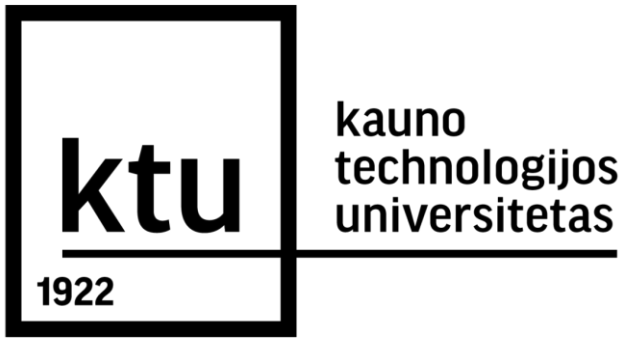
KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS



Intelektikos pagrindai (P176B101)

Laboratorinis darbas Nr.2

Atliko:
IFF-9/8 gr. studentas
Lukas Navašinskas
2022 m. balandžio 5 d.
Priėmė:
lekt. Nečiūnas Audrius
doc. Paulauskaitė-Tarasevičienė Agnė

KAUNAS 2022

Turinys

1. Užduotis

Sukurkite sprendimo priėmimo sistemą remiantis miglotosios logikos teorija (rekomenduojama taikant Mamdani algoritmą, tačiau gali būti naudojamas ir Sugeno modelis). Duomenys gali būti naudojami realūs, iš atvirų šaltinių arba sugalvoti jūsų pačių (dažniausiai studentai sugalvoja savo duomenis ir patiems aktualią problemą – t.y. jūs tampate ekspertais). Sistemos programinė realizacija turi būti atlikta naudojant Python (arba C šeimos kalbomis).

1.1. Reikalavimai

- Aiškus užduoties aprašas, t.y., koks uždavinys, pagal kokius duomenis ką reikia paskaičiuoti. Aprašomi kintamųjų matmenys, jie sugalvoti ar paimti iš išorinių šaltinių ir pan.;
- Sistemos įvesčių kiekis ir fuzzy aibių skaičius: nuo 3 × 3 iki 4 × 4;
- Sistemos išvesčių kiekis ir fuzzy aibių skaičius: nuo 1 × 3 iki 2 × 3;
- Suformuotos ir pateiktos logiškos taisyklės naudojant du/tris skirtingus loginius kintamuosius (And, Or, Not).Visos taisyklės turi būti pateiktos ataskaitoje.
- Pateikti metodai panaudoti implikacijai, agregacijai ir defuzifikacijai. Defuzifikacijai reikia panaudoti du skirtingus atsakymo skaičiavimo metodus: Centroid ir MOM(arba LOM).
- Sudarius modelį reikia pateikti 3 testinių įvesčių reikšmių scenarijus ir gautus atsakymų rezultatus.

1.2. Užduoties aprašas

Sukurkite fuzzy sistemos modelį, kuris apskaičiuotų mašinų avarijos tikimybę įvertinus oro temperatūrą, greitį, atstumą tarp automobilių.

1.3. Įėjimo kintamieji

- Oro temperatūra (C^o): [-10;30]; fuzzy aibės: šalta, šilta, karšta[;
- Automobilio greitis (km/h): [50;130]; fuzzy aibės: mažas, vidutinis, didelis;
- Atstumas tarp automobilių (m): [1;150]; fuzzy aibės: mažas, vidutinis, didelis;

1.4. Išėjimo kintamieji

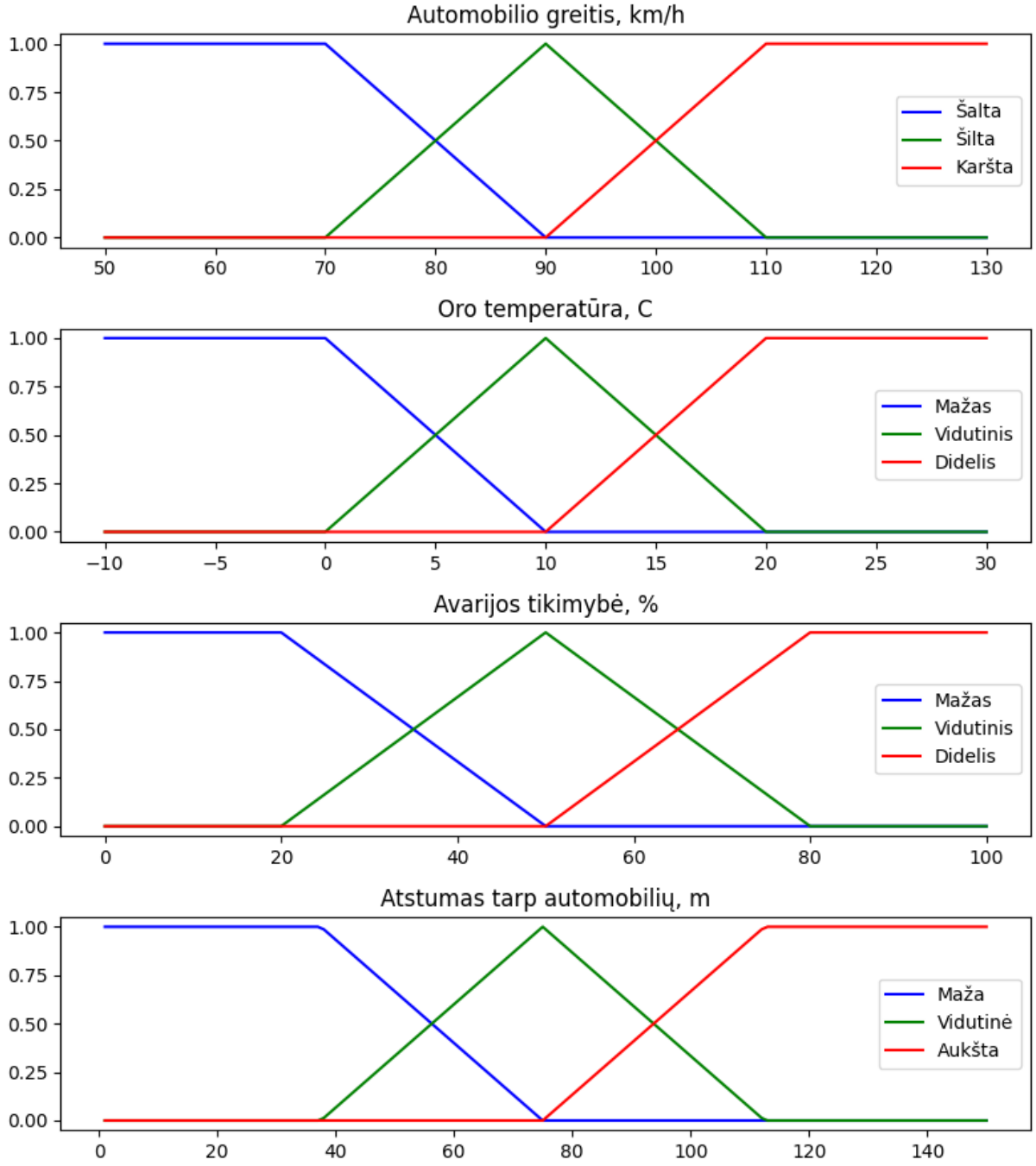
- Avarijos tikimybė (%): [0;100]; fuzzy aibės: žema, vidutinė, aukšta

1.5. Fuzzy taisyklės

Nr.	Oro temperatūra		Atstumas tarp automobilių		Automobilio greitis		Avarijos tikimybė
1	Šalta	or	Mažas	and	Didelis	then	Aukšta
2	Šalta	or	Vidutinis	and	Didelis	then	Aukšta
3	Šalta	or	Didelis	and	Didelis	then	Aukšta
4	Šilta	or	Mažas	and	Didelis	then	Aukšta
5	Karšta	or	Mažas	and	Didelis	then	Aukšta
6	Šilta	or	Vidutinis	and	Didelis	then	Vidutinė
7	Šilta	or	Didelis	and	Didelis	then	Vidutinė
8	Karšta	or	Vidutinis	and	Didelis	then	Vidutinė
9	Karšta	or	Didelis	and	Didelis	then	Vidutinė
10	Šalta	or	Mažas	and	Mažas	then	Vidutinė
11	Šalta	or	Vidutinis	and	Mažas	then	Vidutinė
12	Šilta	or	Mažas	and	Mažas	then	Vidutinė
13	Šilta	or	Vidutinis	and	Mažas	then	Vidutinė
14	Karšta	or	Mažas	and	Mažas	then	Maža
15	Karšta	or	Vidutinis	and	Mažas	then	Maža
16	Karšta	or	Vidutinis	and	Vidutinis	then	Maža
17	Karšta	or	Vidutinis	and	Mažas	then	Maža
18	Šalta	or	Didelis	and	Mažas	then	Maža
19	Šilta	or	Didelis	and	Mažas	then	Maža
20	Šilta	or	Didelis	and	Vidutinis	then	Maža
21	Karšta	or	Didelis	and	Vidutinis	then	Maža

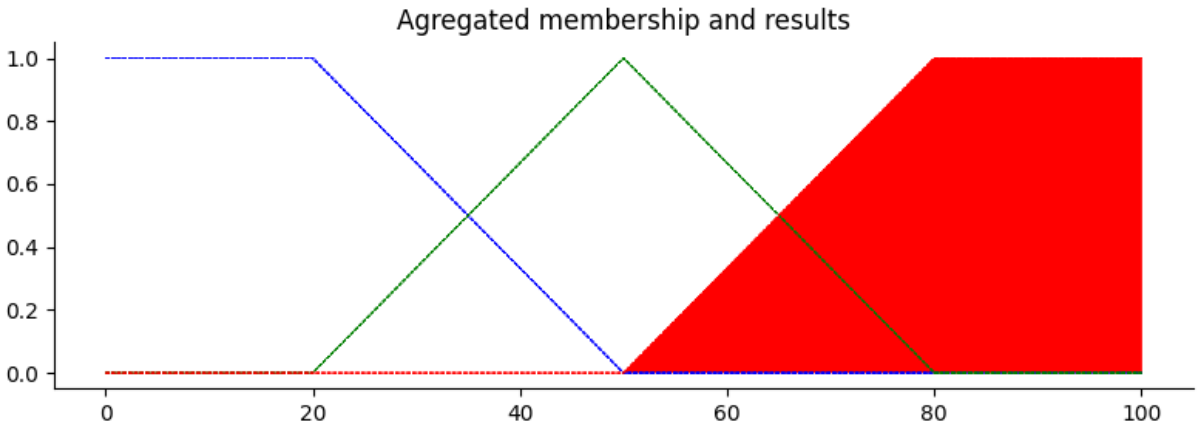
2. Rezultatai

2.1. Įėjimo reikšmių atvaizdavimas

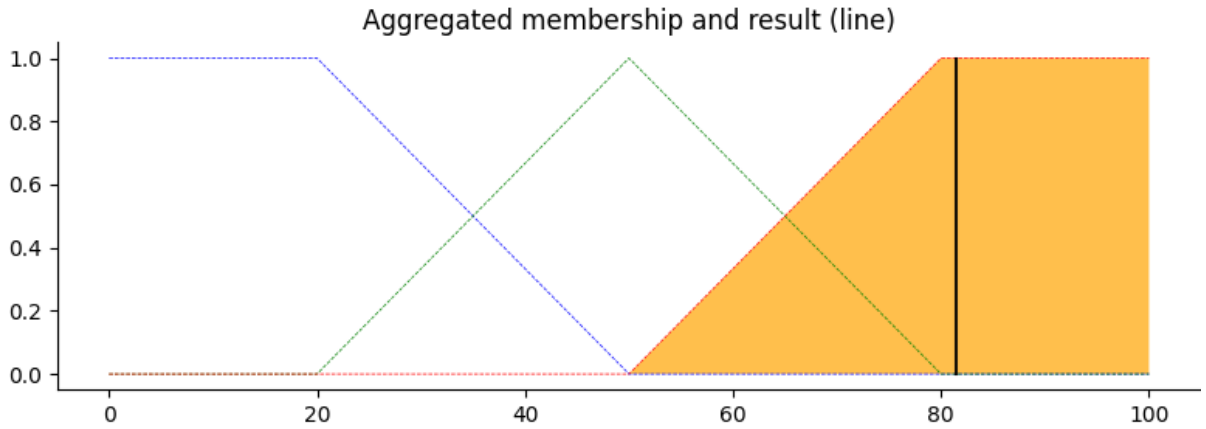


pav. 1 Įėjimo reikšmių grafikai

2.2. Taisyklių pritaikymo grafikai



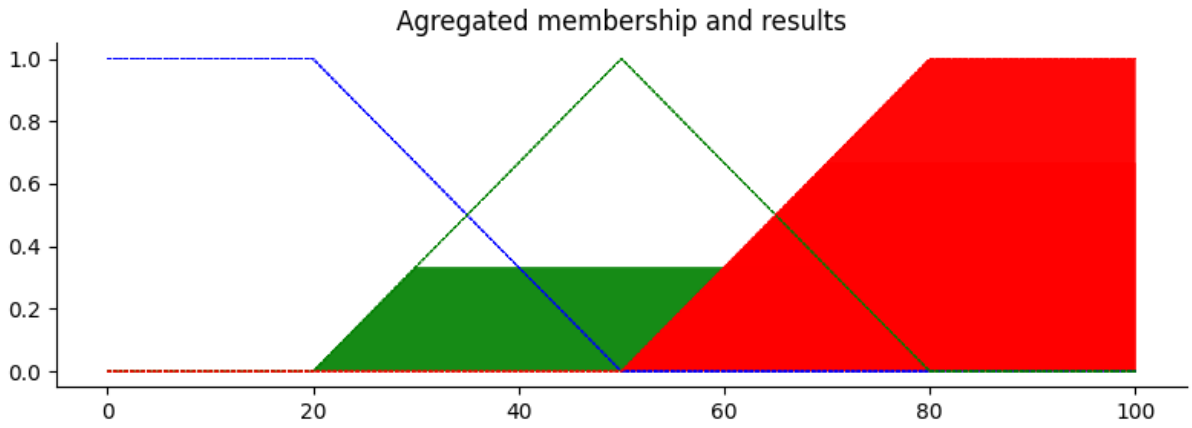
pav. 2 Pritaikytų taisyklių, kai oro temperatūra -10C, greitis 130km/h, atstumas tarp automobilių 5m



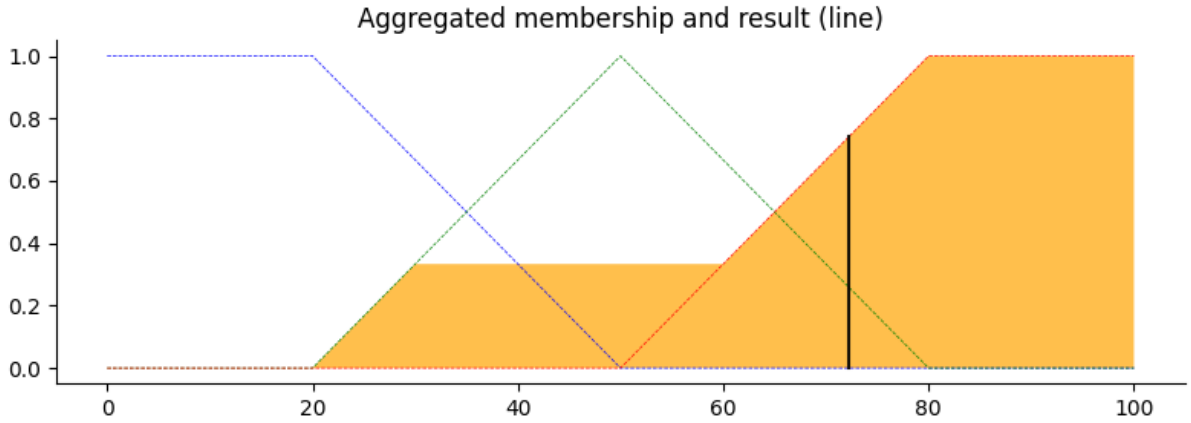
pav. 3 Agregacijos grafikas, kai oro temperatūra -10C, greitis 130km/h, atstumas tarp automobilių 5m

Avarijos tikimybė = 81.42857142857143 %, Svorio centras

Avarijos tikimybė = 90.0 %, Maksimumo vidurkis



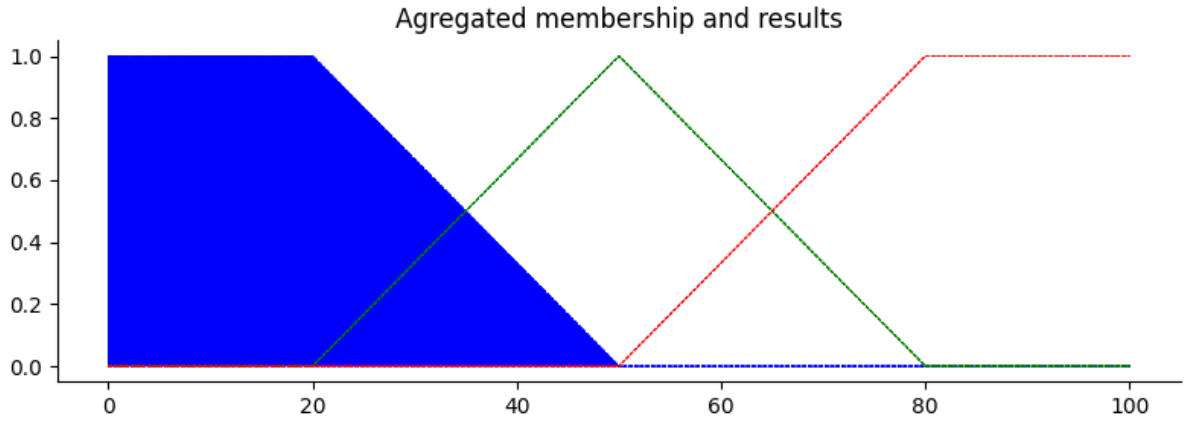
pav. 4 Pritaikytų taisyklių grafikas, kai oro temperatūra 0C, greitis 130km/h, atstumas tarp automobilių 50m



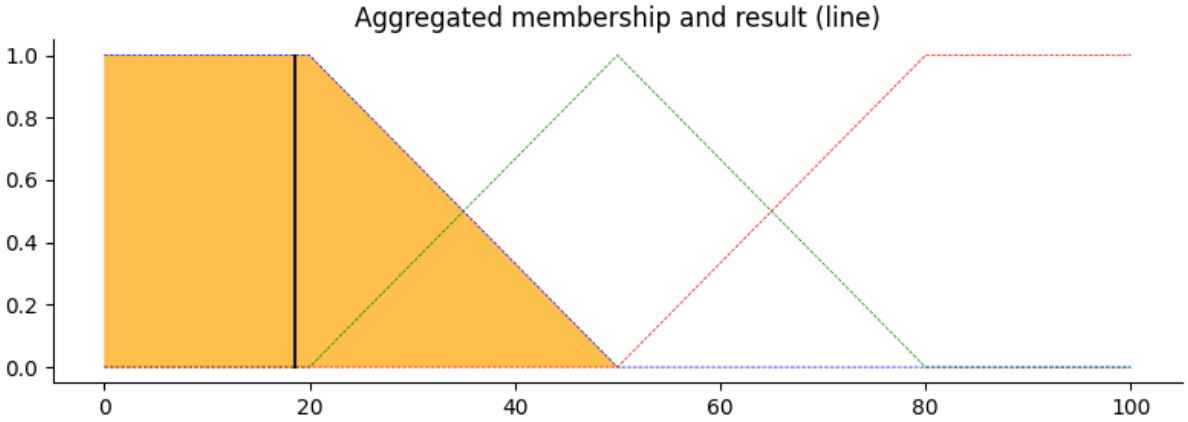
pav. 5 Agregacijos grafikas, kai oro temperatūra 0C, greitis 130km/h, atstumas tarp automobilių 50m

Avarijos tikimybė = 72.22222222222223 %, Svorio centras

Avarijos tikimybė = 90.0 %, Maksimumo vidurkis



pav. 6 Pritaikytų taisyklių grafikas, kai oro temperatūra 30C, greitis 50km/h, atstumas tarp automobilių 150m



pav. 7 Agregacijos grafikas, kai oro temperatūra 30C, greitis 50km/h, atstumas tarp automobilių 150m

Avarijos tikimybė = 18.57142857142857 %, Svorio centras

Avarijos tikimybė = 10.0 %, Maksimumo vidurkis

2.3. Atsakymai

Oro temperatūra, C	Automobilio greitis, km/h	Atstumas tarp automobilių	Avarijos tikimybė, % (Svorio centras)	Avarijos tikimybė, % (Svorio centras)
-10	130	5	81. 429	90
0	130	50	72.222	90
30	50	150	18.571	10

3. Programos kodas

Main.py

```
import numpy as np
```

```
import skfuzzy as fuzz #pip install scikit-fuzzy
import matplotlib.pyplot as plt


def turn_off_top_right_axes(ax0, ax1, ax2):
    for ax in (ax0, ax1, ax2):
        ax.spines['top'].set_visible(False)
        ax.spines['right'].set_visible(False)
        ax.get_xaxis().tick_bottom()
        ax.get_yaxis().tick_left()


def plot_input_graphs(x_carSpeed, x_weather, x_crashProb,
x_distanceBetweenCars, carSpeed_lo, carSpeed_md, carSpeed_hi,
weather_lo, weather_md, weather_hi, crashProb_lo,
                      crashProb_md, crashProb_hi,
distanceBetweenCars_lo, distanceBetweenCars_md,
distanceBetweenCars_hi ):
    fig, (ax0, ax1, ax2, ax3) = plt.subplots(nrows=4, figsize=(8, 9))

    ax0.plot(x_carSpeed, carSpeed_lo, 'b', linewidth=1.5,
label='Šalta')
    ax0.plot(x_carSpeed, carSpeed_md, 'g', linewidth=1.5,
label='Šilta')
    ax0.plot(x_carSpeed, carSpeed_hi, 'r', linewidth=1.5,
label='Karšta')
    ax0.set_title('Automobilio greitis, km/h')
    ax0.legend()

    ax1.plot(x_weather, weather_lo, 'b', linewidth=1.5,
label='Mažas')
    ax1.plot(x_weather, weather_md, 'g', linewidth=1.5,
label='Vidutinis')
    ax1.plot(x_weather, weather_hi, 'r', linewidth=1.5,
label='Didelis')
    ax1.set_title('Oro temperatūra, C')
    ax1.legend()

    ax2.plot(x_crashProb, crashProb_lo, 'b', linewidth=1.5,
label='Mažas')
    ax2.plot(x_crashProb, crashProb_md, 'g', linewidth=1.5,
label='Vidutinis')
    ax2.plot(x_crashProb, crashProb_hi, 'r', linewidth=1.5,
label='Didelis')
    ax2.set_title('Avarijos tikimybė, %')
    ax2.legend()

    ax3.plot(x_distanceBetweenCars, distanceBetweenCars_lo, 'b',
linewidth=1.5, label='Maža')
    ax3.plot(x_distanceBetweenCars, distanceBetweenCars_md, 'g',
linewidth=1.5, label='Vidutinė')
```

```
    ax3.plot(x_distanceBetweenCars, distanceBetweenCars_hi, 'r',
linewidth=1.5, label='Aukšta')
    ax3.set_title('Atstumas tarp automobilių, m')
    ax3.legend()
    plt.tight_layout()
    plt.show()


def plot_applied_rules_graphs(x_crashProb, crashProb0, crashProb_lo,
crashProb_md, crashProb_hi,
                              crashProb_activation_lo1,
crashProb_activation_lo2, crashProb_activation_lo3,
crashProb_activation_lo4, crashProb_activation_lo5,
crashProb_activation_lo6, crashProb_activation_lo7,
crashProb_activation_lo8,
                              crashProb_activation_md1,
crashProb_activation_md2, crashProb_activation_md3,
crashProb_activation_md4, crashProb_activation_md5,
crashProb_activation_md6, crashProb_activation_md7,
crashProb_activation_md8,
                              crashProb_activation_hi1,
crashProb_activation_hi2, crashProb_activation_hi3,
crashProb_activation_hi4, crashProb_activation_hi5):
    fig, ax0 = plt.subplots(figsize=(8, 3))

    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_lo1, facecolor='b', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_lo, 'b', linewidth=0.5,
linestyle='--', )
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_lo2, facecolor='b', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_lo, 'b', linewidth=0.5,
linestyle='--', )
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_lo3, facecolor='b', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_lo, 'b', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_lo4, facecolor='b', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_lo, 'b', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_lo5, facecolor='b', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_lo, 'b', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_lo6, facecolor='b', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_lo, 'b', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_lo7, facecolor='b', alpha=0.7)
```

```
    ax0.plot(x_crashProb, crashProb_lo, 'b', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_lo8, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_lo, 'b', linewidth=0.5,
linestyle='--')

    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_md1, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_md, 'g', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_md2, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_md, 'g', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_md3, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_md, 'g', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_md4, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_md, 'g', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_md5, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_md, 'g', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_md6, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_md, 'g', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_md7, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_md, 'g', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_md8, facecolor='g', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_md, 'g', linewidth=0.5,
linestyle='--')

    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_hi1, facecolor='r', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_hi, 'r', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_hi2, facecolor='r', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_hi, 'r', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_hi3, facecolor='r', alpha=0.7)
```

```
    ax0.plot(x_crashProb, crashProb_hi, 'r', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_hi4, facecolor='r', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_hi, 'r', linewidth=0.5,
linestyle='--')
    ax0.fill_between(x_crashProb, crashProb0,
crashProb_activation_hi5, facecolor='r', alpha=0.7)
    ax0.plot(x_crashProb, crashProb_hi, 'r', linewidth=0.5,
linestyle='--')

    ax0.set_title('Agregated membership and results')
    for ax in (ax0,):
        ax.spines['top'].set_visible(False)
        ax.spines['right'].set_visible(False)
        ax.get_xaxis().tick_bottom()
        ax.get_yaxis().tick_left()

    plt.tight_layout()
    plt.show()


def apply_rules(x_AvarijosTikimybe, automobilioGreitis_Mazas,
automobilioGreitis_Vidutinis, automobilioGreitis_Didelis,
oroTemperatura_Salta, oroTemperatura_Silta, oroTemperatura_Karsta,
                avarijosTikymybe_Maza, avarijosTikymybe_Vidutine,
avarijosTikymybe_Auksta, atstumasTarpAutomobiliu_Mazas,
atstumasTarpAutomobiliu_Vidutinis, atstumasTarpAutomobiliu_Didelis):
    #Nr 1
    active_rule1 =
np.fmin(np.fmax(oroTemperatura_Salta,atstumasTarpAutomobiliu_Mazas),a
utomobilioGreitis_Didelis)
    crashProb_activation_hi1 = np.fmin(active_rule1,
avarijosTikymybe_Auksta)

    #Nr 2
    active_rule2 =
np.fmin(np.fmax(oroTemperatura_Salta,atstumasTarpAutomobiliu_Vidutini
s),automobilioGreitis_Didelis)
    crashProb_activation_hi2 = np.fmin(active_rule2,
avarijosTikymybe_Auksta)

    #Nr 3
    active_rule3 =
np.fmin(np.fmax(oroTemperatura_Salta,atstumasTarpAutomobiliu_Didelis)
,automobilioGreitis_Didelis)
    crashProb_activation_hi3 = np.fmin(active_rule3,
avarijosTikymybe_Auksta)

    #Nr 4
```

```
    active_rule4 =
np.fmin(np.fmax(oroTemperatura_Silta,atstumasTarpAutomobiliu_Mazas),a
utomobilioGreitis_Didelis)
    crashProb_activation_hi4 = np.fmin(active_rule4,
avarijosTikymybe_Auksta)

    #Nr 5
    active_rule5 =
np.fmin(np.fmax(oroTemperatura_Karsta,atstumasTarpAutomobiliu_Mazas),
automobilioGreitis_Didelis)
    crashProb_activation_hi5 = np.fmin(active_rule5,
avarijosTikymybe_Auksta)

    #Nr 6
    active_rule6 =
np.fmin(np.fmax(oroTemperatura_Silta,atstumasTarpAutomobiliu_Vidutini
s),automobilioGreitis_Didelis)
    crashProb_activation_md1 = np.fmin(active_rule6,
avarijosTikymybe_Vidutine)

    #Nr 7
    active_rule7 =
np.fmin(np.fmax(oroTemperatura_Silta,atstumasTarpAutomobiliu_Didelis)
,automobilioGreitis_Didelis)
    crashProb_activation_md2 = np.fmin(active_rule7,
avarijosTikymybe_Vidutine)

    #Nr 8
    active_rule8 =
np.fmin(np.fmax(oroTemperatura_Karsta,atstumasTarpAutomobiliu_Vidutin
is),automobilioGreitis_Didelis)
    crashProb_activation_md3 = np.fmin(active_rule8,
avarijosTikymybe_Vidutine)

    #Nr 9
    active_rule9 =
np.fmin(np.fmax(oroTemperatura_Karsta,atstumasTarpAutomobiliu_Didelis
),automobilioGreitis_Didelis)
    crashProb_activation_md4 = np.fmin(active_rule9,
avarijosTikymybe_Vidutine)

    #Nr 10
    active_rule10 =
np.fmin(np.fmax(oroTemperatura_Salta,atstumasTarpAutomobiliu_Mazas),a
utomobilioGreitis_Mazas)
    crashProb_activation_md5 = np.fmin(active_rule10,
avarijosTikymybe_Vidutine)

    #Nr 11
```

```
    active_rule11 =
np.fmin(np.fmax(oroTemperatura_Salta,atstumasTarpAutomobiliu_Vidutini
s),automobilioGreitis_Mazas)
    crashProb_activation_md6 = np.fmin(active_rule11,
avarijosTikymybe_Vidutine)

    #Nr 12
    active_rule12 =
np.fmin(np.fmax(oroTemperatura_Silta,atstumasTarpAutomobiliu_Mazas),a
utomobilioGreitis_Mazas)
    crashProb_activation_md7 = np.fmin(active_rule12,
avarijosTikymybe_Vidutine)

    #Nr 13
    active_rule13 =
np.fmin(np.fmax(oroTemperatura_Silta,atstumasTarpAutomobiliu_Vidutini
s),automobilioGreitis_Mazas)
    crashProb_activation_md8 = np.fmin(active_rule13,
avarijosTikymybe_Vidutine)

    #Nr 14
    active_rule14 =
np.fmin(np.fmax(oroTemperatura_Karsta,atstumasTarpAutomobiliu_Mazas),
automobilioGreitis_Mazas)
    crashProb_activation_lo1 = np.fmin(active_rule14,
avarijosTikymybe_Maza)

    #Nr 15
    active_rule15 =
np.fmin(np.fmax(oroTemperatura_Karsta,atstumasTarpAutomobiliu_Vidutin
is),automobilioGreitis_Mazas)
    crashProb_activation_lo2 = np.fmin(active_rule15,
avarijosTikymybe_Maza)

    #Nr 16
    active_rule16 =
np.fmin(np.fmax(oroTemperatura_Karsta,atstumasTarpAutomobiliu_Vidutin
is),automobilioGreitis_Vidutinis)
    crashProb_activation_lo3 = np.fmin(active_rule16,
avarijosTikymybe_Maza)

    #Nr 17
    active_rule17 =
np.fmin(np.fmax(oroTemperatura_Karsta,atstumasTarpAutomobiliu_Vidutin
is),automobilioGreitis_Mazas)
    crashProb_activation_lo4 = np.fmin(active_rule17,
avarijosTikymybe_Maza)

    #Nr 18
```

```
    active_rule18 =
np.fmin(np.fmax(oroTemperatura_Salta,atstumasTarpAutomobiliu_Didelis)
,automobilioGreitis_Mazas)
    crashProb_activation_lo5 = np.fmin(active_rule18,
avarijosTikymybe_Maza)

    #Nr 19
    active_rule19 =
np.fmin(np.fmax(oroTemperatura_Silta,atstumasTarpAutomobiliu_Didelis)
,automobilioGreitis_Mazas)
    crashProb_activation_lo6 = np.fmin(active_rule19,
avarijosTikymybe_Maza)

    #Nr 20
    active_rule20 =
np.fmin(np.fmax(oroTemperatura_Silta,atstumasTarpAutomobiliu_Didelis)
,automobilioGreitis_Vidutinis)
    crashProb_activation_lo7 = np.fmin(active_rule20,
avarijosTikymybe_Maza)

    #Nr 21
    active_rule21 =
np.fmin(np.fmax(oroTemperatura_Karsta,atstumasTarpAutomobiliu_Didelis
),automobilioGreitis_Vidutinis)
    crashProb_activation_lo8 = np.fmin(active_rule21,
avarijosTikymybe_Maza)


    crashProb0 = np.zeros_like(x_AvarijosTikimybe)
    plot_applied_rules_graphs(x_AvarijosTikimybe, crashProb0,
avarijosTikymybe_Maza, avarijosTikymybe_Vidutine,
avarijosTikymybe_Auksta,
                              crashProb_activation_lo1,
crashProb_activation_lo2, crashProb_activation_lo3,
crashProb_activation_lo4, crashProb_activation_lo5,
crashProb_activation_lo6, crashProb_activation_lo7,
crashProb_activation_lo8,
                              crashProb_activation_md1,
crashProb_activation_md2, crashProb_activation_md3,
crashProb_activation_md4, crashProb_activation_md5,
crashProb_activation_md6, crashProb_activation_md7,
crashProb_activation_md8,
                              crashProb_activation_hi1,
crashProb_activation_hi2, crashProb_activation_hi3,
crashProb_activation_hi4, crashProb_activation_hi5)

    aggregated_lo = np.fmax(crashProb_activation_lo1,
                            np.fmax(crashProb_activation_lo2,
                                    np.fmax(crashProb_activation_lo3,
                                            np.fmax(crashProb_activat
ion_lo4,
```

```
                                                    np.fmax(crashProb_act
ivation_lo5,
                                                        np.fmax(crash
Prob_activation_lo6,
                                                            np.fm
ax(crashProb_activation_lo7,crashProb_activation_lo8)))))))

    aggregated_md = np.fmax(crashProb_activation_md1,
                            np.fmax(crashProb_activation_md2,
                                    np.fmax(crashProb_activation_md3,
                                            np.fmax(crashProb_activat
ion_md4,
                                                    np.fmax(crashProb_act
ivation_md5,
                                                        np.fmax(crash
Prob_activation_md6,
                                                            np.fm
ax(crashProb_activation_md7,crashProb_activation_md8)))))))
    aggregated_hi = np.fmax(crashProb_activation_hi1,
                            np.fmax(crashProb_activation_hi2,
                                    np.fmax(crashProb_activation_hi3,
                                            np.fmax(crashProb_activat
ion_hi4,crashProb_activation_hi5))))

    aggregated = np.fmax(aggregated_lo,
                         np.fmax(aggregated_md, aggregated_hi))

    crashProb = fuzz.defuzz(x_AvarijosTikimybe, aggregated,
'centroid')
    crashProb_activation = fuzz.interp_membership(x_AvarijosTikimybe,
aggregated, crashProb)  # for plot
    print("Avarijos tikimybė = "+str(fuzz.defuzz(x_AvarijosTikimybe,
aggregated, 'centroid'))+" %, Svorio centras")
    print("Avarijos tikimybė = "+str(fuzz.defuzz(x_AvarijosTikimybe,
aggregated, 'mom'))+" %, Maksimumo vidurkis")

    fig, ax0 = plt.subplots(figsize=(8, 3))

    ax0.plot(x_AvarijosTikimybe, avarijosTikymybe_Maza, 'b',
linewidth=0.5, linestyle='--', )
    ax0.plot(x_AvarijosTikimybe, avarijosTikymybe_Vidutine, 'g',
linewidth=0.5, linestyle='--')
    ax0.plot(x_AvarijosTikimybe, avarijosTikymybe_Auksta, 'r',
linewidth=0.5, linestyle='--')
    ax0.fill_between(x_AvarijosTikimybe, crashProb0, aggregated,
facecolor='Orange', alpha=0.7)
    ax0.plot([crashProb, crashProb], [0, crashProb_activation], 'k',
linewidth=1.5, alpha=0.9)
    ax0.set_title('Aggregated membership and result (line)')

    for ax in (ax0,):
```

```
        ax.spines['top'].set_visible(False)
        ax.spines['right'].set_visible(False)
        ax.get_xaxis().tick_bottom()
        ax.get_yaxis().tick_left()

    plt.tight_layout()
    plt.show()


def execute():
    # -----------DATA-------------#
    x_carSpeed = np.arange(50, 131, 1)
    x_weather = np.arange(-10, 31, 1)
    x_distanceBetweenCars = np.arange(1,151,1)
    x_crashProb = np.arange(0, 101, 1)
    # ----------------------------#

    # -------GRAPHSDATA-----------#
    carSpeed_lo = fuzz.trapmf(x_carSpeed, [50, 50, 70, 90])
    carSpeed_md = fuzz.trimf(x_carSpeed, [70, 90, 110])
    carSpeed_hi = fuzz.trapmf(x_carSpeed, [90, 110, 130, 130])

    weather_lo = fuzz.trapmf(x_weather, [-10, -10, 0, 10])
    weather_md = fuzz.trimf(x_weather, [0, 10, 20])
    weather_hi = fuzz.trapmf(x_weather, [10, 20, 30, 30])

    crashProb_lo = fuzz.trapmf(x_crashProb, [0, 0, 20, 50])
    crashProb_md = fuzz.trimf(x_crashProb, [20, 50, 80])
    crashProb_hi = fuzz.trapmf(x_crashProb, [50, 80, 100, 100])

    distanceBetweenCars_lo = fuzz.trapmf(x_distanceBetweenCars,[0, 0,
37.5, 75])
    distanceBetweenCars_md = fuzz.trimf(x_distanceBetweenCars,[37.5,
75, 112.5])
    distanceBetweenCars_hi =
fuzz.trapmf(x_distanceBetweenCars,[75,112.5,150,150])
    # ----------------------------#

    plot_input_graphs(x_carSpeed, x_weather, x_crashProb,
x_distanceBetweenCars, carSpeed_lo, carSpeed_md, carSpeed_hi,
weather_lo, weather_md,
                      weather_hi, crashProb_lo, crashProb_md,
crashProb_hi,distanceBetweenCars_lo, distanceBetweenCars_md,
distanceBetweenCars_hi)

    temp = 30
    speed = 50
    distance = 150

    carSpeed_level_lo = fuzz.interp_membership(x_carSpeed,
carSpeed_lo, speed)
```

```
    carSpeed_level_md = fuzz.interp_membership(x_carSpeed,
carSpeed_md, speed)
    carSpeed_level_hi = fuzz.interp_membership(x_carSpeed,
carSpeed_hi, speed)

    weather_level_lo = fuzz.interp_membership(x_weather, weather_lo,
temp)
    weather_level_md = fuzz.interp_membership(x_weather, weather_md,
temp)
    weather_level_hi = fuzz.interp_membership(x_weather, weather_hi,
temp)

    distanceBetweenCars_level_lo =
fuzz.interp_membership(x_distanceBetweenCars, distanceBetweenCars_lo,
distance)
    distanceBetweenCars_level_md =
fuzz.interp_membership(x_distanceBetweenCars, distanceBetweenCars_md,
distance)
    distanceBetweenCars_level_hi =
fuzz.interp_membership(x_distanceBetweenCars, distanceBetweenCars_hi,
distance)
    apply_rules(x_crashProb, carSpeed_level_lo, carSpeed_level_md,
carSpeed_level_hi, weather_level_lo, weather_level_md,
                weather_level_hi, crashProb_lo, crashProb_md,
crashProb_hi,
distanceBetweenCars_level_lo,distanceBetweenCars_level_md,
distanceBetweenCars_level_hi)


if __name__ == "__main__":
    execute()
```