

Міністерство освіти і науки, молоді та спорту України
Запорізький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ
до виконання розрахунково-графічного завдання
з дисципліни
«НЕЧІТКЕ ПРОГРАМУВАННЯ»
для студентів спеціальності
8.05010301 “Програмне забезпечення систем”
(усіх форм навчання)

Методичні вказівки до виконання розрахунково-графічного завдання з дисципліни «Нечітке програмування» для студентів спеціальності 8.05010301 “Програмне забезпечення систем” (усіх форм навчання) / Уклад.: С.О. Субботін. – Запоріжжя: ЗНТУ, 2012. – 48 с.

Укладач: Сергій Олександрович Субботін,
кандидат технічних наук, доцент,
доцент кафедри програмних засобів

Рецензент: А.О. Олійник, кандидат технічних наук

Відповідальний
за випуск: В.І. Дубровін, зав. каф. програмних засобів

Затверджено
на засіданні кафедри
програмних засобів

Протокол № 12
від 27.04.2012 р.

ЗМІСТ

Загальні положення	4
1. Розрахункова частина завдання.....	6
2. Графічна частина завдання	9
3. Реферативна (факультативна) частина завдання	10
Література.....	11
Додаток А. Приклад оформлення титульного аркушу звіту про виконання розрахунково-графічного завдання.....	12
Додаток Б. Нечіткі системи та нейро-нечіткі мережі у пакеті Matlab	13
Додаток В. Перелік тем для виконання реферативної частини розрахунково-графічного завдання	48

ЗАГАЛЬНІ ПОЛОЖЕННЯ

Дане видання призначене для вивчення та практичного освоєння студентами усіх форм навчання теоретичних основ та практичних навичок і вмінь з навчальної дисципліни шляхом виконання розрахунково-графічних завдань.

Розрахунково-графічне завдання є різновидом індивідуальних завдань, що виконуються з окремих навчальних дисциплін студентами самостійно при консультуванні викладачем.

Студент виконує розрахунково-графічне завдання протягом семестру відповідно до варіанту завдання, наданого викладачем на початку семестру.

При виконанні розрахунково-графічного завдання студент має використовувати конспект лекцій та рекомендовану літературу.

Для одержання оцінки за виконане розрахунково-графічне завдання здає викладачу цілком оформлений звіт, а також оптичний носій (CD або DVD) у форматі MS-DOS / Windows, перевірений на відсутність вірусів, з текстами розроблених програм, файлами програм, що виконуються, файлами даних і текстом звіту.

Звіт має містити:

- титульний аркуш (на ньому вказують назву міністерства, назву університету, назву кафедри, вид і тему роботи, виконавця та особу, що приймає звіт, рік – приклад оформлення наведено у додатку А);
- мету, номер варіанту і формулювання вирішуваного завдання;
- лаконічний опис використовуваних теоретичних відомостей;
- текст програми, що обов'язково містить коментарі;
- вхідні та вихідні дані програми;
- змістовний аналіз отриманих результатів та висновки.

Звіт виконують на білому папері формату А4 (210 x 297 мм). Текст розміщують тільки з однієї сторони листа. Поля сторінки з усіх боків – 20 мм. Аркуші скріплюють за допомогою канцелярських скріпок. Дозволяється виконання звіту уручну, на друкарській машинці або шляхом комп'ютерного набору із роздруківкою на принтері (рекомендований вид виконання) – за вибором студента.

При виконанні звіту із застосуванням комп'ютерної техніки для набору тексту звіту використовують редактор MS Word (версії 97 і вище) або сумісний з ним. Параметри тексту документу: гарнітура

шрифту – Times New Roman, розмір шрифту: 14 пунктів – для тексту звіту, 10 пунктів – для роздруківок програм та великих таблиць, міжрядковий інтервал: полуторний – для тексту звіту, одинарний – для листингів програм, таблиць і роздруківок даних, абзацний відступ встановлюють 1,5 см. Таблиці і рисунки та перелік посилань оформлюють відповідно до чинних стандартів.

Поданий звіт з виконання розрахунково-графічного завдання та програмне забезпечення на відповідних носіях перевіряється викладачем та захищається студентом шляхом усної або письмової відповіді на питання викладача і демонстрації роботи розробленого програмного забезпечення на ЕОМ.

Під час співбесіди студент повинний виявити знання про мету виконуваного завдання, по теоретичному матеріалу, про методи виконання кожного етапу роботи, по змісту основних розділів розробленого звіту з демонстрацією результатів на конкретних прикладах. Студент повинен вміти правильно аналізувати отримані результати.

Оцінювання розрахунково-графічного завдання здійснюється викладачем за такою шкалою:

– "незадовільно" – студент не виконав, виконав не повністю чи неправильно виконав поставлене завдання або не здатен пояснити хід його виконання, а також у випадку виявлення плагіату (спроба здачі чужої роботи);

– "задовільно" – студент в цілому виконав і захистив поставлене завдання, але з грубими помилками (у тому числі в оформленні звіту);

– "задовільно" – студент в цілому виконав і захистив поставлене завдання, але з незначними помилками (у тому числі в оформленні звіту);

– "відмінно" – студент в цілому виконав і захистив поставлене завдання, але з незначними помилками (у тому числі в оформленні звіту).

Перездача розрахунково-графічного завдання студентом здійснюється лише у випадку отримання ним оцінки "незадовільно".

1 РОЗРАХУНКОВА ЧАСТИНА ЗАВДАННЯ

Мета роботи: вивчити та засвоїти на практиці принципи побудови нейро-нечітких мереж у пакеті MATLAB, навчитися створювати та використовувати програмні засоби, що реалізують моделі нейро-нечітких мереж та методи їх навчання.

Порядок виконання розрахункової частини завдання

1. Ознайомитися з конспектом лекцій та рекомендованою літературою.

2. Згідно з номером індивідуального варіанта студента за журналом згенерувати навчаючу та тестову вибірки даних. Нехай V – номер студента за журналом. Визначимо кількість екземплярів у навчальній вибірці S_n , кількість екземплярів у тестовій вибірці S_t , кількість вхідних змінних (ознак) вибірок N , значення ознак вибірок: навчальної – x_n та тестової – x_t , а також значення цільових ознак для вибірок: навчальної – y_n та тестової – y_t , де

$$N = \begin{cases} V, & \text{якщо } V < 15; \\ V - 14, & \text{якщо } V > 15, \end{cases}$$

$$S_n = \begin{cases} 10V, & \text{якщо } V < 5; \\ 5V, & \text{якщо } 5 \leq V < 10; \\ 2V, & \text{якщо } 10 \leq V < 20; \\ V, & \text{інакше,} \end{cases}$$

$$S_t = \begin{cases} 8V, & \text{якщо } V < 5; \\ 6V, & \text{якщо } 5 \leq V < 10; \\ 3V, & \text{якщо } 10 \leq V < 20; \\ 2V, & \text{інакше,} \end{cases}$$

$$x_n = \{x_j^s\}, y_n = \{y^s\}, s=1,2,\dots,S_n;$$

$$x_t = \{x_j^s\}, y_t = \{y^s\}, s=1,2,\dots,S_t;$$

$$x_j^s = \frac{V}{s \cdot j} \cdot \text{rand}, j=1,2,\dots,N,$$

$$y^s = \begin{cases} 0,1(x_1^s + x_2^s), x_1^s < V; \\ 0,3(x_1^s - x_2^s), x_1^s > V; \\ 0,5x_1^s, x_1^s = V. \end{cases}$$

Тут `rand` – функція, що генерує псевдовипадкові числа у діапазоні [0; 1].

3. Для згенерованих вибірок за допомогою редактору `anfisedit` пакету `Matlab` побудувати нейро-нечітку модель. Спробувати використати різні методи кластеризації, різні кількості функцій належності для входів, різні кількості циклів навчання та різні методи навчання.

4. Протестувати побудовану модель.

5. Виконати пункти 3–4 у режимі командного вікна без застосування редактору `anfisedit`, використовуючі функції модуля `Fuzzy Logic Toolbox`.

Зміст розділу звіту з розрахункової частини

1. Мета роботи.

2. Короткі теоретичні відомості до роботи. У звіті не слід наводити докладне викладення теоретичного матеріалу, необхідно виділити лише найголовніші ідеї, формули, що необхідні для пояснення суті методу, моделі тощо.

3. Текст розробленої програми.

4. Сформована навчальна вибірка $\langle x, y \rangle$.

5. Побудовані моделі.

6. Результати тестування моделей на основі нейро-нечітких мереж.

7. Висновки щодо порівняння різних моделей на основі нейро-нечітких мереж та методів їх синтезу.

Контрольні питання

1. Використання кластер-аналізу при побудові систем розпізнавання образів.

2. Вимоги до навчаючих вибірок даних.

3. Задача кластер-аналізу.

4. Застосування нейро-нечітких мереж для видобування знань з даних.
5. Лінійна роздільність і лінійна нерозділеність класів.
6. Навчання з учителем. Навчання без учителя.
7. Нечіткий кластер-аналіз.
8. Подібність кластер-аналізу і метричної класифікації.
9. Редактор anfisedit.
10. Структура та елементи нейро-нечітких мереж.
11. Функції кластер-аналізу у пакеті MATLAB.
12. Функції пакету MATLAB для створення нейро-нечітких мереж.
13. Що таке генеральна сукупність, вибірка, екземпляр, ознака?
14. Як впливає задана точність навчання на тривалість навчання?
15. Як впливає задана кількість циклів навчання на точність навчання?
16. Який алгоритм кластер-аналізу призводить до отримання мережі меншої складності?
17. Які вимоги мають пред'являтися до навчальної вибірки та як це вплине на процес навчання?

2 ГРАФІЧНА ЧАСТИНА ЗАВДАННЯ

Порядок виконання графічної частини завдання

1. Ознайомитися з конспектом лекцій та рекомендованою літературою.
2. Зобразити графічно схему нейро-нечіткої мережі, побудованої у розрахунковій частині (уручну та за допомогою пакету Matlab).
3. Зобразити на графіках залежності зміни точності (помилки) навчання нейро-нечіткої мережі від номеру ітерації (епохи) навчання.
4. Проаналізувати отримані результати та відповісти на питання: який алгоритм кластер-аналізу призводить до отримання мережі меншої складності; як впливає задана кількість циклів навчання на точність навчання; як впливає задана точність навчання на тривалість навчання; які вимоги мають пред'являтися до навчальної вибірки та як це вплине на процес навчання?

Зміст розділу звіту з графічної частини

1. Зображення структури отриманої мережі.
2. Графіки експериментально отриманих залежностей.
3. Висновки. У висновках треба проаналізувати результати роботи, а також лаконічно відповісти на контрольні питання з розрахункової та графічної частин.

Контрольні питання

1. Засоби двовимірної графіки пакету MATLAB.
2. Засоби тривимірної графіки пакету MATLAB.
3. Порівняння графічних засобів пакету MATLAB та інтегрованих середовищ розробки програм Visual Studio.
4. Як впливає задана точність навчання на тривалість навчання?
5. Як впливає задана кількість циклів навчання на точність навчання?

З РЕФЕРАТИВНА (ФАКУЛЬТАТИВНА) ЧАСТИНА ЗАВДАННЯ

У випадку, якщо студент бажає більш глибоко засвоїти курс лекцій та підвищити оцінку з курсу, пропонується виконати факультативну (необов'язкову) частину розрахунково-графічного завдання шляхом підготовки короткого реферату з курсу.

Реферат з курсу готується за рекомендованими та іншими знайденими студентом джерелами і оформлюється як розділ звіту з розрахунково-графічного завдання та розміщується перед висновками.

Обсяг реферативної частини має становити 3–5 сторінок без урахування рисунків і таблиць, а також текстів програм, що займають повну сторінку звіту.

До виконання реферативної частини студент обирає тему з наведеного у додатку В переліку (або формулює власну тему) і узгоджує її з викладачем.

У переліку посилань звіту з розрахунково-графічного завдання мають бути обов'язково наведені усі використані студентом при написанні реферату джерела, упорядковані за згадуванням у тексті, а у тексті реферативної частини – наведені посилання на відповідні джерела (наводяться у квадратних дужках).

Оцінка за реферат проставляється тільки у випадку його наявності у звіті і ставиться окремо від основної оцінки за розрахунково-графічне завдання за двобальною системою:

– "не зараховано" – у випадку низького рівня виконання або плагіату реферату;

– "зараховано" – у випадку прийнятного рівня виконання.

Якщо оцінка за основну (обов'язкову) частину звіту є нижчою від "задовільно", а оцінка з реферативної частини – "зараховано", тоді загальна оцінка за виконання розрахунково-графічного завдання встановлюється як вища на один рівень від оцінки за основну частину. У випадку, якщо основну (обов'язкову) частину звіту виконано на "відмінно", то оцінка "зараховано" враховується як 10 балів за стобальною системою у загальній сумі балів з відповідного модуля курсу.

ЛІТЕРАТУРА

Основна література

1. Субботін С. О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень : навчальний посібник / С. О. Субботін. – Запоріжжя: ЗНТУ, 2008. – 341 с.
2. Олійник А. О. Інтелектуальний аналіз даних : навчальний посібник / А. О. Олійник, С. О. Субботін, О. О. Олійник. – Запоріжжя : ЗНТУ, 2012. – 271 с.
3. Зайченко Ю.П. Основи проєктування інтелектуальних систем : навчальний посібник / Ю.П. Зайченко. – К.: Слово, 2004. – 352 с.

Додаткова література

4. Рідкокаша А. А. Основи систем штучного інтелекту : навчальний посібник / А.А.Рідкокаша, К.К. Голдер. – Черкаси, "Відлуння–Плюс", 2002.–240 с.
5. Леоненков А. В. Нечеткое моделирование в среде MATLAB / А. В. Леоненков. – СПб.: БХВ–Петербург, 2003. – 736 с.
6. Дубровин В.И. Интеллектуальные средства диагностики и прогнозирования надежности авиадвигателей : монография / [В.И. Дубровин, С.А. Субботин, А.В. Богуслаев, В.К. Яценко] .– Запорожье: ОАО "Мотор–Сич", 2003.– 279 с.
7. Кричевский М.Л. Интеллектуальные методы в менеджменте. / М.Л. Кричевский – СПб.: Питер, 2005.– 304 с.
8. Субботін С. О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей : монографія / С. О. Субботін, А. О. Олійник, О. О. Олійник ; під заг. ред. С. О. Субботіна. – Запоріжжя: ЗНТУ, 2009. – 375 с.
9. Прогрессивные технологии моделирования, оптимизации и интеллектуальной автоматизации этапов жизненного цикла авиационных двигателей : монография / [А. В. Богуслаев, Ал. А. Олейник, Ан. А. Олейник, Д. В. Павленко, С. А. Субботин; под ред. Д. В. Павленко, С. А. Субботина]. – Запорожье: ОАО "Мотор Сич", 2009. – 468 с.
10. Интеллектуальные информационные технологии проектирования автоматизированных систем диагностирования и распознавания образов : монография / С. А. Субботин, Ан. А. Олейник, Е. А. Гофман, С. А. Зайцев, Ал. А. Олейник ; под ред. С. А. Субботина. – Харьков. : Компания СМІТ, 2012. – 318 с.

Додаток А

**ПРИКЛАД ОФОРМЛЕННЯ ТИТУЛЬНОГО АРКУШУ
ЗВІТУ ПРО ВИКОНАННЯ
РОЗРАХУНКОВО-ГРАФІЧНОГО ЗАВДАННЯ**

Міністерство освіти і науки, молоді та спорту України
Запорізький національний технічний університет

Кафедра програмних засобів

ЗВІТ

про виконання розрахунково-графічного завдання
з дисципліни "Назва курсу"
варіант № ____
на тему
"Формулювання теми реферату"

Виконав: ст. гр. ІОТ-410м

І. І. Іванов

Прийняв: доцент, к.т.н.

П. П. Петренко

2012

Рисунок А.1 – Титульний аркуш звіту

Додаток Б

НЕЧІТКІ СИСТЕМИ ТА НЕЙРО-НЕЧІТКІ МЕРЕЖІ У ПАКЕТІ MATLAB

Операції з нечіткою логікою у пакеті MATLAB дозволяє виконувати модуль *Fuzzy Logic Toolbox*. Він дозволяє створювати системи нечіткого логічного виведення і нечіткої класифікації в рамках середовища MatLab, з можливістю їхнього інтегрування в Simulink.

Fuzzy Logic Toolbox містить наступні категорії програмних інструментів: функції; інтерактивні модулі з графічним користувацьким інтерфейсом (з GUI); блоки для пакета Simulink; демонстраційні приклади.

Б.1 Побудова нечітких систем у діалоговому режимі за допомогою модуля Fuzzy Logic Toolbox

Модуль Fuzzy Logic Toolbox дозволяє будувати нечіткі системи двох типів: Мамдані та Сугено. У системах типу Мамдані база знань складається з правил виду “Якщо x_1 =низький та x_2 =середній, то y =високий”. У системах типу Сугено база знань складається з правил виду “Якщо x_1 =низький та x_2 =середній, то $y=a_0+a_1x_1+a_2x_2$ ”. Таким чином, основна відмінність між системами Мамдані та Сугено полягає в різних способах завдання значень вихідної змінної в правилах, що утворюють базу знань. У системах типу Мамдані значення вихідної змінної задаються нечіткими термами, у системах типу Сугено - як лінійна комбінація вхідних змінних.

Розглянемо основні етапи проектування систем типу Сугено на прикладі створення системи нечіткого логічного виведення, що моделює залежність $y=x_1^2\sin(x_2-1)$, $x_1 \in [-7; 3]$, $x_2 \in [-4,4; 1,7]$. Моделювання цієї залежності будемо здійснювати за допомогою наступної бази знань:

1. Якщо x_1 =Середній, то $y=0$;
2. Якщо x_1 =Високий та x_2 =Високий, то $y=2x_1+2x_2+1$;
3. Якщо x_1 =Високий та x_2 =Низький, то $y=4x_1-x_2$;
4. Якщо x_1 =Низький та x_2 =Середній, то $y=8x_1+2x_2+8$;
5. Якщо x_1 =Низький та x_2 =Низький, то $y=50$;
6. Якщо x_1 =Низький та x_2 =Високий, то $y=50$.

Проектування системи нечіткого логічного виведення типу Сугено складається у виконанні наступної послідовності кроків.

Крок 1. Для завантаження основного *fis*-редактора надрукуємо слово *fuzzy* у командному рядку. Після цього відкриється нове графічне вікно.

Крок 2. Виберемо тип системи. Для цього в меню *File* вибираємо в підменю *New fis...* команду *Sugeno*.

Крок 3. Додамо другу вхідну змінну. Для цього в меню *Edit* вибираємо команду *Add input*.

Крок 4. Перейменуємо першу вхідну змінну. Для цього зробимо один щиглика лівою кнопкою миші на блоці *input1*, уведемо нове позначення *x1* у поле редагування імені поточної змінної і натиснемо [Enter].

Крок 5. Перейменуємо другу вхідну змінну. Для цього зробимо один щиглика лівою кнопкою миші на блоці *input2*, уведемо нове позначення *x2* у поле редагування імені поточної змінної і натиснемо [Enter].

Крок 6. Перейменуємо вихідну змінну. Для цього зробимо один щиглика лівою кнопкою миші на блоці *output1*, уведемо нове позначення *y* у поле редагування імені поточної змінної і натиснемо [Enter].

Крок 7. Задамо ім'я системи. Для цього в меню *File* вибираємо в підменю *Export* команду *To disk* і введемо ім'я файлу, наприклад, *FirstSugeno*.

Крок 8. Перейдемо в редактор функцій належності. Для цього зробимо подвійного щиглика лівою кнопкою миші на блоці *x1*.

Крок 9. Задамо діапазон зміни змінної *x1*. Для цього надрукуємо -7 3 у полі *Range* і натиснемо [Enter].

Крок 10. Задамо функції належності змінної *x1*. Для лінгвістичної оцінки цієї змінної будемо використовувати 3 терми з трикутними функціями належності, що встановлені за замовчуванням. Задамо найменування термів змінної *x1*. Для цього робимо один щиглик лівою кнопкою миші на графіку першої функції належності. Потім надрукуємо найменування терму *Низький* у поле *Name*. Потім робимо один щиглик лівою кнопкою миші на графіку другої функції належності і вводимо найменування терму *Середній* у поле *Name*. Ще раз робимо один щиглик лівою кнопкою миші на графіку третьої функції належності і вводимо найменування терму *Високий* у поле *Name* і натиснемо [Enter].

Крок 11. Задамо функції належності змінної x_2 . Для лінгвістичної оцінки цієї змінної будемо використовувати 3 терми з трикутними функціями належності, що встановлені за замовчуванням. Для цього активізуємо змінну x_2 за допомогою щиглика лівої кнопки миші на блоці x_2 . Задамо діапазон зміни змінної x_2 . Для цього надрукуємо -4.4 1.7 у полі Range і натиснемо [Enter]. За аналогією з попереднім кроком задамо наступні найменування термів змінної x_2 : Низький, Середній, Високий.

Крок 12. Задамо лінійні залежності між входами і виходом, приведені в базі знань. Для цього активізуємо змінну y за допомогою щиглика лівої кнопки миші на блоці y . У правому верхньому куті з'явилося позначення трьох функцій належності, кожна з яких відповідає одній лінійній залежності між входами і виходом. У базі знань зазначені 5 різних залежностей: $y=50$; $y=4x_1-x_2$; $y=2x_1+2x_2+1$; $y=8x_1+2x_2+8$; $y=0$. Тому додамо ще дві залежності шляхом вибору команди Add Mfs... меню Edit. У діалоговому вікні, що з'явилося, в поле Number of MFs вибираємо 2 і натискаємо кнопку OK.

Крок 13. Задамо найменування і параметри лінійних залежностей. Для цього робимо один щиглик лівою кнопкою миші по найменуванню першої залежності mf1. Потім друкуємо найменування залежності, наприклад 50, у полі Name, і встановлюємо тип залежності – константа шляхом вибору опції Constant у меню Type. Після цього вводимо значення параметра – 50 у полі Params. Аналогічно для другої залежності mf2 уведемо найменування залежності, наприклад $8+8x_1+2x_2$. Потім укажемо лінійний тип залежності шляхом вибору опції Linear у меню Type і введемо параметри залежності 8 2 8 у полі Params. Для лінійної залежності порядок параметрів наступний: перший параметр – коефіцієнт при першій змінній, другий – при другій і т.д., і останній параметр – вільний член залежності. Аналогічно для третьої залежності mf3 уведемо найменування залежності, наприклад $1+2x_1+2x_2$, укажемо лінійний тип залежності і введемо параметри залежності 2 2 1. Для четвертої залежності mf4 уведемо найменування залежності, наприклад $4x_1-x_2$, укажемо лінійний тип залежності і введемо параметри залежності 4 -1 0. Для п'ятої залежності mf5 уведемо найменування залежності, наприклад 0, укажемо тип залежності - константа і введемо параметр залежності 0.

Крок 14. Перейдемо в редактор бази знань RuleEditor. Для цього виберемо в меню Edit команду Edit rules.... і введемо правила бази

знань. Для введення правила необхідно вибрати відповідну комбінацію термів і залежностей і натиснути кнопку Add rule.

Вікно візуалізації нечіткого логічного виведення активізується командою View rules... меню View. У поле Input указуються значення вхідних змінних, для яких виконується логічне виведення. Значення вихідної змінної розраховується як середнє зважене значення результатів виведення за кожним правилом. Для виведення цього вікна поверхні «вхід-вихід», що відповідає синтезованій нечіткій системі, необхідно використовувати команду View surface... меню View.

Розглянемо інший приклад. Нехай ми вирішуємо задачу визначення діагнозу пацієнта за значеннями симптомів, що характеризують його стан. Вхідними параметрами є симптоми пацієнта: температура тіла (нормальна - близько 36,6 °C, підвищена - більше 37°C), місце болю (у спині, у попереку), вид болю (сильний, слабкий). Вихідним параметром є діагноз пацієнта (хронічний апендицит, гострий апендицит, хронічний пієлонефрит, гострий пієлонефрит). Ми маємо визначені експертами правила (ми будемо вважати, що якщо апендицит або пієлонефрит не є гострими, то вони є хронічними):

ЯКЩО			ТО		
темпера-тура	місце болю	вид болю	апендицит	пієлонеф-рит	гострий
підвищена	у спині	сильний	немає	є	є
підвищена	у животі	сильний	є	немає	є
нормальна	у спині	слабкий	немає	є	немає
нормальна	у животі	слабкий	є	немає	немає

Необхідно побудувати систему нечіткого виведення.

1. Завантажимо пакет MATLAB та запустимо редактор FIS: fuzzy.

2. Створимо три змінні (одна вже є за замовчанням, тому додамо ще дві змінні): у головному меню редактора у підменю Edit оберемо опцію Add variables: Input. Це потрібно зробити 2 рази. Після чого створимо три вихідні змінні (подібним чином, але обираючи Add variables: Output).

3. Переміщуючи курсор миші по областях змінних можна обрати потрібну змінну та задати її назву в області Current Variable: Name.

Назвемо наші змінні: temperature - температура, painlocation - місце болю, paintype - вид болю. Також задамо назви вихідних змінних: apendicite - апендицит, pielonefrit - пієлонефрит, acute - гострий.

4. Подвійним щигликом на області якої-небудь змінної можна активізувати редактор функцій належності. За його допомогою визначимо діапазон кожної змінної та необхідні функції належності.

5. У головному меню редактора FIS оберемо в підменю Edit пункт Rules. Тепер ми маємо можливість задавати наші правила.

6. У головному меню редактора FIS оберемо в підменю View пункт Rules. Переміщуючи курсором миші лінії, що регулюють значення змінних, будемо отримувати значення вихідних змінних при відповідному наборі значень вхідних змінних.

Завантаживши файл Sample.fis із наведеним нижче текстом до редактора FIS можна переглянути описане у прикладі вирішення задачі.

Вміст файлу Sample.fis:

```
[System]
Name='Sample'
Type='mamdani'
Version=2.0
NumInputs=3
NumOutputs=4
NumRules=4
AndMethod='min'
OrMethod='max'
ImpMethod='min'
AggMethod='max'
DefuzzMethod='centroid'
[Input1]
Name='temperature'
Range=[35 42]
NumMFs=2
MF1='normal': 'trimf', [35.9 36.6 37]
MF2='high': 'smf', [36.8 37.19444444444444]
[Input2]
Name='painlocation'
Range=[0 1]
```

```

NumMFs=2
MF1='stiomach': 'trimf', [0 0 0.5]
MF2='spine': 'trimf', [0.5 1 1]
[Input3]
Name='paintype'
Range=[0 1]
NumMFs=2
MF1='acute': 'trimf', [0.5 1 1]
MF2='weak': 'trimf', [0 0 0.5]
[Output1]
Name='appendicite'
Range=[0 1]
NumMFs=1
MF1='is': 'trimf', [0.5 1 1]
[Output2]
Name='pielonefrit'
Range=[0 1]
NumMFs=1
MF1='is': 'trimf', [0.5 1 1.4]
[Output3]
Name='acute'
Range=[0 1]
NumMFs=1
MF1='is': 'trimf', [0 0 0.5]
[Output4]
Name='chronic'
Range=[0 1]
NumMFs=1
MF1='is': 'trimf', [0.5 1 1.4]
[Rules]
1 1 2, 1 0 0 1 (1) : 1
1 2 2, 0 1 0 1 (1) : 1
2 1 1, 1 0 1 0 (1) : 1
2 2 1, 0 1 1 0 (1) : 1

```

Б.2 Кластер-аналіз у середовищі Matlab

Модуль Fuzzy Logic Toolbox пакету MATLAB містить функції для виділення кластерів.

Функція *subclust* визначає координати центрів кластерів шляхом чіткої кластеризації зі зменшенням кількості кластерів.

Функція *subclust* знаходить оптимальну точку даних для визначення центра кластера ґрунтуючись на щільності оточення точок даних. Усі точки даних у межах відстані *RADII* до цієї точки видаляються, щоб визначити наступний кластер даних та його центр. Цей процес повторюється поки усі дані не знаходяться у межах відстані *RADII* до центра кластера.

$[C] = \text{SUBCLUST}(X, \text{RADII})$ кластеризує точки даних $S \times N$ матриці *X*, де *S* - кількість точок даних, *N* - кількість координат точок даних, *RADII* - значення між 0 та 1, що визначає розмір кластера в кожному з вимірювань даних, приймаючи, що дані знаходяться у межах діапазону $[0, 1]$ (Встановлення меншого радіуса кластера буде звичайно створювати більше менших за розміром кластерів. Коли *RADII* є скаляром, він застосовується до усіх вимірів даних. Коли *RADII* є вектором, він має окреме значення для кожного виміру даних), та повертає центри кластерів як рядки матриці *C*, що має розмір $V \times N$, де *V* - кількість кластерів.

$[C] = \text{SUBCLUST}(\dots, \text{XBOUNDS})$ також визначає матрицю *XBOUNDS*, розміром $2 \times N$, що використовується для нормалізації даних *X* у діапазон $[0, 1]$. Кожний стовпець *XBOUNDS* містить мінімальні та максимальні значення для відповідної розмірності даних. Якщо *XBOUNDS* - порожня матриця або не використовується, тоді за замовчуванням використовуються мінімальні та максимальні значення даних *X*.

$[C] = \text{SUBCLUST}(\dots, \text{OPTIONS})$ визначає вектор для зміни значень за замовчуванням параметрів алгоритму: *OPTIONS*(1) - коефіцієнт, що використовується для множення на значення *RADII* для визначення осередку центру кластера, у межах якого існування інших центрів кластерів заборонено; *OPTIONS*(2) - коефіцієнт прийняття, що встановлює потенціал як частку потенціалу центра першого кластера, вище якої інша точка даних буде прийнята як центр кластера; *OPTIONS*(3) - коефіцієнт відхилення, що встановлює потенціал як

частку потенціалу центра першого кластера, нижче якої інша точка даних буде відхилена як центр кластера; `OPTIONS(4)` - ознака відображення поточної інформації, якщо не встановлена як 0.

Значеннями вектора `OPTIONS` за замовчуванням є `[1.25 0.5 0.15 0]`.

Приклад використання `subclust`.

```
X1 = 10*rand(50,1);
X2 = 5*rand(50,1);
X3 = 20*rand(50,1)-10;
X = [X1 X2 X3]; % генеруємо вибірку даних
[C] = subclust(X,0.5); % знаходимо центри кластерів
```

Функція `fcm` здійснює нечітку кластеризацію на основі методу нечітких с-середніх та має формат виклику:

`[CENTER, U, OBJ_FCN] = fcm(DATA, N_CLUSTER, OPTIONS)`
де `N_CLUSTER` - кількість кластерів в наборі даних масиву `DATA`, який має розміри $S \times N$, `S` - кількість точок даних, `N` - кількість координат точок, `CENTER` - матриця з координатами центрів кластерів (кластери містяться у рядках, ознаки - у стовпцях), `U` - матриця функції належності, що містить рівні належності кожної точки масиву `DATA` до кожного кластера, `OBJ_FCN` - значення цільової функції для центрів кластерів, `OPTIONS` - необов'язковий параметр, що задає вектор опцій для процесу кластеризації: `OPTIONS(1)` - експонента для матриці `U` (за замовчуванням: 2.0), `OPTIONS(2)` - максимальна кількість ітерацій (за замовчуванням: 100), `OPTIONS(3)` - мінімально прийнятне покращення цільової функції (за замовчуванням: 10^{-5}), `OPTIONS(4)`: ознака відображення проміжних результатів (за замовчуванням: 1).

На кожній ітерації цільова функція мінімізується для знаходження кращого розташування кластерів. Процес кластеризації зупиняється, коли максимально прийнятна кількість ітерацій є досягнутою, або коли покращення цільової функції між двома послідовними ітераціями змінюється меншою ніж мінімально прийнятний приріст.

Приклад використання `fcm`.

```
data = rand(100,2); % генеруємо вибірку
[center,U,obj_fcn] = fcm(data,2); % виконуємо кластер-аналіз
```

```

plot(data(:,1), data(:,2),'o'); % зображуємо дані
на графіку
hold on;
maxU = max(U);
% Знаходимо точки з найвищим рівнем належності до
першого кластера
index1 = find(U(1,:) == maxU);
% Знаходимо точки з найвищим рівнем належності до
другого кластера
index2 = find(U(2,:) == maxU);
line(data(index1,1),data(index1,2),'marker','*','c
olor','g');
line(data(index2,1),data(index2,2),'marker','*','c
olor','r');
% Зображуємо центри кластерів на графіку
plot([center([1 2],1)], [center([1
2],2)], '*','color','k');

```

Б.3 Використання нейро-нечітких мереж

Розглянемо приклад використання навченої нейро-нечіткої мережі. Нехай ми маємо ємність, у яку наливається вода визначеної температури. Потрібно визначити температуру води через проміжок часу. Входи мережі: x_1 – температура води; x_2 – час остигання води. Вихід мережі: y – температура води після остигання. Нехай у нас є наступна база правил виду: Якщо вода – x_1 та остигає – x_2 , то температура води буде y . Подамо її у вигляді таблиці:

Номер правила	ЯКІЩО (входи)		ТО (вихід)	Вага правила
	x_1	x_2		
11	холодна	недовго	прийнятна	1
12	холодна	довго		1
13	тепла	довго		0,8
14	гаряча	довго	нормальна	0,6
21	тепла	недовго		0,5
22	гаряча	довго		0,8
31	гаряча	недовго	висока	0,9

Тут холодна, тепла, гаряча, довго, недовго – нечіткі терми (нечіткі множини):

$$\mu_{\text{хол}}(x) = \begin{cases} 1, & x \leq 15 \text{ }^{\circ}\text{C}, \\ -\frac{x}{15} + 2, & 15 < x < 30 \text{ }^{\circ}\text{C}, \\ 0, & x \geq 30 \text{ }^{\circ}\text{C}, \end{cases}$$

$$\mu_{\text{тепл}}(x) = \begin{cases} \frac{x}{15} - 1.25, & x \leq 35 \text{ }^{\circ}\text{C}, \\ -\frac{x}{15} + \frac{10}{3}, & x > 35 \text{ }^{\circ}\text{C}, \end{cases}$$

$$\mu_{\text{гар}}(x) = \begin{cases} \frac{x}{15} - \frac{8}{3}, & x \leq 40 \text{ }^{\circ}\text{C}, \\ 1, & x > 40 \text{ }^{\circ}\text{C}, \end{cases}$$

$$\mu_{\text{нед}}(x) = \begin{cases} 1, & x \leq 10 \text{ хв.}, \\ -\frac{x}{10} + 2, & x > 10 \text{ хв.}, \end{cases}$$

$$\mu_{\text{дов}}(x) = \begin{cases} \frac{x}{10} - 1, & x \leq 20 \text{ хв.}, \\ 1, & x > 20 \text{ хв.}, \end{cases}$$

Розбиття на класи для y виглядає таким чином:

прийнятна: [10; 25]

нормальна: [25; 45]

висока: [45; 60]

Таким чином: $y = 10, y_1 = 25, y_2 = 45, \bar{y} = 60$.

Нейро-нечітка мережа зображена на рис. Б.1.

Шар 1. Визначаються функції належності вхідних змінних термам. На виході $\mu^T(x_i), i = 1, 2$.

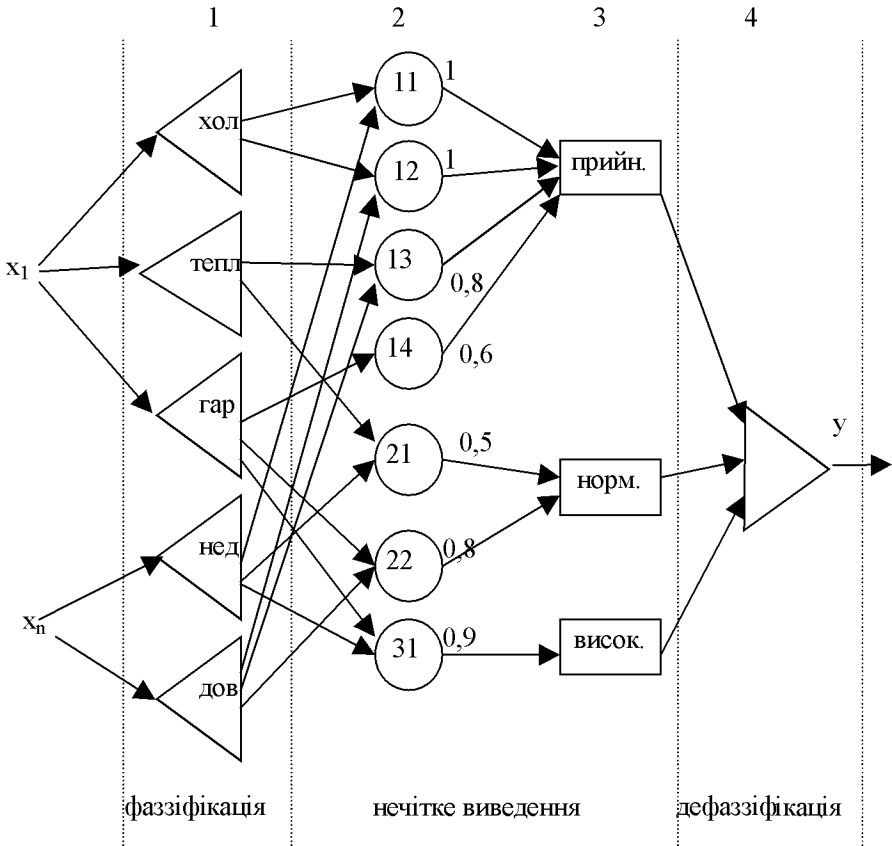


Рисунок Б.1 – Нечітка нейронна мережа

Шар 2. Вибирається мінімальний із входів нейрона. На виході $\min(\mu^T(x_i))$, $j = 1, 2, 3$ – номер класу y .

Шар 3. Вибирається максимальний із входів нейрона (вхід є добутком виходу нейрона попереднього шару і ваги правила). На виході $\max(w_{jp} \cdot \min(\mu^T(x_i)))$, $p = 1, \dots, 7$ – номер правила.

Шар 4. Визначаємо вихід:

$$y = \frac{y \cdot \mu^{d1}(y) + y_1 \cdot \mu^{d2}(y) + y_2 \cdot \mu^{d3}(y)}{\mu^{d1}(y) + \mu^{d2}(y) + \mu^{d3}(y)}.$$

Нехай $x_1 = 22$ і $x_2 = 12$, тоді:

визначення належностей термам (фаззифікація):

$$\mu_{\text{хол}}(x_1) = 0,53; \mu_{\text{тепл}}(x_1) = 0,22; \mu_{\text{гар}}(x_1) = 0;$$

$$\mu_{\text{нед}}(x_2) = 0,8; \mu_{\text{дов}}(x_2) = 0,2.$$

результати виконання нечітких правил:

$$\begin{aligned} 11: \min(0,53, 0,8) &= 0,53; \\ 12: \min(0,53, 0,2) &= 0,2; \\ 13: \min(0,22, 0,2) &= 0,2; \\ 14: \min(0, 0,2) &= 0; \\ 21: \min(0,22, 0,8) &= 0,22; \\ 22: \min(0, 0,22) &= 0; \\ 31: \min(0, 0,8) &= 0. \end{aligned}$$

результати об'єднання правил у класи:

$$\begin{aligned} d_1: \max(1 \cdot 0,53, 1 \cdot 0,2, 0,8 \cdot 0,2, 0,6 \cdot 0) &= 0,53 \\ d_2: \max(0,5 \cdot 0,22, 0,8 \cdot 0) &= 0,11; \quad d_3: \max(0,9 \cdot 0) = 0. \end{aligned}$$

Дефаззифікація:

$$y = (10 \cdot 0,53 + 25 \cdot 0,11) / (0,53 + 0,11) = 12,6.$$

Одержуємо відповідь: вода 22°C після остигання протягом 12 хвилин буде мати температуру 12,6°C.

Розглянемо інший приклад використання нейро-нечіткої мережі. Нехай ми маємо спостереження за погодою, подані наборами значень змінних: x_1 - температура повітря, °C; x_2 - вміст вологи у повітрі, %; y - кількість опадів, мм/см². Ми знаємо, що параметр y залежить від параметрів x_1 та x_2 , але ми не знаємо як. Треба побудувати нейро-нечітку апроксимацію залежності $y(x_1, x_2)$ за даними спостережень, представленими у вигляді таблиці (навчальна вибірка). Для перевірки апроксимаційних властивостей нейро-нечіткої моделі будемо використовувати тестову вибірку, подану у таблиці.

навчальна вибірка		
x ₁	x ₂	y
25	76	2.3
27	79	2.2
23.5	91	1.8
15	89	1.2
14.5	79	1.0
17	61	1.4
20	96	2.0
23	89	2.0
26.5	57	2.6
29	59	2.7

тестова вибірка		
x ₁	x ₂	y
15	0.86	0.9
25	0.47	2.5
26	0.57	2.6
15	0.98	0.9
13.5	0.68	0.9
16.5	0.93	1.6
19.5	0.67	1.9
22	0.58	2.1
25	0.82	2.0
28	0.86	2.8

У середовищі MATLAB створимо змінну навчальної вибірки *p* та змінну тестової вибірки *t*:

p=[25 76 2.3; 27 79 2.2; 23.5 91 1.8; 15 89 1.2; 14.5 79 1.0; 17 61 1.4; 20 96 2.0; 23 89 2.0; 26.5 57 2.6; 29 59 2.7];

t=[15 0.86 0.9; 25 0.47 2.5; 26 0.57 2.6; 15 0.98 0.9; 13.5 0.68 0.9; 16.5 0.93 1.6; 19.5 0.67 1.9; 22 0.58 2.1; 25 0.82 2.0; 28 0.86 2.8];

Запустимо редактор ANFIS, для чого введемо: *anfisedit*

В області Load data задамо змінні тестової вибірки (оберемо в списку Type: Training та в списку from: Worksp., після чого натиснемо кнопку Load data .. та введемо у поле input variable name: *p*) та навчальної вибірки (оберемо в списку Type:Testing та в списку from: Worksp., після чого натиснемо кнопку Load data .. та введемо у поле input variable name: *t*).

В області Generate Fis оберемо: Sub. clustering та натиснемо кнопку Generate Fis. В області Train FIS оберемо метод оптимізації Optim. method: hybrid. Задамо прийнятний рівень помилки Error tolerance: 0.000001. Задамо кількість ітерацій Epochs: 300. Натиснемо кнопку Train now. У графічній області Training Error будемо спостерігати, як змінюється помилка нейро-нечіткої мережі протягом навчання. Натиснувши кнопку Structure отримаємо зображення структури згенерованої нейро-нечіткої мережі.

Після цього отриману FIS структуру можна зберегти на диску або використовувати для прогнозування *y*, наприклад із використанням тестової вибірки *t*. Для тестування оберемо, для якої вибірки здійснюється

тестування (Training - навчальна, Testing - тестова), та натиснемо кнопку Test now. Внизу форми буде вказано середню помилку тестування (Average Error), а на графіку буде зображено синіми точками - цільові значення виходу, а червоними зірками - розрахункові значення виходу нейро-нечіткої мережі.

Б.4 FIS - структура

Базовим поняттям Fuzzy Logic Toolbox є *FIS-структура* - система нечіткого виведення (Fuzzy Inference System). FIS-структура містить усі необхідні дані для реалізації функціонального відображення “входи-виходи” на основі нечіткого логічного виведення відповідно до схеми, приведеної на рис. Б.2.

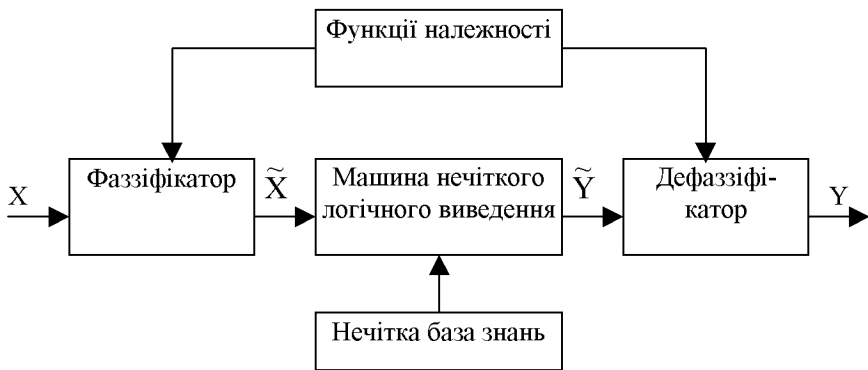


Рисунок Б.2 - Нечітке логічне виведення

Позначення: X - вхідний чіткий вектор; $\tilde{X}$ - вектор нечітких множин, що відповідає вхідному вектору X ; $\tilde{Y}$ - результат логічного виведення у виді вектора нечітких множин; Y - вихідний чіткий вектор.

Система нечіткого логічного виведення подається у робочій області MatLab у вигляді структури даних, зображеної на рис. Б.3.

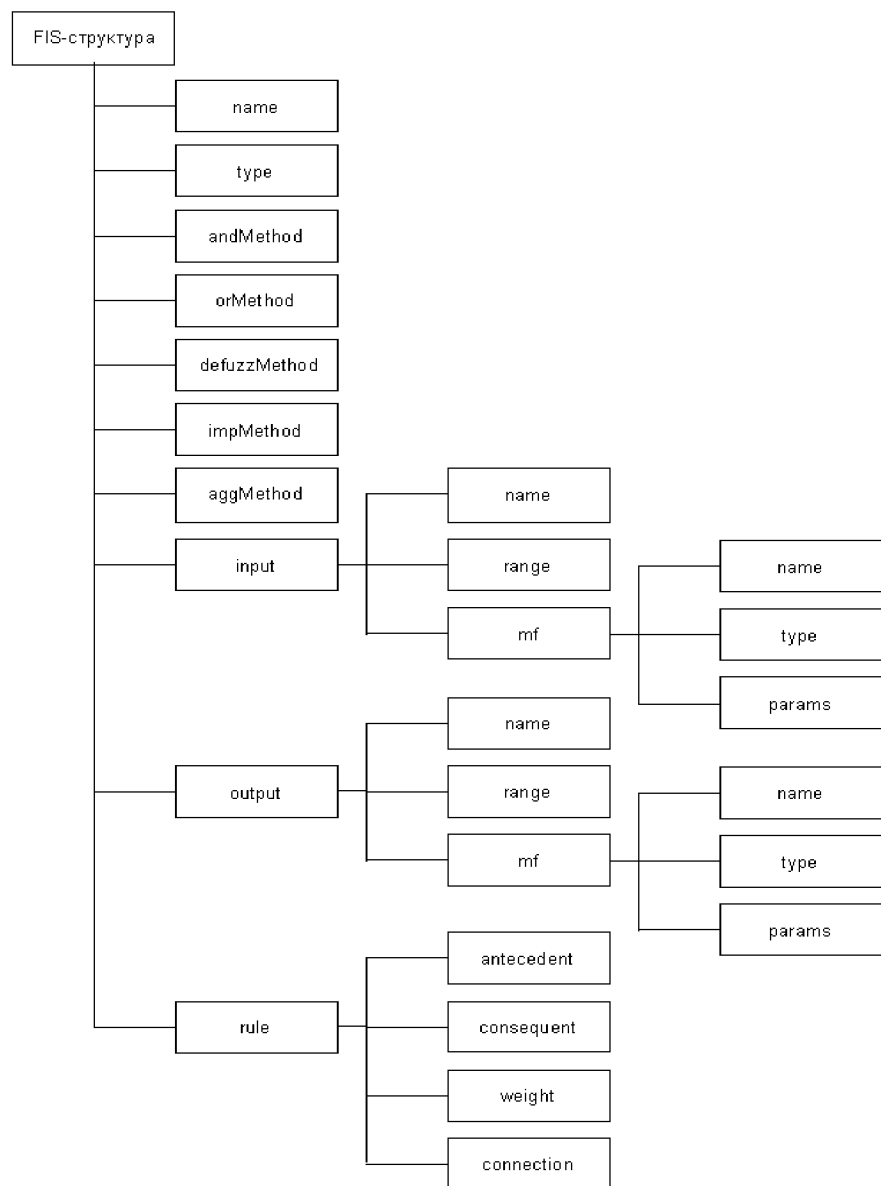


Рисунок Б.3 – FIS-структура

Існує два *способи завантаження FIS* у робочу область: зчитування з диска за допомогою функції `readfis`; передача з основного файлового редактора шляхом вибору в меню File підменю Export і команди To workspace.

Поля структури дані системи нечіткого логічного виведення призначені для збереження наступної інформації:

- name - найменування системи нечіткого логічного виведення;
- type - тип системи (припустимі значення 'Mamdani' та 'Sugeno');
- andMethod - реалізація логічної операції "ТА" (запрограмовані реалізації: 'min' – мінімум і 'prod' – множення);
- orMethod - реалізація логічної операції "АБО" (запрограмовані реалізації: 'max' – максимум і 'probor' – імовірнісне "АБО");
- defuzzMethod - метод дефазифікації (запрограмовані методи для систем типу Мамдані: 'centroid' – центр ваги; 'bisector' – медіана; 'lom' – найбільший з максимумів; 'som' – найменший з максимумів; 'mom' – середнє з максимумів; запрограмовані методи для систем типу Сугено: 'wtaver' – зважене середнє і 'wtsum' – зважена сума);
- impMethod - реалізація операції імплікації (запрограмовані реалізації: 'min' – мінімум і 'prod' – множення);
- aggMethod - реалізація операції об'єднання функцій належності вихідної змінної (запрограмовані реалізації: 'max' – максимум; 'sum' – сума і 'probor' – імовірнісне "АБО");
- input - масив вхідних змінних;
- input.name - найменування вхідної змінної;
- input.range - діапазон зміни вхідної змінної;
- input.mf - масив функцій належності вхідної змінної;
- input.mf.name - найменування функції належності вхідної змінної;
- input.mf.type - модель функції належності вхідної змінної (запрограмовані моделі: `dsigmf` - функція належності у виді різниці між двома сигмоїдними функціями; `gauss2mf` - двостороння гаусівська функція належності; `gaussmf` - гаусівська функція належності; `gbellmf` - узагальнена колоколообразна функція належності; `rimf` - пі-подібна функція належності; `psigmf` - добуток двох сигмоїдних функцій належності; `sigmf` - сигмоїдна функція належності; `smf` - s-подібна функція

належності; trapmf - трапецієподібна функція належності; trimf - трикутна функція належності; zmf - z-подібна функція належності);

- input.mf.params - масив параметрів функції належності вхідної змінної;

- output - масив вихідних змінних;

- output.name - найменування вихідної змінної;

- output.range - діапазон зміни вихідної змінної;

- output.mf - масив функцій належності вихідної змінної;

- output.mf.name - найменування функції належності вихідної змінної;

- output.mf.type - модель функції належності вихідної змінної (запрограмовані моделі для системи типу Мамдані: dsigmf - функція належності у виді різниці між двома сигмоїдними функціями; gauss2mf - двостороння гаусівська функція належності; gaussmf - гаусівська функція належності; gbellmf - узагальнена колоколообразна функція належності; pimgf - пі-подібна функція належності; psimgf - добуток двох сигмоїдних функцій належності; запрограмовані моделі для системи типу Сугено: constatnt – константа (функція належності у виді синглтона); linear – лінійна комбінація вхідних змінних);

- output.mf.params - масив параметрів функції належності вихідної змінної;

- rule - масив правил нечіткої бази знань;

- rule.antecedent - посилки правила (вказуються порядкові номери термів у порядку запису вхідних змінних. Число 0 указує на те, що значення відповідної вхідної змінної не впливає на істинність правила);

- rule.consequent - висновок правила (вказуються порядкові номери термів у порядку запису вихідних змінних. Число 0 указує на те, що правило не поширюється на відповідну вихідну змінну);

- rule.weight - вага правила. Задається числом з діапазону [0, 1];

- rule.connection - логічне зв'язування змінних усередині правила: 1 – логічне "ТА"; 2 – логічне "АБО".

Для доступу до властивостей системи нечіткого логічного виведення досить вказати ім'я відповідного поля. Наприклад, команда `FIS_NAME.rule(1).weight=0.5` встановлює вагу першого правила в 0.5, команда `length(FIS_NAME.rule)` визначає кількість правил у базі знань, а команда `FIS_NAME.input(1).mf(1).name='низький'` перейменує перший терм першої вхідної змінної в "низький".

Б.5 FIS-редактор

FIS-редактор (рис. Б.4) призначений для створення, збереження, завантаження і виведення у друк систем нечіткого логічного виведення, а також для редагування наступних властивостей: тип системи; найменування системи; кількість вхідних і вихідних змінних; найменування вхідних і вихідних змінних; параметри нечіткого логічного виведення.

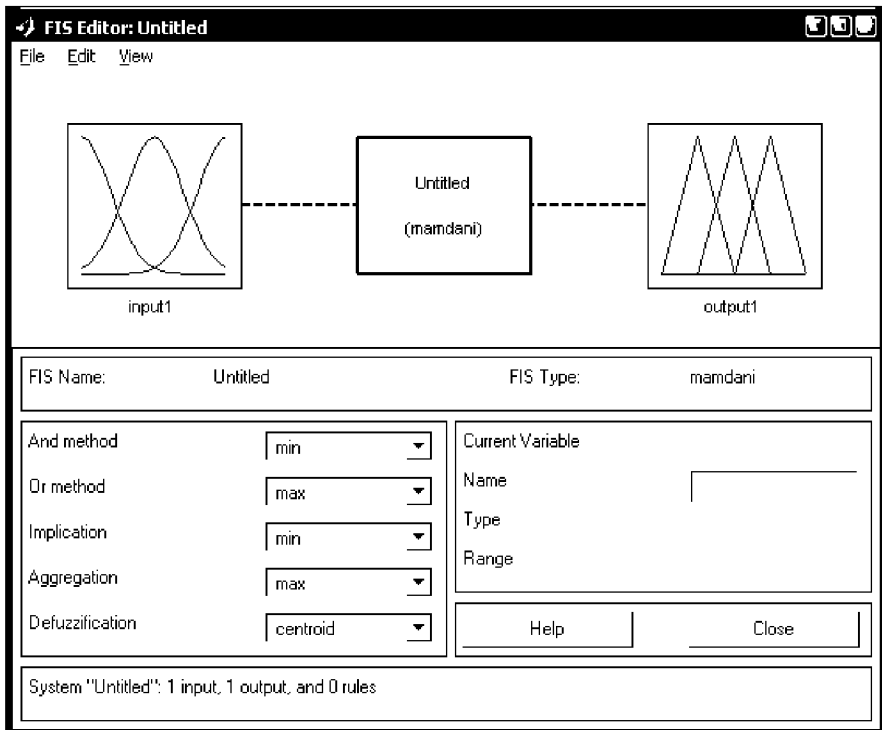


Рисунок Б.4 – FIS-редактор

Завантаження FIS-редактора відбувається за допомогою команди *fuzzy*. У результаті з'являється інтерактивне графічне вікно. У нижній частині графічного вікна FIS-редактори розташовані кнопки *Help* і

Close, що дозволяють викликати вікно довідки і закрити редактор, відповідно.

FIS-редактор містить 8 меню. Це три загальносистемних меню - File, Edit, View, і п'ять меню для вибору параметрів нечіткого логічного виведення – And Method, Or Method, Implication, Aggregation і Defuzzification.

Меню File - це загальне меню для всіх GUI-модулів використовуваних із системами нечіткого логічного виведення.

За допомогою команди New FIS... користувач має можливість створити нову систему нечіткого логічного виведення. При виборі цієї команди з'являться дві альтернативи: Mamdani і Sugeno, що визначають тип створюваної системи. Створити систему типу Mamdani можна також натисканням Ctrl+N.

За допомогою команди Import користувач має можливість завантажити раніше створену систему нечіткого логічного виведення. При виборі цієї команди з'являться дві альтернативи From Workspace... і From disk, що дозволяють завантажити систему нечіткого логічного виведення з робочої області MatLab і з диска, відповідно. При виборі команди From Workspace... з'явиться діалогове вікно, у якому необхідно вказати ідентифікатор системи нечіткого логічного виведення, що знаходиться в робочій області MatLab. При виборі команди From disk з'явиться діалогове вікно, у якому необхідно вказати ім'я файлу системи нечіткого логічного виведення. Файли систем нечіткого логічного виведення мають розширення .fis . Завантажити систему нечіткого логічного виведення з диска можна також натисканням Ctrl+N чи командою fuzzy FIS_name, де FIS_name – ім'я файлу системи нечіткого логічного виведення.

При виборі команди Export з'являться дві альтернативи To Workspace... і To disk, що дозволяють скопіювати систему нечіткого логічного виведення в робочу область MatLab і на диск, відповідно. При виборі команди To Workspace... з'явиться діалогове вікно, у якому необхідно вказати ідентифікатор системи нечіткого логічного виведення, під яким вона буде збережена в робочій області MatLab. При виборі команди To disk з'явиться діалогове вікно, у якому необхідно вказати ім'я файлу системи нечіткого логічного виведення. Скопіювати систему нечіткого логічного виведення в робочу область і на диск можна також натисканням Ctrl+T і Ctrl+S, відповідно.

Команда Print дозволяє вивести на принтер копію графічного вікна. Печатка можлива також по натисканню Ctrl+P.

Команда Close закриває графічне вікно. Закриття графічного вікна відбувається по натисканню Ctrl+W чи однократного щиглика лівої кнопки миші по кнопці Close.

Меню Edit:

Команда Undo скасовує раніше зроблену дію. Виконується також по натисканню Ctrl+Z.

Команда Add Variable... дозволяє додати в систему нечіткого логічного виведення ще одну змінну. При виборі цієї команди з'являться дві альтернативи Input і Output, що дозволяють додати вхідну і вихідну змінну, відповідно.

Команда Remove Selected Variable видаляє поточну змінну із системи. Ознакою поточної змінної є червона окантовка її прямокутника. Призначення поточної змінної відбувається за допомогою однократного щиглика лівої кнопки миші по її прямокутнику. Видалити поточну змінну можна також за допомогою натискання Ctrl+X.

Команда Membership Function... відкриває редактор функцій належностей. Ця команда може бути також виконана натисканням Ctrl+2.

Команда Rules... відкриває редактор бази знань. Ця команда може бути також виконана натисканням Ctrl+3.

Меню View - це загальне меню для всіх GUI-модулів, використовуваних із системами нечіткого логічного виведення. Дане меню дозволяє відкрити вікно візуалізації нечіткого логічного виведення (команда Rules або натискання клавіш Ctrl+5) і вікно виведення поверхні "вхід-вихід", що відповідає системі нечіткого логічного виведення (команда Surface або натискання клавіш Ctrl+6).

Меню And Method - це меню дозволяє установити наступні реалізації логічної операції "ТА": min – мінімум; prod – множення.

Користувач також має можливість установити власну реалізацію операції "ТА". Для цього необхідно вибрати команду Custom... і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію.

Меню Or Method - це меню дозволяє установити наступні реалізації логічної операції "АБО": max – множення; probor - імовірнісне "АБО".

Користувач також має можливість установити власну реалізацію операції "АБО". Для цього необхідно вибрати команду Custom... і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію.

Меню Implication - це меню дозволяє установити наступні реалізації імплікації: min – мінімум; prod – множення.

Користувач також має можливість установити власну реалізацію імплікації. Для цього необхідно вибрати команду Custom... і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію.

Меню Aggregation - це меню дозволяє установити наступні реалізації операції об'єднання функцій належності вихідної змінної: max – максимум; sum – сума; probor - імовірнісне "АБО".

Користувач також має можливість установити власну реалізацію цієї операції. Для цього необхідно вибрати команду Custom... і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію

Меню Defuzzification - це меню дозволяє вибрати метод дефазифікації. Для систем типу Мамдані запрограмовані наступні методи: centroid – центр ваги; bisector – медіана; lom – найбільший з максимумів; som – найменший з максимумів; mom – середнє з максимумів. Для систем типу Сугено запрограмовані наступні методи: wtaver – зважене середнє; wsum – зважена сума.

Користувач також має можливість установити власний метод дефазифікації. Для цього необхідно вибрати команду Custom... і в графічному вікні, що з'явилося, надрукувати ім'я функції, що реалізує цю операцію.

Б.5 Редактор функцій належності

Редактор функцій належності (Membership Function Editor) призначений для задавання наступної інформації про терми-множини вхідних і вихідних змінних: кількість термів; найменування термів; тип і параметри функцій належності, що необхідні для представлення лінгвістичних термів у вигляді нечітких множин.

Редактор функцій належності може бути викликаний з будь-якого GUI-модуля, використовуюваного із системами нечіткого логіч-

ного виведення, командою Membership Functions... меню Edit або натисканням клавіш Ctrl+2.

У FIS-редакторі відкрити редактор функцій належності можна також подвійним щикликом лівою кнопкою миші по полю вхідної або вихідної змінних. Загальний вид редактора функцій належності з ука-зівкою функціонального призначення основних полів графічного вік-на приведений на рис. Б.5.

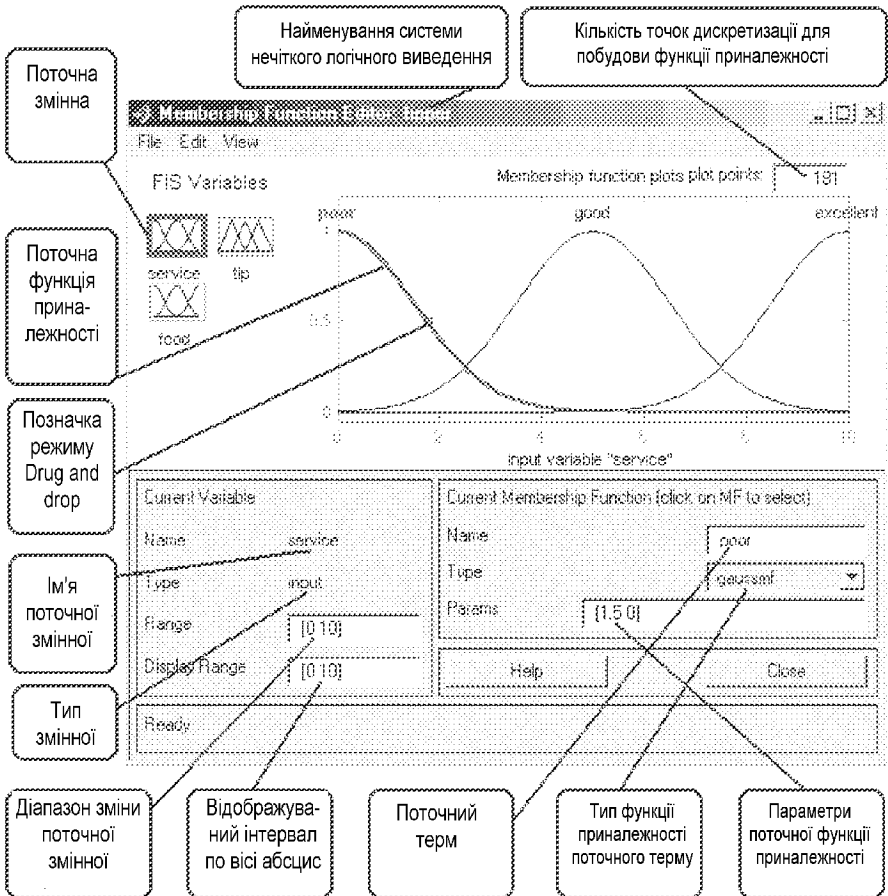


Рисунок Б.5 – Редактор функцій належності

У нижній частині графічного вікна розташовані кнопки Help і Close, що дозволяють викликати вікно довідки і закрити редактор, відповідно.

Редактор функцій належності містить чотири меню - File, Edit, View, Type і чотири вікна введення інформації – Range, Display Range, Name і Params. Ці чотири вікна призначені для завдання діапазону зміни поточної змінної, діапазону виведення функцій належності, найменування поточного лінгвістичного терму і параметрів його функції належності, відповідно.

Параметри функції належності можна підбирати й у графічному режимі, шляхом зміни форми функції належності за допомогою технології “Drug and drop”. Для цього необхідно позиціонувати курсор миші на знаку режиму “Drug and drop”, натиснути на ліву кнопку миші і не відпускаючи її змінювати форму функції належності. Параметри функції належності будуть перераховуватися автоматично.

Меню Edit:

Команда Undo скасовує раніше зроблену дію. Виконується також по натисканню Ctrl+Z.

Команда Add MFs...дозволяє додати терми в терми-множину, використовувана для лінгвістичної оцінки поточної змінної. При виборі цієї команди з'явиться діалогове вікно, у якому необхідно вибрати тип функції належності і кількість термів. Значення параметрів функцій належності будуть встановлені автоматично таким чином, щоб рівномірно покрити область визначення змінної, заданої у вікні Range. При зміні області визначення у вікні Range параметри функцій належності будуть промаштабовані.

Команда Add Custom MF...дозволяє додати одних лінгвістичний терм, функція належності якого відрізняється від убудованих. Після вибору цієї команди з'явиться графічне вікно, у якому необхідно надрукувати лінгвістичний терм (поле MF name), ім'я функції належності (поле M-File function name) і параметри функції належності (поле Parameter list).

Команда Remove Selected MF видаляє поточний терм із терм-множини поточної змінної. Ознакою поточної змінної є червона окантовка її прямокутника. Ознакою поточного терму є червоний колір його функції належності. Для вибору поточного терму необхідно провести позиціонування курсору миші на графіку функції належності і зробити щиглик лівою кнопкою миші.

Команда Remove All MFs видаляє всі терми з терм-множини поточної змінної.

Команда FIS Properties... відкриває FIS-редактор. Ця команда може бути також виконана натисканням Ctrl+1.

Команда Rules... відкриває редактор бази знань. Ця команда може бути також виконана натисканням Ctrl+3.

Меню Type дозволяє установити тип функцій належності термів, використовуваних для лінгвістичної оцінки поточної змінної.

Б.6 Редактор бази знань

Редактор бази знань (Rule Editor) призначений для формування і модифікації нечітких правил. Редактор бази знань може бути викликаний з будь-якого GUI-модуля, використовуваного із системами нечіткого логічного виведення, командою Rules... меню Edit або натисканням клавіш Ctrl+3. У FIS-редакторі відкрити редактор бази знань можна також подвійним щигликом лівою кнопкою миші по прямокутнику з назвою системи нечіткого логічного виведення, розташованого в центрі графічного вікна.

Загальний вид редактора бази знань із указівкою функціонального призначення основних полів графічного вікна приведений на рис. Б.6.

У нижній частині графічного вікна розташовані кнопки Help і Close, що дозволяють викликати вікно довідки і закрити редактор, відповідно.

Редактор функцій належності містить чотири системних меню File, Edit, View, Options, меню вибору термів вхідних і вихідних змінних, поля установки логічних операцій ТА, АБО, НЕ і ваг правил, а також кнопки редагування і перегляду правил.

Для введення нового правила в базу знань необхідно за допомогою миші вибрати відповідну комбінацію лінгвістичних термів вхідних і вихідних змінних, установити тип логічного зв'язування (ТА або АБО) між змінними усередині правила, установити чи наявність відсутності логічної операції НЕ для кожної лінгвістичної змінний, увести значення вагового коефіцієнта правила і натиснути кнопку Add Rule. За замовчуванням установлені наступні параметри: логічне зв'язування змінних усередині правила – ТА; логічна операція НЕ – відсутня; значення вагового коефіцієнта правила – 1.

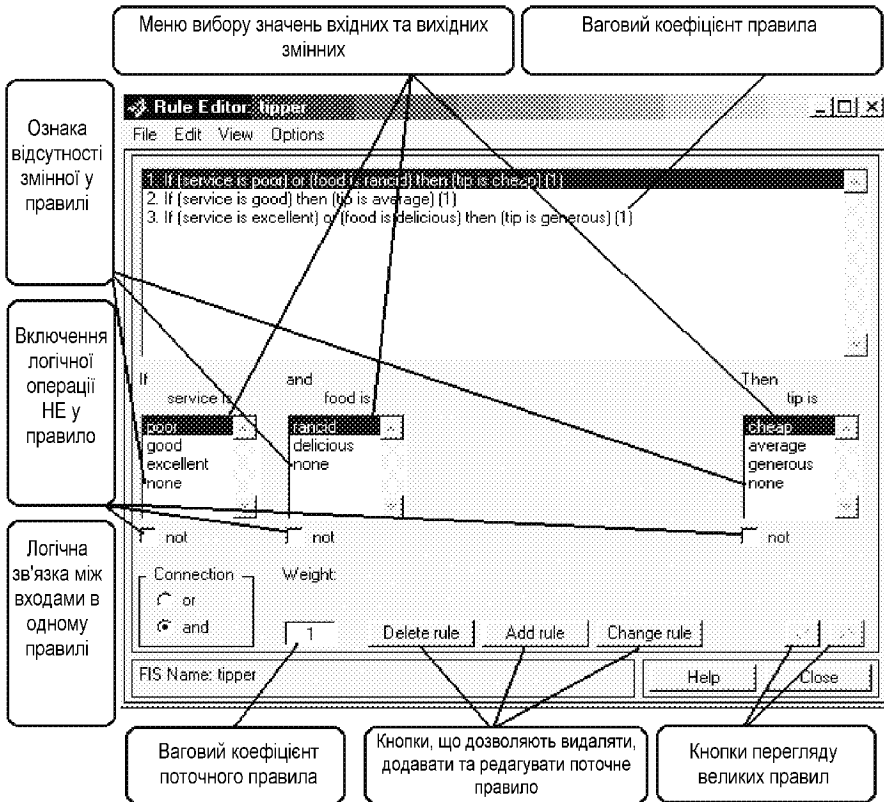


Рисунок Б.6 – Редактор бази знань

Можливі випадки, коли істинність правила не змінюється при довільній значенні деякої вхідної змінної, тобто ця змінна не впливає на результат нечіткого логічного виведення в даній області факторного простору. Тоді як лінгвістичне значення цієї змінної необхідно установити none.

Для видалення правила з бази знань необхідно зробити однократний щиглик лівою кнопкою миші на цьому правилі та натиснути кнопку Delete Rule.

Для модифікації правила необхідно зробити однократний щиглик лівою кнопкою миші на цьому правилі, потім установити необхідні параметри правила і натиснути кнопку Edit Rule.

Меню Edit:

Команда Undo скасовує раніше зроблену дію. Виконується також по натисканню Ctrl+Z.

Команда FIS Properties...відкриває FIS-редактор. Ця команда може бути також виконана натисканням Ctrl+1.

Команда Membership Function...відкриває редактор функцій належностей. Ця команда може бути також виконана натисканням Ctrl+2.

Меню Options дозволяє установити мову і формат правил бази знань. При виборі команди Language з'явиться список мов English (Англійська), Deutsch (Німецька), Francais (Французька), з якого необхідно вибрати одну.

При виборі команди Format з'явиться список можливих форматів правил бази знань: Verbose - лінгвістичний; Symbolic – логічний; Indexed – індексований.

Б.7 Візуалізація нечіткого логічного виведення

Візуалізація нечіткого логічного виведення здійснюється за допомогою GUI-модуля *Rule Viewer*. Цей модуль дозволяє проілюструвати хід логічного виведення за кожним правилом, одержання результатуючої нечіткої множини і виконання процедури дефазифікації. Rule Viewer може бути викликаний з будь-якого GUI-модуля, використовуюваного із системами нечіткого логічного виведення, командою View rules ... меню View чи натисканням клавіш Ctrl+4. Вид Rule Viewer для системи логічного виведення tipper із указівкою функціонального призначення основних полів графічного вікна приведений на рис. Б.7.

Rule Viewer містить чотири меню - File, Edit, View, Options, два поля введення інформації – Input і Plot points та кнопки прокручування зображення вліво - вправо (left-right), униз (up-down). У нижній частині графічного вікна розташовані також кнопки Help і Close, що дозволяють викликати вікно довідки і закрити редактор, відповідно. Кожне правило бази знань представляється у виді послідовності горизонтально розташованих прямокутників. При цьому перші два прямокутники відображають функції належностей термів посилки правила (Якщо-частина правила), а останній третій прямокутник відповідає функції належності терму-наслідку вихідної змінної (То-частина правила).

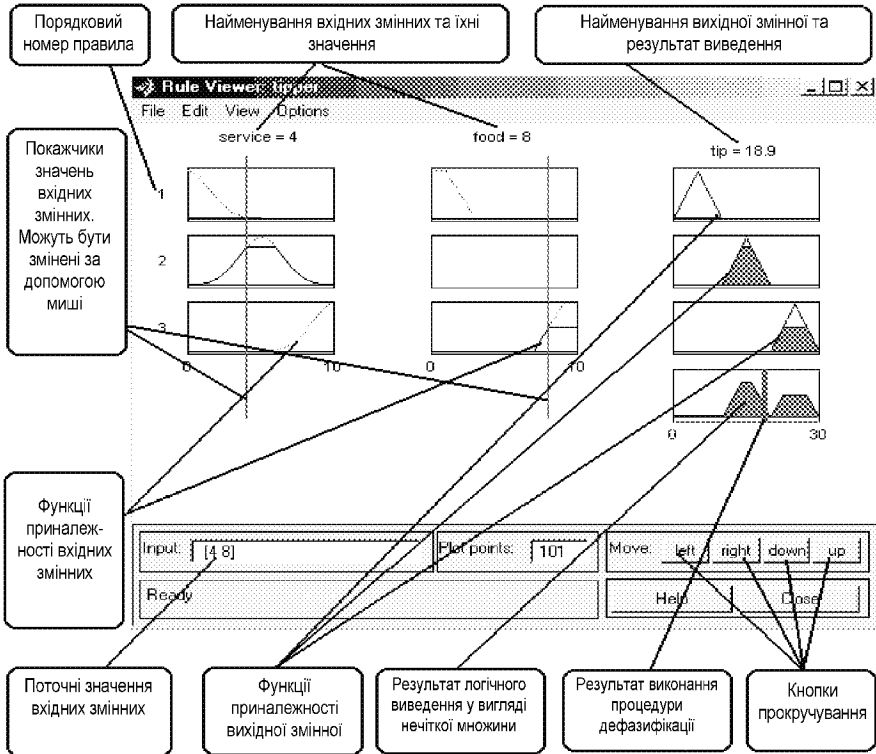


Рисунок Б.7 – Візуалізація логічного виведення для системи tipper за допомогою Rule Viewer

Порожній прямокутник у візуалізації другого правила означає, що в цьому правилі посилка за змінною food відсутня (food is none). Жовте заливання графіків функцій належностей вхідних змінних указує наскільки значення входів, відповідають термам даного правила. Для виведення правила у форматі Rule Editor необхідно зробити однократний щиглик лівої кнопки миші по номері відповідного правила. У цьому випадку зазначене правило буде виведено в нижній частині графічного вікна.

Блакитне заливання графіка функції належності вихідної змінної являє собою результат логічного виведення у вигляді нечіткої множини за даним правилом. Результуючу нечітку множину, що відповідає

логічному виведенню за всіма правилами, показано в нижньому прямокутнику останнього стовпця графічного вікна. У цьому ж прямокутнику червона вертикальна лінія відповідає чіткому значенню логічного виведення, отриманого в результаті дефазифікації.

Уведення значень вхідних змінних може здійснюватися двома способами: шляхом введення чисельних значень у поле Input; за допомогою миші, шляхом переміщення ліній-показчиків червоного кольору.

В останньому випадку необхідно позиціонувати курсор миші на червоній вертикальній лінії, натиснути на ліву кнопку миші і не відпускаючи неї перемістити показчик на потрібну позицію. Нове чисельне значення відповідної вхідної змінної буде перелічено автоматично і виведене у вікно Input.

У поле Plot points задається кількість крапок дискретизації для побудови графіків функцій належності. Значення за замовчуванням – 101.

Меню Edit:

Команда FIS Properties...відкриває FIS-редактор. Ця команда може бути також виконана натисканням Ctrl+1.

Команда Membership Functions...відкриває редактор функцій належностей. Ця команда може бути також виконана натисканням Ctrl+2.

Команда Rules...відкриває редактор бази знань. Ця команда може бути також виконана натисканням Ctrl+3.

Меню Options містить тільки одну команду Format, що дозволяє установити один з наступних форматів виведення обраного правило в нижній частині графічного вікна: Verbose - лінгвістичний; Symbolic – логічний; Indexed – індексований.

Б.8 ANFIS-редактор

ANFIS є аббревіатурою Adaptive Neuro-Fuzzy Inference System – адаптивна нейро-нечітка система. ANFIS-редактор дозволяє автоматично синтезувати з експериментальних даних нейро-нечіткої мережі. Нейро-нечітку мережу можна розглядати як один з різновидів систем нечіткого логічного виведення типу Сугено. При цьому функції належності синтезованих систем налагоджено (навчено) так, щоб мінімізувати відхилення між результатами нечіткого моделювання й експериментальних даних.

Завантаження ANFIS-редактора здійснюється за командою `anfisedit`. У результаті виконання цієї команди з'явиться графічне вікно, зображене на рис. Б.8. На цьому ж рисунку зазначено функціональні області ANFIS-редактора, опис яких приведено нижче.

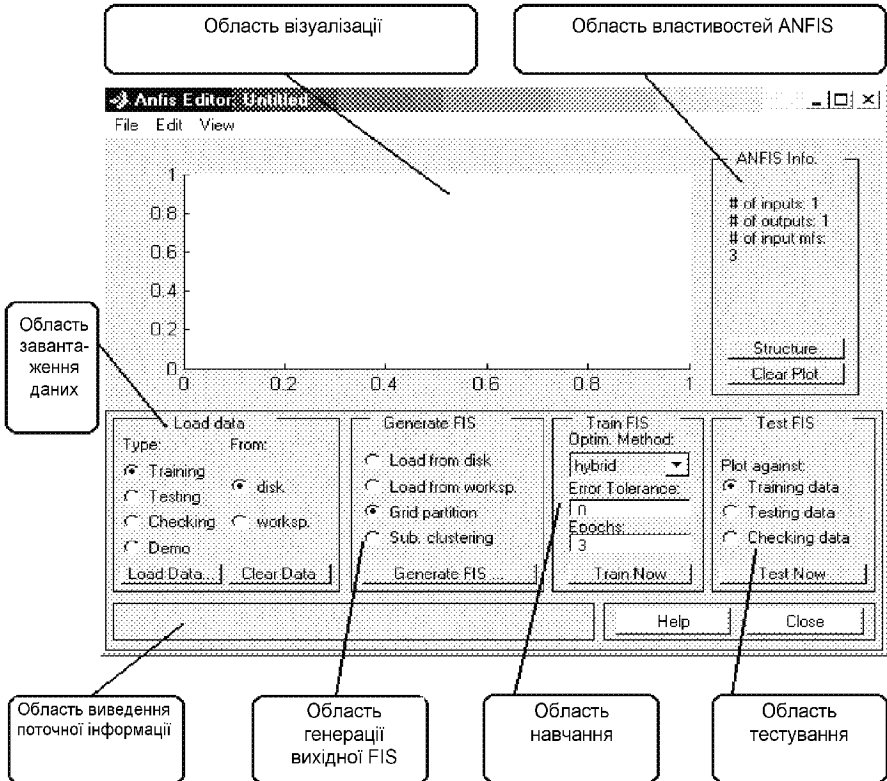


Рисунок Б.8 – Основне вікно ANFIS-редактора

ANFIS-редактор містить 3 верхніх меню - File, Edit і View, область візуалізації, область властивостей ANFIS, область завантаження даних, область генерування вихідної системи нечіткого логічного виведення, область навчання, область тестування, область виведення поточної інформації, а також кнопки Help і Close, що дозволяють викликати вікно довідки і закрити ANFIS-редактор, відповідно.

Меню Edit:

Команда Undo скасовує раніше зроблену дію. Виконується також по натисканню Ctrl+Z.

Команда FIS Properties...відкриває FIS-редактор. Ця команда може бути також виконана натисканням Ctrl+1.

Команда Membership Functions...відкриває редактор функцій належності. Ця команда може бути також виконана натисканням Ctrl+2.

Команда Rules...відкриває редактор бази знань. Ця команда може бути також виконана натисканням Ctrl+3.

Команда Anfis...відкриває ANFIS-редактор. Ця команда може бути також виконана натисканням Ctrl+3. Помітимо, що дана команда, запущена з ANFIS-редактора не приводить до виконання яких-небудь дій, так цей редактор уже відкритий. Однак, у меню Edit інших GUI-модулів, використовуваних із системами нечіткого логічного виведення, додається команда Anfis...,що дозволяє відкрити ANFIS-редактор з цих модулів.

Область візуалізації містить два типи інформації: при навчанні системи – крива навчання у виді графіка залежності помилки навчання від порядкового номера ітерації; при завантаженні даних і тестуванні системи – експериментальні дані і результати моделювання.

Експериментальні дані і результати моделювання виводяться у виді множини крапок у двовимірному просторі. При цьому по вісі абсцис відкладається порядковий номер рядка даних у вибірці (навчальної, тестової або контрольної), а по осі ординат - значення вихідної змінної для даного рядка вибірки. Використовуються наступні маркери: блакитна крапка (.) – тестова вибірка; блакитна окружність (o) – навчальна вибірка; блакитний плюс (+) – контрольна вибірка; червона зірочка (*) – результати моделювання.

В області *властивостей* ANFIS (ANFIS info) виводиться інформація про кількість вхідних і вихідних змінних, про кількість функцій належності для кожної вхідний змінний, а також про кількість рядків у вибірках. У цій області розташовані дві кнопки Structure і Clear Plot.

Натискання кнопки Structure відчиняє нове графічне вікно, у якому система нечіткого логічного виведення представляє у виді нейро-нечіткої мережі. Натискання кнопки Clear Plot дозволяє очистити область візуалізації.

В області завантаження даних (Load data) розташовані: меню вибору типу даних (Type), що містить альтернативи (Traning - навчальна вибірка; Testing - тестова вибірка; Checking - контрольна вибірка; Demo - демонстраційний приклад); меню вибору джерела даних (From), що містить альтернативи (disk – диск; worksp. - робоча область MatLab); кнопка завантаження даних Load Data..., по натисканню якої з'являється діалогове вікно вибору файлу, якщо завантаження даних відбувається з диска, або вікно введення ідентифікатора вибірки, якщо завантаження даних походить з робочої області; кнопка очищення даних Clear Data.

В області генерування (Generate FIS) розташовані меню вибору способу створення вихідної системи нечіткого логічного виведення. Меню містить наступні альтернативи: Load from disk – завантаження системи з диска; Load from worksp. – завантаження системи з робочої області MatLab; Grid partition - генерування системи по методу ґрат (без кластеризації); Sub. clustering – генерування системи за методом субкластеризації.

В області також розташована кнопка Generate, по натисканню якої генерується вихідна система нечіткого логічного виведення.

При виборі Load from disk з'являється стандартне діалогове вікно відкриття файлу.

При виборі Load from worksp. з'являється стандартне діалогове вікно введення ідентифікатора системи нечіткого логічного виведення.

При виборі Grid partition з'являється вікно введення параметрів методу ґрат, у якому потрібно вказати кількість термів для кожен вхідний змінної і тип функцій належності для вхідних і вихідної змінних.

При виборі Sub. clustering з'являється вікно введення наступних параметрів методу субкластеризації: Range of influence – рівні впливу вхідних змінних; Squash factor – коефіцієнт пригнічення; Accept ratio – коефіцієнт, що встановлює у скільки разів потенціал даної точки повинний бути вище потенціалу центра першого кластера для того, щоб центром одного з кластерів була призначена розглянута точка; Reject ratio - коефіцієнт, що встановлює у скількох разів потенціал даної точки повинний бути нижче потенціалу центра першого кластера, щоб розглянута точка була виключена з можливих центрів кластерів.

В області навчання (Train FIS) розташовані меню вибору методу оптимізації (Optim. method), поле завдання необхідної точності навчання (Error tolerance), поле завдання кількості ітерацій навчання

(Epochs) і кнопка Train Now, натискання якого запускає режим навчання. Проміжні результати навчання виводяться в область візуалізації й у робочу область MatLab. У ANFIS-редакторі реалізовані два методи навчання: backpropa – метод зворотного поширення помилки, заснований на ідеях методу найшвидшого спуску; hybrid – гібридний метод, що поєднує метод зворотного поширення помилки з методом найменших квадратів.

В області тестування (Test FIS) розташовані меню вибору вибірки і кнопка Test Now, по натисканню по якій відбувається тестування нечіткої системи з виведення результатів в область візуалізації.

Область виведення поточної інформації: у цій області виводиться найбільш істотна поточна інформація, наприклад, повідомлення про закінчення виконання операцій, значення помилки чи навчання тестування і т.п.

Б.9 Функції Fuzzy Logic Toolbox

Функції, що входять до модуля Fuzzy Logic Toolbox можуть бути викликані з командного рядка. Для отримання переліку функцій слід ввести команду: help fuzzy. Наведемо короткий огляд функцій модуля Fuzzy Logic.

Редактори з графічним інтерфейсом користувача: anfisedit - інструмент для навчання та тестування ANFIS; findcluster - інструмент для кластеризації; fuzzy - базовий редактор FIS; mfeedit - редактор функцій належності; ruleedit - редактор та аналізатор правил; ruleview - демонстратор правил та діаграм нечіткого виведення; surfview - демонстратор вихідної поверхні.

Функції належності: dsigmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, zmf.

Модуль Fuzzy Logic Toolbox пакету MATLAB включає 11 убудованих функцій належності, що використовують такі основні функції: кусочно-лінійну; гаусівський розподіл; сигмоїдну криву; квадратичну та кубічну криві.

Для зручності імена всіх убудованих функцій належності закінчуються на mf. Виклик функції належності здійснюється в такий спосіб: *patemf(x, params)*, де patemf – найменування функції належності; x – вектор, для координат якого необхідно розрахувати значення функції належності; params – вектор параметрів функції належності.

Найпростіші функції належності *трикутна* (trimf) і *трапецієподібна* (trapmf) формуються з використанням кусочно-лінійної апроксимації. Трапецієподібна функція належності є узагальненням трикутної, вона дозволяє задавати ядро нечіткої множини у виді інтервалу. У випадку трапецієподібної функції належності можлива наступна зручна інтерпретація: ядро нечіткої множини – оптимістична оцінка; носій нечіткої множини – песимістична оцінка.

Дві функції належності – *симетрична гаусівська* (gaussmf) і *двостороння гаусівська* (gaussmf) формуються з використанням гаусівського розподілу. Функція gaussmf дозволяє задавати асиметричні функції належності. Узагальнена колоколообразна функція належності (gbellmf) за своєю формою схожа на гаусівські. Ці функції належності часто використовуються в нечітких системах, тому що на всій області визначення вони є гладкими і приймають ненульові значення.

Функції належності sigmf, dsigmf, psigmf засновані на використанні *сигмоїдної кривої*. Ці функції дозволяють формувати функції належності, значення яких починаючи з деякого значення аргументу і до $+\infty$ рівні 1. Такі функції зручні для завдання лінгвістичних термів типу «високий» або «низький».

Поліноміальна апроксимація застосовується при формуванні функцій zmf, pимf і smf, графічні зображення яких схожі на функції sigmf, dsigmf, psigmf, відповідно.

У Fuzzy Logic Toolbox передбачена можливість для користувача створення власної функції належності. Для цього необхідно створити m-функцію, що містить два вхідних аргументи – вектор, для координат якого необхідно розрахувати значення функції належності і вектор параметрів функції належності. Вихідним аргументом функції повинний бути вектор ступенів належності.

Функції FIS: addmf - додає функцію належності до FIS; addrule - додає правило до FIS; addvar - додає змінну до FIS; defuzz - дефузифікує функцію належності; evalfis - здійснює обчислення нечіткого виведення; evalmf - обчислює функцію належності; gensurf - генерує поверхню виходу FIS; getfis - повертає властивості нечіткої системи; mf2mf - транслює параметри між функціями приналежності; newfis - створює нову FIS; parsrule - аналізує нечіткі правила; plotfis - показує діаграму "вхід-вихід" для FIS; plotmf - показує усі функції належності для однієї змінної; readfis - завантажує FIS з диску; rmmf - видаляє функцію належності з FIS; rmvar - видаляє змінну з FIS; setfis - вста-

новлює властивості нечіткої системи; showfis - показує анотовану FIS; showrule - відображує правила FIS; writefis - зберігає FIS на диску.

Функція `output=evalfis(input, fis, numofpoints)`. виконує обчислення для вибірки екземплярів за допомогою вказаної нейро-нечіткої мережі. Результатом є обчислені виходи функції, яку апроксимує нейро-нечітка мережа. Аргументи та результат функції: `input` - входи обчислюваної вибірки; `fis` - нейро-нечітка мережа; `numofpoints` - кількість точок для проведення дефазифікації (рекомендується брати значення не менше 100, зменшення цього параметру прискорює процес обчислень, але зменшує точність); `output` - обчислені виходи;

Вибірки даних та нейро-нечіткі мережі, з якими працюють описані функції повинні зберігатися в робочій області MATLAB (в оперативній пам'яті). В стовпцях матриць, що представляють собою вибірки, зберігаються значення входів (ознак) та виходів, в рядках - екземпляри вибірки.

Для завантаження вибірок з диску або запису на диск необхідно використовувати команду MATLAB: `s=load(filename)`, яка завантажує вміст файлу в змінну `s`. Функція `save(filename, s)` зберігає змінну `s` у файл.

Для можливості використання створеної нейро-нечіткої мережі в наступних сеансах роботи або використання мережі з попередніх слід використовувати такі функції модуля Fuzzy Logic Toolbox: `writefis(fis,filename)` - зберігає нейро-нечітку мережу `fis` в файл; `fis=readfis(filename)` - завантажує з файлу нейро-нечітку мережу в змінну `fis`.

Передові технології: `anfis` - функція навчання для системи Сугено; функції кластер-аналізу: `fcm`, `genfis1`, `genfis2`; `subclust`.

Різні функції: `convertfis` - перетворює нечітку матрицю структури версії 1.0 у матрицю структури версії 2.0; `discfis` - дискретизує FIS; `evalmmf` - використовується для обчислення множинних функцій належності; `fstrvcat` - поєднує матриці різного розміру; `fuzarith` - функція нечіткої арифметики; `findrow` - шукає рядки матриці, що відповідають вхідному рядку; `genparam` - генерує початкові параметри передумов для навчання ANFIS; `probor` - імовірнісне АБО; `sugmax` - максимальний вихідний діапазон для системи Сугено.

Функція `C = fuzarith(X, A, B, OPERATOR)` реалізує базові операції нечіткої логіки та повертає нечітку множину `C` як результат застосування оператора `OPERATOR` до нечітких множин `A` та `B` з уні-

версальної множини X . Змінні A , B та X мають бути векторами однакової розмірності. OPERATOR має бути одним з рядків: 'sum' - сума, 'sub' - вирахування, 'prod' - добуток, 'div' - ділення. Нечітка множина C , яка повертається, є вектор-стовпцем тієї ж довжини, що й A та B . Зауважимо, що ця функція використовує інтервальну арифметику та передбачає, що: A та B є опуклими нечіткими множинами; функції належності для A та B поза X є нулем.

Допоміжні функції графічного користувальницького інтерфейсу: cmfdlg - додає діалог вибору функцій належності; cmthdlg - додає діалог вибору методу виведення; fisgui - дискрипторне посилання на інтерфейсні засоби модуля Fuzzy Logic Toolbox; gfmfdlg - генерує FIS з використанням діалогу методу ґрат; mfdlg - додає діалог функції належності; mfdrag - перетягування функції належності за допомогою миші; popundo - відновлює зі стеку останні зміни (відміняє останні дії); pushundo - передає поточні дані у стек відновлення; savedlg - діалог запису перед закриттям; statmsg - зображує повідомлення у полі статусу; updtfis - оновлює засоби графічного інтерфейсу Fuzzy Logic Toolbox; wsdlg - діалог "відкриття з" / "збереження до" робочої області.

Додаток В

ПЕРЕЛІК ТЕМ ДЛЯ ВИКОНАННЯ РЕФЕРАТИВНОЇ ЧАСТИНИ РОЗРАХУНКОВО-ГРАФІЧНОГО ЗАВДАННЯ

1. Технології інтелектуального аналізу даних.
2. Методи витягу знань з даних.
3. Вирішення завдань технічного і біомедичного діагностування на основі нейро-нечітких мереж.
4. Програмні засоби з елементами штучного інтелекту на основі нейро-нечітких мереж.
5. Нейро-нечіткі моделі об'єктів управління і проектування.
6. Методи обробки результатів моделювання.
7. Критерії оцінювання точності математичних моделей.
8. Методи нечіткого виведення.
9. Типи та застосування нейро-нечітких мереж.
10. Методи навчання нейро-нечітких мереж.
11. Відмінності між нейронними та нейро-нечіткими мережами.
12. Принципи побудови нейро-нечітких моделей.
13. Основні визначення та методи теорії розпізнавання образів.
14. Принципи створення гібридних інтелектуальних систем.
15. Критерії аналізу методів побудови нейро-нечітких мереж.