

Uživatelská dokumentace k projektu

MONEY MAKER



vedoucí projektu

RNDr. David Hoksza

David Matoušek

Jan Raszyk

Petr Švec

Martina Tomisová

2008/2009

UNIVERZITA KARLOVA V PRAZE

matematicko-fyzikální fakulta



Projekt *MoneyMaker* byl vytvořen
v rámci předmětu *Softwarový projekt (NPRG023)*.

Vedoucí projektu:

RNDr. David Hoksza

Autoři projektu:

David Matoušek

Jan Raszyk

Petr Švec

Martina Tomisová

Obsah

Obsah.....	5
Požadavky na uživatelský počítač.....	7
Požadavky na software.....	7
Požadavky na hardware.....	7
Uživatelská dokumentace.....	8
Obecný popis systému MoneyMaker.....	8
Jak napsat vlastní plugin.....	9
Jaké by měly být cíle odhadů.....	9
Sportovní strategie.....	9
Měnové strategie.....	9
Co budete potřebovat:.....	9
Jak na to:.....	10
Křížení proměnných a ukládání hodnot proměnných pro další spuštění.....	11
Datové struktury.....	12
MoneyPack.....	13
MoneyData.....	13
MoneyRateLine.....	14
SportPack.....	15
SuperSportData.....	15
SportData.....	16
CompetitionData.....	16
CompetitionYearData.....	16
TeamData.....	17
MatchData.....	17
Logování.....	18
Jak plugin otestovat.....	18
Odhady kurzů měn.....	18
Odhady výsledků sportovních utkání.....	19
Použití knihoven.....	20
Kurzů měn.....	21
Průměrovací strategie.....	21
Historie se opakuje.....	22
Zrcadlení.....	24
Podle vzorů.....	27
Kurzů vypisované na sportovní utkání.....	31
Průměrovací strategie.....	31
Strategie založená na odhadech skóre zápasu.....	32
Průměrovací strategie (č. 2).....	36
TableRank.....	37
Používání webového rozhraní.....	41
Kurzů měn – vykreslení grafu a stažení textových dat.....	41
Prohlížení sportovních výsledků.....	44
Registrace a přihlašování.....	45

Editace osobních údajů.....	47
Správa strategií.....	47
Vložení strategie do systému.....	48
Zobrazení strategií.....	49
Aktivace strategie.....	49
Ukončení strategie.....	50
Editace parametrů strategie.....	51
Zobrazování výsledků strategie.....	52
Používání Moderátora.....	55
Ovládání Moderátora pro běžného uživatele.....	56
Osobní údaje.....	56
Uložená data.....	57
Strategie.....	57
Další záložky.....	61
Ovládání Moderátora pro administrátora nebo privilegovaného uživatele.....	62
Osobní údaje.....	62
Správa uživatelů.....	63
Skupiny.....	64
Systémové proměnné.....	64
Pro pokročilé – více tipů v jedné strategii.....	67
Implementace.....	67
Plugin nahraný v systému.....	68

Požadavky na uživatelský počítač

Požadavky na software

- .NET 2.0 a vyšší
- Visual Studio 2005
- webový prohlížeč

Požadavky na hardware

- Připojení k Internetu s rychlostí alespoň 64kb/s
- 450 MHz nebo rychlejší
- 192 MB RAM nebo více

Uživatelská dokumentace

Obecný popis systému *MoneyMaker*

Systém *MoneyMaker* poskytuje uživateli možnost pohodlně si vyvíjet strategie pro investování do zahraničních měn a typování sportovních výsledků. Framework *MoneyMaker* strategii testuje a vylepšuje každý den na aktuálních datech a poskytuje uživateli výsledky.

Framework sleduje pouze méně likvidní trhy (kurzy zahraničních měn). Cílem frameworku je pomoci uživateli vyvinout strategii, jež dobře predikuje za běžných okolností. Jakmile se ale ve světě děje něco mimořádného, co ovlivňuje hodnoty na trhu, tak se na ní spolehnout nedá.

Základní myšlenkou systému je křížení strategií. Každá strategie má vstupní parametry a na serveru běží v několika instancích, které se liší právě v hodnotě zadaných parametrů. Instance jedné strategie se mezi sebou každý den kříží genetickými algoritmy (kříží se pouze hodnoty parametrů). Instance jsou monitorovány, prodávající instance zabity a ty nejlépe vydělávající dále vzájemně kříženy. Na počátku si uživatel volí počáteční hodnoty parametrů a jejich smysluplný rozsah.

Uživatel, který chce framework využívat, si napíše plugin (libovolná .NET *assembly*), vyzkouší si ho pomocí jednoduchého programu založeného na historických datech (*BusinessDemo* aplikace). Až mu bude plugin fungovat, nahraje ho do frameworku, kde se bude pravidelně spouštět, vylepšovat a budou se vyhodnocovat jeho výsledky. Křížení parametrů strategií probíhá pouze ve frameworku, nikoli v *BusinessDemo*.

Celý framework běží v diskrétním čase (v cyklech po dnech). V každém cyklu

- se vyhodnotí úspěšnost tipů z předešlého dne,
- geneticky se zkříží nejlépe vydělávající strategie,
- každé živé instanci se aktualizují data (odpovídající aktuální situaci),
- získá se nový tip na další den od každé instance.

I když je strategie zpětně testována na historických datech (to dělá např. *BusinessDemo* aplikace), neví o tom. Framework se vždy tváří, jako by data z následujícího dne neznal, i když už je ve skutečnosti má k dispozici.

Jak napsat vlastní plugin

Než začnete plugin tvořit, je nutné si zvolit, co bude plugin odhadovat – zda kurzy měn, nebo kurzy sázek na sportovních utkání. Plugin, který by mohl odhadovat oboje najednou, totiž nelze napsat. U každého pluginu je třeba si zvolit jeho typ – peníze, nebo sporty.

Pokud se chcete zabývat odhady jak kurzů měn, tak sportovních výsledků, je třeba si napsat alespoň dva pluginy – pro každou oblast jeden. To ovšem není problém, množství pluginů, které budete chtít ve frameworku provozovat zatím není nijak omezené.

Jaké by měly být cíle odhadů

Sportovní strategie

Cílem strategie by mělo být vypsát na zápas takové kurzy, aby na nich sázková kancelář co nejvíce vydělala – toto kritérium systém vyhodnocuje a podle něho ukončuje málo úspěšné strategie a množí ty více úspěšné. K dispozici jsou pro tento účel historická data – výsledky zápasů i kurzy na ně vypsané sázkovými kancelářemi. Kurzy, které strategie odhaduje, by měly být co nejpodobnější kurzům, které vypisují sázkové kanceláře, ideálně pak ještě lepší.

Měnové strategie

Strategie tipující kurzy měn by se měla svým tipem snažit co nejvíce přiblížit skutečné hodnotě kurzu, který nastane.

Co budete potřebovat:

Všechny níže jmenované soubory jsou ke stažení v zip archivu:
<http://navarin.ms.mff.cuni.cz:7070/download/BusinessDemo.zip>.

Zcela nutně musíte mít:

- ➔ *BusinessmanCommon.dll* (základní prostředí pluginu)
- ➔ *BusinessDemoMoney.exe* nebo *BusinessDemoSport.exe*

Nástroje pro testování pluginu – zjednodušeně simulují běh pluginu ve frameworku. Primární úloha těchto souborů je testovat, zda plugin „nespadne“, navíc dávají i užitečné výsledky o jeho odhadech a skutečnosti. První soubor je určen pro testování pluginů pro odhad kurzů měn, druhý pro odhad sportovních výsledků.

- ➔ Textový soubor *MoneyData.txt* nebo *SportData.txt* obsahující historická data. S těmito daty pracují *BusinessDemo** programy.

Doporučujeme také:

- ➔ Statistické knihovny zdarma, které můžete použít ve svém pluginu (tyto knihovny se nacházejí rovněž na serveru, proto je lze použít). Ke stažení jsou na adrese

<http://navarin.ms.mff.cuni.cz:7070/download/statisticalLibraries.zip>.

Vhodné příklady:

→ Jsou ke stažení i prohlédnutí na webu <http://navarin.ms.mff.cuni.cz:7070/examples.php>.

Jak na to:

1. Stáhněte si příslušné soubory z webu:

<http://navarin.ms.mff.cuni.cz:7070/download/BusinessDemo.zip>

případně také:

<http://navarin.ms.mff.cuni.cz:7070/download/statisticalLibraries.zip>

<http://navarin.ms.mff.cuni.cz:7070/examples.php>

2. Otevřete *Microsoft Visual Studio* a klikněte

File → *New* → *Project...*

3. Zvolte *Visual C#* → *Class Library*.

4. Jméno projektu (*Name:*) je vhodné zvolit *BusinessmanPlugin*, abychom pak nemuseli opravovat jméno *namespace*. Jméno solution si můžete zvolit zcela libovolně, například *BusinessmanPluginSimple*.

5. Vytvořte projekt.

6. Soubor *Class1.cs* je z důvodů přehlednosti vhodné přejmenovat na *Plugin.cs* (nepovinný krok).

7. Zkontrolujte, zda je *namespace* pojmenován *BusinessmanPlugin*. Pokud ne, přejmenujte ho tak.

8. Třídu *Class1* přejmenujte na *Plugin* (*Visual Studio* pravděpodobně samo nabídne, zda to má udělat, při přejmenovávání souboru v předchozím kroku, nebo to tak rovnou udělá).

9. Do souboru *Plugin.cs* přidejte řádek

```
using BusinessmanCommon;
```

10. Projekt by nyní měl vypadat takto:

```
using System;
using System.Collections.Generic;
using System.Text;
using BusinessmanCommon;

namespace BusinessmanPlugin
{
    public class Plugin
    {
    }
}
```

11. Plugin chceme podle šablony *BusinessCommon*, tedy opravíme

```
| public class Plugin
```

na

```
| public class Plugin : PluginBase
```

12. Do třídy *Plugin* přidáme potřebné funkce. Funkce *run* se bude volat každý den a jejím úkolem je spočítat tip kurzů na následující den z dat dnešních a starších.

```
| public Plugin():base() { }
```

```
| public override void run(PackBase pack) { }
```

13. Přidejte do projektu knihovnu *BusinessmanCommon.dll*: klikněte pravým tlačítkem v *Solution Explorer* na *References* → *Add Reference...* → tabulka *Browse* → vyberte cestu k souboru *BusinessmanCommon.dll* → OK.

14. Pokud budete používat statistické knihovny, což velmi doporučujeme, musíte je nyní přidat také:

Klikněte pravým tlačítkem v *Solution Explorer* na *References* → *Add Reference...* → tabulka *Browse* → vyberte cestu k souboru *Statistics.dll* → OK.

Nezapomeňte přidat do souboru řádku:

```
| using statistic;
```

Analogicky pro další knihovny, které chcete používat.

15. Napište vlastní kód podle rad následujících níže.

16. Plugin zkompilejte: klikněte v menu na *Build* → *Build*.

17. Když už máte plugin hotový, otestujte ho *BusinessDemo** aplikací (viz. návod níže). Až bude dobře fungovat, nahrajte ho přes webové rozhraní, nebo *Moderatora*, do frameworku – tento krok je také popsán dále.

Křížení proměnných a ukládání hodnot proměnných pro další spuštění

Proměnné lze rozdělit na tři typy:

- proměnné, co se kříží
- proměnné, co se nekříží, ale jejich hodnota je před spuštěním vždy nastavena na hodnotu, kterou měly na konci minulého běhu
- proměnné, co se nekříží a jejich hodnota není před spuštěním inicializována

Proměnné, které chcete nechat křížit, musíte označit jako *public*, deklarovat je přímo v třídě *Plugin* (ne ve funkci *run*), musí být *serializable* a jednoduchého typu. A zároveň všechny proměnné, které budou *public*, *serializable* a jednoduchého typu přímo ve třídě *Plugin*, se budou křížit. Křížit lze jednoduché typy, jako *int*, *uint*, *decimal*, *float*, *real*, *string* (kříží se lexikograficky) atd. Proměnné se kříží pouze, pokud plugin běží v hlavním frameworku.

Do něho bude typicky uživatelem umístěn po otestování pomocí *BusinessDemo*, kde se proměnné nekříží.

Pokud proměnná splňuje podmínky z předchozího odstavce, ale není jednoduchého typu, tak **se nebude křížit, ale její hodnota bude mezi běhy ukládána**.

Ostatní proměnné se **nebudou ani křížit, ani inicializovat poslední hodnotou** z minulého běhu.

Ale **pozor**, proměnné, které spadají do kategorie křížených, se křížit nemusí. U každé z nich budete mít při nahrávání pluginu do systému možnost nastavit, zda se křížit bude, nebo nebude. Toto nastavení lze provést na webu, nebo pomocí *Moderátora*. Volbu, zda se mají proměnné křížit, nebo ne, lze nastavit také pro celou strategii najednou. Pokud má strategie křížení povoleno, bere se ohled na individuální nastavení každé proměnné, zda se má křížit, nebo ne. Pokud má celá strategie křížení zakázáno, nekříží se nikdy žádná proměnná.

Datové struktury

Zde popsané části datových struktur jsou základní. Pokud by vás zajímaly další méně používané funkce a možnosti, přečtěte si odpovídající část technické dokumentace a případně dokumentaci generovanou automaticky.

Rozhodně se ale seznámte s názvoslovím používaným u sportů.

Terminologie

Než si představíme hlavní datovou strukturu, je třeba se seznámit s terminologií, protože v některých případech není tak intuitivní.

<i>Super sport</i>	Tímto termínem označujeme hlavní jméno sportu, tedy např. fotbal, hokej, basketbal.
<i>Sport</i>	<i>Sport</i> je podmnožinou <i>super sportu</i> , je jím např. ženský fotbal, mužský fotbal, ženský hokej, mužský hokej, mužský hokej atd.
<i>Soutěž, competition</i>	Pod <i>soutěží</i> rozumíme skutečně soutěž, tedy např. Pohár UEFA, 1. anglická liga, NHL, NBA... Soutěž je podmnožinou <i>sportu</i> .
<i>Ročník soutěže, competition year</i>	Jedná se o ročník značený jedním, nebo dvěma letopočty. Např. 2008/2009, nebo jenom 2009. <i>Ročník soutěže</i> je podmnožinou <i>soutěže</i> .
<i>Tým, team</i>	Každý <i>tým</i> má své jméno a sport, kterému se věnuje. Např. Sparta. Sparta je ale v našem systému dvakrát – první je spojena s hokejovou soutěží, druhá s fotbalovou.
<i>Zápas, match</i>	Tento termín lze opět chápat zcela intuitivně, jedná se o zápas dvou týmů, který má (bude mít) nějaký výsledek, má datum konání, týmy, které se ho účastní proti sobě, atd.

MoneyPack

MoneyPack je třída obsahující všechna data měn a jejich kurzů.

Člen třídy	Typ	Popis
		Hash-mapa s daty o měnách. Klíč je jméno měny, jako hodnotu vrací kurzy měny. Např.:
Money	Dictionary<string, MoneyData>	<pre>MoneyPack mPack = (MoneyPack)pack; MoneyData mData; mPack.Money.TryGetValue("CZK", out mData);</pre> V mData budeme mít uložené kurzy české koruny vůči všem ostatním měnám.

MoneyData

MoneyData je třída obsahující data vztahující se pouze k jedné dané měně. K dispozici máte:

Člen třídy	Typ	Popis
Id	short	ID měny
Code	string	Pojmenování měny (např.: USD, CZK, EUR)
BornTime	DateTime	Datum vzniku měny
DeathTime	DateTime	Datum ukončení platnosti měny (z minulosti, nebo předpokládané)
		Hash-mapa s kurzu měn. Klíč je jméno druhé měny, hodnota obsahuje kurzy základní měny property MoneyData a druhé měny zadané jako klíč. Např.:
RateLines	Dictionary<string, MoneyRateLine>	<pre>// získání základního balíku dat MoneyPack mPack = (MoneyPack)pack; MoneyData mData; // vybrání balíku dat vztahujícího se k české koruně mPack.Money.TryGetValue("CZK", out mData); MoneyRateLine mrLine; // vybrání balíku dat obsahujícího kurzy mezi českou // korunou a americkým dolarem mData.RateLines.TryGetValue("USD", out mrLine);</pre> V mrLine máme nyní seznam kurzů mezi českou korunou a americkým dolarem.

MoneyRateLine

Třída obsahující data vztahující se ke vzájemnému kurzu dvou měn. Kurz je brán formou „z jedné měny na druhou“, tedy jedna je brána jako „ta, ze které“ a druhá jako „ta, na kterou“.

Člen třídy	Typ	Popis
FromId	short	ID měny „ze které“ převádíme
ToId	Short	ID měny „na kterou“ převádíme
FromCode	String	Jméno měny „ze které“ převádíme
ToCode	String	Jméno měny „na kterou“ převádíme
HistRates	List<decimal>	Seznam obsahující pouze číselné hodnoty kurzů mezi měnami
getLastHistRate	decimal	Vrátí poslední hodnotu kurzu. Pokud žádná neexistuje, vrací decimal.MinValue.
HistDates	List<DateTime>	Seznam obsahující pouze data (datum) patřící k hodnotám kurzů. Tedy HistDates společně s HistRates tvoří „tabulku“: datum – kurz.
getLastHistDate	DateTime	Vrátí poslední datum, kdy je znám kurz. Pokud žádné neexistuje, vrací DateTime.MinValue.
Bets	List<decimal>	Seznam našich minulých tipů
GetLastBet	decimal	Vrátí hodnotu našeho posledního tipu.
BetDates	List<DateTime>	Seznam dat (datum), kdy jsme uložili další tip. Spolu s Bets tvoří dvojice: datum – náš tip příslušící k datu
getLastBetDate	DateTime	Vrátí datum, kdy jsme uložili náš poslední tip. Pokud žádné neexistuje, vrací DateTime.MinValue.
NextBet	decimal	Jediná zapisovatelná property. Do této property uložte svůj aktuální tip.
isThereANextDayRate	bool	Vrací True, pokud má smysl počítat další tip (ještě není spočtený a uložený). Některé dny se totiž nové kurzy nevypisují, ale pluginy se spouští každý den.

SportPack

Funkčnost a struktura třídy SportPack je obdobná, jako u třídy MoneyPack, ale vztahuje se ke sportům. Zde je stručný popis nejdůležitějších proměnných a funkcí.

Člen třídy	Typ	Popis
		Hash-mapa se SuperSporty. Klíč je jméno super sportu, jako hodnotu vrací objekt příslušného super sportu.
		Např.:
SuperSports	Dictionary<string, superSportData>	<pre>SportPack sPack = (SportPack)pack; CompetitionData competition = sPack.getCompetition("FOTBAL", "FOTBAL MUZI", "ANGLICKÁ 1. LIGA MUZI");</pre> V CompetitionData budeme mít uložená data týkající se 1. anglické ligy.

SuperSportData

Objekt této třídy obsahuje data týkající se jednoho super sportu (hokeje, fotbalu...).

Člen třídy	Typ	Popis
Id	int	ID super sportu
Title	string	Pojmenování super sportu (např.: BASKETBAL, HOKEJ)
		Hash-mapa se Sporty. Klíč je jméno sportu, jako hodnotu vrací objekt týkající se příslušného sportu.
		Např.:
Sports	Dictionary<string, SportData>	<pre>SportPack sPack = (SportPack)pack; CompetitionData competition = sPack.getCompetition("FOTBAL", "FOTBAL MUZI", "ANGLICKÁ 1. LIGA MUZI");</pre> V CompetitionData budeme mít uložená data týkající se 1. anglické ligy. Požadovaný sport je zde určen řetězcem "FOTBAL MUZI".

SportData

Objekt tohoto typu sdružuje data týkající se nějakého sportu (např. HOKEJ MUZI).

Člen třídy	Typ	Popis
Id	int	ID sportu
Title	string	Pojmenování sportu (např.: BASKETBAL MUZI)
Competitions	Dictionary<string, CompetitionData>	Hash-mapa soutěží. Klíč je jméno soutěže, jako hodnotu vrací objekt týkající se příslušné soutěže. Např.: <pre>SportPack sPack = (SportPack)pack; CompetitionData competition = sPack.getCompetition("FOTBAL", "FOTBAL MUZI", "ANGLICKÁ 1. LIGA MUZI");</pre> V CompetitionData budeme mít uložená data týkající se 1. anglické ligy. Požadovaná soutěž je zde určena řetězcem "ANGLICKÁ 1. LIGA MUZI".
Teams	Dictionary<int, TeamData>	Hash-mapa týmů. Klíč je id týmu, jako hodnotu vrací objekt týkající se příslušného týmu.

CompetitionData

Třída pro práci s daty o jednotlivých soutěžích.

Člen třídy	Typ	Popis
Id	int	ID soutěže
Title	string	Pojmenování soutěže (např.: NHL)
CompetitionYears	Dictionary<DateTime, CompetitionYearData>	Hash-mapa ročníků soutěží. Klíč je rok, kdy se soutěž koná, jako hodnotu vrací objekt příslušné soutěže.
Teams	TeamData[]	Pole týmů účastnících se soutěže
Matches	MatchData[]	Pole všech zápasů soutěže
PlayedMatches	MatchData[]	Pole odehraných zápasů soutěže
FutureMatches	MatchData[]	Pole chystaných zápasů soutěže
CanceledMatches	MatchData[]	Pole zrušených zápasů soutěže

CompetitionYearData

Objekty pro uchovávání dat vztahujících se k jednomu ročníku soutěže.

Člen třídy	Typ	Popis
Id	int	ID ročníku soutěže
FromTime	DateTime	První rok ročníku

Člen třídy	Typ	Popis
ToTime	DateTime	Druhý rok ročníku, může být totožný s prvním, jestliže je sezóna v rámci jednoho roku.
Teams	TeamData[]	Pole týmů účastnících se tohoto ročníku soutěže
Matches	MatchData[]	Pole všech zápasů tohoto ročníku soutěže
PlayedMatches	MatchData[]	Pole odehraných zápasů tohoto ročníku soutěže
FutureMatches	MatchData[]	Pole chystaných zápasů tohoto ročníku soutěže
CanceledMatches	MatchData[]	Pole zrušených zápasů tohoto ročníku soutěže

TeamData

Třída pro uchovávání a práci s daty týkajícími se jednotlivých týmů.

Člen třídy	Typ	Popis
Id	int	ID týmu
Title	string	Jméno týmu (Sparta, Slavia)
Sport	SportData	Jakému sportu se tým věnuje. Tým téhož jména se může věnovat více sportům, potom objektů TeamData o tomto týmu je tolik, kolika sportům se věnuje. Např. <i>Sparta</i> bude mít první instanci objektu TeamData týkající se hokeje a další vztahující se k fotbalu.

MatchData

Pro popis dat o jednom zápasu.

Člen třídy	Typ	Popis
Id	int	ID zápasu
HomeTeam	TeamData	Tým domácích
VisitTeam	TeamData	Tým hostů
Prolongation	bool	Příznak, zda byl zápas prodlužován
Time	DateTime	Datum a čas konání zápasu
Round	int	Číslo kola v rámci soutěže
IsPlayed	bool	Příznak, zda je již zápas odehraný
IsCanceled	bool	Příznak, zda byl zápas zrušen
IsFuture	bool	Příznak, zda se zápas teprve bude konat
ScoreReaded	bool	Příznak, zda je již načtené skóre do systému
HomeScore	int	Skóre domácích
VisitScore	int	Skóre hostů
Opportunities	decimal[]	Pole má pevnou délku 5 a slouží pro uložení kurzů na zápas, jak je stanovily sázkové kanceláře. Pořadí dat v poli je následující: 0 - výhra domácích 1 - výhra domácích nebo remíza

Člen třídy	Typ	Popis
		2 - remíza 3 - výhra hostů nebo remíza 4 - výhra hostů
Bets	decimal[]	Pole má pevnou délku 5 a slouží pro uložení tipů na kurzy na zápas, jak plugin očekává, že je stanoví sázkové kanceláře. Pořadí dat v poli je následující: 0 - výhra domácích 1 - výhra domácích nebo remíza 2 - remíza 3 - výhra hostů nebo remíza 4 - výhra hostů

Logování

Funkce pro tištění informací za běhu pluginu. Nevrací žádnou hodnotu, její hlavička je:

```
| protected void log(string msg);
```

Příklad použití:

```
| log("Hello world!");
```

Takto si můžete ukládat logy – uloží se do databáze a pak si je můžete kdykoliv později prohlédnout pomocí webového rozhraní nebo *Moderátora*.

Jak plugin otestovat

Vytvořte si pracovní adresář. Do adresáře zkopírujte knihovnu `BusinessmanCommon.dll` a testovací program `BusinessDemoMoney.exe`, nebo `BusinessmanDemoSport.exe`. Pro jednoduchost sem zkopírujte i soubor s historickými daty a Váš plugin (tento krok není povinný, ale připravte se na to, že pokud je sem nenakopírujete, budete muset při testování zadat cestu k jejich aktuálnímu umístění).

Odhady kurzů měn

Po spuštění `BusinessDemoMoney` se objeví návod:

```
| Invalid number of arguments.

| Usage: BusinessDemoMoney plugin datafile outputfile.
| plugin      - Name of the plugin library file.
| datafile    - Name of the CNB currency rate file.
| outputfile  - Name of the output file.
```

Pokud se neobjevil, zkontrolujte, zda jste do adresáře nakopírovali oba povinné soubory.

Dle návodu postupujeme dále, zadáme argumenty:

➔ cesta k pluginu

- ➔ cesta k souboru s historickými daty
- ➔ cesta k souboru, do kterého budou zapsány výsledky

Výsledky se vypíší na konzoli i do souboru. Pořadí argumentů je důležité zachovat.

Příklad spuštění, všechny soubory jsou v aktuálním adresáři:

```
C:\businessdemo> BusinessDemoMoney BusinessmanPlugin.dll MoneyData.txt test_output.txt
MoneyMaker BusinessDemoMoney.

Bet made on 01.01.1999: CZK -> USD = 30.166; Real rate was 29.758
Bet made on 05.01.1999: CZK -> USD = 29.758; Real rate was 29.562
Bet made on 06.01.1999: CZK -> USD = 29.562; Real rate was 29.647
Bet made on 07.01.1999: CZK -> USD = 29.647; Real rate was 29.890
Bet made on 08.01.1999: CZK -> USD = 29.890; Real rate was 29.914
Bet made on 11.01.1999: CZK -> USD = 29.914; Real rate was 30.249
Bet made on 12.01.1999: CZK -> USD = 30.249; Real rate was 30.708
Bet made on 13.01.1999: CZK -> USD = 30.708; Real rate was 30.769
Bet made on 14.01.1999: CZK -> USD = 30.769; Real rate was 30.721
...
```

V závislosti na složitosti výpočtů strategie může být doba běhu *BusinessDemoMoney* poměrně dlouhá.

Odhady výsledků sportovních utkání

Po spuštění *BusinessDemoSport* se objeví návod:

```
Invalid number of arguments.

Usage: BusinessDemoSport plugin datafile outputfile.
plugin      - Name of the plugin library file.
datafile    - Name of the SportsData file.
outputfile  - Name of the output file.
```

Pokud se neobjevil, zkontrolujte, zda jste do adresáře nakopírovali oba povinné soubory.

Dle návodu postupujeme dále, zadáme argumenty:

- ➔ cesta k pluginu
- ➔ cesta k souboru s historickými daty
- ➔ cesta k souboru, do kterého budou zapsány výsledky

Výsledky se vypíší na konzoli i do souboru. Pořadí argumentů je důležité zachovat.

Příklad spuštění, všechny soubory jsou v aktuálním adresáři:

```
C:\businessdemo> BusinessDemoSport BusinessmanPlugin.dll SportsData.txt test_output.txt
MoneyMaker BusinessDemoSport.

Date | Time | Competition | Home Team | Visitor Team | Result | Rate 1 | Rate 10 | Rate 0
| Rate 02 | Rate 2 | Bet 1 | Bet 10 | Bet 0 | Bet 02 | Bet 2 | Profit1 | Profit 10 |
Profit 0 | Profit 02 | Profit 2 | Profit Total

Instance 0: no data for 1. anglicka liga muzi, fotbal
<výstup zkrácen>
```

```

Instance 0: starting...
2009-02-09 | 12:00 | ANGLICKÁ 1. LIGA MUZI | MUZI ASTON VILLA | MUZI WEST BROM | 2:1 |
1.42 | 1.05 | 4.02 | 2.64 | 7.71 | 2.24 | 1.31 | 3.49 | 1.85 | 4.36 | -0.20 | -0.11 |
-0.06 | -0.19 | -0.11 | -0.67

2009-02-09 | 12:00 | ANGLICKÁ 1. LIGA MUZI | MUZI NEWCASTLE | MUZI WEST HAM | 2:2 | 2.22 |
1.31 | 3.20 | 1.58 | 3.12 | 3.18 | 1.59 | 3.24 | 1.36 | 2.38 | 0.11 | -0.10 | 0.01 | 0.13
| -0.11 | 0.04

2009-02-09 | 12:00 | ANGLICKÁ 1. LIGA MUZI | MUZI MIDDLESBROUGH | MUZI SUNDERLAND | 1:1 |
2.18 | 1.30 | 3.20 | 1.61 | 3.22 | 2.49 | 1.40 | 3.22 | 1.51 | 2.87 | 0.04 | -0.04 | 0.00 |
0.05 | -0.04 | 0.02

2009-02-09 | 12:00 | ANGLICKÁ 1. LIGA MUZI | MUZI EVERTON | MUZI HULL | 2:0 | 1.59 | 1.10
| 3.60 | 2.22 | 5.80 | 3.07 | 1.48 | 3.58 | 1.83 | 4.30 | -0.25 | -0.16 | -0.02 | -0.13 |
-0.07 | -0.64

2009-02-09 | 12:00 | ANGLICKÁ 1. LIGA MUZI | MUZI ARSENAL | MUZI BOLTON | 1:0 | 1.34 | 1.03
| 4.40 | 2.99 | 9.34 | 1.47 | 1.06 | 3.95 | 2.58 | 7.52 | -0.05 | -0.02 | -0.03 | -0.05 |
-0.03 | -0.17

2009-02-09 | 12:00 | ANGLICKÁ 1. LIGA MUZI | MUZI STOKE | MUZI LIVERPOOL | 0:0 | 8.46 |
2.86 | 4.33 | 1.04 | 1.37 | 9.59 | 3.12 | 4.65 | 1.05 | 1.38 | 0.01 | -0.02 | -0.01 | 0.03
| -0.02 | -0.01

Instance 0: starting...
2009-02-10 | 12:00 | ANGLICKÁ 1. LIGA MUZI | MUZI WIGAN | MUZI TOTTENHAM | 1:0 | 2.41 |
1.38 | 3.21 | 1.50 | 2.81 | 6.17 | 2.29 | 3.90 | 1.22 | 1.92 | -0.22 | -0.23 | 0.03 |
-0.21 | -0.19 | -0.82

2009-02-10 | 12:00 | ANGLICKÁ 1. LIGA MUZI | MUZI MANCHESTER UTD | MUZI CHELSEA | 3:0 |
2.21 | 1.29 | 3.09 | 1.60 | 3.30 | 4.79 | 1.91 | 4.66 | 2.28 | 6.42 | -0.21 | -0.19 | 0.08
| 0.14 | 0.12 | -0.05

```

V závislosti na složitosti výpočtů strategie může být doba běhu *BusinessDemoSport* poměrně dlouhá.

Dále naleznete několik strategií pro oba druhy pluginů, které mají za hlavní úkol ulehčit novým uživatelům programování – pro měnové strategie od strany 21 a pro sportovní strategie od strany 31. Dá se z nich dobře pochopit, jak by měl kód pluginu vypadat, a jak přistupovat k datovým strukturám.

Použití knihoven

Pokud jste zařadili do svého projektu i statistické knihovny, které jsou k dispozici na serveru (<http://navarin.ms.mff.cuni.cz:7070/download/statisticalLibraries.zip>), jejich použití ukazuje jednoduchý příklad text 1.

Text 1: příklad použití statistické knihovny statistics.dll

```

// store average rate into the 'avg' variable
List<double> rates = new List<double>();
int dayId = Math.Max((int)(mrLine.HistRates.Count - days), 0);
while (dayId < mrLine.HistRates.Count)

```

```

        rates.Add((double)mrLine.HistRates[dayId++]);
        statistics stat = new statistics(rates.ToArray());

        // save 'avg'
        mrLine.NextBet = (decimal)stat.mean();

```

Kurzy měn

Zdrojový kód každé strategie je sice uveden přímo zde, což sice umožňuje číst si ho bez přístupu k počítači, ale během programování se hodí mít ho k dispozici např. pro účely kopírování některých pasáží, proto jsou všechny kódy umístěny také na webu: <http://navarin.ms.mff.cuni.cz:7070/examples.php>.

Průměrovací strategie

Tato strategie je nejjednodušší z uvedených ukázkových strategií. Jako tip na další den uvede aritmetický průměr hodnot za posledních několik dní – počet těchto dní je uložen v proměnné, která se kříží.

Text 2: Kód hlavní funkce pluginu založeného na průměrování minulých hodnot

```

public uint days = 5;           // Define global variable
public decimal dec = 123.123m;
public double d = 12354.456;
public float f = 132.45f;

// A new bet is average rate from the last few days
public override void run(PackBase pack)
{
    // Get money pack
    MoneyPack mPack = (MoneyPack)pack;

    // Get rate lines between CZK and USD
    // Get money lines from CZK
    MoneyData mData;
    if (!mPack.Money.TryGetValue("CZK", out mData))
    {
        log("I can not find CZK!");
        return;
    }

    // Get money lines to USD
    MoneyRateLine mrLine;
    if (!mData.RateLines.TryGetValue("USD", out mrLine))
    {
        log("I can not find USD rate line!");
        return;
    }

    // Check conditions
    // CNB does not make a new rate each day
    if (!mrLine.isThereANextDayRate) // it means something like
        // !(mrLine.getLastHistDate() == mrLine.getLastBetDate())
    {
        log("There is no new data!");
    }
}

```

```

        return;
    }

    // If rate line is too short
    if (days > mrLine.HistRates.Count)
    {
        log("There is no enough data");
        return;
    }

    // make avangare rate into 'avg' variable
    decimal val = 0;
    int dayId = Math.Max((int)(mrLine.HistRates.Count - days), 0);
    while (dayId < mrLine.HistRates.Count)
        val += mrLine.HistRates[dayId++];

    // save 'avg'
    mrLine.NextBet = val / days;
}

```

Historie se opakuje

Ideou tohoto pluginu je opakující se historie. V ideálním případě se nám podaří najít v historii řadu hodnot, která je úplně stejná jako ta, která vznikla za několik posledních dní. Potom lze předpokládat, že se kurz bude i nadále vyvíjet stejně, jako tomu bylo naposledy a následující hodnotu můžeme odhadnout hodnotou, která následovala řadě historických hodnot.

Kód pluginu je v textu 3, podrobnější popis implementace následuje za ním.

Text 3: Kód hlavní funkce pluginu založeného na opakující se historii

```

public uint days = 20;
public decimal tolerance = 0.600m;

public override void run(PackBase pack)
{
    MoneyPack mPack = (MoneyPack)pack;

    MoneyData mData;
    if (!mPack.Money.TryGetValue("CZK", out mData))
    {
        log("I can not find CZK!");
        return;
    }

    MoneyRateLine mrLine;
    if (!mData.RateLines.TryGetValue("USD", out mrLine))
    {
        log("I can not find USD rate line!");
        return;
    }

    if (!mrLine.isThereANextDayRate)
        return;

    // If rate line is too short
    int rl = mrLine.HistRates.Count;

```

```

    if (2*days > rl)
        return;

    uint seq_best_len = 0;
    uint seq_best_start = 0;
    uint seq_best_end = 0;

    decimal avg = 0;
    decimal[] seq = new decimal[days];
    for (int i = 0; i < days; i++)
    {
        seq[i] = mrLine.HistRates[mrLine.HistRates.Count - i - 1];
        avg += seq[i];
    }
    avg /= days;

    for (uint i = 0; i < mrLine.HistRates.Count - 2; i++)
    {
        uint len = 0;
        for (uint j = 0; j < days; j++)
        {
            if (i - j < 0)
                break;

            decimal val = mrLine.HistRates[(int)(i - j)];
            if (!((val >= seq[j] - tolerance) && (val <= seq[j] + tolerance)))
                break;
            len++;
        }

        if ((len >= seq_best_len) && (len > 0))
        {
            seq_best_len = len;
            seq_best_start = i - len;
            seq_best_end = i;
        }
    }

    // average is used if no sequence is found
    mrLine.NextBet = (seq_best_len > 0) && (seq_best_start > 1) ?
        mrLine.HistRates[(int)(seq_best_start - 1)] : avg;
}

```

Najít přesně stejnou řadu v historii, jako nastala v posledních dnech, ale nemusí být vždy možné. Proto je zde určitá míra tolerance vyjádřená jediným číslem, které určuje, o kolik se může nalezená hodnota lišit od hodnoty hledané pro každé číslo z řady. Konkrétně hledáme nejdelší řadu takovou, která je podobná té, co se stala v posledních dnech. Podobná znamená, že žádná ze dvou odpovídajících si čísel v řadách se neliší o více, než je dáno tolerancí. Délka řady je omezena proměnnou. Pokud se najde více řad stejné délky (maximální nalezené délky), které jsou v toleranci, vybere se ta, která nastala chronologicky nejpozději.

Kříží se proměnné obsahující hodnoty tolerance a maximální požadované délky řady.

Zrcadlení

Když se pozorně zadíváme na křivky vývoje kurzů měn, zda se, že se v nich dá většinou nalézt alespoň náznak symetrie. Pro ilustraci slouží obrázek 1. Červeně je vyznačen bod, který lze pokládat za střed symetrie, zeleně pak zrcadlíci se body. Takovýchto bodů je na křivce velmi mnoho, zdá se, že pro téměř každý bod lze naleznout jeho zrcadlový bod. Některé body jsou spolu souměrné středově podle středového bodu, jiné osově podle osy určené středovým bodem, kolmé k ose y grafu. Na této myšlence je postavena následující strategie. Kód strategie



Obrázek 1: Ilustrační křivka popisující vývoj kurzu

je napsán v textu 4, jeho popis následuje pod ním.

Text 4: Kód hlavní funkce pluginu založeného hledání symetrie

```
public uint interval_len = 20;

// A new bet is average rate from the last few days
public override void run(PackBase pack)
{
    log("Instance started");
    decimal middle_value, first_value, last_value, hint_value, bet, difference;

    // Get money pack
    MoneyPack mPack = (MoneyPack)pack;
```

```

// Get rate lines between CZK and EUR
// Get money lines from CZK
MoneyData mData;
if (!mPack.Money.TryGetValue("CZK", out mData))
{
    log("I can not find CZK!");
    return;
}

// Get money lines to EUR
MoneyRateLine mrLine;
if (!mData.RateLines.TryGetValue("EUR", out mrLine))
{
    log("I can not find EUR rate line!");
    return;
}

// Check conditions
// CNB does not make a new rate each day
if (!mrLine.isThereANextDayRate) // it means something like
    (mrLine.getLastHistDate() == mrLine.getLastBetDate())
{
    log("No new data");
    return;
}

// If rate line is too short
if (interval_len > mrLine.HistRates.Count)
{
    log("Not enough of data");
    return;
}

// Find middle value
List<decimal> values = mrLine.HistRates.GetRange(
    (int)(mrLine.HistRates.Count - interval_len), (int)interval_len);
hint_value = values[0];
first_value = values[1];
middle_value = values[(int)(interval_len / 2)];
last_value = values[(int)(interval_len - 1)];

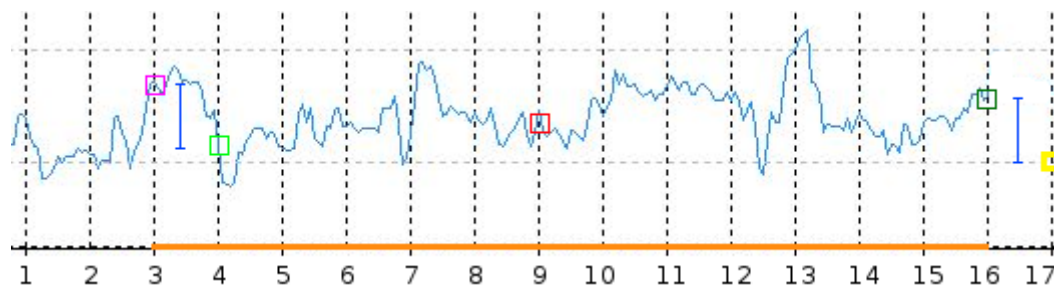
if ((first_value <= middle_value && last_value <= middle_value) ||
    (first_value >= middle_value && last_value >= middle_value))
{
    difference = hint_value - first_value;
    bet = last_value + difference;
}
else
{
    difference = hint_value - first_value;
    bet = last_value - difference;
}

mrLine.NextBet = bet;

log("done!");
}

```

Strategie má daný interval (křížící se proměnná), který udává délku úseku, po kterém se začínají data nějakým způsobem zrcadlit. Na obrázku 2 je vyznačen oranžovou barvou a je dlouhý 14 dní (v pluginu 20). Obrázek znázorňuje situaci, kdy chceme odhadnout kurz na den číslo 17. Strategie si uloží data kurzů z tohoto intervalu posledních dní a vyplní hodnoty několika proměnných:



Obrázek 2: Ilustrace pro osvětlení fungování strategie založené na zrcadlení

- First value je na obrázku znázorněna světle zelenou barvou, je to hodnota v poli intervalu s pořadovým číslem 1 (počítáno od nuly).
- Last value je na obrázku zvýrazněna tmavě zeleně a je to poslední známá hodnota („včerejší“).
- Middle value je hodnota, kterou kurz nabývá uprostřed intervalu, na obrázku značená červeně.
- Hint value je hodnota na počátku intervalu (nultá), na obrázku růžově.

Předpokládá se, že first value a last value se zrcadlí podle middle value. A tedy potom hodnota, kterou chceme odhadnout se zrcadlí podle middle value s hint value. První předpoklad ovšem nemusí vždy platit a tak je strategie napsána opatrněji – je založena na tom, že se zrcadlí rozdíl mezi hint value a first value do rozdílu mezi last value a budoucí hodnotou. Toto je na obrázku znázorněno modrými úsečkami a odhadnutá hodnota je nakreslena žlutě.

Ještě zbývá ujasnit, kdy se spočítaný rozdíl promítá středově souměrně a kdy osově. Pokud jsou obě hodnoty (first value a last value) pod/nad hodnotou middle value, zrcadlí se osově, jinak středově, viz. obrázek 3.



Obrázek 3

Podle vzorů

Strategie odhaduje vývoj kurzu mezi měnami EUR a USD, vývoj kurzu mapuje do několika vzorů a na základě četnosti jednotlivých nalezených vzorů odhaduje příští změnu kurzu.

Strategie projde historii vývoje kurzu za úsek daný proměnnou `daysBack` (případně menší, pokud není dostatek dat). Každá pozitivní změna kurzu (vzrůst kurzu mezi po sobě jdoucími údaji) je zaznamenána, stejně tak pro negativní změny kurzu. Pak je spočítán medián pozitivních a negativních změn kurzu.

Na základě výsledků lze pak každou změnu kurzu zařadit do jedné ze čtyř kategorií:

- významná pozitivní změna
- pozitivní změna
- negativní změna
- významná negativní změna.

Interně se tyto kategorie hodnotí čísly 0 až 3.

V druhém průchodu historií vývoje kurzu jsou brány pětky po sobě jdoucích hodnot kurzů, získají se tak čtyři změny kurzu mezi nimi a tyto údaje jsou zařazeny do výše uvedených čtyř kategorií. Strategie počítá, jak často se v historii kurzu objevila jaká kombinace čtyř po sobě jdoucích změn kurzu. Existuje tedy celkem 4^4 (256) možných "vzorů" vývoje kurzu během po sobě jdoucích čtyř dní.

Následně jsou zjištěny poslední tři změny kurzu a pro každou ze čtyř možných čtvrtých změn je načtena statistika její četnosti.

Například pokud jsou poslední tři změny kurzu z kategorií postupně 1, 3, 0, jsou zjištěny četností vzorů 1300, 1301, 1302 a 1303. Jako vítězný vzor je vybrán ten s nejvyšší četností.

Pro vítězný vzor se zjistí odpovídající kategorie a odpovídající hodnota změny pro danou kategorii – jako medián změny v dané kategorii.

Například pro kategorii "velká pozitivní změna" je odpovídající hodnota 3. kvartil souboru pozitivních změn.

Tato výsledná hodnota je přičtena k poslednímu záznamu historie kurzu a je uložena jako tip strategie na příští hodnotu kurzu.

Kód strategie následuje v textu 5.

Text 5: Kód strategie založené na hledání vzorů

```
enum StepCategory
{
    UPLARGE = 0,
    UPSMALL = 1,
    DOWNSMALL = 2,
    DOWNLARGE = 3
}

struct StepCategorizer
{
    public StepCategorizer(double negative_median, double median,
        double positive_median)
    {
```

```

        this.negative_median = negative_median;
        this.median = median;
        this.positive_median = positive_median;
    }

    public StepCategory categorizeStep(double step)
    {
        if (step > positive_median) return StepCategory.UPLARGE;
        else if (step > median) return StepCategory.UPSMALL;
        else if (step > negative_median) return StepCategory.DOWNSMALL;
        else return StepCategory.DOWNLARGE;
    }

    private double negative_median, median, positive_median;
}

public uint daysBack = 600;
private uint getPatternCode(StepCategory c1, StepCategory c2, StepCategory c3,
    StepCategory c4)
{
    uint v1 = (uint)c1;
    uint v2 = (uint)c2;
    uint v3 = (uint)c3;
    uint v4 = (uint)c4;
    return v1 * 4 * 4 * 4 + v2 * 4 * 4 + v3 * 4 + v4;
}

public override void run(PackBase pack)
{
    log("Jan Money Strategy commencing");
    MoneyPack mPack = (MoneyPack)pack;

    MoneyData mData;
    if (!mPack.Money.TryGetValue("EUR", out mData))
    {
        log("Error: Cannot find EUR.");
        return;
    }

    MoneyRateLine mrLine;
    if (!mData.RateLines.TryGetValue("USD", out mrLine))
    {
        log("Error: Cannot find EUR-USD rate line.");
        return;
    }

    if (!mrLine.isThereANextDayRate)
    {
        log("No next day rate");
        return;
    }

    List<decimal> rates = mrLine.HistRates;
    int valuesCount = rates.Count;

    if (valuesCount < 5)
    {
        log("Not enough data in history");
        return; // not enough data
    }
}

```

```

    }

    int fromPoint = Math.Max(valuesCount - (int) daysBack, 0);

    List<double> positive_steps = new List<double>();
    List<double> negative_steps = new List<double>();

    for (int i = fromPoint; i < valuesCount-1; i++)
    {
        decimal rate1 = rates[i];
        decimal rate2 = rates[i + 1];
        double diff = (double) (rate2 - rate1);
        if (diff > 0)
        {
            positive_steps.Add(diff);
        }
        else
        {
            negative_steps.Add(diff);
        }
    }

    positive_steps.Sort();
    negative_steps.Sort();

    // not enough data checks
    if (positive_steps.Count == 0) return;
    if (negative_steps.Count == 0) return;

    double middle = 0;
    double positive_median = positive_steps[positive_steps.Count / 2];
    double negative_median = negative_steps[negative_steps.Count / 2];
    log(string.Format("Positive median: {0}, negative median: {1}",
        positive_median, negative_median));

    StepCategorizer categorizer = new StepCategorizer(negative_median, middle,
        positive_median);
    Dictionary<uint, uint> patternCounts = new Dictionary<uint, uint>();

    for (int i = fromPoint; i < valuesCount - 4; i++)
    {
        StepCategory step1 = categorizer.categorizeStep(
            (double) (rates[i + 1] - rates[i + 0]));
        StepCategory step2 = categorizer.categorizeStep(
            (double) (rates[i + 2] - rates[i + 1]));
        StepCategory step3 = categorizer.categorizeStep(
            (double) (rates[i + 3] - rates[i + 2]));
        StepCategory step4 = categorizer.categorizeStep(
            (double) (rates[i + 4] - rates[i + 3]));

        uint patternCode = getPatternCode(step1, step2, step3, step4);

        if (!patternCounts.ContainsKey(patternCode))
        {
            patternCounts.Add(patternCode, 0);
        }

        patternCounts[patternCode]++;
    }

```

```

StepCategory lastStep1 = categorizer.categorizeStep(
    (double)(rates[valuesCount - 3] - rates[valuesCount - 4]));
StepCategory lastStep2 = categorizer.categorizeStep(
    (double)(rates[valuesCount - 2] - rates[valuesCount - 3]));
StepCategory lastStep3 = categorizer.categorizeStep(
    (double)(rates[valuesCount - 1] - rates[valuesCount - 2]));

uint pattern1 = getPatternCode(lastStep1, lastStep2, lastStep3,
    StepCategory.UPLARGE);
uint pattern2 = getPatternCode(lastStep1, lastStep2, lastStep3,
    StepCategory.UPSMALL);
uint pattern3 = getPatternCode(lastStep1, lastStep2, lastStep3,
    StepCategory.DOWNSMALL);
uint pattern4 = getPatternCode(lastStep1, lastStep2, lastStep3,
    StepCategory.DOWNLARGE);

uint pcount1 = patternCounts.ContainsKey(pattern1) ? patternCounts[pattern1] : 0;
uint pcount2 = patternCounts.ContainsKey(pattern2) ? patternCounts[pattern2] : 0;
uint pcount3 = patternCounts.ContainsKey(pattern3) ? patternCounts[pattern3] : 0;
uint pcount4 = patternCounts.ContainsKey(pattern4) ? patternCounts[pattern4] : 0;

log(string.Format(
    "Pattern counts: UPLARGE {0}, UPSMALL {1}, DOWNSMALL {2}, DOWNLARGE {3}",
    pcount1, pcount2, pcount3, pcount4));

double step = 0;

if (pcount1 == 0 && pcount2 == 0 && pcount3 == 0 && pcount4 == 0)
{
    log("No pattern won");
    // step = 0;
}
else if (pcount1 >= pcount2 && pcount1 >= pcount3 && pcount1 >= pcount4)
{
    // median of the higher half of positive steps - "average
    // large positive step"
    log("UPLARGE won");
    step = positive_steps[(int) (positive_steps.Count * 0.75)];
}
else if (pcount2 >= pcount1 && pcount2 >= pcount3 && pcount2 >= pcount4)
{
    log("UPSMALL won");
    step = positive_steps[(int) (positive_steps.Count * 0.25)];
}
else if (pcount3 >= pcount1 && pcount3 >= pcount2 && pcount3 >= pcount4)
{
    log("DOWNSMALL won");
    step = negative_steps[(int) (negative_steps.Count * 0.75)];
}
else if (pcount4 >= pcount1 && pcount4 >= pcount2 && pcount4 >= pcount3)
{
    // median of the lower half of negative steps - "average large
    // negative step"
    log("DOWNLARGE won");
    step = negative_steps[(int) (negative_steps.Count * 0.25)];
}
else
{

```

```

        log("Plugin assertion error, incomparable pattern counts.");
        // step = 0;
    }

    log(string.Format("Predicting change of currency by {0}", step));
    mrLine.NextBet = rates[valuesCount - 1] + ((decimal)step);
}

```

Kurzy vypisované na sportovní utkání

Zdrojový kód každé strategie je uveden přímo zde, což umožňuje číst si ho bez přístupu k počítači, ale během programování se hodí mít ho k dispozici např. pro účely kopírování některých pasáží, proto jsou všechny kódy umístěny také na webu: <http://navarin.ms.mff.cuni.cz:7070/examples.php>.

Průměrovací strategie

Tato strategie se snaží odhadovat kurzy na zápasy v rámci 1. anglické ligy. Její idea je velmi jednoduchá. Pro každý zápas, pro který odhaduje kurzy, provede následující operace:

Vytvoří a vynuluje si pět čítačů – pro každou část vypisovaného kurzu jeden.

Projde všechny odehrané zápasy v rámci soutěže a vybírá si ty, kterých se účastnil alespoň jeden z daných týmů. Pokud v takovém zápase hrál nyní hostující tým také jako hostující, kurzy na tento zápas se jednoduše přičtou do čítačů kurzů, pokud hráli současní hosté jako domácí, pak se kurzy prohází a opět přičtou (např. kurz na výhru hostů přičteme ke kurzu na výhru domácích atd.). Analogicky pro tým domácích. Průběžně sčítáme, za kolik zápasů jsme kurzy sečetli.

Když je tato část hotova, spočítáme průměrný kurz (máme součty i počet) a ten uložíme jako náš odhad.

Text 6: Kód hlavní funkce pluginu strategie počítající průměry kurzů na vybrané zápasy

```

public override void run(PackBase pack)
{
    SportPack sPack = (SportPack)pack;
    CompetitionData competition = sPack.getCompetition(
        "FOTBAL", "FOTBAL MUZI", "ANGLICKÁ 1. LIGA MUZI");
    if (competition == null)
    {
        log("no data for 1. anglicka liga muzi, fotbal");
        return;
    }

    log("starting...");

    foreach (CompetitionYearData cyData in competition.CompetitionYears.Values)
    {
        foreach (MatchData futureMatch in cyData.FutureMatches)
        {
            decimal[] opportunities;
            opportunities = new decimal[5];

            uint opportunities_cnt = 0;

```

```

        for (int i=0; i<5; i++)
        {
            opportunities[i] = 0;
        }
        foreach (MatchData match in cyData.PlayedMatches)
        {
            if ((match.HomeTeam == futureMatch.HomeTeam) ||
                (match.VisitTeam == futureMatch.VisitTeam))
            {
                for (int i=0; i<5; i++)
                {
                    opportunities[i] += match.Opportunities[i];
                }
                opportunities_cnt++;
            }
            else
            {
                opportunities[0] += match.Opportunities[4];
                opportunities[1] += match.Opportunities[3];
                opportunities[2] += match.Opportunities[2];
                opportunities[3] += match.Opportunities[1];
                opportunities[4] += match.Opportunities[0];
                opportunities_cnt++;
            }
        }

        for (int i=0; i<5; i++)
        {
            if (opportunities_cnt != 0)
            {
                opportunities[i] = opportunities[i] / opportunities_cnt;
                futureMatch.Bets[i] = opportunities[i];
            }
        }
    }
    log("done");
}

```

Strategie založená na odhadech skóre zápasu

Tato strategie patří myšlenkově mezi ty složitější. Strategie sice samozřejmě odhaduje vypsané kurzy, nicméně k těmto odhadům historické kurzy vůbec nevyužívá. Snaží se odhadnout výsledek zápasu (poměrně smysluplně) a z výsledku se pak snaží spočítat vhodné kurzy. Text 7 ukazuje tělo hlavní funkce, myšlenka výpočtu je podrobněji rozebrána pod textem.

Text 7: Kód hlavní funkce pluginu strategie počítající kurzy z tipovaných výsledků zápasu

```

public override void run(PackBase pack)
{
    SportPack sPack = (SportPack)pack;

    CompetitionData competition = sPack.getCompetition(
        "FOTBAL", "FOTBAL MUZI", "ANGLICKÁ 1. LIGA MUZI");
    if (competition == null)
    {
        log("\"FOTBAL\" \"FOTBAL MUZI\" \"ANGLICKÁ 1. LIGA MUZI\" could not found");
    }
}

```

```

        return;
    }

    foreach (MatchData futureMatch in competition.FutureMatches)
    {
        Console.WriteLine("{0}: {1} vs. {2}", futureMatch.Time,
            futureMatch.HomeTeam.Title.ToString(),
            futureMatch.VisitTeam.Title.ToString());
        int homeTeamScored = 0;
        int homeTeamReceived = 0;
        int homeTeamWin = 0;
        int homeTeamTie = 0;
        int homeTeamLose = 0;

        int visitTeamScored = 0;
        int visitTeamReceived = 0;
        int visitTeamWin = 0;
        int visitTeamTie = 0;
        int visitTeamLose = 0;

        foreach (MatchData match in competition.PlayedMatches)
        {
            if (futureMatch.HomeTeam == match.HomeTeam)
            {
                homeTeamScored += match.HomeScore;
                homeTeamReceived += match.VisitScore;
                if ((match.HomeScore == match.VisitScore) || match.Prolongation)
                    homeTeamTie++;
                else if (match.HomeScore > match.VisitScore)
                    homeTeamWin++;
                else homeTeamLose++;
            }
            else if (futureMatch.HomeTeam == match.VisitTeam)
            {
                homeTeamScored += match.VisitScore;
                homeTeamReceived += match.HomeScore;
                if ((match.HomeScore == match.VisitScore) || match.Prolongation)
                    homeTeamTie++;
                else if (match.HomeScore > match.VisitScore)
                    homeTeamLose++;
                else homeTeamWin++;
            }
            if (futureMatch.VisitTeam == match.HomeTeam)
            {
                visitTeamScored += match.HomeScore;
                visitTeamReceived += match.VisitScore;
                if ((match.HomeScore == match.VisitScore) || match.Prolongation)
                    visitTeamTie++;
                else if (match.HomeScore > match.VisitScore)
                    visitTeamWin++;
                else visitTeamLose++;
            }
            else if (futureMatch.VisitTeam == match.VisitTeam)
            {
                visitTeamScored += match.VisitScore;
                visitTeamReceived += match.HomeScore;
                if ((match.HomeScore == match.VisitScore) || match.Prolongation)
                    visitTeamTie++;
            }
        }
    }
}

```

```

        else if (match.HomeScore > match.VisitScore)
            visitTeamLose++;
        else visitTeamWin++;
    }
}

int homeTeamMatches=homeTeamWin+homeTeamTie+homeTeamLose;
int visitTeamMatches=visitTeamWin+visitTeamTie+visitTeamLose;

Console.WriteLine("Team {0} (home) won {1}, tied {2} and lost {3} matches ({4}
total), scored {5} and received {6} goals",
    futureMatch.HomeTeam.Title.ToString(), homeTeamWin, homeTeamTie,
    homeTeamLose, homeTeamMatches, homeTeamScored, homeTeamReceived);
Console.WriteLine("Team {0} (visit) won {1}, tied {2} and lost {3} matches ({4}
total), scored {5} and received {6} goals",
    futureMatch.VisitTeam.Title.ToString(), visitTeamWin, visitTeamTie,
    visitTeamLose, visitTeamMatches, visitTeamScored, visitTeamReceived);
if ((homeTeamMatches > 2) && (visitTeamMatches > 2))
{
    decimal homeSPM = (decimal)homeTeamScored / (decimal)homeTeamMatches;
    decimal homeRPM = (decimal)homeTeamReceived / (decimal)homeTeamMatches;
    decimal visitSPM = (decimal)visitTeamScored / (decimal)visitTeamMatches;
    decimal visitRPM = (decimal)visitTeamReceived / (decimal)visitTeamMatches;

    Console.WriteLine("Team {0} SPM: {1:0.00}, RPM: {2:0.00}",
        futureMatch.HomeTeam.Title.ToString(), homeSPM, homeRPM);
    Console.WriteLine("Team {0} SPM: {1:0.00}, RPM: {2:0.00}",
        futureMatch.VisitTeam.Title.ToString(), visitSPM, visitRPM);

    decimal predictHomeScore = homeSPM * visitRPM;
    decimal predictVisitScore = visitSPM * homeRPM;
    Console.WriteLine("Predicted score: {0:0.00}:{1:0.00}", predictHomeScore,
        predictVisitScore);

    decimal homeWinWeight = 0;
    decimal homeTieWeight = 0;
    decimal homeLoseWeight = 0;

    for (int goalsHome = 0; goalsHome < 10; goalsHome++)
    {
        for (int goalsVisit = 0; goalsVisit < 10; goalsVisit++)
        {
            decimal scoreHomeDiff = goalsHome - predictHomeScore;
            decimal scoreHomeDiffSq = scoreHomeDiff * scoreHomeDiff;
            decimal scoreVisitDiff = goalsVisit - predictVisitScore;
            decimal scoreVisitDiffSq = scoreVisitDiff * scoreVisitDiff;
            decimal diffSum = scoreHomeDiffSq + scoreVisitDiffSq;
            decimal scoreWeight = diffSum == 0 ? 1 : 1 / diffSum;

            if (goalsHome == goalsVisit) homeTieWeight += 2*scoreWeight;
            else if (goalsHome > goalsVisit) homeWinWeight += scoreWeight;
            else homeLoseWeight += scoreWeight;
        }
    }

    decimal totalWeight = homeWinWeight + homeTieWeight + homeLoseWeight;
    Console.WriteLine("Home team win/tie/lose weights: {0:0.00}/{1:0.00}/{2:0.00}
({3:0.00} total weight)",
        homeWinWeight, homeTieWeight, homeLoseWeight, totalWeight);
}

```

```

decimal homeWinChance = homeWinWeight / totalWeight;
decimal homeTieChance = homeTieWeight / totalWeight;
decimal homeLoseChance = homeLoseWeight / totalWeight;

Console.WriteLine(
    "Home team win/tie/lose chances: {0:0.00}/{1:0.00}/{2:0.00}",
    homeWinChance, homeTieChance, homeLoseChance);

decimal homeWinBet = 0.9m / homeWinChance;
decimal homeTieBet = 0.9m / homeTieChance;
decimal homeLoseBet = 0.9m / homeLoseChance;

if (homeWinBet < 1.05m) homeWinBet = 1.05m;
if (homeTieBet < 1.05m) homeTieBet = 1.05m;
if (homeLoseBet < 1.05m) homeLoseBet = 1.05m;

Console.WriteLine(
    "Home team win/tie/lose bets: {0:0.00}/{1:0.00}/{2:0.00}",
    homeWinBet, homeTieBet, homeLoseBet);

futureMatch.Bets[0] = homeWinBet;
futureMatch.Bets[1] = 0.9m / (0.9m / homeWinBet + 0.9m / homeTieBet);
futureMatch.Bets[2] = homeTieBet;
futureMatch.Bets[3] = 0.9m / (0.9m / homeLoseBet + 0.9m / homeTieBet);
futureMatch.Bets[4] = homeLoseBet;

Console.WriteLine("Home team win/wt/tie/tl/lose bets: {0:0.00}/{1:0.00}/
{2:0.00}/{3:0.00}/{4:0.00}",
    futureMatch.Bets[0], futureMatch.Bets[1], futureMatch.Bets[2],
    futureMatch.Bets[3], futureMatch.Bets[4]);
    }
}
}
}
}

```

Strategie si najde zápasy, na které chce uhodnout výsledek a pro každý takový zápas projde následující procedurou. Nechť právě odhadovaný zápas hrají proti sobě týmy A a B. Najde všechny odehrané zápasy, kterých se účastnil alespoň jeden z týmů A nebo B. Z každého připočte na konto dotčeného týmu: kolik dal gólů, kolik jich inkasoval a zda tým zvítězil, remizoval, či prohrál. Na konci této akce máme tedy k obou týmům (ke každému zvlášť):

- součet gólů, které daly
- součet gólů, které inkasovaly
- kolik zápasů vyhrály
- kolik zápasů remizovaly
- kolik zápasů prohrály

Pokud jeden nebo oba z týmů neodehrály ještě ani tři zápasy, plugin tipovat odmítá. Pokud již ale oba alespoň tři uhrály, pokračuje dál.

Z nasbíraných dat se spočte, kolik dal tým průměrně na zápas gólů a kolik inkasoval. Následuje tipování skóre zápasu a to tak, že skóre týmu A je odhadnuto jako průměrný počet gólů, které dá za zápas vynásobený průměrným počtem gólů, které tým B za zápas inkasuje. Tím se v odhadu skóre potká síla v útoku týmu A a zároveň schopnost obrany týmu B.

Analogicky se počítá odhad uhraných bodů pro tým B.

V poslední fázi se strategie snaží spočítat pravděpodobnosti všech možných různých výsledků zápasu a z nich určit vhodný kurz. Pravděpodobnosti výsledků, kde jeden tým získá více, než devět bodů, zanedbává. Strategie si vynuluje tři čítače: jeden pro případ výhry týmu A, druhý pro remízu a třetí pro případ výhry týmu B. Do čítačů se sčítají příslušné váhy odchylek možných výsledků od odhadu výsledku zápasu. Tedy např. váha odchylky pro skóre zápasu 1:4 za zapíše do čítače pro výhru týmu B, váha odchylky 2:2 do čítače pro remízu a váha odchylky 3:2 do čítače výhry týmu A.

Odchylky se počítají pro všechna možná skóre (dvojice vzniklé kartézským součinem množiny skóre $\{0..9\}$ se sebou samotnou). Zbývá vyjasnit, co se rozumí odchylkou od odhadnutého skóre zápasu a její váhou. V předchozí fázi strategie odhadla výsledek zápasu, pojmenujme si toto odhadnuté skóre jako $a:b$. Odchylka skóre $x:y$ vůči $a:b$ se počítá tímto způsobem:

$$\text{odchylka} = (x-a)^2 + (y-b)^2$$

Váhu získáme tak, že pokud se jednalo o odchylku velikosti 0, je váha jednička. V jiném případě má váha hodnotu $\frac{1}{\text{odchylka}}$. Pokud $x=y$, tedy počítáme odchylku remízy, je hodnota váhy ještě zdvojnásobena.

Ve finále jsou sečteny hodnoty všech tří čítačů a spočítány pravděpodobnosti. Pravděpodobnost výhry týmu A je spočtena jako hodnota příslušného čítače vydělená součtem všech čítačů. Analogicky je spočtena odhadnutá pravděpodobnost remízy a výhry týmu B. Nyní tedy známe odhady pravděpodobností remízy, výher i ne-proher obou týmů.

Kurzy jsou pak odhadnuty tak, jak by je odhadovala sázková kancelář, která se snaží o 10% zisk: číslo 0.9 se vydělí příslušnou pravděpodobností výsledku zápasu.

Průměrovací strategie (č. 2)

Tato strategie opět počítá odhady kurzů z průměrů jiných zápasů, ale složitějším způsobem. Kurz je tipován, pokud se oba týmy odhadovaného zápasu účastnily již alespoň dvou předchozích zápasů. Kurzy na výhru domácích a hostů se počítají stejně, jako je počítala první strategie založená na průměrování. Kurzy na ne-prohry jsou založeny na následujícím faktu: součet pravděpodobností na výhru prvního týmu, remízu a výhru druhého týmu by měl být 1. A když k tomu ještě víme všechny tyto tři jednotlivé pravděpodobnosti, je snadné spočítat pravděpodobnost ne-prohry (ne-prohra prvního týmu a výhra druhého týmu nám také dají součet jedna).

Text 8: Kód hlavní funkce pluginu strategie počítající kurzy průměrů

```
public override void run(PackBase pack)
{
    SportPack sPack = (SportPack)pack;

    CompetitionData competition = sPack.getCompetition("HOKEJ", "HOKEJ MUZI", "NHL MUZI");
    if (competition == null)
    {
        log("\Hokej\" \"HOKEJ MUZI\" \"NHL MUZI\" could not found");
        return;
    }
}
```

```

foreach (MatchData futureMatch in competition.FutureMatches)
{
    decimal favoritK = 0;
    int cntFavoritK = 0;
    decimal visitK = 0;
    int cntVisitK = 0;
    {
        foreach (MatchData match in competition.PlayedMatches)
        {
            if (match.HomeTeam == futureMatch.HomeTeam)
            {
                ++cntFavoritK; favoritK += match.Opportunities[0];
            }
            else if (match.VisitTeam == futureMatch.HomeTeam)
            {
                ++cntFavoritK; favoritK += match.Opportunities[4];
            }
            else if (match.HomeTeam == futureMatch.VisitTeam)
            {
                ++cntVisitK; visitK += match.Opportunities[0];
            }
            else if (match.VisitTeam == futureMatch.VisitTeam)
            {
                ++cntVisitK; visitK += match.Opportunities[4];
            }
        }

        // Pokud existuje vice nez 2 tipy
        if (cntFavoritK >= 2 && cntVisitK >= 2)
        {
            Console.WriteLine(futureMatch);
            favoritK /= cntFavoritK;
            visitK /= cntVisitK;
            decimal favoritP = 0.9m / favoritK;
            decimal visitP = 0.9m / visitK;
            decimal remizaP = 1.0m - favoritP - visitP;
            futureMatch.Bets[0] = favoritK;
            futureMatch.Bets[1] = 0.9m / (visitP + remizaP);
            futureMatch.Bets[2] = 0.9m / remizaP;
            futureMatch.Bets[3] = 0.9m / (visitP + remizaP);
            futureMatch.Bets[4] = visitK;
        }
    }
}

```

TableRank

Strategie sází na sportovní výsledky basketbalu, ligy NBA.

Po spuštění projde strategie všechny existující výsledky odehraných her a pro každý tým počítá počet odehraných her, počet vyhraných her a celkový rozdíl ve skóre vůči soupeři. Následně strategie seřadí týmy podle počtu vyhraných her vztaženému k počtu celkem odehraných her. Pro každý budoucí zápas je týmu hrajícímu v tomto zápase přiřazena hodnota odpovídající jeho pořadí v seřazené tabulce týmů. Např. celkově druhý tým v tabulce o třiceti týmech dostane hodnotu 28.

Hodnota favorita je dále zvýšena o koeficient získaný následující heuristikou.

Je zjištěn očekávaný rozdíl ve výsledném skóre zápasu jako rozdíl průměrného rozdílu skóre obou týmů. Tento rozdíl je následně vztažen k základnímu rozdílu ve skóre, jež se nachází v geneticky křížené proměnné `scoreDiffBase`.

Výsledné hodnoty obou týmů jsou následně převedeny na pravděpodobnost výhry (hodnota týmu vydělená celkovou hodnotou obou), obrácená hodnota těchto pravděpodobností je pak výsledný vypsaný kurz výhry domácího týmu, resp. hostí. Kurzy ne-proher týmů jsou strategií vždy nastaveny na stejnou hodnotu jako kurzy výhry, kurz remízy není strategií vypisován.

Text 9: strategie TableRank

```
class TeamStats : IComparable<TeamStats>
{
    public TeamStats(int id_, uint wins_, int scoreDiffs_, uint totalMatches_)
    {
        id = id_;
        wins = wins_;
        scoreDiff = scoreDiffs_;
        totalMatches = totalMatches_;
    }

    public int CompareTo(TeamStats them)
    {
        int wonRatioComparision = this.wonRatio.CompareTo(them.wonRatio);
        if (wonRatioComparision == 0)
        {
            return this.scoreDiffPerMatch.CompareTo(them.scoreDiffPerMatch);
        }
        else return wonRatioComparision;
    }

    public void calcFinalStats()
    {
        wonRatio = (totalMatches > 0) ? (((double)wins) / ((double)totalMatches)) : 0;
        scoreDiffPerMatch = (totalMatches > 0) ?
            (((double)wins) / ((double)scoreDiff)) : 0;
    }

    public uint wins;
    public int scoreDiff;
    public double scoreDiffPerMatch;
    public uint totalMatches;
    public double wonRatio;
    public uint rank;
    public int id;
}

public class Plugin : PluginBase
{
    public uint scoreDiffBase = 40;
    public double bonusRankCoef = 0.1;

    private void accumulateStats(TeamStats stats, int ourScore, int theirScore)
    {
        stats.totalMatches++;
        stats.scoreDiff += (ourScore - theirScore);
        if (ourScore > theirScore)
```

```

        {
            stats.wins++;
        }
    }

    private void accountResult(Dictionary<int, TeamStats> teamStats, int ourId,
        int ourScore, int theirScore)
    {
        TeamStats stats;

        if (!teamStats.ContainsKey(ourId))
        {
            stats = new TeamStats(ourId, 0, 0, 0);
            teamStats.Add(ourId, stats);
        }
        else
        {
            stats = teamStats[ourId];
        }

        accumulateStats(stats, ourScore, theirScore);
    }

    public override void run(PackBase pack)
    {
        SportPack sPack = (SportPack)pack;

        CompetitionData competition = sPack.getCompetition("BASKETBAL", "BASKETBAL MUZI",
            "NBA MUZI");
        if (competition == null)
        {
            log("Competition 'NBA MUZI' was not found");
            return;
        }

        log("Jan Sport commencing...");

        Dictionary<int, TeamStats> teamStats = new Dictionary<int,TeamStats>();

        // calc stats
        foreach (MatchData match in competition.PlayedMatches)
        {
            accountResult(teamStats, match.HomeTeam.Id, match.HomeScore,match.VisitScore);
            accountResult(teamStats, match.VisitTeam.Id,match.VisitScore,match.HomeScore);
        }

        int totalTeams = teamStats.Count;
        // order by ratio of won matches
        TeamStats[] statsArray = new TeamStats[totalTeams];

        uint i = 0;
        foreach (KeyValuePair<int, TeamStats> dicEntry in teamStats)
        {
            dicEntry.Value.calcFinalStats();
            statsArray[i++] = dicEntry.Value;
        }

        Array.Sort(statsArray);
    }

```

```

// back to dictionary, assign ranks
uint i2 = 0;
foreach (TeamStats stat in statsArray)
{
    stat.rank = i2++;
}

log(string.Format("Lowest ranked team: {0}, Highest ranked team: {1}",
    statsArray[0].id, statsArray[statsArray.Length - 1].id));

log(string.Format("Future matches count: {0}", competition.FutureMatches.Length));

foreach (MatchData futureMatch in competition.FutureMatches)
{
    int homeId = futureMatch.HomeTeam.Id;
    int visitId = futureMatch.VisitTeam.Id;
    if (!teamStats.ContainsKey(homeId) || !teamStats.ContainsKey(visitId))
    {
        log("Stats for one/both of the teams were not found");
        continue;
    }

    TeamStats homeStats = teamStats[homeId];
    TeamStats visitorStats = teamStats[visitId];

    double homeValue = homeStats.rank + (totalTeams * bonusRankCoef);
    double visitValue = visitorStats.rank + (totalTeams * bonusRankCoef);

    double expectedScoreDiff = Math.Abs(homeStats.scoreDiffPerMatch -
        visitorStats.scoreDiffPerMatch);
    double coef = expectedScoreDiff / scoreDiffBase;

    if (homeStats.scoreDiffPerMatch > visitorStats.scoreDiffPerMatch)
    {
        homeValue += homeValue * coef;
    }
    else
    {
        visitValue += visitValue * coef;
    }

    double totalValue = homeValue + visitValue;
    double homePercentage = homeValue / totalValue;
    double visitPercentage = visitValue / totalValue;

    futureMatch.Bets[0] = (decimal)(1 / homePercentage);
    futureMatch.Bets[4] = (decimal)(1 / visitPercentage);
    futureMatch.Bets[1] = (decimal)(1 / homePercentage);
    futureMatch.Bets[3] = (decimal)(1 / visitPercentage);
}
}
}

```

Používání webového rozhraní

Webové rozhraní naleznete na adrese <http://navarin.ms.mff.cuni.cz:7070/>. Přes toto rozhraní můžete nahrávat a spravovat své pluginy, prohlížet si jejich výsledky a dělat další související úkony, jako např. si prohlížet kurzy měn, výsledky zápasů a podobně.



MONEYMAKER

Domů O nás Měny Sporty Registrovat Přihlásit

Vítejte v systému MoneyMaker

Nejste přihlášen(a)

Chcete začít vydělávat na spekulacích, ale nechce se Vám celé dny sledovat situaci, odhadovat další vývoj a pak si vyčítat svá rozhodnutí? Napište si vlastní automatickou strategii! Ano, není to jednoduché, ale náš systém Vám pomůže. Umožní Vám Vaši strategii důkladně otestovat. Budete investovat nanečisto, tedy bez rizika. Výsledky Vaší strategie budete mít přehledně k dispozici. A hlavně: vaše strategie poběží v mnoha instancích s různými modifikacemi hodnot proměnných a bude geneticky křížena - pomůžeme Vám najít ty správné hodnoty, při kterých bude úspěšná.

Systém MoneyMaker je uzpůsoben pro co nejjednodušší použití a maximální přehlednost. Své strategie můžete vyvíjet hned ve třech jazycích: C#, Visual Basic nebo Delphi. V současné době můžete spekulovat nad vývojem kurzů měn nebo nad výsledky sportovních zápasů.

Ke stažení:

- Balíček obsahující BusinessDema aplikace: [BusinessDemo.zip](#)
- Balík statistických knihoven, které můžete použít ve vašem pluginu: [statisticalLibraries.zip](#)
- Příklady vysvětlené v dokumentaci ke stažení zde: [příklady](#)
- Stránka pro studenty je [zde \(zastarelé\)](#).

Copyright © MoneyMaker Team

Obrázek 4: Červenou šipkou jsou vyznačena tlačítka pro změnu jazyku

Při práci s webem si můžete nastavit jazyk, v jakém se mají texty zobrazovat. Webové rozhraní se pokusí na základě dat zaslaných prohlížečem odhadnout jazyk, který preferujete. Pokud se mu to ale nepovede a web se zobrazí v jiném jazyce, než je Vám příjemné, nastavení se provádí kliknutím na vlaječky v pravém horním rohu. Pro český jazyk klikněte na českou vlaječku, pro angličtinu na vlaječku Spojeného království Velké Británie a Severního Irska (Obrázek 4).

Kurzy měn – vykreslení grafu a stažení textových dat

Chcete-li vykreslit graf znázorňující vývoj kurzu některých dvou měn nebo získat tato data v textovém formátu, přejděte na stránku <http://navarin.ms.mff.cuni.cz:7070/charts.php>, např. kliknutím na odkaz *Měny* v hlavním menu webu.

Graf kurzů

Zdrojová měna CZK ▼

Cílová měna USD ▼

Od 08 ▼ Duben ▼ 2009 ▼

Do 06 ▼ Květen ▼ 2009 ▼

Prodej / Nákup ☐ Vykreslit

Zobrazit

Copyright © MoneyMaker Team

Obrázek 5: Stránka pro vykreslování grafů

Pokud potřebujete mít zobrazená data k dispozici v textové podobě (pro další programové zpracování), klikněte na odkaz *Stáhnout textová data*. Nahraje se vám textová stránka

Graf kurzů

Zdrojová měna CZK ▼

Cílová měna EUR ▼

Od 01 ▼ Leden ▼ 2009 ▼

Do 01 ▼ Duben ▼ 2009 ▼

Prodej / Nákup ☐ Vykreslit

Zobrazit



[Stáhnout textová data](#)

Obrázek 6: Zobrazení vývoje koruny vůči euru

obsahující data podobně jako v textu 10. První řádek identifikuje měny, jejichž vztah je zaznamenán, další pak vyjadřují vždy datum a kurz, ve stejném formátu, jako je uvedeno v příkladu, oddělené tabulátorem (YYYY-MM-DD kurz).

Ve formuláři si vyberte měny, které chcete zobrazit. Obrázek bude znázorňovat, kolik jednotek zdrojové měny bylo rovno jedné jednotce cílové měny, v závislosti na čase – viz obrázky 6 a 7. Obrázky se od sebe liší pouze tím, že v druhém je zaškrtnuta volba pro vykreslení pomocí kupní a prodejní ceny.



[Stáhnout textová data](#)

Obrázek 7: Zobrazení vývoje kurzu koruny vůči euru pomocí prodejní a nákupní ceny

Text 10: Kurz koruny vůči euru v textové podobě

CZK/EUR Rate	
2009-03-02	28.30000000
2009-03-03	27.95500000
2009-03-04	27.71000000
2009-03-05	27.74500000
2009-03-06	28.05000000
2009-03-09	27.64000000
2009-03-10	27.21500000
2009-03-11	26.96000000
2009-03-12	27.02000000
2009-03-13	26.59000000
2009-03-16	26.52500000
2009-03-17	26.50000000
2009-03-18	26.99000000
2009-03-19	26.78500000
2009-03-20	26.62500000
2009-03-23	26.82500000
2009-03-24	27.02000000
2009-03-25	27.29500000
2009-03-26	27.23000000

2009-03-27	27.21000000
2009-03-30	27.47000000
2009-03-31	27.38000000
2009-04-01	27.15000000

Prohlížení sportovních výsledků

Pro prohlížení sportovních výsledků klikněte v hlavním menu na odkaz *Sporty*, dostanete se tak na stránku <http://navarin.ms.mff.cuni.cz:7070/sports/>. Kliknutím zvolte, jaký sport vás zajímá a zobrazí se podrobnější menu – menu soutěží (Obrázek 8). Po výběru soutěže se zobrazí tabulka zápasů, které se budou v brzké době hrát a také tabulka všech odehraných zápasů za aktuální rok. Tabulky odpovídají datům v databázi, nemusí být tedy zcela aktuální – databáze je aktualizována pouze jednou denně. V horní části stránky je k dispozici menu s letopočty. Po kliknutí na příslušný rok jsou zobrazeny všechny odehrané zápasy vybrané soutěže (Obrázek 9, strana 45). Pro zobrazení zápasů vybraného týmu lze použít *filtr týmu*.

Sporty	
• HOKEJ • FOTBAL • HAZENÁ • BASKETBAL	
HOKEJ MUŽI	NĚMECKÁ 2. HOKEJOVÁ LIGA
AHL	NĚMECKÁ HOKEJOVÁ LIGA
ČESKÁ 1. HOKEJOVÁ LIGA	NĚMECKÝ HOKEJOVÝ POHÁR
ČESKÁ EXTRALIGA	NHL
ČESKÁ POJIŠTOVNA CUP	NORSKÁ HOKEJOVÁ LIGA
DÁNSKÁ HOKEJOVÁ LIGA	POLSKÁ HOKEJOVÁ LIGA
EURO ICE HOCKEY CHALLENGE	RAKOUSKÁ HOKEJOVÁ LIGA
FINSKÁ HOKEJOVÁ LIGA	RUSKÁ HOKEJOVÁ LIGA
FNSKÁ 2. HOKEJOVÁ LIGA	SLOVENSKÁ HOKEJOVÁ LIGA
KARJALA CUP	ŠVÉDSKÁ 2. HOKEJOVÁ LIGA
KONTINENTÁLNÍ HOKEJOVÁ LIGA	ŠVÉDSKÁ HOKEJOVÁ LIGA
LG HOCKEY GAMES	ŠVÝCARSKÁ 2. HOKEJOVÁ LIGA
LIGA MISTRŮ V HOKEJI	ŠVÝCARSKÁ HOKEJOVÁ LIGA
MS V HOKEJI	

Copyright © MoneyMaker Team

Obrázek 8: Zobrazení seznamu soutěží po kliknutí na odkaz HOKEJ

Sporty: ANGLICKÁ 1. LIGA MUŽI

- [HOKEJ](#)
- [FOTBAL](#)
- [HAZENA](#)
- [BASKETBAL](#)

2008 [2007](#)

Filtr týmu 

Budoucí zápasy

Datum	Týmy
2009-05-24 17:00	SUNDERLAND - CHELSEA
2009-05-24 17:00	MANCHESTER CITY - BOLTON
2009-05-24 17:00	LIVERPOOL - TOTTENHAM
2009-05-24 17:00	HULL - MANCHESTER UTD
2009-05-24 17:00	FULHAM - EVERTON
2009-05-24 17:00	BLACKBURN - WEST BROM
2009-05-24 17:00	ASTON VILLA - NEWCASTLE
2009-05-24 17:00	ARSENAL - STOKE
2009-05-24 17:00	WEST HAM - MIDDLESBROUGH
2009-05-24 17:00	WIGAN - PORTSMOUTH

Odehrané zápasy

Datum	Týmy	Výsledek	1	0	2	10	02
2009-05-18 21:00	PORTSMOUTH - SUNDERLAND	3:1	2.317	3.167	3.055	1.338	1.555
2009-05-17 17:00	CHELSEA - BLACKBURN	2:0	1.269	5.134	11.209	1.017	3.521
2009-05-17 14:30	WEST BROM - LIVERPOOL	0:2	6.670	4.443	1.422	2.667	1.077
2009-05-16 16:00	EVERTON - WEST HAM	3:1	1.725	3.474	4.803	1.153	2.016
2009-05-16 16:00	TOTTENHAM - MANCHESTER CITY	2:1	1.807	3.429	4.295	1.183	1.907

Obrázek 9: Přehled zápasů soutěže (konkrétně 1. anglická liga)

Registrace a přihlašování

Abyste mohli vkládat a spravovat vlastní strategie, musíte být zaregistrovaný uživatel, který má tyto akce povolené. Nejprve se registrujte a pak čekejte, až vám správce udělí povolení – až to udělá, budete automaticky informováni emailem.

Projdeme si celou akci krok po kroku. Na webu klikneme na odkaz *Registrovat*, nebo můžeme otevřít registrační stránku přímo: http://navarin.ms.mff.cuni.cz:7070/register_user.php. Vyplníme požadované údaje – ty, které nejsou označeny hvězdičkou není nutné vyplňovat. Příklad vyplněného formuláře ukazuje obrázek 10. Po odeslání formuláře uvidíte stránku podobnou té, která je zachycena na obrázku 11.

Nyní je třeba čekat, dokud administrátor účet neschválí. Do té doby se na účet nemůžete přihlásit. Až se tak stane, budete informováni emailem (Text 11).

Registrace MoneyMaker uživatele

Uživatelské jméno*	Jirinka
Heslo*	••••••••
Heslo znovu*	••••••••
Jméno*	Jirina
Příjmení*	Kostivalova
Pohlaví*	Žena
Den narození*	06 Duben 1981
E-Mail*	jirina.kostivalova@m
Ulice*	Jirincina
Město*	Krásná ves
PSČ*	12345
Kraj*	Středočeský
Ověření	
Napište tato písmena*	saicrupr
	Odeslat
	* povinný údaj

Copyright © MoneyMaker Team
Obrázek 10: Registrace nového uživatele

Registrace MoneyMaker uživatele

Hotovo, uživatelský účet *Jirinka* byl vytvořen.
Dříve, než se budete moci přihlásit, musí být váš účet aktivován.

Copyright © MoneyMaker Team
Obrázek 11: Výsledek odeslání registračního formuláře

Text 11: Potvrzovací email o povolení účtu

Hello,

Your MoneyMaker account has been activated.
You can now use your account.

Kind Regards,

MoneyMaker team

Nyní už se můžete přihlásit a vytvářet v systému nové strategie. K přihlašování použijte přihlašovací jméno, které jste si zvolili v registračním formuláři (v našem příkladu to je Jirinka) a heslo. **Pozor**, v přihlašovacím jménu i heslu jsou rozlišována malá a velká písmena.

Editace osobních údajů

Pokud si chcete změnit heslo nebo osobní údaje, zvolte, po přihlášení do systému, v menu položku *Profil*. Změňte údaje ve formuláři, jak potřebujete a pomocí tlačítka *Odeslat* je uložte do systému.

Jste přihlášen(a) jako Jirinka.

Jméno*	Jirinka
Příjmení*	Kostivalová
Pohlaví*	Žena
Den narození*	25 Říjen 1990
E-Mail*	Jirinka.Kostival@ma
Ulice*	Jiřínčina
Město*	Krásná ves
PSČ*	12345
Kraj*	Středočeský
Současné heslo	
Nové heslo	
Nové heslo znovu	
	Odeslat

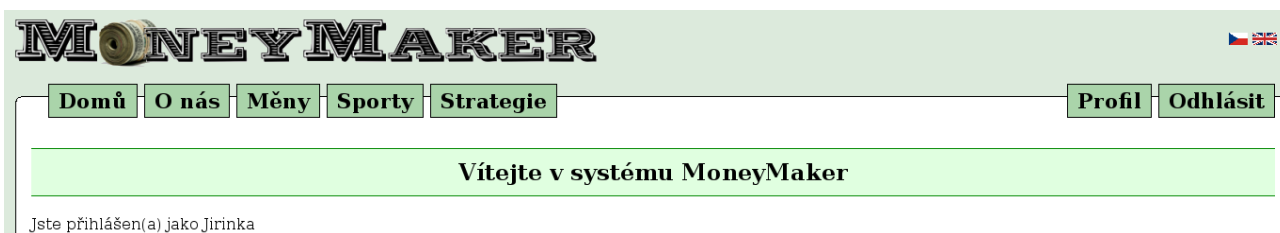
* povinný údaj

Copyright © MoneyMaker Team

Obrázek 12: Změna osobních údajů nebo hesla

Správa strategií

Abyste mohli spravovat strategie, musíte být přihlášení. Po přihlášení se v menu objeví nová položka – *Strategie* (Obrázek 13). Vyberte tuto položku. Zobrazí se stránka obsahující základní údaje o všech vašich strategiích – zpočátku bude prázdná (více v kapitole *Zobrazení strategií* na straně 49).



Obrázek 13: Menu po přihlášení

Vložení strategie do systému

Přidat strategii

Název*	Tajný algoritmus
Plug-in*	/home/sOc/pluginy/Martina_Plu Browse...
Jazyk*	C# ▼
Druh*	Peníze ▼
	☒ Křížit
Maximum instancí	10
Křížených instancí	5
Poznámka	Super tajná strategie, která bude určitě fungovat.
	Odeslat

* povinný údaj

Copyright © MoneyMaker Team

Obrázek 14: Příklad vyplněného formuláře pro vložení strategie

Před tím, než vložíte strategii do systému, ji otestujte pomocí *Demo* aplikace, abyste si byli jisti, že nespadne a že rámcově funguje dobře. Návod naleznete v kapitole *Jak plugin otestovat* na straně 18.

Když jste s funkčností pluginu spokojeni, klikněte na odkaz *Přidat strategii* (nacházíme se na stránce strategií a jsme přihlášení). Vyplňte formulář a odešlete ho. Příklad vyplněného formuláře nabízí obrázek 14. Strategie by měla mít své jméno, které by mělo obsahovat alespoň čtyři znaky a nemělo by přesáhnout délku 64 znaků. Každá strategie musí mít jedinečné jméno (v rámci strategií všech uživatelů).

Dále je nutné zvolit jazyk, ve kterém je plugin napsán a druh odhadů, který vykonává. Pro správné fungování strategie je nutné, abyste obě tyto položky vyplnili správně.

Pokud chcete, aby se proměnné křížily, zaškrtněte příslušné políčko. Pokud pole nezaškrtnete, strategie bude mít vždy pouze jednu instanci. Pokud jste zvolili možnost křížení, má smysl vyplnit i následující dvě pole týkající se počtu instancí a jejich křížení. Vyplněním

prvního pole nastavujete maximální počet instancí vaší strategie, který poběží současně. Druhým polem nastavujete, kolik nejúspěšnějších instancí bude mít možnost stát se rodiči nových instancí. Pokud tato dvě pole nevyplníte, budou vyplněna implicitními hodnotami – v současné době hodnotami 100 a 10. Poznámku můžete nechat prázdnou, nebo si do ní vepsat cokoliv, co potřebujete, ale pouze do délky 255 znaků.

Po odeslání formuláře ale ještě není strategie zcela funkční, je třeba ji aktivovat (viz. kapitola *Aktivace strategie* na straně 49).

Zobrazení strategií

Základní údaje o strategiích se zobrazují na stránce <http://navarin.ms.mff.cuni.cz:7070/strategies/>, na kterou se dostanete po kliknutí na Strategies v menu. Příklad takové stránky ukazuje obrázek 15.

Co řádek, to strategie. Každá strategie má svůj název, datum, kdy jste ji do systému vložili, druh (odhady kurzů měn, nebo výsledků sportovních utkání), jazyk, ve kterém byla napsána, stav a počet instancí ve formátu živé/zabitě. Stavů máme tři:

- CONSTRUCTING – strategie je čerstvě nahraná do systému a ještě nebyla aktivována
- LIVE – strategie počítá odhady a funguje
- ZOMBIE – strategie byla ukončena, již není spouštěna

V dalších sloupcích následuje počet instancí (živých / celkem) a pak přesnosti tipů strategie. Přesnost je udávána ve formě odchylky odhadu oproti skutečné hodnotě, proto čím menší číslo, tím lepší strategie je.

Téměř ve všech zobrazených tabulkách máte možnost třídit řádky podle hodnot ve vybraných sloupcích. Řádky se setřídí po kliknutí na hlavičku tabulky ve sloupci, podle kterého chcete třídit.

Aktivace strategie

Strategie								
• Přidat strategii								
Název	Vloženo	Druh	Jazyk	Stav	Instance	Přesnost Min	Přesnost Avg	Přesnost Max
dve meny	2009-04-23 17:14	MONEY	C#	LIVE	100/150	0.0002	0.0004	0.0008
zrcadleni	2009-04-20 09:53	MONEY	C#	LIVE	100/153	0.0298	0.0402	0.1348
1. anglicka liga fotbal a ms v hokeji	2009-04-25 09:36	SPORT	C#	LIVE	1/1	-	-	-
dva sporty	2009-04-23 17:15	SPORT	C#	LIVE	1/1	-	-	-
moje prumery	2009-04-20 09:53	SPORT	C#	LIVE	1/1	-	-	-
test10	2009-04-26 17:49	MONEY	C#	ZOMBIE	50/62	-	-	-
testicek	2009-04-26 17:45	MONEY	C#	ZOMBIE	80/100	-	-	-
tajná strategie	2009-04-26 17:44	MONEY	C#	ZOMBIE	100/100	-	-	-

Copyright © MoneyMaker Team

Obrázek 15: Zobrazení vložených strategií

Na obrázku 15 vidíme několik strategií, všechny jsou nahrané do systému, první z nich ale ještě nebyla aktivována. Aby začala počítat odhady, je třeba ji aktivovat.

Klikneme na název strategie, kterou chceme aktivovat. Obrázek 17 na straně 51 nám ukazuje, co uvidíme. Červená šipka na obrázku znázorňuje pole, které je třeba zaškrtnout, aby byla strategie aktivována. Po zaškrtnutí pole je nutné formulář odeslat pomocí tlačítka Uložit, které je umístěna dole na stránce.

Nyní už by měl být stav strategie nastaven na hodnotu LIVE a strategie by měla každý den počítat své odhady.

Pozor, ze stavu LIVE se již nelze vrátit do stavu CONSTRUCTING.

Ukončení strategie

Pokud chceme strategii ukončit a dále s ní již nepracovat, postupujeme podobně, jako když jsme ji aktivovali, ale zaškrtneme pole pro deaktivaci strategie a potvrdíme pomocí Uložit (Obrázek 16). Pozor, ze stavu ZOMBIE se již nelze vrátit do jakéhokoliv předchozího stavu.

[Zpět na strategie](#)

Název*	tajná strategie
Druh	MONEY
Jazyk	C#
Vloženo	Apr 26 2009 5:44PM
Stav	LIVE ☒ Aktivováno ☒ Deaktivovat tuto strategii
Poznámka	nejtajnější algoritmus ☒ Křížit
Maximum instancí	100
Křížených instancí	10
interval_len (UNSIGNED INTEGER)	
Křížit	☒ Křížit
Minimum	1
Maximum	100
Výchozí	33
Poznámka	
Upozornění	Strategie je aktivní, měnění parametrů není doporučeno. Uložit

* povinný údaj

Obrázek 16: Umístění pole pro deaktivaci strategie

Editace parametrů strategie

Editace investiční strategie

[Zpět na strategii](#)

Název*

tajná strategie

Druh

MONEY

Jazyk

C#

Vloženo

Apr 26 2009 5:44PM

Stav

CONSTRUCTING

☒ Aktivovat

Poznámka

nejtajnější algoritmus

☒ Křížit

Maximum instancí

100

Křížených instancí

10

interval_len (UNSIGNED INTEGER)

Křížit

☒ Křížit

Minimum

1

Maximum

100

Výchozí

33

Poznámka

Uložit

* povinný údaj

Copyright © MoneyMaker Team

Obrázek 17: Editace strategie (červeně je vyznačeno pole pro aktivaci strategie)

Strategie má své vlastnosti a proměnné, které můžeme měnit – používáme pro to stejný formulář, jako když strategii aktivujeme, nebo deaktivujeme (stav strategie je v podstatě také pouze její vlastnost).

Editační formulář nám tedy také ukazuje obrázek 17. Můžeme měnit jméno strategie, poznámku, maximální počet jejích instancí a počet instancí, ze kterých se křížením zrodí nové instance. Nových instancí se rodí vždy tolik, aby počet instancí strategie dosahoval maximálního počtu instancí. Pokud snížíte číslo maximálního počtu instancí, projeví se to až další den – po spuštění instancí, jejich ukončování a množení.

Následuje seznam všech proměnných, které se kříží. U každé je možné specificky pro ni odškrtnout políčko křížení – proměnná se již dále nebude křížit, pro vznik nových instancí bude použita její výchozí hodnota. V případě, že se proměnná kříží, je možné nastavit, v jakém rozmezí se budou její hodnoty pohybovat – minimální a maximální hodnota.

5 1

Zobrazování výsledků strategie

Zvolte si v menu Strategie, uvidíte přehled strategií (Obrázek 15 na straně 49). Abyste si prohlédli jednotlivé instance dané strategie, klikněte na počet instancí dané strategie (nikoli na jméno strategie). Zobrazí se seznam instancí – Obrázek 18. V tabulce je vidět jednoznačná identifikace instance, která zároveň funguje jako odkaz na podrobnější informace a její výsledky. Dále tabulka ukazuje počet dní, které instance běží, počet vykonaných odhadů, zda je živá, nebo již ukončená (v takovém případě je ve vedlejším sloupci datum a čas ukončení instance) a v posledním sloupci je vidět přesnost typů instance – čím menší číslo, tím vyšší přesnost.

Instance strategie					
Zpět na strategie					
Strategie: dve meny					
Přesnost: min:0.0002, max: 0.0008, průměr: 0.0004					
System ID	Věk	Typy	Živá	Zabito	Přesnost
Instance 987	13 dní	16	Ano		0.0002
Instance 1022	13 dní	16	Ano		0.0003
Instance 978	13 dní	16	Ano		0.0003
Instance 983	13 dní	16	Ano		0.0003
Instance 964	13 dní	16	Ano		0.0003
Instance 1016	13 dní	16	Ano		0.0003
Instance 1000	13 dní	16	Ano		0.0003
Instance 991	13 dní	16	Ano		0.0003
Instance 966	13 dní	16	Ano		0.0003
Instance 1019	13 dní	16	Ano		0.0004
Instance 1030	13 dní	16	Ano		0.0004
Instance 1024	13 dní	16	Ano		0.0005
Instance 1025	13 dní	16	Ano		0.0005
Instance 1026	13 dní	16	Ano		0.0005
Instance 1018	13 dní	16	Ano		0.0005
Instance 1004	13 dní	16	Ano		0.0005
Instance 1008	13 dní	16	Ano		0.0005

Obrázek 18: Začátek seznamu instancí strategie

[Zpět na instance](#)

Strategie: zrcadlení

Instance: 83

Přesnost: 0.0402 (min: 0.070, max: 0.1348)

☐ Ukončit instanci

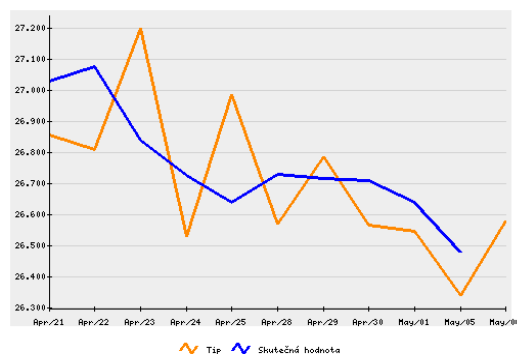
Odeslat

Proměnné

Jméno	Hodnota
interval_len	87

Tipy

CZK -> EUR



Datum	Měna	Tip	Skutečná hodnota	Rozdíl
2009-04-21	CZK->EUR	26.855	27.030	0.175
2009-04-22	CZK->EUR	26.810	27.075	0.265
2009-04-23	CZK->EUR	27.200	26.840	0.360
2009-04-24	CZK->EUR	26.530	26.725	0.195
2009-04-25	CZK->EUR	26.985	26.640	0.345
2009-04-28	CZK->EUR	26.570	26.730	0.160
2009-04-29	CZK->EUR	26.785	26.715	0.070
2009-04-30	CZK->EUR	26.565	26.710	0.145
2009-05-01	CZK->EUR	26.545	26.640	0.095
2009-05-05	CZK->EUR	26.340	26.480	0.140
2009-05-06	CZK->EUR	26.580	neznámá	

Log

Datum	Zpráva
2009-04-21 03:00	run function started
2009-04-21 03:00	done!
2009-04-22 03:00	run function started
2009-04-22 03:00	done!
2009-04-23 03:00	run function started
2009-04-23 03:00	done!
2009-04-24 03:00	run function started
2009-04-24 03:00	done!
2009-04-25 03:00	run function started
2009-04-25 03:00	done!
2009-04-26 03:00	run function started
2009-04-27 03:00	run function started
2009-04-28 03:00	run function started
2009-04-28 03:00	done!
2009-04-29 03:04	run function started
2009-04-29 03:04	done!
2009-04-30 03:00	run function started
2009-04-30 03:00	done!
2009-05-01 03:00	run function started
2009-05-01 03:00	done!
2009-05-02 03:00	run function started
2009-05-03 03:00	run function started
2009-05-04 03:00	run function started
2009-05-05 03:00	run function started
2009-05-05 03:00	done!
2009-05-06 03:00	run function started
2009-05-06 03:00	done!

Copyright © MoneyMaker Team

Obrázek 19: Výsledky instance

Instance odhadující kurzy měn

Příklad zobrazení výsledků pro odhad kurzů měn je obrázek 19. V horní části stránky jsou zobrazeny základní údaje o instanci, jako je jméno strategie, pod kterou patří, její jednoznačná identifikace (číslo instance) a údaje o přesnosti jejích tipů (průměrná, minimální a maximální přesnost). Následuje krátký formulář umožňující instanci okamžitě ukončit. Výsledky jsou zobrazeny pomocí grafu a tabulky, ve spodní části stránky je k dispozici log, který byl vytvořen strategií (pomocí funkce log, viz. kapitola *Logování* na straně 18).

Instance odhadující sportovní kurzy

Příklad zobrazení výsledků ukazuje obrázek 20. Struktura stránky je podobná jako u kurzů měn. V horní části stránky jsou zobrazeny základní údaje o instanci, jako je jméno strategie, pod kterou patří, její jednoznačná identifikace (číslo instance) a údaje o přesnosti jejích typů

Instance strategie

[Zpět na instance](#)
Strategie: moje promery
Instance: 101
Přesnost: 0.0000 (min: 0.0000, max: 0.0000)
☐ Ukončit instanci

Odeslat

Proměnné
Tato instance nemá žádné křížené proměnné.

Tipy

Sport	Soutěž	Domácí tým	Hosté	Příležitost	Tip	Skutečná hodnota	Rozdíl
FOTBAL	ANGLICKÁ 1. LIGA MUZI	WIGAN	MANCHESTER UTD	1	6.735	unknown	
FOTBAL	ANGLICKÁ 1. LIGA MUZI	WIGAN	MANCHESTER UTD	10	2.422	unknown	
FOTBAL	ANGLICKÁ 1. LIGA MUZI	WIGAN	MANCHESTER UTD	0	3.935	unknown	
FOTBAL	ANGLICKÁ 1. LIGA MUZI	WIGAN	MANCHESTER UTD	02	1.167	unknown	
FOTBAL	ANGLICKÁ 1. LIGA MUZI	WIGAN	MANCHESTER UTD	2	1.807	unknown	
FOTBAL	ANGLICKÁ 1. LIGA MUZI	LIVERPOOL	ARSENAL	1	3.379	1.695	29.40%
FOTBAL	ANGLICKÁ 1. LIGA MUZI	LIVERPOOL	ARSENAL	10	1.578	1.144	24.04%
FOTBAL	ANGLICKÁ 1. LIGA MUZI	LIVERPOOL	ARSENAL	0	3.749	3.521	1.73%
FOTBAL	ANGLICKÁ 1. LIGA MUZI	LIVERPOOL	ARSENAL	02	1.880	2.079	-5.09%
FOTBAL	ANGLICKÁ 1. LIGA MUZI	LIVERPOOL	ARSENAL	2	4.662	5.078	-1.76%
FOTBAL	ANGLICKÁ 1. LIGA MUZI	CHELSEA	EVERTON	1	2.412	1.419	29.01%
FOTBAL	ANGLICKÁ 1. LIGA MUZI	CHELSEA	EVERTON	10	1.349	1.052	20.93%
FOTBAL	ANGLICKÁ 1. LIGA MUZI	CHELSEA	EVERTON	0	3.632	4.061	-2.91%
FOTBAL	ANGLICKÁ 1. LIGA MUZI	CHELSEA	EVERTON	02	1.949	2.729	-1.168%

Obrázek 20: Výsledky instance

(průměrná, minimální a maximální přesnost). Následuje krátký formulář umožňující instanci okamžitě ukončit. Výsledky jsou zobrazeny pomocí tabulky. Pokud se již zápas odehrál, jsou zobrazeny i skutečné hodnoty vypsanych kurzů a také rozdíl mezi nimi a vašimi tipy.

Informace o jednotlivých proměnných

Na obrázcích 19 i 20 jste již jistě postřehli, že na začátku reportu je odstavec věnovaný kříženým proměnným instance. Pokud kliknete na jméno proměnné, dozvíte se o ní podrobnější informace (ukázka na obrázku 21).

[Zpět na instanci 43](#)

Proměnná interval_len

Typ: UNSIGNED INTEGER

Křížit: Ano

Rozsah: 2-1000 (default: 20)

Současná hodnota: 321

Strategie: zrcadlení

Historie proměnné

Datum	Hodnota
2009-04-20 19:00	321

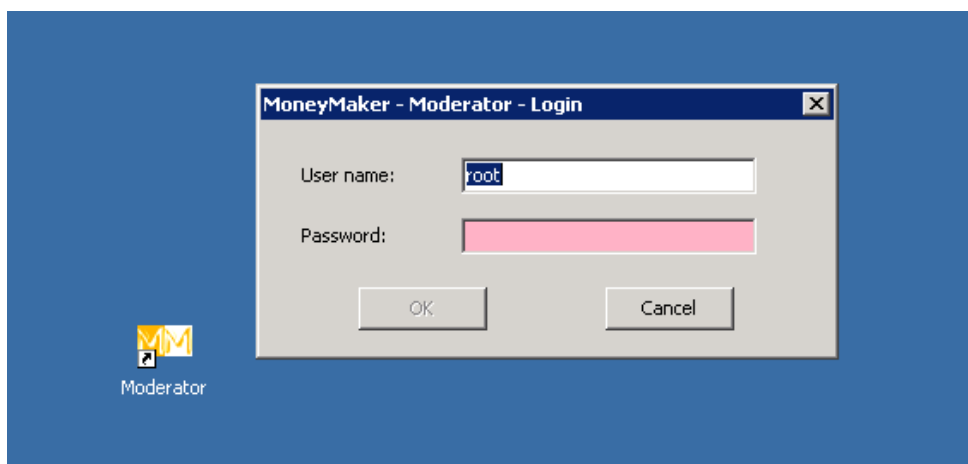
*Obrázek 21: Vlastnosti
jedné konkrétní proměnné*

Ruční ukončení instance

Pokud chcete některou z instancí ručně ukončit, jděte na stránku, kde si prohlédnete její výsledky. V horní části stránky naleznete okénko Ukončit instanci (viz. obrázek 19 na straně 53). Zatrhnete jej a kliknete na tlačítko Odeslat.

Používání Moderátora

Moderátor je takzvaný tlustý klient. Je to program, který běží na serveru a je určen hlavně administrátorovi a privilegovaným uživatelům, pro správu systému a uživatelů, strategií a dalšího. Mohou se do něho samozřejmě přihlásit i běžní uživatelé a provádět zde některé



Obrázek 22: Ikonka moderátora a přihlašovací okno

úkony – ale vzhledem k tomu, že *Moderátor* běží pouze na serveru, většina z nich k němu nemá přístup.

Moderátor by měl mít na ploše svého zástupce se žlutou ikonkou (Obrázek 22). Spusťte ho a přihlaste se pod svým uživatelským jménem a heslem (nebo pod administrátorem, který má vždy uživatelské jméno `root`, případně další zvolená jména). Pole pro zadávání hesla je růžové, pokud je zatím zadané uživatelské heslo příliš krátké na to, aby mohlo být platné.

Ovládání Moderátora pro běžného uživatele

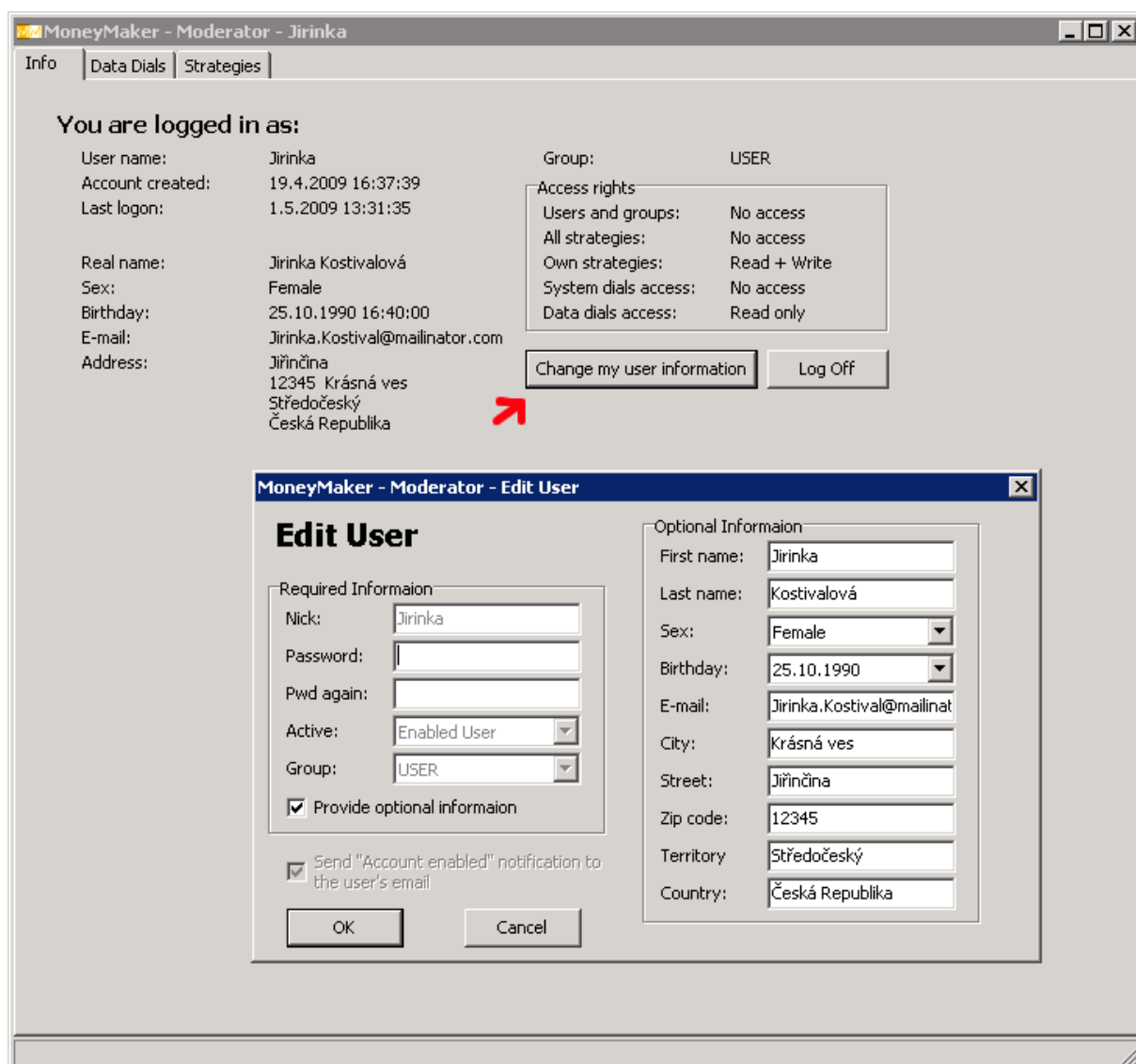
Osobní údaje

Po přihlášení se otevře záložka obsahující osobní informace přihlášeného uživatele.

Tyto informace lze změnit po kliknutí na příslušné tlačítko, vyznačené červenou šipkou na obrázku 23 (strana 57). V levé části formuláře si může uživatel pouze změnit heslo, případně zaškrtnout volbu, že chce zadat další informace, pokud je již nemá zadané. Při zadávání hesla do prvního pole se pozadí pole změní na růžové, dokud heslo nemá požadovanou délku (alespoň osm znaků). Při kontrolním zadávání hesla do druhého pole je pozadí růžové, dokud se znovu zadávané heslo neshoduje s předchozím – tento mechanismus slouží jako kontrola proti překlepům a jiným nepříjemnostem, které částečně brání uživateli omylem zadat jiné heslo, než je přesvědčen, že zadává. Pokud kolonky s hesly nezměníte (necháte prázdné), heslo se nezmění.

V pravé části okna je prostor pro informace o zařazení uživatele do skupiny a jeho právech. Jsou zde rozepsána práva, se kterými uživatel přistupuje k jednotlivým skupinám dat – která smí, nebo nesmí, prohlížet a do kterých může i zapisovat.

Pod Access Rights rámečkem se také nalézá tlačítko Log Off, kterým se můžete odhlásit a následně přihlásit jako jiný uživatel.



Obrázek 23: Změna osobních údajů a hesla

Uložená data

Další záložkou je záložka nazvaná Data Dials. Tato záložka slouží pro zobrazování některých záznamů a proměnných z databáze. V současné době se tento typ záznamů v projektu nevyužívá, proto je záložka, co se týče zobrazovaných dat, prázdná.

Strategie

Pro prohlížení nahraných strategií a jejich nastavení slouží formulář v záložce Strategies. Každá strategie má svého majitele, svůj název, datum, kdy jste ji do systému vložili, druh (odhady kurzů měn, nebo výsledků sportovních utkání), jazyk, ve kterém byla napsána, stav a počet instancí. Stavů máme tři:

- CONSTRUCTING – strategie je čerstvě nahraná do systému a ještě nebyla aktivována
- LIVE – strategie počítá odhady a funguje
- ZOMBIE – strategie byla ukončena, již není spouštěna

Ze stavu `CONSTRUCTING` lze přejít pouze do stavu `LIVE` a z něho pouze do stavu `ZOMBIE`. Tedy změny stavu instance jsou nevratné.

Filtr

Pro zobrazování a vyhledávání strategií slouží filtr. Na obrázku 24 (strana 59) je zobrazen filtr částečně vyplněný. Uživatel vyžadoval zobrazit strategie, které nemají ve jménu slovo „*Hloupá*“, nezáleží na druhu (pro sporty nebo měny), je napsána v *C#*, v libovolném stavu (neaktivovaná, živá, zombie), nezávisle na tom, zda se její proměnné kříží, nezávisle na počtu jejích instancí i počtu křížených instancí, vložena od začátku dubna do 1. května. Po vyplnění filtru je třeba stisknout tlačítko `Execute filter` umístěné na pravé straně okna. V dolní části okna se pak zobrazí odpovídající výsledky. Výsledků se zobrazí vždy maximálně 100 – předpokládá se, že pokud bylo nalezeno více, než 100 záznamů, byl dotaz špatně specifikován a je třeba ho vyplnit více konkrétními daty.

Popišme si nyní filtr podrobněji. Skládá se z několika řádků, co řádek, to možnost si zvolit jednu vlastnost. V levé části řádku je popisek, jaké vlastnosti strategie se řádek týká (např. jméno strategie), následuje výběr samotné vlastnosti (např. obsahuje, neobsahuje, je rovno, začíná, končí, bez omezení) a poslední je třeba specifikovat konkrétní parametr (např. část jména hledané strategie).

MoneyMaker - Moderator - Jirinka

Info

Data Diagnostics

Strategies

Main

Variables - zrcadleni

Instances - zrcadleni

Filter

User name

without limitation

does not contain

Kind

is

Language

is

State

is

Note

without limitation

Genetic mixed

is

Max instances

without limitation

Mixed instances

without limitation

Insert time

between

and

1. 4. 2009

1. 5. 2009

New/Edit Strategy

User name:

Jirinka

Title:

zrcadleni

Kind:

MONEY

Language:

C#

State:

LIVE

Note:

Add strategy

Zombify strategy

Edit strategy

Edit variables

Edit instances

Reset filter

Execute filter

Id	User name	Title	Insert time	Kind	Language	State	Note	Genetic mixed	Max instances	Mixed instances	Inaccuracy - min	Inaccuracy - max	Inaccuracy - avg
19	Jirinka	1. anglicka liga fotbal a ms v nokeji	25.4.2009 9:36:03	SPORT	C#	LIVE		N	100	10			
16	Jirinka	dva sporty	23.4.2009 17:15:45	SPORT	C#	LIVE		N	100	10			
15	Jirinka	dve meny	23.4.2009 17:14:52	MONEY	C#	LIVE		Y	100	10	0	0.001	0.001
2	Jirinka	moje prumery	20.4.2009 9:53:53	SPORT	C#	LIVE		N	100	10			
29	Jirinka	tajná strategie	26.4.2009 17:44:45	MONEY	C#	ZOMBIE	neřekneš algoritmus	Y	100	10			
21	Jirinka	test	26.4.2009 16:43:00	MONEY	C#	ZOMBIE	pokus	Y	100	10			
22	Jirinka	test1	26.4.2009 16:52:08	MONEY	C#	ZOMBIE	test přidání z moderatora	Y	100	10			
31	Jirinka	test10	26.4.2009 17:49:41	MONEY	C#	ZOMBIE	test10	Y	50	20			
25	Jirinka	test2	26.4.2009 16:53:50	MONEY	C#	ZOMBIE	pokus	Y	100	10			
26	Jirinka	test3	26.4.2009 16:54:14	MONEY	C#	ZOMBIE	test z moderatora	Y	100	10			
27	Jirinka	test4	26.4.2009 16:55:41	MONEY	C#	ZOMBIE	dgsd	Y	100	10			
28	Jirinka	test5	26.4.2009 16:57:09	MONEY	C#	ZOMBIE	test	Y	100	10			
30	Jirinka	test6cek	26.4.2009 17:45:38	MONEY	C#	ZOMBIE	zmenen nazev	Y	80	20			
1	Jirinka	zrcadleni	20.4.2009 9:53:03	MONEY	C#	LIVE		Y	100	10	0.018	0.135	0.046

Obrázek 24: Nastavení strategií

Názvy jednotlivých vlastností strategie jsou zřejmé, jedná se postupně o jméno strategie, její název, typ, programovací jazyk, stav, zda má povoleno křížení proměnných, maximální povolený počet instancí, počet nejlepších instancí, které se budou křížit v nové instanci a datum vložení strategie do systému.

Jak je vidět z obrázku 23, uživatel Jirinka nemá žádná práva na strategie ostatních uživatelů. Proto se mu vždy, nezávisle na nastavení filtru, zobrazí pouze jeho (její) strategie.

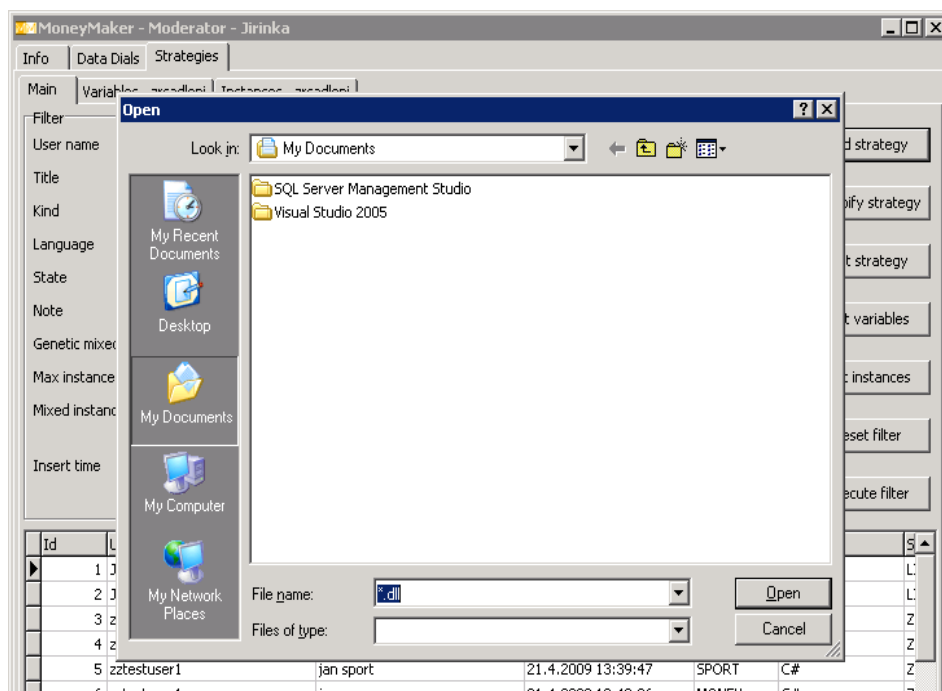
Pozor, **filtr je citlivý na velikost písmen**.

Tvorba nové strategie

K těmto účelům slouží pravá část záložky Strategies. Je třeba vyplnit příslušný formulář, všechny položky jsou povinné. Rovněž je třeba vyplnit všechny správně, jinak nemusí strategie fungovat tak, jak byste si představovali, dokonce může „spadnout“ – zvláště, pokud je špatně vyplněný typ a jazyk strategie.

Pokud se Vám nechce vyplňovat formulář celý znovu, můžete nejdříve vybrat myši ze seznamu strategií takovou strategii, která je vlastnostmi nejvíce podobná té, co chcete vložit. Po tomto kroku se totiž některé vlastnosti vybrané strategie předvyplní do formuláře pro přidání nové strategie.

Když je formulář vyplněn, klikněte na tlačítko Add strategy. Zobrazí se dialog pro vybrání dll souboru strategie (Obrázek 25). Vyberte soubor, který chcete vložit a klikněte na Open.



Obrázek 25: Přidání dll souboru strategie při vkládání nové strategie do systému

Strategie se vloží do systému. Pokud jste zvolili při vkládání stav CONSTRUCTING, je nyní potřeba strategii editovat do stavu LIVE, aby začala v systému fungovat.

Editace strategie

Pokud chcete editovat základní vlastnosti strategie, nejprve ji najděte v seznamu strategií (umístěný ve spodní části okna). Pokud je moc dlouhý, použijte filtr. Klikněte na řádek obsahující strategii, kterou chcete editovat – do pravé horní části okna (rámeček pro editaci) se načtou aktuální vlastnosti strategie. Proveďte požadované změny a klikněte na tlačítko `Edit strategy`.

Editace křížených proměnných strategie

Pokud má strategie proměnné, které se kříží, můžete jim nastavovat některé vlastnosti. V záložce `Strategies` vyberte strategii, jejíž proměnné chcete editovat a klikněte na tlačítko `Edit variables`. Dostanete se tak do další záložky obsahující křížené proměnné vybrané strategie.

Tato záložka (`Variables - <jméno strategie>`) funguje na podobném principu, jako záložka `Strategies`. Vlevo je filtr pro vyhledávání proměnných, který funguje analogicky jako filtr v záložce `Strategies`, vpravo rámeček pro editaci a ve spodní části se zobrazují proměnné, které prošly filtrem.

Najděte si tedy proměnnou, kterou chcete měnit, klikněte na ni, proveďte změny a klikněte na tlačítko `Edit variable`, tím se provedené změny uloží.

V záložce pro editaci proměnných jsou k dispozici ještě dvě další tlačítka – pro vytvoření a pro smazání proměnné. V současné fázi projektu doporučujeme tato tlačítka nepoužívat, protože jejich pomocí určitě neuděláte to, co byste chtěli. Pokud proměnnou smažete, instance sice „nespadne“, ale bude místo hodnoty smazané proměnné používat defaultní hodnotu. Pokud si vytvoříte proměnnou navíc, nic se nestane (proměnná se sice vytvoří, ale nikdo ji nebude používat).

Editace instancí strategie

Také můžete ručně pracovat s jednotlivými instancemi strategie. V záložce `Strategies` vyberte strategii, jejíž instance chcete editovat a klikněte na tlačítko `Edit instances`. Dostanete se tak do další záložky obsahující instance vybrané strategie.

Opět zde funguje filtr pro snazší orientaci v hromadě instancí. Operace, které můžete s instancemi provádět znázorňují tlačítka vpravo:

- kliknutím na `Born instance` se zrodí nová instance
- vybranou instanci můžete ukončit kliknutím na `Kill instance`
- můžete si prohlížet proměnné vybrané instance, pomocí `View variables` tlačítka (u každé proměnné máte potom ještě možnost podívat se na historii jejích hodnot, jak byly měněny)
- a můžete si přečíst log, který instance za svůj život vytvořila, pomocí tlačítka `View log`

Další záložky

Tímto jsme vyčerpali všechny záložky, které běžný uživatel standardně vidí – k jiným datům nemá přístup. Pokud je ovšem nastavení systému benevolentnější, nebo patříte do nějaké speciální uživatelské skupiny, je možné, že vidíte i další záložky – `Users`, `Groups` nebo `System Dials`. Pravděpodobně k nim máte práva pouze na čtení, ale možná i pro editaci (Vaše práva jsou rozepsána v záložce `Info`). Pro více informací o těchto záložkách pokračujte

ve čtení v následující kapitole *Ovládání Moderátora pro administrátora nebo privilegovaného uživatele*.

Ovládání Moderátora pro administrátora nebo privilegovaného uživatele

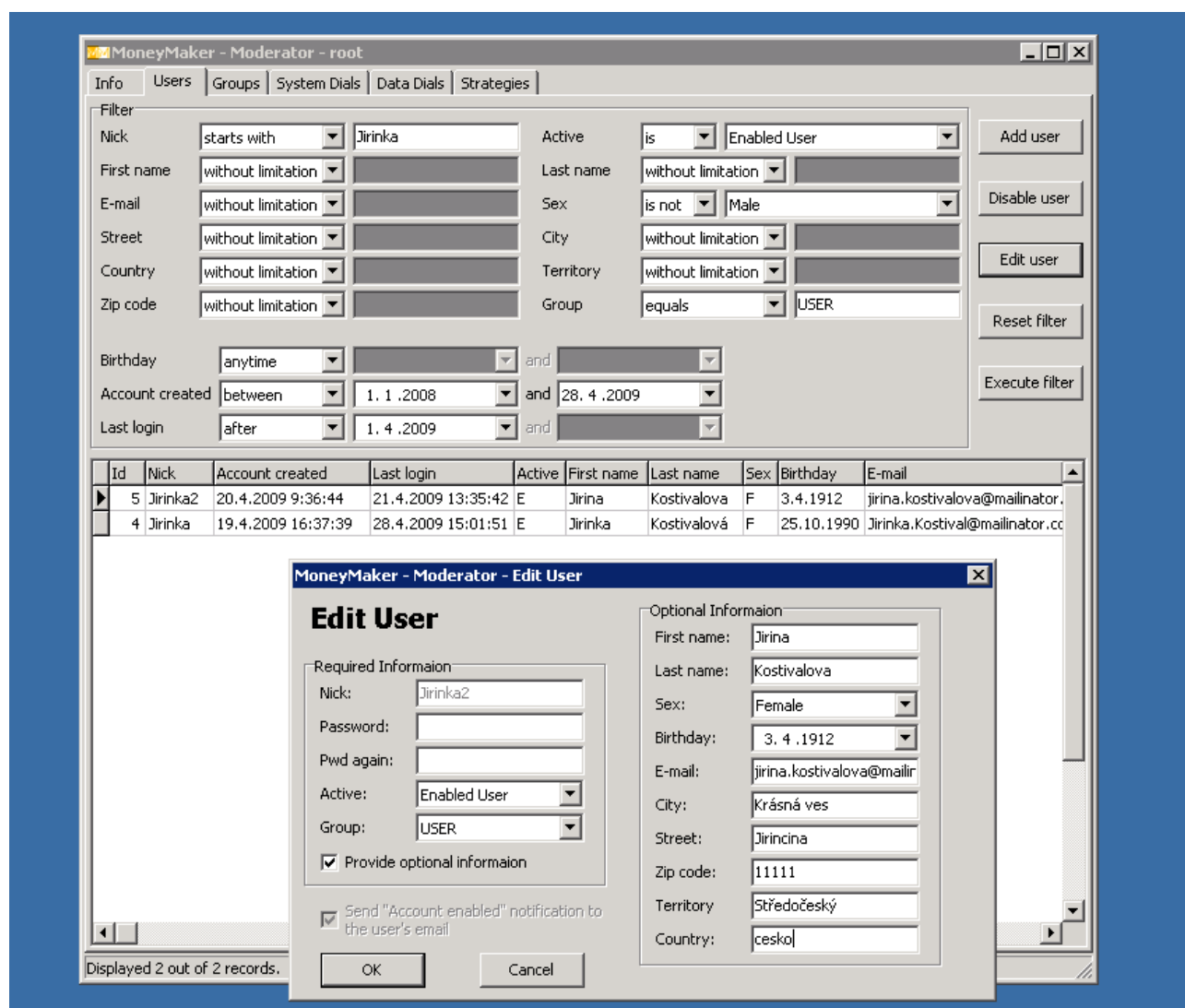
Než se pustíte do čtení následujícího textu, přečte si nejdříve návod pro obyčejného uživatele (začíná na straně 55).

Pro přihlášení jako administrátor slouží obvykle uživatelské jméno `root`, ale pokud byste měli zájem, můžete administrátorských účtů vytvořit více s různými libovolnými přihlašovacími jmény, o tom více později (v sekci o správě uživatelů).

Osobní údaje

Tato sekce se pro administrátora nijak neliší od sekce pro běžného uživatele.

Správa uživatelů



Obrázek 26: Editace uživatele

Uživatele může administrátor spravovat v záložce Users. Správou uživatelů se rozumí editace jejich osobních dat i systémových údajů (skupiny, stav, hesla), jejich přidávání a odstraňování.

Pro přidání nového uživatele klikněte na tlačítko **Add user**. Vyplňte povinné údaje (uživatelské jméno, heslo), nastavte stav, skupinu, do které má patřit a případně můžete vyplnit i doplňující osobní informace. Pokud volíte pro uživatele rovnou stav **Enabled User**, máte možnost dole odškrtnout volbu pro informování uživatele emailem, že byl jeho účet povolen. Klikněte na tlačítko **OK** a uživatel bude vytvořen.

Pokud chcete některému z uživatelů zakázat přihlášení do systému, najděte ho v seznamu. Pro vyhledávání uživatelů lze opět využít filtr*. Vyberte řádek s tímto uživatelem a klikněte na tlačítko **Disable**. U uživatele by se měl změnit stav na **Disabled User** (v tabulce uvidíte písmeno **D** na místo **E**).

* Nezapomeňte, že filtr je citlivý na velikost písmen.

Je-li třeba uživateli pouze pozměnit nebo doplnit některé informace, vyberte řádek s uživatelem z tabulky a klikněte na tlačítko `Edit user`. Zobrazí se podobný dialog, jako když editujete informace o sobě (Obrázek 26, stránka 63). Proveďte potřebné změny. Opět platí, že pokud necháte kolonky na hesla prázdné, bude zachováno původní heslo. Klikněte na `OK` a informace o uživateli budou aktualizované.

Příklad - aktivace uživatele

Administrátor má na starosti aktivaci nových uživatelů. Uživatelský účet není aktivován automaticky, když se uživatel zaregistruje přes webové rozhraní, ale musí být aktivován ručně administrátorem. Důvodem je údržba systému v dobrém stavu, bez neověřených uživatelů, kteří by mohli eventuálně páchat nepříjemnosti ostatním uživatelům. Uživatelé sice nemají práva sahát na strategie ostatních uživatelů, nicméně systém není v tomto ohledu ještě dokonalý a tuto vlastnost lze obejít (přenastavením práv v kódu strategie). Ale i bez této vlastnosti je vhodné uživatele osobně prověřovat, škodliví uživatelé by mohli například zahltit systém velkým množstvím strategií.

Proces aktivace uživatele zahajte tím, že ho pomocí filtru najdete (např. si necháte vypsát všechny nové uživatele). Označte řádek s uživatelem myší a klikněte na tlačítko `Edit`. V řádku `Active` vyberte možnost `Enabled user`. Pokud nechcete, aby byl uživatel informován o aktivaci jeho účtu, zrušte zaškrtnutí položky `Send „Account enabled“ notification to the user's email`. Klikněte na `OK`, tím je uživatelské konto aktivováno.

Skupiny

Práva jednotlivých uživatelů se liší tím, v jaké jsou skupině (skupinu pro uživatele lze nastavit v editaci uživatele). Jaká práva má která skupina, vytváření skupin a odebrání skupin se provádí v záložce `Groups`. Záložka opět obsahuje filtr pro snazší vyhledávání ve skupinách.

Pro přidání skupiny vyplňte formulář v pravé části okna – zadejte požadované jméno* skupiny, práva a případně poznámku. Klikněte na tlačítko `Add group`.

Pokud si přejete skupinu odstranit, vyberte ji v dolní tabulce a klikněte na tlačítko `Delete group`. Ovšem pozor, aby bylo možné skupinu odstranit, nesmí být v této skupině žádní uživatelé (pokud tomu tak nebude, vypíše se chybové hlášení).

Poslední z možností, co lze se skupinami dělat, je jejich editace. Použijte filtr pro nalezení skupiny, označte ji v tabulce. Do formuláře v pravé části okna se načtou informace o skupině, které můžete upravit. Nakonec klikněte na tlačítko `Edit group`, tím se upravené informace uloží do databáze.

Systémové proměnné

V záložce `System dials` lze nastavovat hodnoty proměnných, které přímo ovlivňují chování celého systému.

V levé části okna lze opět nalézt filtr, v pravé formulář pro editaci existujících proměnných a přidávání nových proměnných. Ve spodní části okna se zobrazují proměnné odpovídající

* Skupina musí mít originální jméno, nelze mít více skupin se stejným názvem. Totéž platí i pro tvorbu uživatelů, strategií atd.

aktuálnímu nastavení filtru se všemi svými vlastnostmi. Popsaný stav ilustruje obrázek 27.

S formuláři se pracuje analogicky, jako ve všech předchozích případech. Pro přidání nové proměnné nejdříve vyplňte formulář v pravé části okna a pak klikněte na tlačítko Add value. Při vyplňování je kontrolováno, zda zadaná hodnota proměnné odpovídá zadanému typu proměnné. Pokud tomu tak není, je pozadí vyplňované hodnoty proměnné růžové, je třeba hodnotu nebo typ opravit. Typy proměnných jsou brány dle všeobecných zvyklostí*.

V současné době umí systém pracovat pouze s omezenou množinou proměnných – pokud tedy přidáte proměnnou novou, která do této množiny nepatří, bude sice vytvořena a její hodnota uložena, ale nebude mít na chod systému žádný vliv. Tabulku využitelných

Id	Name	Type	Value	Note
1	MAXLOGINTIME	UINT	10	Logout time for inactive user (in minutes)
2	MAXMONEYIDLE	UINT	6	Rate of money older this value dies (in months)
3	MONEYRATEBUY	REAL	0.005	Increate/degrease price of money to buy (in percent)
4	MONEYRATESELL	REAL	-0.005	Increate/degrease price of money to sell (in percent)
5	PLUGINSPATH	STRING	c:\Plugins\	Directory with all strategies
6	BUSINESSMANCOMMON	STRING	BUSINESSMANCOMMON.DLL	Inserted plugins can not have same name like BUSINESSMANCOM
7	THREADSCOUNT	UINT	8	Size of thread pool in Businessman
8	THREADTIMEOUT	UINT	60000	Maximum timeout of each thread (in miliseconds)
9	ACCURACY	UINT	3	Accuracy of money and sport rates (PLEASE DO NOT MODIFY, IT
10	GENESISFULLNAME	STRING	c:\Genesis\Genesis.exe	Full name of genesis application
11	INSTANCEPROTECTPERIOD	UINT	5	Amount of days in which an invest strategy instance cannot get k
12	MAXINSTANCES	UINT	100	Maximum number of live instances each strategy can have in one
13	BESTINSTANCESMIXED	UINT	10	Count of the best instances will be mixed (MAXINSTANCES <= MA
14	ACCURACYMONITORPERIOD	UINT	30	Accuracy of a strategy instance is monitored up to this amount of
15	MONEYLIVETIMEOUT	UINT	2	Time to declare that inactive money is death (in months)
16	INSTANCEKILLRATIO	REAL	0.25	Ratio of instances to be killed on each genetic phase
17	MUTATIONPROBABILITY	REAL	0.75	Ratio of variables that will get random (min,max) value on initializ
18	MIXINGPROBABILITY	REAL	0.9	Ratio of instance variables that will get value from their parents

Obrázek 27: Prohlížení a editace systémových proměnných

proměnných ukazuje tabulka označená jako Text 12.

* Používá se desetinná tečka, nikoli čárka. Typ BOOLEAN má povolené hodnoty TRUE a FALSE.

Pro odebrání proměnné ji vyberte myší v tabulce a klikněte na tlačítko Delete value. Tuto operaci si ale **dvakrát rozmyslete**, protože to může vážně narušit fungování systému! Ve většině případů systém zcela přestane fungovat. Pokud se Vám to stane, doporučujeme použít Text 12 pro rekonstrukci smazaných proměnných.

A konečně editace proměnné: načtete proměnnou pomocí filtru, označte ji v tabulce. Její hodnoty se načtou do editačního formuláře v pravé části okna, upravte je dle potřeby a klikněte na tlačítko Edit value pro uložení změn. Platí zde stejné varování, jako v předchozím případě: „dvakrát měřte, jednou řežte“.

Jména proměnných nemusí vždy přesně vyjadřovat jejich význam. Aby uživatel pochopil, k čemu přesně proměnná slouží, obsahuje každá proměnná také poznámku. Poznámka by měla podrobně vysvětlovat, jaký má hodnota proměnné význam. Před tím, než proměnnou změníte, přečtěte si poznámku, abyste se ujistili, že se stane to, co chcete. Rovněž, když zakládáte novou proměnnou, nezapomeňte vyplnit poznámku příslušným komentářem.

Text 12: Tabulka používaných proměnných, jejich typů a významu (lze použít, pokud si nedopatřením některou z proměnných smažete či upravíte jinak, než jste zamýšleli)

Proměnná	Typ	Popis	Příklad hodnoty
MAXLOGINTIME	UINT	Po kolika minutách nečinnosti je uživatel automaticky odhlášen	10
MAXMONEYIDLE	UINT	Po kolika měsících ne-aktualizace bude měna prohlášena za zaniklou.	6
MONEYRATEBUY	REAL	Kolik procent je marže při nákupu.	0.005
MONEYRATESELL	REAL	Kolik procent je změna ceny při prodeji.	-0.005
PLUGINSPATH	STRING	Adresář pro ukládání nahraných strategií	C:\Plugins
BUSINESSMANCOMMON	STRING	Jméno hlavní knihovny Businessmanu. (na velikosti písmen nezáleží)	BUSINESSMANCOMMON.DLL
THREADSCOUNT	UINT	Velikost <i>threadpoolu</i> Businessmana	8
THREADETIMEOUT	UINT	Maximální doba běhu jednoho pluginu v milisekundách	60000
ACCURACY	UINT	Přesnost uložení tipů kurzů měn a sportovních kurzů. Nikdy neměňte!	3
GENESISFULLNAME	STRING	Celá cesta k aplikaci Genesis	C:\Genesis\Genesis.exe
INSTANCEPROTECTPERIOD	UINT	Počet dní od prvního spuštění pluginu, kdy nejsou instance zabíjeny a kříženy	5
MAXINSTANCES	UINT	Maximální počet instancí strategie, které mohou být současně živé	100
BESTINSTANCESMIXED	UINT	Počet nejlepších instancí, které jsou kříženy mezi sebou při vzniku nových instancí	10
ACCURACYMONITORPERIOD	UINT	Úspěšnost strategie je vypočítávána	30

Proměnná	Typ	Popis	Příklad hodnoty
		na základě výsledků za posledních ACCURACYMONITORPERIOD dní.	
INSTANCEKILLRATIO	REAL	Kolik procent instancí se bude pravidelně ukončovat, aby uvolnily prostor nově vzniklým. (hodnota v intervalu (0,1))	0.25
MUTATIONPROBABILITY	REAL	Procento proměnných, které získá svoji hodnotu náhodným výběrem z intervalu zadaného jako interval možných hodnot pro danou proměnnou (hodnota v intervalu (0,1))	0.75
MIXINGPROBABILITY	REAL	Procento proměnných, které při křížení získají hodnotu od svého rodiče (hodnota v intervalu (0,1))	0.9
SNUBSBEFOREKILL	INT	Maximální počet napomenutí instance (kolikrát vytvořila chybovou výjimku). Po překročení tohoto čísla je instance ukončena.	5

Pro pokročilé – více tipů v jedné strategii

Jeden plugin může ukládat do systému více odhadů. Odhady ale musejí být stejného typu. Tedy nelze vytvořit plugin, který by zároveň odhadoval vývoj kurzů mezi měnami a vývoj sportovních kurzů, ale je možné udělat plugin, který bude odhadovat kurzy mezi více dvojicemi měn, nebo pro více soutěží, sportů, týmů atd.

Pro sportovní odhady to vlastně není žádná novinka, tato vlastnost je zřejmá již z ukázkových strategií. Z uvedených strategií pro odhadování kurzů měn ale tato možnost nevyplývá. Proto se budeme nadále v této kapitole zabývat pouze prací se strategiemi pro odhadování kurzů měn.

Implementace

Implementace více odhadů v jednom pluginu je intuitivní. Jednoduše uložíte více tipů. Běžně ukládáte odhad do proměnné `mrLine.NextBet`. Text 13 ukazuje, jakým způsobem instanci `mrLine` získáte – je zřejmé, že instance `mrLine` jednoznačně určuje obě měny tvořící vzájemný kurz a tedy uložení hodnoty do proměnné `mrLine.NextBet` jasně určuje, který kurz je odhadován.

Pokud chcete uložit odhady kurzů mezi více dvojicemi měn, musíte získat všechny instance `mrLine` příslušející dotčeným dvojicím a do nich odhady uložit.

Text 13: zjednodušený postup uložení odhadu kurzu mezi českou korunou a eurem

```
mPack = (MoneyPack)pack;  
mPack.Money.TryGetValue("CZK", out mData);  
mData.RateLines.TryGetValue("EUR", out mrLine);  
mrLine.NextBet = my_bet;
```

Plugin nahraný v systému

Další postup zůstává v podstatě stejný. `BusinessDemoMoney` umí s více odhady pracovat stejně, jako s jediným odhadem. Nahrání pluginu do systému také probíhá stejným způsobem, stejně jako jeho správa a editace. Při prohlížení výsledků bude zobrazen vlastní graf pro každý odhad.