VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
INFORMATIKOS KATEDRA

Baigiamasis bakalauro darbas

Algoritminė prekyba naudojant tendencijas sekančias sistemas

(Algorithmic trading using trend following systems)

Atliko: 4 kurso, 1 pirmos grupės studentas
Ąžuolas Gaudutis

(parašas)

Darbo vadovas:
Dr. Aistis Raudys

(parašas)

Recenzentas:
Arūnas Janeliūnas

(parašas)

Vilnius
2019

Turinys

Įvadas

Algoritminė prekyba, tai prekybos sistema, kuri naudojasi pažangiais ir sudėtingais matematiniais modeliais bei formulėmis, siekiant greitų sprendimų ir tikslių transakcijų finansinėse rinkose. Tokios sistemos susideda iš greitų kompiuterinių programų ir sudėtingų algoritmų, kurie siekia maksimalaus pelno. Dažnai algoritminė prekyba yra paremta iš anksto apgalvota strategija, kaip pavyzdžiui tendencijų sekimu.

Naudojantis tendencijas sekančia sistema, yra tikima prielaida, jog rinkoje pokyčiai vyksta ne chaotiškai, bet pagal kryptį. Tikima, jog rinka juda su didėjančia kryptimi arba mažėjančia. Norint pasiekti geriausius rezultatus, reikia kuo greičiau ir tiksliau įžvelgti rinkos judėjimo kryptį ir pagal tai jau spręsti pirkti ar parduoti.

Šio darbo tikslas – ištirti tendencijas sekančių algoritmų veikimą siekiant didžiausios investicijų grąžos. Pagrindiniai uždaviniai – realizuoti tendencijas sekančius algoritmus „Python“ programavimo kalba, pasitelkus dalelių spiečiaus optimizavimo metodą nustatyti greičiau ir tiksliau tendencijas, taip pat ištirti kokią įtaką veikimui daro skirtingi dalelių spiečiaus parametrai bei ištirti algoritmo veikimą įvairiuose akcijų tipuose. Galiausiai modifikuoti algoritmą siekiant greitesnių rezultatų.

Pradžioje bus pateikiama literatūros apžvalga susijusi su algoritminė prekyba, tendencijas sekančiais algoritmais ir dalelių spiečiaus optimizavimo metodu. Toliau sekančiuose skyriuose bus įgyvendinami tendencijas sekantys algoritmai. Viename iš skyrių bus naudojamas dalelių spiečiaus optimizavimo metodas kartu su tendencijas sekančiu algoritmu. Taip pat, šiuose skyriuose bus pateikiami brėžiniai susiję su algoritmo veikimu, kurie buvo nubraižyti pasitelkiant Python kalbą.

Darbo pabaigoje bus pateikiami bendri pamąstymai apie gautus rezultatus, algoritmus ir metodus.

1. Algoritminė prekyba

Kaip jau buvo minėta anksčiau, algoritminė prekyba tai yra kompiuterinė programa, kuri pasikliauna apibrėžtomis taisyklėmis ir instrukcijų rinkiniu, tam kad atlikinėtų pirkimo ar pardavimo veiksmus finansinėse rinkose [WDD09]. Tos apibrėžtos taisyklės ir instrukcijų rinkiniai dažnai susideda iš laiko, kada pirkti ar parduoti, kainos, kiekio ar bet kokio kito matematinio modelio. Tokio tipo prekyba pradėjo populiarėti nuo 1970 metų, kuomet Jungtinių Amerikos Valstijų finansų rinkose buvo pradėtos diegti kompiuterinės sistemos. Dabar tokio tipo prekyba dažnai užsiima investiciniai bankai, pensijų ir investicijų fondai siekdami atlikinėti sudėtingas ir didelės apimties transakcijas efektyviai, kurių praktiškai neįmanomą atlikti žmogui. Algoritminė prekyba gali būti naudojama įvairiose situacijose [NMT+11], pavyzdžiui užsakymo įvykdymui, arbitraže ar strategijose, kurios yra paremtos tendencijų sekimu, apie kurias ir bus plačiau aprašoma šiame darbe.

1.1 Tendencijas sekantys algoritmai

Tendencijas sekančios sistemos nesiekia nuspėti rinkos pokyčių, o bando kuo greičiau įžvelgti, kuria kryptimi rinka juda – kaina didėja (tendencija aukštyn) ar kaina mažėja (tendencija žemyn). Pagrindiniai veiksmai, kuriuos atlieka tendencijas sekanti sistema, tai yra pozicijos atidarymas ir uždarymas. Pozicijos atidarymas, tai yra akcijos nupirkimas, tam tikru laiko momentu, o pozicijos uždarymas akcijos pardavimas. Pozicijos atidarymas tam tikru laiko momentu, siekiant maksimizuoti galutinį pelną ir jos uždarymas kitu laiko momentu yra pagrindiniai uždaviniai, kurie yra sprendžiami tendencijų sekime.

Paprastai analizuojant akcijų kainas yra sunku rasti tendencijas, dėl per didėlio kainų svyravimo. Dažnai akcijų kainos priklauso nuo daug išorinių veiksnių, kaip pavyzdžiui įvairių naujienų, ekonominių rodiklių ar didelio mąsto įvykių, todėl norint įžvelgti tendencijas tenka naudotis techniniais indikatoriais. Patys techniniai indikatoriai ir buvo sukurti, kad būtų galima kuo paprasčiau akcijų kainas analizuoti. Techninių indikatorių yra daug, tačiau tarp paprasčiausių ir labiausiai naudojamų visada būna slenkantys vidurkiai. Slenkantis vidurkis išlygina akcijų kainas ir pateikia jas paprastesniu, labiau skaitomu formatu. Paprastas slenkantis vidurkis yra apskaičiuojamas tokia formule:

$$PSV = \frac{A_1 + A_2 + \cdots + A_n}{n}$$

,kur A_n- akcijos kaina, laiko momentu n. Tačiau, slenkantys vidurkiai turi ir savo trūkumų. Vienas didesnių trūkumų yra tas, jog jie atsilieka ir dažnai dalis duomenų nebūna tinkamai panaudoti, todėl analizuojant akcijų kainas ir siekiant jose įžvelgti tendencijas yra naudojami daugiau nei vienas techninis indikatorius.

Dar vienas populiarus techninis indikatorius, kuris dažnai naudojamas tendencijas sekančiose sistemose, tai būtų OBV (ang, *On-Balance Volume*) indikatorius. Sukurtas 1963 metais [Gra63], šis indikatorius siekia nuspėti akcijų kainas ar rinkos judėjimo kryptis atsižvelgdamas į prieš tai parduotų akcijų kiekį. OBV indikatorius yra apskaičiuojamas tokia formule:

$$OBV = OBV_{pr} + \begin{cases} kiekis, & jei\ uždarymo\ kiekis > uždarymo\ kiekis_{pr} \\ 0, & jei\ uždarymo\ kiekis = uždarymo\ kiekis_{pr} \\ -kiekis, & jei\ uždarymo\ kiekis < uždarymo\ kiekis_{pr} \end{cases}$$

,kur OBV_{pr} – OBV reikšmė praeityje. Jei akcijų kainos kyla, tai ir OBV indikatoriaus reikšmės didėja, jei kainos smunka, tai ir OBV reikšmės mažėja.

Tendencijų sekimas, kaip investavimo būdas yra naudojamas jau senai [HOP17]. Tačiau ši investavimo strategija yra įmanoma ne tik su akcijomis ar indeksais, bet paprastas tendencijas galima rasti ir jomis pasinaudoti valiutų rinkose [Jam03]. Pasinaudojus paprastu slenkančiu vidurkiu [Jam03] buvo atrasta valiutų, kuriose galima įžvelgti tendencijas, tačiau buvo ir tokių, kuriose tendencijų negalima pamatyti. Pavyzdys šiuo atveju būtų USD/CAD (Amerikos dolerio ir Kanados dolerio) pora, kur kurso santykis labai priklauso nuo dviejų šalių ekonomikų.

1.2 Dalelių spiečiaus optimizavimo metodas

Dažnai prekybos sistemas tenka optimizuoti pasitelkus įvairius metodus ir algoritmus. Naudojant tokius metodus yra bandoma rasti geriausius pradinius parametrus [MLW17], kurie duotų labiau patikimus rezultatus. Taip pat dažnai yra siekiama optimizuoti investicijų portfelius, pasitelkus įvairius optimizavimo metodus [DFV+08][JKM05].

Vienas iš optimizavimo metodų, kuris šiame darbe bus naudojamas, tai dalelių spiečiaus optimizavimo metodas. Dalelių spiečiaus optimizavimas tai yra optimizavimo metodas, kuriuo siekiama optimizuoti problemą, iteracijomis gerinant kandidatinį sprendinį pasitelkus sprendžiamą problemą. Pradinė idėja šio algoritmo buvo simuliuoti socialinį elgesį, pasitelkus paukščių ar žuvų būrių judėjimo modelius, tačiau vėliau tai peraugo į optimizavimo metodą [JE95]. Šis metodas sprendžia problemą pradėdamas su populiacija kandidatinių sprendinių, šiuo atvejų dalelių, kurios

juda paieškos erdvėje, skaičiuojant jų poziciją ir greitį toje erdvėje, pagal paprastą matematinę formulę. Kiekviena dalelė juda atsižvelgdama ne tik į savo geriausią poziciją, bet ir į viso spiečiaus geriausią poziciją, kuri yra atnaujinama, kuomet yra randama geresnė pozicija kitų dalelių. Toks navigacijos būdas paieškos erdvėje leidžia visam dalelių spiečiui judėti link geriausio sprendinio. Dalelių pozicija paieškos erdvėje yra apskaičiuojama tokia formule:

$$x_i = x_i(t) + v_i(t+1)$$

,kur t yra laikas, o v_i yra greitis paieškos erdvėje apskaičiuojamas formule:

$$v_i(t+1) = \omega v_i(t) + \phi_1 R_1\big(p_i - x_i(t)\big) + \phi_2 R_2\big(g_i - x_i(t)\big)$$

Šioje formulėje $R_i, i = 1{,}2$ atitinka atsitiktinius skaičius aibėje [0, 1]. ω yra inercijos faktorius ir visuomet yra teigiama konstanta. Parametrai $\phi_{1,2} \geq 0$ yra iš anksto nustatomi. p_i dažnai vadinamas $pbest$ yra dalelės geriausias sprendinys, o g_i, kuris dažnai vadinamas $gbest$ yra viso spiečiaus geriausias sprendinys [LSZ+18].

Šiame darbe dalelių spiečiaus optimizavimo metodas bus naudojamas tendencijas sekančiame algoritme, siekiant nustatyti rinkos judėjimo kryptį ir atrasti taškus, kada galima pirkti ar parduoti. Tačiau šį optimizavimo metodą galima taikyti ne tik tendencijų sekime, bet juo pasinaudojus galima kurti sudėtingus ir paprastus taisyklių rinkinius, kuriais remiantis jau būtų prekiaujama. [WNK+17] [WYC14]

2. Statinis tendencijas sekantis algoritmas

Automatizuotos prekybos algoritmai yra skaidomi į dvi grupes: reaguojantys algoritmai ir nuspėjamieji [FST12]. Visi tendencijas sekantys algoritmai siekia, kuo greičiau nustatyti rinkos judėjimo kryptį, t.y. tendenciją, kad galima būtų labiau tai išnaudoti ir maksimizuoti pelną. Tai ir bus siekiama padaryti šiame skyriuje, realizuojant statinę algoritmo versiją.

2.1 Apie algoritmą

Visi tendencijas sekantys algoritmai turi vieną bendrą bruožą: juos realizuojant yra naudojamos konstantos P ir Q, kurios yra skirtos rinkos krypties nustatymui ir reiškia arba tendencija aukštyn (akcijų kaina rinkoje didėja) arba žemyn (akcijų kaina rinkoje mažėja). Statiniame tendencijas sekančiame algoritme šios reikšmės jas nustačius išlieka nepakitusios ir niekaip nesąveikauja su kitomis algoritmo dalimis.

2.2 Realizacija

Prieš realizuojant patį algoritmą, duomenis būtina nusiskaityti iš duomenų failo. Duomenų failas dažniausiai bus saugomas *.csv* formatu, t.y. visos skirtingos reikšmės bus atskirtos kableliais. Šiame darbe naudoto failo formatas buvo toks:

```
Date,Open,High,Low,Close,Adj Close,Volume
2016-12-01,110.370003,110.940002,109.029999,109.489998,107.757156,37086900
2016-12-02,109.169998,110.089996,108.849998,109.900002,108.160675,26528000
2016-12-05,110.000000,110.029999,108.250000,109.110001,107.383171,34324500
2016-12-08,110.860001,112.430000,110.599998,112.120003,110.345543,27068300
```

Šiame faile pirma eilutė atitinką antraštę, o jau visos sekančios yra skirtos duomenims. Duomenys prasideda formatu *YYYY-MM-DD*, kuris yra skirtas datai nusakyti, o toliau yra naudojamas *OHLC*(*Open, High, Low, Close*) formatas skirtas nusakyti akcijos kainai su kuria buvo pradėta prekiauti (Open), aukščiausia bei žemiausia kaina, kurią akcija pasiekė per šią dieną(High, Low) ir galiausiai kaina su kuria buvo akcija nustota prekiauti (Close). Taip pat paskutinėje reikšmė eilutėje yra skirta nusakyti kiek buvo suprekiauta akcijos per dieną.

Failo nuskaitymas vyksta pasinaudojant „pandas“ bibliotekos *read_csv* funkcija:

```
df = pd.read_csv(file_name)
```

Ši funkcija sugeba nuskaityti failo antraštę (pirmą eilutę faile) ir sukurti bei grąžinti *DataFrame* tipo objektą į kintamąjį df.

DataFrame tipo objektai dažnai bus naudojami šiame darbe, nes jie yra pagrindiniai duomenų tipai, kuomet yra dirbama su „pandas“ biblioteka. *DataFrame* objektai yra tarsi matricos pavidalo:

```
          Stulpelis1      Stulpelis2
Indeksas  Reikšmė1        Reikšmė2
```

Kurios reikšmės yra pasiekiamos nurodant stulpelio pavadinimą ir indekso numerį, t.y. *df[‘stulpelio pavadinimas‘][indeksas]*.

Nusiskaičius failą, jau galima ir pradėti tolimesnį duomenų apdorojimą. Kadangi pagal nuskaitytus duomenis negalime iš karto nuspręsti, kaip realizuoti algoritmą, nes duomenys nėra

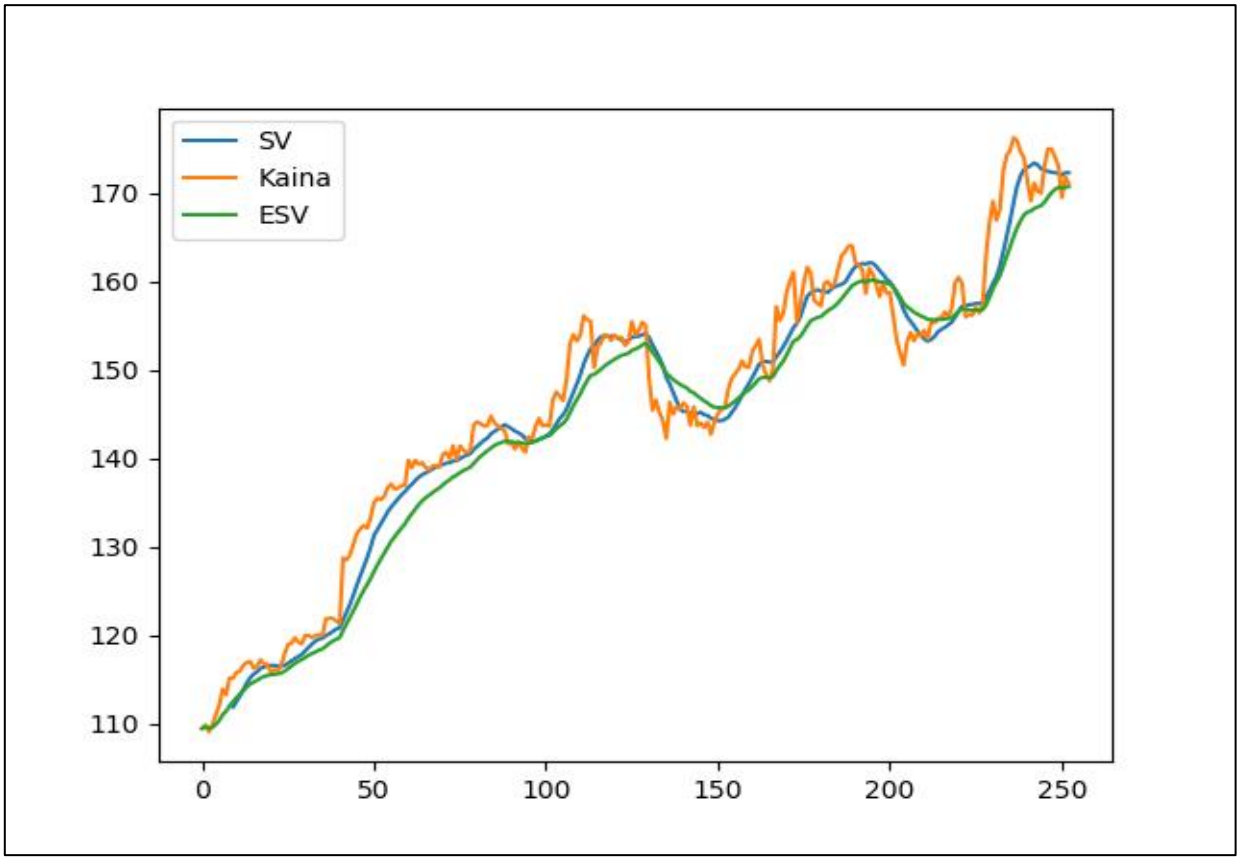
pastovūs nuo rinkos svyravimų, todėl turime pasitelkti techninius indikatorius, kad duomenis būtų galima išlyginti ir tik tuomet pereiti prie kitų veiksmų. [FST12] darbe yra siūloma naudoti šiai operacijai Eksponentinį slenkantį vidurkį (ESV), kuris yra paskaičiuojamas pagal formulę:

$$ESV_{(t)} = \left(kaina_{(t)} - ESV_{(t-1)} \times \frac{2}{n-1}\right) + ESV_{(t-1)}$$

Šis techninis indikatorius yra panašus į paprastą slenkantį vidurkį, tačiau šis indikatorius reaguoja greičiau į kainų pakitimus. Taip pat kainų pokyčiai naudojant šį indikatorių yra labiau išlyginami, nei naudojant paprastąjį slenkantį vidurkį. Tai matyti *1 pav.* diagramoje, kurioje buvo pasinaudota „pandas“ bibliotekos funkcijomis skaičiuojant eksponentinį bei paprastąjį slenkantį vidurkį:

```
df['Close'].rolling(window=10).mean() // paprastas slenkantis vidurkis
df['Close'].ewm(com=10).mean() // eksponentinis slenkantis vidurkis
```

bei „matplotlib“ funkcija *plot()*, kad būtų nubraižytas grafikas:



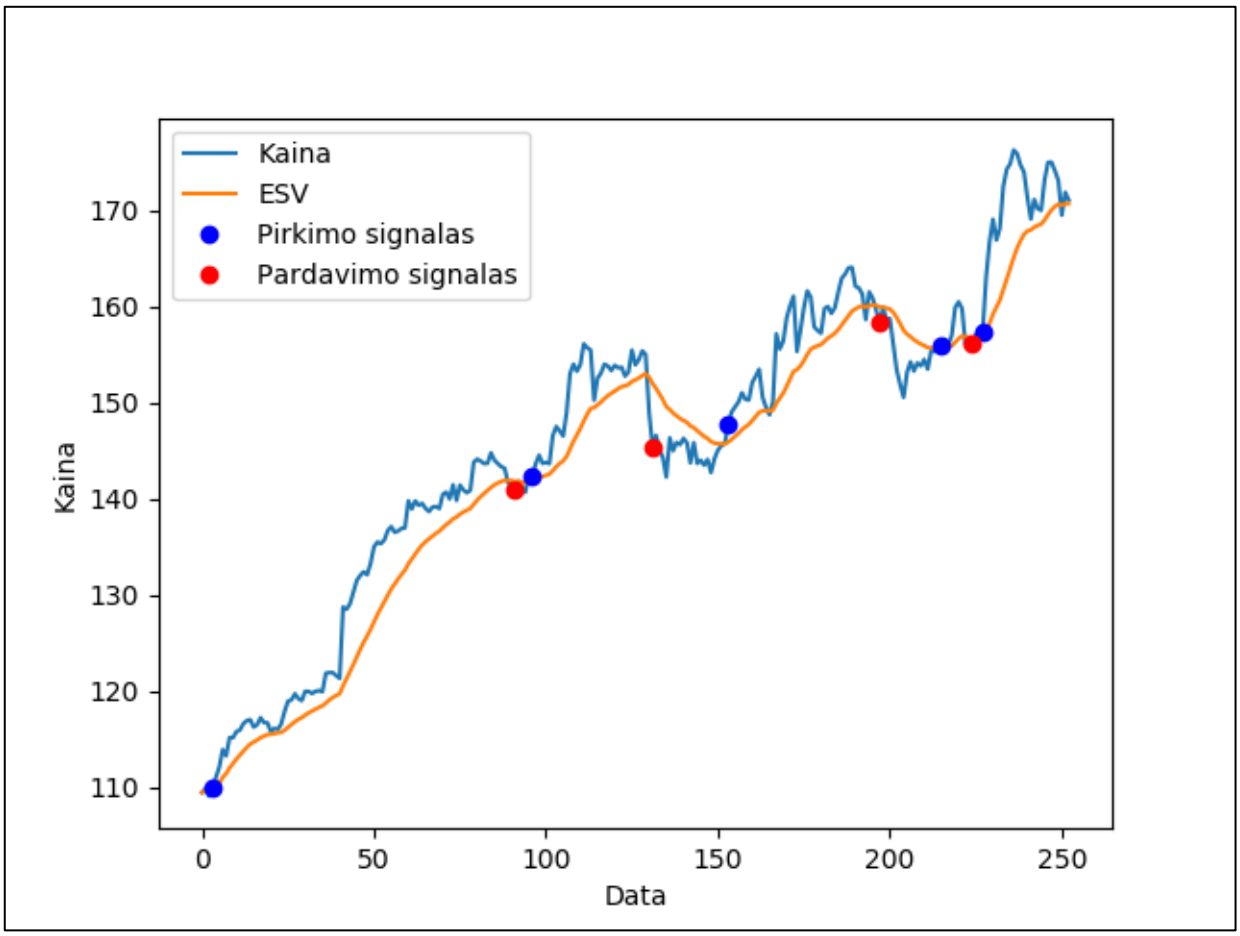
1 pav. Paprastas ir Eksponentinis slenkantis vidurkis kartu su kainos reikšmėmis

Statinio tendencijas sekančio algoritmo pseudokodas [LSZ+18]:

```
While ne simuliacijos pabaiga do
    Apskaičiuoti ESV(t)
    If ESV(t) > ESV(t-1) && ESV(t-2) > ESV(t-1) then
        dlt = ESV(t-1)
        Dabartinė tendencija yra didėjanti
    Else if ESV(t) < ESV(t-1) && ESV(t-2) < ESV(t-1) then
        mlt = ESV(t-1)
        Dabartinė tendencija yra mažėjanti
    If nėra atidarytos pozicijos && dabartinė tendencija yra didėjanti then
        If ESV(t) – dlt >= P then
            Atidaryti poziciją
    Else if yra atidaryta pozicija && dabartinė tendencija yra mažėjanti then
        If mlt – ESV(t) >= Q then
            Uždaryti poziciją
End
If simuliacijos pabaiga && yra atidaryta pozicija then
    Uždaryti poziciją
End
```

Šiame pseudokode iš pradžių yra paskaičiuojamas eksponentinis slenkantis vidurkis, o vėliau jau turint jo reikšmės yra siekiama nustatyti tendencijų kryptį ir kintamuosius *dlt*(didėjimo lūžio taškas) ir *mlt*(mažėjimo lūžio taškas). Turint šiuos kintamuosius bei žinant tendencijos kryptį, jau galima apskaičiuoti tiriamos tendencijos ilgį, palyginti su iš anksto įvestomis P ir Q reikšmėmis ir pagal gautus rezultatus sukurti pirkimo ar pardavimo signalus (atidaryti ar uždaryti pozicijas).

Paėmus P = 3 ir Q = 3 (ši reikšmė buvo parinkta išanalizavus duotus duomenis, kad būtų gauti kuo geresni rezultatai) algoritmo rezultatus galime matyti grafike (*2 pav)*:



2 pav. Pirkimo ir pardavimo signalai gauti pagal Statinį algoritmą

Grafikas buvo nubraižytas pasitelkus „matplotlib“ bibliotekos funkciją *plot()*, jau turint pirkimo ir pardavimo transakcijų aibę. Grafike matyti, kai yra nustatoma tendencija aukštyn(didėjanti), pagal duotą statinę P reikšmę, tuomet yra sukuriamas „Pirkimo signalas“, pažymėtas mėlyna spalva ir atvirkščiai, kuomet nustatoma tendencija žemyn(mažėjanti), pagal Q reikšmę, tai grafike yra atspausdinamas „Pardavimo signalas“ raudona spalva. Galima pastebėti, jog šie pirkimo/pardavimo taškai yra atidedami, ne ant ESV kreivės, o ant Kainos, nors algoritmas parinko, juos nežinodamas apie kainų reikšmes.

2.3 Rezultatai

Testavimai vyko su duomenimis gautais iš „*Yahoo finance*“ puslapio su 7 populiarias indeksais: IXIC, SP500, HSI, TWII, FTSE, DJI, N225. Duomenys buvo imami nuo 2014-08-01 iki 2015-07-30. Papildomai buvo skaičiuojamas sėkmės rodiklis, pagal formulę:

$$SR = \frac{pelningų\ transakcijų\ skaičius}{visų\ transakcijų\ skaičius}$$

bei investicijų grąža:

$$Investicijų\ grąža\ \% = \frac{(pardavimo\ kaina - pirkimo\ kaina) * 100}{pirkimo\ kaina}$$

P ir Q reikšmės buvo imamos nuo 10 iki 100, bet didinamos kiekvienam naujam tyrimui kas 10 vienetų. Testavimo rezultatai yra pateikiami lentelėse:

Akcijų simbolis	**P = 10, Q = 10**		**P = 20, Q = 20**	
	Sėkmės rodiklis	**Investicijų grąža**	**Sėkmės rodiklis**	**Investicijų grąža**
IXIC	0.4	1.63%	0.37	0.65%
SP500	0.3	-2.68%	0.5	-0.67%
HSI	0.35	7.96%	0.41	7.89%
TWII	0.4	-1.11%	0.42	1.6%
FTSE	0.25	-4.82%	0.18	-13.75%
DJI	0.53	27.7%	0.57	20.81%
N225	0.5	26.15%	0.22	-8.24%

1 lentelė. Statinio algoritmo rezultatai, kai P ir Q reikšmės lygios 10 ir 20

Akcijų simbolis	**P = 30, Q = 30**		**P = 40, Q = 40**	
	Sėkmės rodiklis	**Investicijų grąža**	**Sėkmės rodiklis**	**Investicijų grąža**
IXIC	0.5	2.59%	0	-9.76%
SP500	0	-11.75%	0	-9.61%
HSI	0.27	4.9%	0.27	1.95%
TWII	0.4	-1.39%	0.4	-2.95%
FTSE	0.2	-12.61%	0.25	-14.27%
DJI	0.65	16.89%	0.57	12.71%
N225	0.22	-9.6%	0.14	-10.55%

2 lentelė. Statinio algoritmo rezultatai, kai P ir Q reikšmės lygios 30 ir 40

Akcijų simbolis	**P = 50, Q = 50**		**P = 60, Q = 60**	
	Sėkmės rodiklis	**Investicijų grąža**	**Sėkmės rodiklis**	**Investicijų grąža**

IXIC	0	-8.13%	0	-9.32%
SP500	0	-9.53%	0	-6.28%
HSI	0.18	-3.38%	0.18	-5.1%
TWII	0.4	-3.47%	0.5	-0.73%
FTSE	0.125	-15.71%	0.16	-9.92%
DJI	0.57	9.69%	0.25	-30.14%
N225	0.5	16.82%	0.5	15.45%

3 lentelė. Statinio algoritmo rezultatai, kai P ir Q reikšmės lygios 50 ir 60

Akcijų simbolis	**P = 70, Q = 70**		**P = 80, Q = 80**	
	Sėkmės rodiklis	**Investicijų grąža**	**Sėkmės rodiklis**	**Investicijų grąža**
IXIC	0	-4.62%	0	-8.68%
SP500	0	-3.29%	0	-5.20%
HSI	0.2	-6.26%	0.22	-3.56%
TWII	0.33	1.53%	0.33	-0.45%
FTSE	0.33	-10.98%	0.2	-10.24%
DJI	0.26	-26.43%	0.25	-24.87%
N225	0.66	18.61%	0.2	-8.52%

4 lentelė. Statinio algoritmo rezultatai, kai P ir Q reikšmės lygios 70 ir 80

Akcijų simbolis	**P = 90, Q = 90**		**P = 100, Q = 100**	
	Sėkmės rodiklis	**Investicijų grąža**	**Sėkmės rodiklis**	**Investicijų grąža**
IXIC	0	-7.75%	0	-8.18%
SP500	0	-6.28%	0	-6.49%
HSI	0.25	-0.89%	0.28	2.66%
TWII	0.33	-0.75%	0.33	-0.81%
FTSE	0.2	-12.35%	0.25	-9.35%
DJI	0.16	-27.49%	0.16	-29.82%
N225	0.2	-8.88%	0.2	-9.55%

5 lentelė. Statinio algoritmo rezultatai, kai P ir Q reikšmės lygios 90 ir 100

Testuojant algoritmą, buvo imamos įvairios reikšmės, kurios geriausiai atspindėtų jo veikimą, tačiau šios reikšmės dirbant su kitais duomenimis gali visiškai netikti ir generuoti visai kitus rezultatus. Geriausi rezultatai buvo gauti su P,Q = 10. Investicijų grąžos vidurkis siekė net 7,83 procentus. Didinant P ir Q reikšmes kas 10 rezultatai prastėjo ir investicijų grąža beveik visais atvejais būdavo neigiama. Pažvelgus į rezultatus gautus su P ir Q reikšmėmis nuo 40 iki 100, galima matyti, jog indeksu IXIC ir SP500 sėkmės rodikliai visad būdavo 0, o tai reikšdavo, jog visos atliktos transakcijos buvo nepelningos. Statinio Algoritmo didžiausia problema yra ta, jog jo P ir Q reikšmės išlieka nepakitusios, o tai reiškia kad algoritmo veikimo pasisekimas priklauso nuo to, kaip pasiseks nuspėti šias reikšmes teisingai ir tikėtis, jog rinka reaguos į tokį sprendimą geranoriškai.

3. Dinaminis tendencijas sekantis algoritmas

Sekantis algoritmas iš reaguojančiųjų grupės [FST12] yra dinaminis tendencijas sekantis algoritmas.

3.1 Apie algoritmą

Iš pavadinimo galima suprasti, jog šis algoritmas bus kintantis, dinamiškas. Jei prieš tai statiniame algoritme P ir Q reikšmės nepakisdavo, tai dinaminiame jos pradeda kisti pagal rinkos kryptį. Papildomai prie šio algoritmo lyginant su praeitu atsiranda dar vienas techninės analizės indikatorius – reliatyvaus stiprumo indeksas (RSI) [FST12]. RSI – tai pagreičio indikatorius, kuris palygina neseniai pelningas ir nuostolingas akcijas, kad išmatuotų jų greitį bei pokyčius. Šio indikatoriaus reikšmės svyruoja nuo 0 iki 100.

3.2 Realizacija

Duomenų failas bei nuskaitymo procesas išlieka identiškas 2.2 poskyryje nurodytam procesui. Bus naudojami taip pat *DataFrame* tipo objektai. Kadangi šiame skyriuje bus naudojamas RSI indikatorius, o jis priešingai, nei praeitame skyriuje aprašyti indikatoriai, nėra tarp standartinių naudojamos bibliotekos funkcijų, tai jį tenka realizuoti Python funkcija. [FST12] darbe RSI yra realizuojamas funkcija:

$$RSI_{(t)} = 100 - \frac{100}{1 + RS}$$

$$, kur\ RS\ = \frac{AU_{(t)}}{AD_{(t)}}$$

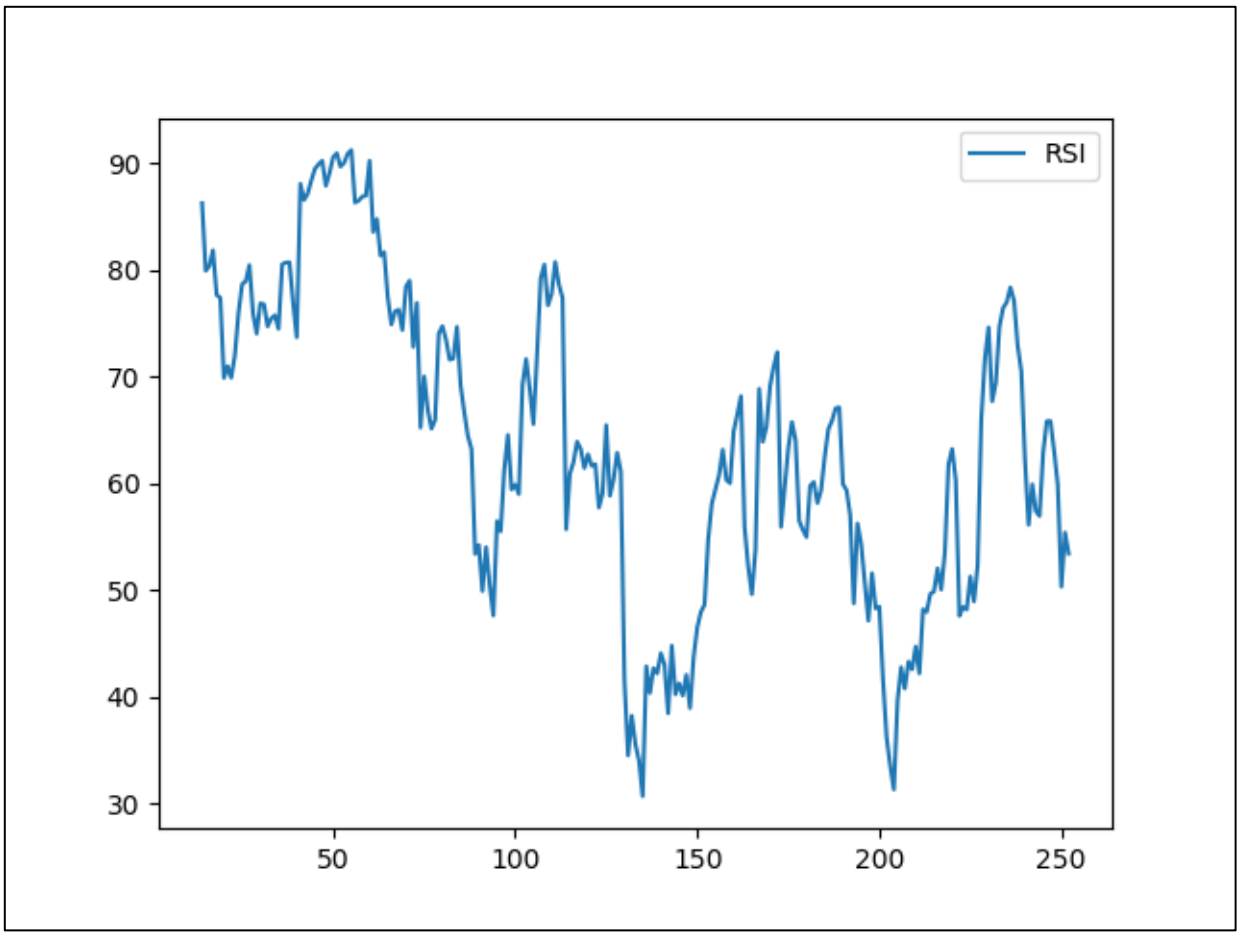
$$ir\ AU_{(t)} = \frac{Aukštyn(t) + Aukštyn(t-1) + \cdots + Aukštyn(t-n+1)}{n}$$

$$bei\ AD_{(t)} = \frac{Žemyn(t) + Žemyn(t-1) + \cdots + Žemyn(t-n+1)}{n}$$

, kur AU yra kainos vidutiniškas kilimas per n periodų, o AD atvirkščiai – kritimas, t yra laikas, n periodų skaičius, kuris dažniausiai atitinka reikšmę 14. RSI realizacija Python programavimo kalba:

```
def RSI(series, period):
    delta = series.diff().dropna()
    u = delta * 0
    d = u.copy()
    u[delta > 0] = delta[delta > 0]
    d[delta < 0] = -delta[delta < 0]
    u[u.index[period - 1]] = np.mean(u[:period])
    u = u.drop(u.index[:(period - 1)])
    d[d.index[period - 1]] = np.mean(d[:period])
    d = d.drop(d.index[:(period - 1)])
    rs = pd.stats.moments.ewma(u, com=period - 1, adjust=False) / pd.stats.moments.ewma(d,
com=period - 1, adjust=False)
    return 100 - 100 / (1 + rs)
```

Funkcija iškviečiama perduodant, kaip parametrus *DataFrame* tipo objekto stulpelį ir periodo dydį, kuris, kaip ir minėta dažniausiai turėtų būti 14. Pavyzdžiui: *df[‚RSI‘] = RSI(df[‘Close’], 14)*. Reliatyvaus stiprumo indekso funkcija pritaikyta rinkos duomenims:



3pav. RSI indikatoriaus grafikas

RSI geriausia naudoti, kaip pagalbinį įrankį kartu su kitais indikatoriais. Pagal [FST12] RSI yra naudojamas dinaminėje strategijoje tam, kad būtų galima suskaičiuoti atitinkamas P ir Q reikšmes realiu laiku. Išanalizavus, kaip šis indikatorius reaguoja į rinkos pokyčius, galima susikurti kriterijų rinkinį, kuriuo remiantis bus generuojami pirkimo ir pardavimo signalai.

P reikšmės nustatymas (pirkimo signalo generavimas):

- Kaina didėja.
- RSI(t) yra didesnis nei ESV(RSI(t)).
- ESV(RSI(t)) yra mažesnis nei 40 arba didesnis nei 60.

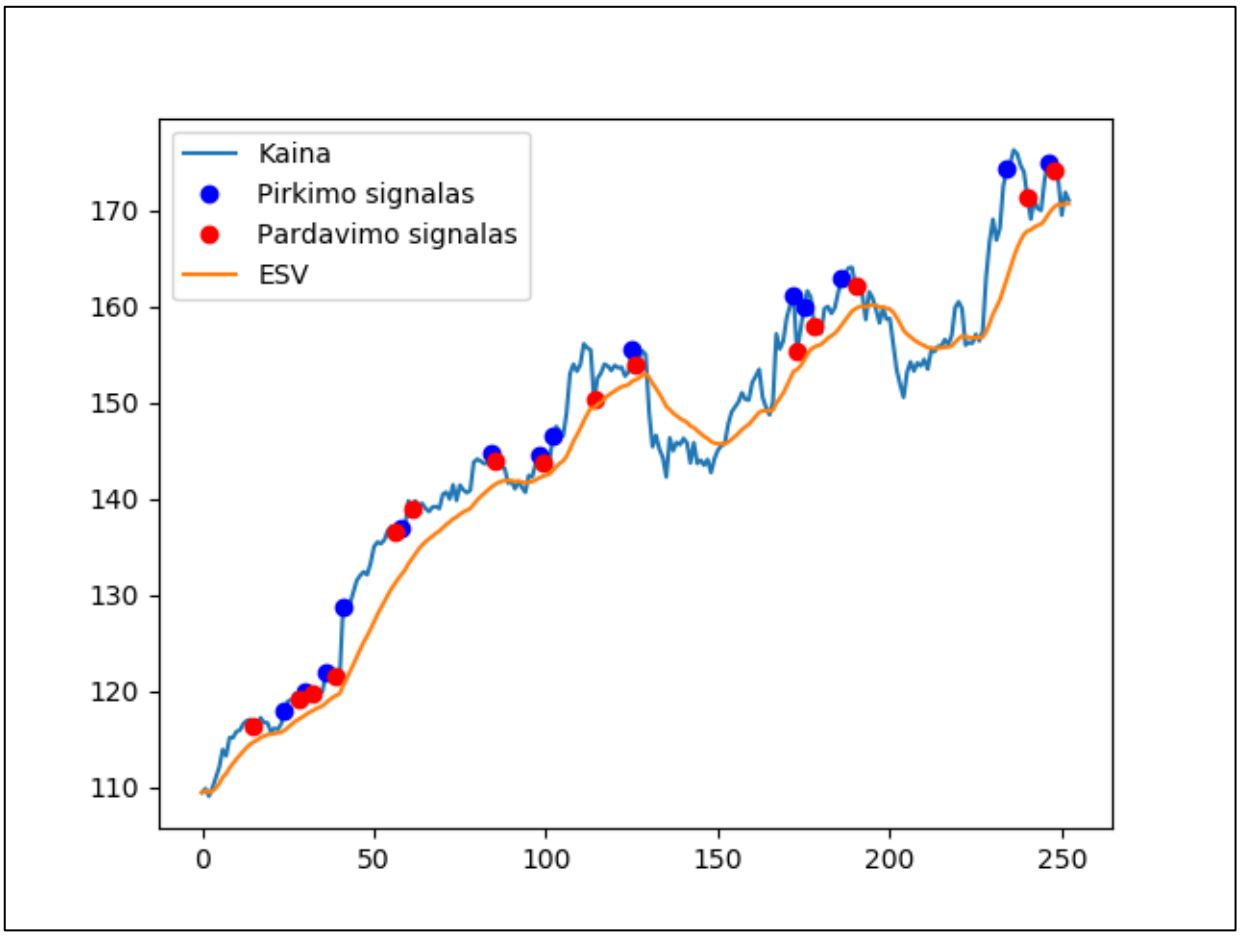
Q reikšmės nustatymas (pardavimo signalo generavimas):

- Kaina mažėja.
- RSI(t) yra mažesnis nei ESV(RSI(t)).
- ESV(RSI(t)) yra mažesnis nei 40 arba didesnis nei 60.

Šiuose kriterijuose reikšmės 40 ir 60 parinktos ne atsitiktinai, bet atsižvelgiant į tai, jog jos leidžia prekiauti su mažesne rizika. Dabar P ir Q reikšmės keičiasi dinamiškai kartu su RSI indikatoriumi. Dinaminio tendencijas sekančio algoritmo pseudokodas [LSZ+18]:

While ne simuliacijos pabaiga **do:**
    Apskaičiuoti RSI(t) ir RSI(ESV(t))
    If RSI(t) > 50 **then**
        Dabartinė tendencija yra didėjanti
    Else if RSI(t) < 50 **then**
        Dabartinė tendencija yra mažėjanti
    If RSI(t) > RSI(ESV(t)) && (40 < RSI(ESV(t)) || (RSI(ESV(t))) > 60) && dabartinė tendencija yra didėjanti && nėra atidarytos pozicijos **then**
        Atidaryti poziciją
    Else if RSI(t) > RSI(ESV(t)) && (40 < RSI(ESV(t)) || (RSI(ESV(t))) > 60) && dabartinė tendencija yra mažėjanti && yra atidaryta pozicija **then**
        Uždaryti poziciją
End
If simuliacijos pabaiga && yra atidaryta pozicija **then**
    Uždaryti poziciją
End

Šiame psedokode iš pradžių yra apskaičiuojamos RSI(t) ir RSI(ESV(t)) reikšmės. Toliau pasinaudojus jau apskaičiuotomis RSI reikšmėmis yra nustatoma tendencija: jei RSI(t) yra daugiau nei 50, tai akcijų kaina auga ir dabartinė tendencija yra didėjanti, o jei RSI(t) yra mažiau nei 50, tai reiškia, jog kaina smunka ir dabartinė tendencija yra mažėjanti. Sekantys algoritmo veiksmai atitinka anksčiau apibrėžus kriterijus. Pasinaudojus „matplotlib“ funkcijos *plot()* galimybėmis rezultatus galime atvaizduoti *4pav* grafiku, kuriame ir bus matomi pirkimo bei pardavimo signalai, kaina ir eksponentinis slenkantis vidurkis.



4pav.Pirkimo ir pardavimo signalai gauti pagal Dinaminį algoritmą

3.3 Rezultatai

Algoritmo testavimui buvo naudojami tie patys duomenys gauti iš tų pačių šaltinių, kaip ir statiniame algoritme, siekiant palyginti jų veikimus ir skirtumus. Rezultatai pateikiami lentelėje:

Akcijų simbolis	Transakcijų skaičius	Pelningos transakcijos	Sėkmės rodiklis	Investicijų grąža
IXIC	5	2	0.4	2.71%
SP500	3	3	1.0	6.63%
HSI	4	1	0.25	-3.07%
TWII	3	1	0.33	-4.36%
FTSE	4	0	0	-10.55%
DJI	12	7	0.58	11.71%
N225	5	3	0.6	4.29%

6 lentelė. Dinaminio algoritmo rezultatai

Šiam algoritmui pavyko gauti gerus rezultatus su indeksu SP500, kur palyginus su dinaminiu visais atvejais būdavo gaunama neigiama investicijų grąža. Investicijų grąžos vidurkis siekė 1,05 procentus. Apžvelgus šio algoritmo veikimą galima daryti prielaidą, jog dinaminis algoritmas yra daug pranašesnis, nes jis veikia dinamiškai, kinta ir prisitaiko kartu su rinkos pokyčiais, tačiau tai ne visada būtų tiesa, nes ši prekybos strategija remiasi RSI indikatoriumi, o jei rinkoje bus daug didelių kainų pakilimų ir kritimų, tuomet bus sugeneruota atitinkamai daug blogų transakcijų, kurios galėtų privesti prie didesnių nuostolių.

4. Prisimenantis algoritmas

Trečiasis algoritmas bus šiek tiek kitoks, tačiau jį taip pat galima priskirti reaguojančiųjų grupei.

4.1 Apie algoritmą

Kaip ir minėta algoritmas skirsis nuo pirmų dviejų. Šio algoritmo skirtumas yra tas, jog jis remiasi ne techniniais indikatoriais ar kokia kita analize, o prieš tai sėkmingomis strategijomis. Kitaip tariant šiam algoritmui reikalingos kitos strategijos, kuriose jau gali būti naudojama techninė analizė, tam, kad jis išsirinktų pačią geriausią pagal jam išskirtus kriterijus.

Yra išskiriami 4 žingsniai, kuriuos šis algoritmas turi atlikti, kad būtų sudarytas pirkimo ar pardavimo sandoris: *pirminis apdorojimas, išsirinkimas, patikrinimas* ir *patvirtinimas* [FTP12]. Pirminio apdorojimo žingsnyje yra sudedamos visos strategijos, kurios bus naudojamos vėliau išsirinkimui, į vieną telkinį. Išsirinkimo žingsnyje telkinyje yra ieškomi, kuo panašesni pavyzdžiai, iš kurių bus renkamas geriausias. Patikrinime yra imamas kiekvienas iš pavyzdžių ir tikrinamas su dabartine rinka. Galiausiai patvirtinime yra nustatoma pagal, kokią strategiją bus toliau prekiaujama.

4.2 Realizacija

Šio algoritmo realizacija bus apžvelgta daugiau iš teorinės pusės, nes pilnai realizacijai yra reikalingi ne tik rinkos duomenys, kaip ir prieš tai naudotuose algoritmuose, tačiau reikalingas ir papildomas ryšys su pačia rinka, kad joje jau būtų galima atlikinėti veiksmus realiu laiku.

Pirmasis algoritmo žingsnis yra *pirminis_apdorojimas*, kuriame reikia surinkti duomenis iš praeitų rinkų ir sudėti į vieną bendrą telkinį. Pagal [FTP12] yra siūloma pavyzdžius šiame telkinyje sudarinėti iš dviejų dalių: to laikotarpio tendencijos bei strategijos pagal, kurią buvo prekiaujama.

Viena iš galimybių, kaip tai galima būtų realizuoti, tai naudojant Python struktūra, kurioje reikšmės yra pasiekiamos nurodant raktą. Pvz:

```
Pavyzdys  = {
   ‘tendencija’: tendencijos_duomenys,
   ‘strategija’: strategija_pagal_kuria_prekiauta
}
```

Kuomet pirmame žingsnyje aprašytas telkinys pasiekia pakankamą dydį (šioje vietoje dydį nusistato pats naudotojas), tai tendencijas prisimenantis mechanizmas yra pasiruošęs veikti ir pereiti į antrą žingsnį – *išrinkimą.* Šio žingsnio tikslas yra surasti, kuo panašesnius pavyzdžius iš telkinio, kad vėliau pagal juos būtų galima atlikinėti prekybos veiksmus. Išrinkimo žingsnyje reikėtų ir naudotojo įsikišimo norint gauti, kuo tikslesnius rezultatus. Naudotojas šiame žingsnyje galėtų pasinaudoti Python bibliotekos „matplotilib“ galimybėmis ir nusibraižyti grafikus bei atitinkamus indeksus, kurie jam padėtų atlikti tikslesnį sprendimą.

Sekantis žingsnis – *patikrinimas,* kuriame kiekvienas pavyzdys iš sąrašo bus tikrinamas su dabartine rinkos padėtimi. Išrikiuoti sąraše nuo 0, kuris labiausiai atitinka dabartinę padėtį rinkoje, jie bus ištestuojami. Kiekviena strategija bus imama iš sąrašo, testuojama ir vėliau pagal rezultatus sąrašas bus iš naujo surikiuojamas pagal, tai, kaip kiekviena iš strategijų surinko taškų iš testo (testas ir taškų kriterijus turėtų būti parinktas pačio naudotojo). Šiame žingsnyje praverstų Python sąrašai ir jų galimybės bei veiksmai su įvairiais duomenims.

Patvirtinimo žingsnyje iš naujai surūšiuoto sąrašo geriausius rezultatus testavimo metu davusi strategija bus imama ir naudojama sprendimų priėmime realiu laiku.

4.3 Rezultatai

Šis algoritmas nebuvo realizuotas su tikrais duomenims, taip, kaip statinis ar dinaminis tendencijas sekantys algoritmai. Tačiau turint prieš tai aprašytų algoritmų bandymus ir rezultatus, galima būtų juos bandyti sujungti ir realizuoti pagal šiame skyriuje aprašytus veiksmus bei tikėtis rezultatų geresnių nei naudojant juos atskirai, bet nereiktų ir pamiršti, jog juos sujungiant šiame algoritme būtų paveldimi ne tik jų pliusai, bet ir minusai.

Rezultatus naudojant šį algoritmą galima apžvelgti tik iš [FTP12] darbo, kuriame jis buvo lygintas kartu su statiniais ir dinaminiais algoritmais. Atsižvelgiant į eksperimento rezultatus tendencijas prisimenantis algoritmas turėjo geriausią pasirodymą su investicijų grąža, kuri siekė net virš 400%, lyginant su statinio ir dinaminio 19% bei 26%.

5. Tendencijas sekančio algoritmo optimizavimas pasitelkus dalelių spiečių

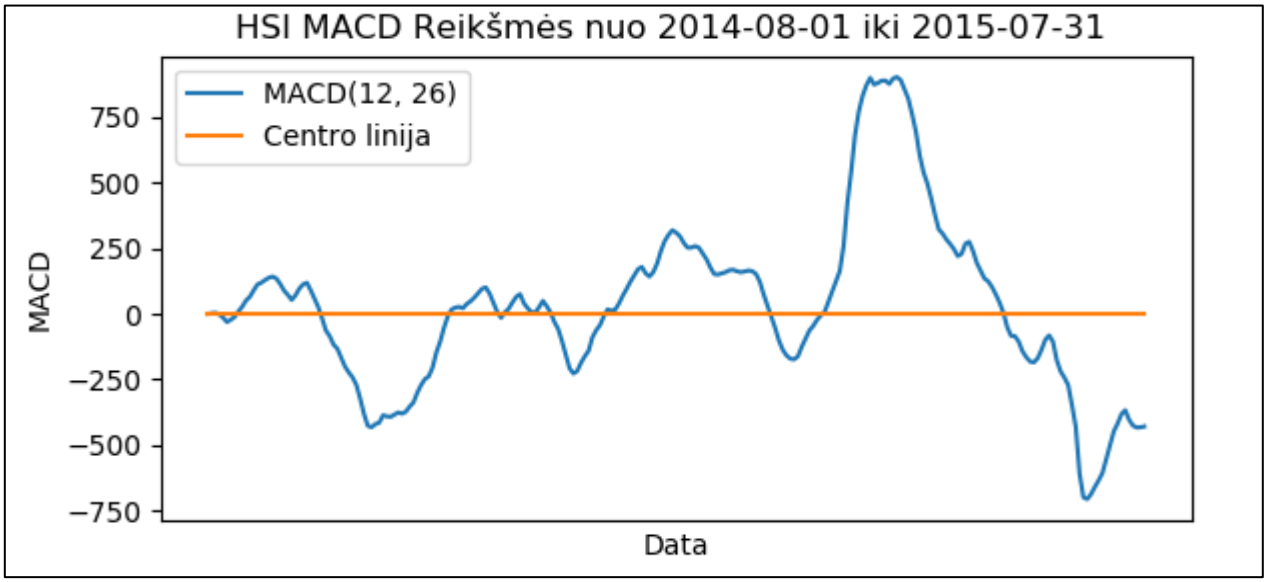
Šiame skyriuje bus siekiama pasinaudoti dalelių spiečiaus optimizavimo metodu ir atrasti prieš tai aprašytas P ir Q reikšmes bei jas gavus pritaikyti tendencijas sekančiame algoritme ir taip siekti didžiausios investicijų grąžos.

5.1 Apie algoritmą

Šiame algoritme be dalelių spiečiaus optimizacijos bus dar reikalingas MACD indikatorius. MACD yra tendencijas sekantis pagreičio indikatorius, kuris parodo sąryšį tarp dviejų slenkančių vidurkių. Tie du slenkantys vidurkiai būna 12 ir 26 dienų periodo ir indikatorius yra apskaičiuojamas juos abu atėmus:

$$MACD(t) = ESV_{12}(t) - ESV_{26}(t)$$

Pritaikius MACD indikatorių HSI indeksui (duomenys yra nuo 2014-08-01 iki 2015-07-31) yra gaunamas toks grafikas:



5 pav. HSI indekso MACD reikšmės kartu su centro linija

Remiantis MACD indikatoriumi yra lengva nustatyti tendencijas. Teigiamos MACD reikšmės atitinka didėjančią (aukštyn) tendenciją, o neigiamos atitinka mažėjančią (žemyn) tendenciją [LSZ+18]. Naudojantis MACD indikatoriumi taip pat galima nustatyti ir tendencijos stiprumą. Didelės teigiamos MACD reikšmės atitinka stiprią didėjančią tendenciją, o mažos neigiamos reikšmės atitinka stiprią mažėjančią tendenciją. Norint tiksliai nustatyti stiprias ir silpnas tendencijas bus naudojamos P ir Q ribos, kurios bus randamos pritaikius dalelių spiečiaus optimizavimo metodą. Jei MACD indikatoriaus reikšmė kils ir kirs P ribą, tai reikš, kad dabartinę tendenciją galima laikyti stipria

didėjančia, jei MACD indikatoriaus reikšmė kris ir kirs Q ribą, tai šiuo atveju tokią tendenciją galima laikyti stipria mažėjančia. Visais kitais atvejais tendencija bus arba silpna didėjanti arba silpna mažėjanti.

Pasinaudojus dalelių spiečiaus optimizavimo metodu ir radus optimalias P ir Q reikšmes, bus naudojami tokios taisyklės generuoti pirkimo ir pardavimo signalus [LSZ+18]:

- Pirkti akciją, kuomet MACD indikatorius kirto P ribą, kadangi ši tendencija buvo identifikuota, kaip stipri ir didėjanti.
- Parduoti akciją, kuomet MACD indikatorius kirto Q ribą, kadangi ši tendencija buvo identifikuota, kaip stipri ir mažėjanti.

Pačio algoritmo pseudokodas yra išreiškiamas taip [LSZ+18]:

```
While ne simulacijos pabaiga do
    Apskaičiuoti MACD(t)
    If MACD(t-1) < P && MACD(t) > P && nėra atidarytos pozicijos then
        Atidaryti poziciją
    End
    If MACD(t-1) > Q && MACD(t) < Q && yra atidaryta pozicija then
        Uždaryti poziciją
    End
End
If simuliacijos pabaiga && yra atidaryta pozicija then
        Uždaryti poziciją
End
```

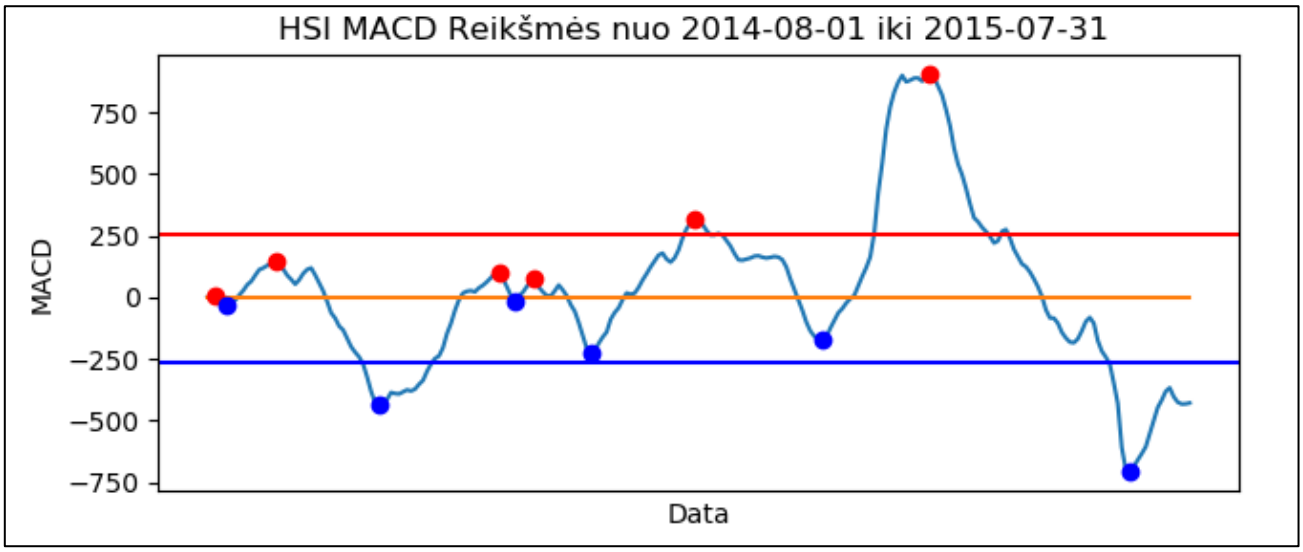
5.2 Realizacija

Kiekvienoje didėjančioje ar mažėjančioje tendencijoje galima rasti maksimalią ir minimalią MACD reikšmę. Maksimali didėjančios tendencijos i MACD indikatoriaus reikšmė yra žymima MD_i, o minimali mažėjančios tendencijos j reikšmė yra žymima MM_j(MM_j visada yra neigiamas skaičius). Kiekvienos tendencijos vidurkio linijos (DTVL – didėjančios tendencijos vidurkio linija, o MTVL – mažėjančios tendencijos vidurkio linija) yra apskaičiuojamos formulėmis:

$$DTVL = \frac{MD_1 + MD_2 + \cdots + MD_n}{n}$$

$$MTVL = \frac{MM_1 + MM_2 + \cdots + MM_n}{n}$$

Suradus MD_i ir MM_j bei apskaičiavus vidurkio linijas didėjančioms ir mažėjančioms tendencijoms galima jas pritaikyti braižant grafikus MACD indikatoriui HSI indeksui:



6pav.Didėjančios ir mažėjančios tendencijos vidurkio linijos, kartu su MACD indikatoriumi HSI indeksui

Raudoni taškai šiame grafike atitinka vienos didėjančios tendencijos MD reikšmę, o mėlyni mažėjančios MM reikšmę. Virš 0 (centro linijos) einanti raudona tiesė atitinka didėjančios tendencijos vidurkio liniją (DTVL), o žemiau 0 einanti mėlyna tiesė atitinka mažėjančios tendencijos vidurkio liniją (MTVL).

Turint šias reikšmes galima jau ir apibrėžti paieškos erdvę riboms P ir Q. [LSZ+18] yra siūloma paieškos erdvę ribai P apibrėžti, kaip [0, $DTVL$], paieškos erdvę ribai Q, kaip [$MTVL$, 0]. Visi kiti parametrai ieškant optimalių P ir Q reikšmių yra apibrėžiami taip:

- Dalelių spiečiaus populiacija yra 30
- Inercijos faktorius ω yra 0.9
- ϕ_1 ir ϕ_2 yra 2
- R_1 ir R_2 yra atsitiktiniai skaičiai paimti iš aibės [0, 1]
- Dalelių greitis paieškos erdvėje P yra atsitiktinis skaičius imamas iš aibės [$-DTVL * 0.2$, $DTVL * 0.2$]

- Dalelių greitis paieškos erdvėje Q yra atsitiktinis skaičius imamas iš aibės [$MTVL * 0.2$, $-MTVL * 0.2$]
- Simuliacija yra nutraukiama jei $pbest$ (dalelės geriausia) reikšmė nepasikeičia per 2000 iteracijų
- Tinkamumo funkcijai yra naudojamas sėkmės rodiklis (SR), kuris yra apskaičiuojamas tokia formule:
- MACD yra naudojamas standartinis 12 ir 26 dienų eksponentinių slenkančių vidurkių skirtumai

Norint rasti optimalias P ir Q reikšmes toliau algoritmas atlieka tokius žingsnius:

1. Inicijuojamos pradinės dalelės. Tai reiškia, jog kiekvienai dalelei yra priskiriamas greitis ir pozicija paieškos erdvėje iš ankščiau aprašytų parametrų.
2. Kadangi dalelės jau yra inicijuotos, jos turi jau ir atsitikinai parinktas P ir Q reikšmes, todėl kiekviena dalelė yra paleidžiama sugeneruoti pirkimo ir pardavimo signalus, pagal anksčiau aprašytą pseudokodą.
3. Šiame žingsnyje visos dalelės jau yra sukūrusios pirkimo ir pardavimo signalus todėl galima apskaičiuoti kiekvienos dalelės tinkamumo reikšmę (sėkmės rodiklį).
4. Šiame žingsnyje yra tikrinama ar dalelės tinkamumo reikšmė yra geresnė negu jos $pbest$. Jei geresnė reikšmė, tai $pbest$ yra atnaujinamas, jei blogesnė, tai $pbest$ išlieka nepakitęs
5. Dabar turint visų dalelių $pbest$ reikšmes yra išrenkama geriausia ir ji yra priskiriama $gbest$.
6. Šiame žingsnyje yra perskaičiuojamas kiekvienos dalelės greitis.
7. Kadangi visų dalelių greičiai perskaičiuoti tada galima atnaujinti kiekvienos dalelės poziciją erdvėje.
8. Paskutinis žingsnis yra tikrinimui ar simuliacijos nutraukimo kriterijus pasiektas ($pbest$ reikšmė nesikeitė 2000 iteracijų), jei kriterijus nepasiektas, tai dalelės su naujomis P ir Q reikšmėmis iš naujo sugeneruoja pirkimo ir pardavimo signalus ir grįžtama į 3 žingsnį. Jei simuliacijos kriterijus yra pasiektas, tai $gbest$ reikšmės atitinka optimalias P ir Q reikšmes.

5.3 Rezultatai

[LSZ+18] siūloma norint rasti tinkamas P ir Q reikšmes, dalelių spiečiaus optimizavimo algoritmą prieš tai apmokyti su istoriniais rinkos duomenimis už 3 metus (nuo 2011-08-01 iki 2014-07-31), o vėliau jau radus optimalias reikšmes naudoti simuliacijose nuo 2014-08-01 iki 2015-07-31. Siekiant išbandyti algoritmo veikimą bei vėliau palyginti su kitais algoritmais, buvo pasirinkti tie patys duomenys, kaip ir [LSZ+18], t.y. 7 gerai žinomi akcijų indeksai – IXIC, SP500, HSI, TWII, FTSE, DJI, N225. Apmokymo metu gauti rezultatai pateikiami lentelėje:

Akcijų simbolis	Paieškos aibė ribai P	Paieškos aibė ribai Q	Optimali P reikšmė	Optimali Q reikšmė
IXIC	[0, 36.89]	[-31.42, 0]	19.55	-22.39
SP500	[0, 30.67]	[-19.36, 0]	13.25	-18.12
HSI	[0, 252.33]	[-306.74, 0]	78.43	-263.6
TWII	[0, 87.66]	[-94.24, 0]	72.58	-27.73
FTSE	[0, 37.17]	[-47.48, 0]	17.20	-12.50
DJI	[0, 99.21]	[-71.42, 0]	10.48	-67.21
N225	[0, 165.43]	[-145.72, 0]	131.64	-49.87

7 lentelė. Apmokymo metu gauti rezultatai

Apmokius algoritmą ir gavus P ir Q reikšmes bei jas pritaikius testavimo duomenų aibei buvo gauti tokie rezultatai:

Akcijų simbolis	Transakcijų skaičius	Pelningos Transakcijos	Sėkmės rodiklis	Investicijų grąža
IXIC	2	1	0.5	3.75%
SP500	2	1	0.5	0.65%
HSI	2	1	0.5	3.24%
TWII	1	0	0	-0.56%
FTSE	3	1	0.33	-5.65%
DJI	6	4	0.66	20.35%
N225	3	1	0.33	2.54%

8 lentelė. Testavimo metu gauti rezultatai

Gavus tokius rezultatus galime pamatyti, jog ne visada testavimo sesijoje yra sugeneruojamos pelningos transakcijos (pavyzdys TWII indeksas). Tokiais atvejais investicijų grąža visada bus neigiama. Taip pat galime pastebėti, jog ne visada sėkmės rodiklis turi būti netoli 1, kad būtų gaunama teigiama investicijų grąža. Tokiais atvejais vienos transakcijos pelnas atsveria net kelių transakcijų nuostolius. Geriausius rezultatus pavyko gauti su indeksu DJI, kurio optimali P reikšmė buvo 10,48, o Q buvo -67,21. Šio indekso investicijų grąža siekė net 20,35 procentus. Visų indeksų investicijų grąžos vidurkis siekė 3,47 procentus.

5.4 Algoritmo taikymas įvairaus tipo akcijoms

Prieš tai aprašyti rezultatai buvo gauti pritaikius algoritmą populiariems indeksams. Siekiant labiau ištirti algoritmo veikimą, papildomai buvo parinktos įvairaus tipo akcijos ir biržoje prekiaujantys investiciniai fondai: SPDR S&P 500 ETF (simbolis - SPY), SPDR Gold Shares (GLD), Energy Select Sector SPDR Fund (XLE), Google (GOOG), Amazon (AMZN). Šio tyrimo metu, apmokymui skirti duomenys buvo imami nuo 2014-01-01 iki 2018-01-01, o testavimui skirti duomenys nuo 2018-01-01 iki 2019-01-01. Apmokymo metu gauti rezultatai (visi kiti algoritmo parametrai liko nepakitę):

Akcijų simbolis	Paieškos aibė ribai P	Paieškos aibė ribai Q	Optimali P reikšmė	Optimali Q reikšmė
SPY	[0, 1.68]	[-1.22, 0]	0.98	-1.06
GLD	[0, 1.06]	[-1.19, 0]	0.55	-0.41
XLE	[0, 0.9]	[-1.39, 0]	0.69	-0.007
GOOG	[0, 9.52]	[-6.04, 0]	3.31	-1.12
AMZN	[0, 12.78]	[-8.003, 0]	11.58	-3.97

9 lentelė. Apmokymo metu gauti rezultatai įvairioms akcijoms

Testavimo metu gauti rezultatai:

Akcijų simbolis	Transakcijų skaičius	Pelningos transakcijos	Sėkmės rodiklis	Investicijų grąža
SPY	3	1	0.33	-10.41%
GLD	1	0	0	-3.94%
XLE	3	1	0.33	-10.97%

GOOG	4	1	0.25	-11.18%
AMZN	2	2	1	28.11%

10 lentelė. Testavimo metu gauti rezultatai įvairioms akcijoms

Iš šių rezultatų matyti, jog algoritmas davė prastus rezultatus iš biržoje prekiaujančių investicinių fondų bei neigiama investicijų grąža buvo gaunama ir su Google akcijomis. Tačiau, su Amazon akcijomis buvo gauti daug geresni rezultatai ir investicijų grąža siekė net 28,11 procentus. Neigiama investicijų grąža galimai buvo gauta dėl to, jog šis algoritmas naudojasi tik stipriomis tendencijomis aukštyn, o silpnas tendencijas aukštyn ignoruoja, kurios ir galėjo dominuoti SPY, GLD, XLE ir GOOG akcijose. Viso šio bandymo investicijų grąžos vidurkis siekė -1,678 procentus.

5.5 Dalelių spiečiaus parametrų tyrimas

Visuose prieš tai atliktuose bandymuose simuliacija būdavo pradedama su 30 dalelių. Papildomai buvo ištirta, kaip algoritmas veikia su 3 ir 300 dalelių. Naudojant 3 ir 300 dalelių nebuvo gauti geresni rezultatai. Tačiau buvo galima pastebėti, jog naudojant 300 dalelių dažnai galutinė $gbest$ reikšmė jau egzistuodavo tarp pradinių inicializuotų dalelių.

Kitas parametras, kuris buvo papildomai ištirtas, tai inercijos faktorius ω. Inercijos faktorius yra labai svarbus parametras šiame algoritme, nes jis padeda dalelėms lengviau žvalgytis įvairių sprendinių ir naudotis geriausias lokaliais ir globaliais sprendiniais. Inercijos faktoriaus parinkimo būdų yra daug [BSS+11], tačiau šiame darbe buvo naudojamas tik konstantos tipo inercinis faktorius. Pritaikius $\omega = 0.2$ ir $\omega = 0.7$ HSI indeksui (kiti algoritmo parametrai išliko nepakitę), buvo gauti tokie rezultatai:

ω reikšmės	Optimali P reikšmė	Optimali Q reikšmė	Investicijų grąža
0,2	121.76	-264.92	2.58%
0,7	92.46	-265.16	2.74%

11 lentelė. ω reikšmių testavimo rezultatai

Palyginus šiuos rezultatus su pradiniu bandymu, kuomet inercijos faktorius buvo lygus 0.9, galima pamatyti, jog investicijų grąža su žemesnėmis inercijos faktoriaus reikšmėmis yra žemesnė.

Paskutinis parametras, kurį galima būtų apžvelgti ir ištirti, tai simuliacijos nutraukimo kriterijus. [LSZ+18] siūlo simuliaciją nutraukti, kai $pbest$ reikšmė nesikeičia 2000 iteracijų. Čia papildomai galime pridėti tikrinimą ar $gbest$ reikšmės sėkmės rodiklis (SR) yra lygus 1 (t.y. 100%). Tokiais

atvejais, jeigu pavyko greičiau nei per 2000 iteracijų rasti tokias P ir Q reikšmes, su kuriomis gaunamas sėkmės rodiklis lygus 1, mums jau toliau neverta atlikinėti kitų algoritmo veiksmų bei ieškoti naujų $pbest$ reikšmių, nes kad ir kokia būtų naujai atrasta $pbest$ reikšmė, jos sėkmės rodiklis nebus didesnis už 1. Taip atmetus nereikalingus skaičiavimus būtų sutaupoma algoritmo veikimo laiko.

5.6 Palyginimas su statiniu ir dinaminiu

Apžvelgus visus bandymų rezultatus, geriausią vidutinę investicijų grąžą pavyko gauti su statiniu algoritmu, kai P ir Q reikšmės buvo lygios 10. Palyginus rezultatus su statiniu, galima pamatyti, jog statinis algoritmas gavo tokią didelė investicijų grąžos procentą, tik dėl gerų rezultatų su DJI ir N225 indeksais, kai P ir Q reikšmės buvo lygios 10. Tačiau tiriant algoritmo veikimą su kitais indeksais, dažniausiai investicijų grąža būdavo neigiama, priešingai nei šio algoritmo, kur investicijų grąža neigiama buvo tik su TWII ir FTSE indeksais, bet ir tuomet tai siekė nedidelius procentus. Jei lygintume rezultatus šio algoritmo ir dinaminio, tai galima pamatyti, jog dinaminis gavo didesnes investicijų grąžas tik su SP500 ir N225 indeksais. Visos kitos investicijų grąžos buvo mažesnės, o ir bendras investicijų grąžos vidurkis buvo mažesnis. Šis algoritmas naudojant ilgesnį laiką duotu geresnius rezultatus, nei statinis ar dinaminis. P ir Q reikšmės jau nėra bandomos nuspėti, kaip statiniame algoritme, bet yra pasinaudojama dalelių spiečiaus optimizavimo metodu senesniems rinkos duomenims ir taip gaunamos optimalios reikšmės tolimesnei prekybai. Turint didesnį duomenų kiekį ir naudojant dažniau šį algoritmą P ir Q reikšmės vis iš naujo prisitaikytų prie rinkos, ko negalima būtų pasakyti apie statinį ar dinaminį algoritmą.

Išvados

Tendencijų sekimas yra viena dažniausiai naudojamų strategijų, kuomet siekiama prekiauti akcijomis ar kitais vertybiniais popieriais. Jos paprastumas leidžia naudotojui naudotis tendencijų bangomis, pasitelkus paprastą techninę analizę nustatyti, kada pirkti ar parduoti.

Iš pradžių šiame darbe buvo apžvelgta, kas yra tendencijas sekančios sistemos ir kokius svarbiausius uždavinius jos sprendžia. Buvo pristatyti keli techniniai indikatoriai, kurie yra dažnai naudojami kuriant tokias sistemas. Taip pat buvo pristatytas dalelių spiečiaus optimizavimo metodas, kurio vėliau pasinaudojus realizuotas vienas tendencijas sekantis algoritmas. Vėliau svarbiausioje šio darbo dalyje buvo aptarti keturi algoritmai, bet savarankiškai realizuoti ir ištirti buvo tik trys iš keturių. Pirmasis tendencijas sekantis algoritmas, kaip ir galima buvo tikėtis davė rezultatus priklausančius nuo daugiau ar mažiau sėkmės. Nuspėjus teisingai tendencijos ilgį galima, kuo greičiau atidaryti pirkimo ar pardavimo pozicijas rinkoje ir taip siekti didesnės investicijų grąžos. Antrajame algoritme jau buvo remiamasi daugiau technine analize ir siekiama sistemai suteikti, kuo daugiau dinamiškumo ir autonomijos. Trečiajame algoritme nebuvo pateikta realizacija ar savarankiškas tyrimas, tačiau išanalizavus literatūrą galima buvo nuspręsti, jog šio algoritmo veikimas ir rezultatai turėtų būti daug geresni, nei prieš tai dviejų aptartų. Šio algoritmo realizaciją ir detalesnį tyrimą galima būtų palikti tolimesniems darbams. Ketvirtas ir paskutinis tendencijas sekantis algoritmas buvo optimizuotas pasitelkus dalelių spiečiaus metodą. Šis algoritmas buvo ištirtas su įvairiomis akcijomis, siekiant apžvelgti jo veikimą ne tik su prieš tai naudotais duomenimis. Papildomai buvo ištirta kaip dalelių spiečiaus parametrai gali paveikti algoritmo veikimą bei buvo pasiūlytas patobulinimas, kuris turėtų sumažinti iteracijų skaičių keliais atvejais. Rezultatai šio algoritmo buvo geresni nei dinaminio ir dažnai investicijų grąža būdavo teigiama, ko negalima buvo pasakyti tiriant statinio algoritmo veikimą.

Visgi prieš pradedant taikyti šiuos algoritmus realiose situacijose ir rinkose reikia atsiminti, jog rinkos pokyčiai dažnai yra nenuspėjami, todėl, kad ir kokie šie algoritmai būtų efektyvūs ir pelningi testavimo metu, nereiškia, jog jie taip pat elgsis ir rinkoje. Taigi, nors ir pristatyti algoritmai įrodo, jog egzistuoja tendencijos bei jie jomis gali ir moka pasinaudoti, visgi to nepakanka norint sukurti pelningą ir pilnai automatizuotą sistema, nes tai reikalauja patirties, gilesnio testavimo, papildomos informacijos iš įvairių šaltinių bei dažnai didesnės sėkmės.

Summary

In this thesis, algorithmic trading is described and how to use trend following strategies or algorithms to create automated trading systems that generate transactions without human input. Additionally some technical indicators are reviewed that can be implemented in other trading systems and strategies. This paper reviews and analyzes 4 trend following algorithms. 3 of the 4 algorithms are implemented using the „Python“ programming language and are tested against real historic market data. One of the trend following algorithms uses „Particle Swarm Optimization“ method to achieve better results in some cases than other researched algorithms. The paper compares the results of the algorithms that were implemented using the „Python“ programming language.

Literatūra

[WDD09] Feng Wang, Keren Dong, Xiaotie Deng. Algorithmic trading system: design and applications. *Frontiers of Computer Science*, pp. 235-246, 2009

[NMT+11] Giuseppe Nuti, Mahnoosh Mirghaemi, Philip Treleaven, Chaiyakorn Yingsaeree. Algorithmic Trading. *Computer*, pp. 61-69, 2011

[Gra63] Joseph Granville. Granville‘s New Key to Stock Market Profits. 1963

[HOP17] Brian Hurst, Yao Hua Ooi, Lasse Heje Pedersen. A Century of Evidence on Trend-Following Investing. *The Journal of Portfolio Management*, pp. 15-29, 2017

[Jam03] Jessica James. Simple trend-following strategies in currency trading. *Quantitive Finance*, pp. 75-77, 2003

[MLW17] Vitaliy Milke, Christina Luca, George Wilson. Minimisation of parameters for optimization of Algorithmic Trading Systems. 2017

[DFV+08] M.A.Dashti, Yaghoub Farjami, A.Vedadi, Mahammad Anisseh. Implementation of particle swarm optimization in construction of optimal risky portfolios. 2008

[JKM05] Kyong Joo Oh, Tae Yoon Kim, Sungky Min. Using genetic algorithm to support portfolio optimization for index fund management. *Expert Systems with Applications*, pp. 371-379, 2005

[WNK+17] Chukiat Worasuscheep, Sirapop Nuannimnoi, Ratchanon Khamvichit, Papon Attagonwantana. An Automatic Stock Trading System using Particle Swarm Optimization. 2017

[WYC14] Fei Wang, Philip L.H. Yu, David W. Cheung. Combining technical trading rules using particle swarm optimization. *Expert Systems with Applications*, pp. 3016-3026, 2014

[JE95] James Kennedy, Russell Eberhart. Particle Swarm Optimization. 1995

[FST12] Simon Fong, Yain-Whar Si, Jackie Tai. Trend following algorithms in automated derivatives market trading. *Expert Systems with Applications*, pp. 11378-11390, 2012

[FTP12] Simon Fong, Jackie Tai, Pit Pichappan. Trend recalling algorithm for automated online trading in stock market. *Journal of Emerging Technologies in Web Intelligence*, pp. 240-251, 2012

[LSZ+18] Jingyuan Liu, Yain-Whar Si, Defu Zhang, Ligang Zhou. Trend following in financial time series with multi-objective optimization. *Applied Soft Computing*, 2018

[BSS+11] J.C.Bansal, P.K.Singh, Mukesh Saraswat, Abhishek Verma, Shimpi Singh Jadon, Ajith Abraham. Inertia Weight Strategies in Particle Swarm Optimization. 2011