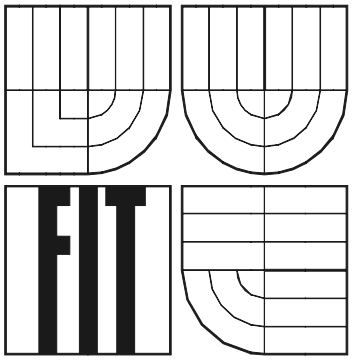
Vysoké učení technické v Brně

Fakulta informačních technologií



Applet pro hraní hry Go

Bakalářská práce

Brno 2007 **Jan Dosoudil**

Zadání

Applet pro hraní hry Go

1. Seznamte se s pravidly hry Go a naučte se ji hrát.
2. Prostudujte nástroje pro tvorbu appletů.
3. Navrhněte, implementujte a otestujte applet umožňující hrát Go dvěma hráčům prostřednictvím internetového prohlížeče.
4. Applet bude kontrolovat přípustnost jednotlivých tahů, počítat výsledek partie a bude také poskytovat rozhraní pro hraní hry Go počítačem.
5. Zhodnoťte dosažené výsledky a diskutujte možnosti dalšího vývoje.

Applet pro hraní hry Go

Odevzdáno na Fakultě informačních technologií Vysokého učení technického v Brně dne 2. 2. 2007

© Jan Dosoudil 2007

Autor práce tímto převádí svá práva na reprodukci a distribuci kopií celého díla i jeho částí na Vysoké učení technické v Brně, Fakultu informačních technologií.

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Bohuslava Křeny, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Jan Dosoudil
V Brně 2. 2. 2007

Poděkování

Děkuji Ing. Bohuslavu Křenovi za trpělivé vedení při zpracovávání této bakalářské práce.

Děkuji také svým rodičům za podporu při studiu.

Abstrakt

Tato bakalářská práce se zabývá tvorbou appletu v jazyce Java, umožňujícího hraní deskové hry Go pomocí počítače a také hru dvou hráčů přes Internet. Cílem práce však není vytvořit applet, který by pomocí vlastní inteligence umožňoval tuto hru. Bylo by to příliš složité, protože o hře go se tvrdí, že je nejobtížněji naprogramovatelnou deskovou hrou. Proto bude applet ke své činnosti využívat již existující Go program (engine) s implementovanou umělou inteligencí, se kterým bude komunikovat pomocí tzv. Go Text Protokolu (GTP).

Protože hra Go není u nás příliš známá, je čtenář nejprve seznámen s její historií a hlavně pravidly.

Klíčová slova

Applet, Java, desková hra Go, Go Text Protocol (GTP)

Abstract

This bachelor thesis deals with development of a Java applet for playing the board game of Go against computer as well as for playing Go by two human players connected by the Internet. The applet does not implement artificial intelligence for ruling and playing the game, however, it exploits an external Go engine trough the Go Text Protocol (GTP). The game of Go is probably the most complex board game and implementation of the engine is out of the scope of this thesis.

The game of Go is not well know in Europe, so, there is given a brief introduction to history and rules of Go for the reader at the beginning of the thesis.

Key Words

Applet, Java, board game Go, Go Text Protocol (GTP)

Obsah

1 Úvod....6
2 Hra Go....7
2.1 Z historie Go....7
2.2 Pravidla Go....7
3 O programech hrajících GO....16
3.1 Druhy Go programů....16
3.2 Komunikační protokoly....18
4 Tvorba appletu....23
4.1 Nástroje pro tvorbu appletů....23
4.2 Návrh appletu....23
4.3 Implementace appletu....24
5 Závěr....29
Literatura....30

1 Úvod

Tato bakalářská práce se zabývá tvorbou appletu v jazyce Java, umožňujícího hraní deskové hry Go pomocí počítače a také hru dvou hráčů přes Internet. Cílem práce však není vytvořit applet, který by pomocí vlastní inteligence umožňoval tuto hru. Bylo by to příliš složité, protože o hře go se tvrdí, že je nejobtížněji naprogramovatelnou deskovou hrou. Proto bude applet ke své činnosti využívat již existující Go program (engine) s implementovanou umělou inteligencí, se kterým bude komunikovat pomocí tzv. Go Text Protokolu (GTP).

Hrou Go se již zabývá několik světových technických universit. Problému se však věnují i skupiny nadšenců či jednotlivců a je samozřejmě řešen i na komerční úrovni. Ve světě se také konají turnaje Go programů.

Protože se i Ústav inteligentních systémů na naší fakultě zajímá o problematiku hraní Go počítačem, byla vypsána tato bakalářské práce. V souvislosti s tím byl vypsán i diplomový projekt „Hraní hry Go počítačem“, jehož výsledkem má být právě program obsahující umělou inteligenci pro hraní Go. Applet vytvářený v rámci této bakalářské práce by pak mohl sloužit jako uživatelské rozhraní k tomuto programu.

Ve druhé kapitole této práce se čtenář seznámí s historií a hlavně s pravidly hry Go samotné. Následující kapitola se pak věnuje právě Go programům – jejich rozdělení a hlavně způsobu vzájemné komunikace, který je zásadní pro tvorbu appletu. Čtvrtá kapitola je pak již věnována vlastnímu appletu. Jsou stručně rozebrány nástroje pro jeho tvorbu, jeho návrh a samotná implementace. A konečné pátá závěrečná kapitola shrnuje dosažené výsledky a možnosti dalšího vývoje.

2 Hra Go

Tato kapitola představuje hru Go jako takovou. Stručně je popsána historie této hry a podrobně jsou pak rozebrána její pravidla.

Zdrojem informací této kapitoly byla kniha Tomáše Grossera – První kniha o Go [1].

2.1 Z historie Go

Go vzniklo asi před čtyřmi až pěti tisíci let v Číně a je tak nejstarší deskovou hrou na světě. Grosser ve své knize uvádí dvě legendy, které se váží ke vzniku Go:

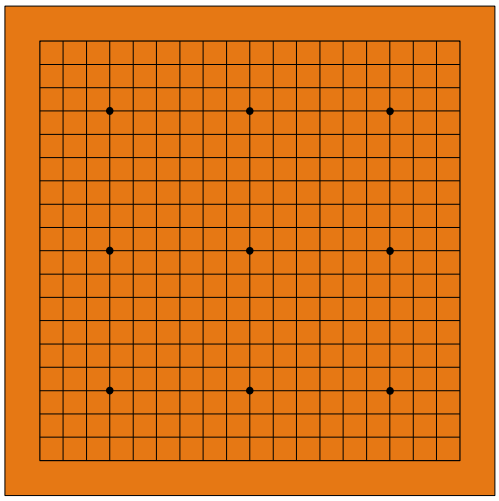
„O jejím vzniku koluje mnoho legend. Jedna z nich například vypráví, že čínský císař Jao nařídil svému rádci, aby vymyslel hru, která by pomohla rozvíjet myšlení jeho duševně zaostalému synovi. Další legenda je o dřevorubci, který vyrazil do hor na dříví a zde viděl dva starce, zřejmě bohy, hrající Go. Když starci dohráli, rozplynuli se a dřevorubec zjistil, že během partie uplynuly desítky let. Dost možná Go původně sloužilo k věštění.“

Asi v 7. století našeho letopočtu se Go dostalo přes Koreu do Japonska. To se stalo jeho hlavní základnou. Kolem roku 1600 zde byly položeny základy profesionálního Go. Došlo k tomu vznikem čtyř vzájemně mezi sebou soupeřících škol.

Do Evropy, a také do Ameriky, se Go začíná šířit až koncem 19. století.

2.2 Pravidla Go

Go je desková hra pro dva hráče – černého a bílého. Hrací deska (**goban**) vypadá jako čtvercová síť s různou velikostí (počtem průsečíků). Nejčastěji se používá velikost 19×19 průsečíků (Obr. 2.1), která je standardem pro profesionální utkání. Začínajícím hráčům jsou doporučeny desky menší, obvykle o velikosti 13×13 nebo 9×9. Partie na menších deskách je kratší a hráč také lépe udrží na menší desce pozornost.



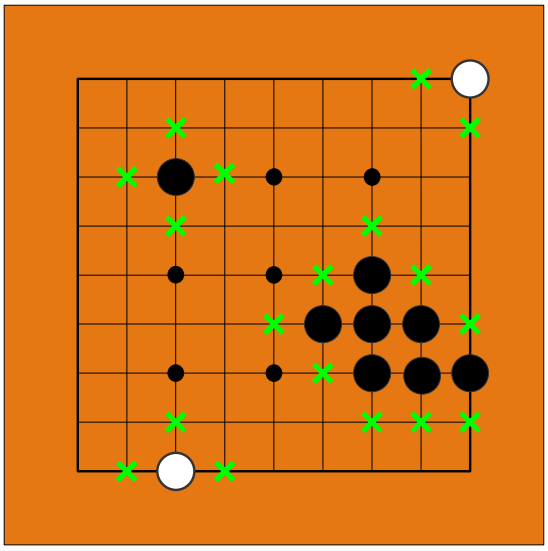
Obr. 2.1: Hrací deska 19×19

Hra se hraje s kameny, které hráči na desku střídavě pokládají (neposouvají se jako např. u šachů). Přičemž hru začíná černý hráč.

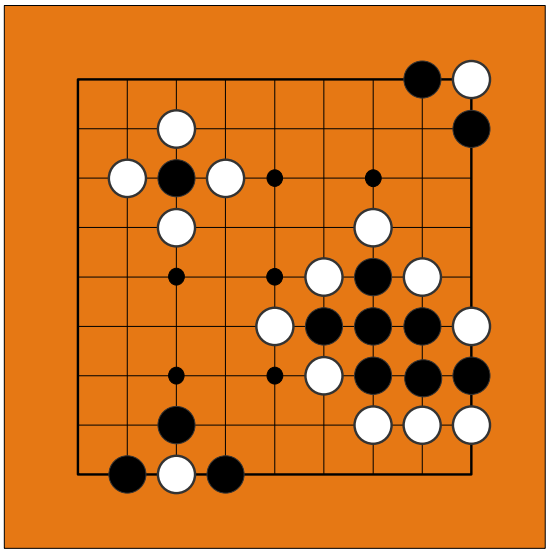
Cílem hry je zajmutí co nejvíce soupeřových kamenů a získání co největšího území, což bude popsáno dále.

2.2.1 Zajímání kamenů

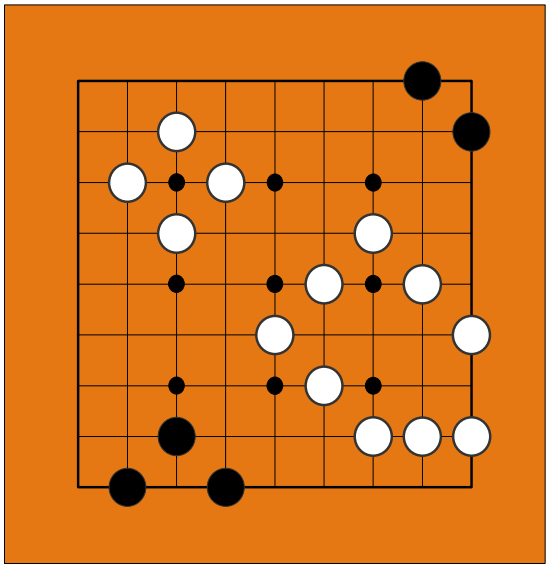
Každý kámen nebo skupina kamenů má tzv. **svobody** – volné sousední průsečíky (Obr. 2.2). Jestliže však dojde ke ztrátě všech těchto svobod (soupeři se podaří položit na volné průsečíky svoje kameny), je kámen soupeřem zajat (Obr. 2.3) a odstraněn z desky (Obr. 2.4). Každý zajatec přináší soupeři jeden bod.



Obr. 2.2: Svobody kamenů v různých místech desky a svobody skupiny kamenů.



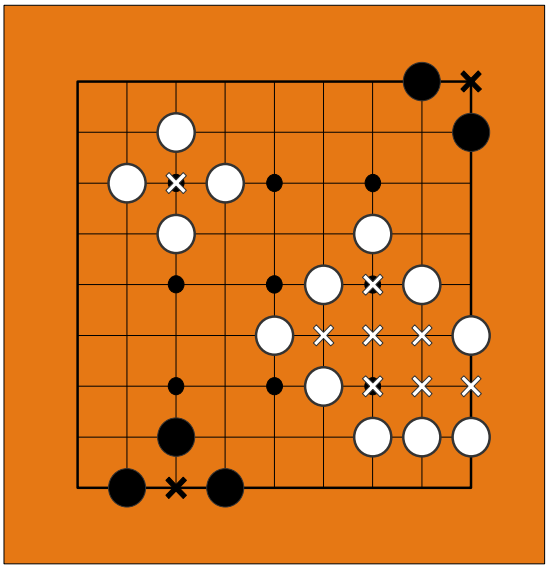
Obr. 2.3: Zajetí (obklíčení) kamenů soupeřem.



Obr. 2.4: Po odstranění zajatců…

2.2.2 Boj o území

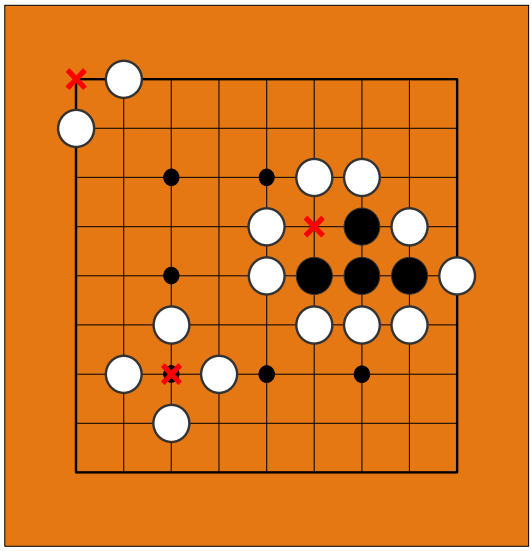
Dalším z cílů hry je ohraničení co největšího území svými kameny. Za každý volný ohraničený průsečík získává hráč jeden bod. Na Obr. 2.5 má tedy bílý ohraničeno celkem 8 průsečíků, zatímco černý jen 2 průsečíky.



Obr. 2.5: Ohraničení průsečíků

2.2.3 Zakázané tahy

V Go existují dvě situace, kdy tah není povolen:

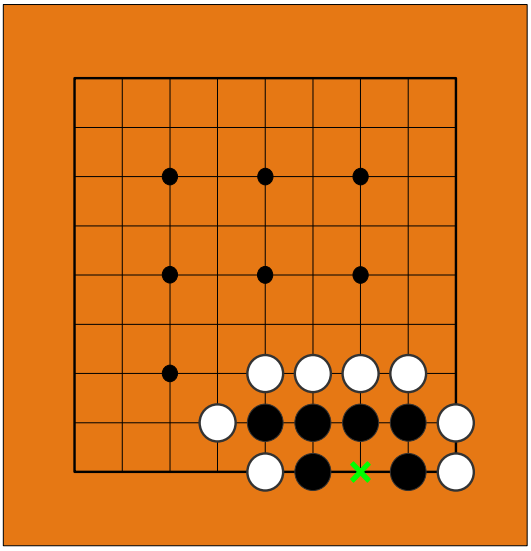


Obr. 2.6: Zákaz sebevraždy

1. **Sebevražda** – hráč nemůže položit kámen tak, aby tento kámen neměl ani jednu svobodu. Případně aby položením kamene do skupiny, tato skupina

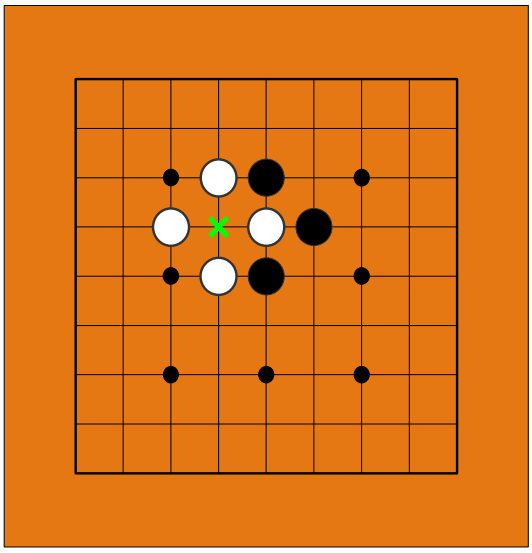
přišla o poslední volný průsečík (svobodu). Názorně to ukazuje Obr. 2.6. Černý nemůže zahrát tah ani na jedno z míst označených křížkem, neboť by to byla sebevražda.

Existuje však výjimka, kdy je takový tah povolen. Je to v případě, že položením kamene dojde k zajetí soupeřova kamene (kamenů), a tím pádem k uvolnění minimálně jednoho sousedního průsečíku (získání svobody). Na Obr. 2.7 může bílý položit kámen na místo křížku, protože tím dojde ke ztrátě všech svobod černých kamenů, jejich odstranění z desky a tím pádem získání svobod pro právě vložený bílý kámen. Černý by na místo křížku kámen položit nemohl.

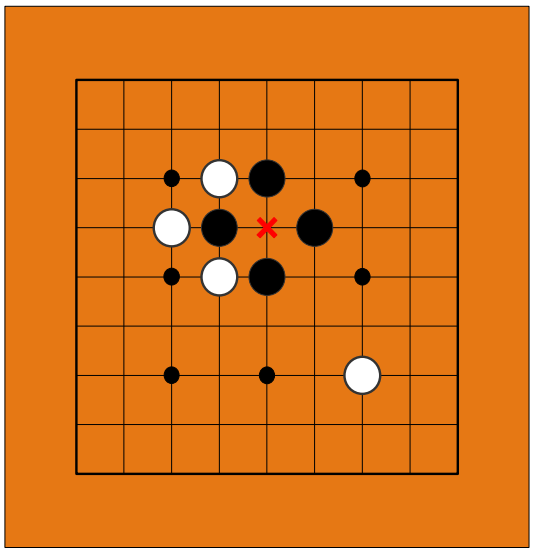


Obr. 2.7: Povolená sebevražda

2. **Kó** (nekonečno) – toto pravidlo ošetřuje situaci, kdy může dojít k donekonečna se opakujícímu použití výjimky z předchozího pravidla. Situaci opět

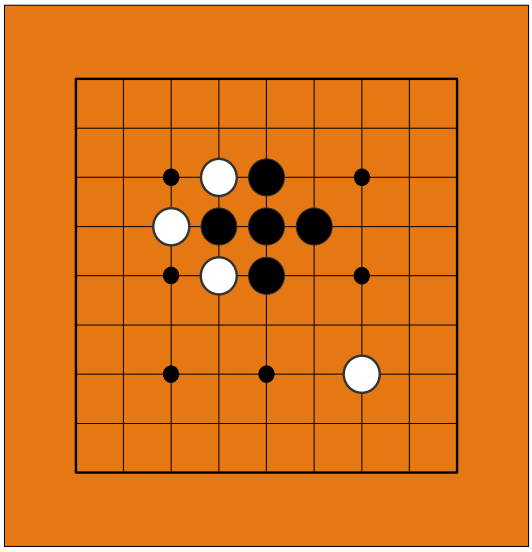


Obr. 2.8: Kó – černý může položit kámen na místo křížku…

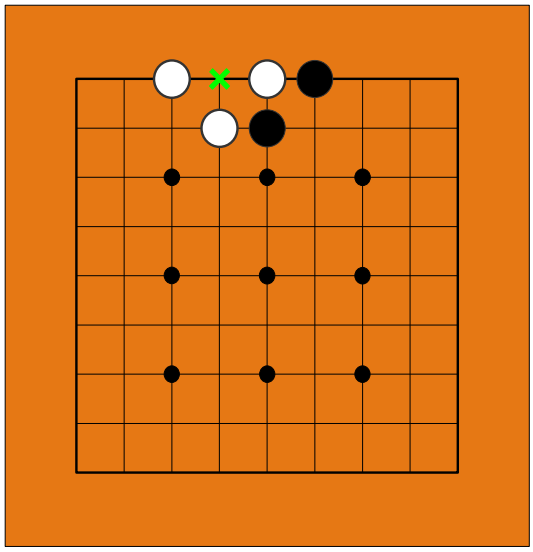


Obr. 2.9: Kó – bílý musí ale táhnout jinam…

přiblíží několik obrázků. Na Obr. 2.8 černý položí kámen na místo křížku a zajme tak bílý kámen. Bílý by nyní na Obr. 2.9 tahem na místo křížku zajal černý kámen a situace by se dostala zpět do stavu na Obr. 2.8. To však nelze, protože to zakazuje právě pravidlo Kó. Bílý proto musí v tomto případě položit kámen na jiný průsečík. To umožní černému ochránit svůj kámen tahem na místo křížku a ukončit tím nekonečnou smyčku (Obr. 2.10). Situace Kó může nastat i na kraji desky, jak ukazuje Obr. 2.11.



Obr. 2.10: Kó – a černý může ochránit svůj kámen



Obr. 2.11: Kó na kraji desky

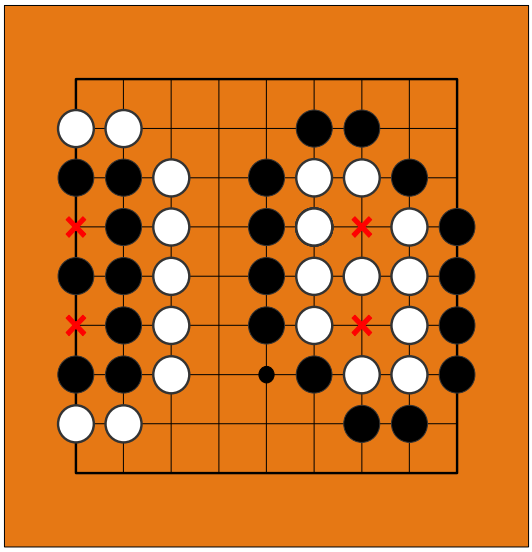
2.2.4 Živé a neživé skupiny

Jako **živé skupiny** jsou v Go označovány takové skupiny kamenů, které soupeř nemůže zajmout. Naopak jako **neživé skupiny** jsou označovány skupiny, které zajmout lze.

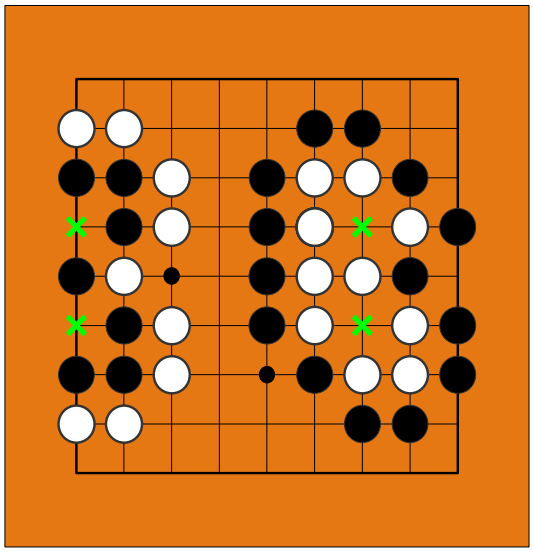
Na Obr. 2.12 jsou zobrazeny dvě živé skupiny obklíčené soupeřovými kameny. K tomu aby je soupeř mohl zajmout, by musel zahrát dva sebevražedné tahy na místa uvnitř skupiny označená křížky. To samozřejmě není možné a tyto skupiny zůstanou na desce do konce partie. Tato místa jsou označována jako **oči**. Pokud tedy skupina má dvě oči, nebo je může vytvořit (Obr. 2.14), jedná se o živou skupinu.

Naproti tomu modifikované skupiny na Obr. 2.13 jsou neživé a soupeř je může dvěma tahy zajmout. Skupiny sice na první pohled vypadají že, mají dvě oči, ale jedná se pouze o **oči nepravé**.

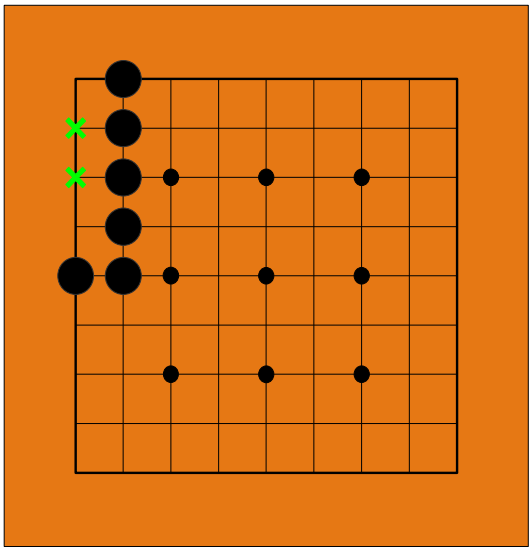
Zvláštní formou života je **seki**. Jedná se o situaci kdy ani jeden z hráčů nechce položit kámen do skupiny, protože by ho protihráč zajal. Tato situace je zobrazena na Obr. 2.15. Pokud jeden z hráčů položí kámen dovnitř skupiny, druhý hráč ho následujícím tahem zajme.



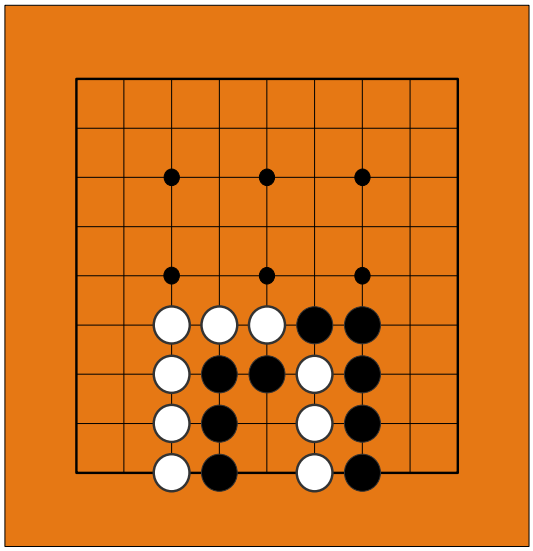
Obr. 2.12: Živé skupiny – mají dvě oči.



Obr. 2.13: Neživé skupiny – mají sice oči, ale nepravé.



Obr. 2.14: Skupina sice nemá dvě oči, ale má dost prostoru je při útoku vytvořit.



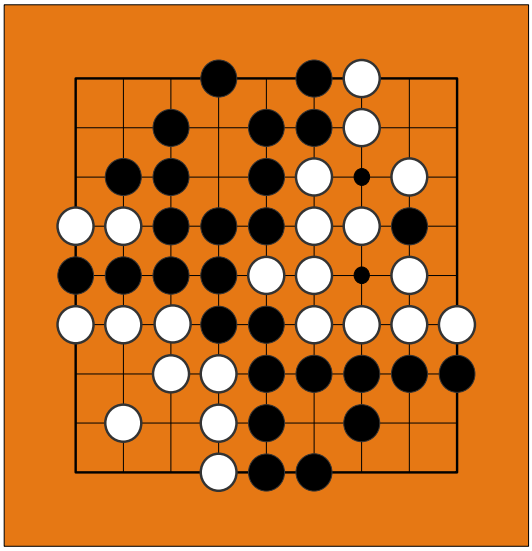
Obr. 2.15: Seki – ani jeden z hráčů nechce zahrát dovnitř skupiny.

2.2.5 Ukončení partie a počítání výsledku

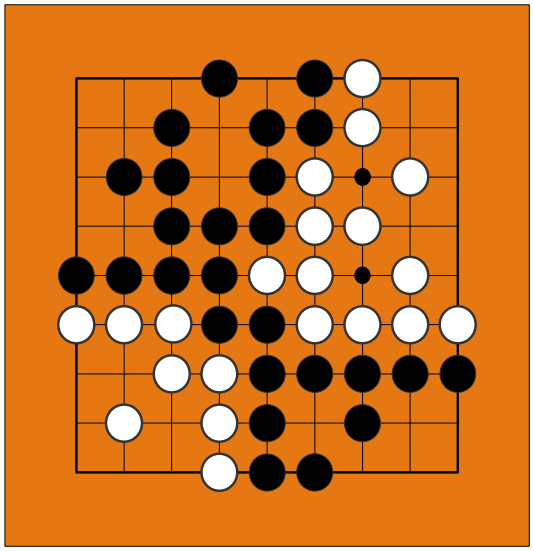
Pokud hráč nemůže již zahrát tah, který by mu přinesl zisk, řekne „pass“. Jestliže i druhý hráč nevidí možnost smysluplného tahu a řekne „pass“, partie končí. Partie Go tedy končí v okamžiku, kdy oba hráči bezprostředně po sobě zahrají (řeknou) **pass**.

Postup počítání výsledku partie bude demonstrován na modelovém příkladu. Stav po ukončení hry je na Obr. 2.16. Nejprve jsou z desky odstraněny neživé

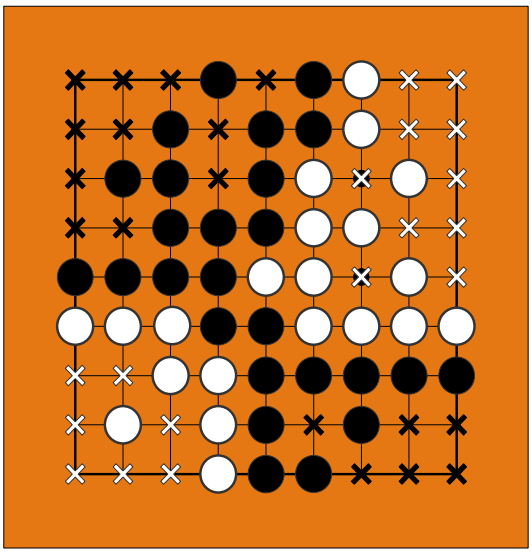
skupiny (zpravidla jen osamocené kameny), které jsou uvnitř soupeřova území a soupeř by je tedy mohl zajmout, ale zbytečně by se připravil o body (volné průsečíky území). Tyto kameny jsou přidány k zajatcům, které již soupeř má. Stav po odebrání kamenů je zobrazen na Obr. 2.17.



Obr. 2.16: Partie právě skončila



Obr. 2.17: Stav po odebrání neživých kamenů



Obr. 2.18: Počítání území

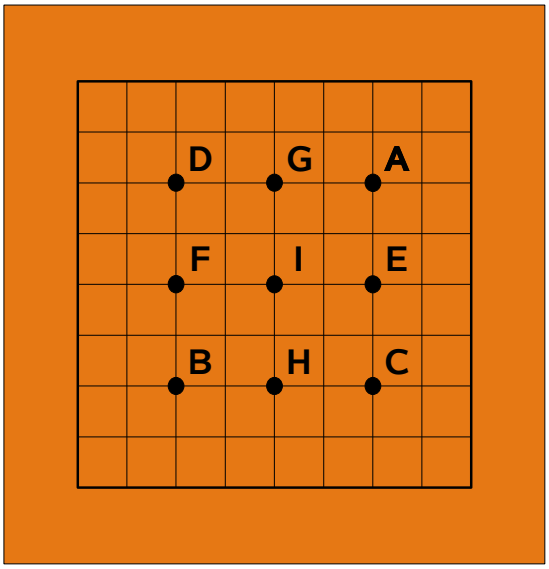
Poté jsou spočteny volné průsečíky na území ohraničeném kameny hráčů, jak ukazuje Obr. 2.18. Černý i bílý mají shodně 17 průsečíků (bodů). Aby byl výsledek konečný, je nutné ještě připočíst počty zajatců a komi.

Komi je číslo vyjadřující bodovou kompenzaci prvního tahu černého hráče bílému hráči. Tuto hodnotu si tedy bílý přičte ke svému konečnému výsledku. Standardní hodnota komi je 6,5 bodu.

Uvažujme, že černý získal během hry 3 zajatce a bílý 2 zajatce. K těmto zajatcům jsou ještě přidány neživé kameny odebrané z desky po skončení partie (viz Obr. 2.16), tj. černý má celkem 5 a bílý 3 zajatce. Po přičtení k bodovému zisku za území má černý 22 a bílý 20 bodů. Bílému je však výsledek korigován ještě hodnotou komi, takže bílý nakonec získá 26,5 bodu a partii vyhrává.

2.2.6 Hendikep

Posledním důležitým termínem týkajícím se Go, je hendikep (**handicap**). Jedná se o číslo zpravidla od dvou do devíti, které vyjadřuje počet černých kamenů, které jsou umístěny na hrací desku před započetím hry na vyznačené body (Obr. 2.19). To například umožňuje hráčům rozdílné síly (třídy[1]) sehrát zajímavou partii (slabší hráč hraje jako černý). Body jsou obsazovány podle klíče uvedeného v tabulce 2.1. Obr. 2.20 zobrazuje desku s kameny pro handicap rovný pěti.



Obr. 2.19: Body na desce, na které jsou pokládány kameny hendikepu.

Handicap	Body
2	A, B
3	A–C
4	A–D
5	A–D, I
6	A–F
7	A–F, I
8	A–H
9	A–I

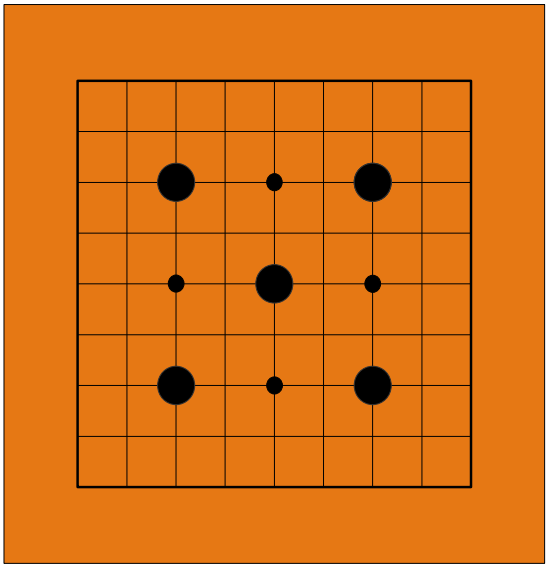
Tabulka 2.1: Pořadí pokládání hendikepu

Položení hendikepu je bráno jako první tah černého, po kterém následuje tah bílého.

Je možné použít i handicap rovný jedné. Černý začne hru běžným způsobem, ale na konci hry není bílému přičteno komi. Při hendikepu větším než 9 je černému po skončení hry připočteno za každý další kámen, který již není kam položit, pět bodů.

Někdy se také používá tzv. **free handicap** – kameny jsou pokládány kamkoli na hrací desce.

1 V Go jsou hráči označováni třídami, podobně jako v jiných asijských sportech. Zjednodušeně lze říci, že čím vyšší třídu hráč má, tím je silnější. Hendikep je pak dán rozdílem jejich tříd.



Obr. 2.20: Handicap 5

3 O programech hrajících GO

Nyní, když jsme seznámeni s pravidly hry Go, se můžeme zaměřit na počítačové programy, které jsou v této oblasti dostupné. Bude probráno jejich dělení podle účelu použití a představeni někteří jejich zástupci. Dále bude probrána komunikace těchto programů mezi sebou, jež je klíčová pro implementaci našeho appletu (viz následující kapitola).

3.1 Druhy Go programů

V současnosti existuje několik skupin (či jednotlivců) nadšenců zabývajících se tvorbou programů pro hraní Go. Některé z těchto skupin vznikly na technických universitách. Kromě toho existuje i několik komerčně vyvíjených programů.

Tyto programy můžeme rozdělit do několika druhů, podle účelu jejich použití.

3.1.1 Programy s inteligencí (engine)

Jedná se o programy implementující inteligenci potřebnou pro hraní Go. Jsou to zpravidla konzolové aplikace, poskytující rozhraní pro hraní Go z příkazového řádku. Mnohem důležitější však je, že tyto aplikace poskytují možnost komunikace s okolním světem pomocí minimálně jednoho ze dvou v současnosti používaných protokolů (bude probráno dále). To umožňuje programům tohoto typu např. hrát proti sobě, nebo komunikaci s grafickým uživatelským rozhraním.

Asi nejznámějším zástupcem těchto programů je GNU Go [2], jehož obrazovka s partií načtenou ze souboru SGF[2] je na Obr. 3.1.

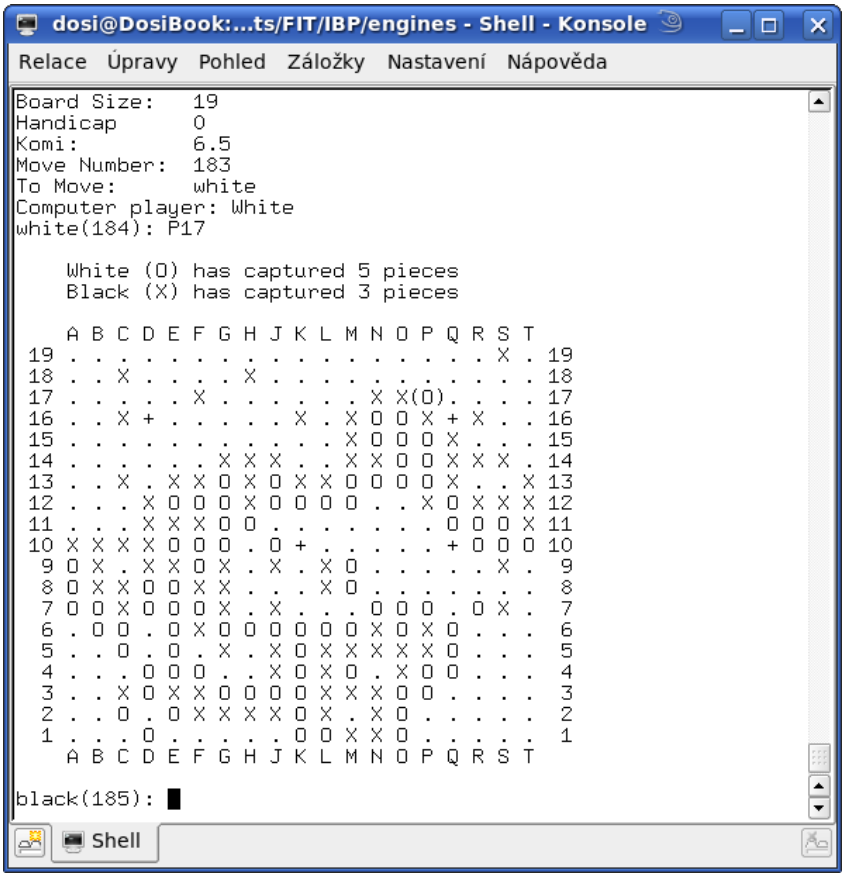
3.1.2 Uživatelská rozhraní (controller)

Jak nadpis napovídá, jde o programy sloužící jako grafická uživatelská rozhraní k programům typu engine. Bývá také možné se pomocí nich připojit ke Go serverům na Internetu. Přičemž komunikace probíhá pomocí dále popsaných protokolů.

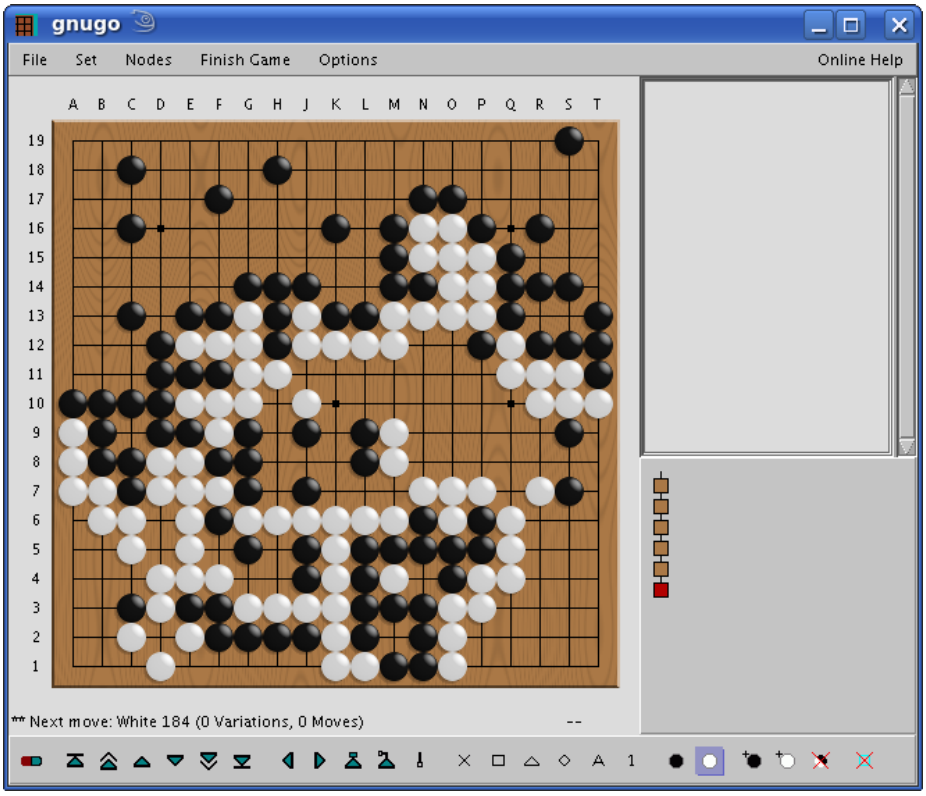
Tyto programy nemají zpravidla vlastní inteligenci, případně mají jen základní inteligenci potřebnou ke kontrole správnosti tahů.

Na Obr. 3.2 je zachycena obrazovka programu JaGo (Java Go)

2 Smart Game Format – využívá se pro ukládání záznamu mnoha různých her. [3]



Obr. 3.1: Obrazovka programu GNU Go (engine)



Obr. 3.2:Obrazovka programu JaGo (controller)

3.1.3 Ostatní programy

V oblasti hraní Go je možné setkat se ještě s následujícími specializovanými typy programů:

- Programy určené k řízení zápasů dvou engine – jedná se vlastně o kontroléry
- Klienti Go serverů
- Další specializované testovací programy, nebo programy používané ke konverzi komunikačních protokolů

3.2 Komunikační protokoly

Go programy mezi sebou komunikují pomocí následujících dvou protokolů:

- **Go modem protocol (GMP)** – Jedná se o standardizovaný, nicméně zastaralý sériový protokol. Je vhodný hlavně pro komunikaci programů pomocí sériové linky (modemu), nebo pro mezi-procesovou komunikaci (roury).
- **Go text protocol (GTP)** – Je mladší než GMP a zatím ne úplně standardizovaný, přesto však hojně používaný. Je vhodný pro komunikaci pomocí TCP/IP a samozřejmě pro mezi-procesovou komunikaci.

Nyní se na oba protokoly podíváme podrobněji.

3.2.1 Go modem protokol

Informace o GMP jsou čerpány z [4].

Komunikace mezi programy probíhá sériově pomocí čtyř bytů dat. První byte je hlavička zprávy, následovaná bytem kontrolního součtu a dvěma byty obsahujícími vlastní zprávu (příkaz):

```
0000 00hy 1sss ssss 1ccc rvvv 1vvv vvvv
```

hy – Jedná se o tzv. „his“ a „your“ bit. Kombinace těchto dvou bitů tvoří identifikátor zprávy a je pro každou novou zprávu změněna (převrácena).
s – bity kontrolního součtu zprávy
c – bity identifikátoru příkazu
r – rezervovaný (nepoužitý) bit – jeho hodnota je 0
v – bity hodnoty zprávy

Vysílání každé zprávy je opakováno, dokud není obdržena odpověď od druhého programu.

Go modem protokol má osm příkazů (tři bity c), které velmi stručně přiblížíme:

0. **OK** – obdržení a provedení zaslaného příkazu

1. **DENY** – obdržený příkaz nebylo možno provést
2. **NEWGAME** – začíná novou hru
3. **Query** – slouží ke zjištění parametrů hry a komunikace
4. **Answer** – odpověď na Query
5. **Move** – tah na určitý průsečík na desce, udaný číslem v rozmezí 1–361 (pro desku 19×19)
6. **Take back move** – vrátí zpět zadaný počet tahů
7. **Extended** – umožňuje zasílání delších zpráv (zpráva má více bytů). Je využit např. ke komunikaci hráčů pomocí textových zpráv (chat).

3.2.2 Go text protokol

Uvedené informace jsou převzaty ze specifikace GTP [5] a z dokumentace GNU Go [6]. Protože GTP byl nakonec vybrán pro implementaci v appletu, seznámíme se s ním poněkud podrobněji, než s předchozím protokolem.

Komunikace probíhá pomocí textových zpráv (příkazů), které je možno jednoduše posílat pomocí TCP/IP, nebo mezi-procesově pomocí rour.

Formát zpráv

Formát příkazu protokolu je:

```
[id] command_name [arguments]
```

- **`id`** – volitelný nezáporný celočíselný identifikátor příkazu. Používán hlavně při hromadném zpracovávání více zpráv najednou. Umožňuje přiřazení odpovědi (odezvy) k odpovídajícímu příkazu.
- **`command_name`** – vlastní příkaz
- **`arguments`** – parametry příkazu, pokud je příkaz vyžaduje

Formát kladné odpovědi na příkaz je pak:

```
=[id] result
```

V případě výskytu chyby je odpověď:

```
?[id] error_message
```

- **`id`** – vyskytuje se jen v případě, že příkaz, ke kterému náleží odpověď, byl označen identifikátorem. V takovém případě je pak tento identifikátor uveden i v odpovědi.
- **`result`** – výsledek příkazu, pokud příkaz nějaký vrací
- **`error_message`** – zpráva s popisem chyby

Každý příkaz nebo odpověď na něj jsou ukončeny dvěma novými řádky.

Datové typy

Protokol používá několik řetězcových entit (datových typů) k reprezentaci různých typů hodnot použitých v parametru příkazu a v odpovědi na něj. Jsou to následující:

- **`int`** – nezáporné celé číslo
- **`float`** – reálné číslo
- **`string`** – libovolný řetězec tisknutelných znaků s rozlišením malých a velkých písmen
- **`vertex`** – reprezentuje souřadnici průsečíku na desce. Jedná se o kombinaci písmene (řádku) a číslice (sloupce), což odpovídá běžnému značení používanému ve hře Go. Z písmen je vyjmuto `"I"`, přičemž nezáleží na jejich velikosti. Pro desku o velikosti 19×19 jsou tedy souřadnice v rozsahu A1–T19. Protokol však podporuje desky až do velikosti 25×25. Kromě souřadnice však může být hodnota `vertex` také `"pass"`.
- **`color`** – barva kamene, reprezentovaná slovy `"white"` a `"black"` nebo znaky `"w"` a `"b"`. Opět nezáleží na velikosti písmen.
- **`move`** – jedná se o kombinaci barvy a souřadnice (`color` a `vertex`) vzájemně oddělených mezerou. Např.: `"white h10"`, `"B F5"`, `"w pass"`.
- **`boolean`** – logická hodnota reprezentovaná řetězcem `"true"` nebo `"false"`.

Dále jsou definovány čtyři složené entity:

- **`Collection`** (`{x y}`) – hodnoty (jednoduché entity) oddělené mezerou. Může být použito i více než dvě hodnoty.
- **`List`** (`x*`) – seznam jednoduchých entit oddělených mezerou
- **`Alternatives`** (`x|y`) – jedna z entit
- **`Multiline list`** (`x&`) – víceřádkový seznam (ukončený dvěma znaky nového řádku)

Vybrané příkazy

Protokol má celkem 17 příkazů, které jsou definovány v jeho specifikaci. Většina programů je však rozšířena o mnohé vlastní příkazy (např. zmiňované GNU Go obsahuje přes sto vlastních příkazů). Aby bylo možné dále demonstrovat komunikaci pomocí GTP, bude nyní přiblíženo 11 základních příkazů GTP, které jsou vyžadovány v jakékoli implementaci:

- **`protocol_version`**
  - získá verzi protokolu
  - výsledek: `int version_number`
- **`name`**
  - získá jméno aplikace

  - výsledek: `string* name`
- **`version`**
  - získá verzi programu
  - výsledek: `string* version`
- **`known_command`** `<string command_name>`
  - umožňuje zjistit, zda program podporuje příkaz uvedený jako parametr.
  - výsledek: `boolean known` (`"true"` pokud je příkaz podporován)
- **`list_commands`**
  - získá jména všech známých příkazů
  - výsledek: `string& commands` (víceřádkový seznam příkazů)
- **`quit`**
  - ukončí komunikaci
- **`boardsize`** `<int size>`
  - nastaví velikosti hrací desky
  - chyby: `"unacceptable size"` (pokud program nemůže zpracovat zadanou velikost)
- **`clear_board`**
  - připraví desku pro novou hru (vyčistí ji)
- **`komi`** `<float new_komi>`
  - nastaví velikost komi
- **`play`** `<move move>`
  - položí kámen zadané barvy na zadanou souřadnici na desce
  - chyby: `"illegal move"` (pokud není možné tah provést)
- **`genmove`** `<color color>`
  - požadavek na vygenerování (pomocí umělé inteligence) pozice pro položení kamene zadané barvy
  - výsledek: `vertex|string` (`vertex`: vygenerovaná pozice kamene, příp. `"pass"`; `string`: pokud engine není schopno vygenerovat pozici, může rezignovat zasláním `"resign"`)

Pokud program neimplementuje zaslaný příkaz (u výše uvedených příkazů se to stát nemělo), zareaguje chybou `"unknown command".` V případě chybné syntaxe parametrů dojde k zaslání chyby `"syntax error"`.

Příklad komunikace

Na následujících řádcích je uveden příklad komunikace pomocí GTP mezi engine[3] a kontrolérem.

```
protocol_version
= 2
```

3 GNU Go spuštěné v režimu GTP. Čtenář si může komunikaci vyzkoušet spuštěním pomocí příkazu: `gnugo --mode gtp`

```
name
= GNU Go

version
= 3.7.2

known_command kvit
= false

known_command quit
= true

list_commands
= aa_confirm_safety
accurate_approxlib
     ⋮                    (vynecháno)
worm_data
worm_stones


boardsize 9
=

clear_board
=

play B C3
=

genmove W
= F6

play B G8
=

genmove W
= F8

play B E5
=

genmove W
= G7
```

```
play B G7
? illegal move

play B H7
=

genmove W
= H6

showboard[4]
=
   A B C D E F G H J
 9 . . . . . . . . . 9
 8 . . . . . O X . . 8
 7 . . + . . . O X . 7
 6 . . . . . O . O . 6
 5 . . . . X . . . . 5
 4 . . . . . . . . . 4
 3 . . X . . . + . . 3
 2 . . . . . . . . . 2     WHITE (O) has captured 0 stones
 1 . . . . . . . . . 1     BLACK (X) has captured 0 stones
   A B C D E F G H J

quit
=
```

4 `showboard` – ladící příkaz, slouží ke znakovému vykreslení desky

4 Tvorba appletu

Tato kapitola pojednává o appletu samotném. Nejprve jsou stručně rozebrány prostředky, pomocí kterých je možné applety v současnosti vytvářet. Dále je proveden návrh appletu, následovaný stručným popisem dosažené implementace.

4.1 Nástroje pro tvorbu appletů

Existuje několik v současnosti používaných prostředků pro tvorbu appletů jako například Adobe Flash nebo Microsoft .NET. Asi nejrozšířenější je však Java od Sun Microsystems. Mezi její další výhody patří přenositelnost, podpora ve všech hlavních webových prohlížečích a robustnost.

Proto jsme se s vedoucím práce rozhodli právě pro Javu, jako prostředí pro implementaci našeho appletu.

Jako prostředek k samotné implementaci jsem pak vybral integrované vývojové prostředí (IDE) rovněž od Sunu – Java Studio Enterprise 8.1. Toto prostředí je volně dostupné (po registraci) pro nekomerční použití na www stránkách firmy Sun [7].

4.2 Návrh appletu

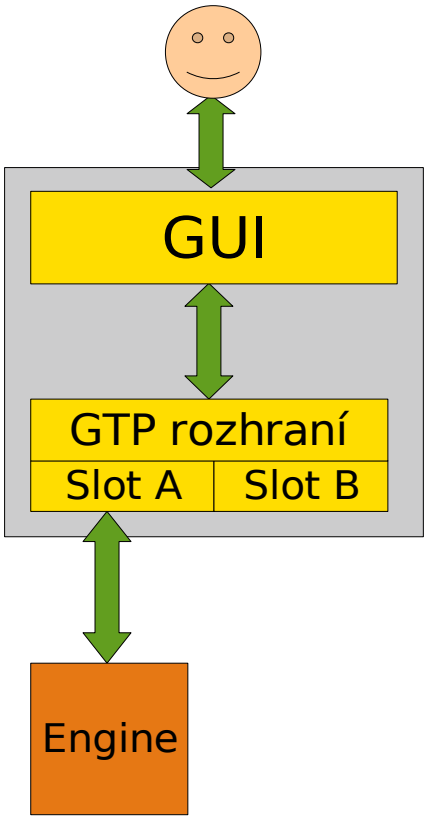
Při návrhu appletu je nutné vycházet ze zadání, to požaduje následující funkce:

- rozhraní pro hraní hry Go, umožňující hru dvou hráčů pomocí internetového prohlížeče
- rozhraní pro hraní hry Go počítačem (člověk vs Go program)
- kontrolu přípustnosti jednotlivých tahů
- počítání výsledku partie

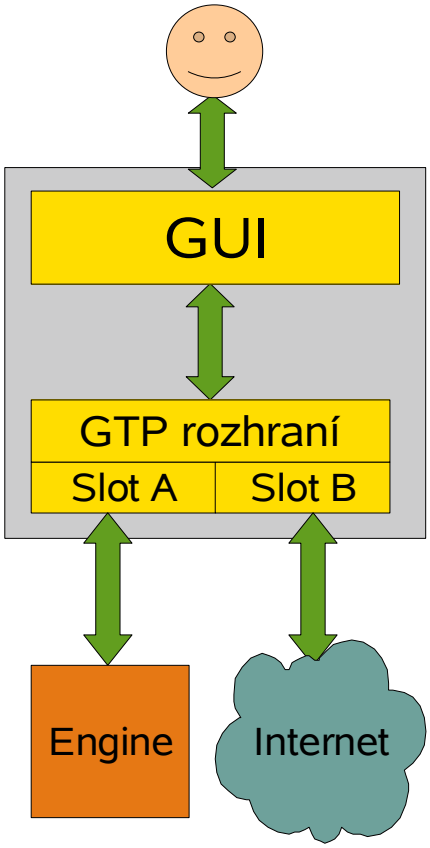
Po dohodě s vedoucím bakalářské práce, bylo dosaženo následující specifikace appletu:

Applet bude ke své činnosti maximálně využívat připojený Go program (engine). Pomocí něj bude kontrolovat přípustnost tahů a také počítat výsledek partie. Případně i další potřebné úkony, pokud to bude možné. Při hře dvou hráčů poslouží engine jako arbitr (rozhodčí). Při hře člověka proti počítači poslouží engine i jako protihráč. Applet bude tedy možno klasifikovat jako kontrolér bez vlastní inteligence.

Schéma činnosti appletu při hře člověka proti Go engine a proti druhému hráči přes Internet znázorňují Obr. 4.1 a Obr. 4.2.



Obr. 4.1:Schéma činnosti appletu při hře člověka proti engine



Obr. 4.2:Schéma činnosti appletu při hře dvou lidí přes Internet

4.3 Implementace appletu

Nyní bude podrobněji popsán implementovaný applet.

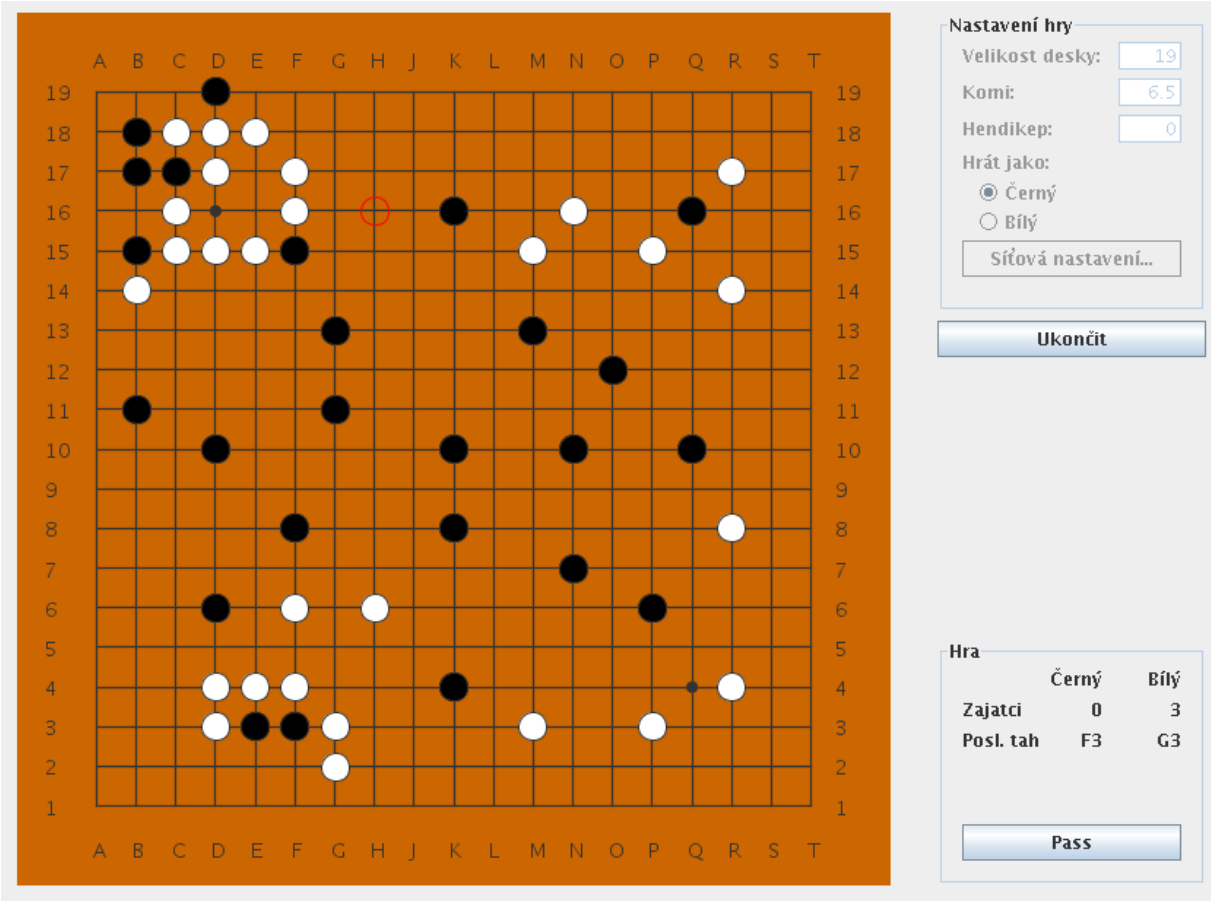
4.3.1 Splnění zadání

Při implementaci se z časových důvodů nepodařilo implementovat hru dvou hráčů přes Internet, požadovanou v zadání.

Applet (Obr. 4.3) tedy umožňuje hru člověka proti počítači s kontrolou přípustnosti tahů a možnosti spočítat výsledek partie.

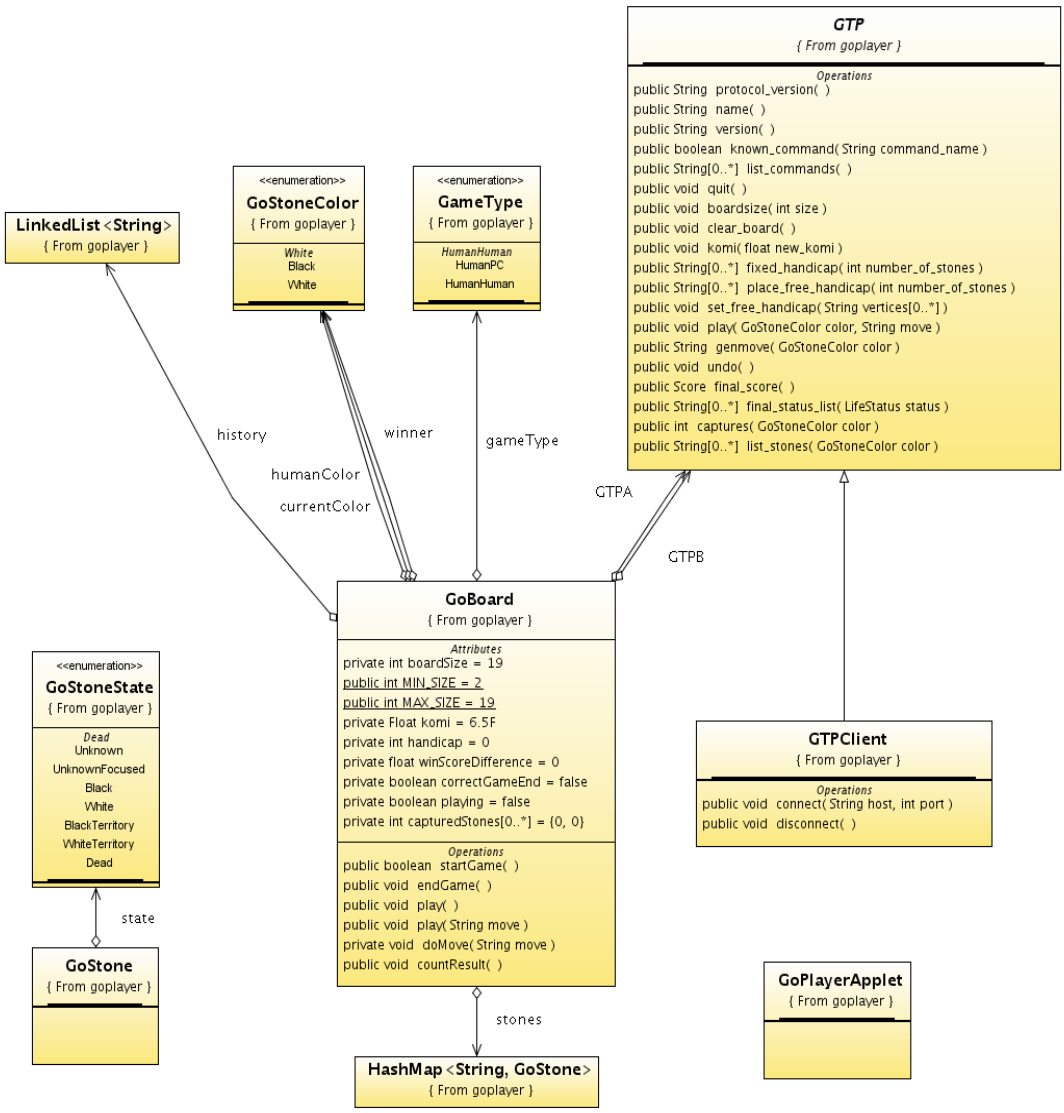
- Struktura appletu

Applet je tvořen několika třídami, které jsou znázorněny na Obr. 4.4. Diagram zobrazuje některé důležité atributy a veřejné metody tříd (kromě metod umožňujících získání hodnoty atributu). Nyní tedy stručně k jednotlivým třídám.



Obr. 4.3: Implementovaný applet s rozehranou partií Go

- **GoPlayerAplet** – je třída vlastního appletu, tj. tato třída je načtena prohlížečem
- **GoBoard** – je centrem celého appletu. Jedná se o grafický prvek představující hrací desku umístěný na formuláři appletu. Úkolem třídy je vykreslovat hrací desku (tj. sebe sama) a spravovat kameny (GoStone) umístěné na desce (kameny jsou na desce umístěny od počátku hry, jsou však průhledné). Především ale třída řídí celou hru za pomoci engine připojeného pomocí GTP.
- **GoStone** – jedná se rovněž o grafický prvek, představující hrací kámen. Úkolem třídy je vykreslovat sama sebe a reagovat na události přicházející od uživatele (ukázání nebo klepnutí myší) a případně o těchto událostech uvědomit mateřskou třídu (GoBoard).
- **GTP** – abstraktní třída implementující potřebné příkazy GTP. Třída však neimplementuje způsob komunikace s Go programem. To umožňuje vytvářet její konkrétní instance pro určitý druh komunikace (TCP/IP, mezi-procesová).



Obr. 4.4:Diagram tříd appletu se zobrazením důležitých atributů a metod

- **GTPClient** – třída implementuje abstraktní metody třídy GTP tak, aby umožňovala komunikaci pomocí TCP/IP.
- **GoStoneColor** – výčtový typ představující barvy kamene
- **GoStoneState** – výčtový typ představující stavy kamene (bezbarvý, zvýrazněný, černý, bílý, černý a bílý křížek a neživý)
- **GameType** – výčtový typ představující typ hry (proti programu, nebo proti člověku přes Internet)

4.3.2 Činnost appletu

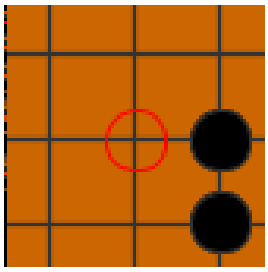
Po spuštění appletu (načtení prohlížečem), je před začátkem hry samotné nutné navázat spojení s engine[5]. Parametry komunikace je možno nastavit v dialogu „Nastavení sítě“. Engine musí podporovat rozšířené (v základní specifikaci neuvedené) příkazy GTP `captures` a `list_stones`, které implementuje například GNU Go [6]. To umožňuje appletu získávat informace o zajatých kamenech a současném stavu hrací desky. Dostupnost těchto příkazů je appletem ověřena. K samotnému připojení dojde (pokud jsou jeho parametry správné) po prvním stisku tlačítka „Start“ (applet se pak již neodpojuje).

Pokud tedy po stisku tlačítka „Start“ došlo k úspěšnému spojení s engine, jsou dále nastaveny parametry hry jako velikost desky, hendikep a komi uvedené v příslušných textových polích. Jestliže přitom nedojde k chybě (neplatné parametry hry), je možné zahájit vlastní hru. V opačném případě je zobrazena chyba a je nutné parametry opravit.

Po úspěšném spuštění hry, změní tlačítko „Start“ svůj popisek na „Ukončit“. Hráč může pomocí tohoto tlačítka hru kdykoli ukončit a případně začít hru novou.

Dále probíhá samotná hra. V případě, že si hráč zvolil bílou barvu, je ihned po stisku tlačítka „Start“ položen počítačem první kámen.

Při přejezdu myší nad plochou hrací desky, jsou zvýrazňovány volné průsečíky na desce – bezbarvý kámen na desce reaguje na ukázání myší vykreslením červené kružnice kolem svého obvodu (Obr. 4.5). Klepnutím na takto zvýrazněný kámen je započato jeho položení na desku.



Obr. 4.5: Zvýrazněný průsečík

Při pokládání kamene hrací deska (GoBoard) ověří pomocí připojeného engine přípustnost tahu. Je-li tah přípustný, je atribut barvy kamene nastaven na barvu hráče, a tím pádem položen na desku.

Deska po každém tahu zjišťuje od Go programu počet zajatých kamenů. V případě že se tento počet změnil (byl zajat nějaký soupeřův kámen), vyžádá si deska od Go programu kompletní seznam soupeřových kamenů a aktualizuje podle něj svoje informace o kamenech.

Počet zajatců a poslední tah každého z hráčů je průběžně zobrazován appletem, který příslušné hodnoty získá z atributů hrací desky.

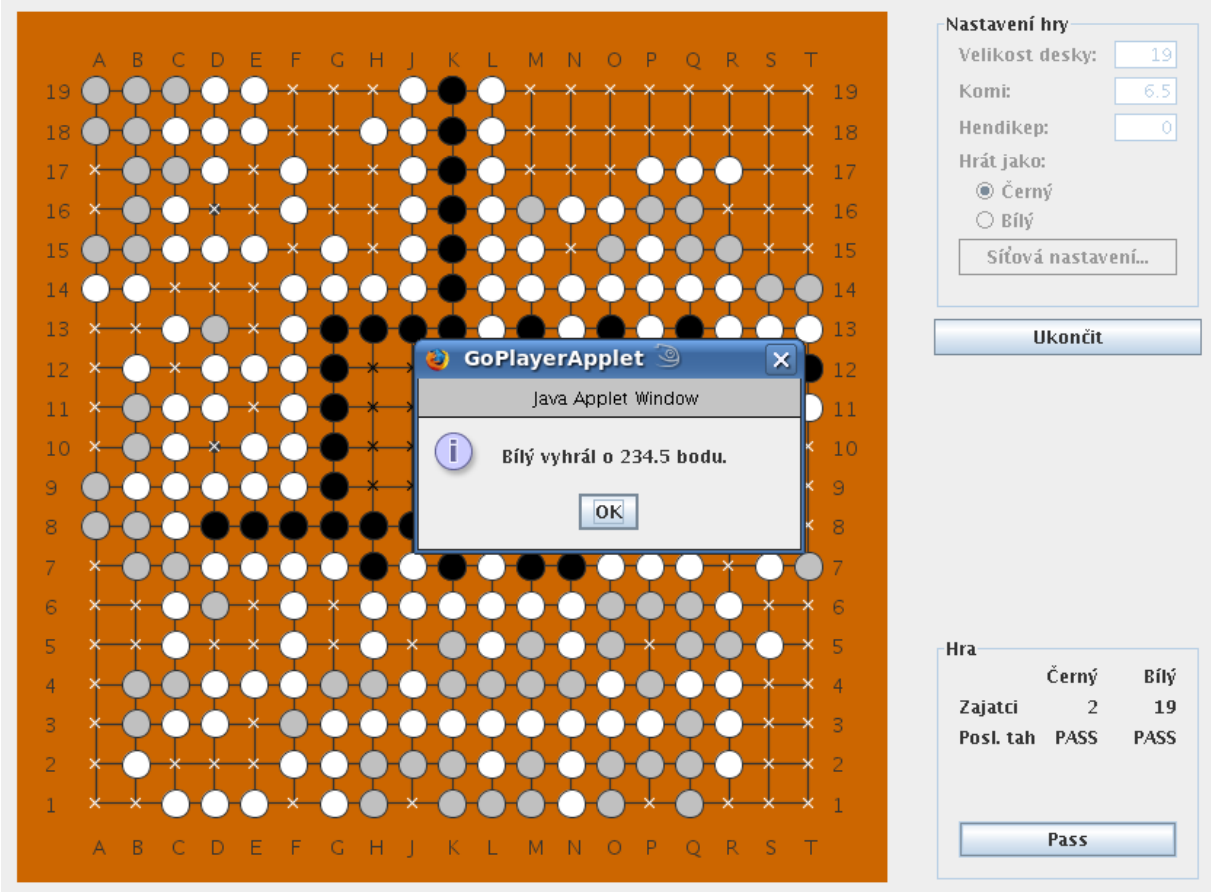
Pokud chce hráč zahrát pass a tím se pokusit ukončit hru, stiskne tlačítko „Pass“.

Hra je ukončena jedním z následujících způsobů:

- Uživatel stiskne tlačítko „Ukončit“
- Poslední dva tahy byly „Pass“
- Engine pošle „Resign“

5 Spuštění GNU Go: `gnugo --mode gtp --gtp-listen 5454`

Pokud není parte přerušena předčasně tlačítkem „Ukončit“, je hráči nabídnuta možnost spočtení výsledku partie (pomocí engine). Applet zobrazí barvu vítězného hráče a o kolik bodů byl jeho výsledek lepší v dialogovém boxu. Kromě toho, je výsledek i graficky znázorněn pomocí černých a bílých křížků, které označují zabrané území a pomocí šedých kamenů, které představují neživé kameny (odebrané při počítání výsledku). Názorně to ukazuje Obr. 2.2.



Obr. 4.6: Zobrazení výsledku partie. Jak je vidět, tato partie se mi moc nevydařila…

5 Závěr

Výsledkem této práce je funkční applet poskytující rozhraní pro hraní hry Go počítačem. Připojený Go program umožňuje appletu kontrolovat přípustnost tahů, počítání výsledku partie a poskytuje další potřebné funkce, které applet využívá. Applet umožňuje nastavení parametrů hry jako je komi, hendikep a velikost desky a zobrazuje počet zajatých kamenů a poslední tah každého z hráčů.

Z časových důvodů, však nebyla dokončena implementace funkce umožňující hru dvou hráčů přes Internet.

Dalším vývojem appletu by byla samozřejmě implementace této funkce. Kromě toho by bylo zajímavé rozšířit applet o funkce jako vrácení tahu, nápovědu dalšího tahu, zobrazení historie tahů nebo o možnost sledování hry dvou připojených Go programů.

Osobně bych se však přikláněl k transformaci appletu na samostatnou aplikaci, protože by pak bylo možné, aby byl Go program touto aplikací přímo spouštěn. Applet, pro jehož funkci je nutné nejdříve ručně spustit program a následně se k němu z appletu připojit pomocí TCP/IP, se mi jeví jako poněkud nešikovný.

Literatura

[1] GROSSER, T.: *První kniha o Go*. Toužim: Groser Tomáš – Tiger A. P. E., první vydání, 2004, ISBN 80-903538-0-0, 93 s.

[2] KOLEKTIV AUTORŮ: *GNU Go*. Změněno: 2006-11-23, <http://www.gnu.org/software/gnugo/>

[3] HOLLOSI, A: *SGF File Format FF[4]*. Změněno: 2006-08-06, <http://www.red-bean.com/sgf/>

[4] WILCOX, B.; FOTLAND, D.; KIERULF, A.: *Go Modem Protocol*. Změněno: 2006-10-13, <http://www.britgo.org/tech/gmp.html>

[5] FARNEBÄCK, G: *Specification of the Go Text Protocol, version 2, draft 2*. Změněno: 2002-10, <http://www.lysator.liu.se/~gunnar/gtp/gtp2-spec-draft2/gtp2-spec.html>

[6] KOLEKTIV AUTORŮ: *GNU Go Documentation*. Změněno: 2004-11-27, <http://www.gnu.org/software/gnugo/gnugo_toc.html>

[7] SUN MICROSYSTEMS, INC.: *Sun Java Studio Enterprise at a Glance*. Změněno: 2007, <http://developers.sun.com/prodtech/javatools/jsenterprise/index.jsp>