МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Государственное образовательное учреждение высшего образования "Заполярный государственный университет имени Н.М.Федоровского"

Кафедра Информационных систем и технологий

**ЛАБОРАТОРНАЯ РАБОТА №2**

По дисциплине: «Нечеткая логика»

Вариант 3

Разработал студент Сидельников М.Э.

группы ИС – 21

Руководитель: Дыптан Е.А.

Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Задание**

На множестве автоматов-укладчиков задать нечеткие отношения: «Часовая производительность автомата *y* немного выше, чем автомата *x*», «Установленные мощности автоматов *x* и *y* примерно одинаковы». Проверить, являются ли указанные отношения рефлексивными, симметричными, транзитивными.

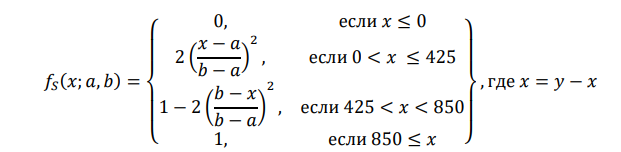
| Показатель | СМ-1030А1030А | СМС-1062 | СМ-19А | ВСКО-9 | Дорстенер 203 | «Ротомат» | Р-550 | «Атлас- Интер техник» |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Часовая производительность (наибольшая в перерасчете на одинарный кирпич) | 3600 | 3600 | 4600 | 4000 | 6000 | 5000 | 5250 | 7600 |
| Установленная мощность, кВт | 2,2 | 1,7 | 2,75 | 2,75 | 16 | 8 | 11 | 10 |

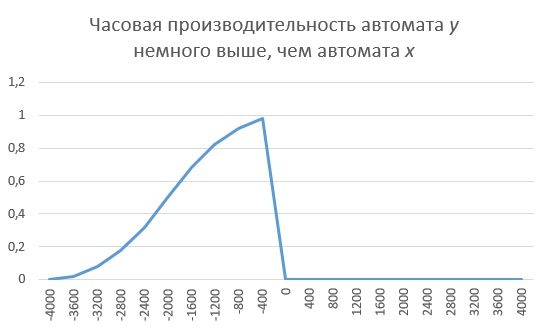
**Формирование универсального множества и нечётких подмножеств**

Универсальное множество автоматов-укладчиков состоит из 8 элементов:

U = {x1, x2, x3, x4, x5, x6, x7, x8}, где x1 - СМ-1030А1030А, x2 - СМС-1062, x3 - СМ-19А, x4 - ВСКО-9, x5 – Дорстенер 203, x6 - «Ротомат», x7 - Р-550, x8 - «Атлас-Интертехник».

Пусть нечеткое множество A (A⊂U) – задает нечеткое отношение «Часовая производительность автомата *y* немного выше, чем автомата *x*».

Функция принадлежности µА(х) является амодальной S - функцией, описываемой выражением:

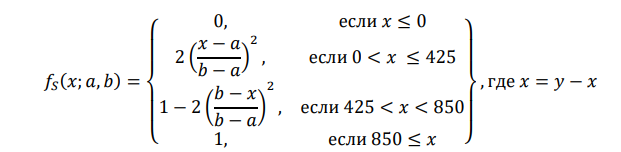


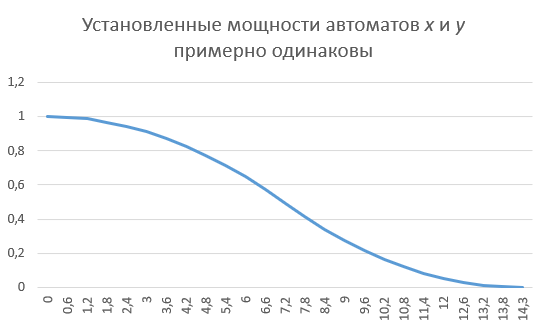
**Таблица принадлежности множества А**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 3600 | 3600 | 4600 | 4000 | 6000 | 5000 | 5250 | 7600 |
| 3600 | 0 | 0 | 0,875 | 0,98 | 0,32 | 0,755 | 0,6596875 | 0 |
| 3600 | 0 | 0 | 0,875 | 0,98 | 0,32 | 0,755 | 0,6596875 | 0 |
| 4600 | 0 | 0 | 0 | 0 | 0,755 | 0,98 | 0,9471875 | 0,125 |
| 4000 | 0 | 0 | 0,955 | 0 | 0,5 | 0,875 | 0,8046875 | 0,02 |
| 6000 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0,68 |
| 5000 | 0 | 0 | 0 | 0 | 0,875 | 0 | 0,9921875 | 0,245 |
| 5250 | 0 | 0 | 0 | 0 | 0,9296875 | 0 | 0 | 0,3403125 |
| 7600 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Пусть нечеткое множество B (B ⊂ U) – задает нечеткое отношение «Установленные мощности автоматов *x* и *y* примерно одинаковы».

Т.к. значения в матрице разности взяты по модулю, функция принадлежности µB(х) является амодальной S - функцией, описываемой выражением:





**Таблица принадлежности множества B**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 2,2 | 1,7 | 2,75 | 2,75 | 16 | 8 | 11 | 10 |
| 2,2 | 1 | 0,997554893 | 0,99704142 | 0,99704142 | 0,002445107 | 0,670986356 | 0,295857988 | 0,41322314 |
| 1,7 | 0,997554893 | 1 | 0,989217077 | 0,989217077 | 0 | 0,611814759 | 0,244510734 | 0,352095457 |
| 2,75 | 0,99704142 | 0,989217077 | 1 | 1 | 0,010782923 | 0,730426916 | 0,357988166 | 0,48611179 |
| 2,75 | 0,99704142 | 0,989217077 | 1 | 1 | 0,010782923 | 0,730426916 | 0,357988166 | 0,48611179 |
| 16 | 0,002445107 | 0 | 0,010782923 | 0,010782923 | 1 | 0,388185241 | 0,755489266 | 0,647904543 |
| 8 | 0,670986356 | 0,611814759 | 0,730426916 | 0,730426916 | 0,388185241 | 1 | 0,911976136 | 0,960878283 |
| 11 | 0,295857988 | 0,244510734 | 0,357988166 | 0,357988166 | 0,755489266 | 0,911976136 | 1 | 0,990219571 |
| 10 | 0,41322314 | 0,352095457 | 0,48611179 | 0,48611179 | 0,647904543 | 0,960878283 | 0,990219571 | 1 |

Проверка свойств нечётких отношений

1. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑥) = 1 – рефлективность
   1. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑦) ≤ 𝜇𝑅(𝑥, 𝑥) – слабая рефлективность
   2. ∀𝑥, 𝑦 ⊆ 𝑋, 𝑥 ≠ 𝑦 ∶ 𝜇𝑅(𝑥, 𝑦) < 1 – сильная рефлексивность
2. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑥) = 0 – антирефлексивность
   1. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑦) ≤ 𝜇𝑅(𝑥, 𝑥) – слабая антирефлексивность
   2. ∀𝑥, 𝑦 ⊆ 𝑋, 𝑥 ≠ 𝑦 ∶ 𝜇𝑅(𝑥, 𝑦) > 0 – сильная антирефлексивность
3. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑦) = 𝜇𝑅(𝑦, 𝑥) – симметричность
4. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑦) ∩ 𝜇𝑅(𝑦, 𝑥) = 0 – асимметричность
5. ∀𝑥, 𝑦 ⊆ 𝑋, 𝑥 ≠ 𝑦 ∶ 𝜇𝑅(𝑥, 𝑦) ∩ 𝜇𝑅(𝑥, 𝑥) – антисимметричность
6. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑦) ∪ 𝜇𝑅(𝑦, 𝑥) = 1 – сильная линейность
7. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑦) ∪ 𝜇𝑅(𝑦, 𝑥) > 0 – слабая линейность
8. ∀𝑥, 𝑦 ⊆ 𝑋 ∶ 𝜇𝑅(𝑥, 𝑦) ∩ 𝜇𝑅(𝑦, 𝑧) ≤ 𝜇𝑅(𝑥, 𝑧) = 0 – транзитивность

Листинг программы

def create\_matrix(size):

    matrix = [0] \* size

    for i in range (size):

        matrix[i] = [0] \* size

    return matrix

def dif\_matrix(matrix, list, size):

    for i in range(size):

        for j in range(size):

            matrix[i][j] = round(list[i]-list[j], 1)

    return matrix

def dif\_ABS\_matrix(matrix, list, size):

    for i in range(size):

        for j in range(size):

            matrix[i][j] = round(abs(list[i]-list[j]), 1)

    return matrix

def find\_max(matrix, size, case):

    maximum = 0

    if case == 'A':

        for i in range(size):

            for j in range(size):

                if matrix[i][j] <= 0 and matrix[i][j]>=maximum:

                    maximum = matrix[i][j]

    elif case == 'B':

        for i in range(size):

            for j in range(size):

                if matrix[i][j] >= maximum:

                    maximum = matrix[i][j]

    return maximum

def find\_min(matrix, size, case):

    minimum = 0

    if case == 'A':

        for i in range(size):

            for j in range(size):

                if matrix[i][j] <=0 and matrix[i][j]<minimum:

                    minimum = matrix[i][j]

    elif case == 'B':

        for i in range(size):

            for j in range(size):

                if matrix[i][j]<minimum:

                    minimum = matrix[i][j]

    return minimum

def dif\_matrix\_output(matrix, size):

    for i in range(size):

        for j in range(size):

            print(f'{matrix[i][j]} |', end=' ')

        print()

def affilation\_matrix(matrix, size, max, min, mid, case):

    if case == 'A':

        for i in range(size):

            for j in range(size):

                if matrix[i][j] < min:

                    matrix[i][j] = 0

                elif matrix[i][j] <= mid and matrix[i][j] >= min:

                    matrix[i][j] = round(2\*(((matrix[i][j] - min)/(max-min))\*\*2), 3)

                elif matrix[i][j] < max and matrix[i][j] >= mid:

                    matrix[i][j] = round(1 - 2 \* (((matrix[i][j] - max)/(max-min))\*\*2), 3)

                elif matrix[i][j] >= max:

                    matrix[i][j] = 0

                else:

                    matrix[i][j] = 0

    elif case == 'B':

        for i in range(size):

            for j in range(size):

                if matrix[i][j] <= min:

                    matrix[i][j] = 1

                elif matrix[i][j] <= mid and matrix[i][j] >= min:

                    matrix[i][j] = round(1 - 2\*(((matrix[i][j]-min)/(max-min))\*\*2), 3)

                elif matrix[i][j]<=max and matrix[i][j] >=mid:

                    matrix[i][j] = round(2\*(((matrix[i][j]-max)/(max-min))\*\*2), 3)

                else:

                    matrix[i][j] = 0

    return matrix

def refl(matrix):

    for i in range(len(matrix)):

        if not matrix[i][i] == 1:

            return False

    return True

def weak\_refl(matrix):

    for i in range(len(matrix)):

        for j in range(len(matrix)):

            if not matrix[i][j] <= matrix[i][j]:

                return False

    return True

def strong\_refl(matrix):

    for i in range(len(matrix)):

        for j in range(len(matrix)):

            if not matrix[i][j] < matrix[i][i]:

                return False

    return True

def irefl(matrix):

    for i in range(len(matrix)):

        if not matrix[i][i] == 0:

            return False

    return True

def weak\_irefl(matrix):

    for i in range(len(matrix)):

        for j in range(len(matrix)):

            if not matrix[i][i] <= matrix[i][j]:

                return False

    return True

def strong\_irefl(matrix):

    for i in range(len(matrix)):

        for j in range(len(matrix)):

            if not matrix[i][i] < matrix[i][j]:

                return False

    return True

def sym(matrix):

    for i in range(len(matrix)):

        for j in range(i, len(matrix)):

            if not matrix[i][j] == matrix[j][i]:

                return False

    return True

def antisym(matrix):

    if not refl(matrix):

        return False

    for i in range(len(matrix)):

        for j in range(i, len(matrix)):

            if i != j:

                if not matrix[i][j] \* matrix[j][i] == 0:

                    return False

    return True

def asym(matrix):

    for i in range(len(matrix)):

        for j in range(i, len(matrix)):

            if not matrix[i][j] \* matrix[j][i] == 0:

                return False

    return True

def weak\_lin(matrix):

    for i in range(len(matrix)):

        for j in range(len(matrix)):

            if i != j:

                if not (matrix[i][j] + matrix[j][i] - (matrix[i][j]\*matrix[j][i]))  > 0:

                    return False

    return True

def strong\_lin(matrix):

    for i in range(len(matrix)):

        for j in range(len(matrix)):

            if i != j:

                if not (matrix[i][j] + matrix[j][i] - (matrix[i][j]\*matrix[j][i]))  == 1:

                    return False

    return True

def trans(matrix):

    for i in range(len(matrix)):

        for j in range(len(matrix)):

            for k in range(len(matrix)):

                if not matrix[i][k] >= matrix[i][j] \* matrix[j][k]:

                    return False

    return True

def get\_relations(matrix, name):

    a = f'Отношение {name} обладает следующими свойствами:'

    if refl(matrix):

        if weak\_refl(matrix):

            a += ' слабая рефлексивность, '

        elif strong\_refl(matrix):

            a += ' сильная рефлексивность,'

    elif irefl(matrix):

        if weak\_irefl(matrix):

            a += ' слабая антирефлексивность, '

        elif strong\_irefl(matrix):

            a += ' сильная антирефлексивность,'

    if sym(matrix):

        a += ' симметричность,'

    elif antisym(matrix):

        a += ' антисимметричность,'

    elif asym(matrix):

        a += ' асимметричность,'

    if weak\_lin(matrix):

        a += ' слабая линейность,'

    elif strong\_lin(matrix):

        a += ' сильная линейность,'

    else:

        a += ' не линейно,'

    if trans(matrix):

        a += ' транзитивность,'

    else:

        a += ' не транзитивно,'

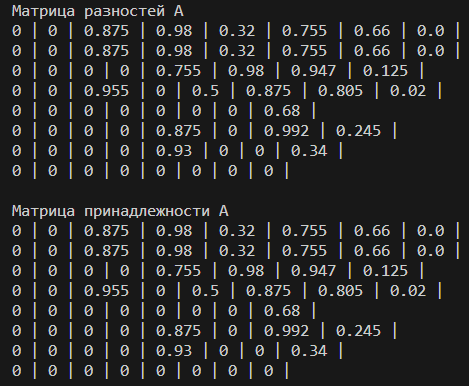
    if a[-1] == ',':

        a = a[:-1] + '.'

    return a

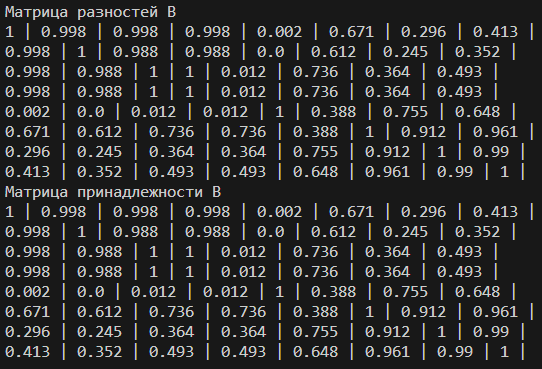
**Результаты работы программы**

Отношение «Часовая производительность автомата *y* немного выше, чем автомата *x*»





Отношение «Установленные мощности автоматов *x* и *y* примерно одинаковы»





Вывод

В данной лабораторной работе было сформировано и исследовано универсальное множество автоматов-укладчиков.

На основании данного множества были сформированы следующие нечёткие отношения:

1. Часовая производительность автомата *y* немного выше, чем автомата *x*;
2. Установленные мощности автоматов *x* и *y* примерно одинаковы.

Для этих отношений были определены значения функций принадлежности, была проведена проверка на наличие следующих свойств: рефлексивность/антирефлексивность (сильная или слабая), симметричность, антисимметричность, асимметричность, транзитивность, линейность (сильная или слабая)/отсутствие линейности.

После проверки можно сделать следующий вывод:

1. Отношение «часовая производительность автомата y немного выше, чем автомата x» слабо антирефлексивно, ассимметрично, не линейно;
2. Отношение «Установленные мощности автоматов *x* и *y* примерно одинаковы» слабо рефлексивно, симметрично, не линейно.