BI-PA2 – semestrální práce: Šachy

Vojtěch Mikšů
miksuvoj@fit.cvut.cz
ČVUT FIT

2010/2011

1 Zadání

Zadáním je vytvořit šachový program, který bude umožňovat hru proti umělé inteligenci. Nejdříve je však potřeba vyřešit hned několik podúkolů. Abych mohl hru vůbec nějak testovat a aby byla hratelná, bylo potřeba vyřešit její ovládání. Jednou z možností bylo ovládání pomocí zadávání tahů prostřednictvím konzole, což není pro normální rekreační hráče, kteří nejsou s šachovou notací seznámeni, moc pohodlné. Také je potřeba vyřešit, jak zobrazovat aktuální rozestavění figurek. Nejprostší variantou by bylo, aby si hráč postavil na stole šachy a tam si partii přehrával. Po zvážení těchto aspektů jsem se nakonec rozhodl zpracovat grafickou (a zvukovou) nadstavbu, kterou by šlo ovládat pomocí myši, což je nepřirozenější a nejpoužitelnější.
Pokud mám vytvořit model hry a umělou inteligenci je nejprve nutné vyřešit hru 2 lidských hráčů proti sobě. Nejdříve jsem tedy naprogramoval hru 2 hráčů proti sobě a až poté doplnil umělou inteligenci. Hru lze tedy spouštět ve 2 režimech. Defaultně se zapíná hra 2 hráčů a s přepínačem UI pak hra proti umělé inteligenci.

2 Možnosti programu

Program implementuje kompletní pravidla FIDE (světové šachové fedarace). Což znamená, že:

- hlídá pohyb všech figurek
- kontroluje, zda králové nejsou v šachu
- kontroluje, zda hráči mohou udělat tah (zda nenastal pat)
- zná všechny typy rošád, hlídá jejich proveditelnost
- zná braní mimochodem
- umí proměnit pěšáka, který dojde na poslední řadu (vyskočí box figurek, ze kterých si hráč může vybrat)
- umí rozpoznat, pokud nastal šach mat nebo pat

- hlídá 3x opakovanou pozici - prohlásí partii za remízu
- hlídá 50 tahů bez pohybu pěšáka / výměny figurek - prohlásí partii za remízu
- zná dvojskok pěšáka ze základní pozice

3 Zvuky

Program zvukově upozorňuje na tyto události:

- začátek hry
- pohyb figurky
- zamýšlený tah není legální
- nastal šach mat
- nastala remíza (pat)

4 Umělá inteligence

Umělá inteligence (dále UI) je realizována pomocí algoritmu minimax. Základem UI je ohodnocovací funkce. Ta projde celou šachovnici a sečte hodnotu figurek jednoho hráče. Každé figurce je přidělena nějaká hodnota. Pěšák - 1, Věž - 5, Kůň - 3, Střelec - 3, Král - 0, Dáma - 9. Čím vyšší hodnota se vrátí, tím je tah nadějnější. Tato ohodnocovací funkce by šla dále vylepšovat. Například pokud se pěšák nachází před proměnou, měla by se jeho hodnota razantně zvýšit. Stejně tak je hodnotnější věž, která stojí na volném sloupci. Na druhou stranu by se pak celý algoritmus zpomalil.
Základní algoritmus tedy vypadá tak, že si necháme vygenerovat seznam všech možných tahů, postupně je zahrajeme, ohodnotíme ohodnocovací funkcí a vrátíme. Pak si najdeme ten tah, který byl nejcennější a ten zahrajeme. Slabinou tohoto postupu je, že vůbec nehodotíme figurky soupeře.
Minimax (který je užitý v mém programu) provádí výše uvedené rekurzivně. Místo ohodnocovací funkce volá sám sebe. Pokud ho použijeme do úrovně 1, stane se to, že se vygeneruje seznam všech našich možných tahů a ke každému z nich seznam všech možných tahů soupeře. Nejdříve ke každému našemu tahu najde nejlepší tah soupeře a ten bude od hodnoty tohoto našeho tahu odčítat. Jako nejlepší náš tah tedy bude ten, který si udrží nejvyšší hodnotu i po odečtení nejlepšího tahu soupeře.
Minimax tedy hodnoty našich nejlepších pozic sčítá a hodnoty nejlepších pozic soupeře odečítá. Proto název Minimax. Pokud bychom nechali tento algortimus pracovat do vysoké úrovně, hrál by téměř dokonale. Bohužel už po 3 úrovni se stává časově nepoužitelný (tah trvá minuty). Proto je čisté využití minimaxu velmi omezující.
Existují i další a složitější algoritmy, ale vzhledem k rozsahu semestrální práce jsem již dále UI nevylepšoval.

5 Jak se program ovládá

- program se ovládá výhradně myší
- nejdříve je potřeba kliknout na figurku kterou táhneme a následně na pole, kam se s ní chceme posunout
- program lze ukončit pomocí klávesy ESC nebo křížkem
- pokud chceme spustit hru proti UI spustíme program **s přepínačem UI (./sachy UI)**
- program nás zvukově upozorní, pokud hra skončila

6 Formáty které program načítá

Program načítá několik desítek obrázků ve formátu **.png** a také 5 zvukových stop ve formátu **.wav**.

7 Knihovny třetích stran

- SDL - knihovna pro práci s grafikou a audiem
- SDL_image - rozšířená práce s obrázky, umožňuje používat i formáty jako .png
- SDL_mixer - rozšířená práce se zvuky, umožňuje jednodušší používání zvuků

8 Popis implementace

- Třída Sachy - stará se o chod hry, inicializuje šachovnici, vykresluje obraz, přehrává zvuky, zpracovává uživatelské vstupy, hlídá konec hry, střídá hráče v nekonečném cyklu, částečně vypomáhá implementovat složitější pravidla jako je rošáda nebo proměna pěšce
- Třída Sachovnice - reprezentuje šachovnici / rozestavění figur, provádí nad šachovnicí některé úkony jako zjišťování toho zda králové nejsou v šachu, zda hráči mohou hrát, hlídá stavy výběru tahu
- Třída Engine - neboli motor, obsahuje statické metody, které tvoří umělou inteligenci
- Třída Figurka - zastřešuje všechny typy figurek, obsahuje údaje o barvě figurky, o možnostech pohybu figurky - tyto metody jsou virtuální a implementovány až v potomcích, s výhodou tady využívám polymorfismu
- Třída Vez - potomek Figurka, implementuje pohyb věže
- Třída Strelec - potomek Figurka, implementuje pohyb střelce
- Třída Kun - potomek Figurka, implementuje pohyb koně

- Třída Kral - potomek Figurka, implementuje pohyb krále
- Třída Dama - potomek Figurka, implementuje pohyb dámy
- Třída Pesak - potomek Figurka, implementuje pohyb pěšáka

9 Závěr a hodnocení

Šachový program je bezesporně velmi zajímovou výzvou. Programátor si vyzkouší řadu technik a při řešení programu se musí postavit spoustě problémů. OOP se pro šachy hodí jenom do té doby, než začnete implementovat speciální pravidla typu rošáda, které najednou začnou celou strukturu narušovat a program ztrací na elegantnosti. Třídy pak musejí být na sobě závislé více, než bychom chtěli. Dalším problémem je i uživání virtuálních metod (polymorfismu), které se při hledání tahů volají třeba milionkrát za sebou, což z hlediska času asi nebude výhodné. Proto se spíše při řešení šachů doporučuje nevyužívat OO návrh. Nicméně lze si na tom pěkně vyzkoušet, jak se dá OOP v programech využívat.

10 Zdroje

- Šachové myšlení: http://www.linuxsoft.cz/article_list.php?id_kategory=241
- SDL: Hry nejen pro Linux - http://www.root.cz/serialy/sdl-hry-nejen-pro-linux/