Министерство образования Республики Беларусь

Учреждение образования

Белорусский государственный университет

информатики и Радиоэлектроники

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

Отчёт

по лабораторной работе №1 по дисциплине:

# “Логические основы интеллектуальных систем”

Выполнил:

Студент гр. 221703 Демидовец Д.В.

Проверил: Ивашенко В.П.

Минск 2024

# СОДЕРЖАНИЕ

1. [Ход выполнения лабораторной работы](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.30j0zll) . . . . . . . . . . . . . . . 2
   1. [Описание](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.3znysh7) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2
   2. [Теоретические сведения](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.tyjcwt) . . . . . . . . . . . . . . . . . . . . . 3
   3. [Формат базы знаний](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.1t3h5sf) . . . . . . . . . . . . . . . . . . . . . . . 3
   4. [Реализация](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.4d34og8) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
   5. [Демонстрация результатов работы программы](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.17dp8vu) . . . . . . . . 9
      1. [Тест 1](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.3rdcrjn) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
      2. [Тест 2](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.26in1rg) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 9
      3. [Тест 3](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.lnxbz9) . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10
   6. [Контрольные вопросы](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.1ksv4uv) . . . . . . . . . . . . . . . . . . . . . . 11
2. [Вывод](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.3j2qqm3) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
   1. [Личный вклад](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.2xcytpi) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

[Список использованных источников](file:///C:\Users\HONOR\Downloads\lois_b.docx#_heading=h.1ci93xb) 15

# ХОД ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

## Вариант 2

**Тема**: Программирование операций обработки и преобразования формул прикладных и неклассических логик.

**Цель**: Приобрести навыки программирования алгоритмов обработки данных в неклассических логических моделях решения задач.

**Задание**: Реализовать систему прямого нечёткого логического вывода. Исходные данные и результаты должны быть сохранены в файле, используя треугольную норму произведения xi\*yi и нечеткую импликацию Гогена.

## Описание

Задача заключается в написании алгоритма прямого нечёткого логического вывода, используя треугольную норму произведения xi\*yi и нечеткую импликацию Гогена.

Входом программы является файл, содержащий множество нечётких правил и фактов.

Для реализации программы использовался язык программирования C++.

Структура проекта выглядит следующим образом.

Файл parser.hpp содержит заголовки и определения структур данных: Fact и Rule. Структура Fact используется для представления фактов нечеткой логики, она хранит имя предиката и его нечеткое множество (fuzzySet). Структура Rule содержит два имени предикатов, описывающих правило. Также в этом файле объявлены функции, которые используются для чтения фактов и правил из входных данных.

Файл parser.cpp реализует функции для парсинга входных данных. В нем реализованы функции лексического анализа строки, такие как nextChar и nextLex, для обработки символов входных данных. Кроме того, здесь описаны функции для парсинга нечетких множеств, фактов и правил. В коде предусмотрены проверки формата входных данных с выбросом исключений в случае ошибок.

Файл main.cpp является основной точкой входа в программу. В нем реализована логика считывания входных данных из файла, содержащего факты и правила. После этого для каждой пары фактов и правил создаются таблицы импликаций с помощью функции createMatr. Программа вычисляет новые множества фактов на основе входных данных, проверяя, возможно ли заключение для заданного множества и выполняя операцию нечеткого заключения с использованием t-нормы. В процессе работы используется очередь и множество для вычисления и сохранения всех новых выводов. Результаты выводятся в консоль в формате множества.

Дополнительно в main.cpp определена структура Matrix, которая используется для хранения правила и матрицы значений импликаций между элементами двух нечетких множеств. Функция Implication реализует операцию нечеткой импликации, а calcTNorma выполняет расчет t-нормы (умножение).

Входные данные поступают из файла input.txt, где первая часть содержит факты в формате: имя = {<элемент, значение>, ...}, а вторая часть — правила в формате:

имя1 ~> имя2. Программа выводит результат в виде множества новых фактов в формате {<элемент, значение>, ...}.

## Теоретические сведения

Нечеткий предикат – это нечеткое множество, значения которого интерпретируются как значения истинности.

Импликация – бинарная логическая связка, по своему применению приближенная к союзам «если..., то...».

Правило – импликация, которая выражает зависимость между наблюдаемыми причинами и следствиями.

Прямой нечеткий логический вывод – композиция между двумя нечеткими предикатами, один из которых рассматривается как унарный (посылка), а второй бинарный (импликация фактов по заданному правилу).

Нечеткое высказывание – утверждение, в котором истинность оценивается с использованием степени принадлежности к нечеткому множеству.

Нечеткая импликация нечетких высказываний - это операция, которая определяет отношение между двумя нечеткими высказываниями.

## Формат базы знаний

<база знаний> ::= <список фактов>| <список фактов><новая строка><список правил>

список фактов» ::= <факт>|<факт> <новая строка> <список фактов»

<список правил> ::= <правило> | <правило> <новая строка><список правил>

<факт> ::= <имя нечеткого предиката><равенство><нечеткое множество>

<правило> ::= <имя нечеткого предиката> <нечеткая импликация> <имя нечёткого предиката>

<нечеткое множество>::= <открывающая фигурная скобка> <список пар нечеткой принадлежности> <закрывающая фигурная скобка>

<список пар нечеткой принадлежности> ::= <пара нечеткой принадлежности> |<пара нечеткой принадлежности><запятая>< список пар нечеткой принадлежности>

<пара нечеткой принадлежности> ::= <открывающая угловая скобка><элемент><запятая><степень принадлежности><закрывающая угловая скобка>

<имя нечеткой принадлежности> ::= <имя><открывающая полукруглая скобка><имя><закрывающая полукруглая скобка>

<имя> ::= <буква> |<буква><символы>

<символы> ::= <символ>|<cимвол> <символы>

<степень принадлежности> ::= <действительное число от 0 до 1>

<действительное число с 0 до 1> ::= <единица> |<единица> <точка><нули>|<действительное число с 0 до 1>

<действительное число с 0 до 1> ::= <ноль>|<ноль><точка><цифры>

<цифры> ::= <цифра>|<цифра><цифры>

<символы> ::= <буква>|<цифра>

<буква>A|B|C|D|F|G|H|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z|a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z

<цифра> ::= 0|1|2|3|4|5|6|7|8|9

<единица> ::=1

<ноль>::=0

<точка> ::=.

<запятая>::=,

<равенство>::= =

<нечеткая импликация> ::=~>

<открывающая угловая скобка> ::= <

<закрывающая угловая скобка> ::=<

<открывающая фигурная скобка> ::= {

<закрывающая фигурная скобка>::=}

<открывающая полукруглая скобка> ::= (

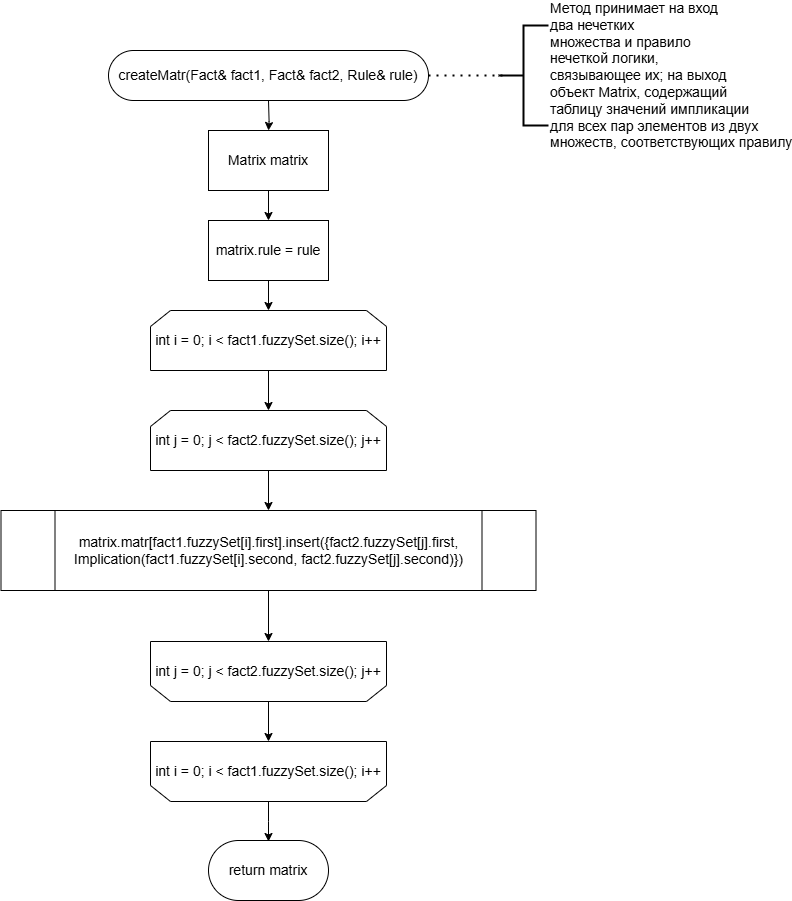
<закрывающая полукруглая скобка> ::=)

## Реализация прямого нечеткого логического вывода

В программе алгоритмы прямого нечеткого логического вывода реализованы в следующих основных функциях:

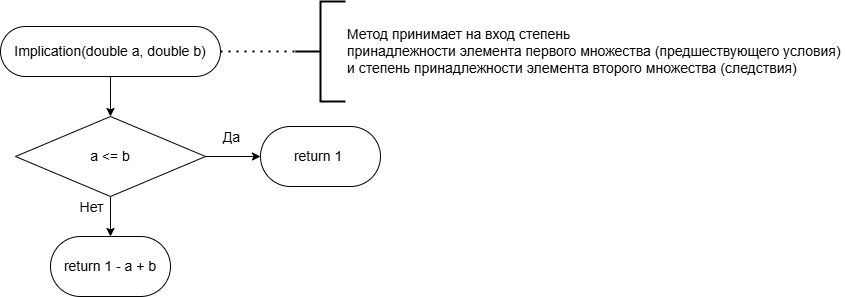
* *Matrix createMatr(Fact& fact1, Fact& fact2, Rule& rule)*

Эта функция создает таблицы импликаций (матрицы значений) между элементами двух нечетких множеств на основе заданного правила. Она вычисляет значения нечеткой импликации для всех пар элементов двух фактов.



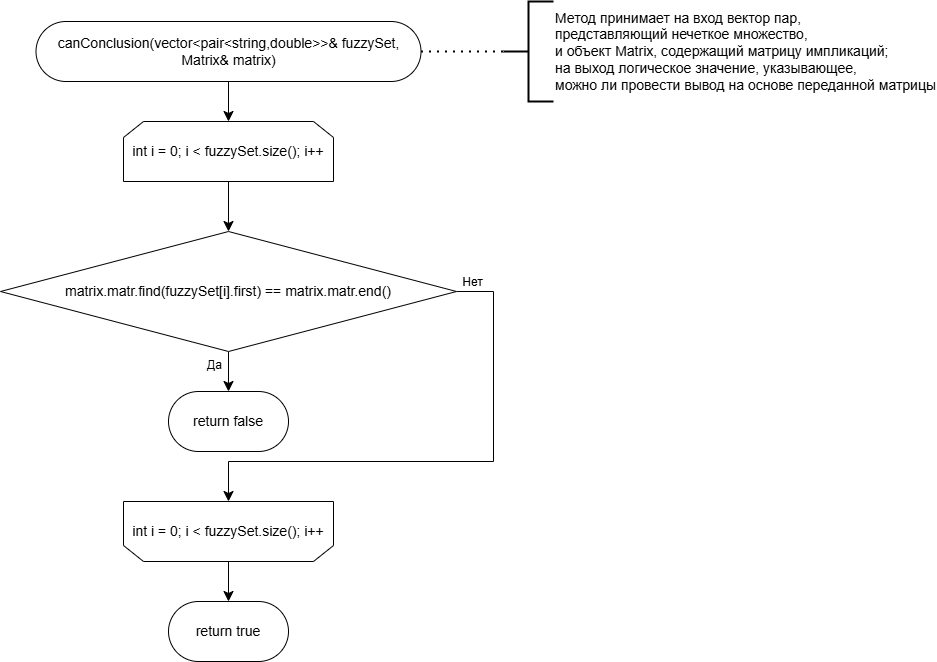
* *Implication(double a, double b)*

Функция используется для вычисления степени истинности связи между двумя нечеткими переменными. В данном случае параметр a представляет степень принадлежности элемента первого множества (предшествующее условие), а параметр b — степень принадлежности элемента второго множества (следствие).



* *bool canConclusion(vector<pair<string,double>>& fuzzySet, Matrix& matrix)*

Проверяет возможность применения правила для заданного нечеткого множества, определяя, содержатся ли все элементы этого множества в таблице импликаций.



* *vector<pair<string,double>> fuzzyConclusion(vector<pair<string, double>>& fuzzySet, Matrix& matrix)*

Выполняет вычисление нового нечеткого множества (вывод) на основе входного множества и таблицы импликаций. Для этого используется t-норма для комбинирования значений.



Основной цикл программы в main последовательно применяет правила к каждому входному множеству, формируя новые множества выводов. Этот процесс завершается, когда перестают появляться новые результаты.

## Демонстрация результатов работы программы

* + 1. **Тест 1**

Входные данные:

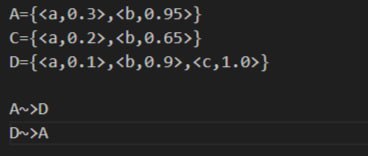


Рисунок 1.1 – Входные данные для теста 1

Результат работы программы для теста:

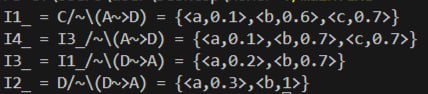


Рисунок 1.2 – Результат работы программы для теста 1

## Тест 2

Входные данные:

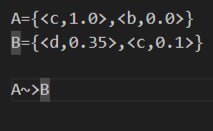


Рисунок 1.3 – Входные данные для теста 2

Результат работы программы для теста:



Рисунок 1.4 – Результат работы программы для теста 2

## Тест 3

Входные данные:

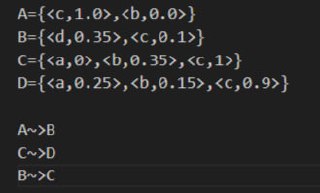


Рисунок 1.5 – Входные данные для теста 3

Результат работы программы для теста:

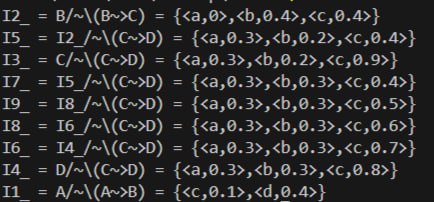


Рисунок 1.6 – Результат работы программы для теста 3

## Контрольные вопросы

## Нечеткое множество называется нормальным, если точная верхняя грань его функции принадлежности равна 1

1. **При каких A’(x) можно получить субнормальное нечёткое множество B’(y), когда посылка и заключение нечёткой импликации являются нормальными нечёткими множествами?**

Пример на рисунке 1.7:

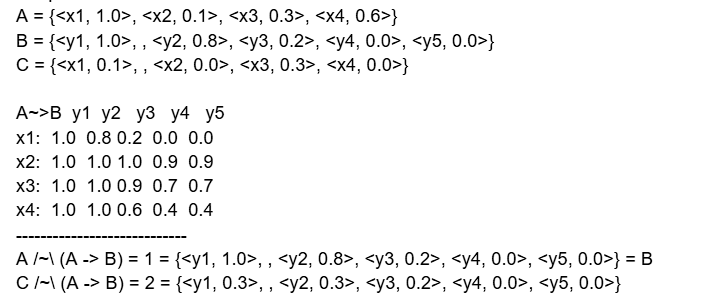


Рисунок 1.7 – Пример вывода субнормального предиката из нормальных *α* и *β*

Предикаты A, B являются нормальными, предикат C является субнормальным, результат I1 является субнормальным.

## При каких A’(x) можно получить нормальное нечёткое множество B’(y), когда посылка и заключение нечёткой импликации являются субнормальными нечёткими множествами м?

Пример на рисунке 1.8:

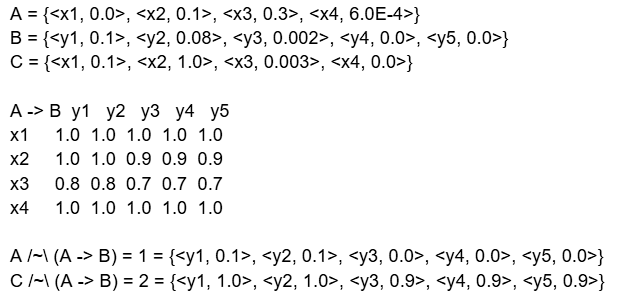
**

Рисунок 1.8 – Пример вывода субнормального предиката из нормальных *α* и *β*

Предикат *A*, *B* являются субнормальными, предикат C нормальным, результат I1 является нормальным.

## Можно ли получить нечёткое множество B’(y)=B(y), когда посылка и заключение нечёткой импликации являются субнормальными нечёткими множествами?

Возможно для некоторых случаев.  
Пример на рисунке 1.9:

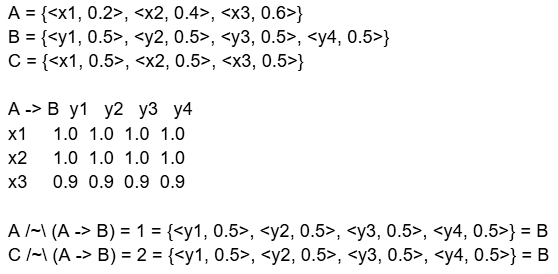


Рисунок 1.9 – Пример вывода предиката B’(y)=B(y) из субнормальных *α* и *β*

## Можно ли получить нечёткое множество B’(y)=B(y), когда посылка и заключение нечёткой импликации являются нормальными нечёткими множествами?

Возможно для некоторых случаев.  
Пример на рисунке 1.10:

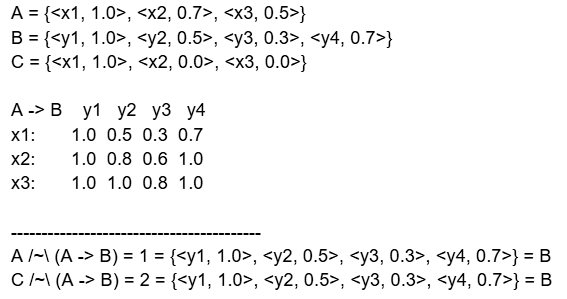


Рисунок 1.10 – Пример вывода предиката B’(y)=B(y) из нормальных *α* и *β*

## Можно ли получить нечёткое множество B’(y)=B(y), когда посылка является нормальным нечётким множеством, а заключение – субнормальным?

Возможно для некоторых случаев.  
Пример на рисунке 1.11:

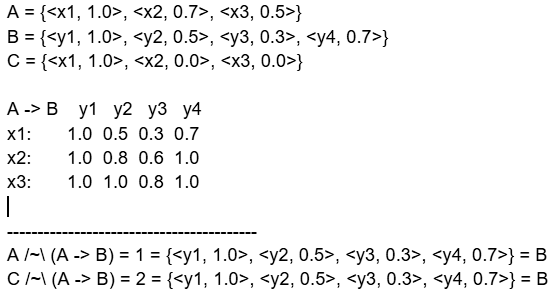


Рисунок 1.11 – Пример вывода предиката B’(y)=B(y)

из нормального *α* и субнормального *β*

## Можно ли получить нечёткое множество B’(y)=B(y), когда посылка является субнормальным нечётким множеством, а заключение – нормальным?

Возможно для некоторых случаев.  
Пример на рисунке 1.11:

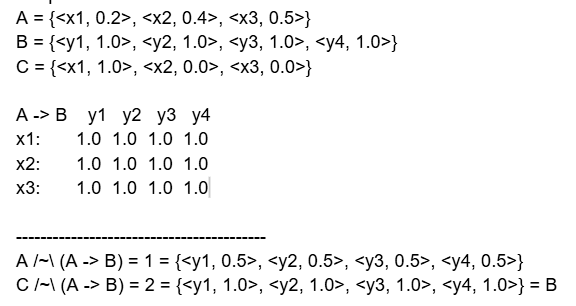


Рисунок 1.11 – Пример вывода предиката B’(y)=B(y)

из субнормального *α* и нормального *β*

# ВЫВОД

В ходе выполнения лабораторной работы были приобретены навыки разработки программных систем, реализующих методы нечеткой логики. В рамках работы создан программный продукт, включающий модули для анализа текста базы знаний и вычисления нечеткого вывода. Реализован алгоритм построения таблиц импликаций и выполнения нечеткого заключения на основе входных фактов и правил. Программа успешно продемонстрировала возможность получения корректных выводов в заданных сценариях, что подтверждено результатами проведенных тестов. Также в процессе выполнения лабораторной работы были даны ответы на контрольные вопросы, что свидетельствует о понимании темы.

# 2.1 Личный вклад:

Дмитрий Козырев занимался программированием поставленного задания.

Дарья Демидовец отвечала за работу над отчётом и вопросами.

# CПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Голенков, В. В. Логические основы интеллектуальных систем. Практикум: учеб.-метод. пособие / В. В. Голенков. — БГУИР, 2011

[2] ГОСТ 19.701–90 (ИСО 5807–85) «Единая система программной документации»