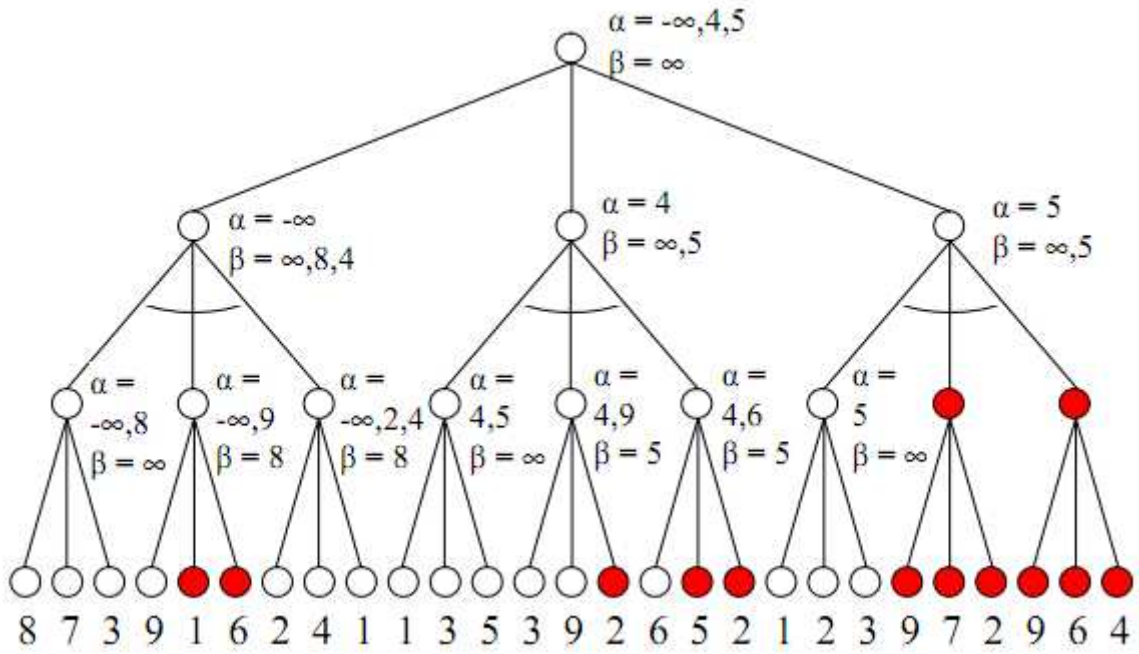
Obrázek 5.1 Průběh algoritmu Minimax, převzato z [13]

Algoritmus má malou paměťovou složitost, jelikož nemá uložen celý strom v paměti, ale pouze cestu od kořene k aktuálnímu uzlu. Časová složitost záleží na faktoru větvení. Předpokládejme, že z každého uzlu je možné provést x tahů a výpočet se provádí do hloubky h, časová náročnost je potom rovna x^h. Časovou složitost lze zmenšit použitím algoritmu Alfa-beta ořezávání.

6.2.2 Alfa-Beta ořezávání

Alfa-beta je vylepšený algoritmus Minimax. Jeho princip spočívá v předčasném ukončení výpočtu uzlů, které nemohou obstát proti již spočítaným uzlům. Jak je zobrazeno na Obrázku 5.2, červené uzly se již nemusí počítat. Tato metoda zavádí do algoritmu 2 meze, horní (β) a dolní (α). Pokud při

výpočtu uzlu bude dolní mez rovna nebo vyšší horní mezi, může být výpočet uzlu ukončen a výpočet jeho potomků bude vynechán. Při inicializaci algoritmu je nastavena α na –∞ a β na ∞. Při výpočtu každého potomka se načtou aktuální hodnoty od rodiče. Pokud je aktuální hodnota uzlu, kde je na tahu hráč A, vyšší než α, uloží se tato hodnota do α. Pokud je aktuální hodnota uzlu, kde je na tahu hráč B, vyšší než β, uloží se tato hodnota do β. Obrázek 5.2 znázorňuje výpočet nejlepšího tahu za použití Alfa-beta ořezávání.



Obrázek 5.2 Průběh algoritmu Alfa-beta, převzato z [14]

6.2.3 Podpora ořezávání

Účinnost ořezávání lze zvýšit přesunem tahů s vyšším ohodnocením na začátek prohledávání. Toho se snažím dosáhnout 2 způsoby: přesunem tahů, které odebírají cizí figurky na začátek vygenerované množiny tahů a generováním tahů pro jednotlivé figurky v takovém pořadí, v kterém se s nimi statisticky nejvíce pohybuje. Tabulka 6.1 ukazuje četnost pohybů s jednotlivými figurkami z historie her popsané v Kapitole 5.2.1 a jejich nové pořadí při generování.

Typ figurky	Počet tahů na typ	Počet tahů na jednu figurku	Pořadí zpracování
Slon	97 081	97 081	1.
Velboud	43 793	43 793	2.
Kůň	56 272	28 136	5
Pes	81 763	40 882	3.
Kočka	57 954	28 977	4.
králík	121 213	15 152	6.

Tabulka 6.1 Četnost pohybů figurek

Tabulka 6.2 udává počet vytvořených uzlů ve všech prohledávacích stromech během hry mezi hráči bot_Clueless2011P1 a bot_ugrh o délce 72 tahů, dostupné na herním serveru [15] pod id 227555 . Testování udává, že díky podpoře ořezávání se vytváří okolo 50% uzlů méně než bez ní.

Metoda	Počet uzlů	Počet uzlů na tah	Počet uzlů
Minimax	14 069 822 065	97 707 098	100,00%
Alfa-beta ořezávání	231 151 421	1 605 218	1,64%
Alfa-beta ořezávání s podporou ořezávání	120 471 043	836 604	0,86%

Tabulka 6.2 Počet vytvořených uzlů během prohledávání

6.3 Hodnocení podle dosažení cíle v příštím tahu

Toto hodnocení zkoumá, zda existuje možnost přesunout během jednoho tahu králíka na poslední řadu a tím ukončit hru. Na rozdíl od Hodnocení podle dosažení cíle, popsaného v Kapitole 4.2.1, zde mohou pomoci ostatní figurky. Prověřuje se každý králík zvlášť. Podle vzdálenosti králíka od poslední řady a podle zvolené cesty se určí možná pomoc okolních figurek:

- **Bez pomoci** (cesta může mít velikost 1-4 pole) – ekvivalentní s výše zmíněným hodnocením. Králík se může pohybovat pouze po volných polích, na kterých nebude zmrazen.
- **S pomocí, během 1 kroku** (cesta může mít velikost 1-3 pole) – pomocná figurka může králíka odmrazit, odmrazit pole na jeho cestě nebo mu uvolnit pole na jeho cestě.
- **S pomocí, během 2 kroků** (cesta může mít velikost 1-2 pole) – pomocná figurka může odtáhnut/odtlačit nepřátelskou figurku, která mrazí králíka nebo pole na cestě, nebo může odtáhnout nepřátelskou figurku, která se nachází na cestě, pokud pak pole, na kterém stála, nebude mrazit. Také může odmrazit králíka nebo pole na cestě pomocí 2 kroků.

Toto hodnocení, na rozdíl od těch uvedených v Kapitole 3, rozlišuje, který hráč je zrovna na tahu. Pokud je na tahu hráč, který může během svého tahu přesunout králíka na poslední řadu, je hodnocení této pozice rovno jeho maximu -1. Jednička se odečítá proto, aby byla upřednostněna ta rozestavení, kde se již králík na poslední řadě nalézá. Pokud je na tahu druhý hráč, určí se hodnocení podle Tabulky 6.3 a podle počtu potřebných kroků.

Počet kroků	Hodnocení	
	Hráč je na tahu	Hráč není na tahu
1	maximum -1	2
2	maximum -1	1,8
3	maximum -1	1,4
4	maximum -1	1

Tabulka 6.3 Hodnocení podle dosažení cíle

6.4 Hodnocení podle odebrání figurky v příštím tahu

Toto hodnocení hledá nejsilnější nepřátelskou figurku, kterou by mohl hráč během jednoho tahu odstranit v některé z pastí. Nahrazuje hodnocení kontrolování pastí, kde se použila stejná váha každé hlídané pasti, ať už mohla být nepřítelem využita nebo ne. Hodnocení vychází z Tabulek 4.1 a 4.2 u Hodnocení materiálu. Hodnocení je také závislé na hráči, který je na tahu. Pokud je hráč na tahu, získává lepší hodnocení než hráč, který je na tahu až jako druhý, a tedy nemá jistotu, že se tato možnost naskytne i po tahu prvního hráče.

Algoritmus hledání zobrazuje Diagram 6.1. Je rozdělen na 2 části. Nejprve se snaží nalézt soupeřovy figurky, které se nacházejí v sousedství pasti a které by v ní mohl odstranit. Pokud neuspěje, hledá figurky, které by mohl do pasti dostat dvojitým tlačením či tažením.

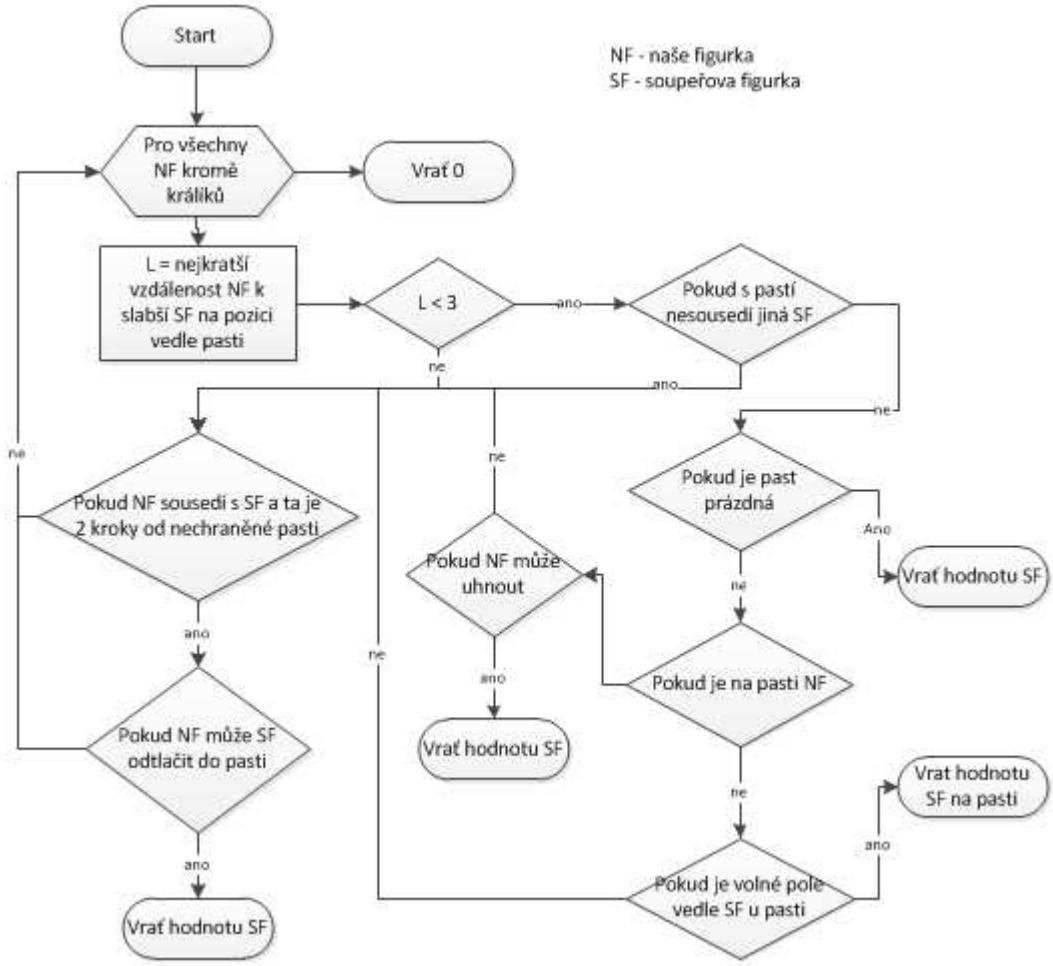


Diagram 6.1 Hledání soupeřových figurek, které je možné odebrat v jednom tahu

7 Implementace

Program je napsán v programovacím jazyce C++. Je kompatibilní s operačními systémy Windows i Linux a nevyžaduje žádné nadstandardní knihovny nebo podpůrné aplikace. Na herním serveru [16] vystupuje pod jménem bot_ugrh.

Jelikož implementace jednotlivých algoritmů je již popsána v předešlých kapitolách, je zde uvedena pouze kostra programu, která všechny výše uvedené části propojuje. Dále jsou zde popsány použité datové struktury.

7.1 Schéma programu

Program se skládá z několika samostatných částí, které jsou (kromě Engine a Structures) zapouzdřeny do svých vlastních tříd. Zde je uveden jejich popis:

- **AEI** - Třída AEI slouží pro komunikaci programu s herním serverem. Uvnitř této třídy je implementován celý AEI protokol [10]. Komunikace slouží nejprve pro nastavení potřebných proměnných hry, jako je například čas na tah nebo hloubka prohledávání. Poté je jednoduchými příkazy řízena celá hra.
- **Structures** – Zde jsou definovány základní konstanty a funkce používané v ostatní třídách. Také je zde definována struktura Options ukládající všechny proměnné definované serverem přes AEI protokol.
- **Generate** – Tato třída generuje množinu možných tahů z výchozího rozestavení podle 5. kapitoly. Je tu definováno generování podle jednotlivých vrstev, včetně množiny rozgenerovaných tahů a seznamu hashů již vygenerovaných rozestavení. Dále se zde ukládá mnoho proměnných, jejichž neustálý výpočet by běh generování zpomaloval.
- **Evalution** – Tato třída obsahuje jednotlivé hodnotící funkce definované v kapitole 4, včetně hodnotících funkcí nahrazujících prohledávání dalšího tahu z 6. kapitoly.
- **Board** – Třída Board v sobě uchovává rozestavení figurek na hrací desce pomocí datové struktury Bitboard popsané níže. Dále umožňuje uložit jeden tah, kterým lze do tohoto rozestavení přejít. Tento tah se používá při generování nových tahů, kdy je zároveň s tahem uloženo i následující rozestavení. Také obsahuje mnoho metod pro práci s tímto rozestavením nebo tahem. Jedná se o metody, které toto rozestavení analyzují, mění nebo z něj získávají jednotlivé informace.
- **Search** – třída Search provádí vyhledávání podle metod popsaných v 6. kapitole. Také obsahuje seznam všech herních rozestavení, která se již ve hře objevila a jejich počet. Tímto kontroluje, aby nedošlo k zakázanému třetímu opakování stejného rozestavení.

- **Engine** – Tento modul je centrem celého programu. Na základě zpracovaných vstupů od AEI udržuje herní rozestavení aktuální a spouští prohledávání pro nalezení nového tahu. Schéma propojení jednotlivých tříd je zobrazeno na Diagramu 7.1.

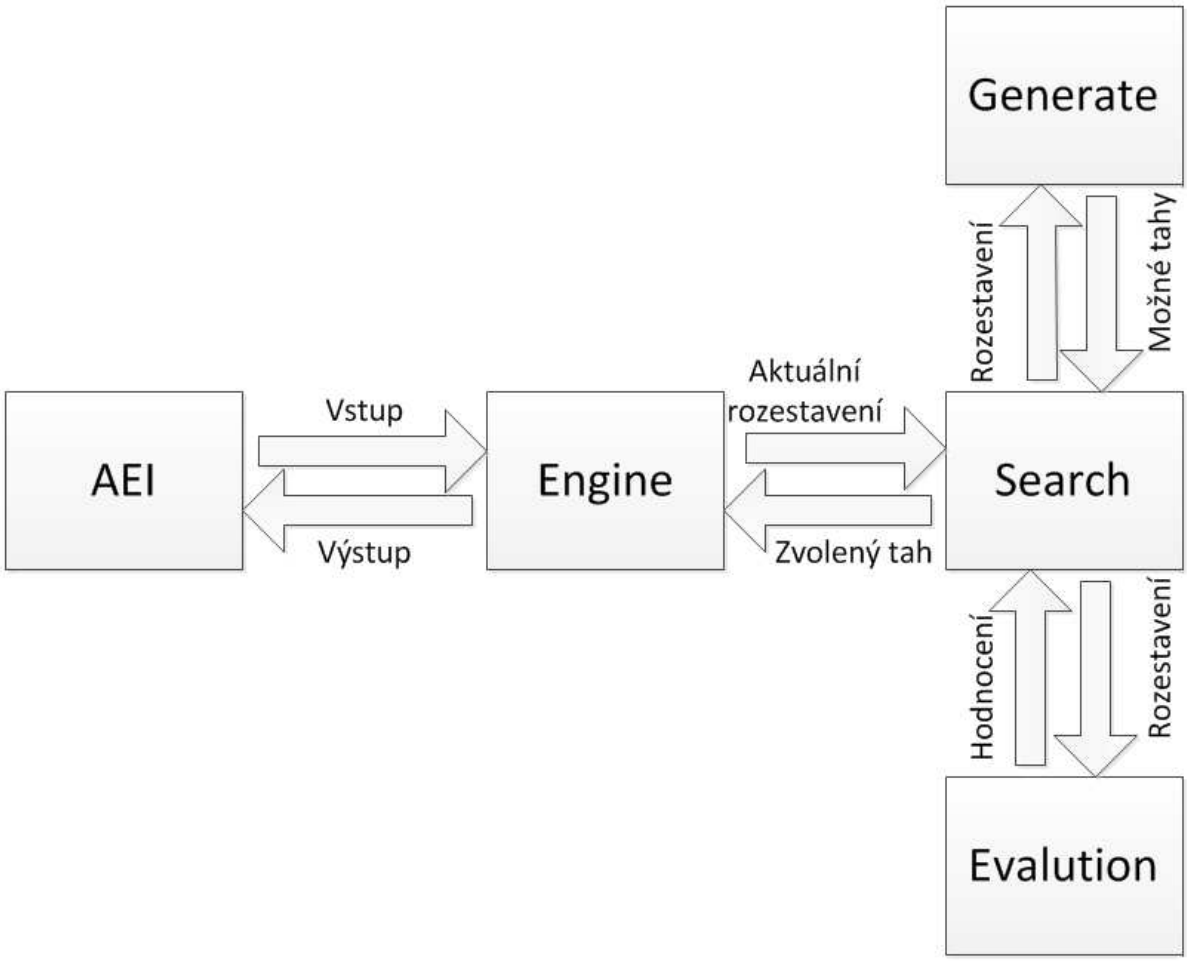
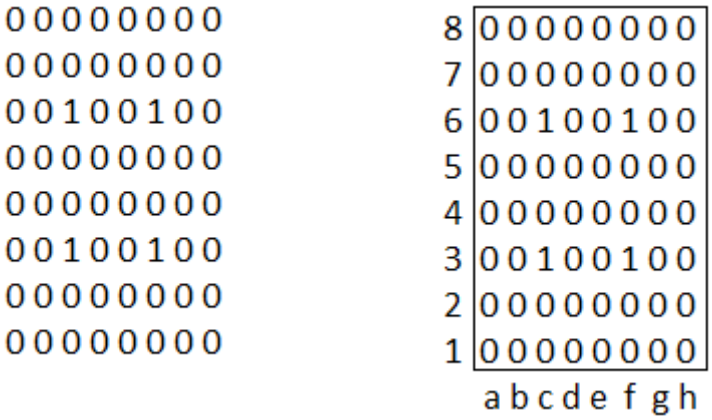


Diagram 7.1 Schéma programu

7.2 Bitboard

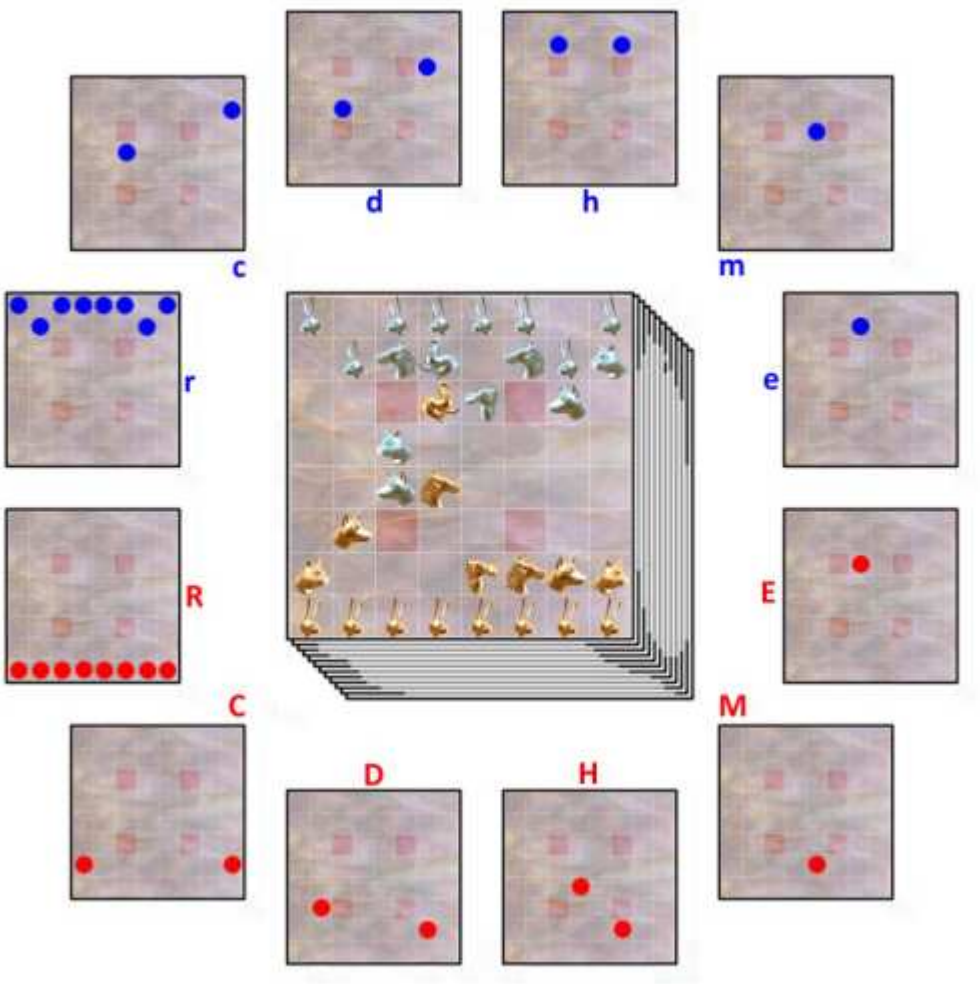
Volba správné datové struktury pro reprezentaci hrací desky a rozestavení figurek je klíčové pro rychlý běh programu. Základními požadavky jsou rychlé generování tahů a malá velikost. Proto jsem zvolil reprezentaci pomocí bitboard [16], která se používá u mnoha deskových her.

Jedná se o sadu 64-bitových integerů, takzvaných bitboard, kde každý bit reprezentuje jedno pole na hrací desce a jeho hodnota výskyt nějakého jevu. Například výskyt pastí na hrací desce by byl v bitboard zapsán 64-bitovým hexadecimálním číslem 0x0000240000240000. Tento zápis je málo srozumitelný, proto lze zobrazovat bitboard v binární podobě ve tvaru hrací desky, jak zobrazuje Obrázek 7.1, na druhém obrázku je zobrazena přesná pozice každého bitu na hrací desce. Bitboard se používá pro uložení pozic figurek a také jako maska pro různé transformace.



Obrázek 7.1 Bitboard uchovávající pozice pastí

V programu je použito 14 bitboardů. Jeden pro každou figurku a jeden pro všechny figurky jedné barvy, tedy 7 bitboardů pro každého hráče. Jak je ukázáno na Obrázku 7.2, složením jednotlivých bitboardů získáme kompletní rozestavení figurek na hrací desce.



Obrázek 7.2 Rozestavení figurek složené z bitboardů, převzato z [17]

Pro změnu nebo testování se používají bitové operace negace, AND, OR, XOR nebo bitový posun. Použití těchto operací značně zrychlí veškerou práci s daty při generování nebo hodnocení. Při těchto operacích se nemusí pracovat s každou figurkou zvlášť, ale lze aplikovat postupy na všechny figurky najednou. Například při testu, zda má zlatý hráč králíka na poslední řadě, lze jednoduše aplikovat operaci AND na bitboard obsahující pole poslední řady a na bitboard obsahující všechny zlaté králíky.

Hash by měl mít co nejmenší velikost, ale zároveň by mělo být vysoce nepravděpodobné, že dvěma různým rozestavením odpovídá stejný hash. Albert Zobrist už roku 1970 vymyslel účinnou metodu, jak tento hash vytvářet, s názvem Zobrist keys [18].

Při inicializaci je pro každou trojici barva x figurka x pozice vygenerováno náhodné 64-bitové číslo a je uloženo do trojrozměrného pole zobrist[barva][figurka][pozice]. Hash zadané pozice je vypočten provedením operace XOR na hodnoty z pole zobrist pro všechny figurky, tedy provedením níže uvedené operace pro každou figurku na hrací desce.

hash = hash XOR zobrist [figurka.barva][figurka.typ][figurka.pozice]

Pro aktualizaci hashe se nemusí tato metoda provádět celá. Stačí po provedení každého kroku jednoduše aktualizovat v hash pouze figurku, se kterou se pohybovalo.

hash = hash XOR zobrist [barva][typ][výchozí_pozice] XOR zobrist [barva][typ][koncová_pozice]

Podle Christ-Jan Coxe je pravděpodobnost, že během prohledávání budou mít dvě rozestavení stejný hash, menší než 2.2×10^{-5} [5].

8 Učení z herní historie

Tato kapitola se zabývá určením vah jednotlivých hodnotících funkcí, které jsou popsány v předchozích kapitolách. Každé hodnocení má jinou důležitost, která bude určena pomocí her z historie popsané v Kapitole 5.2.1. Výběr těchto her má přísnější kritéria, aby se program učil jen od nejlepších hráčů. Proto jsou vybrány pouze hry, kde mají oba dva soupeři rating přes 2000. Z důvodu velké časové náročnosti je použito pouze 10 prvních her, z kterých se dále použijí jen tahy vítěze odehrané po 20. kole. Na začátku hry má každý hráč svou specifickou strategii, kterou nelze naučit pouze změnou vah hodnotících funkcí.

Jelikož se jednotlivé váhy navzájem ovlivňují, zvolil jsem jako učící metodu genetické algoritmy. Hodnocení vah je velice časově náročné a kompletní testování všech možností by bylo z časového hlediska téměř nemožné. Tato metoda dokáže najít již za krátký čas rozumné řešení a zároveň může najít i řešení, které by klasické testování nenašlo kvůli lokálním maximům.

8.1 Váhový vektor

Pro uložení všech vah najednou se používá váhový vektor w. Jde o vektor s 11 prvky z oboru reálných čísel. Každý prvek značí jednu určitou váhu daného hodnocení. Pouze Hodnocení podle mobility a Hodnocení podle odebrání figurky v příštím tahu mají 2 různé váhy. Zde jsou uvedené jednotlivé váhy podle pořadí ve vektoru:

- Hodnocení podle pozice
- Hodnocení podle materiálu
- Hodnocení podle dosažení cíle
- Hodnocení podle mobility - pohyblivost
- Hodnocení podle mobility - zmrazení
- Hodnocení podle přístupu k centru
- Hodnocení podle králičího útoku
- Hodnocení podle králičí obrany
- Hodnocení podle dosažení cíle v příštím tahu
- Hodnocení podle odebrání figurky v příštím tahu – 1. hráč na tahu
- Hodnocení podle odebrání figurky v příštím tahu – 2. hráč na tahu

Na základě vlivu jednotlivých hodnocení na průběh hry jsem stanovil počáteční váhový vektor, který bude dále použit jako výchozí bod pro učení.

w_1 = (0.15, 1.8, 1.0, 1.0, 1.0, 1.0, 1.5, 2.0, 1.0, 1.1, 0.7)

8.2 Genetické algoritmy

Genetické algoritmy patří mezi tzv. evoluční algoritmy, které jsou inspirovány evolučními procesy v přírodě. Jsou to konvergující a komplexní algoritmy, které vždy vedou alespoň k suboptimálnímu řešení. Hledání globálního maxima může být časově náročné. Algoritmus využívá chromozomy, neboli jedince, kteří se skládají z jednotlivých genů a obsahují zakódované řešení daného problému. Z jedinců se skládá populace, kde se jedinci navzájem kříží a mutují a tím vytvářejí novou silnější populaci. Genetické algoritmy však nejsou primárním cílem této práce, proto zde shrnu jen jejich implementaci.

Genetický algoritmus se skládá z těchto kroků: [19]

- **Reprezentace problému** – řešení je zakódováno do chromozomu o velikosti 11 genů. Každý gen obsahuje přirozené číslo z rozsahu <0, 60>. Tento gen reprezentuje váhu v rozmezí <0.0, 3.0> s nejmenším krokem 0,05.
- **Inicializace počáteční populace** – každý jedinec je generován z výchozího vektoru w1, přičemž se každý gen náhodně změní o náhodné číslo z rozmezí <-5, 5>.
- **Ohodnocení jedinců v populaci** – toto je časově nejnáročnější etapa. Každý jedinec je ohodnocen fitness funkcí, určující jeho kvalitu podle zadaných kritérií. Toto hodnocení je specifické pro každé z učení, a proto je uvedeno v následující kapitole.
- **Selekce párů rodičovské populace** - pomocí rulety se vyberou dva jedinci a následně se zvolí ten s lepším hodnocením. Každý jedinec má v ruletě takovou šanci na výběr, jaké je jeho hodnocení v poměru se zbytkem populace.
- **Křížení** – křížení jedinců probíhá na dvou různých bodech. Tyto body jsou zvoleny náhodně a na základě křížení 2 rodičů v těchto bodech vznikají dva noví jedinci.
- **Mutace** – jednotlivé geny náhodně mutují. Podobně jako při inicializaci nové populace se každému zmutovanému genu přičte náhodná hodnota z rozsahu <-5, 5>.
- **Obnova populace** – populace se obnovuje pouze částečně, aby se zachoval správný směr vývoje.
- **Ukončení algoritmu** – vzhledem k velké časové náročnosti nebude možné určit, zda nejlepší nalezený jedinec je optimální a učení bude ukončeno v co nejdelším možném termínu.

8.3 Učení na základě hodnocení tahu

Učení probíhalo na již zmíněných hrách z herní historie. Hodnotící funkce použila postupně všechny tahy vítězného hráče, které hodnotila podle všech hodnotících funkcí s váhovým vektorem zakódovaným v chromozomu. Všechna hodnocení těchto tahů byla nadále sečtena a použita jako hodnocení daného chromozomu. Toto učení mělo mít za následek maximalizaci ohodnocení tahů, které byly zahrány. Výsledky testování ukázaly, že v tomto případě poměry mezi jednotlivymi vahami nejsou podstatné, a jak je zřejmé z výsledného váhového vektoru w_1, jednotlivé hodnoty pouze směřovaly k maximu či minimu podle výsledného součtu jednotlivých hodnocení.

W_2 = (3.0, 0, 0.35, 3.0, 2.7, 3.0, 3.0, 3.0, 0.6, 0.15, 2.9)

8.4 Učení na základě rozdílu nejlepšího tahu a zahraného tahu

Předešlé učení se soustředilo pouze na zvýšení hodnocení zahraných tahů, ale nezkoumalo, jak hodnocení tohoto tahu uspělo proti hodnocení dalších možných tahů. Sice se dosáhlo zvýšení hodnocení odehraných tahů, ale bohužel také hodnocení ostatních tahů, které stále převyšovalo. Nová fitness funkce proto zjišťuje, o kolik procent je zahraný tah horší než tah nejlépe hodnocený. Tudíž se snaží najít tahový váhový vektor, díky kterému by zahraný tah, nebo jemu podobné, získaly nejvyšší hodnocení od hodnotících funkcí. Jelikož je pro každou pozici potřeba najít nejlépe hodnocený tah, je tato fitness funkce velice časově náročná. Při počtu 5 testovaných her se výpočet hodnocení jednoho jedince pohybuje kolem 2 hodin. Proto byla velikost populace nastavena pouze na 20 jedinců. Po 360 hodinách učení byl nejlepší nalezený váhový vektor w_3 s 22% zlepšením oproti počátečnímu vektoru w_1.

W_3 = (0.15, 3.0, 1.2, 1.05, 1.0, 1.2, 1.9, 1.85, 1.05, 0.8, 0.6)

8.5 Testování

Toto testování porovnává vektor w_1, který byl stanoven mnou, s vektorem w_3, který byl vytvořen pomocí genetických algoritmů. Hledá se ten vektor, se kterým bude program dosahovat lepších herních výsledků.

Nejprve probíhá testování, které je založeno na stejném principu jako hodnocení jedinců během učení. Pro různé herní rozestavení se porovnává hodnocení tahu, který zahrál některý z dobrých hráčů

s hodnocením tahu, který by zahrál náš program. Čím je rozdíl menší, tím se tah vybraný naším programem více podobá tahu zahranému hráčem. Pro testování je použito 50 her z herní historie popsané v úvodu kapitoly. Toto testování se provádí na jiných hrách než byly použity pro učení, mají však stejné parametry. Výsledky tohoto testování jsou zobrazeny v Tabulce 8.1. Pokud hráč, jehož hry jsou použity v testu, přehlédne tah, po kterém by mohl po jakémkoliv soupeřově tahu vyhrát, je rozdíl v tazích několik tisíc procent. Z tohoto důvodu je rozdíl v jednom tahu tak vysoký. Váhový vektor však tento fakt neovlivní, tudíž to neovlivní ani fakt, že díky váhovému vektoru w_3 se vybírají tahy podobnější hráčovým.

	Počet tahů	Celkový rozdíl	Rozdíl v jednom tahu
Váhový vektor w_1	1022	40903,40%	40,02%
Váhový vektor w_3	1022	36474,00%	35,69%

Tabulka 8.1 Testování váhových vektorů na hrách z historie

Při druhém testování hrál program sám proti sobě. Program na jedné straně měl zvolen váhový vektor w_1 a na druhé straně w_3. Program s váhovým vektorem w_3 vyhrál jako zlatý i stříbrný hráč. Jelikož program neobsahuje žádnou náhodnou veličinu, další hry by byly totožné.

Nakonec byl program s oběma vektory postaven proti botům na herním serveru. Výsledky jsou zaznamenány v Tabulce 8.2.

Bot	Rating	Váhový vektor w_1	Váhový vektor w_3
bot_Badger2010P2	1701	0:2	0:2
bot_GnoBot2010P2	1734	2:0	2:0
bot_Clueless2010P2	1689	1:1	1:1
bot_Sharp2011P1	1689	1:1	1:1
bot_Briareus2011P2	1686	2:0	1:1

Tabulka 8.2 Testování váhových vektorů proti cizím botům

Váhový vektor w_3 dosáhnul lepších výsledků při prvním a druhém testování. Při třetím testování byly výsledky obou vektorů skoro rovnocenné. Toto testování dokázalo, že lze díky záznamům her změnit váhový vektor tak, aby program dosahoval lepších výsledků. Toto učení je ovšem velice časově náročné. Vzhledem k výsledkům je vektor w_3 nastaven jako výchozí vektor pro hodnotící funkce.

9 Herní strategie

Tato kapitola se zabývá změnou herní strategie v průběhu hry. Jelikož se hráčům během hry mění dílčí cíle a plány, je potřeba, aby i program změnil svou strategii podle aktuální situace na hrací desce.

První věc, kterou musí hráč rozhodnout, je volba startovní strategie. V první části kapitoly je tato problematika popsána a také jsou zde navrženy 2 varianty startovního rozestavení, které program využívá.

Zkušený hráč se nesnaží vyhrát během několika prvních tahů. Nejprve si zvolí strategii, podle které odehraje první tahy. Tehdy se snaží získat materiální nebo alespoň poziční výhodu svých figurek pro další vývoj hry. Tato část probíhá do té doby, než se úbytkem figurek na hrací desce nebo jejich pozicí neuvolní cesta, kudy by bylo možné přesunout králíka na vítěznou řadu. Pro svůj program jsem zvolil strategii Útok osamoceným slonem, která je, včetně své implementace, popsána v další části této kapitoly.

Konec kapitoly se zabývá střední hrou a koncovkami. Zde se již používá nezměněný váhový vektor určený v předešlé kapitole. Jsou zde uvedeny faktory, podle nichž program pozná, že již počáteční fáze hry skončila a začíná střední hra.

9.1 Startovní rozestavení

Určovat startovní pozice svých figurek začíná zlatý hráč, proto by měl volit neutrální symetrické rozestavení. Toto rozestavení sice nebude mít zvláštní výhodu proti stříbrnému hráči, ale stříbrný hráč také nedokáže přijít s žádným dominantním rozestavením. Navrhl jsem proto tři z nejpoužívanějších symetrických rozestavení. Všechny mají silné jednotky v centru, s možností rychlého přesunu na obě křídla.

První, které je zobrazeno na Obrázku 7.1, má označení 99of9. Koně jsou umístěni tak, aby chránili pasti z vnějších stran a zároveň králíky na stranách. Aby tyto králíky mohl soupeř vytáhnout, musí použít nejsilnější figury, a tím se oslabí.

Druhé a třetí rozestavení, zobrazené na Obrázcích 7.2 a 7.3, mají všechny své králíky schované za silnějšími jednotkami, a tudíž nehrozí jejich vytažení do středních řad. Liší se jen v pozici koní a psů, které jsou přehozené.

Všechna tato rozestavení jsou vhodná zejména pro bránící strategie. Tato rozestavení jsou použita i pro stříbrného hráče, díky jejich všestrannosti a vhodnosti pro níže popsanou strategii Útoku osamoceným slonem.

Obrázek 7.1 Startovní rozestavení A

Obrázek 7.2 Startovní rozestavení B

Obrázek 7.3 Startovní rozestavení C

9.2 Útok osamoceným slonem

Tato strategie patří k obranným strategiím a je velmi bezpečná [20]. Útok provádí sám slon, který se snaží chytit osamocenou nepřátelskou figurku a odtáhnout či odtlačit ji k pastím na své startovní straně. Tam bude odstraněna nebo alespoň zablokována i s figurkou, která ji bude bránit před odstraněním. Slon se zvláště soustředí na soupeřovy králíky, kteří se nemohou vrátit na svou stranu. Také může zaútočit na špatně hlídané pasti na nepřátelské straně a využít rychlé pomoci velblouda, který je připraven mezi centrem a svou startovní pozicí. Ostatní figurky jsou v obranném postavení a brání obě pasti na své startovní straně.

Protiútoky nepřítele, které využívají nevýhody této strategie, jsou:

- Zablokování slona mezi vlastní a nepřátelské figurky tak, aby se ani pomocí tlačení či táhnutí nemohl pohnout ze své pozice.
- Silný útok několika figurkami na jednu z pastí, obvykle na tu, která se nachází na druhé straně než náš slon.
- Použití stejné strategie a snaha o odtlačení či odtáhnutí naší slabší figurky k pastím na startovní straně protihráče.

Herní strategie je implementována pomocí změny vah jednotlivých hodnotících funkcí a změny hodnotících tabulek u Hodnocení podle pozice. Jednotlivé váhy jsou upraveny násobkem své původní hodnoty. Jednotlivé násobky těchto vah jsou uvedeny v závorkách za jménem hodnocení.

- **Hodnocení podle mobility, penalizace za nemožný směr pohybu** (0.5) – touto změnou je dosaženo toho, aby se figurky nepřesouvaly z obranného postavení kvůli lepší pohyblivosti, která zde není tolik nutná.
- **Hodnocení podle mobility, penalizace za zmrazení** (2.5) – toto násobení se snaží zabránit tomu, aby naše figurky byly zmraženy a odsunuty z obranného postavení a zároveň, aby náš slon soupeřovy figurky zmrazil a odtáhnul k pasti u startovní řady.
- **Hodnocení podle pozice** (1.7) – tímto zvýšením je dosaženo toho, že se naše figurky drží pouze v obranném postavení.
- **Hodnocení podle materiálu** (1.5) – v této strategii je hodnota figurek nadřazena hodnotě dobrému rozestavení.
- **Hodnocení podle odebrání figurky v příštím tahu** (1.5) – zvýšení této hodnoty má za následek, že slon využije každé příležitosti k odebrání soupeřovy figurky.
- **Hodnocení podle přístupu k centru** (0.3) – toto hodnocení by nutilo silné figurky přesouvat se do centra hrací desky, kde by byly kořistí nepřítele, který by je snadno zmrazil a odtáhnul, proto je toto hodnocení radikálně sníženo.

Nové tabulky pro Hodnocení podle pozice tvoří Přílohu B. Králíci mají nejvyšší hodnoty na první řadě, kde jsou kryty ostatními figurkami před odtažením. Kočky, psi a koně mají nejvyšší hodnoty na 2. a 3. řadě, kde je jejich úkolem bránit pasti. Velbloud má určenou pozici blízko centra mezi pastmi, odkud se může v případě potřeby přesunout a pomoci zmrazené figurce nebo útočícímu slonovi. Slon by se měl pohybovat v centru, kde bude vyčkávat na nepřátelské figurky, které by mohl zmrazit a přesunout.

9.3 Střední hra

V této fázi hry se strategie více zaměřuje na útok pomocí králíků a také na obranu své poslední řady. Váhový vektor se změní na původní, který byl určen v předchozí kapitole a tabulky pro Hodnocení podle pozice se nastaví do úvodního stavu, uvedeného u popisu této metody. Hra je více zaměřena na pohyblivost jednotlivých figurek, protože jsou možné konfrontace u všech 4 pastí.

Program aktivuje toto nastavení, pokud pro jednoho z hráčů bude platit jedna z následujících situací:

- V sloupci, v němž se nachází některý králík, a v přilehlých sloupcích se bude před tímto králíkem nacházet 3 a méně figurek protihráče.
- Některý z králíků může během 8 kroků dojít na cílovou řadu pouze po volných polích, kde nedojde k jeho zmrazení nebo po polích, kde stojí figurky stejné barvy.

9.4 Testování

Toto testování má za úkol zjistit, zda některá startovní rozestavení nejsou pro jednu ze dvou herních strategií dominantní. Při testování se program postavil sám proti sobě s jiným typem počáteční strategie nebo herním rozestavením. Pro každou dvojici nastavení proběhly 2 hry tak, aby každé nastavení bylo na zlaté i na stříbrné straně. Jelikož program neobsahuje žádné rozhodnutí založené na náhodě, další hry byly totožné s prvními dvěma. Výsledky jsou zapsány v Tabulce 9.1.

Strategie a rozestavení		Bez počáteční strategie			Útok osamoceným slonem			Celkem
		A	B	C	A	B	C	
Bez počáteční strategie	A	XXX	0:2	2:0	0:2	1:1	1:1	4:6
	B	2:0	XXX	1:1	1:1	1:1	1:1	6:4
	C	0:2	1:1	XXX	2:0	1:1	0:2	4:6
Útok osamoceným slonem	A	2:0	1:1	0:2	XXX	2:0	1:1	6:4
	B	1:1	1:1	1:1	0:2	XXX	2:0	5:5
	C	1:1	1:1	2:0	1:1	0:2	XXX	5:5

Tabulka 9.1 Testování herních strategií a počátečních rozestavení

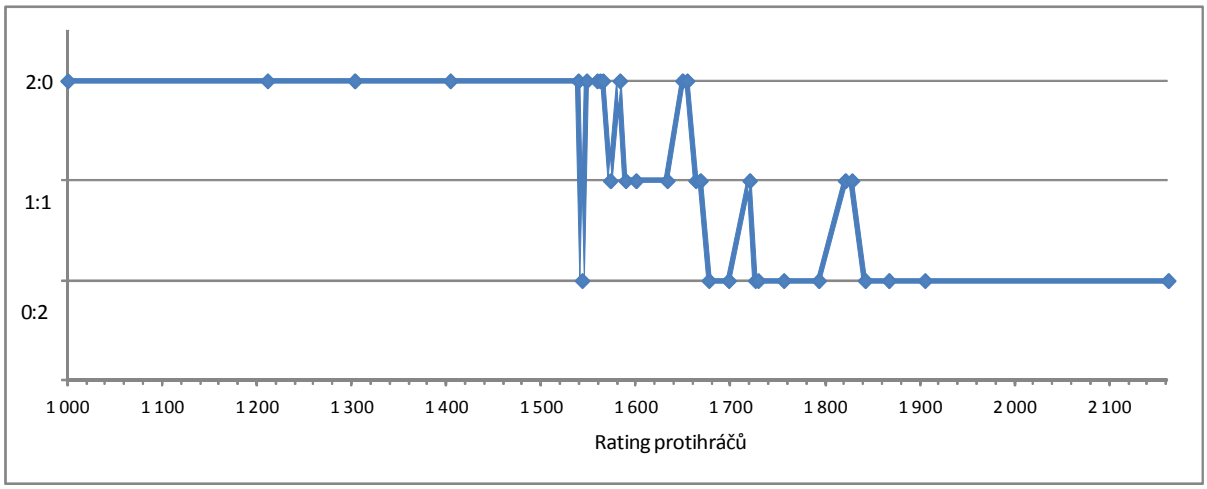
Z výsledků vyplývá, že jsou obě strategie téměř vyrovnané. O 2 body vítězí Útok osamoceným slonem. Ani u rozestavení nejsou zásadní bodové rozdíly. Rozestavení B bez počáteční strategie jako jediné ani jednou neprohrálo. Útok slonem dosahuje nejlepších výsledků s rozestavením A.

10 Testování

Každá část byla v průběhu vývoje důkladně otestována. Toto testování sloužilo jako potvrzení správného směru vývoje, nebo k výběru těch nejlepších hodnot. Testy jsou uvedeny přímo v jednotlivých kapitolách a jejich výsledky byly použity k sestavení současné podoby programu. V této kapitole je testován již hotový program a výsledky mají určit, jak si tento program stojí proti ostatním botům a živým hráčům.

10.1 Rating

Herní server nabízí 134 botů (respektive jejich verzí), které jsou on-line spustitelní pro hru. Většina botů je vyvíjena již několik let a je zde umístěno několik různých verzí od stejného autora. Jsou zde verze s různým časem na hru a verze podle stavu vývoje. Každá z těchto verzí má ještě několik podverzí podle roku, kdy byla vydána. Z tohoto počtu je jich 82 s časovou variantou 2 minuty na tah, na kterou jsem se zaměřil. Jejich rating je v rozmezí 1000 až 2162. Graf 10.1 zobrazuje výsledky her s ostatními boty. Z grafu lze odhadnout pravděpodobný rating v rozmezí 1600 až 1650. Podrobné výsledky a záznamy jednotlivých her se nachází zde [21].



Graf 10.1 Výsledky programu proti ostatním botům

10.2 Výsledky na herním serveru

Během vývoje a testování odehrál bot_ugrh na herním serveru 397 partií, z nichž 171 vyhrál a 226 prohrál. Vývoj ratingu popisuje Graf 10.2. Jak je zřejmé, část proher je z první doby vývoje, kdy bylo potřeba doladit komunikační a systémové chyby, a kdy server ukončil hru předčasně kvůli chybě.

K datu 21. května 2012 má aktuální rating 1612 a ze všech registrovaných botů je v průběžném pořadí 28 z 109. Bot dokázal porazit 54 z 82 botů dostupných na herním serveru, které mají časovou kontrolu nejméně minutu na jeden tah. Nejsilnější bot, kterého program dokázal porazit, byl bot_Clueless2009P2 s ratingem 1828.

Graf 10.2 Vývoj ratingu pro hráče bot_ugrh, graf byl vygenerován na herním serveru [21]

Testování s živými hráči nebylo možné provést systematicky, protože je na vůli každého hráče, zda s botem bude hrát nebo ne. Celkem s ním hrálo 16 hráčů v 42 hrách, přičemž jich 18 dokázal vyhrát. Nejsilnější hráč, kterého dokázal porazit, byl hráč OffbeatJM s ratingem 1776 . Podrobnější informace jsou dostupné na statistikách herního serveru [21].

11 Závěr

Cílem této práce bylo prozkoumat možnosti využití umělé inteligence při hraní deskových her. Zaměřil jsem se na hru Arimaa, která byla vytvořena speciálně pro tento účel. Navrhl jsem program schopný hrát tuto hru proti ostatním botům i živým hráčům. Program se skládá ze 3 klíčových částí: generování tahů, prohledávání a hodnotící funkce. Velkou výhodu vidím v použití bitboardu pro reprezentaci hrací desky, který výrazně urychlil generování tahů i hodnotící funkce. Podařilo se mi navrhnout několik metod, které tento program zrychlí a dopomůžou k lepším herním výsledkům. Testování ukázalo, že se podle ratingu program pohybuje ve středu žebříčku všech programů vytvořených pro tuto hru. Jelikož je většina botů vyvíjena již řadu let, jsem s tímto umístěním velmi spokojen.

Toto téma je velice rozsáhlé a naskýtá se mnoho dalších cest, kterými by se dalo ve vývoji pokračovat; například nalezení sofistikovanějších faktorů pro zmenšení množiny vygenerovaných tahů a rychlé hodnotící funkce pro jejich řazení. V hodnocení by se měl více hodnotit dlouhodobější vývoj hry. Měly by se stanovit cíle, které by měly jednotlivé figurky plnit, například přesun figurek na klíčová místa během volných kroků v několika tazích. Na řešení těchto nápadů jsou ale potřeba velké hráčské zkušenosti, které mi prozatím bohužel chybí.

Literatura

[1] SYED, Omar. *Arimaa Official Homepage* [online]. 1999 [cit. 2012-01-05]. Dostupné z: http://arimaa.com/arimaa/

[2] *Arimaa Tutorial (Flash version)* [online]. [cit. 2012-01-05]. Dostupné z: http://arimaa.com/arimaa/learn/flash/jc/new/

[3] IBM Research: Deep Blue. [online]. 2001 [cit. 2012-01-05] Dostupné z: http://www.research.ibm.com/deepblue/meet/html/d.3.shtml

[4] ALLIS, Louis Victor. *Searching for solutions in games and artificial intelligence*. Maastricht: Rijksuniversiteit Limburg, 1994. ISBN 90-900-7488-0.

[5] COX, Christ-Jan. *Analysis and Implementation of the gane Arimaa*. Hoensbroek, březen 2006. Master Thesis. Universiteit Maastricht.

[6] WU, David Jian. *Move Ranking and Evaluation in the Game of Arimaa*. Massachusetts, 31. březen, 2011. Thesis. Harvard College.

[7] HASKIN, Brian. Arimaa Game Graphs. *Janzert's Arimaa Projects* [online]. 2006 [cit. 2012-01- 05]. Dostupné z: http://arimaa.janzert.com/gamegraphs

[8] Arimaa. In: *Wikipedia*: *the free encyclopedia* [online], 2001- [cit. 2012-01-09]. Dostupné z: http://en.wikipedia.org/wiki/Arimaa

[9] KOZELEK, Tomáš. *Methods of MCTS and the game Arimaa*. Praha, 5. srpna 2009. Master's thesis. Charles University in Prague Faculty of Mathematics and Physics. Vedoucí práce RNDr. Jan Hric.

[10] HASKIN, Brian. AEI - Arimaa Engine Interface. *Janzert's Arimaa Projects* [online]. 2009 [cit. 2012-01-09]. Dostupné z: http://arimaa.janzert.com/aei/

[11] FOTLAND, David. "Building a World Champion Arimaa Program". In: JAAP VAN DE HERIK H., BJÖRNSSON, Yngvi, NETANYAHU, Nathan S. (eds.). *Computers and Games (4th International Conference)*. Israel, červenec 5-7, 2004.

[12] ZHONG, Haizhi. *Building a Strong Arimaa-playing Program*. Edmonton, Alberta, 2005. Master Thesis. University of Alberta.

[13] Minimax (algoritmus). In: *Wikipedia*: *the free encyclopedia* [online], 2001- [cit. 2012-01-05]. Dostupné z: http://cs.wikipedia.org/wiki/Minimax_%28algoritmus%29

[14] VUT FIT. Předmět IZU, *Metody hraní her (minimax, alfa-beta, hry s náhodou).* [online] Dostupné z: https://www.fit.vutbr.cz/study/courses/IZU/private/1011izu_5.pdf

[15] SYED, Omar. *Arimaa Gameroom* [online]. [cit. 2012-05-17]. Dostupné z: http://arimaa.com/arimaa/gameroom/

[16] CARLINI, Stefano. *Arimaa: from rules to bitboard analysys*. Knowledge Representation Thesis, 2008. University of Modena and Reggio Emilia.

[17] CARLINI, Stefano. *Arimaa, a New Challenge for Artificial Intelligence*. Apr 2010. Thesis. University of Modena and Reggio Emilia, Italy.

[18] ZOBRIST, L. Albert. *A new hashing method with application for game playing*. April 1970. Technical report. University of Wisconsin.

[19] SATOLA, Richard. *Genetické algoritmy* [online]. [cit. 2011-05-05]. Dostupné z WWW: https://www.fit.vutbr.cz/study/courses/EVO/private/VYUKA/prednasky_2011/P1/GA_sato.pps

[20] Arimaa/Lone Elephant Attacks. In: *Wikibooks*: *Open books for an open world* [online]. [cit. 2012-05-17]. Dostupné z: http://en.wikibooks.org/wiki/Arimaa/Lone_Elephant_Attacks

[21] Bot_ugrh: Player #7424. *Arimaa Gameroom* [online]. [cit. 2012-05-17]. Dostupné z: http://arimaa.com/arimaa/gameroom/playerpage.cgi?id=17716

Seznam příloh

Příloha A

Tabulky pro Hodnocení podle pozice figurek

0	0	1	2	2	1	0	0
0	0	2	3	3	2	0	0
1	2	-3	4	4	-3	2	1
2	3	4	5	5	4	3	2
2	2	4	5	5	4	3	2
1	2	-3	4	4	-3	2	1
0	0	2	3	3	2	0	0
0	0	1	2	2	1	0	0

Tabulka A.1 Slon

0	0	1	2	2	1	0	0
0	0	1	3	3	2	0	0
0	2	-3	4	4	-3	2	0
2	3	4	5	5	4	3	2
1	2	4	4	4	4	3	2
0	1	-3	3	3	-3	1	0
0	0	2	3	3	2	0	0
0	0	1	2	2	1	0	0

Tabulka A.2 Velbloud

0	0	1	2	2	1	0	0
0	0	1	3	3	2	0	0
1	2	-3	4	4	-3	2	1
2	3	4	5	5	4	3	2
1	2	4	4	4	4	3	2
1	2	-3	3	3	-3	2	1
0	0	2	3	3	2	0	0
0	0	1	2	2	1	0	0

Tabulka A.3 Kůň

1	2	3	2	2	3	2	1
2	3	4	3	3	4	3	2
3	4	-2	4	4	-2	4	3
1	1	2	1	1	2	1	1
0	0	1	0	0	1	0	0
0	0	-2	0	0	-2	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Tabulka A.4 Pes a kočka

2	2	2	2	2	2	2	2
2	2	3	1	1	3	2	2
2	2	-2	0	0	-2	2	2
2	1	0	0	0	0	1	2
2	1	0	0	0	0	1	2
2	2	-4	0	0	-4	2	2
4	4	4	2	2	4	4	4
8	8	8	8	8	8	8	8

Tabulka A.5 Králík

Příloha B

Tabulky pro Hodnocení podle pozice figurek při Útoku osamoceným slonem

0	0	1	2	2	1	0	0
0	0	2	2	2	2	0	0
1	2	-3	2	2	-3	2	1
2	2	3	3	3	3	2	2
2	4	4	5	5	4	4	2
1	4	-3	5	5	-3	4	1
0	0	2	3	3	2	0	0
0	0	1	2	2	1	0	0

Tabulka B.1 Slon

0	0	1	2	2	1	0	0
0	0	2	4	4	2	0	0
2	4	-1	5	5	-1	4	2
1	3	4	4	4	4	3	1
1	2	2	2	2	2	2	1
0	1	-4	1	1	-4	1	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Tabulka B.2 Velbloud

0	0	1	2	2	1	0	0
0	2	3	4	4	3	2	0
2	4	-1	5	5	-1	4	2
1	2	2	2	2	2	2	1
1	1	1	1	1	1	1	1
0	1	-3	1	1	-3	1	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Tabulka B.3 Kůň

1	2	3	2	2	3	2	1
2	3	4	3	3	4	3	2
3	4	-2	4	4	-2	4	3
1	1	2	1	1	2	1	1
0	0	1	0	0	1	0	0
0	0	-4	0	0	-4	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Tabulka B.4 Pes a kočka

2	2	2	2	2	2	2	2
2	1	2	1	1	2	1	2
0	1	-2	0	0	-2	1	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	-4	0	0	-4	0	0
0	0	0	0	0	0	0	0
8	8	8	8	8	8	8	8

Tabulka B.5 Králík