Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE

Tomáš Starý

Prostředí pro hru Atari Go

Matematicko-fyzikální fakulta

Vedoucí bakalářské práce: *RNDr. Jan Hric, Katedra teoretické informatiky a matematické logiky*

Studijní program: *Informatika, Programování*

2009

Poděkování

Tímto bych chtěl vyjádřit své poděkování panu RNDr. Janu Hricovi za jeho pomoc a trpělivost při vedení této bakalářské práce.

Prohlašuji, že jsem svou bakalářskou práci napsal samostatně a výhradně s použitím citovaných pramenů. Souhlasím se zapůjčováním práce.

V Praze
dne Jméno

Abstrakt

Název práce: *Prostředí pro hru Atari Go*
Autor: *Tomáš Starý*
Katedra (ústav): *Matematicko-fyzikální fakulta*
Vedoucí bakalářské práce: *RNDr. Jan Hric*
e-mail vedoucího: *jan.hric@mff.cuni.cz*
Abstrakt: Hra Go a také její jednodušší varianta hra Atari Go znamenají v dnešní době pro informatiku a její podoblast umělou inteligenci velkou výzvu. Hra Go představuje zatím nevyřešenou otázku při programování umělé inteligence, která by v této hře byla silnější než člověk.
Tato bakálářská práce si klade za cíl vytvořit počítačový prostředí pro hru Atari Go na malých deskách do rozměru 9 x 9 průsečíků. Cílem tohoto programu pak bude představit uživatelům hru Atari Go a naučit je hrát tuto. Dalším cílem, který program sleduje, je motivovat uživatele k přechodu od hry Atari Go k její složitější variantě Go. Cílovou skupinou uživatelů programu jsou ti uživatelé, kteří tuto hru zatím nehráli, ale mají chuť se naučit Atari Go.
Stejně dobře program poslouží uživatelům, kteří už Atari Go někdy hráli a chtěli by se procvičit ve hře nebo zjistit více informací o hře Go.
Klíčová slova: *Atari Go, Ponnuki-Go, Go, Počítačová hra, Umělá inteligence*

Abstract

Title: *Environemt for Atari Go Game*
Author: *Tomas Stary*
Department: *Mathematical-Physical Faculty*
Supervisor: *RNDr. Jan Hric*
Supervisor's e-mail address: *jan.hric@mff.cuni.cz*
Abstract: These days Go game and its simpler variant game Atari Go means a major challenge for Computer Science and Artificial Intelligence (AI). The Go game represents yet unresolved question in the programming of Artificial Intelligence that would have been in the game stronger than man.
This Bachelor Thesis aims to create a computer environment to play Atari Go on small boards in dimension to 9 x 9 intersections. The objective of this program is to introduce users to Atari Go game and teach them how to play this game. Another aim of this program is to motivate users to switch from Atari Go game to a more difficult variant Go game. The target group of users are those users who have not played this game, but they want to learn the joy of Atari Go.
Program will serve equally well to users who have tried Atari Go game and would like to get practice in the game or find out more information about the Go game.
Keywords: *Atari Go,Ponnuki-Go, Go, Computer Game, Artificial Intelligence*

Obsah

I. ANOTACE ... 5

1. PŘEDSTAVENÍ HRY GO ... 6

1.1. HRÁČI, HRACÍ DESKA A KAMENY ... 6
1.2. KAMENY A OBKLIČOVÁNÍ KAMENŮ ... 6
1.3. ZJEDNODUŠENÁ VERZE GO = ATARI GO ... 7
1.4. ZAKÁZANÉ TAHY V ATARI GO ... 7
1.5. SEBEVRAŽDA ... 7
1.6. POVOLENÁ SEBEVRAŽDA ... 8
1.7. ZÁKLADNÍ TECHNIKY HRY V ATARI GO ... 8
1.8. ŽIVÉ SKUPINY ... 8
1.9. DVOJITÉ ATARI ... 9
1.10. SCHODY ... 9
1.11. HANDICAP – METODA PRO VYVÁŽENÍ SÍLY HRÁČŮ ... 9

2. PŘEDSTAVENÍ PROGRAMU *ATARIGO* ... 10

2.1. SEZNÁMENÍ UŽIVATELE S PROGRAMEM *ATARIGO* ... 10
2.2. SEHRÁNÍ PARTIE V PROGRAMU *ATARIGO* ... 11
2.3. NASTAVOVÁNÍ VLASTNOSTÍ HRY ... 13
2.4. OKNO ÚDAJE O TÉTO HŘE ... 13
2.5. OKNO NASTAVENÍ HRÁČŮ ... 14
2.6. OKNO NASTAVENÍ HRACÍ DESKY ... 14
2.7. OKNO PŘEVÍJENÍ HRY ... 15
2.8. OKNO DESKA JE PŘIPRAVENA KE HŘE ... 15
2.9. OKNO NASTAVENÍ STATICKÉ FUNKCE ... 15
2.10. OKNO PŘEHLED TAHŮ PARTIE ... 16

3. PROGRAMÁTORSKÁ DOKUMENTACE PROGRAMU ... 16

3.1. OBECNÝ POPIS VLASTNOSTÍ PROGRAMU *ATARIGO* ... 17
3.2. DATOVÉ STRUKTURY POUŽITÉ PRO REPREZENTACI HRACÍ DESKY ... 17
3.3. HRACÍ DESKA A VYKRESLOVÁNÍ KAMENŮ NA DESKU ... 18

4. PROGRAMOVÁNÍ AUTOMATICKÉHO POČÍTAČOVÉHO HRÁČE A UMĚLÁ INTELIGENCE (AI) ... 19

4.1. PROHLEDÁVÁNÍ STAVOVÉHO PROSTORU ANEB PROČ PROHLEDÁVAT ... 19
4.2. PŘEHLED PROHLEDÁVACÍCH TECHNIK ... 21
4.3. ALGORITMUS MINIMAX ... 21
4.4. PSEUDOKÓD PRO ALGORITMUS MINIMAX ... 21
4.5. ALGORITMUS ALPHABETA ($\alpha\beta$) PROHLEDÁVÁNÍ ... 22
4.6. PSEUDOKÓD PRO ALGORITMUS ALPHABETA ($\alpha\beta$) ... 22
4.7. OHODNOCOVACÍ FUNKCE ... 23
4.8. KONSTANTY OHODNOCOVACÍ FUNKCE ... 25

5. ALGORITMUS HRY POČÍTAČOVÉHO HRÁČE V PROGRAMU *ATARIGO* ... 26

6. SHRNUTÍ BAKALÁŘSKÉ PRÁCE ... 28

7. SEZNAM LITERATURY A ZDROJŮ ... 29

8. PŘÍLOHY ... 30

8.1. PROGRAM *ATARIGO* VČETNĚ ZDROJOVÝCH KÓDŮ PROGRAMU NA CD ... 30
8.2. SLOVNÍČEK POJMŮ VE HŘE GO ... 30
8.3. UKÁZKA PROSTŘEDÍ PROGRAMU ... 33

I. Anotace

Hra Go a také její jednodušší varianta hra Atari Go znamenají pro informatiku a její oblast umělou inteligenci v dnešní době velkou výzvu. I v době plné výkonných počítačů představují doposud nevyřešenou otázku naprogramování umělé inteligence, která by byla v těchto hrách silnější než člověk. Je až s podivem, že narozdíl od hry šachy, zůstává hra Go umělou inteligencí doposud nepřemožena. Přestože je hra Go velmi stará asijská hra, v západním světě zatím nemá tak širokou hráčskou základnu jakou má například již zmíněná hra šachy. Hlavním důvodem obtíží umělé inteligence při hraní partie Go je to, že na velkých deskách nelze prozkoumat všechny možné tahy kvůli jejich obrovskému množství. Umělé inteligenci zatím také chybí kvalitní a rychlá ohodnocovací funkce. Člověk narozdíl od počítače tak zde má výhodu nadhledu na desku a logického myšlení. Tím člověk dokáže hned vyloučit tahy, které počítač musí vždy prozkoumat.

Situace okolo hry Atari Go je pro umělou inteligenci více příznivější. Jsou známy řešení na malých deskách do velikosti 5x5 průsečíků (včetně) a hledají se řešení pro větší desky. Hra Atari Go pak slouží velmi dobře jako testovací prostředí pro algoritmy, které by se daly uplatnit ve hře Go. Hra Atari Go na malých deskách tak představuje zajímavé výzkumné téma pro bakalářskou práci.

Tato bakálářská práce si klade za cíl vytvořit grafické počítačové prostředí pro hru Atari Go na malých deskách do rozměru 9 x 9 průsečíků (včetně). Cílem programu je představit uživatelům hru Atari Go a naučit je hrát tuto. Dalším cílem, který program sleduje, je motivovat uživatele k přechodu od hry Atari Go k její složitější variantě Go.

Celé prostředí pro hru Atari Go je v češtině. V prostředí může uživatel sehrát partii Atari Go proti jinému uživateli, nebo proti počítači. Prostředí umožňuje také sledovat hru počítače proti počítači. Každou rozehranou partii může uživatel uložit na disk. Později tuto uloženou partii může načíst a pokračovat ve hře. Pro výukové účely je v prostředí podporuja funkce vrácení tahu zpět. Hráč se tak během partie může vrátit dokonce o libovolně kroků zpět (maximálně však do výchozího stavu na hrací desce). Jako protiváhu k vrácení tahu zpět prostředí podporuje také funkci posunutí o krok vpřed. Pro lepší přehled o hře si také uživatel může partii kdykoli přehrát od začátku až do posledního položeného kamenu na desku. Inteligenci zabudovaného počítačového hráče v programu může uživatel ovlivňovat nastavením 5 různých konstant. V neposlední řadě pak v prostředí uživatel nalezne odkazy na internetové stránky, kde si může přečíst vše o hrách Atari Go a Go.

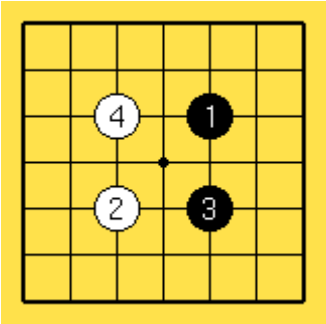
Cílovou skupinou uživatelů programu jsou ti uživatelé, kteří hru Atari Go zatím nehráli, ale mají chuť se ji naučit a zahrát si. Stejně dobře program poslouží uživatelům, kteří už Atari Go někdy hráli a chtěli by se v této hře procvičit nebo zjistit informace o hře Go. Naopak, pro pokročilé hráče Go tento program nepředstavuje relevantního soupeře ve hře.

Jednotlivé části této bakalářské práce se zaměří na vysvětlení hry Atari Go, popis pravidel této hry a její vztah ke hře Go. Hlavní část práce pak bude věnována popisu programu *Atarigo* z pohledu uživatelského a z pohledu programátorského.

1. Představení hry Go

1.1. Hráči, hrací deska a kameny

Hra Go je desková stolní hra pro dva hráče. První hráč má černé kameny a druhý má bílé kameny. Každá partie se hraje na desce (v originálním názvosloví gobanu), což je čtvercová síť o velikosti 19 x 19 průsečíku. Na klasické desce se tedy nachází 361 volných průsečíků, kam je možné položit hráčské kameny. Bežně se také hraje na deskách velikosti 13 x 13 a 9 x 9. Partie na těchto menších deskách je snazší a trvá kratší dobu. Menší desky jsou díky tomu vhodné hlavně pro začátečníky ve hře Go.



Obrázek 1:

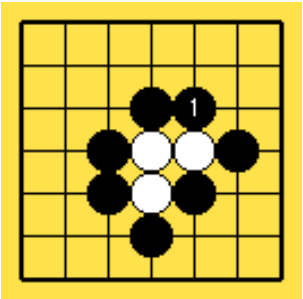
Hrací deska s položenými kameny.

Každá partie obyčejně začíná na prázdné desce. Je však také možné začínat na desce, kde už jsou položeny kameny jednoho z hráčů, standardně černého. Taková situace se nazývá handicap a používá se na vyrovnání síly hráčů. Handicap proto, že se jedná o znevýhodnění hráče druhého hráče, který na desce zatím položeny kameny nemá. Velikost handicapu se měří počtem soupeřových kamenů položených na desce. Více o handicapech se dozvíte v odstavci 1.11. Handicap – Metoda pro vyvážení síly hráčů.

Hrací deska a kameny je vše, co potřebujete ke hraní hry Go. Během hry se hráči střídají v pokládání kamenů na volné průsečíky desky. Je zvykem, že hru začíná hráč s černými kameny. S položenými kameny na desce se nehýbá.

1.2. Kameny a obkličování kamenů

Jedním z hlavních cílů hry Go je obklíčit co největší počet kamenů soupeře. V souvislosti s tímto cílem se definují pojmy *svobody kamene*, příp. *svobody skupiny kamenů*.



Obrázek 2:

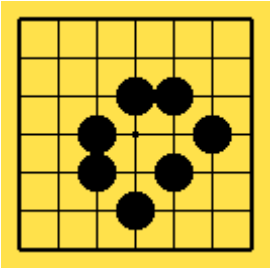
Skupina bílých kamenů obklíčená černým hráčem před vyjmutím z desky.

Pojem svoboda kamene je definován jako počet volných průsečíků, které přímo sousedí s kamenem. Pro názornost uvedu pár příkladů svobod: pokud hráč položí svůj kámen doprostřed prázdné hrací desky, má tento kámen 4 svobody, při položení kamene na stěnu – za předpokladu, že kámen s žadným jiným kamenem přímo nesousedí – má tento kámen svobody jen 3. Při položení kamene do rohu hrací desky – pokud s žádným jiným kamenem přímo nesousedí - má kámen svobody jen 2.

Kameny stejné barvy nazývame skupinou, pokud spolu kameny vzájemně sousedí a počet takto vzájemně propojených kamenů je alespoň 2. Abychom byli technicky přesní, budeme i 1 osamělý kámen také označovat jako skupinu. Pojem skupina kamenů pak bude označovat kameny jedné barvy vzájemně propojené uvnitř této skupiny a oddělené od ostatních skupin.

Pojem svoboda skupiny je definován jako počet volných průsečíků desky, s kterými přímo sousedí daná skupina.

Skupinu kamenů jedné barvy nazýváme obklíčenou, je-li obklopena kameny soupeře. Obklíčená skupina kamenů je tedy taková skupina, jež má 0 svobod a je obklopena kameny soupeřovy barvy. Takto obklíčené kameny (někdy také nazývané zajatci) se ve hře vyberou z hrací desky.



Obrázek 3:

Černý hráč vyhrál hru Atari Go. Bílé kameny vyjmuty z hrací desky.

1.3. Zjednodušená verze Go = Atari Go

Pravidla hry Atari Go jsou velice snadná. Ten hráč, který první zajme skupinu soupeřových kamenů, vyhrál partii. Jeho počet bodů za výhru je roven počtu jeho zajatců = obklíčených kamenů.

Hru Atari Go začíná obvykle hráč s černými kameny. Klasická pravidla hry Atari Go definují výhru, pokud hráč svým tahem zajme alespoň jeden soupeřův kámen.

Složitější, nicméně také zajímavější, je potom ta varianta hry, kdy k výhře je nutné zajmout alespoň 2 nebo obecně $n > 1$ kamenů.

1.4. Zakázané tahy v Atari Go

Pravidla hry Atari Go, jak jste již možná sami zjistili, jsou velice jednoduchá: dva hráči se střídají na tahu a snaží se obklíčit soupeřovy skupiny. Kameny obklíčené skupiny se vyberou z desky. Kameny je přitom možné položit na volné průsečíky po celé desce prakticky bez omezení. S položenými kameny je zakázáno hýbat.

Přesto i v Atari Go se během partie mohou vyskytnout průsečíky, na které hráč nesmí položit kamen. Těchto průsečíků je na desce pouze poskrovnu a pravidla popisující zakázané tahy v Atari Go jsou právě tato dvě:

1.5. Sebevražda

Zcela logickým pravidlem je pravidlo zakazující sebevraždu vlastní skupiny. Zní takto: Hráč nesmí zahrát na průsečík, na němž by jeho kámen neměl ani jednu svobodu a byl obklopen kameny soupeře, příp. pokud by hráč tímto tahem sebral poslední svobodu nějaké své skupině. Tato situace, jak čtenář jistě dovede nahlédnout, by vedla k okamžitému vítězství soupeře, proto se tento tah nazývá sebevražda.

Hráč tedy nesmí zahrát na sebevraždu a musí zahrát na jiný průsečík, na který je tah dovolený. Zároveň se v partii může stát, že všechny zbývající volné průsečíky na desce jsou tahem na sebevraždu, a tedy hráč již nemá tah, který by zahrál. V takovém případě hra končí vítězstvím soupeře. Jedná se o výhru vyčerpáním povolených tahů hráče.

1.6. Povolená sebevražda

Pro někoho možná trochu matoucím po předešlém pravidle o zákazu sebevraždy bude pravidlo o povolené sebevraždě. Jeho název je možná trochu zavádějící, nicméně pravidlo je naprosto logické a vystihuje takovouto situaci na hrací desce: hráč může zahrát na průsečík, kde by jeho kámen měl 0 svobod pouze tehdy, pokud tímto tahem obklíčí a sebere nějaké soupeřovy kameny. Tím si svobody hned zase přidělá, a tedy nepůjde o tah na sebevraždu. Jak jste jistě již sami doznali, v Atari Go tímto tahem navíc i skončí hra.

Pro úplnost je vhodné ještě zmínit, že hra Go má více než tyto dvě výše jmenovaná pravidla.

1.7. Základní techniky hry v Atari Go

Tato část již nepatří úplně do problematiky hry Atari Go a řadí se spíše do tématiky hry Go. Mezi hráči Go se tyto techniky nazývají původním japonským slovem *tesudži*. V této sekci představím čtenáři několik technik, které povedou k pokročilejšímu hraní hry Go a také hry Atari Go.

Protože se celá práce věnuje hře Atari Go, uvedu zde pouze malou skupinku takových technik, které lze uplatnit pro chytřejší hraní hry Atari Go.

1.8. Živé skupiny

Čtenáře jistě zaujal název této kapitoly. Pojem živá skupina se ve hrách Go i Atari Go používá pro takovou skupinu, kterou nelze v žádném případě zajmout.

Vytvořit skupinu, kterou nelze zajmout je proto hráčské umění, které stojí za to se naučit. Technice vytvoření živé skupiny se říká Vytváření dvou očí.

Technika spočívá v tom, že hráč si uměle vytvoří skupinu svých kamenů s dvěma volnými průsečíky rozmístěnými uprostřed skupiny. Pokud si počíná chytře, vyrobí si skupinu takovou, že ony 2 volné průsečíky nejsou hned vedle sebe, ale jakoby rozdělené, tzn. ob několik svých kamenů obklopené dalšími svými kameny. Jeho soupeř, aby tuto skupinu zajal, by musel zahrát na ony dva volné průsečíky (kde je v sebevraždě) současně. Tomu se říká, že skupina má dvě oči. Aby skupina byla živá, musí mít dvě oči.

Pokud má skupina jen jedno oko, pak by soupeř potřeboval nejprve oblehnout tuto skupinu svými kameny a teprve poté může zahrát na volný průsečík uvnitř (bude se jednat o povolenou sebevraždu), což je ale pro soupeře náročné na počet tahů.

V Atari Go se tato technika hodí například v situaci, kdy si obsadíte rohy hrací desky živými skupinami a postupně utahujete svého soupeře na středu hrací desky. Tím soupeři zmenšujete prostor povolených tahů a chytrou hrou můžete docílit toho, že vyhrajete na vyčerpání soupeřových povolených tahů.

1.9. Dvojité atari

Další šikovnou technikou, hojně používanou ve světě Go, je technika s názvem dvojité atari.

Nejprve je potřeba čtenáři vysvětlit, co znamená pojem atari. Jako atari je označován tah, který soupeřově skupině kamenů zanechává poslední svobodu. Tedy jinými slovy, soupeř v atari musí svou ohroženou skupinu kamenů bezprostředně zachránit. Pokud soupeř skupinu nezachrání, jeho ohroženou skupinu dalším tahem zajmete a kameny této skupiny vyberete z desky (a pokud hrajete Atari Go, pak také vyhrajete).

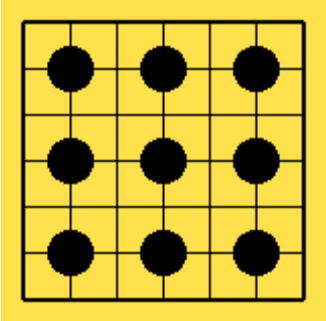
Dvojité atari je takový tah, díký kterému bezprostředně ohrozíte více soupeřových skupin najednou a každou z nich postavíte do pozice atari. Váš soupeř potom nemá na výběr a jednu ze skupin Vám bude muset obětovat. V Atari Go jste tak i vyhráli partii.

1.10. Schody

Často používanou technikou ve světe Go jsou schody. Využívá se při nich posloupnosti tahů (často jde o dlouhou posloupnost tahů), které protihráče zanechají v atari. Hráč stavící schody střídavě dává své kameny z obou stran soupeřovy skupiny a směřuje tuto skupinu na stěnu hrací desky, kde lze tuto skupinu snadněji zajmout.

Je zajímavou otázkou, jak dopadne situace na desce, kdy v cestě schodům jsou nějaké soupeřovy kameny. V takovém případě schody fungovat mohou, ale také nemusí. Vše je potřeba dobře propočítat před zahájením stavění schodů.

1.11. Handicap – metoda pro vyvážení síly hráčů



Obrázek 4:

Handicap o velikosti 9 kamenů.

Handicap není goistickou technikou v pravém slova smyslu. Pokud spolu hrají hru Go nebo hru Atari Go dva hráči nestejných kvalit, je technika handicapu tou pravou metodou, jak kompenzovat slabost jednoho z hráčů, a učinit tak partii více vyrovnanou.

Handicap spočívá v tom, že si černý (slabší) hráč položí na hrací desku určitý počet svých kamenů ještě před začátkem hry. Jde tedy o nevýhodu (handicap) pro bílého (silnějšího) hráče.

Podle velikosti hrací desky je obyčejně určen i počet kamenů, které si černý hráč smí na desku položit. Při velikosti hrací desky 13 x 13 průsečíků je to běžně 5 černých kamenů, při desce 19 x 19 průsečíků je to standardně 9 nebo 13 černých kamenů. Pokud jsou handicapy použity, pak hru zahajuje hráč s bílými kameny.

Kompletní popis pravidel hry Go a dalších herních technik pro hru Go naleznete v knize od Tomáše Grossera [1].

2. Představení programu *Atarigo*

2.1. Seznámení uživatele s programem *Atarigo*

V této kapitole se čtenář seznámí s programem *Atarigo*, který umožňuje hru Atari Go na počítači.

Program *Atarigo* je grafické herní prostředí pro Atari Go na počítači. Celé prostředí pro hru Atari Go je v češtině. V prostředí může uživatel sehrát partii Atari Go proti jinému uživateli, nebo proti počítači. Prostředí umožňuje také sledovat hru počítače proti počítači. Každou rozehranou partii může uživatel uložit na disk. Později tuto uloženou partii může načíst a pokračovat ve hře. Pro výukové účely je v prostředí podporuja funkce vrácení tahu zpět. Hráč se tak během partie může vrátit dokonce o libovolně kroků zpět (maximálně však do výchozího stavu na hrací desce). Jako protiváhu k vrácení tahu zpět prostředí podporuje také funkci posunutí o krok vpřed. Pro lepší přehled o hře si také uživatel může partii kdykoli přehrát od začátku až do posledního položeného kamenu na desku. Inteligenci zabudovaného počítačového hráče v programu může uživatel ovlivňovat nastavením 5 různých konstant. V neposlední řadě pak v prostředí uživatel nalezne odkazy na internetové stránky, kde si může přečíst vše o hrách Atari Go a Go.

Myšlenkou programu *Atarigo* je probudit v uživatelích programu, kterými by podle předpokladů měli být převážně nováčci ve hře Go, zájem do dalšího studia a aktivního hraní hry Go.

Přestože je prostředí programu v českém jazyce, zdrojový kód programu je pro větší přenositelnost napsán v anglickém jazyce. Ovládání programu je pomocí myši, uživatel může využít i klávesových zkratek, kterými zavolá vybrané funkce programu na jedno stisknutí klávesy. Pokud by význam nějakého prvku v programu nebyl uživateli jasný, pak se mu po chvíli, když podrží nad daným prvkem kurzor myši, objeví vyskakovací nápověda.

Program *Atarigo* dokáže řídit hru Atari Go ve třech režimech:

- *člověk vs. člověk* – v tomto módu program řídí hru dvou lidí. Prostředí programu vyhodnocuje povolené a zakazané tahy a kontroluje hrací desku na vítězství jednoho z hráčů partie

- *člověk vs. počítač* – v tomto módu hraje člověk proti počítačovému hráči. Na základě nastavení uživatele počítač hraje buď s černými, nebo bílými kameny. Prostředí programu pracuje stejně jako v předchozím módu

- *počítač vs. počítač* – tzv. mód pozorovatele. V tomto módu uživatel pozoruje hru počítače proti počítači. Prostředí programu pracuje stejně jako v předešlých režimech.

2.2. Sehrání partie v programu *Atarigo*

Po startu programu se uživateli zobrazí hrací deska (někdy také nazýváno goban) v levé časti hlavního okna a dále několik podoken pro nastavení parametrů hry a parametrů hrací desky v pravé části hlavního okna vedle hrací desky.

Před startem každé partie je nutné vybrat hráče hry, pokud chcete hráče změnit. Standardně totiž zůstávájí hráči vybráni tak, jak hráli v prostředí hráli poslední hru. Například pokud člověk hrál za černého a počítač za bílého, potom v nové partii bude opět člověk za černého a počítač za bílého.

Změnu hráčů provedete před zahájením partie v podokně *Nastavení hráčů hry*. V tomto podokně jednoduše vyberete, kdo bude hrát za černého a kdo bude za bílého. Provedené změny není třeba nijak potvrzovat. Při zahájení hry budou provedené změny potvrzeny automaticky.

Pokud si přejete změnit hráče v průběhu partie, také není problém. Ve stejném podokně Nastavení hráčů hry vyberete požadovanou volbu. Změna nastavení se projeví v následujícím tahu hráče, pro kterého jste změnu provedli.

Novou partii začnete kliknutím na tlačítko **Začít hru** v podokně *Připraveno ke hře*.

Jakmile jste začli partii, okno *Připraveno ke hře* zmizí a na jeho míste ho nahradí okno *Rychlé upravení hrací desky*. V tomto podokně můžete provést změnu parametrů hrací desky, tj. upravit velikost hrací desky, zvolit jedno ze tří startovních rozložení kamenů na desce a také můžete určit, který hráč bude začínat partii (přestože je proti oficiálním pravidlům, aby partii začínal bílý). Partii zahajuje vždy černý, není-li uživatelem vybráno jinak.

Pro načtení nové desky se změněnými parametry desky stiskněte tlačítko **Nová.** Po stisknutí tlačítka Nová se přepnete zpět do podokna *Připraveno ke hře* a program bude čekat na start nové partie.

Před startem partie můžete na desku přidat handicapy. Vysvětlení pojmu handicap bylo provedeno v první kapitole tohoto textu. Handicapy můžete na desku přidat pouze před začátkem partie. Prostředí umožňuje na hrací desku přidat libovolný počet kamenů barvy černé i bílé, tedy uživateli dovoluje experimentovat a nastavit si obtížnost hry, jak mu bude vyhovovat.

Pro přidání černých kamenů na desku stiskněte ikonu s černým kamenen na hlavní liště programu nahoře nad hrací deskou. Tlačítko se zvýrazní. Tím tlačítko indikuje, že jste v režimu přidávání černých kamenů na hrací desku. Nyní najetím kurzoru myši nad zvolený průsečík a stisknutím levého tlačítka myši (klasickým klepnutím na průsečík) položíte černý kámen na desku. Jste-li hotovi s přidáváním černých kamenů, můžete na hrací desku přidat bílé kameny nebo přidávání handicapů libovolně kombinovat.

Pro přidání bílých kamenů je postup analogický, pouze s tím rozdílem, že místo ikonky s černým kamenem vyberete ikonku s bílým kamenem.

Kameny na desku můžete přidávat jako handicapy (pokud na desku dáte odpovídající počet kamenů slabší barvy), nebo pro vytvoření startovní pozice[1] na desce, abyste učinili partii více zajímavou.

Více o nastavení parametrů naleznete v odstavci *2.3. Nastavování vlastností hry*.

Hrajete-li hru člověka proti člověku, potom se hráči střídají po odehraném tahu (barva pokládaných kamenů se po odehraném tahu střídá). Který hráč je právě na tahu, vidíte vpravo nahoře vedle hrací desky v okně *Informace o této hře*.

Hrajete-li v módu člověk proti počítači, potom na desku pokládáte jen kameny v barvě, za kterou hrajete. Počítačový hráč provede svůj tah ihned po Vašem tahu.

Při hře počítače proti počítači nepokládáte na desku žádné kameny. Kameny jsou pokládány automaticky a Vy můžete pouze pozorovat hru.

Poslední položený kámen na desku je pro lepší přehled o partii označen červeným čtverečkem uprostřed.

Během hry program automaticky kontroluje, zda lze na dané políčko hrát (není-li zamýšlený tah sebevražda – v případě, že tah je sebevražda, potom uživateli zobrazí chybovou hlášku a počká na opravný tah stejného hráče).

Dále program kontroluje hrací desku na výhru jednoho z hráčů. V případě výhry hráče ukončí partii a zobrazí uživateli zprávu o vítězi partie.

Aktuální stav hry naleznete v okně *Údaje o této hře* vpravo nahoře vedle hrací desky. V tomto okně vidíte, který hráč je právě na tahu. Také zde vidíte, kolik tahů již bylo během této partie odehráno. Pokud již partie má svého vítěze, naleznete zde údaj o tom, který hráč je vítěz a také tlačítko **Ukázat zajaté kameny**, jímž na desce můžete znovu zobrazit zajaté a odebrané kameny.

Pokud se chcete podívat na přehled odehraných tahů v partii, tento přehled naleznete v podokně *Přehled tahů partie*. V tomto podokně je na jednom řádku zobrazeno číslo tahu společně s barvou kamenu a souřadnicemi na desce, na které byl tah zahrán. Pokud si přejete zvýraznit daný kámen na hrací desce, stačí kliknout na vybraný řádek a kámen na desce se zobrazí s červeným čtverečkem uprostřed.

Uživatel také může současnou partii přehrát od prvního až do posledního tahu. Stačí kliknout na ikonu s obrázkem filmového pásku na hlavní liště programu. Hrací deska potom bude vymazána a postupně se na ní budou objevovat kameny v takovém pořadí, v jakém byly položeny na desku. Na konci přehrávání partie se zobrazí hláška o tom, že přehrávání bylo ukončeno. Pokud partie ještě nemá svého vítěze, pak po konci přehrávání můžete pokračovat ve hře. V opačném případě je deska již zamknuta a nelze na ni hrát.

[1] Pokud vytvoříte startovní pozici na desce takovou, že některá skupina kamenů bude zajmuta ještě před startem partie, program tyto zajaté kameny vybere z desky. Na danou desku však již nebudete moci přidat žádný kámen a při požadavku na start partie na takové desce jen dostanete upozornění, že partie již má svého vítěze.

Pro uživatele zajímavou vlastností programu, vhodnou převážně pro výuku hraní hry a poučení se ze zahraných chyb, je možnost převíjet hru libovolně zpět a dopředu od aktuálního stavu na hrací desce.

Jestliže je na hrací desce položen alespoň jeden kámen, uživatel se může vrátit zpět do stavu desky před položením tohoto kamene na desku. Krok zpět provedete kliknutím na tlačítko se šipkou zpět na hlavní liště programu nahoře nad hrací deskou. Krok zpět můžete provádět opakovaně do té doby, než bude deska prázdná. Pak se toto tlačítko zamkne a nebude možné na něj kliknout.

Jakmile v programu vrátíte hru do stavu o několik kroků vzad, máte také možnost provést kroky vpřed, a vrátit se tak do výchozího stavu na hrací desce, tj. do stavu před kroky zpět. Krok vpřed provedete stiskem tlačítka se šipkou dopředu na hlavní liště programu nahoře nad hrací deskou.

Během posunování partie je partie pozastavena a hrací deska zamknuta proti případnému položení kamenů. Během posunování partie se zobrazí okno *Převíjení hry*, které obsahuje tlačítko **Zpátky do hry**.

Otázkou úzce související s posunováním partie vzad a vpřed je otázka, z kterého stavu si přejeme pokračovat dál ve hře (tj. odemknout desku a prohlásit současný stav na hrací desce jako výchozí pro případné další pousování tahů partie). V programu *Atarigo* si uživatel zvolí, z kterého stavu hrací desky bude dál pokračovat ve hře, stiskem tlačítka **Zpátky do hry**. Okno *Převíjení hry* pak zmizí a partie může dál pokračovat.

Neodmyslitelnou součástí každého herního prostředí je i možnost uložit a načíst rozehranou hru. V programu *Atarigo* uložíte probíhající partii kliknutím na tlačítko **Uložit rozehranou hru do souboru** (také stisknutím klávesy F3) v hlavním menu programu. Načtení hry provedete také z menu kliknutím na tlačítko **Načíst uloženou hru ze souboru** (také stisknutím klávesy F4). Otevře se Vám nové okno, ve kterém si vyberete soubor, který chcete pro uložení/načtení použít. V případě ukládání máte možnost napsat jméno souboru, a vytvořit tak nový soubor.

2.3. Nastavování vlastností hry

V pravé polovině hlavního okna vedle hrací desky vidíte několik menších oken. Většina z těchto oken již byla popsána v předešlé časti *Sehrání partie v programu Atarigo*. Pro kompletní přehled o funkcích a možnostech programu však všechna tato okna nyní vyjmenuji a vysvětlím jejich funkce a možnosti znovu.

2.4. Okno Údaje o této hře

V tomto okně naleznete základní informace o právě probíhající hře. Je zde zobrazeno, který hráč je právě na tahu, kolik tahů již bylo odehráno a kdo vyhrál ve hře (pokud již vítěze hry známe). Jestliže je již vítěz známý, pak v tomto okně naleznete také tlačítko **Ukázat zajaté kameny**, kterým na desce můžete znovu zobrazit kameny, které byly zajaté a odebrané z desky.

2.5. Okno Nastavení hráčů

V tomto okně nastavíte hráče pro Vaši partii. Vyberete, kdo bude hrát za černého hráče, a kdo bude hrát za bílého. Pro každou barvu kamenů máte na výběr ze 3 možností hráčů:

- *Člověk* = za vybraného hráče bude hrát člověk

- *Greedy* = počítač bude hrát algoritmem Greedy (v překladu nenasytný). Tento algoritmus je rychlý, protože hledá nejlepší tah pouze do hloubky 0 v herním stromě[2]. Veškerá jeho síla nebo slabost proto závisí na kvalitní statické ohodnocovací funkci[3]. Jeho výkon bývá obyčejně slabý, je však vykoupen rychlostí provedení algoritmu

- *AlphaBeta* = počítač bude hrát algoritmem αβ. Tento algoritmus hledá nejlepší tah do hloubek větších než 0 v herním stromě. Opět se však spoléhá na kvalitní ohodnocovací funkci. Algoritmus AlphaBeta je silnější než Greedy, jeho výkon je ovšem vykoupen rychlostí, resp. pomalostí provedení algoritmu[4]

Změny hráčů není nutné nijak potvrzovat, budou potvrzeny automaticky při startu nové partie. Také můžete změnit hráče za obě barvy kdykoli během partie. Změna se projeví při dalším tahu hráče, pro kterého jste změnu provedli. Při změně hráče se pro Vaši lepší kontrolu změní i malá ikonka indikující, kdo za danou barvu kamenů bude hrát.

2.6. Okno Nastavení hrací desky

V tomto okně nastavujete parametry hrací desky (gobanu). Pomocí posuvníku vyberete velikost hrací desky od 5 x 5 průsečíků do 9 x 9 průsečíků s krokem po jednotce. Na menších deskách než 5 x 5 průsečíků je hra Atari Go bezpředmětná, naopak pro desky větší než 9 x 9 průsečíků se pak již obvykle přechází na hru Go.

Dále vyberete jedno ze tří možných startovních rozložení kamenů na desce. Na výběr máte prázdnou desku, stabilní centrum desky nebo crosscut (křížový střih) v centru desky[5]. Při stabilním centru i při crosscutu v centru jsou na desku položeny vždy 4 startovní kameny. Tyto startovní rozložení pomáhají při zahájení partie.

Zaškrtnutím volby *Začíná bílý* můžete určit, jestli bude partii začínat hráč s bílými kameny.

---

[2] Vysvětlení pojmů herní strom a hloubka prohledávaného stromu naleznete v následující části *3. Programátorská dokumentace programu*.
[3] Vysvětlení pojmu statická ohodnocovací funkce naleznete v části *4.7. Ohodnocovací funkce*.
[4] Algoritmus αβ (AlphaBeta) může být i velmi! pomalý, pokud je přinucen hledat do velkých hloubek v herním stromě.
[5] Pokud je velikost desky lichá, pak počáteční rozestavení kamenů není přímo v centru hrací desky.

2.7. Okno Převíjení hry

Okno *Převíjení hry* uvidíte pouze během krokování hry směrem vzad nebo vpřed. Okno upozorňuje uživatele, že právě probíhající partie je pozastavena (hrací deska je zamknuta na položení kamene) a uživatel může hru libovolně krokovat vzad nebo vpřed.

Okno obsahuje jediné tlačítko **Zpátky do hry**. Kliknutím na toto tlačítko okno *Převíjení hry* zmizí a partie bude pokračovat z takového rozložení kamenů, jaké na hrací desce zrovna je. Krokování hry vzad a vpřed je vhodné zejména pro výuku hry, neboť uživatel může svůj tah kdykoli vrátit; dokonce i v případě, že partie již byla vyhrána některým z hráčů, je možné se vrátit do předchozích stavů a poučit se z chybných tahů.

2.8. Okno Deska je připravena ke hře

Toto okno je viditelné kdykoli, když je deska ve stavu, kdy se čeká na zahájení partie. Dokud není partie zahájena, je možné na desku položit handicapy pro vytvoření startovní pozice.

Okno v sobě shrnuje a zobrazuje všechny údaje o hrací desce jako je její velikost, startovní pozice partie a počet handicapů na desce.

Okno také obsahuje dvě tlačítka. Tlačítkem **Zpět k nastavení** se vrátíte do okna *Nastavení hrací desky*. Tlačítkem **Rozehrát** potom začnete partii.

2.9. Okno Nastavení statické funkce

Okno *Nastavení statické funkce* umožňuje uživateli měnit kvalitu hry automatického počítačového hráče.

Změnou konstant α, β, γ, δ, ε ovlivníte chytrost hry počítače jak při použití herního algoritmu Greedy, tak při použití herního algoritmu AlphaBeta (αβ), protože oba algoritmy používají stejnou statickou ohodnocovací funkci[6] pro číselné ohodnocení aktuálního rozložení kamenů na hrací desce.

Význam výše jmenovaných parametrů je následující:

- α (parametr Alfa) = váha počtu svobod 1. řádu podle Manhattanské vzdálenosti[7]. Čím větší hodnota parametru α, tím více bude počítač umísťovat kameny na průsečíky, které po položení kamenu budou mít maximálně svobod

- β (parametr Beta) = váha počtu svobod 2. řádu podle Manhattanské vzdálenosti. Čím větší hodnota parametru β, tím více bude počítač umísťovat kameny na otevřená prostranství

[6] Podrobnější informace o statické ohodnocovací funkci naleznete v části Ohodnocovací funkce.

[7] Manhattanská vzdálenost n-tého řádu vybírá všechny průsečíky v mřížce, k nimž se z daného výchozího umístění lze dostat po právě n krocích, kde můžete chodit jen po hranách a jeden krok je cesta po hraně od jednoho průsečíku ke druhému. Název je odvozen od situace taxíků na Manhattanu v New Yorku, které cenu jízdného měří právě Manhattanskou vzdáleností.

- γ (parametr Gamma) = váha počtu svobod 3. řádu podle Manhattanské vzdálenosti. Čím větší hodnota parametru, tím více bude počítač umísťovat kameny do otevřených prostranství

- δ (parametr Delta) = váha parametru útočnost vs. obrana při pokládání kamenů. Čím menší hodnota parametru Delta, tím bude počítač hrát útočnější, tím pádem i riskantnější, hru

- ε (parametr Epsilon) = váha parametru sdružování kamenů pro vytváření silnějších skupin a zároveň vytváření očí pro skupiny kamenů stejné barvy. Čím větší číslo pro parametr Epsilon nastavíte, tím roztroušenější budou kameny, které pokládá počítač, na hrací desce. Záporné hodnoty parametru Epsilon naopak vedou k pokládání kamenů do sousedství jiných kamenů stejné barvy tak, aby kameny tvořily větší skupiny. Malé skupiny kamenů jsou snadnější k zajmutí, zatímco spojením dvou malých skupin do jedné větší získáte skupinu, která bude odolnější proti zajmutí

Během vývoje programu se konstanty statické ohodnocovací funkce vyladí a vytvoří se několik úrovní síly automatického počítačového hráče, kde každá úroveň odpovídá určitému nastavení konstant. Uživateli jsou pak obyčejně dány k dispozici pouze tyto výstupní úrovně. V programu *Atarigo* však může uživatel nastavit každý z výše uvedených parametrů, a sám si tak vyzkoušet jaký dopad změna bude mít na kvalitu počítačového hráče.

Vhodným nastavením těchto 5 parametrů můžete získat počítačového hráče, jehož hra odpovídá přibližně středně pokročilému lidskému hráči. Pro získání silnějšího hráče by bylo nutné navrhnout složitější statickou ohodnocovací funkci s větší sadou rysů, které funkce bude na desce sledovat a vyhodnocovat.

2.10. Okno Přehled tahů partie

V tomto okně naleznete v tabulce přehledný popis všech tahů právě probíhající partie. Při každém odehraném tahu je do této tabulky přidán zaznám obsahující číslo tahu, barvu hraného kamenu a souřadnice průsečíku, na který byl kámen položen.

Kliknutím na řádek v tabulce zvýrazníte daný kámen na hrací desce. Zvýraznění je provedeno malým červeným čtverečkem uprostřed kamenu.

3. Programátorská dokumentace programu

V této části dokumentace společně nakouknene pod pokličku programu *Atarigo*. Čtenář si zde přečte, jaké datové struktury jsou použity pro reprezentaci hrací desky a jaké řešení je použito pro vykreslování hrací desky a kamenů.

Stěžejní bod této části pak bude spočívat v popisu různých programátorských přístupů a algoritmů k naprogramování automatického počítačového hráče hry Atari Go.

Po obecném úvodu do programování umělé inteligence pro hry přistoupíme k popisu řešení, které pro automatického hráče vyžívá program *Atarigo*.

3.1. Obecný popis vlastností programu *Atarigo*

Celý program je napsán v prostředí CodeGear RAD Studio Delphi 2007 pro platformu Win32. Výhodou programu je, že ke své práci nepotřebuje podporu runtime knihovny .NET. Díky platformě Win32 tak tento program mohou bez problémů spustit a používat všichni uživatelé počítačů s operačním systémem Windows.

Prostředí CodeGear RAD Studio Delphi 2007 jsem zvolil kvůli jeho rozsáhlým možnostem při vytváření GUI aplikací (tj. aplikací, které mají grafické prostředí pro interakci s uživatelem) a přehlednému prostředí při vývoji programů.

Program se drží myšlenek strukturovaného programování. Během psaní programu jsem se snažil jednotlivé výpočtové funkce vždy od sebe důsledně oddělit tak, aby vznikaly malé a efektivní funkce. Složitá funkce pak vznikla z několika menších. Je to tak proto, aby bylo umožněno snadněji změnit cílovou funkci změnou základních funkcí. Změnou základních funkcí tak ovlivníte výpočet složitých funkcí, jako je např. funkce pro výpočet nejlepší pozice pro tah hráče.

Jedno z praktických využití této vlastnosti pro programátory je možnost vytvořit si počítačového hráče podle svých představ například změnou statické ohodnocovací funkce, což se následně promítne do funkce pro určení nejlepšího tahu hráče a do funkce pro položení kamene dané barvy na danou pozici. Změny ohodnocovací funkce lze dosáhnout například i změnou jednotlivých vah ve funkci pro výpočet hodnoty E (Eulerova čísla). Při změně ohodnocovací funkce pak zbytek programu zůstává beze změny.

3.2. Datové struktury použité pro reprezentaci hrací desky

Pro reprezentaci hrací desky program *Atarigo* používá dynamické dvourozměrné pole složeného typu, v programu pod svým identifikátorem Board.

Slovo dynamické v názvu značí, že operační paměť použitá pro toto pole je alokována dynamicky při načtení nové desky podle velikosti hrací desky. Dynamická alokace tedy přidělí právě tolik paměti, kolik pro reprezentaci desky a kamenů na ní potřebujeme. Je to metoda vhodná pro situace, kdy předem neznáme potřebnou velikost paměti. Přitom pokud už není pole potřeba, je paměť přidělená poli zase vrácena zpět operačnímu systému. Nedochází tedy k plýtvání s operační pamětí.

Výraz složeného typu pak označuje, že na každé místo [x,y] v dvourozměrném poli je možné zaznamenat několik hodnot, ekvivalentně lze říci, že na každé místo [x,y] lze zaznamenat strukturu.

V programu *Atarigo* představuje každé místo Board[x,y] jeden průsečík na hrací desce se souřadnicemi [x,y]. Bod Board[0,0] označuje levý horní roh hrací desky.

Pro každý průsečík na hrací desce si program pamatuje tuto strukturu:

- *obsazenost průsečíku* = proměnná typu Char = {'E','B','W'} po řadě značí, že daný průsečík je volný (Empty); nebo je na něm položen černý kámen (Black); nebo je na něm položen bílý kámen (White)

- *svobody kamenu* = proměnná typu Integer označuje počet svobod položeného kamenu na průsečíku

- *zajatec* = proměnná typu Boolean (True/False) označuje, zda byl kámen zajat (slouží pro vyjímaní zajatých kamenů z desky a pro spočítání, kolik kamenů bylo zajato).

- *zvýraznění kamene* = proměnná typu Boolean (True/False) udává, jestli má být kámen označen červeným čtverečkem uprostřed

3.3. Hrací deska a vykreslování kamenů na desku

Hrací deska je kreslena na komponentu TDrawGrid, která umožňuje vykreslit různou bitmapu na jednotlivá políčka komponenty.

Jednotlivá políčka na komponentě TDrawGrid jsou přístupná přes souřadnice [ACol,ARow]. Políčka TDrawGrid[ACol,ARow] odpovídají obsahu Board[ACol,ARow]. Převodní funkce mezi těmito dvěma tabulkami je proto velmi jednoduchá.

Použité bitmapy jsou uloženy přímo v programu v jeho zdrojovém kódu. Odpadá tak nutnost načítat bitmapy z externích zdrojů/externích souborů (přestože je protředí programu i na tuto možnost připraveno). Také se tím vyvarujeme chyb a problémů v programu, jestliže některá očekávaná bitmapa chybí.

Při jakékoli změně plochy hlavního okna programu nebo při položení nového kamenu na desku je vyvolána událost Repaint pro překreslení políček komponenty TDrawGrid.

Hlavní okno programu využívá techniku dvojitého bufferování pro rychlé překreslování hrací desky bez obtěžujícího blikání. Technika dvojitého bufferování spočívá v tom, že namísto přímého kreslení na obrazovku se výstupní bitmapa nejprve celá vykreslí do interní tabulky v programu a teprve z této tabulky je pak vykreslena na obrazovku. Tímto postupem se eliminuje problíkávání při překreslování hrací desky v hlavním okně programu.

4. Programování automatického počítačového hráče a umělá inteligence (AI)

V této kapitole se dostáváme k nejzajímavější a zároveň nejobtížnější náplni celé práce vytváření programu. Tou náplní je naprogramování automatického počítačového hráče pro hru Atari Go.

Nejdříve se zaměříme na vysvětlení pojmu prohledávání stavového prostoru ve hře a také popíšeme důvody, proč prohledáváme. Následně přejdeme k popisu prohledávacích technik, které se běžně užívají v současných počítačových programech. Na závěr popíšeme metodu prohledávání použitou v programu *Atarigo*.

4.1. Prohledávání stavového prostoru aneb Proč prohledávat

Hra Go stějně jako hra Atari Go je hra dvou hráčů s nulovým součtem a úplnou informací. Výraz s nulovým součtem znamená, že ve hře existuje pouze jeden výherce, jehož výhra (zisk) se rovná prohře (ztrátě) druhého hráče. Výraz s úplnou informací znamená, že oba hráči mají kompletní přehled o všech tazích druhého hráče.

Hru Go můžeme reprezentovat jako orientovaný graf, jehož každý vrchol představuje jeden možný stav rozložení kamenů na hrací desce a hrany v tomto grafu představují tah, kterým deska přechází mezi dvěma svými povolenými stavy.

Je zvykem tento orientovaný graf nazývat strom hry. Jistě jste již seznámeni s pojmem strom hry, proto není nutné tento pojem dlouze uvádět.

Pojmem prohledávaný strom označíme takovou část herního stromu, která je prozkoumávána hráčem (ať už počítačem, nebo člověkem).

Prohledávaný strom má jeden kořenový uzel, který odpovídá současné situaci na hrací desce. Tato situace je výchozí situací pro hráče, který je právě na tahu. Všechny povolené tahy z této výchozí situace jsou reprezentovány hranami, které vedou mezi výchozí situací (kořen stromu) a novou situací na desce (uzel stromu).

Všechny povolené tahy z výchozí situace tedy rozvětví prohledávaný strom do vzdálenosti 1 od kořene stromu. Uzly ve vzdálenosti 1 mohou být rozvětveny na uzly ve vzdálenosti 2 od kořene stromu a takto bychom mohli pokračovat do dalších hladin. Kořen prohledávaného stromu si můžeme představit jako uzel v hloubce 0 (od kořene stromu). Analogicky uzly ve vzdálenosti 1 od kořene jsou uzly v hloubce 1, atp. Počet hran vycházejících z jednoho uzlu je nazýván větvící faktor.

Průměrný větvící faktor a hloubka d, do jaké byl kořen prohledávaného stromu prozkoumán, jsou důležité ukazatele složitosti stromu hry.

Pokud je uzel v hloubce d rozvětven na uzly v hloubce $d+1$, pak jsou tyto nové uzly nazývány synové uzlu. Rozvětvený uzel je pak nazývám rodičem. Pokud už uzly nejsou dále větveny, nazývají se listy.

Existuje několik důvodů, proč nejsou listy dále větveny:

- pozice odpovídající listu může být konečná pozice, kdy je již znám výsledek hry (takové listy jsou pak označovány jako koncové listy)
- pro další rozvoj listu nemáme dostatek potřebných zdrojů (například času nebo paměti na prozkoumání všech možných synů uzlu)
- další větvení listu již není potřeba

Proces větvení prohledávaného stromu a ohodnocování uzlů tohoto stromu za účelem nalezení optimálního tahu hráče nazýváme hledání.

Čistě teoreticky tedy můžeme postavit herní strom, prohledat ho od kořene až po listy, které odpovídají konečným (výherním) pozicím hry a potom cestou zpátky do kořene stromu (backtrackingem) najít tah, který nám zaručí optimální hru.

Pro jednoduché hry jako jsou například Piškvorky na malé desce lze takový herní strom sestavit až do konečných listů a najít optimální tah hráče i v praxi. Pro hru Go nebo Atari Go je však takový herní strom kvůli omezeným zdrojům paměti a času příliš velký na to, aby byl postaven a prozkoumán celý až do konečných listů.

Pro lepší představu nyní uvedu odhad velikosti herního stromu pro hru Go. Tento odhad předpokládá průměrný větvící faktor uzlu 250 a průměrnou délku hry pouze 150 tahů, což odpovídá hloubce herního stromu 150 (poznamenejme přitom, že nejdelší hry profesionálních hráčů mají více než 400 tahů). Potom má herní strom velikost $250^{150} \approx 10^{360}$ uzlů[8], proto je technicky nemožné herní strom plně rozvětvit až do koncových listů.

Protože je rozvoj stromu až do koncových nemožný téměř ve všech případech, listy prohledávaného stromu běžně neodpovídají koncovým listům. Abychom však mohli použít prohledávání, musíme umět ocenit i pozici, která neodpovídá konečné pozici.

Musíme tedy na základě znalostí hry navrhnout ohodnocovací funkci, která by dokázala odhadnout hodnotu desky, která neodpovídá konečné pozici. Ohodnocovací funkce tedy ocení hodnotu podstromu daného listu, který však nebyl dále větven. Taková ohodnocovací funkce se potom nazývá heuristická ohodnocovací funkce.

Není přitom bez zajímavosti, že teoreticky bychom vystačili s perfektní heuristickou ohodnocovací funkcí, která by ocenila listy v hloubce 1 (tedy pouze bychom rozvětvili kořen prohledávaného stromu).

Protože však v praxi je vytvoření perfektní heuristické ohodnocovací funkce stejně jako plné rozvětvení stromu hry do koncových listů nemožné, většina programů, které používají prohledávaní, využívá vyvážený poměr mezi prohledáváním a znalostmi hry, kdy některé pozice jsou ohodnocené přímo na základě znalostí hry heuristickou ohodnocovací funkcí, zatímco jiné pozice jsou nejdříve větveny a následně ohodnoceny.

[8] Odhad na velikost herního stromu pochází od autora Erika van der Werfa z jeho internetových stránek http://erikvanderwerf.tengen.nl/

4.2. Přehled prohledávacích technik

Základem téměř všech algoritmů, které prohledávají strom hry, je algoritmus Minimax. Prestože je algoritmus Minimax průkopníkem myšlenky prohledávaní stromu hry, žádný z moderních programů, které prohledávají strom hry, již algoritmus Minimax ve své původní podobě nepoužívá.

Namísto čistého Minimaxu tyto moderní hrací programy používají některou z forem $\alpha\beta$ prohledávaní, které má dnes mnoho implementačních variant.

4.3. Algoritmus Minimax

Algoritmus Minimax je rekurzivní algoritmus pro nalezení nejlepšího tahu hráče. V algoritmu Minimax jsou dva typy uzlů ve stromě. Prvním je uzel typu Max, kde se hráč na tahu snaží maximalizovat zisk. Kořen prohledávaného stromu (v hloubce 0) a všechny uzly na sudých hladinách prohledávaného stromu jsou uzly typu Max. Druhým typem uzlu je uzel Min, kde se protihráč snaží minimalizovat minimalizovat zisk. Všechny uzly na lichých hladinách stromu jsou uzly typu Min.

Algoritmus Minimax je rekurzivní algoritmus, který začíná v listech prohledávaného stromu a postupně se vrací zpět do kořene stromu. Vybírá maximální hodnotu ze všech synů uzlu, pokud je uzel typu Max. Analogicky minimální hodnotu ze všech synů uzlu, pokud je uzel typu Min.

Takto algoritmus ohodnotí uzly na jednotlivých hladinách stromu, až se dostane zpět do kořene prohledávaného stromu. Výstupem algoritmu tedy bude hodnota hrací desky a s ní spojený nejlepší tah, který na danou desku vede.

4.4. Pseudokód pro algoritmus Minimax

```
Minimax (node){
 if node is a leaf
   return the heuristic value of node

 if node is a maximizing node{
   score = -∞
   for each child of node{
     val = minimax(child)
    if val > score
      score = val
    }
   return score
 }

 if node is a minimizing node{
   score = +∞
   for each child of node{
     val = minimax(child)
    if val < score
      score = val
    }
   return score
 }
```

4.5. Algoritmus AlphaBeta ($\alpha\beta$) prohledávání

V předchozím odstavci jsme se seznámili s algoritmem Minimax, který slouží k nalezení nejlepšího tahu hráče. Nevýhodou algoritmu Minimax je však velký počet uzlů, které algoritmus musí vyhodnotit, a tím pádem i velká časová a paměťová náročnost algoritmu.

Algoritmus $\boldsymbol{\alpha\beta}$ prohledávání dokáže vrátit stejnou hodnotu hrací desky a stejný nejlepší tah pro hráče jako algoritmus Minimax, je však rychlejší a efektivnější než algoritmus Minimax. Tím umožňuje prohledat herní strom do větší hloubky ve stejném čase jako Minimax.

Vylepšených vlastností algoritmu $\boldsymbol{\alpha\beta}$ je dosaženo použitím dvou mezních hodnot $\boldsymbol{\alpha}$ a $\boldsymbol{\beta}$ pro hodnotu hrací desky během prohledávání stromu hry.

Spodní hranice, hodnota $\boldsymbol{\alpha}$, představuje nejhorší možnou hodnotu hrací desky pro hráče Max. Protože hráč Max vybírá maximální hodnotu z uzlů na dané hladině, potom jakýkoli podstrom uzlu, který má hodnotu menší než $\boldsymbol{\alpha}$, již nemusíme prohledávat, neboť Max takový tah určitě nezvolí (tato situace je nazývána $\boldsymbol{\alpha}$ řez).

Naopak horní hranice, hodnota $\boldsymbol{\beta}$, představuje nejhorší možnou hodnotu hrací desky pro hráče Min. Jestliže z nějakého uzlu na Min hladině nalezneme tah, který vede na hrací desku s hodnotou větší než $\boldsymbol{\beta}$, potom daný uzel nemusíme dále prohledávat, neboť hráč Min jistě takový tah nezvolí (tato situace je nazývána $\boldsymbol{\beta}$ řez).

4.6. Pseudokód pro algoritmus AlphaBeta ($\alpha\beta$)

```
alphabeta (node, alpha, beta){
    if node is a leaf
    return the heuristic value of node

 if node is a maximizing node{
                for each child of node{
    beta = min(beta, alphabeta(child, alpha, beta))
                      if beta <= alpha
                           return alpha
                    }
                    return beta
                }

              if node is a minimizing node{
                for each child of node{
                    alpha = max(alpha, alphabeta(child, alpha, beta))
                      if beta <= alpha
                        return beta
                    }
                    return alpha
                }
            }
```

4.7. Ohodnocovací funkce

V předchozí části jsme si vyjmenovali základní používané algoritmy pro hledání ve stromu hry. Všechny tyto algoritmy mají jednu společnou věc. U všech uvedených algoritmů se hledání v herním stromě řídí tím, jaké ohodnocení desky jim předá ohodnocovací funkce. Můžeme proto bez újmy na obecnosti říci, že kvalitní ohodnocovací funkce je základem silného počítačového hráče.

Čtenář této práce jistě již pronikl do problematiky této hry, a dovede si proto představit, jak těžké je najít kvalitní a zároveň rychlou ohodnocovací funkci pro tuto hru.

Ohodnocovací funkce použitá v programu *Atarigo* se zaměřuje na následující 4 body:

1. *maximalizovat svobody kamenů*
2. *maximalizovat území hráče*
3. *spojovat kameny do skupin*
4. *vytvářet živé skupiny (vytvářet skupinám oči)*

První bod ohodnocovací funkce maximalizovat svobody hráčových kamenů vychází přímo z cíle hry Atari Go zajmout kameny soupeře (minimalizovat jejich svobody). Počet svobod kamene (skupiny kamenů) je dolní odhad na počet tahů, které jsou nutné k zajmutí kamene (skupiny kamenů). Je proto zřejmé, že snaha maximalizovat svobody svých kamenů je dobrou defensivní strategií, zatímco snaha o minimalizování svobod soupeřových kamenů je ofensivní strategií, která má za cíl vyhrát partii.

Druhý bod ohodnocovací funkce maximalizovat území obsazené hráčem je chytrou a zároveň nenápadnou strategií, jak porazit soupeře vyčerpáním jeho povolených tahů (sebevražda je v pravidlech hry zakázána). Pokud se hra dostane do stavu, že žádným tahem již nelze soupeřovu skupinu kamenů obklíčit, pak hráč, který kontroluje větší území na desce, vydrží déle. Hráč pokládá kameny dovnitř svého kontrolovaného území (dokud ho nezaplní úplně) a počká, až soupeř vyčerpá své povolené tahy.

Třetí bod ohodnocovací funkce vychází z pozorování, že mnoho malých skupinek kamenů je náročnějších na obranu než jen málo velkých skupin kamenů. Spojovat kameny do větších skupin je proto dobrá defensivní strategie, jak vytvořit velkou skupinu, která je odolnější proti zajmutí.

Čtvrtým bodem ohodnocovací funkce v programu *Atarigo* je vytváření očí ve skupinách. Technika vytváření dvou očí, popsaná v této práci výše v části *Základní techniky hry v Atari Go*, je technikou, jak vytvořit živou skupinu. Ve hře Go je živá skupina taková skupina kamenů, která je odolná vůči zajmutí.

Ve hře Atari Go je možné zajmout i živou skupinu. Podle pravidel hry musí hráč vždy udělat nějaký povolený tah, je zakázáno vzdát se tahu. Pokud už hráč nemá tah, tak partii prohrál. Ve hře Atari Go jsou tedy živé skupiny zajaty jen tehdy, pokud už hráč vyčerpal všechny ostatní tahy a je donucen zaplnit oči své skupiny.

Pro kameny soupeře platí předchozí 4 body v negovaném znění.

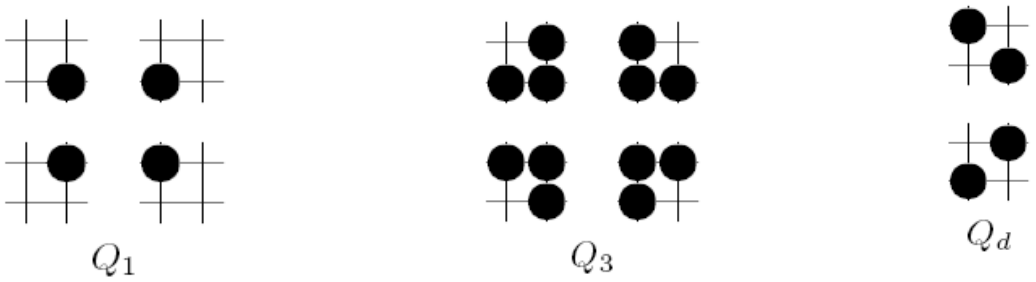
Pro první dva body ohodnocovací funkce (maximalizovat svobody kamenů a maximalizovat území) program počítá váženou sumu ze svobod 1., 2. a 3. řádu. Svoboda *n*-tého řádu je definovaná jako počet prázdných průsečíků na desce v Manhattanské vzdálenosti *n* od hráčových kamenů. Svobody řádů větších než 3 nejsou v programu počítány, protože se během testování ukázalo, že nemají přínos ke zlepšení kvality ohodnocovací funkce a jejich výpočet zdržoval výpočet ohodnocení desky. Místo počítání svobod pro jednotlivé skupiny kamenů, program vypočítá sumu ze svobod hráčových kamenů pro celou hrací desku.

Spojování kamenů a vytváření očí již není tak jednoduché na výpočet jako spočítat svobody kamenů na desce. Naštěstí můžeme v programu využít drobný trik, který dává dobrý dolní odhad na tyto zbylé dva body ohodnocovací funkce. Namísto složité detekce očí a spojení kamenů spočítáme Eulerovo číslo *E* pro binární obraz.

Eulerovo číslo *E* pro binární obraz je definováno jako počet skupin na hrací desce mínus počet děr v těchto skupinách.

Minimalizací totoho Eulerova čísla pro binární obraz pak dokážeme spojovat kameny a vytvářet oči ve skupinách.

Předností Eulerova čísla pro binární obraz je to, že je lehce vypočitatelné. K jeho výpočtu stačí posunovat okénko velikosti 2 x 2 průsečíky po hrací desce a počítat výskyty figur *Q1*, *Q3* a *Qd* na hrací desce.



Obrázek 5:

Rozložení kamenů pro figury Q1, Q3 a Qd ve čtverci 2 x 2 průsečíky.

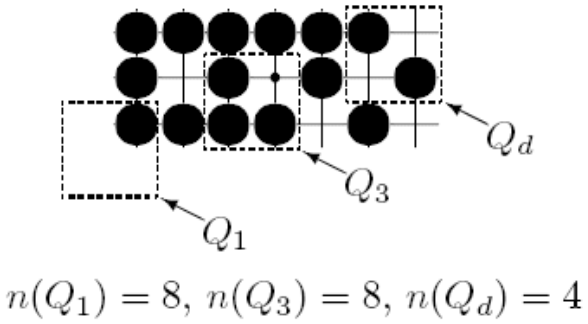
Skenováním desky získáme čísla *n(Q1)*, *n(Q3)* a *n(Qd)*. Tyto čísla odpovídají počtu výskytů figur *Q1*, *Q3* a *Qd* na desce.

Číslo *E* je Eulerovo číslo pro binární obraz. Na základě získaných informací ho vypočítáme jako

$$E = \frac{n(Q1) - n(Q3) + 2n(Qd)}{4}$$

Na příkladu výpočtu na Obrázku 6 vidíme, že Eulerovo číslo E = 2, což odpovídá 3 skupinám a 1 díře ve skupině.

V tomto případě bylo pouze levé oko ve skupině započteno jako díra. Abychom dokázali uznat více očí jako díry, bylo by nutné snížit váhu figury *Qd* nebo použít takový skenovací algoritmus hrací desky, které bere v úvahu také protihráčovy kameny při počítání výskytu jednotlivých figur na desce.



Obrázek 6: Příklad výpočtu čísel *n(Q1)*, *n(Q3)* a *n(Qd)*.

Pro zlepšení kvality ohodnocovací funkce je v programu implementován takový skenovací algoritmus, který bere v úvahu také protihráčovy kameny při počítání výskytu jednotlivých figur. Dále se při výpočtu ohodnocení desky počítají Eulerova čísla *E* zvlášť pro černé kameny a zvlášť pro bílé kameny. Hrany hrací desky jsou vždy počítány jako prázdné.

Ohodnocení hrací desky je pak vypočteno podle vzorce:

$$V = \min(\max(\alpha f_1 + \beta f_2 + \gamma f_3, -\delta), \delta) + \varepsilon f_4$$

Konstanty α, β, γ, δ, ε ve formuli pro výpočet funkce *V* jsou konstanty, které uživatel může změnit v okně *Nastavení statické funkce*.

Hodnota f_1 je počet svobod 1. řádu pro černé kameny mínus počet svobod 1. řádu pro bílé kameny.

Hodnota f_2 je počet svobod 2. řádu pro černé kameny mínus počet svobod 2. řádu pro bílé kameny.

.

Hodnota f_3 je počet svobod 3. řádu pro černé kameny mínus počet svobod 3. řádu pro bílé kameny.

Nakonec hodnota f_4 je číslo Eulerovo číslo *E* pro černé kameny mínus Eulerovo číslo *E* pro bílé kameny.

Navržená ohodnocovací funkce počítá hodnotu hrací desky pro černé kameny. Pro výpočet ohodnocení pro bílé kameny se jednoduše neguje znaménko ohodnocení.

4.8. Konstanty ohodnocovací funkce

Během vývoje a ladění programu se z vyzkoušených hodnot konstant nejlépe osvědčila tato kombinace hodnot: α = 1; β = 1; γ = 0.75; δ = 3.45; ε = -4 při použití algoritmu Alpha – Beta s hloubkou prohledávání 3.

Nastavením konstant lze docílit znatelného zlepšení (nebo také zhoršení kvality) hry počítačového hráče. I přesto počítač stále zůstává porazitelný středně pokročilým hráčem hry Go.

Pro výrazné zlepšení kvality hry počítačového hráče by bylo nutné provést další vylepšení ohodnocovací funkce, spočívající například v hodnocení hrací desky podle větší množiny rysů než je nyní na desce sledována společně s úpravou prohledávacího algoritmu herního stromu o řazení tahů a heuristiku tahů (v angličtině známé pod označením Killer Heuristic, příp. History Heuristic). Tato rozšíření do programu *Atarigo* implementována nebyla.

5. Algoritmus hry počítačového hráče v programu *Atarigo*

Nyní, kdy už máme specifikovánu použitou ohodnocovací funkci a popsány algoritmy pro prohledávání herního stromu, můžeme popsat algoritmus, jakým je v programu vybírán tah pro počítačového hráče.

Tah počítače je vybírán ve třech krocích v následujícím pořadí kroků:

I. *Pravidlo útoku*

II. *Pravidlo obrany*

III. *Prohledávání s heuristikou*

V prvním kroku Pravidlo útoku se ověří, zdali se někde na hrací desce nachází takový volný průsečík, který by nám jedním tahem umožnil okamžité zajmutí soupeřovy skupiny kamenů. Pokud se na desce takový průsečík nachází, pak jsme schopni vyhrát tuto partii hry Atari Go.

Ve druhém kroku se kontroluje Pravidlo obrany, které se aplikuje, jestliže na desce nenalezneme takový průsečík, který by splňoval Pravidlo útoku. Obrana je inverzní akcí k útoku, proto se ve druhém kroku snažíme najít takový volný průsečík na hrací desce, který by soupeři na jeden tah umožnil zajmutí našich kamenů. Na tento průsečík musíme zahrát, abychom zamezili výhře soupeře.

Aplikací výše zmíněných dvou pravidel tak můžeme získat souřadnice průsečíku na hrací desce, kde položením svého kamenu vyhrajeme partii, nebo souřadnice, kde položením kamenu musíme bránit výhře soupeře. Tahem na získané souřadnice dokážeme okamžitě vyhrát partii, nebo zamezit (resp. oddálit) prohru. Tato pravidla však nedokáží vidět dopředu a hrát hru cíleně na výhru. Pohledu dopředu je docíleno teprve ve třetím kroku, prohledávání s heuristikou.

Jestliže v prvním ani druhém kroku (Pravidlo útoku a Pravidlo obrany) nový tah nenajdeme, pak se v posledním třetím kroku použije prohledávání herního stromu. Podle nastavení počítačového hráče se herní strom prohledává buď algoritmem Greedy, anebo sofistikovanějším algoritmem Alpha – Beta do určené hloubky. Hloubku prohledávání při

algoritmu Alpha – Beta je možné uživatelsky nastavit[9]. Oba algoritmy se rozhodují na základě heuristického ohodnocení hrací desky. Ohodnocení hrací desky v listech herního stromu je dáno statickou ohodnocovací funkcí.

Algoritmus Greedy se ve třetím kroku rozhoduje na základě ohodnocení hrací desky získaného pro každý svůj povolený tah. Ze získané množiny ohodnocení hrací desky pak vybere ohodnocení s největší hodnotou (resp. s nejmenší hodnotou pokud hrajeme za hráče s bílými kameny) a vykoná tah, který přísluší vybranému ohodnocení hrací desky.

Algoritmus Alpha – Beta dokáže simulovat hru obou hráčů od kořene herního stromu až do uživatelem nastavené hloubky prohledávání. Na základě heuristického ohodnocení hrací desky v dané hloubce s přihlédnutím k barvě hráčových kamenů pak algoritmus propaguje nalezený nejlepší tah do kořene herního stromu.

Maximálně po třetím kroku tedy algoritmus vrátí souřadnice pozice na hrací desce, kterou vyhodnotil jako nejlepší tah pro počítačového hráče.

Při použití pouze samotného prohledávání s heuristikou počítač několikrát zaváhal a nezahrál na průsečík, kde mohl partii okamžitě vyhrát. Také se často nedokázal účinně bránit a hrál na průsečíky, které mu vstříc hrozicí prohře vůbec nepomohly. Proto je prohledávání s heuristikou až posledním krokem algoritmu pro výpočet nejlepšího tahu.

První dva kroky lze s výhodou vypočítat najednou. Během každé partie se počítají svobody pro každý kámen položený na hrací desce zvlášť. Hledání vhodných průsečíků v prvním a druhém kroku se tak může omezit na najití právě těch kamenů, které mají jednu svobodu. Pro každý takový kámen zkusíme dosadit na jeho poslední svobodu. Jestliže dokážeme kámen (který může být součástí skupinu kamenů; pak musíme ověřit celou tuto skupinu) zajmout, pak dokážeme partii vyhrát jedním tahem právě na tento průsečík. Pak stačí na všechny takto ověřované průsečíky v prvním kroku dosadit protihráčův kámen a ověřit, jestli by při takovém tahu dokázal protihráč zajmout naši skupinu kamenů.

[9] Hloubku prohledávání lze nastavit v rozmezí 1-5. Hloubka 0 není povolena. Rozumných časů odezvy programu se dosahuje při hloubkách 1-3. Při hloubce prohledávání větší než 3 je nutné počítat s delší dobou odezvy programu, zvláště na velkých deskách!

6. Shrnutí bakalářské práce

Na závěr této práce bych čtenáři chtěl poděkoval za jeho zájem o hry Go a Atari Go a také za čas, který věnoval čtení této práce. Věřím, že tato práce byla pro čtenáře přínosem a představila mu obě hry a program *Atarigo* v dobrém světle.

Na prezentovaném programu *Atarigo* by bylo možné provést některá vylepšení, především pak vylepšit kvalitu hry počítačového hráče. Vhodným rozšířením této práce by bylo zaměřit se na ohodnocovací funkci a navrhnout kvalitnější heuristiku pro ohodnocovací funkci. Jinou zajímavou možností rozšíření programu by bylo použití algoritmů typu Monte Carlo pro nalezení nejlepšího tahu.

Cílem této práce však bylo vytvořit prostředí pro hru Atari Go na počítači a seznámit čtenáře se hrou Atari Go pro naučení základních pravidel a technik hry. Tato práce tak splnila svůj cíl.

7. Seznam literatury a zdrojů

[1] **Grosser, Tomáš.** *První kniha o Go.* Toužim : vydavatelství Tiger A.P.E., 2004.

[2] **Werf, E.C.D. van der; Herik, H.J. van den; Uiterwijk, J.W.H.M.** *Solving Go on Small Boards.* Amsterdam : vydavatelství Universiteit van Amsterdam, 2003.

[3] **Werf, E.C.D. van der; Herik, H.J. van den; Uiterwijk, J.W.H.M.** *Learning Connectedness in Binary Images.* Amsterdam : vydavatelství Universiteit van Amsterdam, 2001.

[4] **Werf, E.C.D. van der.** *AI Techniques for the Game of Go.* Maastricht : vydavatelství Universiteit Maastricht, 2004.

[5] **Hollosi, Arno; Morten, Pahle.** *Sensei´s Library.* [Web] 2008. http://senseis.xmp.net/.

[6] **Zajic Garko.** *About Delphi Programmming.* [Web] 2008. http://delphi.about.com/.

[7] **Peresada, Maxim.** *Torry`s Delphi Pages.* [Web] 2008. http://www.torry.net/.

[8] **Werf, E.C.D. van der.** *Homepage of Erik van der Werf.* [Web] 2008, http://erikvanderwerf.tengen.nl/.

8. Přílohy

8.1. Program *Atarigo* včetně zdrojových kódů programu na CD

Program Atarigo ve formě instalačního souboru včetně zdrojových kódů programu a tohoto textu ve formátu PDF naleznete na CD přiloženém k této práci.

8.2. Slovníček pojmů ve hře Go

Ve hře Go se používá celá řada odborných výrazů, které převážně pochází z japonštiny. Přikládám proto k práci malý slovníče, ve kterém se pokusím shrnout základní pojmy hry Go.

Znalost těchto pojmů je výhodou při komunikaci s profesionálními hráči Go, při rozborech hry nebo při hraní na Internetu. V závorce lze nalézt přepis původních japonských výrazů do angličtiny, pokud takový existuje.

Adži (Aji)
: V překladu "chuť". Možnost, potenciál do budoucna.

Adži-keši (Aji-keshi)
: Zrušení si svého adži.

Atari
: Hrozba na zajmutí jednoho nebo více kamenů soupeřem.

Bóši (Boshi)
: Zaútočení na kameny soupeře seshora. Klobouk.

Cuke (Tsuke)
: Přiložení kamenu ke kamenu soupeře.

Cume-go (Tsume-go)
: Úlohy na ožívání a umrtvování skupin kamenů.

Dame
: Svoboda nebo neutrální bod.

Dame-zumari
: Nedostatek svobod.

Dan
: Mistrovská třída, původně v japonských sportech a hrách. Ve hře Go jsou danové stupně rozděleny na amaterské (od 1. do 8. danu) a profesionální (od 1. do 9. danu).

Dango
: Neefektivní formace kamenů na desce. Hráč ma na jednom místě desky příliš mnoho kamenů, které mu nic nepřinášejí.

Džigo (Jigo)
: Remíza.

Džoseki (Joseki)
: Ustálená sekvence tahů, která oběma hráčům dává vyrovnaný výsledek.

Furikawari
: Výměna.

Fuseki
: Zahájení

Geta
Hnízdo. Taktika na chycení kamene nebo skupiny kamenů.
Goban
Hrací deska.
Gote
Tah, na který soupeř nemusí odpovědet. Viz Sente.
Handicap
Znevýhodnění silnějšího hráče. Slabší hráč si může na hrací desku položit několik svých kamenů před zahájením hry.
Hane
Ohnutí přes soupeřův kámen.
Hane-cugi (Hane-tsugi)
Ohnutí a spojení.
Hasami
Sevření.
Hnízdo
Viz *Geta*.
Honte
Pomalý a pevný tah, který nenechá v pozici žádné slabiny.
Hoši (hoshi)
Zvýrazněné body na gobanu pro pokládání handicapových kamenů.
Hvězda
Viz *Hoši*.
Ikken-tobi
Jednobodový skok.
Insei
Žák profesionální školy Go.
Jigo
Viz Džigo.
Jose (Yose)
Koncovka.
Joseki
Viz *Džoseki*.
Kakari
Napadení rohu. Viz. *Šimari*.
Keima
Koňský skok. Tah podobný tahu koně v šachách. Viz Ogeima.
Kikaši (Kikashi)
Nátlakový tah vyžadující odpověď soupeře.
Kó (Ko)
Pozice na desce, v níž by se hráči mohli donekonečna vybírat z desky. Jedno z pravidel hry Go hovoří o zákazu přímého opakovaní Kó pozice.
Komi
Kompenzace hráči s bílými kameny za první tah černého.
Kosumi
Diagonální tah.
Kjú (kyu)
Žákovská třída. Původně v japonských sportech a hrách. V Go je nejslabší 20. kjú a nejsilnější 1. kjú. Následující třídy pokračují stupni dan.

Miai

Záměnné body. Jestliže hráč obsadí jeden z bodů, může soupeř jít na druhý z bodů a obráceně.

Mojo (Moyo)

Náznak velkého území.

Nidan-bane

Dvojité ohnutí. Viz *hane*.

Nigiri

Losování barvy kamenů.

Nobi

Natažení.

Nozoki

Nahlédnutí, hrozba rozdělení.

Ógeima (Ogeima)

Velký koňský skok. Viz. *Keima*.

Ponnuki

Tvar na hrací desce po sebrání jednoho kamene.

Prázdný trojúhleník

Nejznámější špatný tvar kamenů. V naprosté většině případů je tento tvar velmi neefektivní.

San-ren-sei

Často hraná zahajovací formace. V překladu znamená „tři hvězdy v řadě“.

Seki

Tvar, v němž ani jeden hráč nemůže dobrat druhého.

Semeai

Souboj dvou skupin, který končí zničením jedné z nich.

Sente

Tah, na který soupeř musí odpovědět. Viz. *Gote*.

Schody

Posloupnost tahů na obklíčení soupeřovy skupiny. Hráč záměrně klade kameny z obou stran soupeřovy skupiny, nechává ho v atari a vede ho na stěnu, kde ho snadno zajme.

Skvíz (Squezze)

V překladu znamená vymačkání. Těsný útok na kameny soupeře, při němž hráč tyto kameny zajme nebo ohraničí na uzavřeném prostoru.

Šinogi (Shinogi)

Zachraňování slabých skupin, které jsou pod tlakem soupeře.

Tasuki

Zahájení, při němž hráči obsadí rohy diagonálně.

Tengen

Bod uprostřed desky.

Tenuki

Ignorování soupeřova tahu a zahrání do jiné části desky. Opuštění situace. Používá se v souvislosti, kdy se dala čekat odpověď na soupeřův tah.

Tesudži (Tesuji)

Chytrý tah v lokálním boji, při němž hráč například zajme soupeřovy kameny, zachrání své kameny či propojí své slabé skupiny apod.

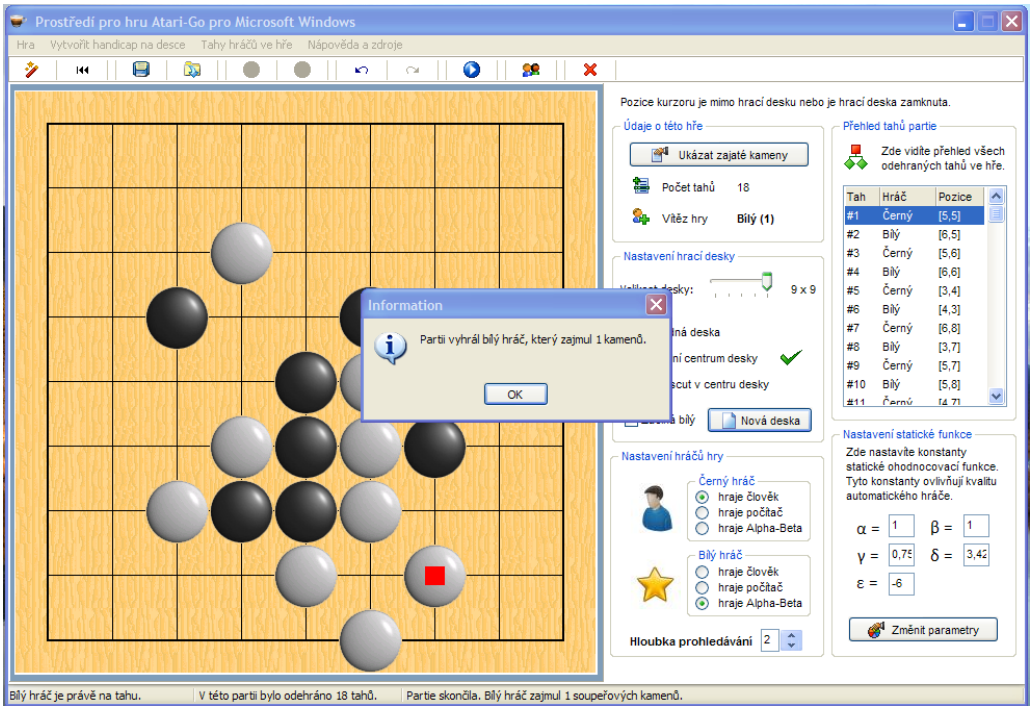
Tobi

Skok.

Učikomi (Uchikomi)
Invaze.

8.3. Ukázka prostředí programu

Obrázek 1 Hlavní okno programu po startu.



Obrázek 2 Okno programu na konci partie. Partii vyhrál hráč s bílými kameny, který zajmul 1 soupeřův kámen.