ТЕМА 1. Історія розвитку та застосування нечіткої математики при математичному та комп'ютерному моделюванні

Вивчивши цю тему, студент повинен знати:

- історію зародження нечіткої математики;

- імена засновників теорії нечітких множин та нечіткої логіки;

- застосування нечіткої математики в промисловості та різних областях знань;

- способи представлення чітких множин;

- основні поняття теорії множин;

- функції приналежності та методи їх побудови

- практичне застосування нечіткої логіки;

- операції з нечіткою логікою у пакеті Matlab.

Питання лекції

Вступ

1. Історія виникнення теорії нечітких множин
2. Нечіткість знань
3. Основні поняття теорії множин
   - 3.1 Способи представлення чітких множин
   - 3.2 Потужність множини, види множин
   - 3.3 Поняття універсальної множини
4. Функції приналежності та методи їх побудови
   - 4.1 Способи представлення нечітких множин
   - 4.2 Нормування нечітких множин
   - 4.3 Носій нечіткої множини
   - 4.4 Зрізи нечіткої множини
   - 4.5 Відношення між множинами
5. Нечіткі множини в системах керування
6. Практичне застосування нечіткої логіки
7. Операції з нечіткою логікою у пакеті Matlab

Вступ

У даний час не можна назвати галузь людської діяльності, в якій у тій чи іншій мірі не використовувалися б методи моделювання. Особливо це стосується сфери управління різними системами, де основними є процеси прийняття рішень на основі одержаної інформації. Гіпотези та аналогії, що відображають реальний, об’єктивно існуючий світ, повинні володіти наочністю або зводитися до зручних для дослідження логічних схем, що спрощують міркування, логічні побудови та дозволяють проводити експерименти, уточнюючи природу явищ, називаються моделями.

Модель – це об’єкт-заступник об’єкта-оригіналу, що забезпечує вивчення деяких властивостей оригіналу. Заміщення одного об’єкта іншим для одержання інформації про найважливіші властивості об’єкта-оригіналу за допомогою об’єкта-моделі називається **моделюванням**.

Таким чином, **моделювання** може бути визначене як уявлення об’єкта моделлю для отримання інформації про цей об’єкт шляхом проведення експериментів з його моделлю.

Теорія заміщення одних об’єктів (оригіналів) іншими об’єктами (моделями) і дослідження властивостей об’єктів на їхніх моделях називається **теорією моделювання**.

Найбільш вражаючою властивістю людського інтелекту є здатність приймати правильні рішення в умовах неповної і нечіткої інформації. Побудова моделей, які відтворюють мислення людини і використання їх у комп'ютерних системах на сьогодні є однією з найважливіших проблем науки. . Основною причиною появи теорії нечітких множин стала наявність нечітких і наближених міркувань під час опису людиною процесів, систем, об'єктів.

1. Історія виникнення теорії нечітких множин

Теорія нечітких множин (fuzzy sets theory) веде свій початок з 1965 р., коли професор Лотфі Заде (Lotfi Zadeh) з університету Берклі опублікував основну роботу "Fuzzy Sets" в журналі "Information and Control". Концепція нечіткої множини зародилася у Заде "як незадоволеність математичними методами класичної теорії систем, яка змушувала домагатися штучної точності, недоречної в багатьох системах реального світу, особливо в так званих гуманістичних системах, що включають людей". Прикметник «fuzzy» (нечіткий, розмитий) введено в назву нової теорії з метою відокремлення від традиційної чіткої математики, що оперує з чіткими поняттями: «належить – не належить», «істина – хибність».

Початком практичного застосування теорії нечітких множин можна вважати 1975р., коли Е.Мамдані (E. Mamdani) і Ассіліан побудували перший нечіткий контролер для управління простим паровим двигуном. У 1982 Холмблад і Остергад розробили перший промисловий нечіткий контролер, який був впроваджений в управління процесом випалу цементу на заводі в Данії. Успіх першого промислового контролера, заснованого на нечітких лінгвістичних правилах "якщо - то" привів до сплеску інтересу до теорії нечітких множин серед математиків і інженерів. Трохи пізніше Бартоломеєм Коско (B. Kosko) була доведена теорема про нечіткої апроксимації, згідно з якою будь-яка математична система може бути апроксимована системою, заснованої на нечіткій логіці. Іншими словами, за допомогою природничо-мовних висловлювань-правил "якщо - то", з подальшою їх формалізацією засобами теорії нечітких множин, можна скільки завгодно точно відобразити довільний взаємозв'язок "вхід-вихід" без використання складного апарату диференціального й інтегрального числення, традиційно застосовуваного в управлінні та ідентифікації.

Системи з нечіткою логікою доцільно застосовувати для складних процесів, коли експертні знання про об’єкт або про процес можна сформулювати тільки в лінгвістичній формі.

Системи, що базуються на нечіткій логіці, застосовувати недоцільно, якщо необхідний результат може бути отриманий яким-небудь іншим (стандартним) шляхом, або якщо для об’єкта або процесу вже знайдена адекватна і легко досліджувана математична модель.

Основні недоліки систем з нечіткою логікою:

- вихідний набір нечітких правил, що постулюються, сформулюється експертом-людиною і може виявитися неповним або суперечливим;

- вид і параметри функцій належності, що описують вхідні і вихідні змінні системи, вибираються суб’єктивно і можуть виявитися такими, що цілком не відображають реальну дійсність.

Системи, засновані на нечітких множинах, розроблені і успішно впроваджені в таких областях, як: управління технологічними процесами, управління транспортом, в автомобільної, аерокосмічної та транспортної промисловості, в області виробів побутової техніки, у сфері фінансів, аналізу та прийняття управлінських рішень медичній діагностиці, технічній діагностиці, у фінансовому менеджменті, біржовому прогнозуванні, розпізнаванню образів та багатьох інших. Спектр застосування дуже

широкий - від відеокамер і побутових пральних машин до засобів наведення ракет ППО і управління бойовими вертольотами. Практичний досвід розробки систем нечіткого логічного висновку свідчить, що терміни і вартість їх проектування значно менше, ніж при використанні традиційного математичного апарату, при цьому забезпечується необхідний рівень робастности і прозорості моделей.

Отже, крім вище наведеного, слід зазначити, що перш ніж нечіткий підхід до моделювання складних систем отримав визнання у всьому світі, минуло не одне десятиліття з моменту зародження теорії нечітких множин. І на цьому шляху розвитку нечітких систем прийнято виділяти три періоди.

Перший період (кінець 60-х – початок 70 рр.) характеризується розвитком теоретичного апарату нечітких множин (Л. Заде, Е. Мамдані, Беллман).

У другому періоді (70-80-ті роки) з'являються перші практичні результати в області нечіткого управління складними технічними системами (парогенератор з нечітким керуванням). Одночасно почали приділяти увагу питанням побудови експертних систем, побудованих на нечіткій логіці, розробці нечітких контролерів. Нечіткі експертні системи для підтримки прийняття рішень знаходять широке застосування у медицині та економіці.

Нарешті, в третьому періоді, що триває з кінця 80-х років і продовжується в даний час, з'являються пакети програм для побудови нечітких експертних систем, а сфери застосування нечіткої логіки помітно розширюються. Тріумфальна хода нечіткої логіки по світу почалася після доказу наприкінці 80-х Бартоломеєм Коско знаменитої теореми FAT (Fuzzy Approximation Theorem). У бізнесі та фінансах нечітка логіка отримала визнання після того, як у 1988 році експертна система на основі нечітких правил для прогнозування фінансових індикаторів єдина передбачила біржовий крах. І кількість успішних фазі-застосувань нині обчислюється тисячами.

2. Нечіткість знань

При розробці інтелектуальних систем знання про конкретну предметну область, для якої створюється система, рідко бувають повними й абсолютно достовірними. Навіть кількісні дані, отримані шляхом досить точних експериментів, мають статистичні оцінки вірогідності, надійності, значимості і т. д. Інформація, якою заповнюються експертні системи, отримується у

результаті опитування експертів, думки яких є суб'єктивними і можуть розходитися. Поряд із кількісними характеристиками в базах знань інтелектуальних систем повинні зберігатися якісні показники, евристичні правила, текстові знання і т. д. При обробці знань із застосуванням механізмів формальної логіки виникає протиріччя між нечіткими знаннями і чіткими методами логічного виведення. Розв'язати це протиріччя можна шляхом подолання нечіткості знань (коли це можливо) або використанням спеціальних методів подання й обробки нечітких знань.

Зміст терміна **нечіткість** багатозначний та включає такі основні компоненти:

- недетермінованість виведень,
- багатозначність,
- ненадійність,
- неповнота,
- неточність.

Недетермінованість виведень - це характерна риса більшості систем штучного інтелекту. Недетермінованість означає, що заздалегідь шлях вирішення конкретної задачі в просторі її станів визначити неможливо. Тому в більшості випадків методом проб і помилок обирається деякий ланцюжок логічних висновків, що узгоджуються з наявними знаннями, а у випадку якщо він не призводить до успіху, організується перебір з поверненням для пошуку іншого ланцюжка і т. д. Такий підхід припускає визначення деякого первісного шляху. Для вирішення подібних задач запропоновано багато евристичних методів. Недетермінованість виведень варто враховувати при розробці ефективних способів подання і збереження знань, а також при побудові методів пошуку й обробки знань, що дозволяють одержати рішення задачі за найменше число кроків. Для побудови таких методів звичайно застосовуються евристичні метазнання (знання про знання).

Багатозначність інтерпретації - звичайне явище в задачах розпізнавання. При розумінні природної мови серйозними проблемами стають багатозначність змісту слів, їхня підпорядкованість, порядок слів у реченні і т. п. Проблеми розуміння змісту виникають у будь-якій системі, які взаємодіють з користувачем природною мовою. Розпізнавання графічних образів також пов'язано з вирішенням проблеми багатозначної інтерпретації. При комп'ютерній обробці знань багатозначність необхідно усувати шляхом вибору правильної інтерпретації, для цього розроблено спеціальні методи.

Ненадійність знань і виведень означає, що для оцінки вірогідності знань не можна застосувати двобальну шкалу (1 - абсолютно достовірні знання, 0 - недостовірні знання). Для більш тонкої оцінки вірогідності знань

застосовується імовірнісний підхід, заснований на теоремі Байєса, і інші методи (наприклад, метод виведення з використанням коефіцієнтів упевненості). Широке застосування на практиці одержали нечіткі виведення, які будуються на базі нечіткої логіки, що веде своє походження від теорії нечітких множин.

Неповнота знань і немонотонна логіка. Абсолютно повних знань не існує, оскільки процес пізнання є нескінченним. У зв'язку з цим стан бази знань повинний змінюватися з часом. На відміну від простого додавання інформації, як у базах даних, при додаванні нових знань виникає небезпека одержання суперечливих висновків: тобто висновки, отримані з використанням нових знань, можуть спростовувати ті, що були отримані раніше. Ще гірше, якщо нові знання будуть знаходитися в протиріччі зі старими, тоді механізм виведення може стати непрацездатним.

Більшість експертних систем першого покоління були засновані на моделі закритого світу, обумовленій застосуванням апарату формальної логіки для обробки знань.

Модель закритого світу припускає твердий добір знань, що включаються в базу, а саме: база знань заповнюється виняткова вірними поняттями, а усе, що ненадійно або невиразно, свідомо вважається помилковим. Така модель має обмежені можливості подання знань і ховає у собі небезпеку одержання протиріч при додаванні нової інформації. Недоліки моделі закритого світу пов'язані з тим, що формальна логіка виходить з передумови, відповідно до якої набір визначених у системі аксіом (знань) є повним (теорія є повною, якщо кожний її факт можна довести, виходячи з аксіом цієї теорії). Для повного набору знань справедливість раніше отриманих виведень не порушується з додаванням нових фактів. Ця властивість логічних виведень називається **монотонністю**. На жаль, реальні знання, що закладаються в експертні системи, вкрай рідко бувають повними.
Неточність знань. Відомо, що кількісні дані (знання) можуть бути неточними, при цьому існують кількісні оцінки такої неточності (довірчий інтервал, рівень значимості, ступінь адекватності і т. д.). Лінгвістичні знання також можуть бути неточними. Для врахування неточності лінгвістичних знань використовується теорія нечітких множин. Фактично нечіткість може бути ключем до розуміння здатності людини справлятися з задачами, що занадто складні для вирішення на ЕОМ. Розвиток досліджень в області нечіткої математики призвів до появи нечіткої логіки і нечітких виведень, що виконуються з використанням знань, представлених нечіткими множинами, нечіткими відношеннями, нечіткими відповідностями і т. д.

3. Основні поняття теорії множин

Множина – це просте математичне поняття, тобто не має визначення, а пояснюється лише прикладами. Наприклад, множина точок на дошці, книжок на полиці, студентів в групі і т. д. Під множиною розуміють сукупність елементів будь-якої природи (точок, книг, студентів), що називають елементами даної множини, що володіють якою-небудь загальною для цієї множини властивістю (знаходження на дошці, на полиці, в складі групи). Засновник теорії множин Г. Кант так трактував поняття множини: «Велике, мислиме як єдине ціле».

Множини прийнято позначати великими буквами латинського алфавіту, наприклад: A, B, C і т. д., а його елементи – відповідно малими буквами, наприклад: a, b, c і т. д. Часто використовують індексовані імена множин і їх елементів, наприклад: A_1, B_1, C_1 і т. д., a_1, a_2, a_3 і т. д.

3.1 Способи представлення чітких множин

Для представлення чітких множин використовують наступні три способи.

1). **Шляхом перерахування всіх елементів множини.** Формат запису $A = \{a_1, a_2, \dots, a_n\}$, де A – ім'я множини, $a_1, a_2, \dots, a_n$ – імена елементів цієї множини, $n-$ кількість елементів в множині. При цьому кожен елемент в списку є оригінальним (без повторів).

2). **Шляхом визначення характеристичної властивості елементів множини.** Формат запису $A = \{a: P(a)\}$, де A – ім'я множини, a – узагальнене ім'я елемента, $P(a)$ – предикат, який є логічною умовою або процедурою для перевірки того чи належить даний елемент цій множині або ні.

3). **Графічний спосіб.** Для цього використовують кола Ейлера або діаграми Вена. Множина представляється кругом, в якому будь-яка точка всередині цього круга представляє його елемент, а поза цим кругом – елемент, який не належить цій множині. Нижче на рисунку 1.1 множина A

представлена кругом Ейлера. Тут точка x належить множині A, а точка y не належить множині A. Формальний запис приналежності - $x \in A$ і $y \notin A$.

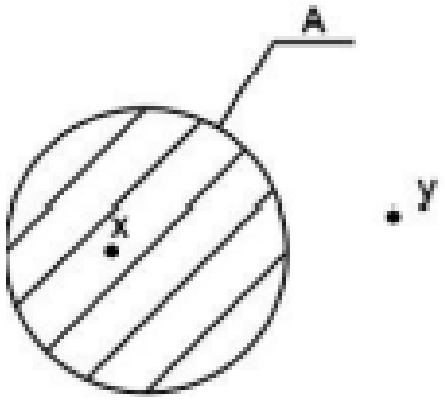

Рисунок 1.1 – Приклад графічного представлення чіткої множини

3.2 Потужність множини, види множин

Потужність або об'єм множини $A = \{a_1, a_2, \dots, a_n\}$ – це числова характеристика, визначається кількістю елементів цієї множини. Потужність позначається прямими дужками, тобто $|A| = n$.

Всі множини можна розділити на наступні види.

1). Якщо $n = 0$, то множина називається пустою. Для позначення пустої множини використовують спеціальний символ $\emptyset$, наприклад, $A = \emptyset$.

2). Якщо $n < \infty$, то і множину називають скінченною.

3). Якщо $n \to \infty$, то і множина називається нескінченно зліченна або нескінченно незліченна (для неперервних множин).

4). Скінченні і нескінченні множини діляться на впорядковані і неупорядковані множини. В упорядкованих множинах на відміну від неупорядкованих множин порядок перерахунку елементів важливий, кожен елемент займає в списку цілком визначене місце.

3.3 Поняття універсальної множини

Універсальна множина – ця чітка множина, яку використовують для побудови будь-яких інших множин. Для універсальної множини прийнято використовувати наступний запис: $U = \{u_1, u_2, \dots, u_n\}$. Універсальну множину можна використовувати в наступних аспектах.

1). Всеосяжна універсальна множина, що включає всі об'єкти навколишнього світу.

2). Універсальна множина в рамках деякої практичної задачі (перед розв'язком задачі необхідно обов'язково визначити цю множину).

4 Функція приналежності, способи її обчислення

Поняття функції приналежності увів засновник теорії нечітких множин Л. Заде [1,2]. Воно принципіальним чином відрізняється від поняття характеристичної властивості елементів, яке використовувалося раніше для побудови чітких множин.

Функція приналежності встановлює відповідність між елементами універсальної множини $U = \{u_1, u_2, \dots, u_n\}$ і числовими значеннями їх *степенем приналежності* деякій новій множині A $\mu_A(u)$ на відрізку $[0,1]$. Значення функції приналежності $\mu_A(u)$ для деякого елемента $u \in U$ показує, в якій мірі, в якій степені цей елемент належить множині A.

Якщо степені приналежності $\mu_A(u)$ приймають лише два значення 0 або 1, то множина А є чіткою, в протилежному випадку ця множина є нечіткою (степені приналежності можуть приймати будь-які значення на відрізку [0,1], наприклад, 0,2 або 0,8).

Для обчислення значень функції приналежності при побудові множини використовують наступні способи.

1). **Прямі (експертні) методи.** В цих методах розв'язок про степінь приналежності приймає експерт, який тим самим виражає свою думку на основі досвіду, що в нього є.

2). **Непрямі методи.** В цих методах степінь приналежності визначається на основі зміни властивостей елементів.

Слідує зауважити, що використання різних методів приводить до встановлення цілком конкретних (невипадкових) числових значень степенів приналежності.

4.1 Способи представлення нечітких множин

Для представлення нечітких множин використовують наступні три способи.

1). **Шляхом перерахунку всіх елементів множин.** Формат запису $A = \mu_A(u_1)/u_1 + \mu_A(u_2)/u_2 + \cdots + \mu_A(u_n)/u_n$, де A – ім'я множини, $u_1, u_2, \dots u_n$ – імена елементів універсальної множини, на якому побудована множина A, $\mu_A(u_1), \mu_A(u_2), \dots, \mu_A(u_n)$ – степені їх приналежності до множини A, n – кількість елементів в універсальній множині. Знак «+» в цій формі означає не сумування, а об'єднання елементів у множину. При перерахунку пропускаються ті елементи універсальної множини, степені приналежності яких рівні 0. Часто використовують наступну звернуту форму запису: $A = \sum_{i=1}^{n} \mu_A(u_i)/u_i$.

2). Шляхом визначення функції приналежності. Формат запису $\mu_A(u): P(u)$ – предикат, який є логічною умовою або процедурою для оцінки того, в якій степені даний елемент належить множині A.

3). **Графічний спосіб.** Для цього використовується *діаграма Заде*. Діаграма Заде нечіткої множини A представляє собою фігуру під ламаною лінією, побудованої на основі функції приналежності $\mu_A(u)$. На рисунку 1.2 наведено приклад діаграми Заде, в якій представлена нечітка множина $A = 0{,}2/u_1 + 0{,}5/u_2 + 1/u_3 + 0{,}6/u_4 + 0{,}8/u_5 + 0{,}2/u_8$, побудована на основі універсальної множини $U = \{u_1, \dots, u_8\}$. Значення функції приналежності 0,5 є найбільш нечітким і визначає *лінію перегину* діаграми. Чим ближче степінь приналежності деякого елемента до лінії перегину, тим більш нечітким є цей

елемент. Чим ближче степінь приналежності деякого елемента до 0 або 1, тим більш чітким є цей елемент. Для прикладу, наведеного на рис. 1.2, найбільш чітким елементом в множині A є u_2, а найбільш чіткими – u_3, u_6, u_7.

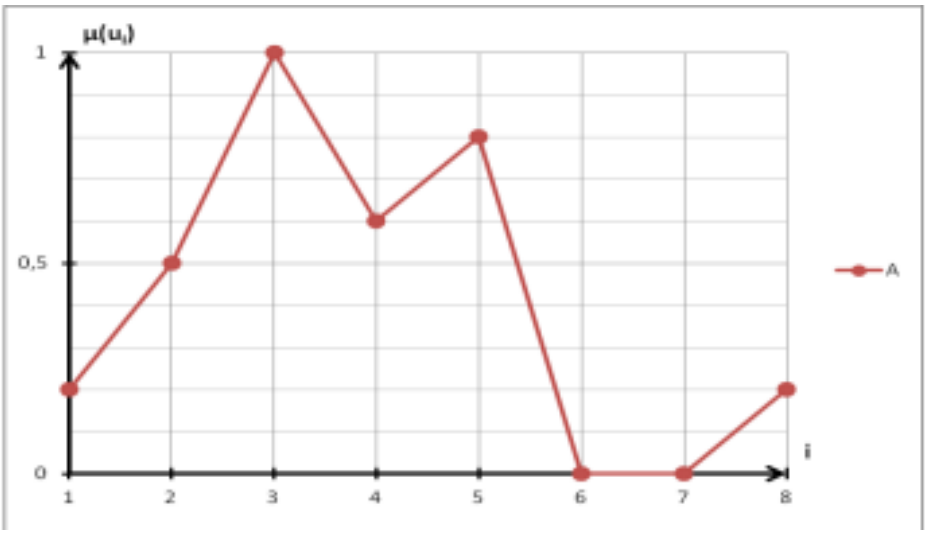

Рисунок 1.2 – Приклад графічного представлення нечіткої множини

Приклад 1.1. Задана універсальна множина $U = \{1,2,\dots,10\}$. На ній треба побудувати дві множини A і X, які представлені наступними функціями приналежності $\mu_A(u): u < 7, \mu_X(u): u$ трішки менше, чим 7.

Функції приналежності для множин A і X є логічними умовами, їх перевірка встановлює степені приналежності для кожного елемента заданої універсальної множини U. Причому перша умова є чіткою (так або ні), а друга умова – нечіткою, що вимагає застосування того чи іншого методу обчислення степенів приналежності. Наприклад, деякий експерт виразив свою думку відносно степенів приналежності так, як показано в таблиці нижче.

u	1	2	3	4	5	6	7	8	9	10
$\mu_A(u)$	1	1	1	1	1	1	0	0	0	0
$\mu_X(u)$	0	0,1	0,3	0,5	0,7	0,9	0	0	0	0

Перерахуємо елементи цих множин з використанням відповідних форматів запису.

$$A = \{1, 2, 3, 4, 5, 6\}$$

$$X = 0{,}1/2 + 0{,}3/3 + 0{,}5/4 + 0{,}7/5 + 0{,}9/6$$

4.2 Нормування нечітких множин

Нечітка множина $\mathrm{A} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \mu_{\mathrm{A}}(\mathrm{u_i})/\mathrm{u_i}$ називається **нормальною**, якщо в ній є хоча б один елемент універсальної множини зі степінню приналежності 1 (тобто $\exists u \in U: \mu_A(u) = 1$), в протилежному випадку множина A називається **субнормальною**.

Будь-яка субнормальна множина A можна привести до вигляду нормальної множини A' за допомогою наступної **операції нормування**

$\forall u \in U: \mu_{A'}(u) = \frac{\mu_A(u)}{MAX}$, де $MAX = \max\limits_{i=1,..,n}(\mu_A(u_i))$.

Приклад 1.2. Множина X із приклада 1.1 є субнормальною, для її нормування визначимо спочатку $MAX = \max\limits_{i=1,..,10}(\mu_A(u_i)) = 0{,}9$ і побудуємо нормальну множину X'.

u	1	2	3	4	5	6	7	8	9	10
$\mu_X(u)$	0	0,1	0,3	0,5	0,7	**0,9**	0	0	0	0
$\mu_{X'}(u)$	0	1/9	1/3	5/9	7/9	1	0	0	0	0

4.3 Носій нечіткої множини

Носієм або **супортом** нечіткої множини $\mathrm{A} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \mu_{\mathrm{A}}(\mathrm{u_i})/\mathrm{u_i}$ називається чітка множина $supp(A)$, яке складається тільки із таких елементів універсальної множини U, для яких виконується умова $\mu_{\mathrm{A}}(\mathrm{u}) > 0$.

Приклад 1.3. Множина $X = 0{,}1/2 + 0{,}3/3 + 0{,}5/4 + 0{,}7/5 + 0{,}9/6$ із прикладу 1.1 побудована на універсальній множині $U = \{1,2,\dots,10\}$. Тоді її носієм буде $supp(A) = \{2, 3, 4, 5, 6\}$.

4.4 Зрізи нечіткої множини

Нехай задане деяке число $0 < \alpha \leq 1$. **Множиною α-рівня** або **α-зрізом** нечіткої множини $\mathrm{A} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \mu_{\mathrm{A}}(\mathrm{u_i})/\mathrm{u_i}$ називається чітка множина A_α, яка складається тільки з таких елементів універсальної множини U, для яких виконується умова $\mu_{\mathrm{A}}(\mathrm{u}) \geq \alpha$.

Приклад 1.4. Множина $X = 0{,}1/2 + 0{,}3/3 + 0{,}5/4 + 0{,}7/5 + 0{,}9/6$ побудована на універсальній множині $U = \{1{,}2, \ldots, 10\}$. Тоді його α-зрізом з $\alpha = 0{,}3$ буде множина $X_{0{,}3} = \{3{,}4{,}5{,}6\}$, а α-зрізом з $\alpha = 0{,}7$ – множина $X_{0{,}7} = \{5{,}6\}$.

4.5 Відношення між множинами

Нехай на універсальній множині U побудовані дві множини $\mathrm{A} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \mu_{\mathrm{A}}(\mathrm{u_i})/\mathrm{u_i}$ і $B = \sum_{\mathrm{i}=1}^{\mathrm{n}} \mu_{\mathrm{B}}(\mathrm{u_i})/\mathrm{u_i}$. Говорять, що множину B включено в множину A (B є підмножиною A) тоді і тільки тоді, коли виконується наступна умова:

$$\forall u \in U: \mu_A(u) \geq \mu_B(u). \qquad (1.1)$$

Ця умова є нестрогою і записується як $B \subseteq A$.

Зауваження.

1). Якщо $\forall u \in U: \mu_A(u) \geq \mu_B(u)$, то множина B строго включена в A і є її **власною підмножиною**. В цьому випадку слідує використовувати запис $B \subset A$.

2). Якщо $\forall u \in U: \mu_A(u) = \mu_B(u)$, то множини B і A **рівні між собою** (окремий випадок відношення включення). В цьому випадку слідує використовувати запис $B = A$.

3). Будь-яка множина, побудована на універсальній множині U, є її підмножиною.

4). Діаграма Заде множини не нижче діаграми будь-якої її підмножини.

5). Для двох зрізів нечіткої множини A вірне наступне твердження

$$\alpha_1 > \alpha_2 \Longrightarrow A_{\alpha_1} \subseteq A_{\alpha_2}. \qquad (1.2)$$

Приклад 1.5. Множина $X = 0{,}1/2 + 0{,}3/3 + 0{,}5/4 + 0{,}7/5 + 0{,}9/6$ побудована на універсальній множині $U = \{1,2,\dots,10\}$. Тоді множини Y і Z в таблиці нижче є її підмножинами (виконується умова (1.1)), причому Z є власною підмножиною, тобто $Y \subseteq X$ і $Z \subset X$.

u	1	2	3	4	5	6	7	8	9	10
$\mu_X(u)$	0	0,1	0,3	0,5	0,7	0,9	0	0	0	0
$\mu_Y(u)$	0	0,1	0,3	0,4	0,5	0,7	0	0	0	0
$\mu_Z(u)$	0	0	0,1	0,3	0,5	0,6	0	0	0	0

Приклад 1.6. Множина $\mathrm{X} = 0{,}1/2 + 0{,}3/3 + 0{,}5/4 + 0{,}7/5 + 0{,}9/6$ побудована на універсальній множині $\mathrm{U} = \{1,2,\dots,10\}$. Побудуємо її зрізи за умовою $\alpha_1 > \alpha_2 > \alpha_3$. Наприклад, $\mathrm{X}_{0,7} = \{5,6\}$, $\mathrm{X}_{0,3} = \{3,4,5,6\}$, $\mathrm{X}_{0,2} = \{3,4,5,6\}$. Очевидно, що у відповідність з умовою (1.2) $\mathrm{X}_{0,7} \subseteq \mathrm{X}_{0,3} \subseteq \mathrm{X}_{0,2}$, причому в цьому прикладі $\mathrm{X}_{0,7} \subset \mathrm{X}_{0,3}$ і $\mathrm{X}_{0,7} \subset \mathrm{X}_{0,2}$, а $\mathrm{X}_{0,2} = \mathrm{X}_{0,3}$

5. Нечіткі множини в системах керування

Розглянемо приклад застосування теорії нечітких множин в системах керування.

Вдалим застосуванням теорії нечітких множин є контролери нечіткої логіки. Їх функціонування дещо відрізняється від роботи звичайних контролерів; для опису системи замість диференційних рівнянь використовуються знання експертів. Ці знання можуть бути виражені за допомогою лінгвістичних змінних, які описані нечіткими множинами.

Фазифікація - зіставлення множини значень *x* з її функцією приналежності *M (x)*, тобто переведення значень *x* в нечіткий формат (приклад з терміном молодий).

Дефазифікація - процес, зворотний до фазифікації.

Всі системи з нечіткою логікою функціонують за одним принципом: показання вимірювальних приладів фазифікуються (переводяться в нечіткий формат), обробляються, дефазифікуються і у вигляді звичних сигналів подаються на виконавчі пристрої.

Ступінь приналежності - це не ймовірність, т.к. невідома функція розподілу, немає повторюваності експериментів. Так, якщо взяти з прикладу прогнозу погоди дві взаємовиключні події: буде дощ і не буде і присвоїти їм деякі ранги, то сума цих рангів необов'язково буде дорівнювати 1, але якщо рівність все-таки є, то нечітка множина вважається нормованою. Значення функції приналежності М (х) можуть бути взяті тільки з апріорних знань, інтуїції (досвіду), опитування експертів.

Загальна структура нечіткого мікроконтролера

Загальна структура мікроконтролера, що використовує нечітку логіку, показана на рис. Вона містить у своєму складі наступні складові:

- Блок фазіфікації.
- База знань.
- Блок рішень.
- Блок дефазіфікації.

Блок фазіфікації перетворює чіткі величини, які виміряні на виході об'єкта керування у нечіткі величини, що описані лінгвістичними змінними в базі знань.

Блок рішень використовує нечіткі умовні (if - then) правила, що закладено в **базу знань**, для перетворення нечітких вхідних даних в керуючі впливи, які мають також нечіткий характер.

Блок дефазіфікації перетворює нечіткі дані з виходу блоку рішень в чіткі величини, які використовуються для керування об'єктом.

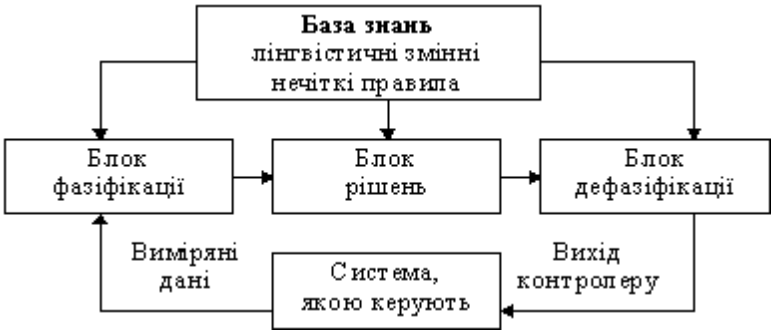

Рис. 1. Загальна структура нечіткого мікроконтролера

Розглянемо випадок керування мобільним роботом, задачею якого є об'їзд перешкод. Введемо дві лінгвістичні змінні: ДИСТАНЦІЯ (відстань від робота до перешкоди) і НАПРЯМОК (кут між подовжньою віссю робота та напрямком до перешкоди).

Розглянемо лінгвістичну змінну ДИСТАНЦІЯ. Значеннями її можна визначити терми ДАЛЕКО, СЕРЕДНЬО, БЛИЗЬКО і ДУЖЕ БЛИЗЬКО. Для фізичної реалізації лінгвістичної змінної необхідно визначити точні фізичні значення термів цієї змінної. Нехай змінна ДИСТАНЦІЯ може приймати будь-які значення з діапазону від нуля до нескінченності. Відповідно до теорії нечітких множин, в такому випадку кожному значенню відстані із зазначеного діапазону може бути поставлене у відповідність деяке число від нуля до одиниці, що визначає ступінь приналежності даної фізичної відстані (припустимо 40 см) до того чи іншого терму лінгвістичної змінної ДИСТАНЦІЯ.

Ступінь приналежності визначаємо функцією приналежності M(d), де d-відстань до перешкоди. В нашому випадку відстань 40 см. Можна задати ступінь приналежності до терму ДУЖЕ БЛИЗЬКО, що дорівнює 0,7 , а до терму БЛИЗЬКО - 0,3 (рис. 2.). Конкретне визначення ступеня приналежності визначається експертами.

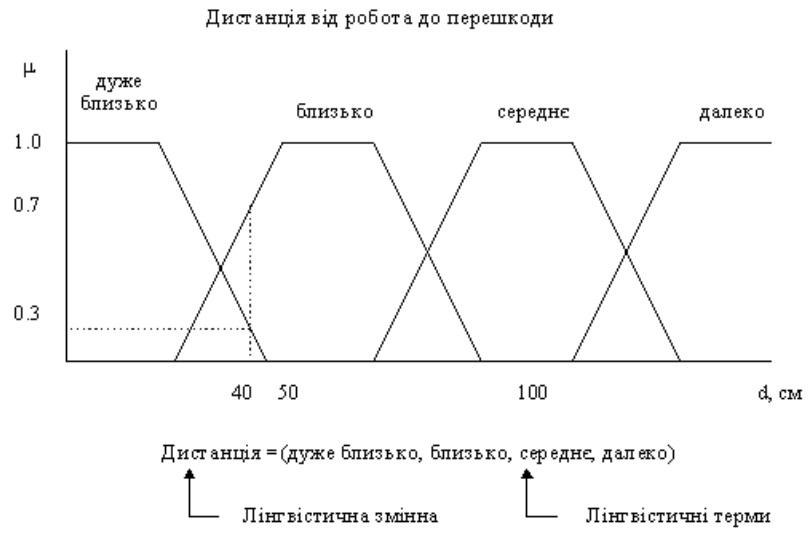

Рис. 2. Лінгвістична змінна і функція приналежності

Змінній НАПРЯМОК, яка може приймати значення в діапазоні від 0 до 360 градусів, задамо терми ВЛІВО, ПРЯМО і ВПРАВО.

Тепер необхідно задати вихідні змінні. У даному прикладі достатньо однієї, яку назвемо КУТ ПОВОРОТУ. Вона може містити терми: РІЗКО ВЛІВО, ВЛІВО, ПРЯМО, ВПРАВО, РІЗКО ВПРАВО. Зв'язок між входом та виходом запам'ятовується в таблиці нечітких правил.

Таблиця нечітких правил

напрямок \ дістанція	дуже близько	близько	середнє	далеко
правий	різко вліво	різко вліво	вліво	прямо
прямий	різко вліво	вліво	вліво	прямо
лівий	різко вправо	різко вправо	вправо	прямо

Кожний запис в даній таблиці відповідає своєму нечіткому правилу, наприклад: Якщо дистанція до перешкоди - «близько» і напрямок «правий», тоді кут повороту «різко вліво».

Таким чином, мобільний робот з нечіткою логікою буде працювати за наступним принципом: дані з сенсорів про відстань до перешкоди та напрямок до неї будуть фазифіковані, оброблені згідно табличних правил, дефазифіковані і отримані дані у вигляді керуючих сигналів надходять на приводи робота.

Переваги нечітких систем

- Можливість оперувати вхідними даними, заданими нечітко: наприклад, дані, які неперервно змінюються в часі (динамічні задачі), значення, що неможливо задати однозначно (результати статистичних опитувань, рекламні компанії);
- Можливість нечіткої формалізації критеріїв оцінки і порівняння: оперування критеріями "більшість", "можливе", переважно" тощо.;
- Можливість проведення якісних оцінок як вхідних даних, так і виведених результатів: значення даних, їх ступень достовірності (не плутати з імовірністю!) та її розподілом;
- Можливість проведення швидкого моделювання складних динамічних систем та їх порівняльний аналіз із заданим ступенем точності: оперуючи принципами поведінки системи, описаними fuzzy-методами:
    - можна швидко з'ясувати точні значення змінних і скласти правила, що їх описують,
    - можна оцінити різні варіанти вихідних значень.

6. Практичне застосування нечіткої логіки

Коли тільки з'явилася теорія нечіткої логіки, в наукових журналах можна було знайти статті, присвячені її можливим областям застосування. У міру просування розробок в даній області число практичних застосувань для нечіткої логіки почало швидко зростати. В даний час цей перелік був би надто довгим, але ось кілька прикладів, які допоможуть зрозуміти, наскільки широко нечітка логіка використовується в системах управління і в експертних системах.

Сьогодні елементи нечіткої логіки можна знайти в багатьох промислових виробах - від систем керування електропоїздами і бойовими вертольотами до побутової техніки. Без застосування нечіткої логіки немислимі сучасні ситуаційні центри керівників західних країн, в яких приймаються ключові політичні рішення і моделюються всілякі кризові ситуації.

Активними споживачами нечіткої логіки є банкіри і фінансисти, а також фахівці в області політичного й економічного аналізу, задачі яких вимагають щоденного прийняття правильних рішень у складних умовах

непередбаченого ринку. Вони використовують нечіткі системи для створення моделей різних економічних, політичних, біржових ситуацій.

Слідом за фінансистами, когнітивними нечіткими схемами зацікавилися промислові гіганти США. Motorola, General Electric, Otis Elevator, Pacific Gas & Electric, Ford і інші на початку 90-х почали інвестувати в розробку виробів, що використовують нечітку логіку. Маючи солідну фінансову "підтримку", фірми, що спеціалізуються на нечіткій логіці, отримали можливість адаптувати свої розробки для широкого кола застосувань.

- Пристрої для автоматичної підтримки швидкості руху автомобіля і збільшення ефективності / стабільності роботи автомобільних двигунів (компанії Nissan, Subaru).
- Системи розпізнавання рукописного тексту в PDA (компанія Sony).
- Поліпшення систем безпеки для атомних реакторів (компанії Hitachi, Bernard, Nuclear Fuel Div.).
- Управління роботами (компанії Toshiba, Fuji Electric, Omron).
- Промислові системи управління (компанії Aptronix, Omron, Meiden, Sha, Micom, Nisshin-Denki, Mitsubishi, Oku-Electronics та ін.).

Поширені помилкові уявлення про нечіткої логіки

Нечітка логіка є неточною: по своїй основі нечітка логіка не більше неточна, ніж стандартна арифметика. Фактично вона набагато більш точна при роботі з неточною інформацією.

В основі нечіткої логіки лежать імовірнісні міркування. Імовірність має справу з шансами виникнення тих чи інших подій, а нечітка логіка - з самими цими подіями. Зазвичай нечітка логіка має справу з двозначністю, а не з невизначеністю.

Нечітка логіка побудована на основі ряду евристичних припущень.

Хоча через інтуїтивної природи нечіткої логіки з першого погляду і може здатися, що лежать в її основі правила вибрані довільно або засновані тільки на здоровому глузді, але насправді було строго доведено, що ці правила є вірними.

7. Операції з нечіткою логікою у пакеті Matlab

Одним з найпотужніших засобів для побудови нечітких моделей є пакет Matlab. Операції з нечіткою логікою у пакеті Matlab дає змогу виконувати модуль Fuzzy Logic Toolbox. Він дозволяє створювати системи нечіткого логічного виведення і нечіткої класифікації в рамках середовища MatLab, з можливістю їхньої інтеграції в засіб Simulink пакета Matlab.

Перша категорія програмних інструментів пакета Fuzzy Logic Toolbox містить функції, які можуть бути викликані безпосередньо шляхом набору імені функції в командному вікні або із власних призначених для користувача додатків. Більшість із цих функцій є функціями **Matla**b у вигляді *m*-файлів. У даному випадку користувач може подивитися запрограмовані в цих функціях алгоритми, а також редагувати і корегувати ці файли.

Друга категорія програмних інструментів пакета **Fuzzy Logic Toolbox** містить діалогові модулі.

Блоки для пакета Simulink містять модулі, які забезпечують інтеграцію систем нечіткого логічного висновку з пакетом Simulink: *Fuzzy Logic Toolbox* - це пакет прикладних програм, що входять до складу середовища MatLab. Він дозволяє створювати системи нечіткого логічного висновку і нечіткої класифікації в рамках середовища MatLab, з можливістю їх інтеграції в Simulink.

Базовим поняттям Fuzzy Logic Toolbox є FIS-структура - система нечіткого висновку (Fuzzy Inference System). FIS-структура містить усі необхідні дані для реалізації функціонального відображення «входи-виходи» на основі нечіткого логічного висновку.

Fuzzy Logic Toolbox містить такі категорії програмних інструментів:

- функції;
- інтерактивні модулі з графічним, призначеним для користувача, інтерфейсом (GUI). Крім того, ці модулі забезпечують зручне середовище для проектування, дослідження і впровадження систем на основі нечіткого логічного висновку. Для запуску інтерактивних модулів досить надрукувати ім’я модуля в командному рядку;
- блоки для пакета Simulink;
- демонстраційні приклади.

Модуль Fuzzy Logic Toolbox містить такі категорії програмних інструментів: функції; інтерактивні модулі з графічним користувальницьким інтерфейсом (GUI).

Блоки для пакета Simulink 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: findcluster

- інструмент для кластеризації; fuzzy

- базовий редактор FIS;

mfedit - редактор функцій приналежності;

ruleedit - редактор та аналізатор правил;

ruleview - демонстратор правил та діаграм нечіткого виведення;

surfview - демонстратор вихідної поверхні.

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

Для зручності імена всіх убудованих функцій приналежності закінчуються на mf.

Виклик функції приналежності здійснюється в такий спосіб:

namemf(*x*, params),

де namemf - найменування функції приналежності;

x - вектор, для координат якого необхідно розрахувати значення функції приналежності;

params -вектор параметрів функції приналежності.

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

Функції FIS:

addmf - додає функцію приналежності до FIS;

addrule - додає правило до FIS;

addvar - додає змінну до FIS;

defuzz - дефазифікує функцію приналежності;

evalfis - здійснює обчислення нечіткого виведення;

evalmf - обчислює функцію приналежності;

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 приналежності для однієї змінної;

readfis - завантажує FIS із диска;

rmmf - видаляє функцію приналежності з FIS;

rmvar - видаляє змінну з FIS;

setfis - установлює властивості нечіткої системи;

showfis - показує анотовану FIS;

showrule - відображує правила FIS; writefis - зберігає FIS на диску.

Функції кластер-аналізу: fcm, genfis1, genfis2, subclust.

Різні функції: convertfis - перетворює нечітку матрицю структури версії 1.0 на матрицю структури версії 2.0;

Discfis - дискретизує FIS;

evalmmf - використовується для обчислення множинних функцій приналежності;

fstrvcat - поєднує матриці різного розміру;

fuzarith - функція нечіткої арифметики;

findrow - шукає рядки матриці, що відповідають вхідному рядку;

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 (‘sum’ - сума, ‘sub’- вирахування, ‘prod’ - добуток, ‘div’ - ділення) до нечітких множин *A* та *B* з універсальної множини *X*. Змінні *A, B* та *X* мають бути векторами однакової розмірності. Нечітка множина *C*, яка повертається, є вектор-стовпцем тієї самої довжини, що й *A* та *B*.

Зауважимо, що ця функція використовує інтервальну арифметику та передбачає, що: *A* та *B* є опуклими нечіткими множинами; функції приналежності для *A* та *B* поза *X* є нулем.

Допоміжні функції графічного користувальницького інтерфейсу:

cmfdlg - додає діалог вибору функцій приналежності;

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 приналежності;

mfdrag - перетягування функції приналежності 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 зони.

FIS-структура (Fuzzy Inference System — система нечіткого виведення) містить усі необхідні дані для реалізації функціонального відображення «входи-виходи» на основі нечіткого логічного виведення.

Поля структури даних системи нечіткого логічного виведення призначені для збереження такої 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 приналежності вихідної змінної (запрограмовані реалізації: ‘max’ - максимум; ‘sum’ - сума і ’probor’ - імовірнісне «АБО»); - найменування вхідної змінної; input.range

• input - масив вхідних змінних: input.name - діапазон зміни вхідної змінної; input.mf - масив функцій приналежності вхідної змінної; input.mf.name - найменування функції приналежності вхідної змінної; input.mf.type - модель функції приналежності вхідної змінної; input.mf.param - масив параметрів функції приналежності вхідної змінної;

• output - масив вихідних змінних: output.name - найменування вихідної змінної; output.range - діапазон зміни вихідної змінної; output.mf - масив функцій приналежності вихідної змінної; output.mf.name - найменування функції приналежності вихідної змінної; output.mf.type - модель функції приналежності вихідної змінної; output.mf.params - масив параметрів функції приналежності вихідної змінної;

• rule - масив правил нечіткої БЗ: rule.antecedent - посилки правила (указуються 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 (указуються 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 треба вказати ім’я відповідного поля. Система нечіткого виведення зберігається на диску у вигляді fis-файла - текстового файла спеціального формату.

Функції readfis та writefis використ- ***Меню File*** - це загальне меню для всіх GUI-модулів, використовуваних із системами нечіткого виведення. За допомогою команди New-FIS користувач має можливість створити нову систему нечіткого виведення.

У разі вибору цієї команди з’являться дві альтернативи: **Mamdani** та **Sugeno,** що визначають тип створюваної системи.

За допомогою команди Import користувач має можливість завантажити раніше створену систему нечіткого логічного виведення.

Команда Export дозволяє зберегти систему нечіткого виведення в робочу зону або на диск.

Команда Print дозволяє вивести на принтер копію графічного вікна.

Команда Close закриває графічне вікно.

Меню Edit.

Команда Undo скасовує раніше зроблену дію.

Команда Add Variable дозволяє додати в систему ще одну змінну: вхідну (input) або вихідну (output).

Команда Remove Selected Variable видаляє поточну змінну із системи.

Команда Membership Function відкриває редактор функцій приналежностей.

Команда Rules відкриває редактор БЗ.

Меню View - це загальне меню для всіх GUI-модулів, використовуваних із системами нечіткого виведення. Це меню дозволяє відкрити вікно візуалізації нечіткого виведення (команда Rules) та вікно виведення поверхні «вхід-вихід», що відповідає системі нечіткого виведення (команда Surface).

Меню ***And Method, Or Method, Implication*** **та** ***Aggregation*** дозволяють установити реалізації логічних операцій «ТА», «АБО», імплікація, об'єднання, відповідно.

Меню ***Defuzzification*** дозволяє вибрати метод дефазифікації.

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

Панель ***Current Variable*** дозволяє для поточної змінної, ім'я котрої задається у полі Name, задати у полі Type тип (вхідна - товуються для завантажена у робочу зону та збереження на диску таких файлів.

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

FIS-редактор містить три загальносистемних меню: File, Edit, View та п'ять меню для вибору параметрів нечіткого виведення: And Method, Or Method, Implication, Aggregation та Defuzzification - діапазон значень.

FIS-редактор також має інформаційну панель, що відображає ім'я поточної FIS (поле FIS-Name) та її тип (поле FIS-Type). Крім того FIS-редактор має панель із графічним зображенням системи виведення, вхідних та вихідних змінних.

Редактор функцій приналежності (Membership Function Editor) призначений для задавання такої інформації про термимножини вхідних і вихідних змінних: кількість термів; найменування термів; тип і параметри функцій приналежності, котрі необхідні для подання лінгвістичних термів у вигляді нечітких множин. Редактор функцій приналежності містить меню File, Edit, View та панелі введення інформації.

Меню Edit. Команда Add MFs дозволяє додати терми у терммножину, використовувану для лінгвістичної оцінки поточної змінної. При виборі цієї команди з'явиться діалогове вікно, в якому необхідно вибрати тип функції приналежності та кількість термів. Значення параметрів функцій приналежності будуть установлені автоматично таким чином, щоб рівномірно покрити область визначення змінної, заданої у полі Range. При

зміні області визначення у полі Range параметри функцій приналежності будуть промаштабовані.

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

Команда Remove Selected MF видаляє поточний терм із терммножини поточної змінної. Ознакою поточної змінної є червона окантовка її прямокутника. Ознакою поточного терму є червоний колір його функції приналежності. Для вибору поточного терму необхідно позиціонувати курсор миші на графіку функції приналежності і клацнути лівою кнопкою миші.

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

Панель ***Current Variable*** дозволяє для поточної змінної, ім'я котрої вказане у полі Name, а тип у полі Type, задати у полі Range діапазон значень, а у полі Display Range - діапазон відображення.

Панель ***Current Membership Function*** дозволяє для поточної функції приналежності, ім'я терму котрої вказують у полі Name, задати тип (назву Matlab-функції) у полі Type та параметри у полі Params. Редактор також має панель із зображенням графіка поточної функції приналежності (вхідна - input, вихідна - output), а також у полі Range

Редактор БЗ (Rule Editor) призначений для формування і модифікації нечітких правил. Редактор функцій приналежності 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 та перегляду правил.

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

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

Для видалення правила з БЗ потрібно клацнути один раз лівою кнопкою миші на цьому правилі та натиснути кнопку Delete Rule.

Для модифікації правила необхідно клацнути один раз лівою кнопкою миші на цьому правилі, потім установити необхідні параметри правила і натиснути кнопку Edit Rule.

Меню Options дозволяє установити мову і формат правил БЗ. 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 БЗ: Verbose — лінгвістичний; Symbolic — логічний; Indexed — індексований.

Модуль візуалізації нечіткого виведення (Rule Viewer) дає змогу проілюструвати хід логічного виведення за кожним правилом, одержати результативну нечітку множину й виконати процедуру дефазифікації.

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), догори - униз (up-down).

Кожне правило БЗ подається у вигляді послідовності горизонтально розташованих прямокутників. Порожній прямокутник у візуалізації правила означає, що в цьому правилі посилки за змінною немає. Жовте заливання графіків функцій приналежностей вхідних змінних указує, наскільки значення входів відповідають термам даного правила. Для виведення правила у форматі Rule Editor необхідно клацнути один раз лівою кнопкою миші по номеру відповідного правила. У цьому випадку зазначене правило буде виведено в нижній частині графічного вікна. Блакитне заливання графіка функції приналежності вихідної змінної являє собою результат логічного виведення у вигляді нечіткої множини за даним правилом. Результативну нечітку множину, що відповідає логічному виведенню за всіма правилами, показано в нижньому прямокутнику останнього стовпця графічного вікна. У цьому самому прямокутнику червона вертикальна лінія відповідає чіткому значенню логічного виведення, отриманого в результаті дефазифікації.

Уведення значень вхідних змінних можна здійснювати двома способами: шляхом уведення чисельних значень у поле Input; за допомогою миші, шляхом переміщення ліній-покажчиків червоного кольору.

У полі Plot points задасться кількість крапок дискретизації для побудови графіків функцій приналежності. Значення за замовчуванням — 101.