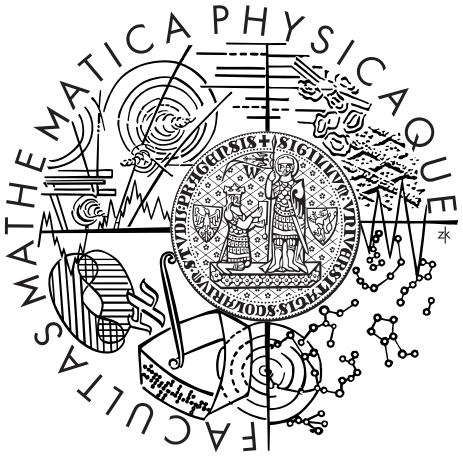
Univerzita Karlova v Praze

Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Tomáš Krejčí

Genetické programování pro predikci finančních trhů

Katedra softwarového inženýrství

Vedoucí diplomové práce: RNDr. David Bednárek, Ph.D.

Studijní program: Informatika

Studijní obor: Obecná informatika

Praha 2015

Tímto bych chtěl poděkovat RNDr. Davidovi Bednárkovi, Ph.D., vedoucímu mé bakalářské práce, za hodnotné připomínky při vypracování této práce.

Prohlašuji, že jsem tuto diplomovou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V Praze dne 21. května 2015 Podpis autora

Název práce: Genetické programování pro predikci finančních trhů

Autor: Tomáš Krejčí

Katedra: Katedra softwarového inženýrství

Vedoucí diplomové práce: RNDr. David Bednárek, Ph.D., Katedra softwarového inženýrství

Abstrakt: Cílem práce je otestovat vhodnost užití genetického programování pro predikci finančních trhů v závislosti na jejich předchozím vývoji. Obsahem práce je studium metod genetického programování použitých či použitelných v oblasti predikce trhů. Praktickou částí je implementace vybraných metod genetického programování a testování jejich úspěšnosti na základě dostupných historických dat z finančních trhů.

Klíčová slova: optimalizace, genetické algoritmy, evoluční algoritmy, finanční trhy

Title: Genetic programming in financial markets forecasting

Author: Tomáš Krejčí

Department: Department of Software Engineering

Supervisor: RNDr. David Bednárek, Ph.D., Department of Software Engineering

Abstract: The aim of this thesis is to test usability of the genetic programming for predicting of the financial markets based on historical prices. The thesis includes the study of genetic programming techniques used or useful for the market prediction. The practical part of thesis is implementation of selected methods and testing their performance on available historical data from financial markets.

Keywords: optimization, genetic algorithms, evolutionary algorithms, financial markets

Obsah

1. Úvod

Finanční trhy jsou podstatnou složkou celosvětové ekonomiky a dění na nich ovlivňuje mnoho oblastí. Například na jedné z největších světových burz The New York Stock Exchange (NYSE) se každý den zobchodují až stovky miliard amerických dolarů [12].

Motivací pro většinu subjektů participujících na finančních trzích je zisk. Aby však byli schopni svůj zisk realizovat, musejí mít k dispozici predikci budoucího vývoje ceny aktiva, se kterým obchodují.

Genetické programování je technikou pro automatické (black-box [21]) řešení problémů. Za více než dvacet let své existence bylo použito v mnoha oblastech, jako je zpracování obrazu a signálu ([19], [3]), vojenství ([42]), medicíně ([20], [30]), počítačových hrách ([45]), umění ([36]) a mnoho dalších. Pro některé problémy dokonce poskytlo řešení, které překonalo všechna stávající nebo našlo inovativní řešení, které bylo možné patentovat nebo bylo v minulosti patentováno.

Cílem práce je navrhnout možné přístupy jak aplikovat genetické programování v oblasti finančních trhů a vybrané modely poté implementovat, analyzovat jejich výsledky a případně navrhnout možná rozšíření.

V rámci práce se zaměřuji na vývoj dvou principielně odlišných způsobů predikce cen aktiv. Prvním je klasický statistický přístup, kdy se na cenu díváme jako na časovou řadu a snažíme se vytvářet model, který by na základě předchozích hodnot dokázal predikovat hodnoty budoucí. Druhý přístup je tvorba obchodních systému, tedy agenta, jehož vstupem je historický vývoj ceny a ten realizuje na trhu konkrétní příkazy pro nákup a prodej aktiv s cílem maximalizovat svoji účelovou funkci, obvykle zisk. Hovoříme potom o trading agents.

Práce je strukturována následovně: nejprve v kapitole 2 uvádím související práci. Potom v kapitole 3 základní pojmy z genetického programování, finančních trhů a časových řad. V kapitole 4 pojednávám o tom, jak lze genetické programování aplikovat na tvorhu prediktivních modelů a trading agents. Kapitola 5 detailně popisuje všechny implementované modely, jejichž výsledky jsou potom detailně rozebrány v kapitole 6.

2. Související práce

Během své existence bylo genetické programování více či méně úspěšně použito na mnoho problémů souvisejících s finančními trhy a trading agents.

Některé fundamentální vlastnosti trhu, jako je závislost ceny a objemu zobchodovaného aktiva, studovali Chen a Liao v [22]. Martinez-Jarmillo a Tsang [35]. Postupovali tak, že vytvářeli virtuální finanční trhy s různými agenty mající stejné cíle jako agenti na stutečných trzích. Následně takto vzniklé modely porovnávali se skutečnými trhy a snažili se vysvětlovat jejich chování.

Prediktivní modely byly a jsou extenzivně studovanou oblastí. Chen, Wang a Zhang použili evoluční algoritmy pro predikování akciového indexu Hang-Seng. Ve své práci se zaměřovali zejména na oblast vysokofrekvenčních časových řad. Kaboudan ukázal, že genetické programování může být použito pro predikování měnových kurzů v Kaboudan [27] a akcií Kaboudan [26], Kaboudan [25]. Z hlediska teorie efektivních trhů jsou zajímavé výsledky pocházející od US Federal Reserve Bank. Nelly a Weller [38] a Nelly [37] tvrdí, že trhy jsou skutečně efektivní, protože se jim nepodařilo najít žádný systém, která by systematicky generoval zisky převyšující bezúročnou míru. Jejich práce však kritizovali Marney, Miller, Fyfe a Tarbert [34]. Nellz, Weller a Ulrich [40] dále poukázali, že předchozí výsledky více odpovídají teorii adaptivních trhů, kterou poprvé uvedl Lo [32]. Tato diskuse ukazuje použití genetického programování na skutečné testování teorie efektivních trhů, která byla po svém publikování obecně přijímána za platnou.

Také obchodní systémy byly studovány za užití genetického programování. Yu a Chen [47] tvrdí, že na studovaném indexu S&P 500 nalezli silné statistické důkazy o tom, že genetické programování je schopné najít úspěšné obchodní systémy generující zisk i při různých podmínkách na trhu. Dempster ve spolupráci s HSBC Global Markets úspěšně studoval tvorbu obchodních systémů pro trhy s měnami v pracech Austin, Bates, Dempster, Leemans a Williams [14], Dempster a Jones [16] a Dempster, Payne, Romahi a Thompson [17].

3. Základní pojmy

V této kapitole uvedeme základní pojmy, které budeme dále v textu používat.

Kapitola je rozčleněna do několika částí. V části 3.1 uvedu genetické programování (GP) jako nástroj pro globální optimalizaci včetně základních metod zde používaných. Předpokládá se od čtenáře předchozí základní znalost evolučních algoritmů. Část 3.3 definuje některé základní pojmy z finančního sektoru. Od čtenáře se nepředpokládá žádná předchozí znalost této problematiky. V části 3.2 uvádíme pojem časové řady a jejich předpovědí.

3.1 Genetické programování

Genetické programování je součást většího celku - evolučních algoritmů. Ty se obecně užívají jako nástroj pro globální optimalizaci. Jejich výhodou je, že je lze použít pro optizaci problémů bez toho, aniž by autor musel problému detailně rozumět. Navíc bylo v průběhu času ukázáno, že v některých oblastech dokáží poskytovat lepší řešení než člověk, nebo dokonce řešení, která jsou patentovatelná nebo byla již dříve úspěšně patentována. Velmi známým příkladem je anténa vytvořená pomocí genetického programování, která má netrdiční tvar a přesto splnila veškeré požadavky a byla nasazena v misi *Space Technology 5*. Detaily popsali Lohn, Hornby a Linden [33]

Genetické programování se velice podobá genetickým algoritmům a užívá se zde i mnoho přístupů z této oblasti. Rozdíl mezi nimi je v reprezentaci jedince a genetických operátorech s tím spojených.

3.1.1 Reprezentace jedince

Nejjednoduší formou reprezentace jedince je zde strom. Příklad takovéto reprezentace jedince je na obrázku 3.1. Ten odpovídá výrazu $\max\{x + y, x + 3 * y\}$. Vnitřní uzly (max, $+$, $*$) nazýváme funkce a listy stromu (x, y a 3) terminály. Dohromady potom tvoří množinu primitiv.

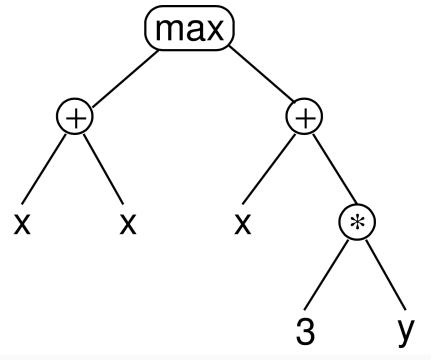
Množinu funkcí označujeme písmenem $\mathcal{F}$, množinu terminálů $\mathcal{T}$ a množinu primitiv $\mathcal{P}$.

Číslo

$$\frac{|\mathcal{T}|}{|\mathcal{T}| + |\mathcal{F}|} \tag{3.1}$$

nazýváme poměr terminálů.

Množina primitiv by měla splňovat alespoň dva požadavky - úplnost a uzavřenost.



Obrázek 3.1: Příklad jedince reprezentovaného stromovou strukturou.

Úplnost znamená, že je možné z dané množiny primitiv sestavit optimálního, nebo alespoň optimu blízkého jedince. Tento požadavek lze zajistit pouze v případech, kde předem víme, nebo máme představu o tom, jak by měl optimální jedinec vypadat.

Požadavek na uzavřenost znamená, že každá operace má dobře definovaný výstup pro všechny vstupy, které mohou po čas běhu algoritmu nastat. To může být problém například pro funkci podílu, kde ve jmenovateli nesmí být nula. Tento problém řešíme tak, že používáme bezpečné varianty těchto operátorů. Pro podíl může bezpečná varianta být definována jako

$$f_{/}(x,y) = \begin{cases} x/y & \text{pro } |y| > \epsilon \\ 1 & \text{jinak} \end{cases} \tag{3.2}$$

3.1.2 Inicializace populace

K inicializaci populace při takovéto reprezentaci používáme jednu ze tří metod *full*, *grow* nebo *Ramped half-and-half*.

Metoda *full* generuje jedince tak, že se předem zvolí náhodně z určeného rozsahu požadovaná hloubka stromu d. Potom se strom vytváří postupně od kořene z množiny přimitiv až do úrovně $d-1$. Na úrovni d se použijí pouze terminály. Název *full* je odvozen od toho, že všechny terminály leží ve stejné hloubce a strom je tak "plný".

Metoda *grow* je obdobná. Hloubka terminálů se ale nestanovuje předem. Určuje se pro každou větev stromu zvlášť z předem daného intervalu a navíc pravděpodobnost, že vygenerujeme terminál za předpokladu, že jsme v předem stanoveném intervalu hloubky je rovna poměru terminálů v množině primitiv. Obě metody implementuje algoritmus 1.

Ramped half-and-half je kombinací obou předchozích, kde v 50% případů se použije *full*, jinak *grow*.

Algoritmus 1 Algoritmus inicializace jedince podporující metody *full* i *grow*

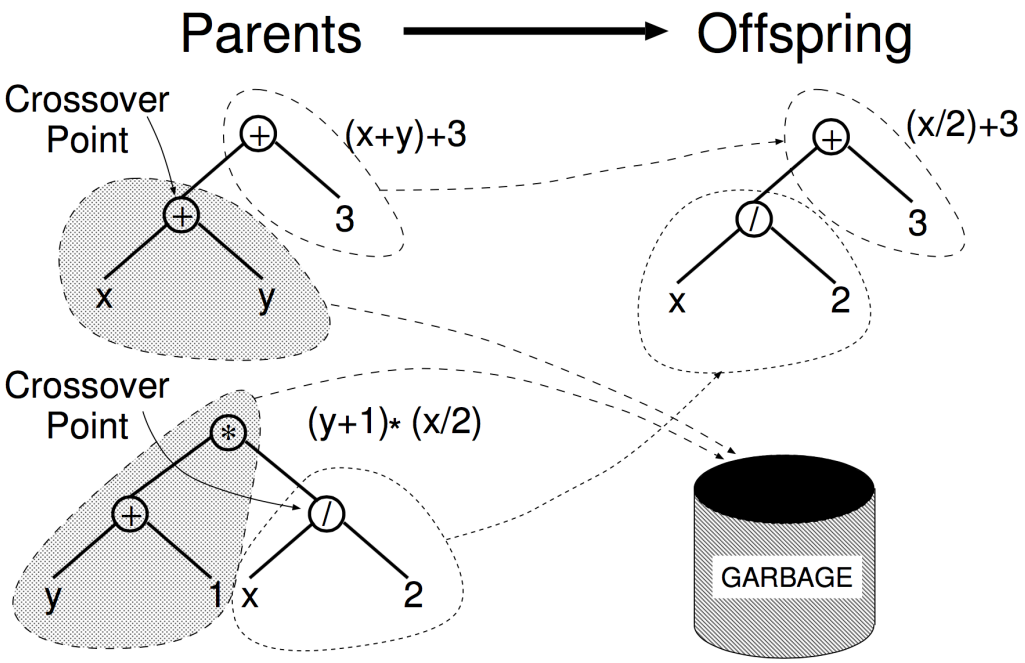
```
1: procedure GenRndExpr(F, T, MaxD, Method)
2:     F: množina funkcí
3:     T: množina terminálů
4:     MaxD: maximální hloubka stromu
5:     Method: použitá metoda (full nebo grow)
6:     if MaxD = 0 or (Method = grow and rand() < |T| / (|T|+|F|)) then
7:         expr = ChooseRandom(T)
8:     else
9:         func ← ChooseRandom(F)
10:        for i ← 0, arity(func) do
11:            arg_i ← GenRndExpr(F, T, MaxD, Method)
12:        end for
13:        expr ← (func, arg_1, arg_2, ...)
14:    end if
15:    return expr
16: end procedure
```

3.1.3 Selekce

Jako selekci můžeme použít libovolnou techniku používanou pro genetické algoritmy, obvyklá je potom ruletová nebo turnajová selekce.

3.1.4 Křížení

Nejčastěji užívaný operátor křížení pro genetické programování je obdobou jednobodového křížení používaného u genetických algoritmů. Operátor funguje tak, že na vstupu dostane dva jedince, náhodně vybere jeden podstrom z každého z nich, ten nazýváme bod křížení, a ty mezi sebou vymění. Tento operátor můžeme nejčastěji potkat ve variantě kdy vrací jediného jedince vybraného náhodne a druhého zahazuje. Méně častou variantou je implementace taková, že vrací jedince oba. Další variace na tento operátor může být operátor nevybírající body křížení náhodně, ale může preferovat nahrazování funkce malé resp. velké arity za funkce velké resp. malé arity. Tím můžeme ovlivnit růst stromu "do šířky"nebo "do hloubky". Koza [29] navrhl vybírat za body křížení v 90% případů funkce a ve zbylých 10% terminály a redukovat tak fakt, že jednobodové křížení má tendenci vybírat malé podstromy co do počtu uzlů. To může být nežádoucí, protože se mezi jedinci přenáší malé množství genetického materiálu. Ukázka jednobodového křížení je na obrázku 3.2.



Obrázek 3.2: Ukázka jednobodového křížení vracejícího jednoho jedince.

3.1.5 Mutace

Obvyklou implementací mutace je *subtree* mutace. Ta funguje tak, že náhodně zvolí podstrom a nahradí jej nově vygenerovaným jedincem. Ukázka takové mutace je na obrázku 3.3. Pokud nový jedinec sestává pouze z jediného terminálu, hovoříme potom o *shrink* mutaci.

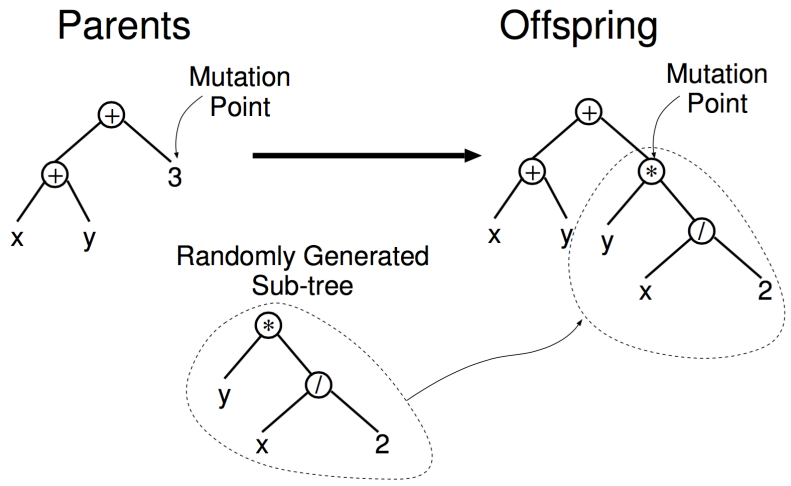
Pokud takovouto mutaci implementujeme pomocí operátorů křížení a generování nového jedince, nazýváme ji *headless-chicken* mutace.

Obdobou *bit-flip* mutace z genetických algoritmů je *subtree* mutace. Ta funguje tak, že náhodně zvolí uzel stromu a nahradí jej náhodným uzlem stejné arity a stejného typu (terminál nebo funkce).

3.1.6 Měření kvality jedince

Jediným ukazatelem kvality jedince je *fitness* funkce. Ta skrz selekci ovlivňuje průběh celého algoritmu.

Samotná *fitness* funkce může vyjadřovat nejrůznější míry, například chybu, které se model dopustil, přesnost se kterou prováděl požadované akce a další. Většinou ohodnocuje jedince reálnými čísly, případně vektorem reálných čísel pro vícekriteriální optimalizaci.



Obrázek 3.3: Ukázka *subtree* mutace

3.1.7 Terminační kritéria

Terminační kritéria určují, kdy se má evoluce zastavit. Užívaná kritéria jsou dosažení určitého počtu generací nebo vyhodnocení fitness funkce, dosažení dostatečně dobrého jedince, nebo se kvalita nejlepších jedinců dále nezlepšuje. Jako terminační kritérium můžeme také použít libovolnou kombinaci předešlých kritérií.

3.1.8 Silně typované genetické programování

Pro některé problémy předem známe přibližnou strukturu požadovaného jedince a chtěli bychom, aby se jí evoluce řídila. Možností je použít *silně typované genetické programování*, kdy každé hodnotě přiřadíme navíc typ. Tomu je třeba přizpůsobit i použité genetické operátory.

3.1.9 Vícekriteriální optimalizace

Vícekriteriální optimalizaci používáme v případech, kdy nelze jedince ohodnotit jedinou hodnotou, ale je třeba uvážit více různých kritérií.

Namísto klasické fitness funkce definujeme vektor kritérií jako $\mathbf{f} = (f_1, f_2, \ldots, f_n)$.

Definice 1. *Pro každé dva vektory obodnocení x a y řekneme že*

- *x slabě dominuje y ($x \preceq y$) pokud $\forall_i \in \{1, 2, \ldots n\} : f_i(x) \leq f_i(y)$.*
- *x dominuje y ($x \prec y$) pokud $\forall_i \in \{1, 2, \ldots n\} : f_i(x) < f_i(y)$.*
- *x a y jsou neporovnatelné, pokud neplatí $x \preceq y$ ani $y \preceq x$*
- *x nedominuje y ($x \not\preceq y$) pokud $y \preceq x$ nebo jsou neporovnatelné*

Definice 2. *Pareto optimální fronta je množina P^* všech jedinců takových, že neexistují žádné $x, y \in P^*$ takové, že $x \preceq y$.*

Definice 3. *Aproximace Pareto množiny je libovolná $P \subseteq P^*$.*

Pro porovnání jedinců v genetickém algoritmu můžeme použít *agregaci funkcí*, kde jedince ohodnotíme jako

$$f = \sum_{i=1}^{n} w_i f_i \tag{3.3}$$

pro nějaký jednotkový vektor $\mathbf{w}$. Tento přístup není moc vhodný zejména proto, že metoda nedává žádný návod, jak takový vektor vybrat.

Pokud n je malé, je možné modifikovat selekci tak, aby jedince vybírala na základě dominance. Pokud jsou jedinci neporovnatelní, selekce skončí neúspěchem a opakuje se.

Existuje řada algoritmů postavených na vícekriteriální optimalizaci. Za zmínku zejména stojí NSGA II [15]. Ten je založený na dominanci jedinců a obsahuje i elitismus respektující aproximaci Pareto množiny obsaženou v generaci.

3.2 Časové řady

Definice 4. Časová řada *je posloupnost pozorování $x_0, x_1, \ldots, x_t$ chronologicky uspořádaných v čase.*

Obvykle jsou navíc jednotlivá pozorování stejně vzdálena v čase. Hovoříme potom o časových řadách minutových, měsíčních, ročních a podobně.

Časová řada může vznikat při pozorování nějakého diskrétního nebo spojitého procesu. Ve spojitém případě proces diskretizujeme pomocí vzorkování.

3.2.1 Předpovídání časových řad

Předpověď časové řady vytvořenou v čase t pro čas $t + \tau, \tau \in \mathbb{N}_{>0}$ označujeme jako $\hat{x}_{t,t+\tau}$.

Každá předpověď je zatížená nějakou chybou, která může vznikat v důsledku náhodnosti predikovaného procesu nebo nedokonalsotí prediktivního modelu. Chyba předpovědi z času $t - \tau$ pro čas t je

$$e_t = x_t - \hat{x}_{t-\tau,t} \tag{3.4}$$

Vyhodnocení správnosti prediktivního modelu provádíme tak, že zakryjeme část posledních k pozorování, na nezakrytou část aplikujeme model a předpovědi porovnáme se zakrytými hodnotami, címž vznikne posloupnost chyb.

Abychom mohli porovnávat kvalitu různých modelů mezi sebou, používáme některou z měr podobnosti časových řad. Zřejmě nejjednodušší je střední chyba (MD)

$$MD = \frac{1}{k} \sum_{i=t-k}^{t} e_t \tag{3.5}$$

Vyhledem k tomu, že pro chyby symetricky distribuované okolo nuly, bude hodnota MD blízká nule se tato funkce používá málo, nebo ve spojení s rozptylem. Lze ji použít také jako míru vychýlenosti předpovědí.

Další funkcí je střední absolutní chyba (MAD).

$$MAD = \frac{1}{k} \sum_{i=t-k}^{t} |e_t| \tag{3.6}$$

Jednou z nejpoužívanějších funkcí je střední čtvercová chyba (MSE)

$$MSE = \frac{1}{k} \sum_{i=t-k}^{t} e_t^2 \tag{3.7}$$

Vzhledem k tomu, že hodnoty MSE nejsou ve stejných jednotkách jako původní časová řada, někdy se namísto ní používá odmocnina střední čtvercové chyby (RMSE), která již ve stejných jednotkách je.

$$RMSE = \sqrt{\frac{1}{k} \sum_{i=t-k}^{t} |e_t|} \tag{3.8}$$

Existují také míry nezávislé na měřítku původní časové řady. Příkladem je střední procentuální chyba (MPE)

$$MPE = \frac{1}{k} \sum_{i=t-k}^{t} \frac{e_t}{x_t} \tag{3.9}$$

Ta má stejné nedostatky jako MD. Z toho důvodu se častěji používá střední absolutní procentuální chyba (MAPE)

$$MAPE = \frac{1}{k} \sum_{i=t-k}^{t} \left| \frac{e_t}{x_t} \right| \tag{3.10}$$

Existuje a v praxi se používá mnoho dalších měr. Pro účely této práce si však vystačíme s těmito základními.

3.3 Finanční trhy

Finanční trhy je široký pojem zahrnující veškerá tržiště, kde se kupující a prodávající scházejí, aby spolu obchodovali finanční aktiva jako jsou akcie, měny, deriváty, dluhopisy a další.

Princip fungování trhů

Na finančních trzích proti sobě stojí kupující a prodávající. Obě strany komunikují skrz příkazy odesílané na burzu, čímž vyjadřují svůj záměr nakoupit nebo prodat podkladové aktivum za podmínek specifikovaných v příkazu. Pokud v danou chvíli na burze existují dva příkazy pro nákup a prodej stejného množství aktiva za stejnou cenu (až na bid-ask rozpětí, které bude popsáno níže), burza takový obchod uskuteční - exekuuje.

Bid je cena, za kterou jsou prodávající ochotni nakoupit aktivum. *Ask* je naopak cena, za kterou jsou prodávající ochotni aktivum prodat.

Vzhledem k tomu, že na trhu jsou různí účastníci ochotní koupit nebo prodat podkladové aktivum za různé ceny, je jako *bid* resp. *ask* cena brána tato cena u posledního obchodu provedeného nad tímto aktivem.

Neshoda na *bid* a *ask* cenách se označuje jako *hloubka trhu*.

Aby se usnadnilo oběma stranám exekuovat příkazy (zvýšila se likvidita trhu), vystupuje se často na trzích tzv. tvůrce trhu. To je subjekt, který stojí na obou stranách a kupuje a prodává aktivum proto, aby jej mohl poté (často jen za několik málo milisekund) podstoupit protistraně. Při této transakci však změní cenu ve svůj prospěch o *bid-ask* rozpětí. Mezi likviditou trhu a velikostí *bid-ask* rozpětí existuje nepřímá úměra - likvidnější trhy jej mají obvykle menší.

Teorie efektivního trhu

Lo [32] ve spojení s touto teorií uvádí anekdotu, která ji podle něj blízce vystihuje: Prochází se spolek ekonomů, když v tom uvidí stodolarovou bankovku. Jeden z ekonomů se k ní shýbá, aby ji sebral, když v tom ho zastaví jiný a říká: "Neobtěžuj se. Kdyby to byla opravdová bankovka, někdo by ji už sebral".

Teorie efektivních trůh říká, že trh je složen z racionálních agentů majících úplnou informaci. To fakticky znamená, že je trh nepředvídatelný pomocí prostředků technické analýzy.

Od roku 1965, kdy s teorií efektivních trhů přišel Paul Samuelson, byla tato teorie s různými výsledky testována a v současné době není jednoznačná shoda akademické obce ohledně její platnosti. Alternativou k ní je teorie adaptivního trhu.

3.3.1 Finanční instrumenty

Na finančních trzích se obchoduje mnoho různých instrumentů, které se podstatně liší svými vlastnostmi.

- **Akcie** jsou cenné papíry opravňující jejich držitele participovat na rozhodování a zisku společnosti, která tyto akcie vydala. Obchodování s nimi je jednou z nejjednodušších forem finančních trhů.
- **Futures** vyjadřují závazek vystavovatele dodat v předem daný čas smluvené množství podkladového aktiva za smluvenou cenu. Tento instrument je historicky motivovaný snahou zemědělců pojistit se proti nejistotě v rozdílu ceny rýže v době jejího sázení a sklízení. Běžné je, že pro jedno podkladové aktivum existuje jeden futures kontrakt s datem dodání pro každý měsíc po dobu až několika let. Futures mohou být vystaveny na libovolné podkladové aktivum. Běžné jsou komoditní futures.
- **Forwardy** jsou podobné futures, ale může se s nimi obchodovat mimoburzovně a operace *mark to market* ([23]) na nich, narozdíl od futures, neprobíhá denně.
- **Opce** je předkupní právo opravňujícího jejího držitele, který za ni zaplatil *opční prémium*, k nákupu v případě *put opce* nebo prodeji v případě *call opce* podkladového aktiva za předem stanovenou cenu v předem stanoveném čase. V případě, že se cena podkladového aktiva nevyvíjí ve prospěch jejího držitele, nemusí tuto opci využít.

V souvislosti s finančními trhy budeme používat některých termínů z oblasti finančních trhů a proto je zde vysvětlíme.

- **Broker** - Protože obchodování na některých trzích vyžaduje speciální licenci, která by byla pro většinu subjektů obtížně získatelná, můžeme využít služeb brokera, který tuto licenci má a za úplatu nám povolí obchodovat na jeho účet.
- **Podkladové aktivum** - Některé instrumenty jsou založené na konkrétní aktivum. Pro futures kontrakty to může být káva, obiloviny, kukuřice, vepřové maso a podobně, pro opce je to instrument, na který nám vzniká nárok nákupem opce.
- **Kontrakt** - Aby se proces obchodování s futures zjednodušil, vytvořily se standardizované kontrakty, se kterými je možné na finančních trzích obchodovat. Kontrakt se vztahuje na dodání v předem stanovený čas daného množství podkladového aktiva.

- **Ticker, symbol** - Abychom mohli jednotlivé kontrakty snadno identifikovat, mají přidělené jednoznačné jméno, tedy symbol (například ES pro e-mini S&P 500, GC pro zlato a další). Specifikováním měsíce, kdy máme podkladové aktivum kontraktu dodat vznikne jeho jednoznačný identifikátor - ticker (například pro e-mini S&P 500 s datem dodání v prosinci 2016 bude mít označení ESZ16). Pro ostatní trhy se pojmy symbol a ticker užívají zaměnitelně a znamenají identifikátor daného instrumentu.

3.3.2 Časové řady na finančních trzích

Každý obchod je zaznamenán v podobě tzv. *ticku*. Ty se potom agregují do časových řad a dále zpracovávají.

Definice 5. Tick t *je trojice* (τ, p, v) *kde*

- τ *je čas, ve kterém obchod proběhl*
- p *je cena obchodu*
- v *je počet zobchodovaných lotů,* $v > 0$ *značí nákup a* $v < 0$ *prodej aktiva*

Definice 6. *Posloupnost* $t_0, t_1, \ldots, t_n$ *nazýváme* tick-by-tick *časovou řadou.*

Definice 7. *Agregovaná časová řada je posloupnost šestic* (τ', O, H, L, C, V)*, kde každá šestice reprezentuje nějakou souvislou disjunktní podposloupnost* $t_{k_1}, t_{k_2}, \ldots, t_{k_m}$ tick-by-tick *časové řady a*

- τ' *je čas* τ *ticku* t_{k_1}
- $O = p(t_{k_1})$ *je otevírací,*
- $H = \max\limits_{j \in \{1,2,\ldots,m\}} p(t_{k_j})$ *je nejvyšší,*
- $L = \min\limits_{j \in \{1,2,\ldots,m\}} p(t_{k_j})$ *je nejnižší,*
- $C = p(t_{k_m})$ *je uzavírací cena a*
- $V = \frac{1}{2} \sum\limits_{j=1}^{m} \left| v(t_{k_j}) \right|$ *je zobchodovaný objem.*

Pokud jsou navíc podposloupnosti z definice agregované časové řady takové, že každá podposloupnost odpovídá jedné minutě, hovoříme o minutové časové řadě. Stejně tak se používají pětiminutové, čtvrthodinové, hodinové, denní, týdenní, roční a další časové řady.

V případě, že je každá podposlounost dlouhá právě n ticků, nazýváme ji n-tickovou časovou řadou.

Poslední z používaných agregací je V-objemová, kdy podposloupnosti $t_{k_1}, t_{k_2}, \ldots, t_{k_m}$ jsou takové, že

$$\frac{1}{2}\sum_{j=1}^{m} \left|v(t_{k_j})\right| = V$$

Tick-by-tick data s hloubkou trhu jsou nejdetailnější informace, které lze přímo z trhů získat.

S daty z finančních trhů se pojí několik problémů.

- **Kvantita**: Ne všichni participanti finančních trhů mají k dispozici stejná data. To může být způsobeno tím, že nějaký *broker* disponuje *clearing house*m a příkazy tedy nepropaguje přímo na burzu, ale pokud může, vyřizuje je z interních zdrojů (broker sám vystupuje jako tvůrce trhu nebo párováním s příkazy ostatních klientů).
- **Kvantita**: Ze své povahy finanční trhy generují velké množství dat. To současně s požadavkem na rychlou odezvu klade velké nároky i na současný hardware.
- **Cena**: *Intradenní tick-by-tick* data jsou velmi drahá. Cena takovýchto dat se pohybuje ve stovkách dolarů za *ticker*.

3.3.3 Analýza trhu

Abychom byli schopni na trhu realizovat zisky, je třeba ho analyzovat a predikovat jeho budoucí vývoj. Takovou analýzu dělíme do dvou odvětví

Fundamentální analýza

Fundamentální analýza je přístup, kdy prognózy tvoříme na základě znalostí o podkladovém aktivu a faktorech jej ovlivňujících. Tento přístup se více užívá u obchodů uzavíraných na období delší než dny až týdny.

Technická analýza

Technická analýza naopak využívá pouze znalosti historických cen aktiva. Pro takovou analýzu pak používáme různé nástroje

- *Indikátory* pomocí transformování historických dat do různé podoby pomáhají ochodníkům vyhlazovat jinak zašuměné grafy, měřit volatilitu trhů nebo identifikovat překoupené nebo přeprodané trhy.
- *Price-action* zkoumá historickou cenu pomocí vyhledávání různých vzorů v jejich grafech.

- Analýza *průlomů* využívá faktu, že trhy často odolávají určitým cenovým hranicím vytvořeným iracionálním chováním obchodníků, a v případě jejich prolomení očekává dramatickou změnu ceny.

Pod technickou analýzu také spadají všechny přístupy strojového učení, včetně genetického programování.

Indikátory

Jedním z nejpoužívanějších indikátorů je jednoduchý klouzavý průměr (SMA), který má jeden parametr.

$$SMA = \frac{1}{n}(x_t + x_{t-1} + \ldots + x_{t-n}) \tag{3.11}$$

SMA se používá pro vyhlazování cenového grafu, aby v něm byly lépe vidět různé vzory. Čím větší n je, tím hladší je výsledná křivka. Pokud nejsou data symetricky rozložena okolo svého průměru, jeví se výsledná křivka jako posunutá vůči původnímu grafu o $n/2$. Proto se místo SMA někdy používá centrovaný jednoduchý klouzavý průměr (CMA)

$$CMA = \frac{1}{n}(x_{t-n/2} + x_{t-n/2+1} + \ldots + x_{t+n/2-1} + x_{t+n/2}) \tag{3.12}$$

pro nějaké n liché číslo. Hlavní nevýhodou CMA je, že samotná křivka je oproti původní řadě zkrácená o $n/2$.

Jiným možným řešením problému je použít vážený průměr. Podle vah, které použijeme rozlišujeme několik druhů průměrů. Vážený klouzavý průměr (WMA) je druh klouzavého průměru, kde jako váhy použijeme lineárně klesající posloupnost

$$WMA = \frac{nx_t + (n-1)x_{t-1} + \ldots + x_{t-n}}{n + (n-1) + \ldots + 2 + 1} = 2\frac{nx_t + (n-1)x_{t-1} + \ldots + x_{t-n}}{n(n-1)} \tag{3.13}$$

Alternativně můžeme namísto lineárně klesajících vah použít váhy klesající geometrickou řadou. Takovému průměru se potom říká exponenciální klouzavý průměr (EMA) a jako parametr má hodnotu $\alpha \in (0, 1)$, která určuje kvocient geometrické řady.

$$EMA = \frac{x_t + (1-\alpha)x_{t-1} + (1-\alpha)^2 x_{t-2} + \ldots (1-\alpha)^t x_0}{1 + (1-\alpha) + (1-\alpha)^2 + \ldots + (1-\alpha)^t} \tag{3.14}$$

Tyto indikátory jsou ve stejných jednotkách jako cena samotná, a v grafu se většinou zakreslují přes ni. Existují také indikátory, které mají vlastní interpre-

taci. Mezi ty patří i commidity channel index (CCI), který je definován jako

$$CCI = \frac{1}{0.015}\frac{p_t - SMA(p_t, p_{t-1}, \ldots, p_{t-n})}{\sigma(p_t, p_{t-1}, \ldots, p_{t-n})} \tag{3.15}$$

kde p_t je tzv. typická cena v čase t definovaná jako $\frac{H+L+C}{3}$ a

$$\sigma(x_1, x_2, \ldots, x_n) = \frac{1}{n}\sum_{i=1}^{n}\left|x_i - \frac{1}{n}\sum_{j=1}^{n} x_i\right|$$

Autor tohoto indikátoru Donald R. Lambert ve své práci [31] uvádí, že konstanta 0.015 byla zvolena tak, aby přibližně 70% až 80% hodnot leželo v rozmezí od +100 do -100. Tamtéž také navrhl, že pokud indikátor překročí hranici +100, znamená to signál pro nákup. Návrat pod tuto hodnotu značí uzavření pozice. Naopak překročení hodnoty -100 směrem dolů znamená vstup do krátké pozice a návrat nad ni její uzavření.

Obdobně se užívá relative streight index (RSI), který je definován jako

$$RSI = 100 - \frac{100}{1 - RS} \tag{3.16}$$

kde

$$RS = \frac{\sum_{i=1}^{n}\max\left(x_{t-i+1} - x_{t-i}, 0\right)}{\sum_{i=1}^{n}\min\left(x_{t-i+1} - x_{t-i}, 0\right)}$$

RSI nabývá hodnot mezi 0 a 100 a jako hranice pro dlouhou pozici se používá 70 a pro krátkou 30, jinak je systém stejný jako v případě CCI.

Další z oblíbených indikátorů je moving average convergence divergence (MACD). Ten je definovaný jako

$$RSI = EMA_{26} - EMA_{12} \tag{3.17}$$

do stejného grafu se pak vykreslí tzv. signální hladina - EMA_9. Pokud MACD překročí signální hladinu směrem dolů, znamená to signál k prodeji. Naopak překročení signální hladiny směrem nahoru znamená signál k nákupu.

Existují také indikátory, které nejsou určené jako indikátory trendu. Jeden takový je average true range (ATR). Ten slouží jako indikátor volatility. To může napomoci identifikovat období zejména vhodné pro vstup do pozice.

$$ATR = \frac{1}{n}\sum_{i=1}^{n} TR_{t-i+1} \tag{3.18}$$

kde

$$TR_t = \max\left(H_t - L_t, \left|H_t - C_{t-1}\right|, \left|L_t - C_{t-1}\right|\right)$$

Existuje a v praxi se používá mnoho dalších indikátorů, pro účely práce si však vystačíme s těmito.

4. Prediktivní modely

V této kapitole navrhneme několik možných přístupů v predikci finančních trhů s využitím genetického programování.

Předkládáme zde pouze některé možnosti jak zakódovat úlohu predikování finančních trhů do jedinců a jejich různé *fitness* funkce, které pro taková zakódování lze použít. Ostatní parametry genetického programování lze libovolně měnit nezávisle na zakódování, a nebudeme je proto zde rozebírat.

V rámci práce byly zkoumány dva možné přístupy k predikci finančních trhů. Prvním je přístup klasického predikování časových řad, kdy se snažíme predikovat přesnou hodnotu časové řady v předem daném okamžiku. Pro tento účel existuje široké množství metod klasické analýzy časových řad. Druhý přístup je tvorba obchodních systémů, které pomocí příkazů pro nákup a prodej vytvářejí portfolio s cílem maximalizovat zisk. Tento přístup lze také považovat za metodu predikce vzhledem k tomu, že příkazem pro nákup resp. prodej kontraktu vyjadřujeme domněnku, že cena kontraktu bude růst resp. klesat. Tato metoda je však jednodušší na řešení, protože nespecifikuje kdy a o jakou vydálenost se trh změní.

Pozorování 1. *Mějme dvě shodné minimalizační úlohy lišící se pouze aritou modelu. Potom platí, že úloha s větší aritou má alespoň tak dobré optimální řešení jako úloha druhá.*

Důkaz. Mějme dvě úlohy A_1 a A_2 jako v tvrzení a funkci f, dle které optimalizují. Bez újmy na obecnosti nechť model úlohy A_1 má větší aritu než úlohy A_2. Dále buď c_1^* optimální řešení A_1 a c_2^* optimální řešení A_2.

Chceme ukázat, že

$$f(c_1^*) \leq f(c_2^*)$$

To platí proto, že c_2^* je také přípustné řešení A_1. Další parametry obsažené v A_1 oproti A_2 pak mohou řešení pouze zlepšit. □

4.1 Predikování časových řad

Výstupem finančního trhu je časová řada X, řízená procesem

$$x_t = f(x_{t-1}, \ldots, x_{t_0}, E) + \epsilon_t$$

kde E jsou vnější vlivy (například sezonnost, vlivy počasí, události jako vydání nového produktu společností a pod.) a $\epsilon_t \sim iid(\mu = 0, \sigma = \epsilon)$.

Některé složky časové řady nám však nejsou známe nebo jsou velmi obtížně zjistitelné. Proto se snažíme řadu modelovat jako zjednodušený proces odpovídající

$$x_t = f(x_t, t_{t-1}, \ldots, x_{t-k})$$

Některé z možných modelů jsou vyjmenovány v odstavcích déle.

Predikování ceny o τ kroků dopředu

Přímočarý způsob, jak implementovat predikci je využít standardní reprezentaci jedince a modelovat hodnotu $x_{t,t+\tau}$.

Fitness funkcí je potom libovolná z metrik uvedených v sekci 3.2.1.

Takovýto přístup imituje klasické statistické metody predikování časových řad.

Predikování změny ceny o τ kroků dopředu

Tento přístup je shodný s předchozím, ale snažíme se předpovídat změnu ceny, tedy číslo $x_{t,t+\tau} - x_t$.

Predikování směru

Jednodušší variantou předchozího je odhadovat pouze $\text{sign}(x_{t,t+\tau} - x_t)$.

Za fitness funkci pro tento model lze vzít procento špatně klasifikovaných případů.

Tento přístup však není prakticky použitelný vzhledem k tomu, že neudává jak hodně se trh pohne a vzhledem k tomu, že většina změn je blízkých nule přičemž obchodník inkasuje největší zisky při velkých pohybech.

Predikování směru s možností "nevím"

Nedostatek předchozího přístupu lze odstranit tak, že přidáme modelu možnost neodpovědět. Očekávaný výstup je, že pro velké pohyby vydá předpověď, zatímco pro malé neodpoví.

Fitness funkce je stejná jako v předchozím případě s tím, že v případech, kdy model neučinil předpověď, penalizujeme číslem $\epsilon > 0$. Tím docílíme, aby model neodpovídal pouze "nevím". Pro ϵ velké je tento přístup shodný s předchozím. Možným rozšířením je penalizace za nepředpovězení směru při velkém pohybu. To nutí algoritmus k tomu, aby neodpovídal "nevím"v případě velkých pohybů, což není žádoucí.

4.2 Obchodní systémy

Pro sestavení experimentů s obchodními systémy pomocí genetického programování potřebujeme vždy stanovit alespoň tři parametry: fitness funkci, výstup modelu a obsažená primitiva.

4.2.1 Fitness funkce

V případě prediktivních modelů je vždy zřejmé, co má být správný výstup. Pro obchodní systémy je situace složitější. Protože nelze říct, jaká má být odpověď algoritmu po každém kroku, obvykle se zaměřujeme na porovnávání výkonů portfolií sestávajících z této jediné modelované strategie. Pro porovnávání portfólií již metody existují, nevedou však na jednoznačné kritérium, podle kterého bychom měli optimalizovat. Klasickým případem je *risk-reward-ratio*, kdy musíme volit mezi riskantními obchody, které mohou vést k větším ziskům za cenu toho, že zisky portfolia jsou značně turbulentní, nebo méně riskantní přístup, který ale z pravidla vede k nižším ziskům.

Příkladem kritérií může být

- **Konečná hodnota portfolia**. To znamená součet množství peněz na účtu a hodnoty všech aktiv držených v otevřených pozicích.
- **Maximální drawdown**. Drawdown je procento o které se zmenší hodnota účtu po řadě ztrátových obchodů.
- **Sharpeho poměr** [43]. Je jedním z řady poměrových ukazatelů, definovaný jako

$$S_a = \frac{E[r - r_f]}{\sqrt{\text{var}(r_i - r_f)}} \tag{4.1}$$

  kde r je průběh hodnoty portfolia a r_f je benchmark, obvykle bezúročná míra. Pro intradenní strategie se vynechává benchmark, protože se mění na úrovni dní. Sharpeho poměr je kritizován za to, že penalizuje stejně zisky i ztráty.

- **Teynorův poměr** [46]. Je definován jako

$$\text{Teynor}_i = \frac{E[r_i] - r_f}{\beta_i} \tag{4.2}$$

  kde β_i je koeficient u lineárního členu z rovnice lineálrní regrese vedené křivkou zisků portfolia.

- **Jensenova alfa** [24] Užívá se pro měření zisků, které přesahují očekávaný zisk buy-and-hold strategie. Je definována jako

$$\alpha_i = E[r_i] - r_f - \beta_i(r_M - r_f) \tag{4.3}$$

kde r_M jsou zisky buy-and-hold strategie na stejném trhu.

- **Omega** [28] Je poměr

$$\Omega_i = \frac{E[r_i] - \tau}{\mathrm{LPM}_{1i}(\tau)} + 1 \tag{4.4}$$

kde τ je minimální přijatelný zisk a LMP (lower partial moment) je

$$\mathrm{LMP}_{ni}(\tau) = \frac{1}{T}\sum_{t=1}^{T} \max[\tau - r_{it}, 0]^n \tag{4.5}$$

- **Sortinův poměr** [44] Je číslo

$$\mathrm{Sortino}_i = \frac{E[r_i] - \tau}{\sqrt{\mathrm{LPM}_{2i}(\tau)}} \tag{4.6}$$

Vedle výše zmíněných měr existuje a užívá se množství dalších a zřejmě neexistuje jediná míra, která by byla dostatečně vypovídající. Obvykle proto používáme více měr najednou.

4.2.2 Výstup modelu

Výsledek experimentu může značně záviset na volbě výstupu, který model poskytuje. Uvedeme zde některé možné.

- **Příkazy**. Sestavit model tak, aby jako výstup dával konkrétní příkazy, které se mají exekuovat je velmi přirozené, protože i obchodník tímto způsobem komunikuje s trhy. Takovýto přístup ale skrývá alepoň dvě nevýhody. Tou první je, že výstupem je vždy právě jeden příkaz, což není omezující pouze za předpokladu, že pracujeme nad jediným aktivem. V opačném případě to ale dělá tento přístup velmi limitující. Další nevýhodou je, že genetické programování se bude obtížně startovat (bootstrap problem), protože počáteční řešení budou exekuovat příkazy náhodně a najít tak nějaké řešení, které je dostatečně dobré je obtížnější.

- **Stav portfolia**. Dalším možným výstupem je cílový stav portfolia s tím, že výstup zpracujeme, a do podoby konkrétních příkazů převedeme ručně. Tím částečně eliminujeme obě předchozí nevýhody předchozího přístupu. První,

protože jako výstup můžeme očekávat $n$tici, kde každý prvek představuje cílový stav daného aktiva. Druhou, protože příkazy vytváří implementátor modelu a má tak úplnou kontrolu nad jejich podobou.

Vzhledem k tomu, že v rámci takovýchto modelů pracujeme s různými typy, je třeba použít silně typované genetické programování.

4.2.3 Obsažená primitiva

Narozdíl od prediktivních modelů, kde jako primitiva byly obsažené pouze standardní matematické funkce, u obchodních systémů pracujeme s různými typy a pro každý typ musíme přidat alespoň základní sadu primitiv pro práci s ním a pro přetypování.

Dále je třeba zahrnout primitiva pro práci s trhy samotnými.

První skupinou takových primitiv jsou indikátory samotné. Tím poskytneme modelu možnost pracovat s historickými cenami.

Dále můžeme přidat funkce cross_above a cross_below, které se velmi často používají v rámci jednoduchých strategií, definované jako

$$\text{cross\_above}(I_1, I_2) = I_1(t-1) < I_2(t-1) \wedge I_1(t) > I_2(t) \tag{4.7}$$

$$\text{cross\_below}(I_1, I_2) = I_1(t-1) > I_2(t-1) \wedge I_1(t) < I_2(t) \tag{4.8}$$

Další možností je předprogramovat některé známé strategie, například ve formě funkce vracející *True*, pokud strategie vygenerovala signál k nákupu, jinak *False*. Takovými strategiemi mohou být například:

- **Dvojitý klouzavý průměr**. V rámci této strategie máme dva klouzavé průměry (libovolný typ klouzavého průměru, nejčastěji jednoduchý) s různými periodami. V případě, že průměr s kratší periodou přetne druhý průměr zespoda nahoru, znamená to signál k nákupu. V případě, kdy jej přetne opačným směrem, je to signál k prodeji.
- **Strategie založená na CCI**. Do grafu umístíme indikátor CCI a v případě, že překročí hranici +100, je to signál k nákupu. V případě, že opět klesne pod tuto hranici, je to signál pro uzavření této pozice. Opačně je tomu tak pro hranici −100 a signál k prodeji.

Použít lze také libovolnou strategii založenou na *price-action*, nebo strategii popsanou v části 3.3.3.

5. Implementace vybraných metod

V této kapitole popíšeme implementaci všech provedených experimentů.

5.1 Podmínky experimentů

5.1.1 Programové vybavení

Všechny experimenty byly naprogramovány v Pythonu za použití knihoven třetích stran. Vzhledem k nekompatibilitě některých knihoven uvádím v tabulce 5.1 seznam všech knihoven včetně jejich verzí použitých v experimentech.

Za zmínku stojí zejména knihovny Distributed Evolutionary Algorithms in Python (DEAP) [18] a Zipline [11].

DEAP je framework pro evoluční algoritmy. Jeho hlavní výhodou je jednoduchost se kterou lze zapisovat i relativně složité programy na několik málo řádků. V neposlední řadě disponuje také snadnou podporou pro paralelní a distribuované výpočty.

Zipline je knihovna pro algoritmické obchodování na akciových trzích. Tato knihovna také slouží jako jádro pro open-source obchodní platformu Quantopian [9]. Lze ji použít pro tvorbu jednoduchých i středně složitých strategií. Knihovna byla použita zejména proto, že je aktivně vyvíjena a obsahuje mnoho funkcionalit, které jsou při vlastním programování poměrně náchylné na chyby vytvořené programátorem a které mohou mít za následek znehodnocení celého experimentu. Nevýhodou je zejména fakt, že je určena pouze pro obchodování na akciových trzích a ne pro futures, které jsou velmi volatilní a tak potenciálně velmi ziskové. Vzorově implementovaná strategie Buy and hold pro ticker AAPL je implementovaná v algoritmu 2.

Analýza algoritmů byla provedena v prostředí IPython notebook [5], které umožňuje interaktivní práci se zdrojovým kódem. Pro vykreslování obrázků byla použita knihovna Matplotlib [6] a pro grafy pygraphviz [7].

5.1.2 Historická data

Pro experimenty byly vybrány akciové trhy z důvodu dostupnosti historických cen a snadné proveditelnosti experimentů pomocí dostupného programového vybavení. S ostatními druhy finančních trhů jsou navíc spojené další problémy, které mohou negativně ovlivnit výsledek experimentu, příkladem je volba vedoucího

Algoritmus 2 Buy and hold strategie pro AAPL

```
def initialize(context):
    context.i = 0

def handle_data(context, data):
    if context.i == 0:
        order_target(symbol('AAPL'), 100)

    context.i += 1
    record(ticker=data[symbol('AAPL')].price)

algo = TradingAlgorithm(initialize=initialize,
                        handle_data=handle_data)
algo.run(data)
```

alabaster==0.7.4	mistune==0.5.1	requests==2.7.0
Babel==1.3	nose==1.3.6	scipy==0.15.1
certifi==2015.4.28	numpy==1.9.2	scoop==0.5.3
Cython==0.22	numpydoc==0.5	six==1.9.0
deap==1.1.0	pandas==0.16.0	snowballstemmer==1.2.0
docutils==0.12	patsy==0.3.0	Sphinx==1.3.1
gnureadline==6.3.3	ptyprocess==0.4	sphinx-rtd-theme==0.1.7
greenlet==0.4.6	Pygments==2.0.2	statsmodels==0.6.1
ipython==3.1.0	pygraphviz==1.3rc2	TA-Lib==0.4.8
Jinja2==2.7.3	pyparsing==2.0.3	terminado==0.5
jsonschema==2.4.0	python-dateutil==2.4.2	tornado==4.1
Logbook==0.9.0	pytz==2015.2	zipline==0.7.0
MarkupSafe==0.23	pyzmq==14.6.0	
matplotlib==1.4.3	Quandl==2.8.6	

Tabulka 5.1: Seznam balíčků včetně verzí používých v experimentech

měsíce (kontraktu) na kterém budou obchody probíhat, zda-li zpětně upravovat cenu při přechodu mezi měsíci, margin call a další.

Volně dostupná data jsou zejména ze serverů Quandl [8], Google Finance [4] a Yahoo Finance [10]. Jejich porovnání je v tabulce 5.2. Vzhledem k dostupnosti intradenních dat i oficiálního API jsem zvolil jako zdroj veškerých dat Yahoo Finance.

Zdroj	Intradenní data	Dostupné API
Quandl	ne	ano
Google Finance	ano - minutová data	ne
Yahoo Finance	ano - minutová data	ano

Tabulka 5.2: Seznam zdrojů volně dostupných databází hisotrických dat finančních trhů

Intradenní jsou dostupná pouze za několik málo posledních dní, proto byla v pravidelném intervalu stahována a ukládána z veřejně dostupných zdrojů. Tato databáze je dostupná na přiloženém CD. Denní data jsou dostupná v dostatečné délce a proto nejsou na CD obsažena a stahují se před každým během experimentu.

5.1.3 Paralelizace a distribuování výpočtů

Vzhledem k velké časové náročnosti výpočtů bylo nutné, aby celý výpočet probíhal paralelně. Klasická paralelizace v Pythonu však vzhledem k jeho různým omezením téměř není možná, proto byla implementována pomocí více navzájem komunikujících procesů. Komunikace mezi procesy probíhala pomocí síťové komunikace, což přineslo teoreticky velmi snadné distribuování výpočtů na více strojů.

V praxi se však ukázalo, že zajistit dostatečnou dostupnost několika strojů, stejnou verzi zdrojových kódů na všech počítačích a stejné nebo kompatibilní verze knihoven je poměrně náročné. Proto nakonec výpočty probíhaly paralelně na jediném stroji.

5.1.4 Checkpointy

V průběhu simulace se vždy po každé generaci provedlo kompletní uložení stavu, včetně stavu generátoru náhodných čísel. K tomuto kroku bylo nutné přikročit, aby se předešlo ztrátě mezivýsledků při přerušení běhu algoritmu z libovolného důvodu, které se pro dlouhé výpočty ukázalo jako relativně časté.

ticker	p-value
AAPL	$2{\cdot}662419e{-}30$
PEP	$1{\cdot}989910e{-}23$
KO	$3{\cdot}233230e{-}24$
^GSPC	$7{\cdot}666568e{-}26$

Tabulka 5.3: p-hodnoty podle Shapiro-Wilkova testu normality pro vybrané tickery

5.1.5 Benchmark

Pro oba přístupy jsme stanovili jednoduché benchmarky, vůči kterým poměřujeme kvalitu prediktivních modelů. Při výběru benchmarků jsme volili co nejjednodušší modely, které lze velmi snadno interpretovat a zároveň se používají v odborné literatuře.

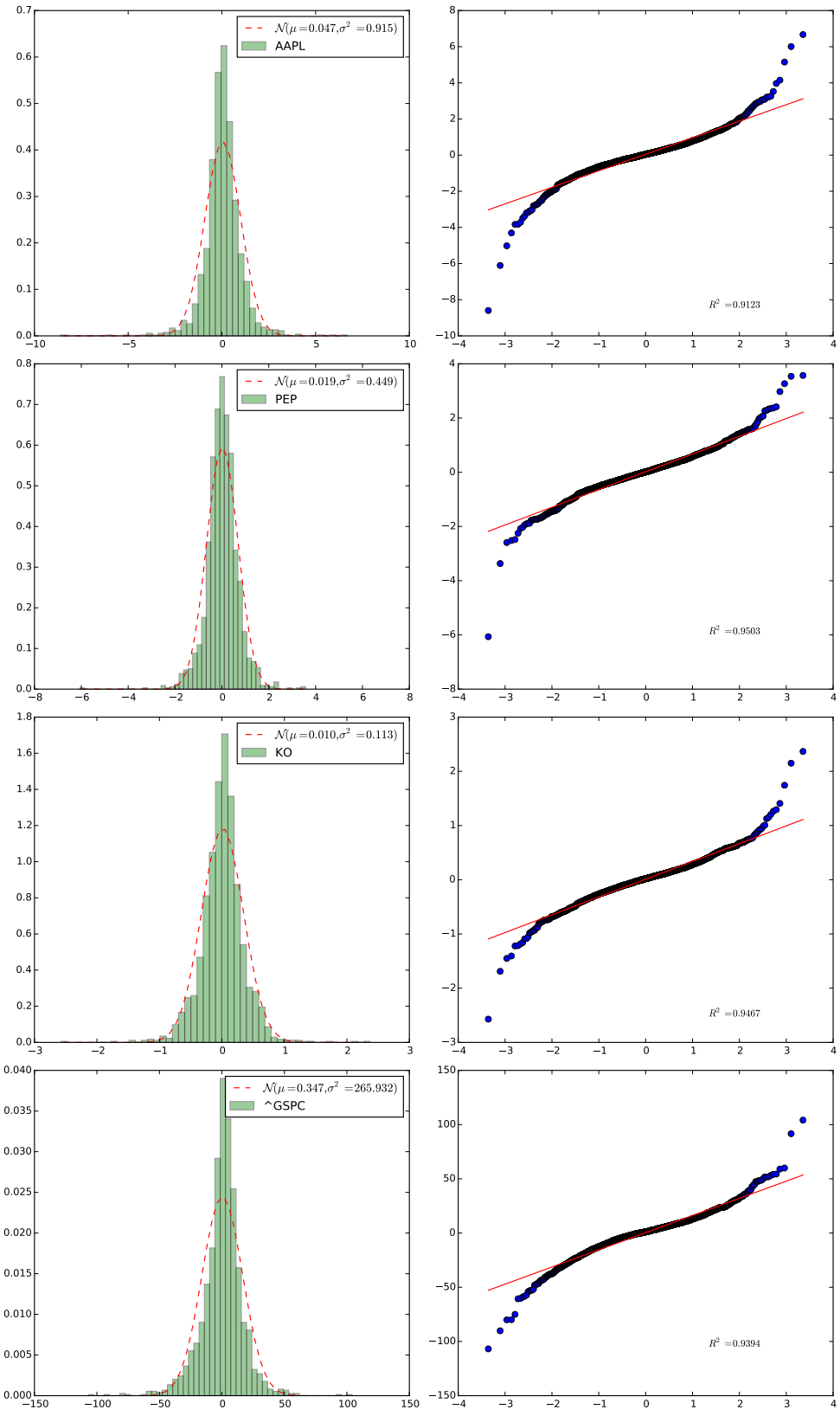
Pro prediktivní modely jsme jako benchmark zvolili funkci

$$x_t = x_{t-1} \tag{5.1}$$

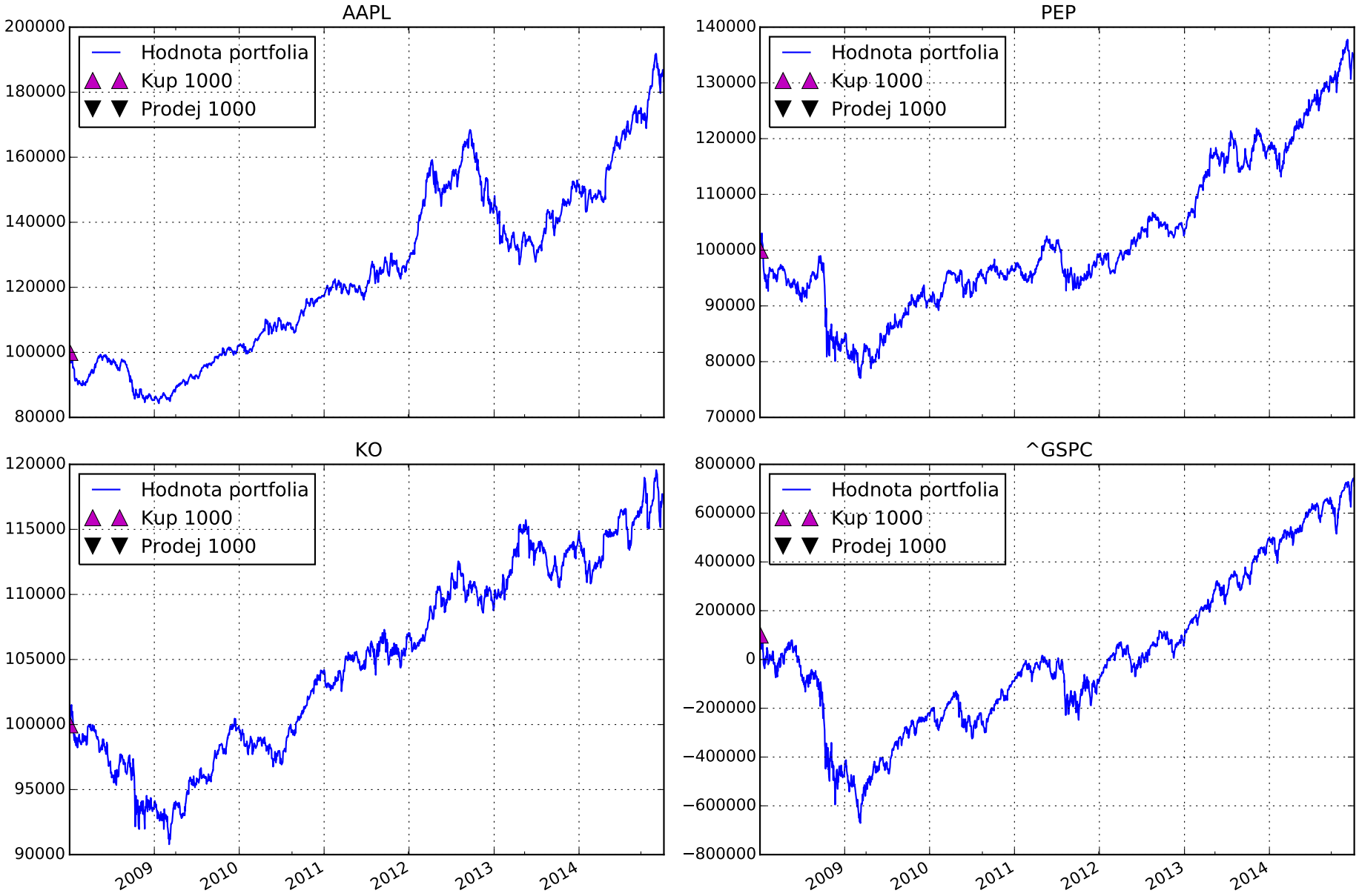
Histogram reziduí tohoto modelu je znázorněn na obrázku 5.1 společně s Q-Q grafy reziduí tohoto modelu pro vybrané tickery. Dle histogramu je vidět, že rezidua jsou přibližně normálně rozdělená, na Q-Q grafu se však ukazuje, že ani pro jeden ticker neodpovídají normálnímu rozdělení, zejména v oblastech daleko od průměru. Toto tvrzení dokazuje i Shapiro-Wilkův test normality, jehož p-hodnoty jsou v tabulce 5.3. Na hladině významnosti $\alpha = 0.01$ pro každý ticker zamítáme nulovou hypotézu, že rezidua jsou normálně rozdělená.

I přes tento fakt je model dle vzorce 5.1 vhodný, přestože vyjadřuje domněnku, že trh zůstane neměnný, což zřejmě není pravda. Za předpokladu, že platí teorie efektivního trhu (EMH), pochází veškeré impulzy pro změnu ceny z vnějších vlivů a ne z trhu samotného a navíc tyto změny znají všichni účastníci na trhu. Z pohledu ceny podkladového aktiva je takovýto model nejlepší možný, protože pouze na základě historického vývoje ceny podkladového aktiva nemůžeme predikovat její budoucí vývoj.

Pro obchodní systémy existuje ustálený benchmark - strategie Buy and hold. V této strategii na začátku koupíme předem dané množství podkladového aktiva a držíme jej po dobu celého testovacího období. Celková hodnota portfolia tedy koreluje s cenou aktiva samotného. Vývoj hodnoty této strategie je pro vybrané tickery znázorněn na obrázku 5.2.



Obrázek 5.1: Grafy distribucí chyb benchmarku prediktivních modelů: Na každém řádku jsou dva grafy pro vybrané aktivum. Vlevo histogram reziduí. Zeleně je vyznačen histogram a červenou čárkovanou čárou odpovídající normální rozdělení. Vpravo Q-Q graf, kde na ose x je kvantil normálního rozdělení a na ose y kvantil reziduí.



Obrázek 5.2: Vývoj hodnoty buy and hold strategie: Osa x odpovídá datu od začátku roku 2008 do konce roku 2014. Na ose y je hodnota portfolia buy and hold strategie při nákupu 1000 lotů podkladového aktiva.

atribut	hodnota
velikost populace	50
počet generací	400
pravděpodobnost křížení	0·6
pravděpodobnost mutace	0·02
počet opakování algoritmu	5

Tabulka 5.4: Parametry genetického algoritmu pro tvorbu prediktivních modelů

5.2 Prediktivní modely pomocí jednoduchého genetického programování

V tomto experimentu bylo použito jednoduché genetické programování pro tvorbu prediktivních modelů.

5.2.1 Algoritmus

Algoritmus probíhal ve stejném schématu jako jednoduchý genetický algoritmus. Pouze byla upravena reprezentace jedince a genetické operátory.

Parametry algoritmu jsou popsány v tabulce 5.4.

5.2.2 Fitness funkce

Jako fitness funkce byla zvolena MSE, protože patří mezi nejběžnější metodu měření chyby ať už v oblasti časových řad či jinde. Jedná se tedy o minimalizační úlohu.

5.2.3 Historická data

Experiment probíhal na denních cenách tickerů Apple Inc. (AAPL) Pepsico, Inc. (PEP), The Coca-Cola Company (KO) a S&P 500 (^GSPC). Trénovací data byla v rozmezí od 1. ledna 2008 do 31. prosince 2013. Validační potom od 1. ledna 2014 do 31. prosince 2014.

5.2.4 Jedinec

Kódování jedince bylo v podobě netypovaného stromu. Každý model byl k-arní funkce, pro $k \in \{13, 30, 50\}$, tedy

$$\hat{x}_{t,t+1} = f(x_{t-k}, x_{t-k+1}, \ldots, x_t) \tag{5.2}$$

$$
\begin{aligned}
f_1(x,y) &= x + y & f_7(x) &= \cos(x) \\
f_2(x,y) &= x - y & f_8(x) &= \sin(x) \\
f_3(x,y) &= x * y & f_9(a,b,x,y) &= \begin{cases} a & \text{if } a < b \\ b & \text{else} \end{cases} \\
f_4(x,y) &= \begin{cases} 1 & \text{if } |x| < 10^{-5} \\ x/y & \text{else} \end{cases} & f_{10}(a,b,x,y) &= \begin{cases} a & \text{if } a \leq b \\ b & \text{else} \end{cases} \\
f_5(x) &= -x \\
f_6(x) &= \log(|x| + 1)
\end{aligned}
$$

Tabulka 5.5: Množina funkcí prediktivních modelů

$$
c_1 = -1 \qquad c_2 = 10 \qquad c_3 = 2 \qquad c_4 = e \qquad c_5 = \pi
$$

Tabulka 5.6: Množina terminálů prediktivních modelů

5.2.5 Množina primitiv

Množina funkcí je zapsána v tabulce 5.5. Množina terminálů potom v 5.6.

Celkem tedy 10 funkcí, z toho 4 binární, 4 unární a 2 4-ární. Déle 5 předdefinovaných konstant a jednu konstantu pro každý parametr modelu, který odpovídá $x_t, x_{t-1}, \ldots, x_{t-k}$ pro k-ární model.

Funkce *if* nebyla do sady primitiv zahrnuta z důvodu fragmentace chování modelu na různých vzorcích, což by mohlo mít za následek, že na některém fragmentu se bude model chovat nepředvídatelně, protože v trénovací množině nebyl dostatečně pokryt testovacími daty.

5.2.6 Genetické operátory

- **Inicializace**: Jako operátor inicializace jedinců byl vybrán *ramped half-and-halt* generující stromy s hloubkou v rozmezí od 1 do 2.
- **Křížení**: Bylo zvoleno jednobodové křížení bez modifikace navržené Kozou.
- **Mutace**: Pro mutaci jedinců byla zvolena *subtree* mutace, kde nové podstromy byly generovány metodou *full* s hloubkou od 2 do 4.
- **Selekce**: Jakožto operátor selekce byla vybrána turnajová selekce s turnajem velikosti 3. Tato selekce byla upřednostněna před ruletovou selekcí, protože dobře funguje i v případě, že hodnoty fitness funkce jsou řádově větší než rozdíly mezi hodnotami fitness funkce dvou jedinců.

Abychom zabránili nadměrnému růstu stromu, omezili jsme maximální hloubku stromu na 17. Operace probíhala tak, jak ji navrhl Koza [29]. V případě, že nějaký z operátorů vygeneroval jedince porušující tuto podmínku, náhodně byl vybrán jako výsledek operace jeden z rodičů.

5.2.7 Detaily

Po každé generaci probíhalo kompletní ukládání stavu včetně stavu generátoru náhodných čísel, takže v případě neočekávaného přerušení experimentu šlo navázat na mezivýsledky a neovlivnilo to nijak běh algoritmu.

Všechny parametry byly experimentálně odladěny a vybrány takové, které ukazovaly nejvetší potenciál. Počet generací je však omezen i přes to, že zvýšení by vedlo k lepším výsledkům, jak je naznačeno ve výsledcích. To omezení je zavedeno z důvodu velké časové náročnosti výpočtu, která byla ve stovkách procesorových hodin.

5.3 Obchodní systém založený na NSGA-II a indikátoru SMA

V tomto experimentu byl implementován obchodní systém založený na indikátoru SMA, obchodující nad minutovou časovou řadou.

5.3.1 Fitness funkce

Fitness funkce byla implementována dle NSGA-II algoritmu. Vektor kritérii sestával z pěti kritérií: hodnota portfolia, Jensenova alfa, Sharpeho poměr, Sortinův poměr a velikost jedince měřeno v uzlech stromu. Dle všech kritérií se maximalozovalo až na poslední, který byl minimalizační.

5.3.2 Algoritmus

Protože vícekriteriální optimalizace je obtížnější než je tomu v případě optimalizace jednokriteriální, bylo použito dobře známého algoritmu NSGA-II.

Parametry algoritmu jsou v tabulce 5.7

5.3.3 Historická data

Testování probíhalo nad historickými daty symbolu AAPL v době od 3. února 2015 do 17. února 2015.

atribut	hodnota
velikost populace	40
počet generací	100
pravděpodobnost křížení	0·6
pravděpodobnost mutace	0·05
počet opakování algoritmu	1

Tabulka 5.7: Parametry genetického algoritmu pro tvorbu obchodních systémů

5.3.4 Jedinec

Jedinec byl zakódován jako funkce pěti proměnných - open, high, low, close a volume v daném časovém úseku. Výstup byl ve formě burzovního příkazu s omezením na počet držených lotů -100, 0 a +100.

5.3.5 Množina primitiv

Množina primitiv obsahovala primitiva typů *flaot*, *bool*, *Order* a *Indicator*.

Aby se zjednodušila implementace genetických operátorů, bylo třeba zajistit, následující podmínky

- pro každý typ můsí existovat alespoň jeden terminál nebo nulární funkce
- pro každý typ musí existovat alespoň jedna funkce, která jej vrací

Abychom mohli zajistit tyto podmínky, bylo třeba do množiny primitiv přidat některé nadbytečné funkce, jako je f_{18}.

Protože velikost prohledávaného prostoru roste přibližně exponenciálně s velikostí množiny primitiv, snažili jsme se ji zredukovat na co nejmenší velikost. Z toho důvodu je v množině primitiv například pouze funkce $<$ a nikoliv $\leq$, $>$ nebo $\geq$.

Z implementačních důvodů jsou příkazy *buy*, *sell*, *clear* a *hold* implementovány jako nulární funkce a ne jako konstanty.

Použitá množina funkcí je znázorněná v tabulce 5.8. Množina terminálů obsahovala konstanty 0.1, -0.1 a e pro typ *float*, *True* a *False* pro typ *bool* a konstanty SMA_{10}, SMA_{15}, SMA_{30} a SMA_{50} pro typ *Indicator*.

5.3.6 Genetické operátory

- **Inicializace**: Pro inicializaci byl použit operátor *ramped half-and-half* s minimální hloubkou stromu 1 a maximální hloubkou 4.
- **Křížení**: Jako operátor křížení bylo vybráno standardní jednobodové křížení bez modifikace navržené Kozou.

float(float)	$f_1(x) = \cos(x, y)$
	$f_2(x) = \sin(x, y)$
	$f_3(x) = e^x$
float(float, float)	$f_4(x, y) = x + y$
	$f_5(x, y) = x - y$
	$f_6(x, y) = x * y$
	$f_7(x, y) = \begin{cases} 1 & \text{if } \lvert x \rvert < 10^{-5} \\ x/y & \text{else} \end{cases}$
bool(float, float)	$f_8(x, y) = x < y$
float(bool, float, float)	$f_9(c, a, b) = \begin{cases} a & \text{if } c \\ b & \text{else} \end{cases}$
Order()	$f_{10}() = \text{buy}$
	$f_{11}() = \text{sell}$
	$f_{12}() = \text{clear}$
	$f_{13}() = \text{hold}$
Order(bool, Order, Order)	$f_{14}(c, a, b) = \begin{cases} a & \text{if } c \\ b & \text{else} \end{cases}$
float(Indicator)	$f_{15}(i) = \text{GetIndicatorValue}(i)$
bool(Indicator, Indicator)	$f_{16}(a, b) = \text{CrossesAbove}(a, b)$
	$f_{17}(a, b) = \text{CrossesBelow}(a, b)$
Indicator(Indicator)	$f_{18}(i) = \text{Identity}(i)$

Tabulka 5.8: Množina primitiv obchodních systémů

- **Mutace**: Operátor mutace byl stejný jako v případě prediktivních modelů, tedy *subtree* mutace, která generovala nové podstromy pomocí metody *full* s hloubkou v intervalu $[1, 4]$.
- **Selekce**: Jelikož byl v tomto experimentu použit algoritmus NSGA-II, který má definovanou vlastní mutaci, byla použita ta.

Navíc operátory křížení a mutace byly upraveny tak, aby generovaly stromy maximální hloubky 25 stejným způsobem jako u prediktivních modelů.

5.3.7 Detaily

Experiment byl naprogramován tak, aby podmínky simulace byly co možná nejrealističnější.

Byl zakomponován *slippage model*, který simuluje zpožděné exekuování příkazu, nebo dokonce jeho neexekuování při nedostatečné likviditě trhu, což se při skutečném obchodování stává. Model pracoval tak, že pokud požadované množství příkazem přesáhlo 1/4 množství zobchodovaného na skutečném trhu, transakce se neprovedla, protože pravděpodobně ani na skutečném trhu by nebylo dostatek likvidity. Zpožděná exekuce byla simulována posunutím ceny, za kterou se příkaz exekuoval. Velikost penalizace byla úměrná druhé mocnině procenta velikosti objemu požadovaného příkazu a skutečně zobchodovaného na trhu.

Poplatky za obchod byly obsaženy v modelu také. Celkově 0.03 USD za každý zobchodovaný lot.

5.4 Obchodní systém založený na NSGA-II a indikátoru CCI

Abychom byli schopni odhadnout přínosy indikátorů, implementovali jsme ještě stený algoritmus jako v 5.3 se dvěma rozdíly. Namísto indikátoru SMA byl použit CCI se stejnými periodami. To proto, abychom byli schopni posoudit jak velký rozdíl může být mezi jednotlivými indikátory. Další rozdíl je, že tomuto modelu nebylo povoleno držet jakékoliv otevřené pozice přes noc. K tomuto omezení jsme přistoupili, aby model nemohl snadno inkasovat zisky realizované přes noc, které často převyšují zisky z celého dne.

Aby byly oba modely alepoň částečně porovnatelné, zůstaly ostatní parametry nezměněny.

6. Experimentální výsledky a diskuse

V této kapitole prezentujeme výslekdy provedených experimentů a porovnáváme je mezi sebou. Každá z částí pojednává o algoritmu popsaném v stejnojmenné části kapitoly 5. Nakonec diskutujeme závěry.

6.1 Prediktivní modely pomocí jednoduchého genetického programování

Průběh fitness funkce po generacích je znározněn pro ticker AAPL na obrázku 6.1, pro PEP na 6.2, pro KO na 6.3 a pro ^GSPC na 6.4. Na obrázcích je vždy světle zelenou barvou a čárkovanou čárou vyznačena chyba modelu na validačních datech a plnou zelenou čárou jejich průměrná hodnota. Stejně tak je modrou barvou znázorněna chyba na trénovacích datech.

Z obrázků je zřejmé, že během žádného algoritmu nevzrůstala chyba na validačních datech jak tomu může být pro přeučené modely.

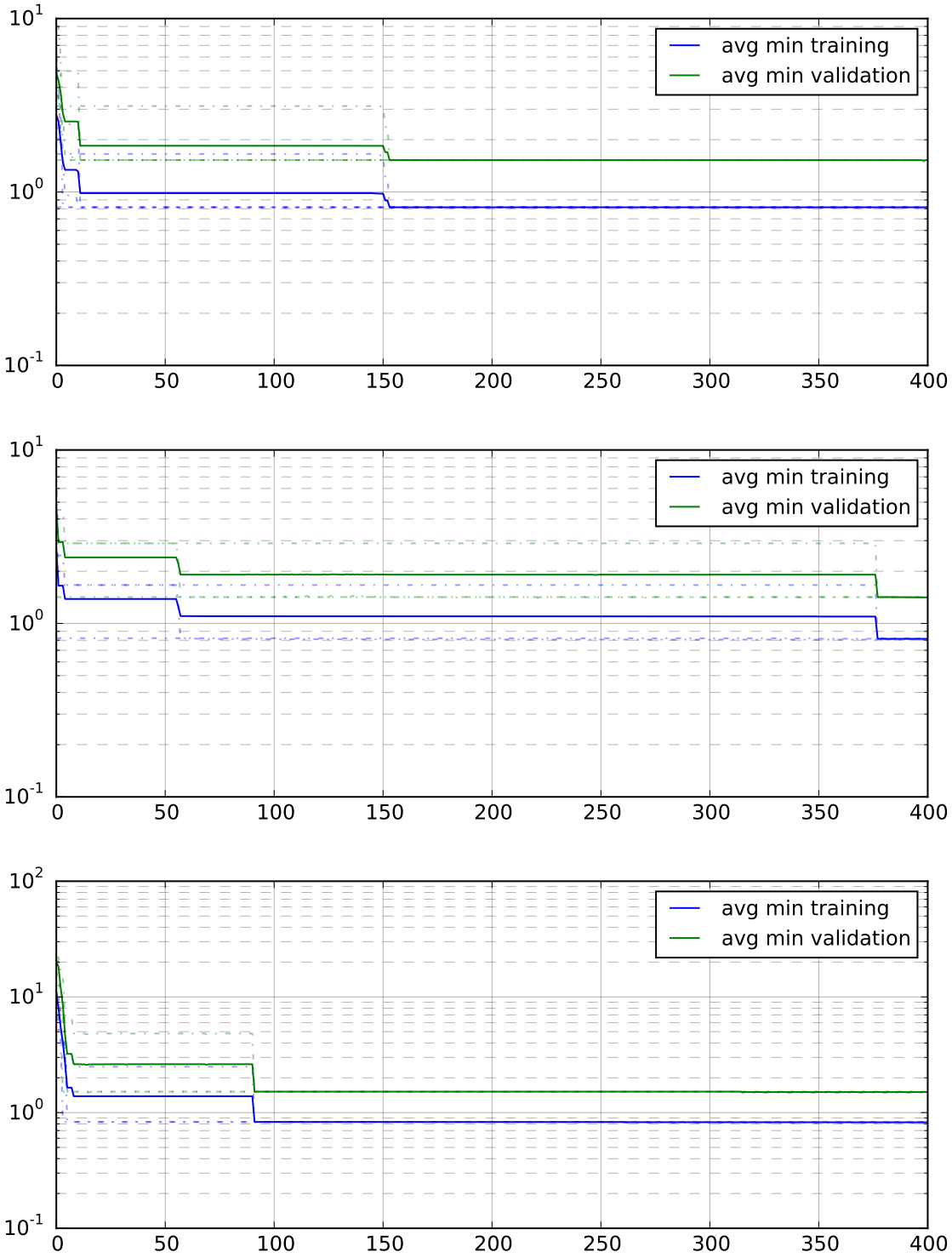
Obrázek 6.5 znázorňuje distribuci chyb jednotlivých modelů. Je vidět, že rezidua jsou přibližně normálně distribuované se střední chybou blízkou nule. Obrázek 6.6 potom ukazuje stejné informace, ale na validačních datech. Pro snadné porovnání je osa y ve stejném měřítku pro trénovací i validační data.

Z těchto grafů je také vidět, že na validačních datech je menší rozpětí extrémních hodnot. Tento fakt nelze přisuzovat kvalitě modelů, ale finanční krizi v roce 2008, během které docházelo ke značně velkým pohybům, navíc často v záporném směru, což nepozorujeme v období po jejím skončení.

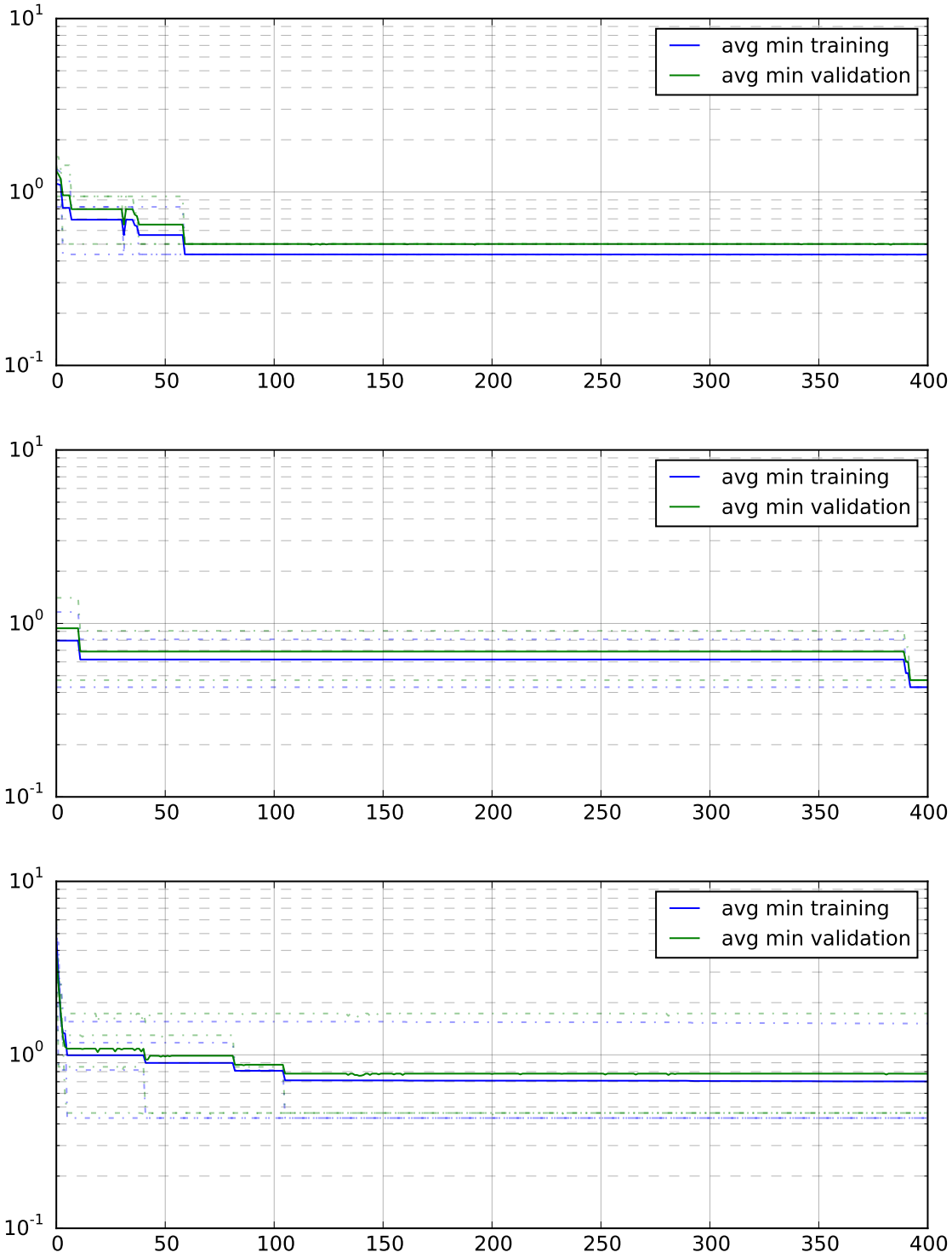
V tabulce 6.1 jsou znázorněny chyby modelů měřené různými metrikami na trénovacích a validačních datech. Kritérium BM (benchmark match) říká, pro kolik procent predikcí platilo, že byly od benchmarku vzdálené méně než 10^{-6}. Toto kritérium nevyjadřuje kvalitu modelu a má pouze informační charakter. Pro snadnější porovnání vůču benchmarku jsou potom v tabulce 6.2 znázorněné chyby jako procentuální změna vůči benchmarku. Pro kritéria MD a MPE jsou lepší čísla bližší nule, pro ostatní kriteria platí, že méně znamená lepší model.

Model PEP(30) se dle BM shodoval s benchmarkem ve 100% případů. Naopak model ^GSPC(15) se vždy lišil. Model ^GSPC(50) se lišil v $69,67\%$ případů. Všechny ostatní se shodovali s benchmarkem ve více než 99% případů a to jak na trénovacích tak validačních.

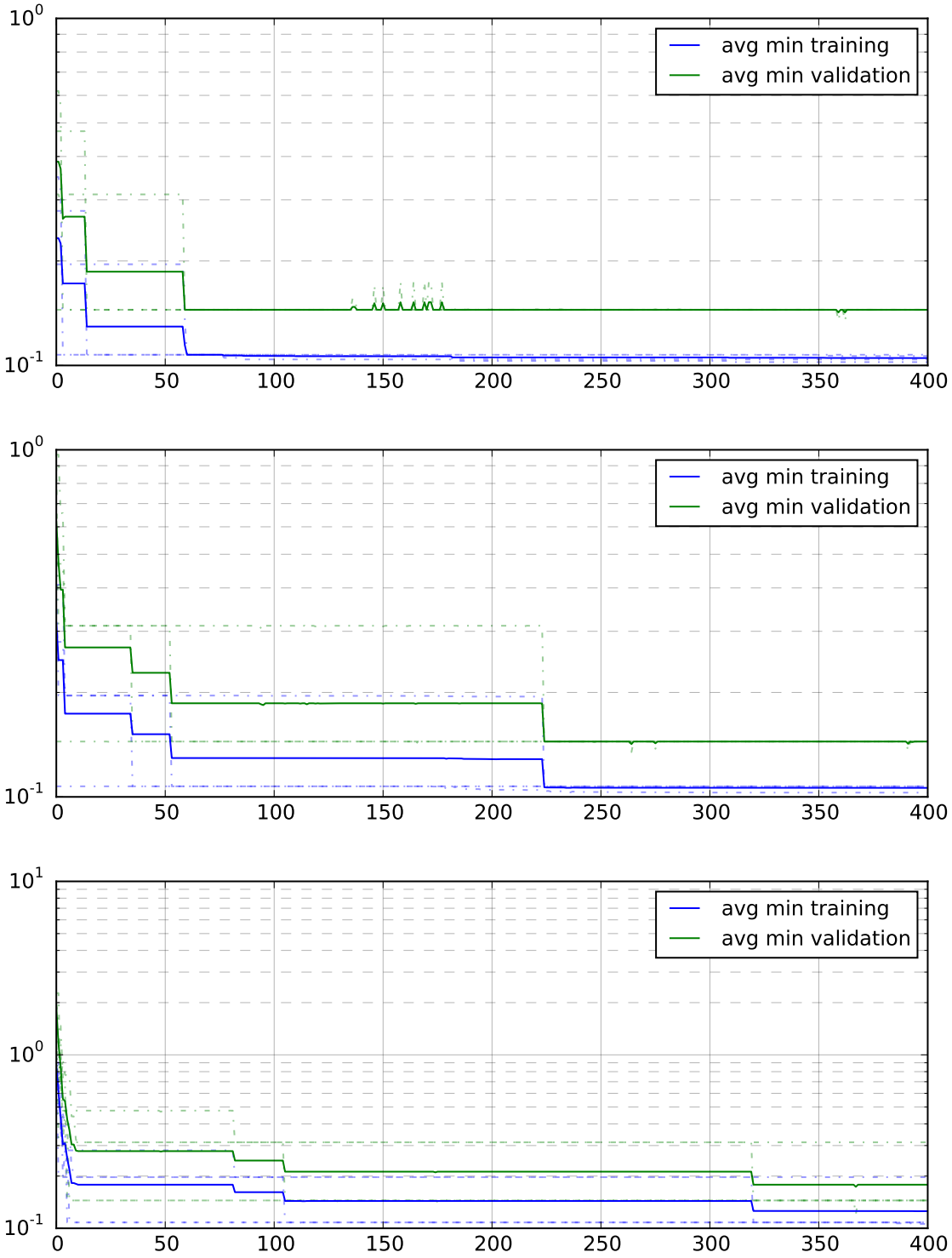
Na trénovacích datech je vždy vidět zlepšení vůči benchmarku v kritériu MSE,



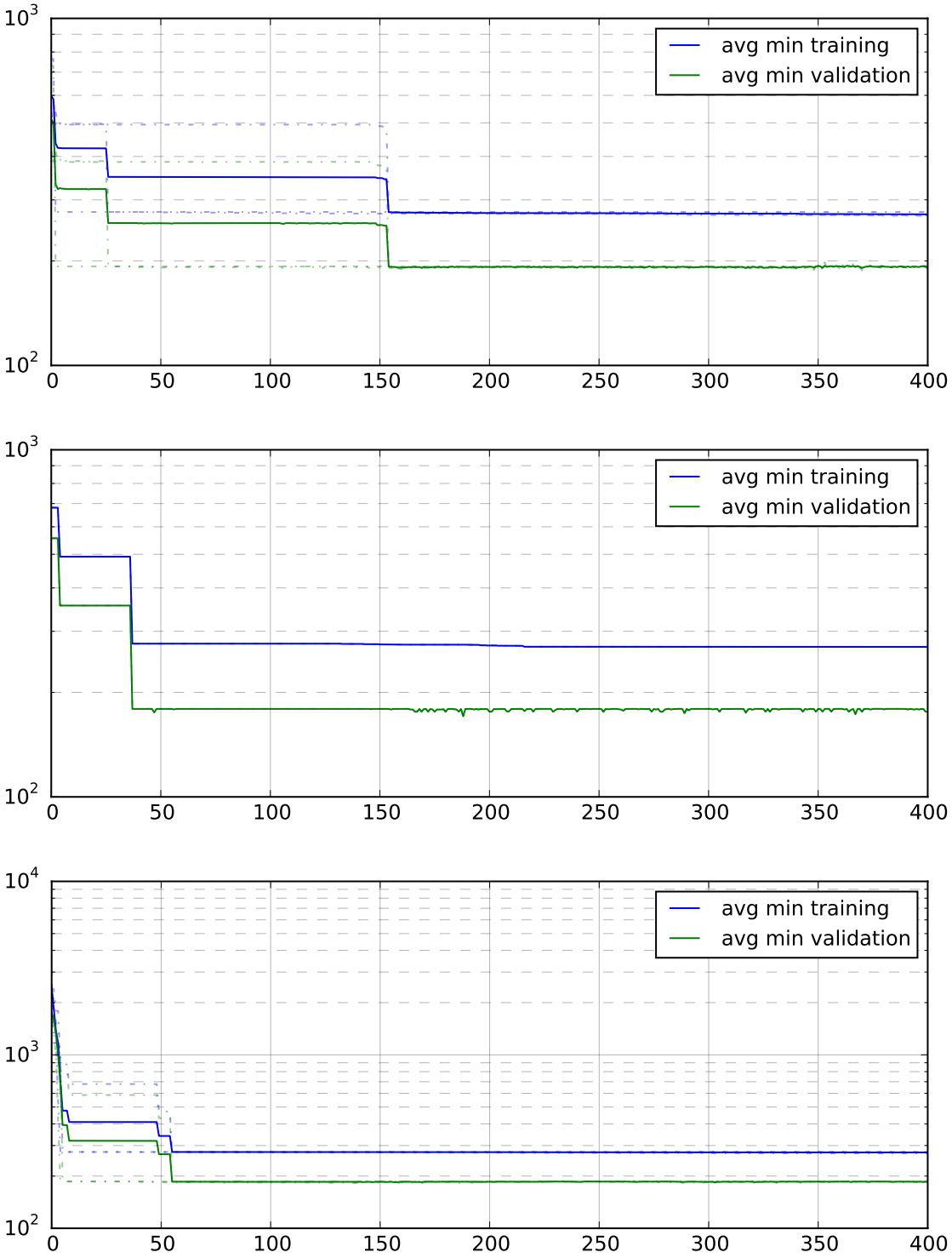
Obrázek 6.1: Průběh fitness funkce a validační chyby pro ticker AAPL Arita modelu odzhora 15, 30, 50



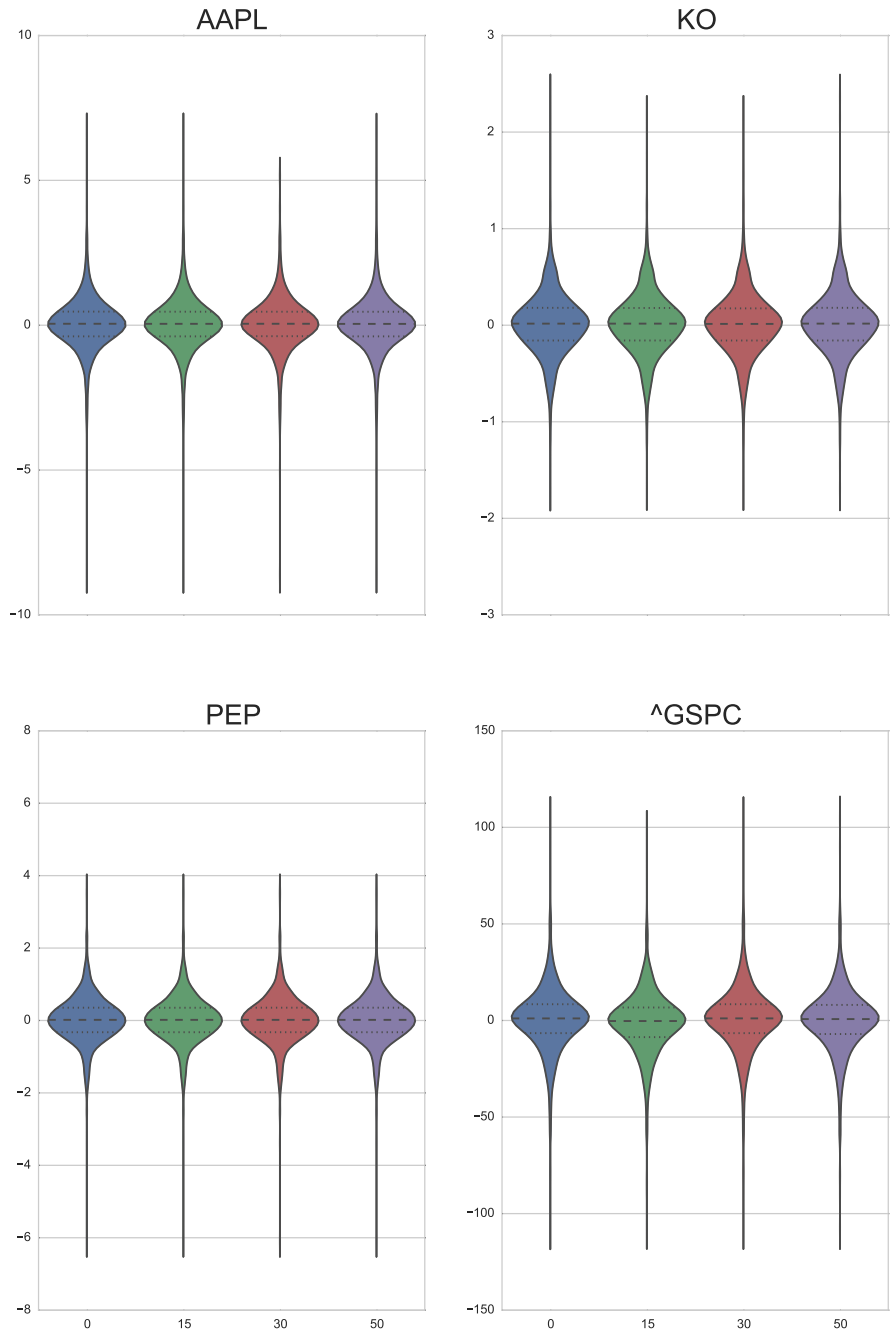
Obrázek 6.2: Průběh fitness funkce a validační chyby pro ticker PEP Arita modelu odzhora 15, 30, 50



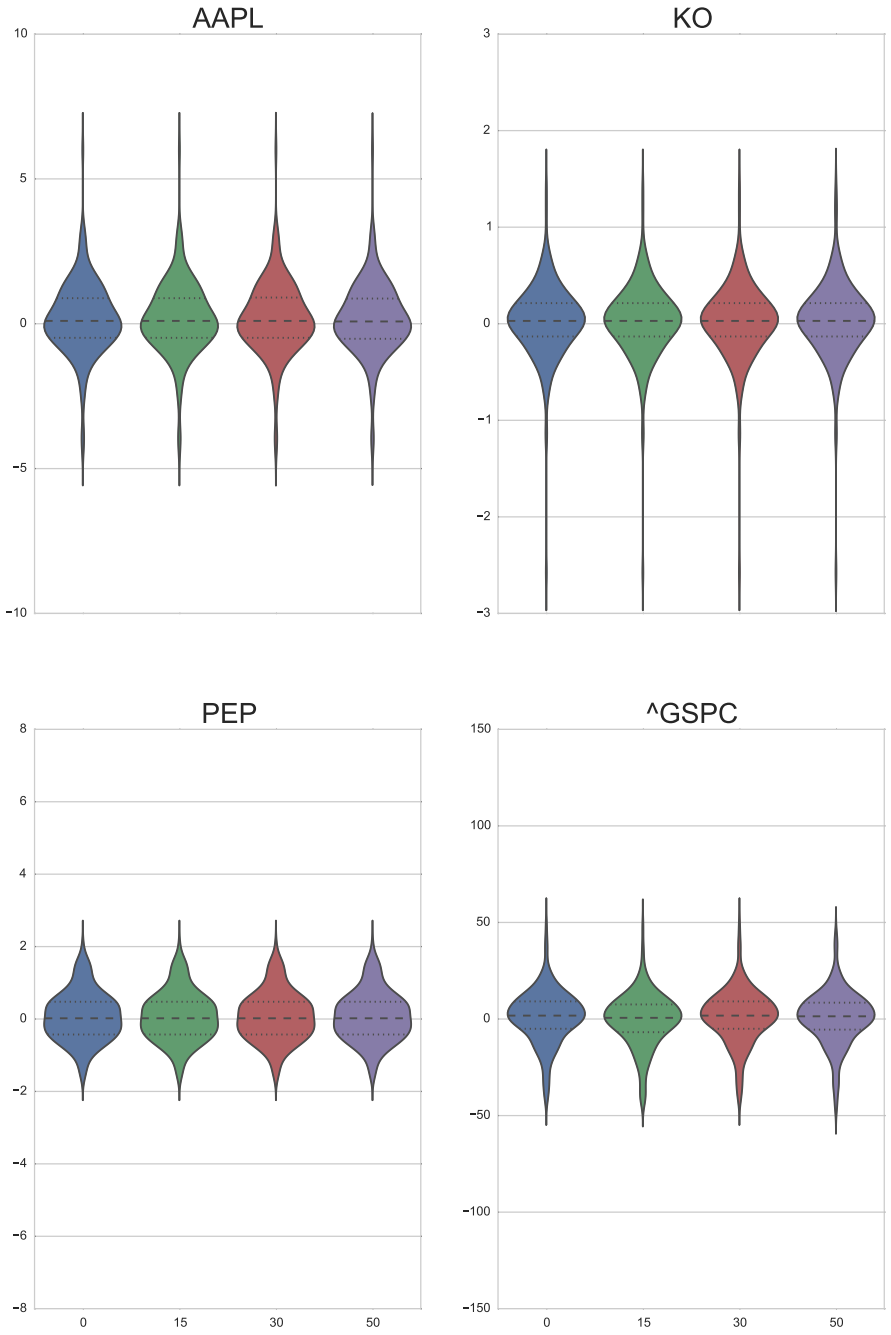
Obrázek 6.3: Průběh fitness funkce a validační chyby pro ticker KO Arita modelu odzhora 15, 30, 50



Obrázek 6.4: Průběh fitness funkce a validační chyby pro ticker ^GSPC
Arita modelu odzhora 15, 30, 50



Obrázek 6.5: Chyby jednotlivých modelů
Výraznou čárkovanou černou čarou je znázorněn průběr, slabši tečkovanou čárou směrodatná odchylka. Tvar odpovídá kernel density estimation (KDE). Indexem 0 je označený benchmark.



Obrázek 6.6: Chyby jednotlivých modelů na validačních datech
Indexem 0 je označený benchmark.

které bylo fitness funkcí. S výjimkou modelů ^GSPC(15) a ^GSPC(30) platí to samé i na datech validačních.

Navzdory pozorování 1 platí, že pro $\succeq$ vyjadřující relaci "je dle optimalizačního kritéria horší"je

$$PEP(30), PEP(50) \succeq PEP(30)$$

$$KO(50) \succeq KO(30) \succeq KO(15)$$

$$\hat{}GSPC(50) \succeq \hat{}GSPC(30) \succeq \hat{}GSPC(15)$$

To znamená, že řešení nalezená algoritmy PEP(30), PEP(50), KO(50), KO(30), ^GSPC(50) a ^GSPC(30) nejsou optimální. O optimalitě řešení ostatních modelů tvrzení nic nehovoří. Samotný fakt, že nalezená řešení nejsou optimální je vzhledem k velikosti prohledávaného prostoru očekávaný. Nasvědčuje to ale tomu, že pro tyto modely byl zvolen malý počet generací, protože modely větší arity nepřenaly modely arity menší.

Dalším významným faktem je, že dle kritérií MD a MPE jsou modely ^GSPC(15) a ^GSPC(50) v procentuální odchylce oproti benchmarku řádově horší. Pro první model to platí i na datech validačních. To je způsobeno tím, že jsou chyby předpovědí více vychýlené od nuly.

6.2 Obchodní systém založený na NSGA-II a indikátoru SMA

Průběh fitness funkce dle jednotlivých kritérií je znázorněn na obrázku 6.7. Na ose x jsou vyneseny generace. Osa y je pro každé kritérium ve vlastních jednotkách. Na každém grafu je znázorněna vždy minimální, průměrná a maximální hodnota kritéria v populaci.

Jensenova alfa, zřejmě nejdůležitější kritérium, jelikož odráží celkovou ziskovost jedince, dosáhla téměr maxima už v počáteční generaci a poté se již zlepšovala pouze málo. Sharpeho poměr se v průběhu dokonce nezlepšoval vubec. V případě Sortinova poměru tomu tak ale nebylo. Toto kritérium se zlepšovalo v průběhu celé evoluce. Kritérium velikost jedince je zde v roli Ockhamovi břitvi, protože předpokládáme, že jednodušší jedinci budou lépe generalizovat. Kritérium minimalizující počet obchodů je spíše informačního charakteru, protože simulace obsahuje transakční poplatny.

Kvůli vysoké dimenzionalitě nemůžeme aproximaci Pareto množiny zobrazit přímo. Abychom i přesto poskytli nějaký náhled na to jak vypadá, je tato fronta znázorněna na obrázku 6.8 vždy pro každou dvojici kritérií.

Trénovací data

ticker	arita	MD	MAD	MSE	RMSE	MPE	MAPE	BM
AAPL	0	0.0415	0.6184	0.8307	0.9114	0.0790%	1.5537%	100.00%
AAPL	15	0.0407	0.6178	0.8301	0.9111	0.0723%	1.5483%	99.86%
AAPL	30	0.0394	0.6146	0.8146	0.9026	0.0730%	1.5440%	99.29%
AAPL	50	0.0339	0.6118	0.8119	0.9011	0.0693%	1.5422%	99.50%
PEP	0	0.0163	0.4630	0.4323	0.6575	0.0165%	0.8277%	100.00%
PEP	15	0.0148	0.4615	0.4301	0.6558	0.0139%	0.8251%	99.79%
PEP	30	0.0163	0.4630	0.4323	0.6575	0.0165%	0.8277%	100.00%
PEP	50	0.0161	0.4630	0.4321	0.6574	0.0161%	0.8278%	99.93%
KO	0	0.0109	0.2353	0.1081	0.3288	0.0260%	0.8948%	100.00%
KO	15	0.0103	0.2329	0.1029	0.3207	0.0236%	0.8818%	99.93%
KO	30	0.0095	0.2325	0.1037	0.3220	0.0194%	0.8807%	99.01%
KO	50	0.0108	0.2343	0.1060	0.3255	0.0259%	0.8891%	99.79%
^GSPC	0	0.3647	11.4060	275.5161	16.5987	0.0113%	0.9868%	100.00%
^GSPC	15	-1.4555	11.2277	268.4873	16.3856	-0.1416%	0.9741%	0.00%
^GSPC	30	0.3920	11.3403	269.5958	16.4194	0.0141%	0.9802%	99.57%
^GSPC	50	-0.2011	11.3027	270.1355	16.4358	-0.0362%	0.9774%	69.67%

Validační data

ticker	arita	MD	MAD	MSE	RMSE	MPE	MAPE	BM
AAPL	0	0.0415	0.6184	0.8307	0.9114	0.0790%	1.5537%	100.00%
AAPL	15	0.0407	0.6178	0.8301	0.9111	0.0723%	1.5483%	99.86%
AAPL	30	0.0394	0.6146	0.8146	0.9026	0.0730%	1.5440%	99.29%
AAPL	50	0.0339	0.6118	0.8119	0.9011	0.0693%	1.5422%	99.50%
PEP	0	0.0163	0.4630	0.4323	0.6575	0.0165%	0.8277%	100.00%
PEP	15	0.0148	0.4615	0.4301	0.6558	0.0139%	0.8251%	99.79%
PEP	30	0.0163	0.4630	0.4323	0.6575	0.0165%	0.8277%	100.00%
PEP	50	0.0161	0.4630	0.4321	0.6574	0.0161%	0.8278%	99.93%
KO	0	0.0109	0.2353	0.1081	0.3288	0.0260%	0.8948%	100.00%
KO	15	0.0103	0.2329	0.1029	0.3207	0.0236%	0.8818%	99.93%
KO	30	0.0095	0.2325	0.1037	0.3220	0.0194%	0.8807%	99.01%
KO	50	0.0108	0.2343	0.1060	0.3255	0.0259%	0.8891%	99.79%
^GSPC	0	1.0781	10.0601	186.1216	13.6426	0.0529%	0.5168%	100.00%
^GSPC	15	-1.4555	11.2277	268.4873	16.3856	-0.1416%	0.9741%	0.00%
^GSPC	30	0.3920	11.3403	269.5958	16.4194	0.0141%	0.9802%	99.57%
^GSPC	50	0.5408	9.9296	184.7637	13.5928	0.0252%	0.5100%	77.63%

Tabulka 6.1: Chyby prediktivních modelů
Aritou 0 je označený benchmark.

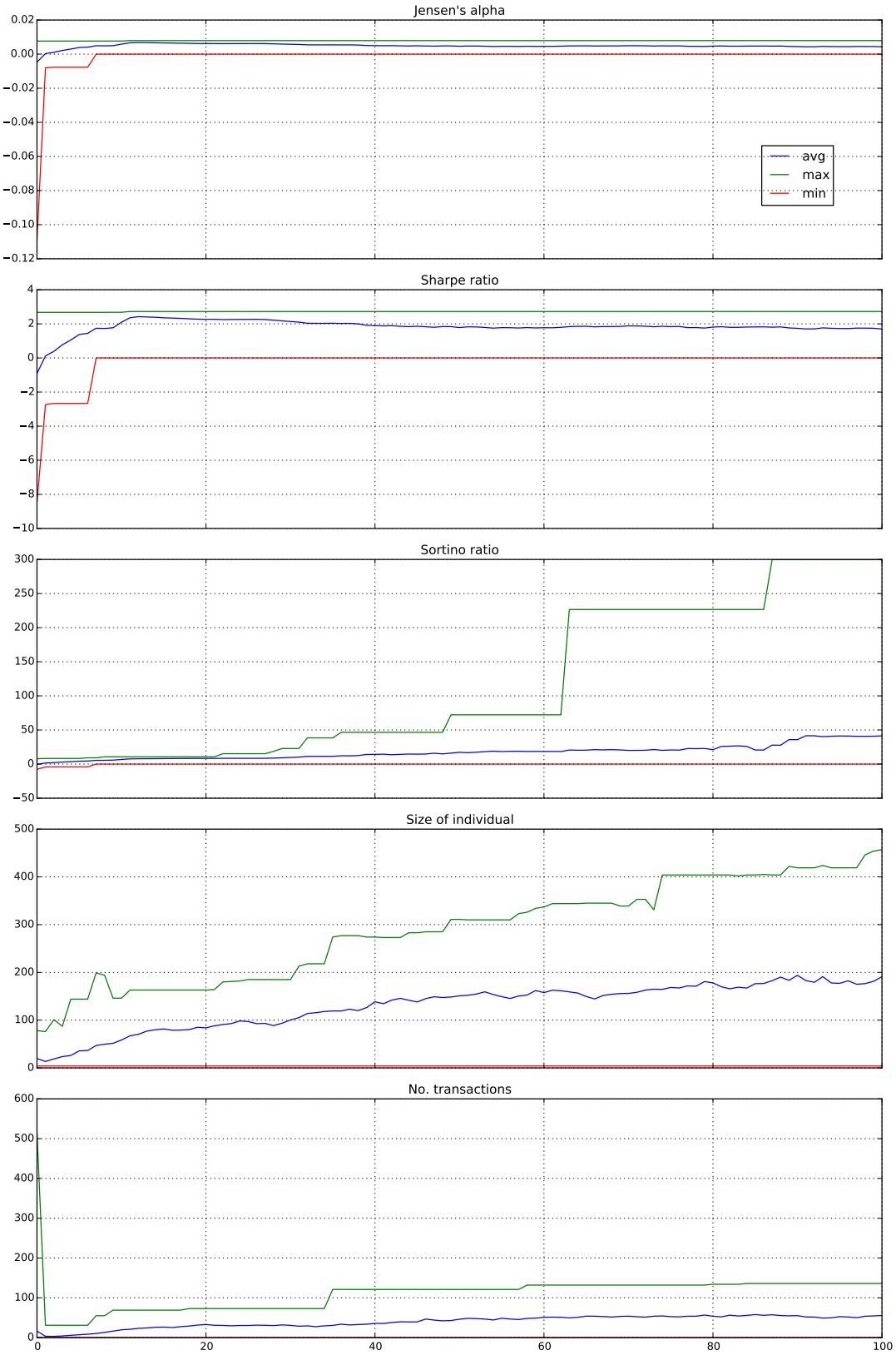
Trénovací data

ticker	arita	MD	MAD	MSE	RMSE	MPE	MAPE	BM
AAPL	15	-2.03%	-0.10%	-0.07%	-0.04%	-8.50%	-0.35%	-0.14%
AAPL	30	-5.11%	-0.61%	-1.93%	-0.97%	-7.54%	-0.63%	-0.71%
AAPL	50	-18.33%	-1.08%	-2.26%	-1.14%	-12.28%	-0.74%	-0.50%
KO	15	-5.04%	-1.04%	-4.86%	-2.46%	-9.10%	-1.45%	-0.07%
KO	30	-12.58%	-1.19%	-4.13%	-2.09%	-25.28%	-1.58%	-0.99%
KO	50	-0.52%	-0.45%	-1.98%	-1.00%	-0.14%	-0.64%	-0.21%
PEP	15	-8.91%	-0.31%	-0.51%	-0.26%	-15.83%	-0.32%	-0.21%
PEP	30	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
PEP	50	-0.94%	0.01%	-0.03%	-0.02%	-1.99%	0.01%	-0.07%
^GSPC	15	-499.06%	-1.56%	-2.55%	-1.28%	-1349.12%	-1.28%	-100.00%
^GSPC	30	7.46%	-0.58%	-2.15%	-1.08%	24.39%	-0.66%	-0.43%
^GSPC	50	-155.15%	-0.91%	-1.95%	-0.98%	-419.07%	-0.95%	-30.33%

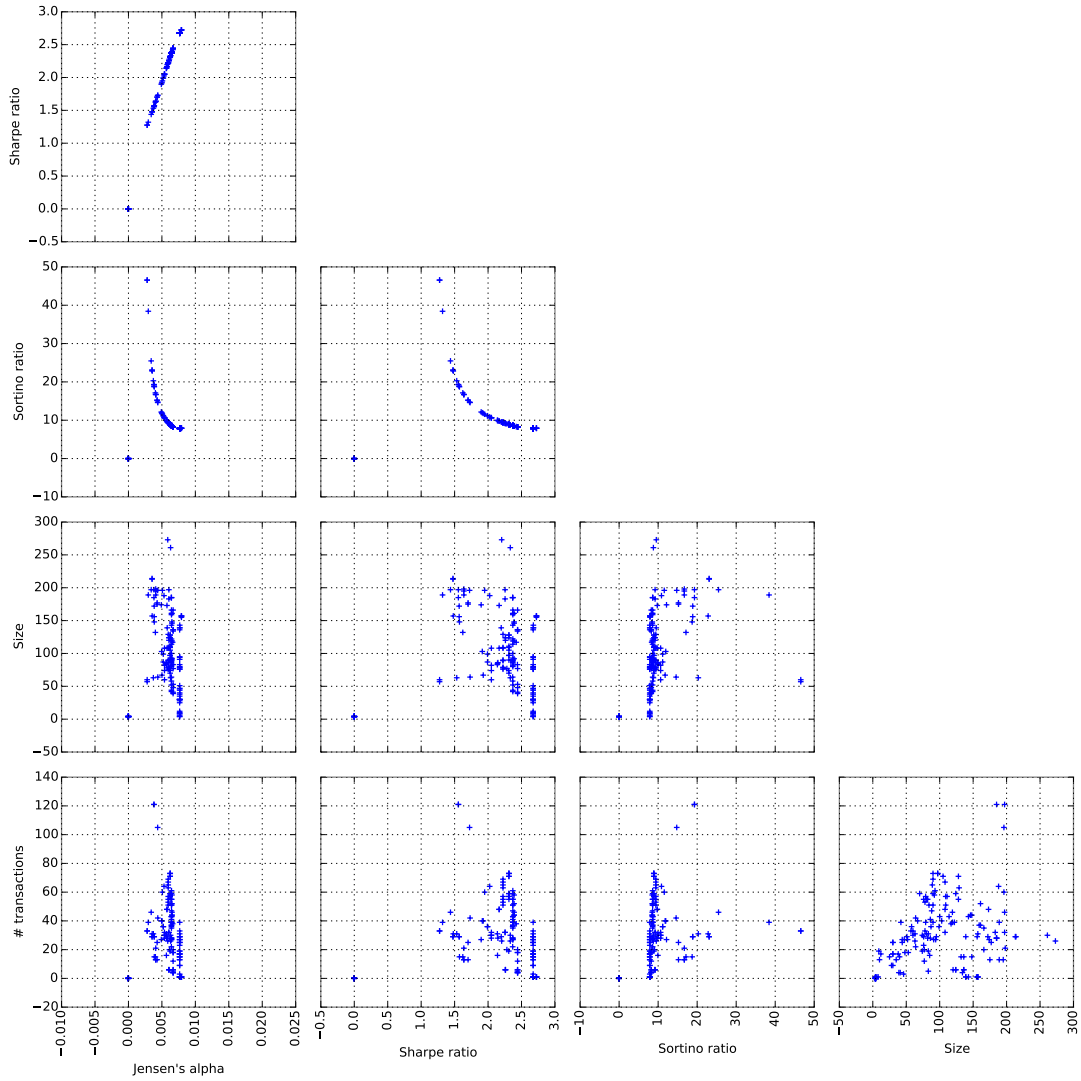
Validační data

ticker	arita	MD	MAD	MSE	RMSE	MPE	MAPE	BM
AAPL	15	-2.03%	-0.10%	-0.07%	-0.04%	-8.50%	-0.35%	-0.14%
AAPL	30	-5.11%	-0.61%	-1.93%	-0.97%	-7.54%	-0.63%	-0.71%
AAPL	50	-18.33%	-1.08%	-2.26%	-1.14%	-12.28%	-0.74%	-0.50%
KO	15	-5.04%	-1.04%	-4.86%	-2.46%	-9.10%	-1.45%	-0.07%
KO	30	-12.58%	-1.19%	-4.13%	-2.09%	-25.28%	-1.58%	-0.99%
KO	50	-0.52%	-0.45%	-1.98%	-1.00%	-0.14%	-0.64%	-0.21%
PEP	15	-8.91%	-0.31%	-0.51%	-0.26%	-15.83%	-0.32%	-0.21%
PEP	30	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
PEP	50	-0.94%	0.01%	-0.03%	-0.02%	-1.99%	0.01%	-0.07%
^GSPC	15	-235.01%	11.61%	44.25%	20.11%	-367.70%	88.47%	-100.00%
^GSPC	30	-63.64%	12.73%	44.85%	20.35%	-73.34%	89.66%	-0.43%
^GSPC	50	-49.84%	-1.30%	-0.73%	-0.37%	-52.34%	-1.32%	-22.37%

Tabulka 6.2: Chyby prediktivních modelů na - procento odchylky od benchmarku



Obrázek 6.7: Průběh fitness funkce dle jednotlivých kritérií pro obchodní systém založený na NSGA-II a indikátoru SMA



Obrázek 6.8: Porovnání kritérií fitness funkce po dvojicích pro jedince z aproximace Pareto množiny pro obchodní systém založený na NSGA-II a indikátoru SMA

Hodnoty fitness funkcí na validačních datech nejsou zobrazeny v průběhu evoluce, kvůli časové náročnosti na jejich spočítání. Pro jednotlivé jedince byla provedena analýza na trénovacích i validačních datech. Příklad takové analýzy je pro jedince s nejvýšší hodnotou Jensenovy alfy v průběhu celé evoluce znázorněn na obrázku 6.9. Na ose x je čas. Červenou svislou čarou je znázorněn předěl mezi trénovacími a validačními daty. Na grafu nahoře, je modrou čarou znázorněn průběh ceny podkladového aktiva - tickeru AAPL, této čáře patří levá osa y. Pravá patří poloprůhledné modré čáře, která znázorňuje zobchodovaný objem. Na grafu uprostřed je hodnota portfolia a v tabulce je vypsána fitness funkce. Na spodním grafu je modrou čarou, které patří levá osa y znárorněna hodnota portfolia pro benchmark. Světlou šedou čarou, které patří osa y napravo, je znázorněn zisk strategie oproti benchmarku.

Z tohoto obrázku je vidět, že strategie v průběhu prvního obchodního dne udělala několik obchodů a na konci zůstala v otevřené pozici až do konce simulace. Stejné chování poté pokračuje i na validačních datech.

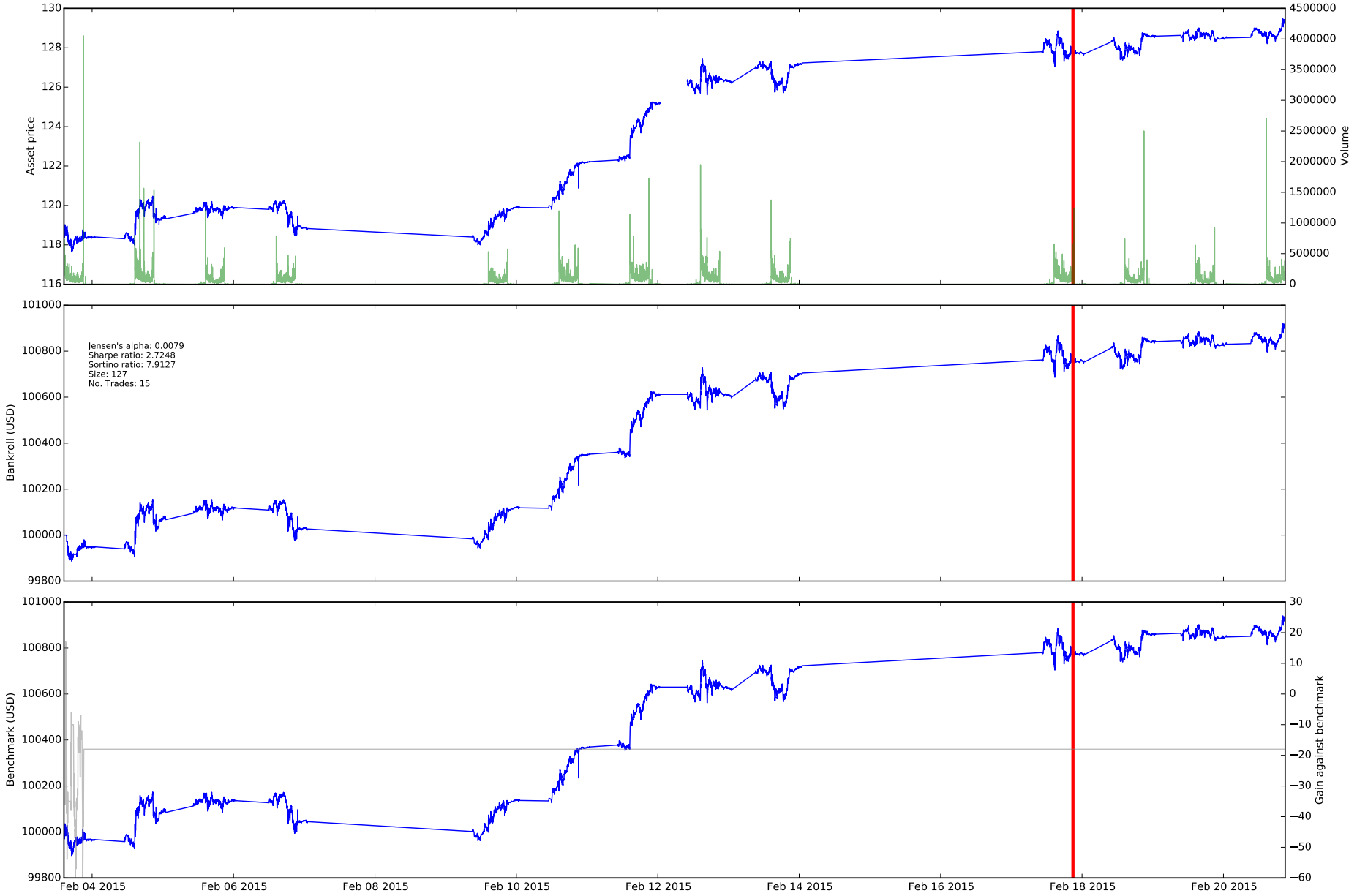
Takovéto chování se vyvinulo v celé aproximaci Pareto množiny a to včetně validačních dat. Z toho můžeme usuzovat, že jedinci nebyli dostatečně rozvinutí na to, aby obsahovali nějaké sofistikovanější chování. Inkasovali tedy pouze snadno dosažitelné zisky. Nic ale nenasvědčuje tomu, že by se takovéto chování po dostatečně dlouhé době vyvinout nemohlo.

Tento jedinec je vykreslený na obrázku 6.10. Na první pohled obsahuje mnoho zbytečných uzlů, které lze odstranit. Pseudokód tohoto jedince je v algoritmu 3. Jedinec se vyvinul do podoby, která je podobná obchodním systémům, které používají lidé obchodující na finančních trzích. Lze proto předpokládat, že i když tento konkrétní jedinec se dostatečně nevyvinul, má potenciál dosáhnout výsledků systematicky převyšujících benchmark.

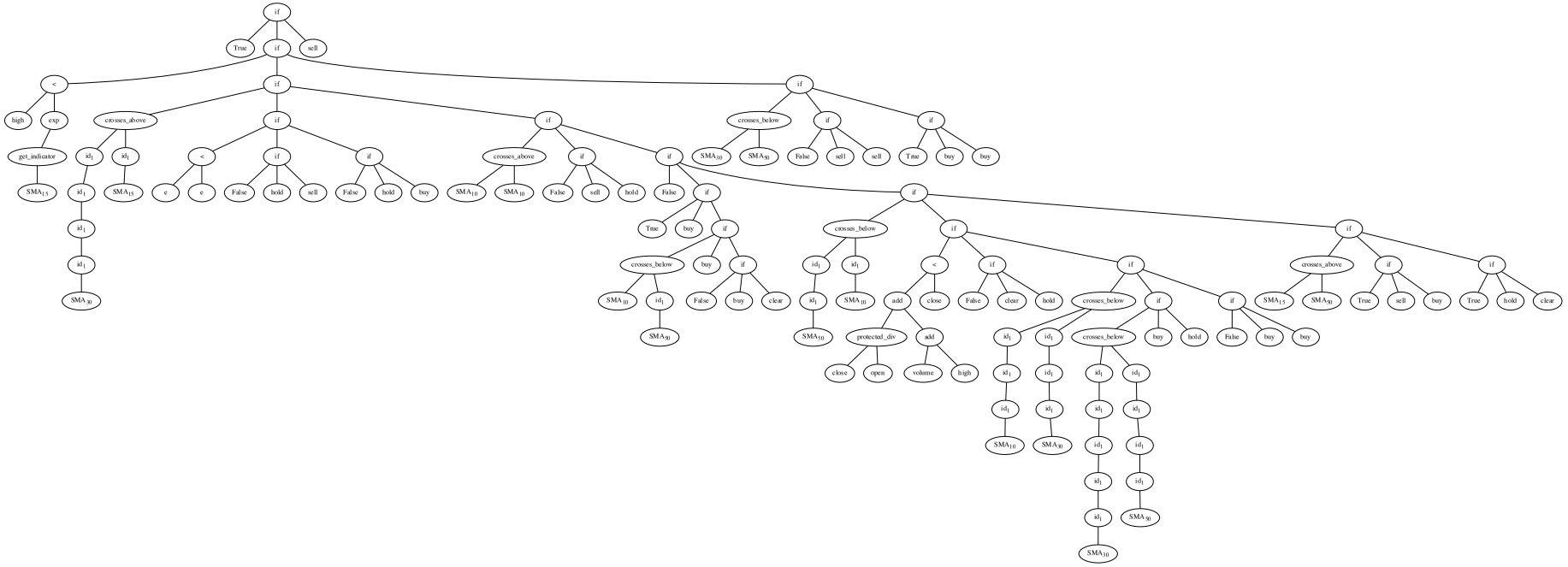
6.3 Obchodní systém založený na NSGA-II a indikátoru CCI

Na obrázku 6.11 je znázorněn průběh jednotlivých složek fitness funkce stejně tak, jako tomu bylo u předešlého modelu. Můžeme zde pozorovat jednu podstatnou změnu a to růst kritérií Jensenova alfa a Sharpeho poměr v průběhu celé evoluce. Nutno podotknout, že obě hodnoty jsou stále nižší než tomu bylo u modelu s indikátorem SMA, to ale může být částečně následkem pouze denního obchodování.

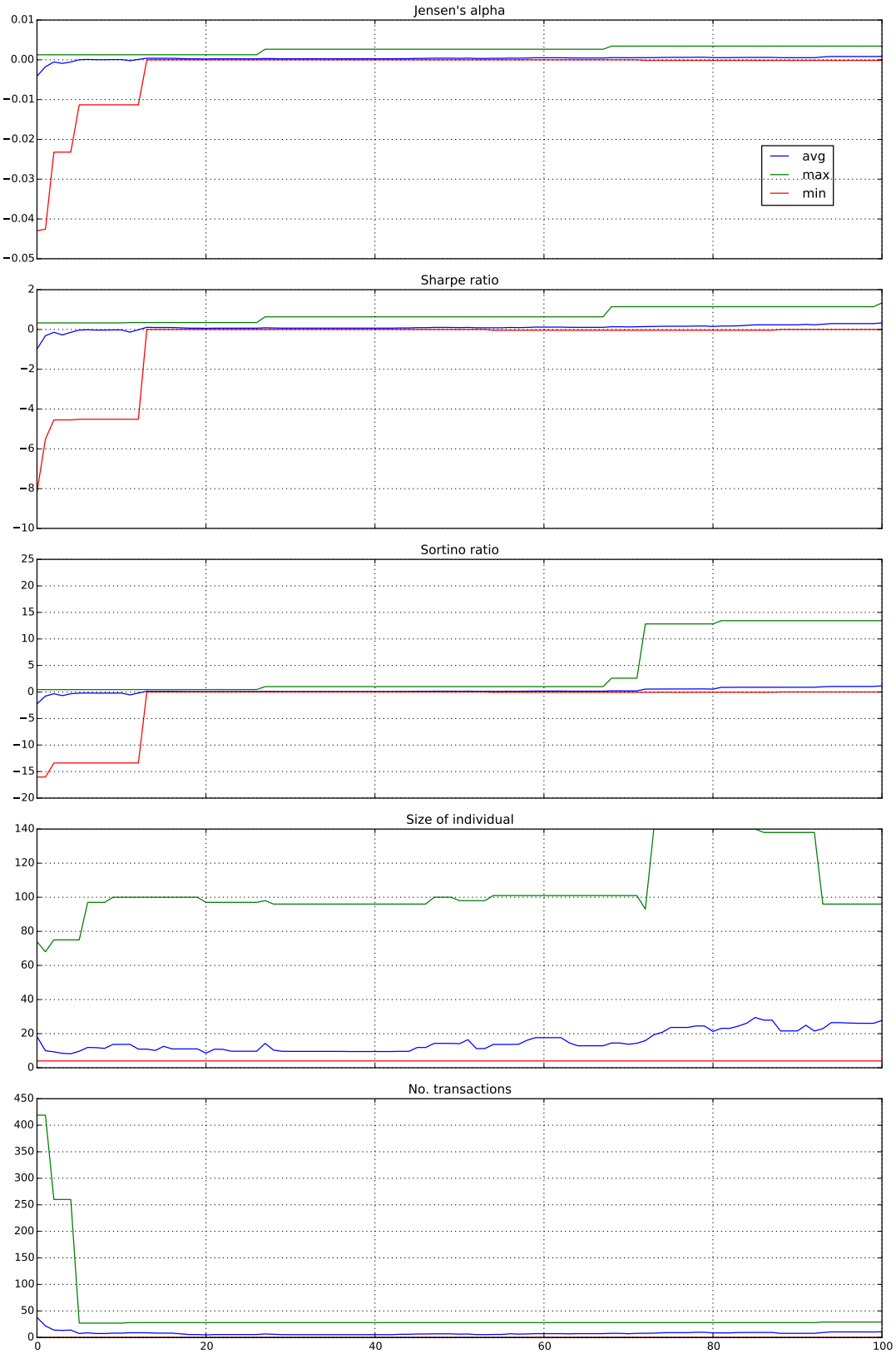
Obrázem 6.12 ukazuje podobu jednotlivých složek fitness funkce. Na tomto obrázku stojí za znínku zejména vztah mezi velikostí jedince a Sharpeho pomě-



Obrázek 6.9: Výkon nejlepšího jedince dle Jensenovy alfy pro obchodní systém založený na NSGA-II a indikátoru SMA



Obrázek 6.10: Grafické znázornění nejlepšího jedince obchodního systému s indikátorem SMA



Obrázek 6.11: Průběh fitness funkce dle jednotlivých kritérií pro obchodní systém založený na NSGA-II a indikátoru CCI

Algoritmus 3 Vybraný jedinec obchodního systému s indikáterem SMA po zjednodušení

```
1: procedure INDIVIDUAL(open, high, low, close, volume)
2:     if crosses_below(SMA15, SMA30) then
3:         return buy
4:     else
5:         if crosses_above(SMA10, SMA50) then
6:             return hold
7:         else
8:             if crosses_above(SMA15, SMA50) then
9:                 return sell
10:            else
11:                return hold
12:            end if
13:        end if
14:    end if
15: end procedure
```

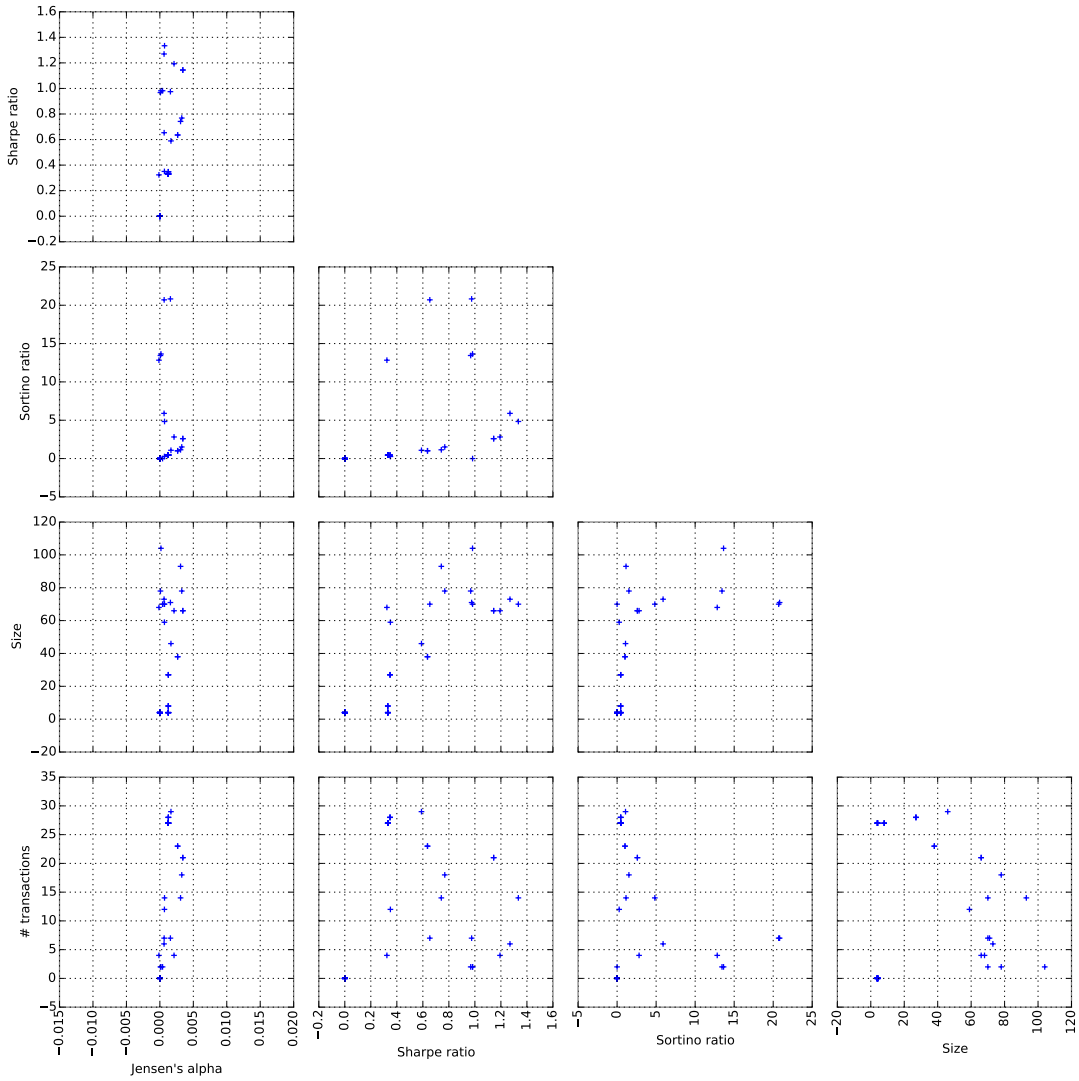
rem, který se zdá být lineární. To lze vysvětlit alspoň dvěma způsoby. Buď to může být známka přeučení, protože složitější jedinci jsou k tomuto problému náchylnější, nebo to můžeme vysvětlit tak, že aby byla strategie zisková, musí nutně být dostatečně komplexní. Jak ale uvádíme dále, jedinci pravděpodobně nebyli přeučení. Spíše tuto závislost tedy vysvětlíme druhým způsobem. Takováto závislost u modelu s indikátorem SMA nenastala.

Výkonnost jedince s nejlepší hodnotou Jensenovy alfy je znázorněna obrázkem 6.13. Tento systém inkasoval menší zisky oproti benchmarku a není patrná žádná známka sofistikovanějšího chování. Strategie však obchodovala narozníl od předchozí v průběhu celého testovacího období včetně validačních dat.

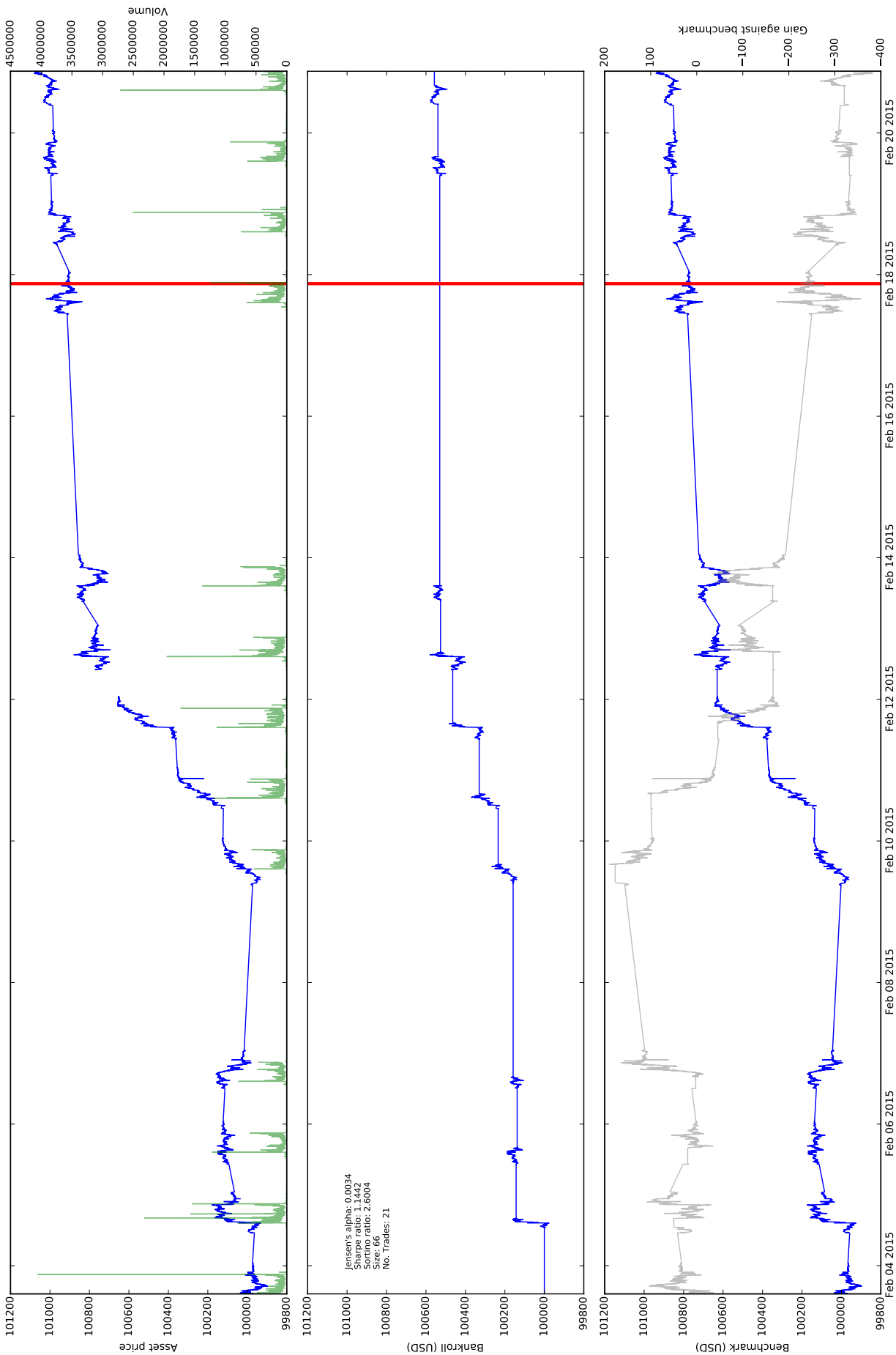
Konkrétní podoba tohoto jedince vykreslena na obrázku 6.14.

6.4 Diskuse

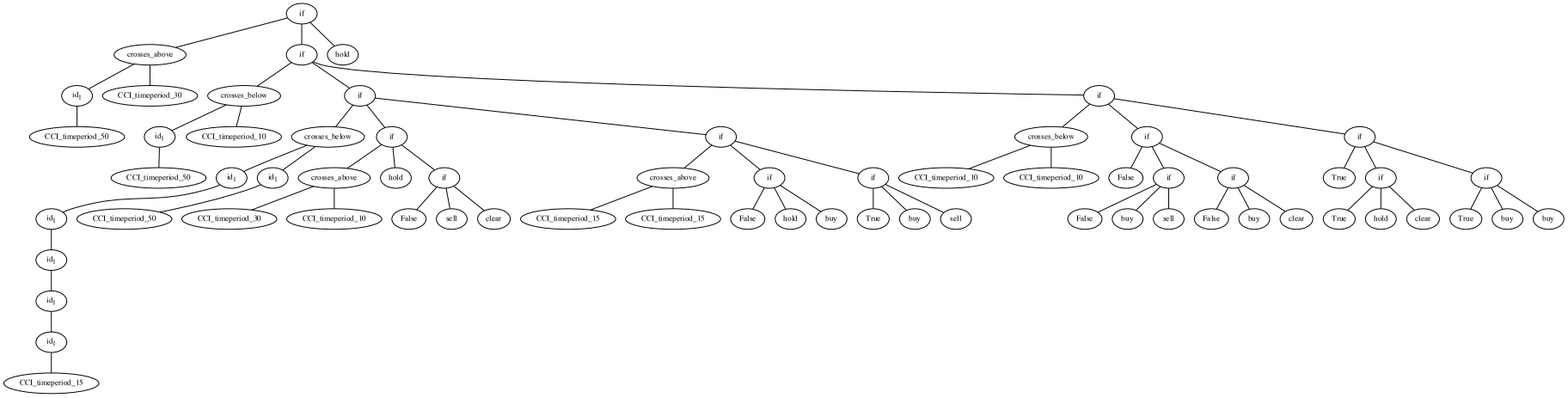
Na základě zjištěných faktů lze říci, že prediktivní modely nalezené pomocí použitých algoritmů jsou ve většině případů lepší než benchmark. Zlepšení je však velmi malé, ale po širším testování by zřejmě na této metodě šel postavit ziskový obchodní systém. Očekávaný zisk by však byl malý a dost možná by nepřekonal inflaci spolu s náklady na tvorbu takového modelu. Z těchto důvodů nelze doporučit genetické programování jako prostředek pro tvorbu ziskových obchodních systémů založených na prediktivních modelech. Nutno ale podotknout, že neschopnost najít dostatečně dobré řešení nemusí být způsobena nevhodností GP pro řešení této úlohy, ale obtížností (a dle teorie efektivního trhu (EMH) neřešitelností) úlohy samotné.



Obrázek 6.12: Porovnání kritérií fitness funkce po dvojicích pro jedince z aproximace Pareto množiny pro obchodní systém založený na NSGA-II a indikátoru CCI



Obrázek 6.13: Výkon nejlepšího jedince dle Jensenovy alfy pro obchodní systém založený na NSGA-II a indikátoru CCI



Obrázek 6.14: Grafické znázornění nejlepšího jedince obchodního systému s indikátorem CCI

I přes fakt, že žádný obchodní systém, se nevyvinul do podoby, kdy bychom ho mohli nechat samostatně obchodovat na finančních trzích, ukázal tato přístup podstatně větší potenciál, který je možné dále rozvíjet.

V průběhu celé práce se nám nepodařilo nalázt žádný model, který by překonával zvolený benchmark.

7. Závěr

V rámci této práce jsme navrhli dva možné přístupy k aplikaci genetického programování na oblast finančních trhů a pro každý z nich jsme uvedli několik variant. Vybrné varianty jsme implementovali a otestovali na skutečných datech z dostupných finančních trhů. Na závěr práce jsme uváděli dosažené výsledky a diskutovali je mezi sebou.

Žádnému z implementovaných přístupů se nepodařilo nalézt model, který by překonával benchmark. Tento výsledek však s ohledem na výsledky předcházející výzkum není překvapující. Přístup pomocí obchodních systémů však vykazoval potenciál k překonání benchmarku.

Problémem všech implementovaných modelů byla velká časová náročnost, což mělo za následek nedostatečné rozvinutí dobrých vlastností, které se u některých modelů ukazovaly. Jako pokračování by proto bylo vhodné implementovat samotný testovací engine v nějakém nízkoúrovňovějším prostředí a více využít snadné paralelizovatelnosti úlohy například pomocí implementace na čipu Intel Xeon Phi, nebo distribuovaných výpočtů.

Podstatné zvýšení výpočetní síly by také umožnilo zvětšit velikost populace, aby se jedinci mohli lépe rozprostřít skrz prohledávaný prostor. V případě velkých populací by bylo možné se zaměřit i na vícepopulační evoluci, například pomocí ostrovního systému a migrování dostatečně silných jedinců mezi ostrovy, což by umožňovalo současný vývoj několika různých strategií.

Také by to umožňovalo do množiny primitiv přidat více složitějších primitiv, což by mohlo evoluci podstatně pomoci.

Seznam použité literatury

[1] URL: https://docs.jboss.org/drools/release/6.0.0.Beta2/optaplanner-docs/html/scoreCalculation.html#scoreTrap.

[2] URL: http://www.investopedia.com/terms/e/efficientmarkethypothesis.asp.

[3] URL: http://www.evostar.org/2015/cfp_evoiasp.php.

[4] URL: https://www.google.com/finance.

[5] URL: http://ipython.org.

[6] URL: http://matplotlib.org.

[7] URL: http://pygraphviz.github.io.

[8] URL: https://www.quandl.com.

[9] URL: https://www.quantopian.com.

[10] URL: http://finance.yahoo.com.

[11] URL: http://www.zipline.io.

[12] *2010 through Present NYSE Group Daily Share Volume*.

[13] *80-20 Rule*. URL: http://www.investopedia.com/terms/1/80-20-rule.asp.

[14] Mark P Austin et al. "Adaptive systems for foreign exchange trading". In: *Quantitative Finance* 4.4 (2004), s. 37–45.

[15] Kalyanmoy Deb et al. "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II". In: *Lecture notes in computer science* 1917 (2000), s. 849–858.

[16] MAH Dempster a CM Jones. "A real-time adaptive trading system using genetic programming". In: *Quantitative Finance* 1.4 (2001), s. 397–413.

[17] Michael Alan Howarth Dempster et al. "Computational learning techniques for intraday FX trading using popular technical indicators". In: *Neural Networks, IEEE Transactions on* 12.4 (2001), s. 744–754.

[18] Félix-Antoine Fortin et al. "DEAP: Evolutionary Algorithms Made Easy". In: *Journal of Machine Learning Research* 13 (čvc 2012), s. 2171–2175.

[19] R. J. Hampo a K. A. Marko. "Application of Genetic Programming to Control of Vehicle Systems". In: *Proceedings of the Intelligent Vehicles '92 Symposium*. Detroit, Mi, USA: IEEE, červ. 1992, s. 191–195. ISBN: 0-7803-0747-X. DOI: doi:10.1109/IVS.1992.252255.

[20] Simon Handley. "Automatic Learning of a Detector for alpha-helices in Protein Sequences Via Genetic Programming". In: *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*. Ed. Stephanie Forrest. University of Illinois at Urbana-Champaign: Morgan Kaufmann, 17-21 07 1993, s. 271–278. URL: http://www-leland.stanford.edu/~shandley/postscript/alpha-helices.ps.gz%20broken.

[21] Nikolaus Hansen et al. "Comparing Results of 31 Algorithms from the Black-box Optimization Benchmarking BBOB-2009". In: *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation*. GECCO '10. Portland, Oregon, USA: ACM, 2010, s. 1689–1696. ISBN: 978-1-4503-0073-5. DOI: 10.1145/1830761.1830790. URL: http://doi.acm.org/10.1145/1830761.1830790.

[22] Shu-Heng Chen a Chung-Chih Liao. "Agent-based computational modeling of the stock price-volume relation". In: *Information Sciences* 170.1 (18 02 2005), s. 75–100. DOI: doi:10.1016/j.ins.2003.03.026. URL: http://www.sciencedirect.com/science/article/B6V0C-4B3JHTS-6/2/9e023835b1c70f176d1903dd3a8b638e.

[23] Investopedia. *Mark to Market - MTM*. URL: http://www.investopedia.com/terms/m/marktomarket.asp.

[24] Michael C Jensen. "The performance of mutual funds in the period 1945–1964". In: *The Journal of finance* 23.2 (1968), s. 389–416.

[25] M. A. Kaboudan. "A Measure of Time Series' Predictability Using Genetic Programming Applied to Stock Returns". In: *Journal of Forecasting* 18.5 (zář. 1999), s. 345–357. ISSN: 1099-131X. DOI: doi:10.1002/(SICI)1099-131X(199909)18:5<345::AID-FOR744>3.0.CO;2-7.

[26] M. A. Kaboudan. "Genetic Programming Prediction of Stock Prices". In: *Computational Economics* 16.3 (pros. 2000), s. 207–236. ISSN: 0927-7099. DOI: doi:10.1023/A:1008768404046.

[27] Mak Kaboudan. "Extended daily exchange rates forecasts using wavelet temporal resolutions". In: *New Mathematics and Natural Computing* 1.1 (břez. 2005), s. 79–107. ISSN: 1793-0057. DOI: doi:10.1142/S1793005705000056.

[28] Con Keating a William F Shadwick. "A universal performance measure". In: *Journal of performance measurement* 6.3 (2002), s. 59–84.

[29] John R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992. ISBN: 0-262-11170-5. URL: http://mitpress.mit.edu/books/genetic-programming.

[30] John R. Koza a David Andre. "Classifying Protein Segments as Transmembrane Domains Using Architecture-Altering Operations in Genetic Programming". In: *Advances in Genetic Programming 2*. Ed. Peter J. Angeline a K. E. Kinnear, Jr. Cambridge, MA, USA: MIT Press, 1996. Kap. 8, s. 155–176. ISBN: 0-262-01158-1. URL: http://www.genetic-programming.com/jkpdf/aigp2aatmjk1996.pdf.

[31] Donald R. Lambert. "Commodity Channel Index: Tool for Trading Cyclic Trends". In: *Stocks & Commodities* 1 (říj. 1980), s. 120–122.

[32] Andrew W Lo. "The adaptive markets hypothesis: Market efficiency from an evolutionary perspective". In: *Journal of Portfolio Management, Forthcoming* (2004).

[33] Jason D Lohn, Gregory S Hornby a Derek S Linden. "An evolved antenna for deployment on NASA's space technology 5 mission". In: *Genetic Programming Theory and Practice II*. Springer, 2005, s. 301–315.

[34] John Paul Marney et al. "Risk Adjusted Returns to Technical Trading Rules: a Genetic Programming Approach". In: *7th International Conference of Society of Computational Economics*. Yale, 28-29 06 2001. URL: http://EconPapers.repec.org/RePEc:sce:scecf1:147.

[35] Serafin Martinez-Jaramillo a Edward P. K. Tsang. "An Heterogeneous, Endogenous and Coevolutionary GP-Based Financial Market". In: *IEEE Transactions on Evolutionary Computation* 13.1 (ún. 2009), s. 33–55. ISSN: 1089-778X. DOI: doi:10.1109/TEVC.2008.2011401.

[36] Jon McCormack. "New Challenges for Evolutionary Music and Art". In: *SIGEvolution* 1.1 (dub. 2006), s. 5–11. URL: http://www.sigevolution.org/2006/01/issue.pdf.

[37] Christopher J. Neely. "Risk-adjusted, ex ante, optimal technical trading rules in equity markets". In: *International Review of Economics and Finance* 12.1 (Spring 2003), s. 69–87. DOI: doi:10.1016/S1059-0560(02)00129-6. URL: http://research.stlouisfed.org/wp/1999/1999-015.pdf.

[38] Christopher J. Neely a Paul A. Weller. "Technical trading rules in the European Monetary System". In: *Journal of International Money and Finance* 18.3 (1999), s. 429–458. DOI: doi:10.1016/S0261-5606(99)85005-0. URL: http://research.stlouisfed.org/wp/1997/97-015.pdf.

[39] Christopher J. Neely, Paul A. Weller a Rob Dittmar. "Is Technical Analysis in the Foreign Exchange Market Profitable? A Genetic Programming Approach". In: *The Journal of Financial and Quantitative Analysis* 32.4 (pros.

1997), s. 405–426. ISSN: 00221090. URL: http://links.jstor.org/sici?sici=0022-1090(199712)32:4%3C405:ITAITF%3E2.0.CO;2-T.

[40] Christopher J. Neely, Paul A. Weller a Joshua M. Ulrich. "The Adaptive Markets Hypothesis: Evidence from the Foreign Exchange Market". In: *Journal of Financial and Quantitative Analysis* 44.2 (2009), s. 467–488. ISSN: 0022-1090. DOI: doi:10.1017/S0022109009090103.

[41] Ramin Rajabioun a Ashkan Rahimi-Kian. "A Genetic Programming Based Stock Price Predictor together with Mean-Variance Based Sell/Buy Actions". In: *Proceedings of the World Congress on Engineering, WCE 2008*. Ed. S. I. Ao et al. London: Newswood Limited, ún. 2008, s. 1136–1141. URL: http://www.iaeng.org/publication/WCE2008/WCE2008_pp1136-1141.pdf.

[42] Ken C. Sharman a Anna I. Esparcia-Alcazar. "Genetic Evolution of Symbolic Signal Models". In: *Proceedings of the Second International Conference on Natural Algorithms in Signal Processing, NASP'93*. Essex University, UK, 15-16 11 1993. URL: http://www.iti.upv.es/~anna/papers/natalg93.ps.

[43] William F Sharpe. "Mutual fund performance". In: *Journal of business* (1966), s. 119–138.

[44] Frank A Sortino a Robert Van Der Meer. "Downside risk". In: *The Journal of Portfolio Management* 17.4 (1991), s. 27–31.

[45] Ivan Tanev a Katsunori Shimohara. "Evolution of Human Competitive Driving Agent Operating a Scale Model of a Car". In: *Proceedings of SICE Annual Conference*. Kagawa University, Japan. 17-20 09 2007, s. 1582–1587. DOI: doi:10.1109/SICE.2007.4421235.

[46] Jack L Treynor. "How to rate management of investment funds". In: *Harvard business review* 43.1 (1965), s. 63–75.

[47] Tina Yu, Shu-Heng Chen et al. *Using genetic programming with lambda abstraction to find technical trading rules*. Tech. zpr. Society for Computational Economics, 2004.

Seznam tabulek

Seznam obrázků

Seznam algoritmů

Seznam použitých zkratek

^GSPC S&P 500 29, 35, 39, 42, 62

AAPL Apple Inc. 29, 31, 35, 36, 47, 62

ATR average true range 16

CCI commidity channel index 16, 22, 34

CMA centrovaný jednoduchý klouzavý průměr 15

DEAP Distributed Evolutionary Algorithms in Python 23

EMA exponenciální klouzavý průměr 15, 16

EMH teorie efektivního trhu 26, 51

GP genetické programování 4, 51

KDE kernel density estimation 40

KO The Coca-Cola Company 29, 35, 38, 42, 62

MACD moving average convergence divergence 16

MAD střední absolutní chyba 10

MAPE střední absolutní procentuální chyba 10

MD střední chyba 10, 35, 42

MPE střední procentuální chyba 10, 35, 42

MSE střední čtvercová chyba 10, 29, 35

NYSE The New York Stock Exchange 2

PEP Pepsico, Inc. 29, 35, 37, 42, 62

RMSE odmocnina střední čtvercové chyby 10

RSI relative streight index 16

SMA jednoduchý klouzavý průměr 15, 16, 34, 47, 51

WMA vážený klouzavý průměr 15