Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫ
УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет Информационных технологий и управления

Кафедра Интеллектуальных информационных технологий

Отчёт по лабораторной работе №1
по дисциплине
«Логические основы интеллектуальных систем»

Выполнили студенты гр. 121702 В.И. Буланович,
Р.А. Заломов
И.А. Готин

Проверил: В. П. Ивашенко

Минск 2023

Тема: Представление и обработка информации в условиях наличия не-факторов в рамках логических моделей

Цель: Приобрести навыки программирования алгоритмов обработки структур и формул в нечёткой логике

Задача: Реализовать прямой нечеткий логический вывод, используя импликацию Гогена

Описание лабораторной работы:

Задача заключается в создании программного модуля, который будет выполнять прямой нечеткий логический вывод на основе импликации Гогена. Данная задача предполагает работу с нечеткими правилами и фактами, а также применение операции импликации Гогена для получения нечеткого вывода.

Ход выполнения работы:

1. Изучение нечеткой логики и импликации Гогена:
    - Изучить основы нечеткой логики, включая нечеткие множества, функции принадлежности и операции нечеткого вывода.
    - Ознакомиться с операцией импликации Гогена, которая используется для выполнения нечеткого логического вывода.
2. Разработка программного модуля:
    - Реализовать функции, решающие поставленную задачу.
3. Тестирование и анализ результатов
    - Написать несколько тестовых случаев, чтобы проверить работу программного модуля. В тестах учесть различные комбинации правил и фактов
    - Запустить тесты и проанализировать результаты

Теоретические сведения:

Прямой нечеткий логический вывод - процесс, при котором из нечетких посылок получают некоторые следствия, возможно, тоже нечеткие.

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

Нечеткий предикат - это нечеткое множество, значения которого интерпретируется как значения истинности.

Импликация Гогена - определение степени истинности высказывания на основе степени истинности условия и следствия.

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

Нечеткая импликация нечетких высказываний - это бинарная логическая операция, результат которой является нечетким высказыванием «из А следует В», «если А, то В».

Нечеткое высказывание - это исполненная мысль, об истинности или ложности которой можно судить только с некоторой степенью истинности, принимающей значения в отрезке [0;1].

Описание алгоритма:

В программе были реализованы такие функции как:

- def __gauguin_norm_delta(f_belonging_degree: int | float, s_belonging_degree: int | float) -> float

  - Вычисление степени принадлежности пары, принадлежащей нечёткому отношению, которое является результатом нечёткой импликации Гогена.

- def __fuzzy_implication(cls, fuzzy_set_1: dict, fuzzy_set_2: dict) -> pd.DataFrame

  - Вычисление результата операции нечёткой импликации над двумя нечёткими предикатами.

- def __fuzzy_conclusion(cls, fact: dict, implication_matrix: pd.DataFrame) -> dict | None

  - Вычисление результата прямого нечёткого логического вывода.

- def __solve_implications(cls, parse_result: dict) -> dict | None

  - Вычисление всевозможных нечётких импликаций (на основе данных правил).

- def solve(cls, parse_result: dict) -> list[NamedFuzzyConclusion] | None

  - Метод, собирающий все функции решателя. Принимает результат парсинга файла с описанием правил и фактов.

- def __get_all_conclusions(cls, program_file: str) -> list

  - Метод класса, объединяющего решатель и парсер. Получает готовые для вывода на экран результаты прямых нечётких логических выводов.

- def print_conclusions_results(cls, program_file: str)

  - Метод класса, объединяющего решатель и парсер. Выводит на экран результаты прямых нечётких логических выводов.

- def fuzzy_set_dict_to_str(conclusion_result: dict | None) -> str | None

  - Превращение читаемого для системы формата нечёткого предиката в читаемую для пользователя системы строку.

- def __parse_fuzzy_set(cls, raw_line: str) -> dict | None

  - Парсинг строки с нечётким предикатом в удобный для системы формат.

- def __parse_program_file(cls, file_dir: str) -> dict | None

  - Певрвоначальный парсинг текстового файла, содержащего описание фактов и правил.

- def __parse_program_file_result(cls, raw_parse: dict) -> dict | None

  - Второй этап парсинга текствого файла, содержащего описание фактов и правил. Результат является интерпретируемым системой.

- def __parse_fuzzy_implication(cls, raw_line: str) -> FuzzyImplication

  - Парсинг строки, содержащей правило.

- def parse(cls, file_dir: str = 'program') -> dict | None

  - Метод, объединяющий работу всех методов парсинга парсера системы.

Тесты:

Для проверки работоспособности программы были проведены тесты, которые указаны ниже:

Тест 1.

```
A={<x1,0.0>,<x2,0.1>,<x3,0.3>,<x4,1.0>}.
B={<y1,1.0>,<y2,0.8>,<y3,0.2>,<y4,0.0>}.
C={<x1,0.0>,<x2,0.1>,<x3,0.3>,<x4,1.0>}.
D={<z1,0.2>,<z2,0.3>,<z3,0.1>,<z4,0.8>}.

A~>B.
B~>D.
```

Рис. 1. Входные данные теста 1.

```
Give path to file: program
All possible conclusions results:
|-1.{C,(A~>B)}|-{<y1,1.0>,<y2,0.8>,<y3,0.2>,<y4,0.0>}=P#1
    |-2.{P#1,(B~>D)}|-{<z1,0.2>,<z2,0.3>,<z3,0.1>,<z4,0.8>}=P#2

Process finished with exit code 0
```

Рис. 2. Выходные данные теста 1.

Тест 2.

```
A={<x1,0.0>,<x2,0.1>}.
B={<y1,1.0>,<y2,0.8>}.
C={<u1, 0.5>, <u2, 1.0>}.
D={<z1, 0.2>, <z2, 0.1>}.
F={<x1, 0.1>, <x2, 1.0>}.


A~>B.
B~>D.
C~>B.
B~>A.
D~>C.
```

Рис. 3. Входные данные теста 2.

```
Give path to file: program
All possible conclusions results:
|-1.{F,(A~>B)}|-{<y1,1.0>,<y2,1.0>}=P#1
    |-2.{P#1,(B~>D)}|-{<z1,0.25>,<z2,0.125>}=P#2
    |-2.{P#1,(B~>A)}|-{<x1,0.0>,<x2,0.125>}=P#3
        |-3.{P#2,(D~>C)}|-{<u1,0.25>,<u2,0.25>}=P#4
            |-4.{P#4,(C~>B)}|-{<y1,0.25>,<y2,0.25>}=P#5

Process finished with exit code 0
```

Рис. 4. Выходные данные теста 2.

Тест 3.

```
A={<y1,0.0>,<y2,0.1>}.
B={<y1,1.0>,<y2,0.8>}.
C={<x1,0.0>, <x2,0.0>}.


B~>C.
```

Рис. 5. Входные данные теста 3

```
Give path to file: program
All possible conclusions results:
|-1.{A,(B~>C)}|-{<x1,0.0>,<x2,0.0>}=P#1

Process finished with exit code 0
```

Рис. 6. Выходные данные теста 3.

Тест 4.

```
A={<y1,0.0>,<y2,0.1>,<y3,0.25>}.
B={<y1,1.0>,<y2,0.8>,<y3,0.13>}.
C={<x1,0.0>,<x2,0.0>,<x3,0.5>}.
D={<z1,0.6>,<z2,0.8>,<z3,0.9>}.
E={<u1,1.0>,<u2,0.4>,<u3,0.1>}.
F={<z1,0.3>,<z2,0.2>,<z3,0.2>}.
G={<m1,0.0>,<m2,0.0>,<m3,0.2>}.


B~>C.
A~>F.
C~>G.
D~>G.
F~>G.
F~>E.
```

Рис. 7. Входные данные теста 4.

```
Give path to file: program
All possible conclusions results:
|-1.{A,(B~>C)}|-{<x1,0.0>,<x2,0.0>,<x3,0.25>}=P#1
|-1.{B,(A~>F)}|-{<z1,1.0>,<z2,1.0>,<z3,1.0>}=P#2
|-1.{D,(F~>G)}|-{<m1,0.0>,<m2,0.0>,<m3,0.9>}=P#3
|-1.{D,(F~>E)}|-{<u1,0.9>,<u2,0.9>,<u3,0.45>}=P#4
|-1.{F,(D~>G)}|-{<m1,0.0>,<m2,0.0>,<m3,0.1>}=P#5
    |-2.{P#1,(C~>G)}|-{<m1,0.0>,<m2,0.0>,<m3,0.1>}=P#6

Process finished with exit code 0
```

Рис. 8. Выходные данные теста 4.

Контрольные вопросы (ответы приведены для импликации Гогена, в соответствии с вариантом):

1 Если множества α и β являются нормальными, то возможен ли случай при каких-либо значениях α', когда результат не будет являться нормальным множеством?

Ответ:

```
A={<x1,1.0>,<x2,0.1>,<x3,0.25>}.
B={<y1,1.0>,<y2,0.8>,<y3,0.1>}.
C={<x1,0.1>,<x2,0.2>,<x3,0.12>}.



A~>B.
```

Рис 9. Входные данные, иллюстрирующие ответ на контрольный вопрос 1.

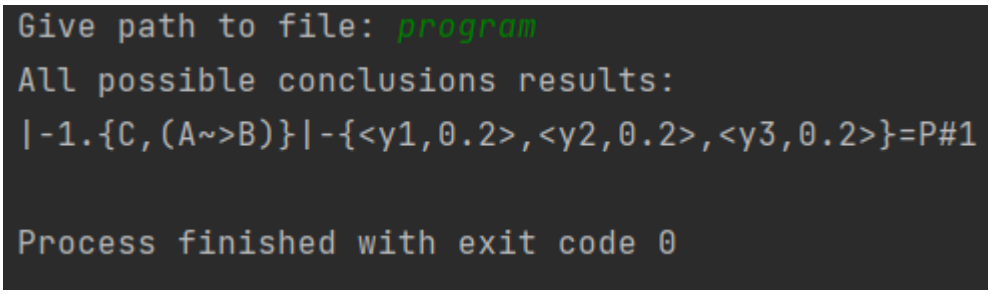

Рис 10. Выходные данные, иллюстрирующие ответ на контрольный вопрос 1.

Да. Множества A и B - нормальные, множество C - субнормальное. Учитывая данную треугольную норму, выходит, что если предикат, который используется при выводе (в этом случае C) не является нормальным нечётким множеством, то и результат не будет нормальным нечётким множеством, т.к. при умножении на единицу числа, меньшего единицы, получается число меньшее единицы. Это будет действительным для каждой строки матрицы нечёткого отношения (результат нечёткой импликации A и B), а значит итоговые степени принадлежности в любом случае будут числами меньше 1, а это, в свою очередь, означает, что максимальное число оттуда тоже будет меньше 1, что приведёт к получению субнормального множества как результата нечёткого прямого логического вывода.

2 Если множества α и β не являются нормальными, то возможен ли случай при каких-либо значениях α', когда результат будет являться нормальным множеством?

Ответ:

```
A={<x1,0.8>,<x2,0.7>,<x3,0.5>}.
B={<y1,0.3>,<y2,0.4>,<y3,0.6>}.
C={<x1,0.2>,<x2,0.2>,<x3,1.0>}.


A~>B.
```

Рис 11. Входные данные, иллюстрирующие ответ на контрольный вопрос 2.

```
Give path to file: program
All possible conclusions results:
|-1.{C,(A~>B)}|-{<y1,0.6>,<y2,0.8>,<y3,1.0>}=P#1

Process finished with exit code 0
```

Рис 12. Выходные данные, иллюстрирующие ответ на контрольный вопрос 2.

Да. Множества A и B - субнормальные (все степени принадлежности меньше 1). Результирующее множество может быть нормальным, если для факта, который используется в выводе и для фактов, участвующих в правиле, выполняются следующие условия : существует такой элемент из носителя факта, участвующего в выводе, степень принадлежности которого принимает значение 1 (в множестве C это элемент x3 – рис 11.). В свою очередь, в факте, который является первой импликантой нечёткой импликации (в иллюстрирующем примере это факт A – рис. 11), этот же элемент носителя должен иметь степень принадлежности, меньшую чем степень принадлежности какого-нибудь элемента из второй импликанты (элемент y3 из множества B имеет степень принадлежности 0.6, а элемент x3 из A имеет степень принадлежности 0.5). Результирующее множество получилось нормальным (множество

P#1 из рис. 12)

3 Какими значениями α', α и β можно гарантировать, что результат будет являться нормальным множеством?

```
A={<x1,0.0>,<x2,0.1>,<x3,0.3>}.
B={<y1,0.3>,<y2,0.8>,<y3,0.1>}.
C={<x1,1.0>,<x2,0.0>,<x3,0.12>}.


A~>B.
```

```
Give path to file: program
All possible conclusions results:
|-1.{C,(A~>B)}|-{<y1,1.0>,<y2,1.0>,<y3,1.0>}=P#1

Process finished with exit code 0
```

Некоторые случаи возможности получения требуемого условия:

1. Входные данные и условия аналогичные ответу на контрольный вопрос 2.

2. Вторая импликанта нечёткой импликации – нормальное множество, факт, используемый при выводе – нормальное множество.

Личный вклад разработчиков системы

Заломов Р.А. – разработка парсера, решателя, сборка компонентов системы, ответы на контрольные вопросы.

Готин И.А. – разработка парсера, решателя, сборка компонентов системы.

Буланович В.И. – тесты, составление отчёта, ответы на контрольные вопросы.

Вывод:

В ходе лабораторной работы были приобретены практические навыки программирования обработки структур и формул нечёткой логики посредством реализации программной системы прямого нечеткого логического вывода с использованием импликации Гогена. Разработанный программный модуль позволяет выполнять нечеткий логический вывод на основе заданных нечетких правил и фактов. В ходе тестирования была проверена работоспособность реализованного программного модуля с субнормальными и нормальными нечеткими множествами. Тестирование показало соответствие результатов ожидаемым значениям вывода.

Список использованных источников:

Логические основы интеллектуальных систем. Практикум : учеб.- метод. пособие / В. В. Го- ленков [и др.]. – Минск : БГУИР, 2011. – 70 с. : ил. ISBN 978-985-488-487-5.