ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«МЕТІНВЕСТ ПОЛІТЕХНІКА»

ІНТЕЛЕКТУАЛЬНІ СИСТЕМИ УПРАВЛІННЯ

курс лекцій з навчальної дисципліни

«Інтелектуальні системи управління»

(за освітньо-професійною програмою другого (магістерського) рівня освіти «Інтелектуальні системи управління та робототехнічні комплекси в гірничо-металургійному виробництві» спеціальності 174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка»)

Рекомендовано Науково-методичною радою
ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«МЕТІНВЕСТ ПОЛІТЕХНІКА»
(протокол № 6 від «24» травня 2024 р.)
Обов'язково до розміщення в репозиторії

Запоріжжя 2024

Інтелектуальні системи управління: курс лекцій з дисципліни «Інтелектуальні системи управління» (за освітньо-професійною програмою другого (магістерського) рівня освіти «Інтелектуальні системи управління та робототехнічні комплекси в гірничо-металургійному виробництві» спеціальності 174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка») / Уклад. О.В. Разживін. Запоріжжя: ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА», 2024. 345 с.

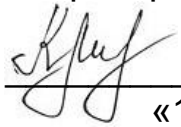
Курс лекцій присвячено кіберфізичним системам та синтезу нечітких систем керування технологічними обєктвми. Кіберфізичні системи розглядаються як складні розподілені системі, керовані або контрольовані комп'ютерними алгоритмами, з їх тісною інтеграцією з Інтернет та його користувачами. Розглянута технологія інтернет речей (Internet of Things, або IoT), які працюють у різних просторових і часових масштабах, з виявленням безлічі різних поведінкових модальностей та взаємодіють один з одним безліччю способів, які змінюються залежно від контексту..

Рекомендовано для здобувачів вищої освіти спеціальності за освітньо-професійною програмою другого (магістерського) рівня освіти «Інтелектуальні системи управління та робототехнічні комплекси в гірничо-металургійному виробництві» спеціальності 174 «Автоматизація, комп'ютерно-інтегровані технології та робототехніка» та усіх форм навчання другого (магістерського) рівня вищої освіти)..

Самостійне електронне текстове мережеве видання

Затверджено на засіданні кафедри
автоматизації, електро- та робототехнічних систем
Протокол № 8 від «30» квітня 2024 р.

Узгоджено:
Секретар Редакційної ради


_____ Малій Х. В.
«17» травня 2024 р.

ЗМІСТ

1 ВВЕДЕННЯ В КІБЕРФІЗИЧНІ СИСТЕМИ (КФС)

1.1 Поняття КФС. Основні принципи організації КФС

Четверта індустріальна революція (Індустрія 4.0) **-** перехід на повністю автоматизоване цифрове виробництво, кероване інтелектуальними системами в режимі реального часу в постійній взаємодії із зовнішнім середовищем, що виходить за межі одного підприємства, з перспективою об'єднання у глобальну промислову мережу Речів та послуг.

У вузькому значенні Індустрія 4.0 (Industrie 4.0) – це назва одного з 10 проектів державної Hi-Tech стратегії Німеччини до 2025 року, що описує концепцію розумного виробництва (Smart Manufacturing) на базі глобальної промислової мережі Інтернет речей та послуг (Internet of Things and Services) .

У широкому сенсі, Індустрія 4.0 характеризує поточний тренд розвитку автоматизації та обміну даними, який включає *кіберфізичні системи, Інтернет Речів і хмарні обчислення*. Являє собою новий рівень організації виробництва та управління ланцюжком створення вартості протягом усього життєвого циклу продукції, що випускається.

Кіберфізичні системи – поняття досить комплексне. однозначного і загальноприйнятого визначення нині вони отримали, оскільки ці системи перебувають у перетині відразу кількох сфер і, залежно від реалізації. їх головною загальною характеристикою є дуже щільна взаємодія між обчислювальними процесами і фізичними процесами, тому можна сказати, що *кіберфізична система* (CPS) - це комплексна система з обчислювальних і фізичних елементів, яка постійно отримує дані з навколишнього середовища і використовує їх для подальшої оптимізації процесів управління.

Основним принципом роботи кіберфізичних систем можна назвати глибокий взаємозв'язок між їх фізичними та обчислювальними елементами. «мозок» системи у вигляді штучного інтелекту та інших технологій отримує дані від сенсорів у реальному світі, аналізує ці дані та використовує їх для подальшого управління фізичними елементами. Завдяки такій взаємодії кіберфізична система здатна ефективно працювати в умовах, що змінюються, як аналог людського організму або сучасна компанія, яка аналізує ситуацію на ринку, щоб розробити саме той продукт, який йому зараз потрібен. причому цикл «управління – отримання даних – обробка даних – управління» при налагодженій роботі системи щоразу має давати позитивні результати та створювати нову цінність.

Технологія киберфізичних систем

Кіберфізична система - являє собою складну розподілену систему, керовану або контрольовану комп'ютерними алгоритмами, тісно інтегровану з Інтернет та його користувачами. Її технологічною основою

стала технологія інтернет речей (Internet of Things, або IoT). У системах CPS фізичні та програмні компоненти тісно взаємопов'язані. Кожна компонента працює у різних просторових і часових масштабах, виявляє безліч різних поведінкових модальностей і взаємодіє один з одним безліччю способів, які змінюються залежно від контексту. Зростання складності завдань управління зумовлює застосування принципово нових методів та систем управління.

Кіберфізичні системи здійснюють обчислювальні процедури у своїй розподіленій структурі, вони включають «розумні вузли» і уможливлюють реконфігурувати потоки в мережі залежно від умов. Таким чином, кіберфізичні системи є розподіленими системами з можливістю інтелектуальної обробки та реконфігурації потоків за рахунок інтелектуального управління. CPS застосовує трансдисциплінарні підходи, поєднуючи теорію кібернетики, мехатроніки, проектування та науки про процеси. Управління процесом часто називають вбудованими системами (embedded systems). У цих системах акцент найчастіше робиться на обчислювальних елементах і менше на інтенсивному зв'язку між обчислювальними та фізичними елементами. CPS схожа на технологію «Інтернет речей», що використовують одну й ту саму базову архітектуру. Однак CPS представляє більш високу комбінацію та координацію між фізичними та обчислювальними елементами. З системних позицій CPS є розподіленою субсидіарною системою. Ця система має багато рівнів, але на кожному рівні масштабу та складності з'являються нові властивості, які не зводяться до властивостей простих рівнів. На відміну від традиційних вбудованих систем, повнофункціональні CPS розробляються як мережа взаємодіючих елементів з фізичним введенням і виведенням, а не як автономні пристрої. Поняття CPS тісно пов'язане з концепціями робототехніки та сенсорних мереж із інтелектуальними механізмами власне обчислювального інтелекту, що ведуть шлях. Постійний прогрес у науці та техніці дозволяє створювати зв'язок між обчислювальними та фізичними елементами за допомогою інтелектуальних механізмів, значно збільшуючи адаптивність, автономність, ефективність, функціональність, надійність, безпеку та зручність використання кіберфізичних систем. Деякі приклади практичного застосування кіберфізичних систем

В виробничому середовищі

Кіберфізичні системи можуть покращити виробничі процеси, забезпечуючи обмін інформацією реального часу між промисловим обладнанням, виробничим ланцюжком поставок, постачальниками, системами управління бізнесом та клієнтами. Крім того, кіберфізичні системи можуть підвищувати ефективність цих процесів завдяки автоматичному моніторингу та контролю всього виробничого процесу та адаптації виробництва для задоволення переваг клієнтів. Кіберфізичні системи підвищують прозорість та керованість ланцюжків поставок,

покращуючи відстежуваність та безпеку товарів.

В охороні здоров'я

Кіберфізичні системи використовуються для дистанційного моніторингу фізичних показників пацієнтів у реальному часі з метою зменшення потреб у госпіталізації (наприклад, пацієнтів з хворобою Альцгеймера) або для покращення догляду за інвалідами та літніми людьми. Крім того, кіберфізичні системи застосовуються в нейробіологічних дослідженнях для вивчення функцій організму людини з використанням інтерфейсів між мозком та обладнанням та терапевтичною робототехнікою.

У відновлюваній енергетиці

Інтелектуальні енергомережі являють собою кіберфізичні системи, в яких датчики та інші пристрої забезпечують моніторинг мережі з метою контролю, підвищення надійності та енергоефективності. В інтелектуальних будівлях: спільна робота інтелектуальних пристроїв та кіберфізичних систем дозволяє скоротити енергоспоживання, підвищити безпеку та захищеність, а також створити комфортніші умови для мешканців. Наприклад, кіберфізичні системи можуть підтримувати моніторинг енергоспоживання та використання систем регулювання для реалізації концепції будинку з нульовим споживанням електроенергії. Крім того, їх можна використовувати для визначення ступеня шкоди для будівель у результаті непередбачених подій та запобігання руйнуванню конструкцій.

На транспорті

Транспортні засоби та інфраструктура можуть взаємодіяти між собою, обмінюючись в реальному часі інформацією про дорожній рух, місцезнаходження та проблеми, запобігаючи транспортним інцидентам і дорожнім пробкам, підвищуючи безпеку і зрештою заощаджуючи час і гроші.

В сільському господарстві

Кіберфізичні системи можуть використовуватися для створення більш сучасного та ефективного сільського господарства. Вони можуть збирати важливу інформацію про клімат, ґрунт та інші дані для більш точного управління сільськогосподарськими роботами. Датчики кіберфізичних систем можуть вести постійний моніторинг різних показників, таких як зрошення ґрунту, вологість повітря та здоров'я рослин для підтримки оптимальних навколишніх умов.

В обчислювальних середовищах

Кіберфізичні системи дозволяють краще розуміти поведінку систем та користувачів для підвищення продуктивності та більш ефективного управління ресурсами. Наприклад, можна оптимізувати роботу програм з урахуванням контексту та дій користувачів або відстежувати доступність ресурсів. Крім того, популярні соціальні мережі та сайти електронної комерції зберігають інформацію про дії користувачів та затребуваний контент, аналізують цю інформацію, щоб

передбачати, що може бути цікаво користувачам, та пропонувати рекомендації щодо друзів, публікацій, посилань, сторінок, подій чи продуктів.

Концепції Індустрії 4.0, що характеризується впровадженням «кіберфізичних систем» у заводські процеси. Четверта промислова революція пов'язана не тільки з розумними та взаємопов'язаними машинами та системами. Її спектр дії значно ширший. Одночасно виникають хвилі подальших проривів у різних областях: від розшифровки інформації, записаної в людських генах до нанотехнологій, від відновлюваних енергоресурсів до квантових обчислень. Саме синтез цих технологій та їх взаємодія у фізичних, цифрових та біологічних доменах становлять фундаментальну відмінність четвертої промислової революції від усіх попередніх революцій. Класичне визначення кіберфізичних виробничих систем – людська праця, «розумні» машини та транспорт, інтегровані в єдиному цифровому просторі за допомогою мереж, «розумних» пристроїв, сенсорних систем, аналітичних платформ та хмарних обчислень. Ключовими відмінностями кіберфізичних виробничих систем від традиційних виробничих систем є децентралізація, висока стійкість, абсолютна гнучкість та здатність до безперервної та нескінченної самооптимізації. Обов'язковою ознакою кіберфізичних виробничих систем є наявність у їх складі автономних «розумних» пристроїв, машин та розумного транспорту, розподіленої системи інтелектуальних сенсорів, з'єднаних між собою з платформами хмарних обчислень та аналітики. У новій виробничій реальності кіберфізичні системи будуть допомагати людині справлятися з дедалі більшою складністю виробничих завдань, що стоять перед нею.

Рисунок 1.1 – Технологічний прогрес на найближчі п’ять років

Основні принципи концепції Індустрії 4.0: функціональної сумісності людини та машини можливості контактувати безпосередньо через інтернет; прозорості інформації та здатності систем створювати віртуальну копію фізичного світу.

1.2 Нові технології автоматизації. цикли розвитку інформаційних систем

Сучасні IT інновації, їх поява та тренди розвитку знаходяться під безперервною увагою аналітиків. Групові роботи аналітиків дозволяють IT компаніям найбільш відповідально вкладати кошти у розвиток та використання сучасної автоматизації.

Згідно з базовими принципами аналітики перспектив розвитку автоматизації (гіперавтоматизації) виділяються такі посилки:

- майбутнє можна створити залежно від докладених зусиль;
- майбутнє можна змінювати залежно від рішень заінтересованих людей;
- майбутнє неможливо передбачити достовірно, але можна підготуватися.

Розрізняють два підходи такої аналітики: аналітичні звіти та форсайт (погляд у майбутнє). Обидві технології спрямовані на оцінку довгострокових перспектив науки, технологій, економіки та суспільства та визначення стратегічних напрямів досліджень та нових технологій, здатних принести найбільші соціально-економічні блага людству.

У зарубіжній практиці бізнес технологій цифровізації використовує рекомендації провідних аналітичних компаній, зокрема: Gartner, International Data Corporation, Orange Business Services.

Gartner (Гартнер) – дослідницька та консалтингова компанія, що спеціалізується на ринках інформаційних технологій. Дослідженням Gartner регулярно присвячуються статті у таких виданнях, як Financial Times, The Wall Street Journal, The New York Times, Der Spiege, The Register, ZDNet. Компанія вважається ключовим дослідником ринків технологій цифровізації.

Для того, щоб спертися на найбільшу репрезантивність аналітичних звітів, звернемося до історичного розвитку консалтингових компаній. Починаючи з першої половини 1990-х років, Gartner поглинула більше 30 компаній, в основному - конкурентів, що діють на ринку досліджень, причому як працювали в окремих регіонах, так і глобальних. Серед найбільших поглинань:

- AMR Research (2009) – аналітик ринків корпоративних інформаційних технологій, що спеціалізується в основному на ERP-, SCM- та PLM-системах (сума угоди склала $64 млн);
- Burton Group (2009) – глобальний аналітик IT-ринків, що спеціалізується на практичному досвіді впроваджень;
- IDEAS International (2012) - аналітична IT-компанія, що

спеціалізувалась на технологічних оцінках та порівняннях конкурентних серверних технологій та технологій зберігання даних;

- Software Advice (2014) - компанія, що надає дослідження та відгуки користувачів на програмні продукти в таких галузях як автоматизація маркетингу та Business Intelligence середнього та малого рівня бізнесу;
- Capterra (2015) – власник інтернет сервісу з інструментами для пошуку, підбору та порівняння програмного забезпечення, аналітичними статтями та оглядами за відомими постачальниками додатків для бізнесу.

Іншою міжнародною аналітичною компанією є International Data Corporation (IDC). IDC – провідний постачальник інформації та консультаційних послуг, організатор заходів на ринках інформаційних технологій, телекомунікацій та споживчої техніки. IDC допомагає професіоналам IT, керівникам бізнесу та інвесторам приймати обґрунтовані рішення про закупівлю та їх вибір. IDC є підрозділом видавничої компанії International Data Group. За власними даними на IDC працюють понад 1100 аналітиків у 110 країнах світу, які збирають та обробляють інформацію про місцеві ринки IT.

Експерти цієї компанії вважають, що у найближчі 5 років найвищі темпи зростання очікуються від ринку консалтингу з безпеки. Зростання складності загроз, цифрова трансформація бізнес сфери і динамічна нормативно-правова база, що часто оновлюється, сприяють швидкому розвитку сегментів ринку керованих послуг безпеки: керованих пристроїв UTM (Unified Threat Management) і управлінню безпекою та вразливістю. Попит на послуги корпоративної кібербезпеки в Росії найвищий у фінансах, промисловості та енергетиці.

Ще однією компанією аналітиків перспектив розвитку технологій цифровізації є Orange Business Services. Orange Business Services – підрозділ Orange Group, який працює на ринку B2B. У компанії працює 21 000 співробітників, які підтримують процеси цифрової трансформації міжнародних корпорацій на п'яти континентах. Orange Business Services не тільки телеком-оператор, але і інтегратор IT рішень і постачальник сервісів з доданою вартістю. Інтегровані технології Orange Business Services охоплюють цілий спектр рішень – від програмно-визначуваних мереж (SDN/NFV), Big Data та IoT до хмарних обчислень, уніфікованих комунікацій, засобів спільної роботи та кіберзахисту. Серед клієнтів Orange Business Services понад 3 000 відомих міжнародних корпорацій, що працюють на глобальному рівні, а також понад 2 мільйони професіоналів та компаній.

Форсайт – формат комунікації та соціальна технологія, що дозволяє учасникам визначити бажане майбутнє та домовитися про подальші дії.

У списку стратегічних напрямів в автоматизації виділяються три напрями розвитку технологій, пов'язаних із гіперавтоматизацією:

робототехніка, мобільні технології, комп'ютерна графіка та гейміфікація.

Робототехніка. У 2018 році обсяг світового ринку побутових роботів досяг 3,02 млрд дол.

Експерти прогнозують, що попит на побутові роботи збільшується завдяки їхній практичності та зручності, які досягаються за рахунок таких функцій, як візуалізація приміщень та визначення зон, у які не повинні потрапляти роботи. Інтеграція з «розумними» голосовими помічниками від Amazon та Google розширює функціональність такої техніки. У 2017 році у світі продано понад 381 тис. промислових роботів на $16,7 млрд.

Мобільні, вбудовані та переносні пристрої, а також програмне забезпечення для них. До цього напряму відносять проекти, пов'язані зі створенням мобільних додатків і пристроїв, що носяться (wearables), проекти сфери мобільної охорони здоров'я тощо.

Нові розробки в комп'ютерній графіці та гейміфікації. Комп'ютерна графіка дуже потрібна для створення програмного забезпечення для сучасного бізнесу. Сучасні комп'ютерні ігри вимагають серйозної наукової бази – йдеться і про можливості побудови реалістичного 3D-зображення, і про серйозну графіку, і навіть про нові матеріали та пристрої

Провідними компаніями аналітики перспектив розвитку цифрових технологій виділяються 5 основних трендів (див. рис 1.2).

Тренд перший – демократизація штучного інтелекту. Як пишуть аналітики Gartner, вже найближчими роками технології штучного інтелекту (AI) будуть усюди. Деякі з них (наприклад, технології глибокого навчання та віртуальних помічників) стануть загальнодоступними у наступні 2-5 років. У всіх звітах консалтингових компаній дійсно багато технологій, пов'язаних із штучним інтелектом – автономні автомобілі, автономні літаючі транспортні засоби, віртуальні помічники, автономні роботи, розумні платформи для спілкування, глибоке навчання тощо

Тренд другий – цифрові екосистеми. У компанії Gartner впевнені, що перехід від роз'єднаних технічних інфраструктур до екосистемних платформ відкриють нові бізнес моделі, які допоможуть сформувати міст між людьми і технологіями. Що таке "цифрові екосистеми"? Аналітики Gartner під цим терміном розуміють різні технології: блокчейн та платформи для інтернету речей (IoT Platform), кіберфізичні системи. Також до цієї категорії потрапили блокчейн для безпеки даних, цифрові двійники, графи знань (Knowledge Graphs). Блокчейн та платформи для інтернету речей вже пройшли пік галасу, і досягнуть зрілості протягом наступних найближчих років.

Тренд третій – diy-біохакінг. У наступні десять років, як пишуть у Gartner, люди навчяться зламувати біологічні процеси та адаптувати їх під свій стиль життя та інтереси. Проте аналітики зазначають, що залишаються питання, наскільки суспільство готове прийняти такі зміни та які етичні проблеми виникають. До цього розділу аналітики віднесли

– біочіпи, штучні та вирощені тканини та органи, інтерфейс мозок-комп'ютер, доповнена реальність (Augmented Reality), змішана реальність (Mixed Reality), розумний одяг та матеріали. Цікаво, що доповнена реальність (Augmented Reality), як і розумна одяг, далі за інших просунувся у впровадженні. Проте аналітики, як і раніше, вважають, що до масовості використання пройде ще 5-10 років.

Рисунок 1.2 – Основні тренди перспектив розвитку цифрових технологій

Тренд четвертий – людиноцентричні технології. Під цим трендом у Gartner мають на увазі те, що технології продовжать ставати все більш людиноцентричними. Такі технології розширять можливості просторів, де буває людина, і дозволять розумніше жити і працювати. До подібних технологій в нинішньому звіті віднесені 4D-принтінг, системи, що самовідновлюються, розумний пил, розумне робоче місце, батарейки з кремнієвим анодом (ємність яких набагато більше звичайних), стереодисплей.

Тренд п'ятий – повсюдна інфраструктура. Сенс цього тренду у цьому, що інфраструктура більше перестав бути стримуючим чинником у розвиток компаній. Масова популярність хмарних обчислень та численних варіацій цієї технології дають компаніям доступні та майже безмежні обчислювальні потужності. Цей тренд є такими технологіями, як 5G, карбонові нанотрубки, нейроморфні мікросхеми, квантові обчислення.

Згідно з прогнозом подальшого розвитку, у період з 2018 по 2022 р. включно інвестиції в обладнання, програмне забезпечення, послуги та зв'язок, залучені до створення рішень інтернету речей, зростатимуть у середньому на 18% щорічно.

Hype cycle IT компанії Gartner Hype cycle – графічне відображення проникнення, адаптації та соціального впливу нових технологій цифровізації (див. рис. 1.3).

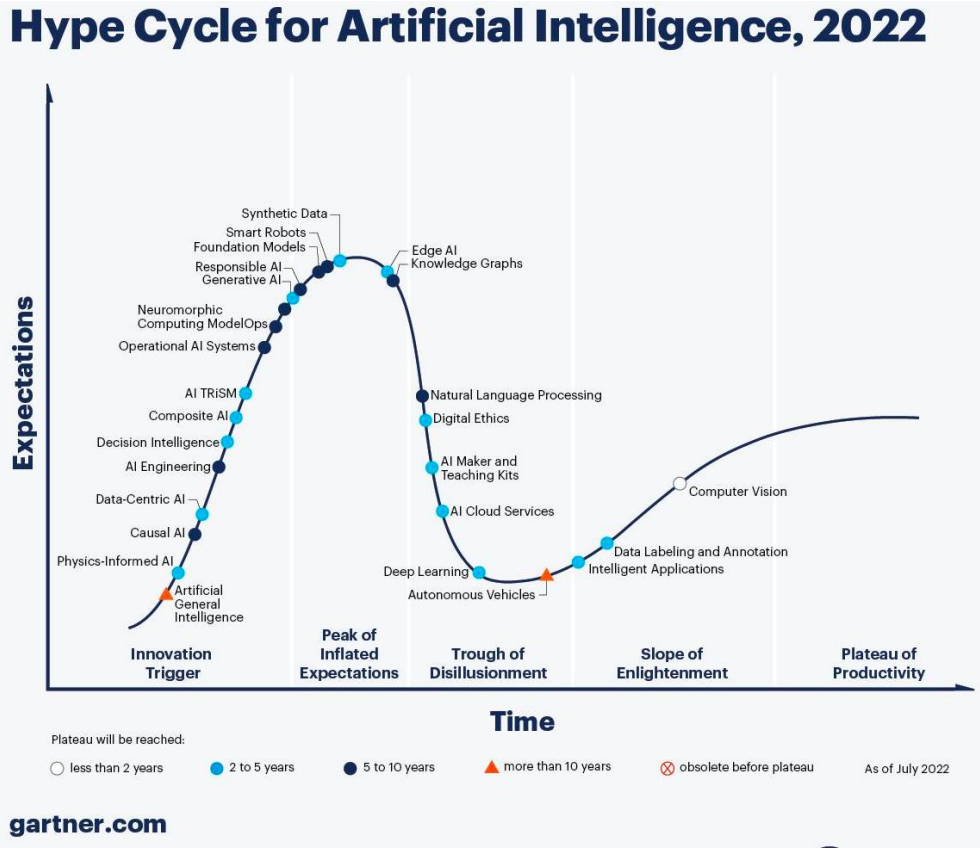


Рисунок 1.3 - Прогнози Gartner за роками (Нові технології)

С 1995 Gartner використовує цю методику для опису та оцінки ентузіазму, який викликає у користувачів появу нових технологічних рішень. Хайп-цикл розвитку сучасних IT компанії Гартнер графічно представляється S-подібною кривою із виділенням п'яти ділянок. На думку експертів, будь-яка сучасна інноваційна технологія проходить через дільниці хайп-циклу. Дослідники Gartner вважають, що реалізовані ними дослідження та супутня інфографіка допомагають професіоналам відокремити «мрії» ЗМІ від реальності. Зокрема, таким чином, IT-директори та генеральні директори компаній можуть приймати більш точні рішення про використання чи не використання новинок.

У кривої Hype cycle перехідний процес коливального типу. Виділяють 5 ділянок, які проходить будь-яка інноваційна технологія.

«Запуск технологій» характеризується появою технологій та обговоренням перспектив розвитку спочатку розробниками, пізніше з рекламою в ентузіастів.

"Завищені очікування" характеризується застосуванням окремих компаній даної технології на своїх потужностях.

«Ділянка розчарування» характеризується виявленням недоробок та недоліків інновації, виникненням розчарування.

«Ділянка освіти» характеризується виправленням та доповненням розробниками інноваційного продукту, що стає якіснішим, а також появою інтересу меншого, ніж у пік очікувань.

«Ділянка продуктивності» характеризується завоюванням інновації ринку.

За прогнозами аналітиків, до 2025 року потенційний економічний ефект від впровадження технологій кіберфізичних систем складе близько $14-33 трлн. Цей аналіз заснований на поглибленому аналізі ключових потенційних переваг, у тому числі якісніших продуктів і нижчих цін. До звіту увійшли такі технологічні напрями:

- мобільний інтернет;
- автоматизація;
- Інтернет речей;
- хмарні обчислення;
- удосконалена робототехніка;
- автономні транспортні засоби;
- геноміка наступного покоління;
- нові засоби накопичення енергії;
- ЕБ-друк;
- покращені матеріали та паливо;
- поновлювані джерела енергії.

2 АВТОМАТИЗОВАНІ СИСТЕМИ УПРАВЛІННЯ ГІРНИЧО-МЕТАЛУРГІЙНОГО КОМПЛЕКСУ. КІБЕРФІЗИЧНІ СИСТЕМИ

2.1 CALS, ERP, MES і АСУ ТП

Сучасні реалії такі, що підприємства різних галузей незалежно від виду діяльності дедалі частіше стикаються з необхідністю автоматизації діяльності загалом. Неможливо вручну обробляти величезні масиви неструктурованих даних, необхідні аналітичного аналізу та прийняття рішень. Для вирішення цієї проблеми були створені системи автоматизації: CALS, ERP, MES і АСУ ТП.

Автоматизована система (АС) – це програмно-технічний комплекс, який виконує як функції контролю за параметрами та характеристиками процесу, результатом чого є сигналізація оператору про вихід контрольованих параметрів (ключових показників ефективності) за межі встановленого контрольного інтервалу (інформаційна система управління), так і функції локального управління, надання безпосереднього на контрольовану частину процесу, у тому, що не порушувалися встановлені контрольні кордону.

В даний час поширеним рішенням АС є:

Автоматизована система управління технологічним процесом (АСУ ТП) – система контролю, управління та захисту технологічного процесу, побудована на засобах вимірювання, обчислювальної техніки, виконавчих пристроїв та механізмів і призначена для забезпечення комплексної автоматизації технологічних операцій на виробництві.

Розподілена система управління (РСУ) – сукупність територіально та функціонально розподілених автоматизованих та автоматичних підсистем з єдиним інформаційним простором, у якій кожна підсистема може використовувати параметри та результати обчислень інших підсистем. У зарубіжних джерелах розподілена система управління носить назву DCS (англ. Distributed Control System) – це комплекс технічних та програмних рішень для побудови АСУ ТП, характерною рисою якої є децентралізована обробка вимірювань КВП та наявність розподілених систем введення та виведення інформації, підвищена відмовостійкість, стандартна система диспетчерського керування.

Автоматизована система управління виробничими процесами цеху (MES, manufacturing execution system) – це система менеджменту виробничої діяльності, яка ініціює, спрямовує, реагує та повідомляє людям, керуючим виробництвом, про роботу виробництва в режимі «on-line» та реальний масштаб часу. Ця система допомагає діяльності у сфері виконання промислових замовлень. Це система автоматизованого управління цеховим рівнем підприємства, призначена для виробництва необхідних виробів або надання інфраструктурних послуг, що включає контроль якості, управління

документообігом, внутрішньозаводське диспетчерське управління, відстеження незавершеного виробничого процесу, контроль за дотриманням операційної технологічної карти, протоколювання виробничого процесу, управління ресурсами та виправленням бракованих виробів, контрольно-вимірювальні процедури та збір даних.

Автоматизована система керування підприємством (ERP/MRP II). Це системи автоматизованого планування ресурсів підприємства (ERP) та планування виробничих ресурсів (MRP II), системи, які забезпечують фінансування, управління замовленнями, управління кадрами, планування продукції та матеріалів та надають відповідні функції. Сучасні системи цього призначення зосереджені на виконанні комплексного планування, ділових процесів та їх виконанні по всьому підприємству.

Автоматизована система керування життєвим циклом продукції CALS. Перша частина абревіатури CALS – Continuous Acqusition [Support] (безперервний збір даних) означає безперервність інформаційної взаємодії під час формалізації потреб клієнта, формування замовлення, процесу проектування та виготовлення.

Друга частина – Life Cycle Support (підтримка життєвого циклу виробу) – означає системність підходу до інформаційної підтримки всіх процесів життєвого циклу виробу, насамперед процесів експлуатації, обслуговування, ремонту та утилізації тощо.

Автоматизована система управління технологічним процесом (SCADA, Supervisory Control And Data Acquisition), диспетчерське управління та збір даних – це програмний пакет або програмно керований комплекс, призначений для розробки та забезпечення роботи в реальному часі систем збирання, обробки, відображення та архівування інформації про об'єкт моніторингу або управління.

Крім того, для управління технологічними процесами та діяльністю підприємств використовуються окремі системи автоматизації: управління взаємовідносинами з клієнтами (CRM); управління бізнес-процесами (BPM); технічне обслуговування та ремонт (TOiP); управління виробничими активами (EAM); розширене планування та планування (APS); керування життєвим циклом продукції (PLM); система керування лабораторною інформацією (LIMS).

Спільне застосування подібних систем дозволяє вибудувати гнучку вертикаль управління діяльністю підприємства, починаючи від автоматичного збирання інформації та закінчуючи отриманням зведених аналітичних звітів.

Використання цих систем забезпечує управління інформацією у масштабах всього підприємства на стратегічному, тактичному та оперативному рівнях.

Відповідно до назв рівнів (див. рис. 2.1), встановлюється область відповідальності рівня управління, що відрізняється в першу чергу частотою обробки даних та ієрархічною значимістю.

Оперативний рівень відрізняється високою частотою обробки ділової інформації, а стратегічний рівень – низькою. Це ускладнює узгодження часу спільної обробки даних. Однак, менеджмент воліє виконувати обробку інформації на всіх рівнях у реальному масштабі часу.

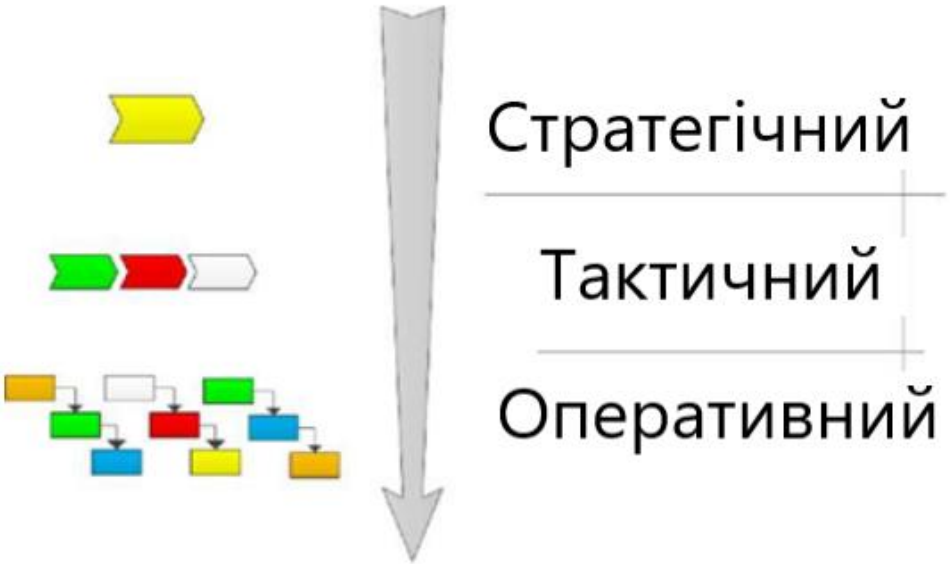


Рисунок 2.1 – Рівні управління

Користувачами цих систем є:

- директор, управління підприємством, ERP;
- начальник виробництва, управління виробництвом, MES;
- оператор, керування технологічним процесом, SCADA, DCS;
- бізнес-аналітик, моніторинг ключових показників ефективності (KPI) BI;
- менеджер, управління бізнес-процесом, BPM;
- механік, управління технічним обслуговуванням, TOiP, EAM;
- менеджер, управління виробничим плануванням, APS;
- конструктор, керування життєвим циклом виробу, PLM;
- технолог, лабораторний контроль якості, LIMS.

Завданнями учасників виробничих процесів є безперервне керування процесами виробництва:

- оператор/робочий реєструє виконання технологічних операцій, вказує причини простоїв обладнання, робить запити на переналагодження тощо, система дозволяє йому бачити своє вироблення;
- наладчик обробляє запити на переналагодження обладнання та реєструє факт його здійснення;
- майстер/начальник дільниці контролює виконання операцій на ділянці та підтверджує виконання завдань;
- контролер служби якості підтверджує проведення контрольних

операцій;

- диспетчер веде оперативне планування у ручному чи автоматичному режимі залежно від специфіки роботи ділянки;

- працівник інструментальної комори обробляє заявки від наладчиків та операторів на отримання та повернення ріжучого інструменту та оснащення.

Основні функції SCADA, MES, ERP:

1. Введення-виведення інформації: формування графічного інтерфейсу користувача;

- обмін даними з іншими інформаційними системами.

2. Обробка даних (бізнес-логіка).

3. Зберігання даних (база даних).

У системі ERP забезпечується управління фінансово-господарською діяльністю підприємства, планування та облік у виробництві на міжцеховому рівні. Основні завдання ERP:

- управління фінансами;
- - керування персоналом;
- - керування операціями (логістика, постачання, збут).

Основні завдання MES:

1. Оперативне керування виробництвом:
- виробничими замовленнями;
- запасами та незавершеним виробництвом;
- технологічними маршрутами;
- операціями;
- сертифікатами;
- специфікаціями;
- рецептами;
- трудовими ресурсами;
- документами;
- відстеження виробничих подій;
- відстеження життєвого циклу продукції.

2. Аналіз ефективності технологічного устаткування. 3. Управління якістю продукції.

Основними завданнями SCADA є:

1. Візуалізація технологічного процесу (HMI).

2. Оперативне керування технологічним процесом.

3. Управління аварійними повідомленнями та подіями.

4. Аналіз історичних даних (трендів).

5. Генерація звітів.

Система MES застосовується для внутрішньоцехового управління виробництвом: побудови виробничих розкладів та контролю їх виконання на кожній ділянці, одиниці обладнання:

- щоденне планування робіт з дільниць та окремих одиниць обладнання з урахуванням плану цеху на поточний період;

- оперативна зміна добового плану з урахуванням поточної

ситуації в цеху (проломка обладнання, нестача інструменту або матеріалу, відсутність співробітника тощо);

- диспетчеризація та облік виконання робіт, візуалізація виробничого процесу у реальному часі;

- отримання інформації про реальне завантаження обладнання у цехах;

- скорочення паперового документообігу за рахунок ведення внутрішньоцехових документів в електронному вигляді (змінно-добових завдань, маршрутних карток, різних виробничих журналів);

- ведення заявок на отримання інструменту та оснащення в інструментальних коморах.

Система SCADA – диспетчерське управління та збирання даних, призначена для збору, обробки та відображення інформації у виробничих цехах на технологічному рівні виробництва. З її допомогою можна відстежувати стан обладнання, а саме: яке обладнання зараз працює, яке вимкнено, яке простоює і чому це відбувається.

Для забезпечення інтеграції та синхронізації обміну даних розробники більшості ERP та MES-систем орієнтуються на міжнародний стандарт МЕК 62264, що включає опис об'єктів, атрибутів і моделей інтеграції.

Асоціація MESA (www.mesa.org) пропонує таке визначення MES: це «...система, що складається з набору програмних та апаратних засобів, що забезпечують функції управління виробничою діяльністю – від замовлення виготовлення партії продукції і до завершення виробництва. Використовуючи своєчасні та точні дані, MES-система ініціює, веде, реагує на ситуацію, що змінюється, складає звіти про виробничі процеси в міру їх протікання. MES-система дозволяє обмінюватися MESA інформацією про виробничі процеси з іншими інженерними та бізнес-підрозділами підприємства та ланцюжками його поставок через двонаправлені канали зв'язку».

Асоціація визначила 11 основних функцій MES:

1. Контроль стану та розподіл ресурсів (RAS, Resource Allocation and Status) – управління ресурсами виробництва (машинами, інструментальними засобами, методиками робіт, матеріалами, обладнанням) та іншими об'єктами (наприклад, документами про порядок виконання кожної) виробничої операції). У рамках цієї функції описується детальна історія ресурсів та гарантується правильність налаштування обладнання у виробничому процесі, відстежується стан обладнання у режимі реального часу.

2. Оперативне/детальне планування (ODS, Operations/Detail Scheduling) – оперативне та детальне планування роботи, засноване на пріоритетах, атрибутах, характеристиках та властивостях конкретного виду продукції, детальний та оптимальний розрахунок завантаження обладнання під час роботи конкретної зміни.

3. Диспетчеризація виробництва (DPU, Dispatching Production

Units) – поточний моніторинг та диспетчеризація процесу виробництва, відстеження виконання операцій, зайнятості обладнання та людей, виконання замовлень, обсягів, партій, контроль у реальному часі виконання робіт відповідно до плану. У режимі реального часу відстежуються всі зміни, що відбуваються, і вносяться коригування в план цеху.

4. Управління документами (DOC, Document Control) – контроль змісту та проходження документів, які повинні супроводжувати виріб, включаючи інструкції та нормативи робіт, способи виконання, креслення, процедури стандартних операцій, програми обробки деталей, записи партій продукції, повідомлення про технічні зміни, передачу інформації від зміни до зміни, а також забезпечення можливості вести планову та звітну цехову документацію. Передбачається архівування інформації.

5. Збір та зберігання даних (DCA, Data Collection/Acquisition) – інформаційна взаємодія різних виробничих підсистем для отримання, накопичення та передачі технологічних та керуючих даних, що циркулюють у виробничому середовищі підприємства. Дані про хід виробництва можуть вводитись як вручну персоналом, так і автоматично із заданою періодичністю з АСУТП або безпосередньо з виробничих ліній.

6. Управління персоналом (LM, Labor Management) – надання інформації про персонал із заданою періодичністю, включаючи звіти про час та присутність на робочому місці, стеження за відповідністю сертифікації, а також можливість враховувати та контролювати основні, додаткові та сумісні обов'язки персоналу, такі як виконання підготовчих операцій; розширення зони роботи.

7. Управління якістю продукції (QM, Quality Management) – надання даних вимірювань про якість продукції, у тому числі виробничого рівня, забезпечення належного контролю якості та особливий контроль «критичних точок». Може запропонувати дії щодо виправлення ситуації в даній точці на основі аналізу кореляційних залежностей та статистичних даних причинно-наслідкових зв'язків контрольованих подій.

8. Управління виробничими процесами (PM, Process Management) – відстеження заданого виробничого процесу, автоматичне внесення коректив чи пропозицію відповідного рішення оператору для виправлення чи підвищення якості поточних робіт.

9. Управління виробничими фондами (техобслуговування) (MM, Maintenance Management) – підтримка процесу технічного обслуговування, планового та оперативного ремонту виробничого та технологічного обладнання та інструментів протягом усього виробничого процесу.

10. Відстеження історії товару (PTG, Product Tracking and Genealogy) – надання інформації у тому, де й у порядку велася робота

з цією продукцією. Інформація про стан може включати звіт про персонал, що працює із цим видом продукції, компоненти продукції, матеріали від постачальника, партію, серійний номер, поточні умови виробництва, невідповідність встановленим нормам, індивідуальний технологічний паспорт виробу.

11. Аналіз продуктивності (PA, Performance Analysis) – звіти про реальні результати виробничих операцій, порівняння їх з попередніми та очікуваними результатами. Подані звіти можуть включати такі вимірювання, як використання ресурсів, наявність ресурсів, час циклу виробничого ресурсу, відповідність плану, стандартам та ін. насамперед за умов обмежених фінансових коштів (інвестицій).

Системи автоматизації (BI, ERP, MES, АСУ ТП) і рівні управління можна співвіднести у вигляді піраміди (див. рис. 2.2), в якій основи кожної її частин відображають обсяги інформації для прийняття рішень. Системи виду BI та ERP визначають стратегічний рівень управління, MES-системи задають тактичний рівень, а АСУ ТП – оперативний. Аналіз проблем, що виникають при інтеграції подібних систем, вимагатиме детального опису кожного рівня.

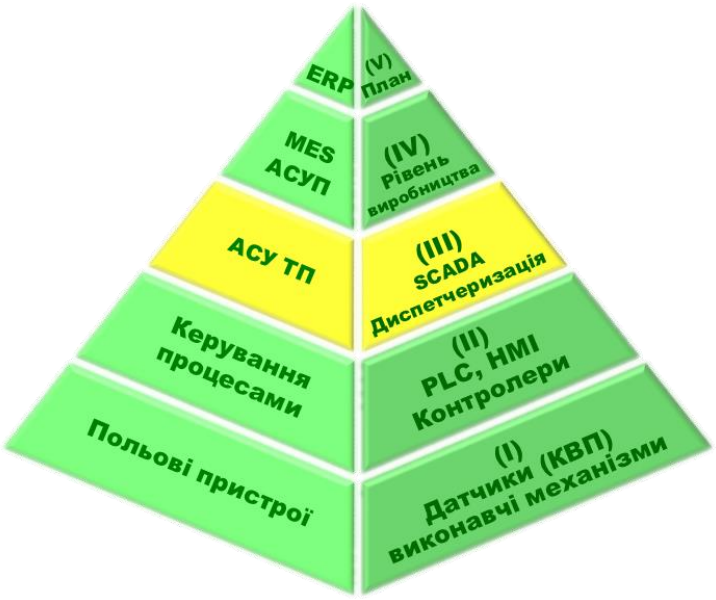


Рисунок 2.2 – Рівні автоматизації діяльності підприємства

Контур управління рівня АСУТП (технологічний) є найінтенсивнішим за обсягом інформації та найжорсткішим за часом реакції, що може становити хвилини, секунди і навіть мілісекунди. У верхньому рівні шару АСУТП - у SCADA-системах відбувається накопичення та обробка великої кількості технологічних параметрів та створюється інформаційна база вихідних даних для MES-рівня.

Контур управління рівня MES (оперативно-виробничий) спирається на відфільтровану та оброблену інформацію, що надходить

як від АСУТП, так і інших служб виробництва (постачання, технічної підтримки, технологічних, планово-виробничих і т.д.). Інтенсивність інформаційних потоків тут суттєво нижча та пов'язана із завданнями оптимізації заданих виробничих показників (якість продукції, продуктивність, енергозбереження, собівартість тощо). Типові часи циклів управління складають годинник, зміни, тижні. Оперативне управління виробництвом у цьому контурі управління здійснюється фахівцями, які детальніше, ніж вищий менеджмент, володіють виробничою ситуацією (керівники виробничих цехів, ділянок, основні технології, енергетики, механіки та інших.). Завдяки ефективним контурам управління MES забезпечується підвищення якості виробництв та ефективність прийнятих рішень у межах делегованих менеджменту цеху повноважень.

Контур управління рівня ERP (стратегічний) завдяки MES та АСУ ТП звільняється від вирішення оперативних завдань виробництва та забезпечує підтримку бізнес-процесів підприємства загалом. Потік інформації від виробничого блоку стає мінімальним і включає агреговану керуючу та звітну інформацію за стандартами ERP з типовими часами контролю тиждень, декада, місяць, квартал, а також «аварійні» подієві сигнали, що вимагають негайного втручання вищого менеджменту підприємства.

Спільне застосування BI, ERP, MES та АСУ ТП систем дозволяє вибудовувати єдину систему управління підприємством, у якій кожен рівень інтеграції виконує строго задану функцію: формування аналітичної звітності, ведення об'ємно-календарного планування, розрахунок оптимального виробничого розкладу та контроль технологічних процесів. ERP системи постійно розвиваються і внаслідок цього з'являються нові автоматизовані системи підприємства, що забезпечують управління взаємовідносинами з постачальниками та клієнтами, життєвим циклом продукції та ланцюжками постачання.

На сучасному ринку засобів автоматизації, незважаючи на безліч виробників обладнання, програмного забезпечення та великих інтеграторів, галузі відповідальності АСУТП та систем управління виробництвом (MES, ERP) досить усунуті один від одного, хоч і мають явні точки перетину. Понад те, елементи інформаційних систем виробничо-господарську діяльність найчастіше реалізуються у різні періоди часу (від введення об'єкта в експлуатацію до їх застосування може пройти досить тривалий час).

На початковому етапі АСУ ПП (ERP, MRP) та АСУ ТП розвивалися відокремлено та незалежно один від одного. Спочатку вони були підпорядковані єдиним цілям і завданням, залишалися слабко пов'язаними фізично й інформаційно, а частіше не пов'язані зовсім. Програмне забезпечення АСУ ПП та АСУ ТП досить довго розвивалося автономно та не передбачалася стандартизація каналів для обміну інформацією між ними. Технології, на яких вони проектувалися, не

враховували спільної роботи цих систем у єдиному інформаційному просторі. Комутаційне обладнання відповідало лише своєму рівню.

План виробництва кожного цеху формується у системі ERP і завантажується у систему MES. На основі інформації бази даних система може побудувати оптимальний виробничий графік для кожної ділянки основного виробництва. Використання систем MES та SCADA дозволяє значно спростити роботу всіх учасників виробничого процесу.

Оператори обладнання отримують завдання відповідно до виробничого плану цеху, мають можливість переглядати креслення та 3D-моделі установок, своєчасно повідомляти про необхідність ремонту та налагодження обладнання, заміни інструменту, отримувати звітність про виконані роботи за встановлений період часу.

Керівництво цеху має можливість планувати, управляти та контролювати хід виробництва в інформаційній системі, а також усі необхідні дані для оперативної реакції на можливі зміни у виробничому процесі (зміна планів, затримка у виконанні попередніх, вихід з ладу обладнання тощо). Диспетчеризація виробництва на технологічному та цеховому рівнях управління виконується з використанням людино-машинного інтерфейсу (HMI).

Виділяють керуючі, інформаційні та допоміжні функції HMI. Так SCADA дозволяє контролювати окремі технологічні змінні процесів, вести програмне управління групою обладнання, технологічними режимами чи окремими ділянками процесів, а також контролювати та вимірювати технологічні параметри процесів. Приклад графічного інтерфейсу SCADA-системи показаний на рисунку 2.3. Екранні форми проектуються відповідно до сценаріїв, які дотримуються регламентних документів технологічних процесів.

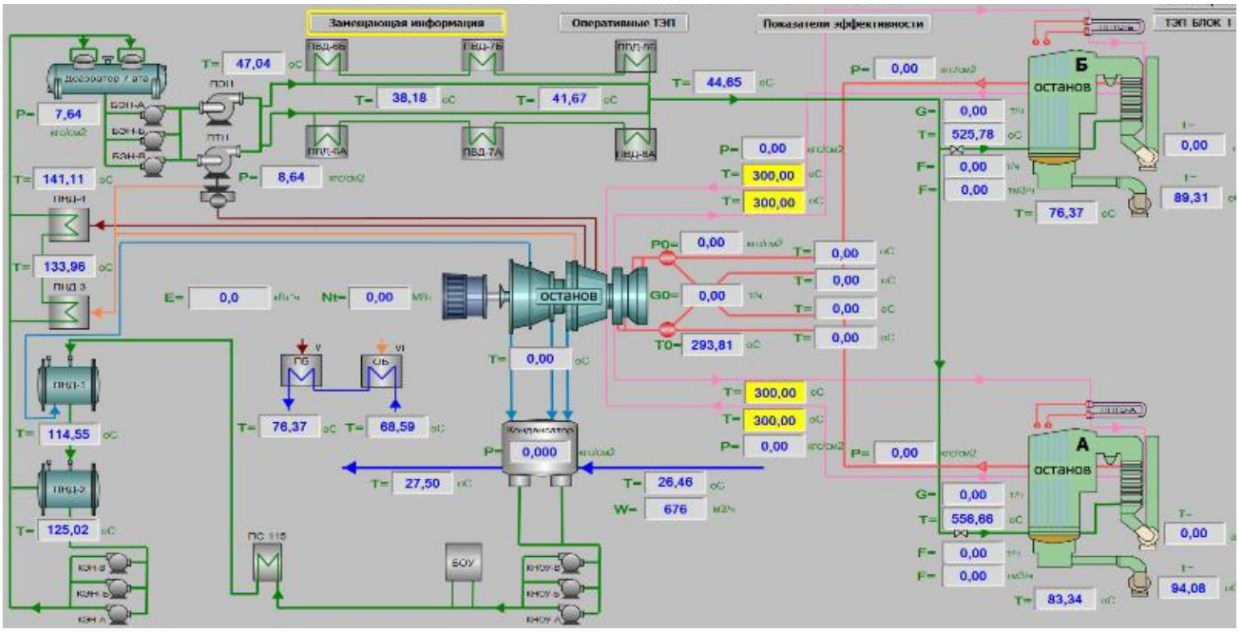


Рисунок 2.3 – Приклад графічного інтерфейсу АСУ ТП

MES-рівень визначає автоматизацію виробничої діяльності підприємства, що дозволяє в режимі реального часу планувати, оптимізувати, контролювати і документувати виробничі процеси від формування замовлення до випуску готової продукції. Виділяють такі функції HMI MES-систем, як автоматизований контроль стану та розподілу ресурсів, оперативне/детальне планування, диспетчеризація виробництва, управління якістю продукції, виробничими процесами, техобслуговуванням та ремонтом обладнання, а також аналіз продуктивності.

Рівень ERP-систем дозволяє реалізувати стратегію інтеграції логістичних (закупівлі, виробництво, збут), фінансових (дебітори, кредитори, банки) та кадрових функцій компанії, орієнтовану на оптимізацію ресурсів підприємства за допомогою спеціалізованого програмного забезпечення. ERP-системи здебільшого ведуть обробку транзакційних даних і відносяться до класу систем OLTP (OnLine Transactional Processing). Аналітична обробка транзакційних даних, зібраних засобами ERP-систем, ведеться лише на рівні OLAP (OnLine Analytical Processing), з допомогою автоматизованих BI-систем (Business Intelligence).

Спільне використання зазначених рівнів автоматизації формує єдине інформаційне середовище підприємства. Так, рівень АСУ ТП, заданий програмованими логічними контролерами, SCADA-системами та базами даних, дозволяє вести збір та обробку технологічних даних у режимі реального часу. Оброблена інформація передається на рівень MES-систем і використовується для оперативного управління виробництвом з урахуванням взаємозамінності та переналагодження обладнання. Оперативний план виробництва цього рівня співвідноситься з результатами роботи ERP-систем зі стратегічного планування та управління адміністративно-господарськими операціями компанії. Зведена аналітична звітність, отримана на основі транзакційних даних ERP-рівня, визначатиме фінальний крок автоматизації засобами BI-систем (рис.3.9). Описаний процес взаємодії рівнів інтеграції підприємства дозволяє сформулювати завдання, вирішення яких необхідне побудови виробничої інформаційного середовища. Якщо розглянути роботу інтегрованого середовища ERP, MES та АСУ ТП зверху-вниз, то ERP-системи формують календарний план виробництва на основі стандарту MRPII (Material Requirement Planning). Створений план, переданий на рівень MES, є основою для формування та подальшої оптимізації виробничого розкладу. Виробничий розклад визначає технологічні процеси, що проводяться та контрольовані на АСУ ТП рівні. Слід зазначити, що можливі різні сценарії інтеграції систем, включаючи повну її відсутність. В останньому випадку кожна система працюватиме незалежно: так, ERP-система використовуватиметься для об'ємного планування та фіксації результатів виробництва, MES – об'ємного/детального планування та

управління виробництвом, а АСУ ТП – для управління технологічними установками.

2.2 Об'єкт і суб'єкт цехового керування

Для того щоб ефективно управляти процесом виробництва або потоком робіт у виробництві, потрібне чітке визначення самого об'єкта керування. Таким об'єктом можна вважати деталь, з її різними, але в якійсь мірі формалізованими властивостями (креслення, технологічні маршрути, матеріали й т.п.). Суб'єктом же керування у виробництві очевидно можна вважати людину з його різними властивостями. Об'єкти й суб'єкти завжди створюють у виробництві системні взаємодії.

І якщо уважно придивитися до будь-якого заводу, то можна виявити якусь сформовану невипадковим чином «правильну систему керування цехом», сутність якої визначає суб‘ктивність. Як правило, суб'єкт сміло й наївно вважає, що управляє процесом виробництва деталей (об'єктів) і робить це ефективно, однак, можна ще сміливіше припустити, що не люди управляють виробництвом, а деталі, що перебувають у виробництві, управляють діями людей.

Слід прийняти до уваги, що всі дії керуючого процесом суб'єкта при виготовленні виробів можна досить просто підрозділити на: дії, що додають цінність кінцевому продукту, дії, що не створюють цінності, але неминучі за якимись причинами, і дії, що не додають цінності взагалі (повністю даремні дії). Виникає питання: як побачити пошуки дії або бездіяльності? Що насправді може бачити керівник підприємства, вдивляючись в «звітні цифри», що надходять із виробництва? Як можна забезпечити належну прозорість виробництва, оцінити його ефективність?

Основним критерієм ефективності організації виробничої системи цехового керування з дискретним, позаказним типом виробництва, є відношення часу дії, протягом якого створюється цінність при обробці деталі на верстаті (Т обр.), до часу втрат, тобто бездіям, коли цінність не створюється. До втрат часу слід віднести очікування деталлю наступної обробки, час переміщення від одного робочого місця до іншого, часу для здійснення контролю, втрат часу, викликаних несинхронністю потоків робіт, часу на виправлення браку деталей й т.п. (Т сум. втрат.). Це відношення можна визначити відповідним коефіцієнтом ефективності:

$$K_{еф} = (T_{обр.} / T_{сум.\ втрат})*100\ \%.$$

Наприклад, якщо сумарний час обробки однієї деталі на всьому технологічному маршруті становить 19,2 хвилин, а сумарний час знаходження деталі у виробництві від запуску в роботу до передачі на склад становить 4 робочих зміни по 8 годин (1920 хв), то:

$$К_{еф} = (19,2 \text{ хв} / 1920 \text{ хв}) * 100\ \% = 1\%.$$

Цей простий показник насправді говорить багато про що. І про те, як організована міжопераційна взаємодія в організації виробничого потоку, і про те, яка фондовіддача технологічного встаткування, він також пов'язаний з обсягом незавершеного виробництва й т.п. Ну й найголовніше – як ефективно й наскільки корисно використовується час у виробництві.

Метод тривіальний і простий. Його ще називають: «картування потоку створення цінності».

Якщо Кеф буде рівний 50%, то можна вважати, що ефективність організації виробничого потоку перебуває на світовому рівні. Але, якщо Кеф=1%, те це означає, що для досягнення світового рівня організованості, цикл виробництва однієї деталі потрібно скоротити за часом в 50 раз.

Якщо вироблений виріб складається з 10 деталей, а цикл виробництва цих деталей удалося скоротити в загальному потоці робіт в 50 раз, то сам виріб буде проводитися в 5 раз швидше, а нова виробнича система буде в 5 раз продуктивніше колишньої при тому ж устаткуванні й тих же його властивостях. І відбудеться це тільки за рахунок більш ефективної організації виробничого процесу.

Таким чином, аналіз співвідношення часу обробки деталі відносно часу втрат дозволяє виявити величезні резерви для росту продуктивності верстатної системи.

Повернемося до об'єкта керування – деталі. Деталь – це результат перетворення властивостей вихідного матеріалу, яке відбувається з ним у ході проходження по певному технологічному маршруту через робочі місця (РМ), їх ще називають робочими центрами.

Суть маршрутизації руху деталі така: добитися того, щоб технологічний процес (маршрут) був би поновлюваним, і повторювався б з тими самими тимчасовими параметрами обробки на кожному РМ маршруту. І тут виникають три непрості проблеми: варіабельність тимчасових параметрів обробок в умовах позаказного виробництва, точна поновлюваність процесу й те, що деталь попадає у виробництво, у якім уже обробляються інші деталі.

Нехай кожна деталь (або ж її партія), потрапляючи на різні РМ, обробляється з однаковим нормованим часом. Потік робіт на РМ можна представити у вигляді послідовності тимчасових відрізків (рис 2.4), величини яких будуть визначатися нормою часу обробки деталей.

Кольорові прямокутники, на рисунку 1.1 відповідають послідовно оброблюваним деталям 1, 2, 3 і т.д. і лежать на осі часу (абсцисі). Покроковий технологічний маршрут розташовується по осі ординат. Таким чином, ми одержуємо найпростішу графічну виставу про потік виробництва деталей в одному й тому ж технологічному ланцюжку подій. Така вистава називається діаграмою Гантта.

Деталі 1, 2, 3, 4, 5 і 6 мають рівні відрізки часу обробки на всіх РМ того самого технологічного маршруту. Деталь 1, пройшовши обробку на РМ 1 - відразу переходить на РМ 2 і так далі. Наступна за нею деталь 2 без тимчасових втрат так само переходить від одного РМ до іншого й коли деталь 5 зайде в процес обробки, завантаження всіх верстатів стане повним.

Представлена на рисунку 2.4 діаграма є моделлю організації виробництва деяких абстрактних деталей. Дану модель можна легко перетворити в конкретні завдання (виробничі розклади), які можна видати на кожне РМ ще до того, як процес виробництва настав. А вже потім, після старту завдань організувати різними способами зворотний зв'язок із РМ і вже в реальному часі відслідковувати хід виконання виробничих розкладів. Саме по такому принципу й улаштована MES-система.

Рисунок 2.4 – Ідеальний варіант розподілу деталей

З викладеного стає ясно, що в ідеальному варіанті виробничий розклад створюється дуже просто. Однак у реальності процес розподілу по робочих місцях викликає значні складності. Реальний варіант ілюструється рисунком 2.5.

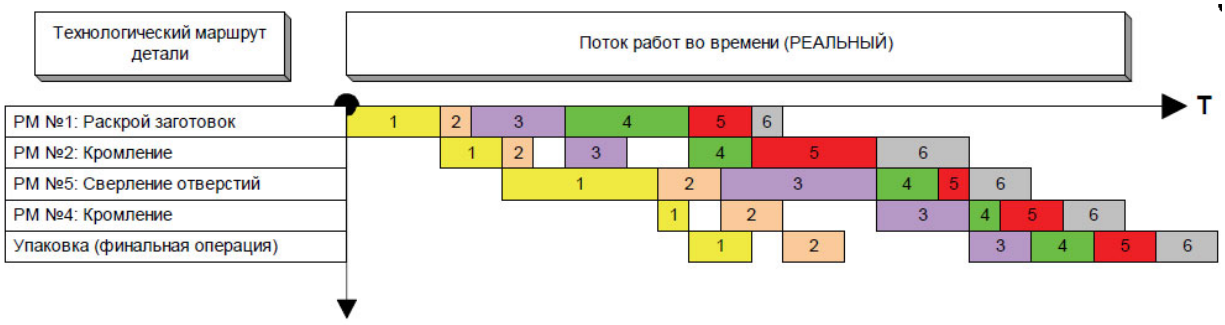


Рисунок 2.5 – Реальний розподіл деталей

Отут деталі мають уже різну трудомісткість і, відповідно, різні відрізки часу, відведені на їхню обробку, що викликає в технологічному ланцюжку тимчасові втрати (порожнечі на діаграмі рис. 2.5) і множинні

дисбаланси в завантаженні робочих місць. Так утворюються «блукаючі вузькі місця» (пробки в потоках), викликані очікуванням обробки, а також інші труднощі, що викликає втрати часу. Ці ситуації візуально проявляються по скупченню заготовок біля деяких верстатів.

Із виникаючими у різних місцях і в різний час виробництва труднощами й борються майстри, які управляють виробництвом в основному візуально. І ніхто з них не скаже точно, коли ж деталі пройдуть усі стадії обробки. Але ж деталі постійно надходять у виробництво й при позаказному виробництві міняються не тільки самі деталі, але й розмірність партій однакових деталей. Їх може бути тисячі, технологічних маршрутів – десятки, і багато з маршрутів у силу універсальності робочих місць перетинаються один з одним.

Таким чином, виробничий процес завжди представляє собою хаос, з яким намагаються управлятися в рамках суб'єктивних і візуальних уявлень.

Як же діють у такій ситуації виробничники? Дуже просто: вони працюють зі збільшеним міжопераційним запасом (заділом). Цей організаційний розв'язок застосовується повсюдно. Така організація виробництва зручна в умовах масового виробництва, але зовсім не ефективна в позаказному (дрібносерійному). Вона погіршує економічні показники підприємства в цілому, тому що приводить до значного зв'язування обігового капіталу. До того ж робота про запас може взагалі бути марною в умовах ринкових капризів.

Ще одним важливим організаційним недоліком виробництва є значне збільшення виробничого циклу, яким намагаються нівелювати диспропорції потоків робіт. А все це в комплексі збільшує строки виконання замовлення. І якщо конкурент робить ту ж продукцію й поставляє її на ринок швидше – ринок, природно, прийде до нього. На сучасному ринку клієнт віддає перевагу коротким строкам виконання своїх замовлень і вимогливий до ціни.

Таким чином, можна зробити висновок про те, що в умовах агресивного ринкового середовища саме способи ефективної організації виробництва стають одним з основних конкурентних переваг промислових підприємств. Організація масового й крупносерійного виробництва ефективна лише в дуже окремих випадках. Основна ж маса виробництв повинна вміти підтримувати дрібносерійну або одиничну організаційні форми керування. Виходячи із цього, цілком логічного твердження розглянемо можливості систем керування виробництвом на базі логістичної теорії й теорії розкладів, які використовуються при створенні MES-систем.

3 РОЗРОБКА ТЕХНОЛОГІЇ ШТУЧНИХ АГЕНТІВ

Розробка технології штучних агентів, створення багатоагентних систем (МАС) у віртуальних середовищах представляє найбільш важливу та багатообіцяючу галузь розвитку нових інформаційних та комунікаційних технологій (НІКТ), де сьогодні формуються кіберфізичні системи, відбувається інтеграція сучасних мережевих WWW-технологій, методів та засобів штучного інтелекту (ШІ), включаючи великі бази даних/знань, багатокомпонентні вирішувачі.

Агентом є все, що може розглядатися як розумна сутність, що сприймає навколишнє середовище за допомогою сенсорів і впливає на нього за допомогою виконавчих механізмів. Найпростішим видом агента є простий рефлексний агент. Такі агенти вибирають дії з урахуванням поточного сприйняття середовища, ігноруючи історію попередніх актів сприйняття. Наприклад, агент-пилосос, являє собою простий рефлексний агент, оскільки його рішення засновані тільки на інформації про поточне місцезнаходження та про те, чи воно містить сміття.

Такий вид агентів використовує зв'язок типу умова-дія. Тобто, якщо виконується якась умова, то у відповідь на нього агент використовує цю дію. Такий зв'язок дуже часто використовують люди, наприклад, водії автомобілів: якщо водій бачить, що попереду машина гальмує, то він теж починає гальмувати. Це найпростіший тип зв'язку «умова-дія».

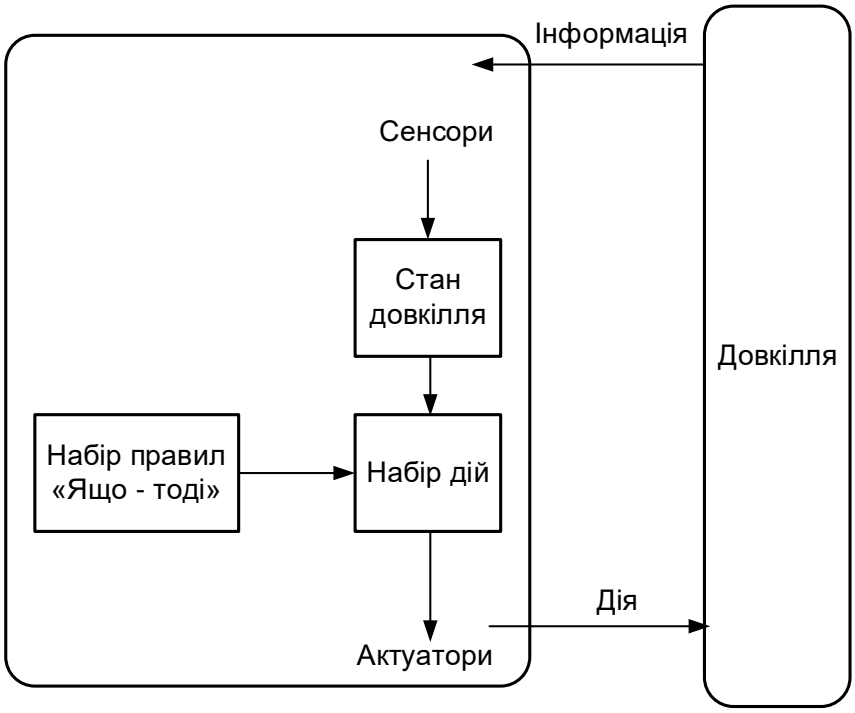


Рисунок 3.1 - Функціональна схема простого інтелектуального агента

Суть агентно-орієнтованого підходу дуже проста. Замість того, щоб намагатися формалізувати такі складні поняття, як «розум» та «інтелект», треба зробити щось простіше і корисніше. Цим простим та корисним виявилося поняття «раціональність» або «раціональна поведінка» реінжинірингу бізнесу та побудови віртуальних підприємств, імітаційного моделювання інтегрованих виробничих систем та електронної торгівлі, організації роботи колективів роботів та розподіленої (сумісної) розробки комп'ютерних програм.

Існує кілька міжнародних підходів до створення мультиагентних систем:

• OMG MASIF, створений Object Management Group, основу якого поняття мобільний агент;

• специфікації FIPA (Foundations for Intelligent Physical Agents), що ґрунтуються на припущенні про інтелектуальність агента;

• DARPA стандарти, розроблені дослідницьким підрозділом Пентагону – Агентством Передових Оборонних Наукових Досліджень (Defense Advanced Research Projects Agency).

Прості рефлексні агенти характеризуються чудовою особливістю, що вони надзвичайно прості. Робот-пилосос працює тільки якщо правильне рішення може бути прийняте на основі виключно поточного сприйняття, інакше кажучи, тільки якщо середовище таке, що повністю спостерігається.

Внесення навіть невеликої частки не спостерігальності може спричинити серйозне порушення його роботи. Прості рефлексні агенти є надзвичайно примітивними, вони обмежені у функціоналі. Для простих рефлексних агентів, що діють у середовищах, які не повністю спостерігається - характерні попадання в нескінченні цикли.

У повсякденному житті до ухвалення рішення завжди потрібно достатньо інформації з довкілля. Нехай людина підходить до перехрестя, вона має на вибір три напрямки руху. Що вибрати? І в цьому випадку він звертається до своєї мети. В аналогічній ситуації агенту також може знадобитися не тільки інформація про світ, внутрішній стан, а й інформація про мету. Тоді програма агента комбінуватиме всі види інформації для вибору дій, які дозволять досягти мети.

Завдання вибору дій на основі мети вирішується досить просто, коли досягнення мети стає результатом єдиної дії, але коли досягнення мети ускладняється, агенту потрібно розглянути послідовності дій, щоб знайти потрібний спосіб досягнення мети.

Ухвалення рішень на основі мети повністю відрізняється від правила «умова-дія». Головною відмінністю від стандартного правила «умова-дія» є те, що агент на основі мети повинен відповідати на запитання: «Ця дія дозволить досягти мети?» або "Що буде, якщо я зроблю так?".

Приклад поведінки такої моделі агента наведено рисунку 3.2.

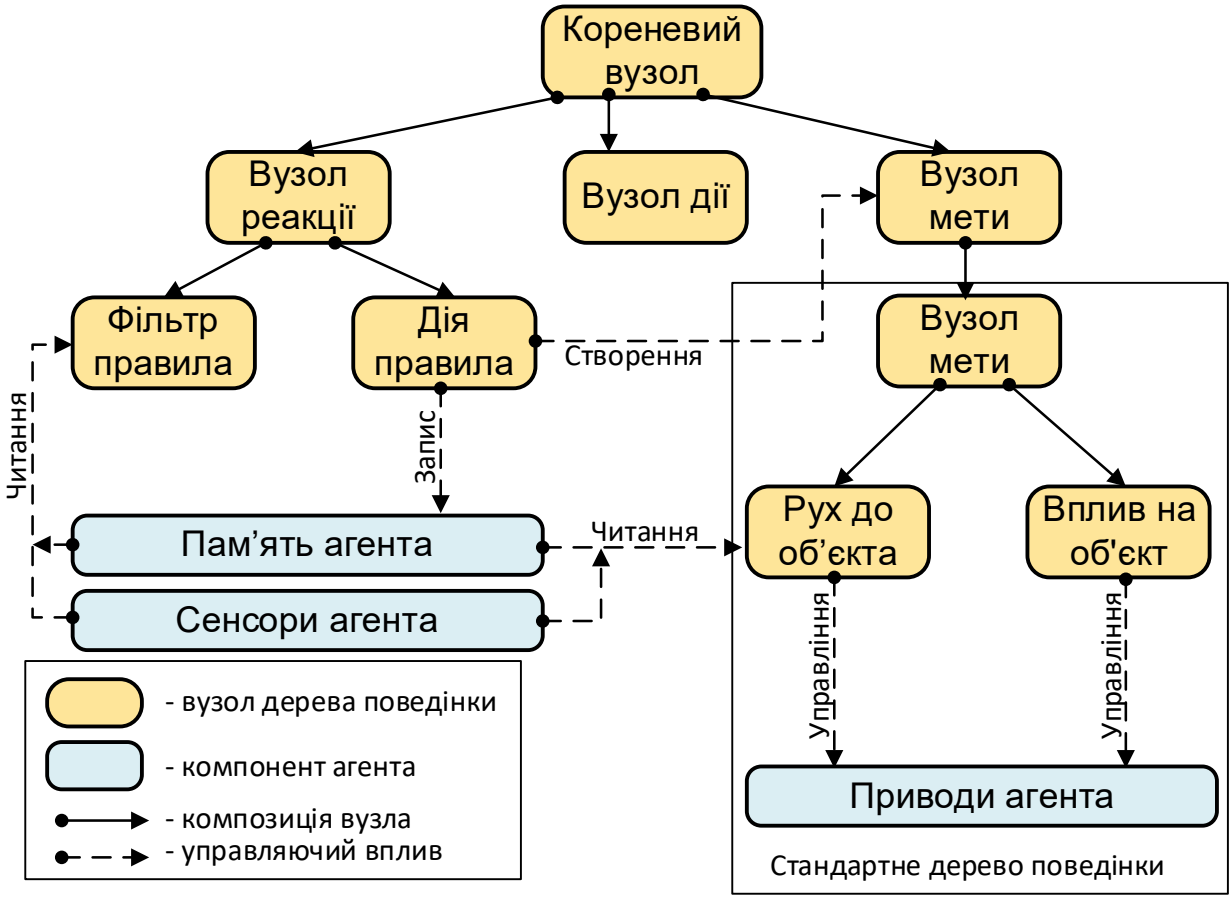


Рисунок 3.2 - Модель поведінки агенту на основі досягнення мети

Агент, який використовує ситуаційний аналіз та планування дій для досягнення мети, стає більш гнучким у своїх діях, оскільки він використовує знання. Знання дозволяють реалізовувати алгоритми виконання дій, які можуть змінюватися, якщо змінюється довкілля. Агент перебудовує пріоритети, змінює правила «умова-дія» і, таким чином, може обирати найкращу дію і цим значно відрізняється від агентів рефлексивного типу.

Однак ці види агентів мають одну загальну ваду - відсутність можливості навчатися. Навчання є первинною якістю інтелектуального агента. Шляхом навчання, агент може функціонувати в спочатку невідомих йому варіантах середовища і ставати більш корисним і значущим, ніж він був спроектований спочатку. Таким чином, агент набуває особливостей штучного інтелекту (ШІ). На рисунку 3.3 показана структура такого агента навчання.

Агент, що навчається, має чотири концептуальних компоненти: блок дій, навчальний компонент, блок критики (оцінки ситуації) і генератор потоку завдань.

Сенсори безперервно вимірюють особливості навколишнього середовища. Блок дій впливає середу з допомогою виконавчих механізмів.

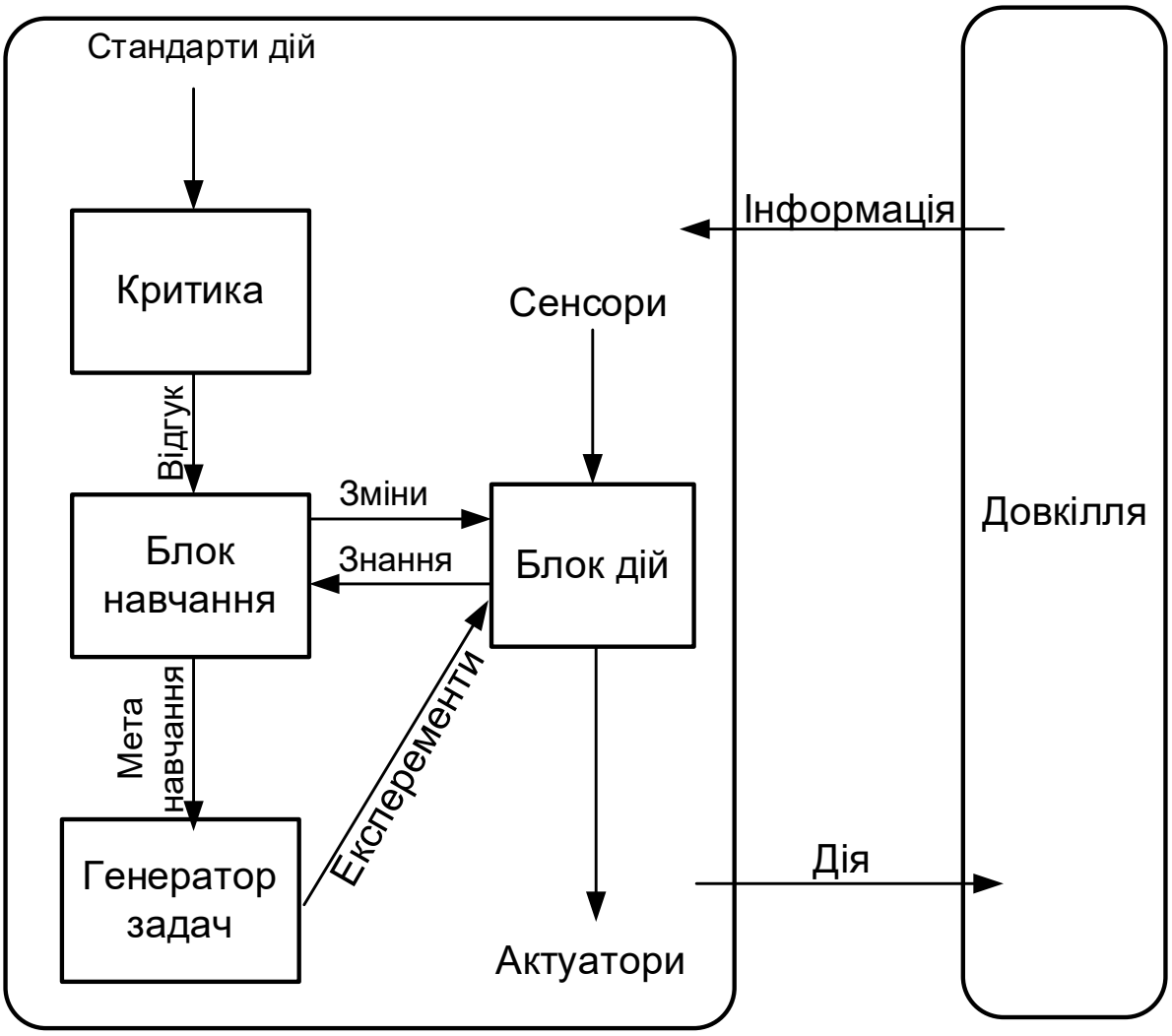


Рисунок 3.3 – Блок схема агента навчання

Критик-блок виконує функції оцінювача дій агента з урахуванням встановлених розробником стандартів дій. Цей блок необхідний у цій структурі, оскільки агенту необхідно оцінювати успішність його дій.

Блок навчання відповідає за внесення удосконалень та вибір своїх дій. Навчальний компонент використовує інформацію від критик-блоку з оцінкою успішності (ефективності) дій агента і містить продуктивну частину для завдання необхідних дій.

Генератор завдань служить для вибору дій, які мають призвести до отримання нового інформаційного досвіду. Оскільки продуктивний компонент вибирає тільки найбільш ефективні дії, це може звести всю інтелектуальність агента до одних і тих же дій, вважаючи, що вони є найкращими. Тому генератор проблем дозволяє виконувати експерименти на вибір інших дій, які можуть бути неоптимальними на початку, але можливо будуть найкращими в кінцевому результаті.

Модуль прийняття рішень є найважливішим компонентом агента. Для його реалізації застосовують модель дерева поведінки. Дерево поведінки відповідає за миттєві реакції агента на події, що відбуваються,

послідовне виконання спланованих дій і контролює досягнення цілей.

Одним з найбільш ефективних способів організації роботи в умовах часткового спостереження середовища є відстежуваність агентом частини світу, яка сприймається ним в даний момент часу.

На апаратному рівні інтелектуальний агент складається з: сенсорів, електричних приводів виконавчих пристроїв, пам'яті та моделі поведінки (програми агента). Сенсори та електричні приводи є інтерфейсом для взаємодії між агентом та навколишнім середовищем. У пам'яті зберігаються результати сприйняття довкілля сенсорами. Програма агента реалізує його модель поведінки та є найскладнішим компонентом. Внутрішнє уявлення програми агента насамперед залежить від його завдання.

Агенти можуть формувати спільноти (мережі). Такі мультиагентні об'єднання є особливо корисними при побудові віртуального шару кіберфізичних систем. Особливість такого підходу до побудови КФС у тому, що у нижньому рівні її піраміди будуються агентноориентированные моделі поведінки окремих сутностей у конкретному технологічному процесі. Причому кожен із агентів слідує своїм власним правилам, живе у технологічному середовищі виробництва та взаємодіє з середовищем та іншими агентами з урахуванням індивідуальної логіки поведінки агентів.

Передбачається, що реалізується здатність взаємодії між агентами, коли один агент може виробити запит іншому агенту на передачу деяких даних або виконання певних дій. При цьому завдання може бути розбите на кілька підзадач, які розподіляються між іншими агентами.

Мультиагентные системи виконують дії у реальному часі є ефективним інструментом керувати складними процесами, у яких бере участь багато активів виробничих процесів. До таких процесів належать потоки виробництва виробів машинобудування, інфраструктурні потоки міського руху, логістичні системи та ін.

Методи мультиагентного підходу використовуються також для пошуку та опрацювання інформації в інформаційних мережах, системах управління автономними роботами. Перспективним напрямком розвитку мультиагентних систем є розробка безпілотних автомобілів та літальних апаратів.

Схема колективної взаємодії інтелектуальних агентів представлена рисунку 3.4. Користувач може бути як людина, так і фізичний об'єкт.

Використання ідеї колективної поведінки агентів призводить до безлічі рішень. Серед них, зокрема, у CPS слід виділити такі рішення, як формування спільних планів дій, можливість урахування спільних інтересів агентів, синхронізація спільних дій. У процесі виконання колективних процесів. Можлива поява конфліктуючих цілей агентів, виникнення конкуренції за спільні ресурси. Ці проблеми можуть

вирішуватися за допомогою організації переговорів про спільні дії, розпізнавання необхідності кооперації, вибір відповідного партнера, навчання поведінки у колективі, декомпозиція завдань і поділ обов'язків, правила поведінки у колективі, спільні зобов'язання тощо.

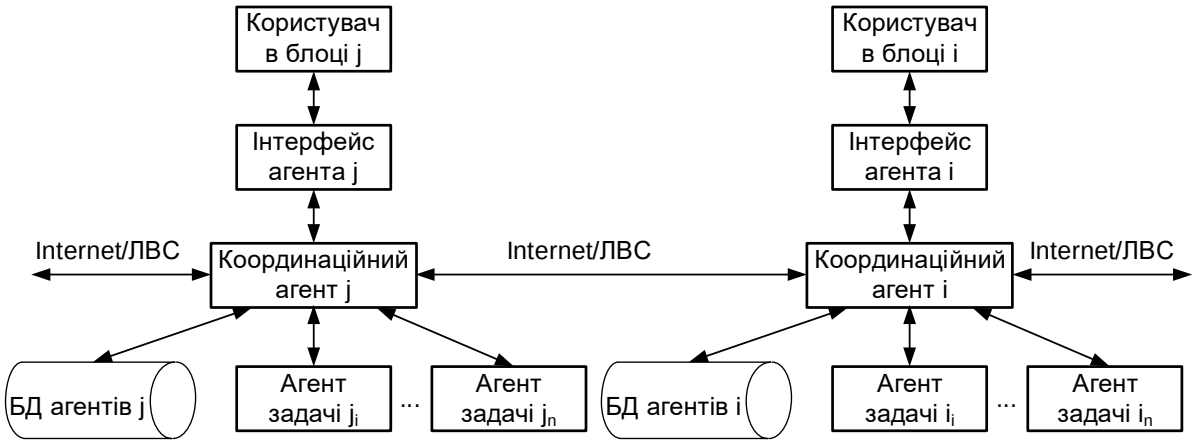


Рисунок 3.4 – Схема колективної взаємодії інтелектуальних агентів

У процесі формування кооперативного рішення спільнотою агентів у CPS-середовищі виділяють чотири етапи:

1. *Розпізнавання ситуації*. Процес вибору рішення у мультиагентному середовищі починається тоді, коли агент розпізнає доцільність кооперативної дії. Наприклад, агент із встановленим сервісним набором встановлює мету, досягти якої через свою обмеженість він не здатний, або для її досягнення вибирає кооперацію.

2. *Формування мультиагентного співтовариства*. На цій стадії агент, який встановив можливість спільної дії, шукає партнерів. При успішному завершенні цієї стадії утворюється група агентів, мають спільні послуги для колективних процесів.

3. *Формування спільного плану*. Цей етап, на якому агенти домовляються, виробляють спільний план, який на їхнє переконання приведе до бажаної мети.

4. *Спільні дії*. На цьому етапі агенти діють згідно з виробленим планом, підтримуючи взаємодію один з одним, згідно з прийнятими на себе зобов'язаннями.

Формування спільного плану починається за умови, якщо розпізнано ситуацію та сформовано мультиагентну команду для виконання встановленої мети. Однак колективні дії не можуть початися доти, доки в групі не буде досягнуто згоди, що конкретно робитиме кожен агент. Для вироблення такої угоди є стадія формування спільного плану. Мультіагентні переговори є механізмом вироблення такої угоди. На стадії формування спільного плану агенти групи здійснюють спільні спроби досягти такого стану групи, в якому всі агенти узгодили б спільний план. Процедура узгодження реалізується спеціальним

алгоритмом пошуку угоди.

Під час переговорів агенти пропонують плани, уточнюють їх з іншими агентами, модифікують запропоновані плани, доки всі агенти не погодяться з єдиним планом.

При успішному завершенні стадії планування починається стадія спільних дій. При нормальному ході цього процесу дії виконуються згідно з прийнятим планом аж до його завершення.

В даний час багатоагентна модель широко застосовується при проектуванні систем автоматизації виробництва на різних рівнях. Зручність такого підходу та широта його використання обумовлені схожістю багатоагентної моделі з реальними процесами, що відбуваються в людському середовищі. Дійсно, в класичній багатоагентній системі під агентом розуміється певна розумна сутність, як правило, активна і здатна взаємодіяти з навколишнім середовищем. Інтелектуальна поведінка цих сутностей підтримується спільною роботою таких компонентів, як блок вирішальних правил для обчислення плану, блок правил для управління завданнями, їх декомпозицією та розміщенням, а також блок правил для підтримки угод з іншими агентами під час кооперативного вирішення завдань.

Поведінка реалізується за допомогою багаторівневої системи управління (див. рис. 3.5), яка реагує на зміну стану робочої пам'яті (при надходженні нових результатів розв'язання задачі, встановленні цілей або повідомлень, а також зміні наявних даних, міжагентських угод або станів завдань).

У такій багаторівневій структурі виділяються такі рівні управління мультиагентної команди:

1. Рівень специфічних предметних знань, у якому містяться предметні знання.

2. Рівень знань про процедури виведення. Цей рівень містить декларативні правила висновку, які мають застосовуватися до предметних знань щодо конкретного об'єкта, щоб вивести нові дані. Цей рівень є основою управління мультиагентної команди.

3. Менеджер завдань відповідальний за декомпозицію завдань на підзавдання та його розподіл за відповідними агентами, і навіть за управління переходами станів завдань. Управління кооперацією агентів використовує механізм, «...заснований на взаємних зобов'язаннях агентів (будь-який агент згоден робити схему дій, яка має на меті виконати завдання за відповідний час), та угоди про те, за яких умов агент може відмовитися від своїх зобов'язань і як він повинен поводитися по відношенню до інших агентів, коли такі обставини виникнуть».

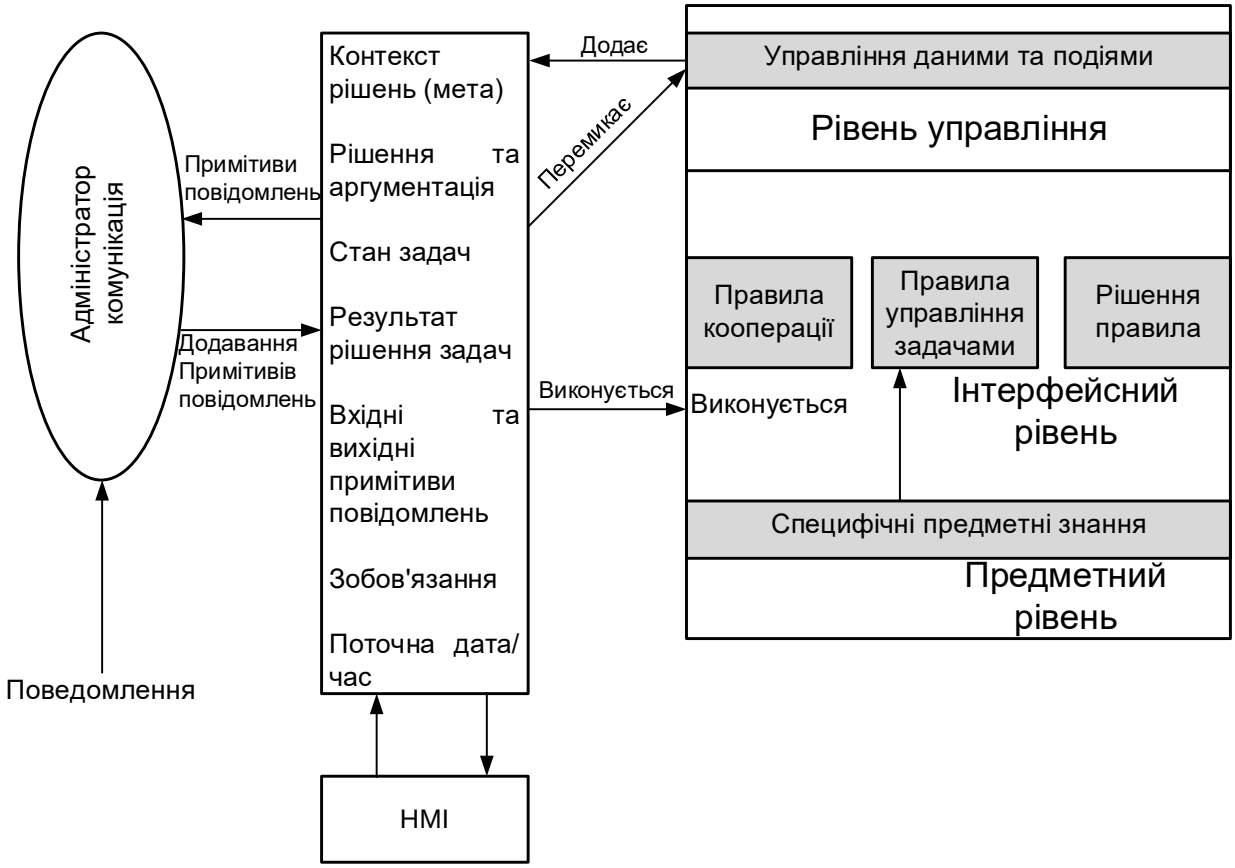


Рисунок 3.5 – Система управління мультиагентною командою

4. Рівень знань, що керують, який використовує онтологію предметних знань, щоб генерувати схему виведення. На цьому рівні додаються нові знання.

Такий функціональний поділ знань на предметні знання, знання про процедури виведення та керуючі знання структурує належним чином їх подання, повторне використання та їх використання для досягнення цілей.

Будучи об'єднаними у колективи, такі агенти здатні вирішувати завдання набагато складніші, ніж міг би вирішити один агент.

Інтелектуальні агенти можуть отримувати дані від фізичних пристроїв технологічних процесів фізичного світу, а й інших віртуальних агентів, і навіть від сервісних запитів інформаційних систем управління виробництвом.

Перспективною віхою розвитку автоматизації підприємства в даний час є впровадження технологічних процесів, що самоорганізуються. Така технічна самоорганізація в автоматизації відноситься до ультрасучасних методів побудови автоматизації технології та виробництва.

Агентно-орієнтований підхід, заснований на використанні інтелектуальних (раціональних) агентів, це розуміння обчислювальної частини КФС як ШІ, планування здатності досягати поставленої мети.

Організація архітектури агентів на принципах штучного інтелекту (ШІ) має переваги з погляду зручності використання методів та засобів символьного представлення знань, розроблених у рамках штучного інтелекту.

Штучний інтелект - область інформатики, що займається розробкою інтелектуальних комп'ютерних систем, що мають можливості людського розуму (розуміння мови, навчання, здатність розмірковувати та вирішувати проблеми).

Термін «штучний інтелект» стосується конкретної галузі обчислювальної техніки, яка фокусується на створенні систем, здатних збирати дані та приймати рішення та/або вирішувати проблеми.

Приклад базового ШІ є комп'ютер, який може приймати 1000 фотографій кішок для введення, визначати, що робить їх схожими, а потім знаходити фотографії кішок в інтернеті. Комп'ютер навчився, наскільки це можливо, як виглядає фотографія кішки та використовує цей новий інтелект, щоб знайти речі, які схожі. Структура базового рівня області штучного інтелекту показано рисунку 3.6.

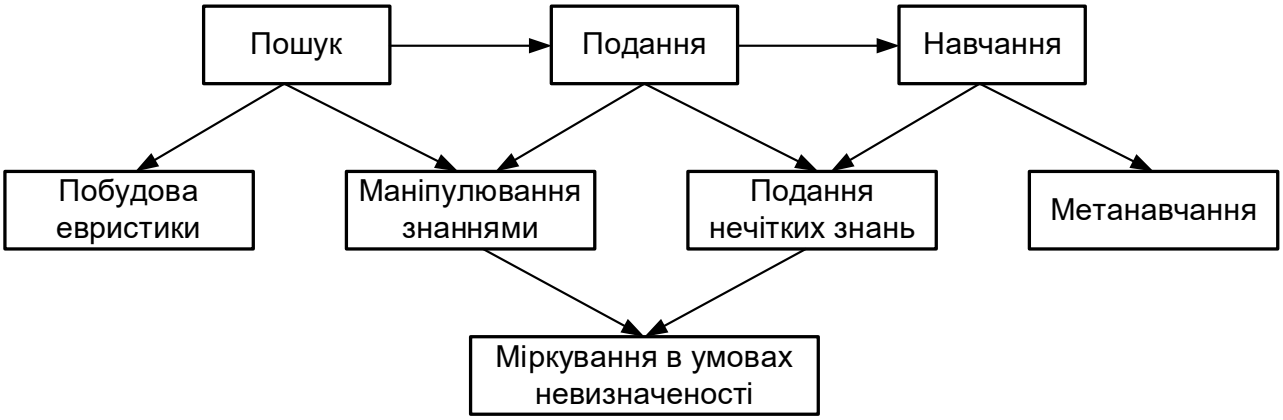


Рисунок 3.6 – Структура базового рівня галузі штучного інтелекту

ШІ має властивість автономності. Автономність означає, що конструкція ШІ не потребує допомоги людей. Безпілотні автомобілі ілюструють термін «автономний» різною мірою. Автономія четвертого Рівень являє собою транспортний засіб, який не потребує кермового колеса або педалей: йому не потрібна людина всередині нього, щоб працювати на повну потужність. Автомобіль, який працює без водія і не потребує підключення до будь-якої мережі, сервера, GPS або іншого зовнішнього джерела для функціонування - рівень автономії п'ять.

Все, що виходить за межі цього, можна було б назвати розумним, і, незважаючи на стрибки, які були зроблені нещодавно в області ШІ, сингулярність (подія, що представляє ШІ, яка стає самосвідомою) на даний момент є суто теоретичною.

Основними властивостями ШІ є: розуміння мови, навчання та здатність мислити та діяти. ШІ - комплекс споріднених технологій і

процесів, що розвиваються якісно та стрімко. Прикладами ШІ є (див. рис. 3.7):

- обробка тексту природною мовою;
- машинне навчання;
- експертні системи;
- віртуальні агенти (чат боти та віртуальні помічники);
- системи рекомендацій.

Фахівці можуть створити занадто потужний штучний інтелект, який спрямований на досягнення своїх цілей. І якщо ці цілі не збігатимуться з людськими, то люди будуть мати проблеми. Ось чому перейнялися міжнародні комісії з ШІ.

Рисунок 3.7 - Технології штучного інтелекту широко потрібні в різних галузях.

Перспективи розвитку ШІ несуть як надії людства, а й потенційні небезпеки.

25 квітня 2018 року Європейська комісія представила звернення «Штучний інтелект для Європи», адресоване Європарламенту, Раді Європи, європейським економічним та соціальним комітетам та комітетам регіонів. У документі порушуються питання майбутньої політики ЄС щодо ШІ, розвитку технологічної та індустріальної бази, встановлення етичних норм та норм відповідальності щодо ШІ. До Звернення додається проект документа «Відповідальність щодо нових цифрових технологій», в якому подано аналіз існуючих правил і принципів чинного законодавства, аналіз виробленої практики та ін.

Відповідно до цих рекомендацій Інститут інженерів електротехніки та електроніки (IEEE) розробив три нові стандарти етики для штучного інтелекту:

1. Стандарт для етичного впливу роботизованих та інтелектуальних систем. У цьому стандарті розглядаються дії ШІ, які приховано або явно впливають на поведінку та емоції людини. Щоб роботи дотримувалися у цьому питанні загальноприйнятої етики та

принципів моралі, інженери та філософи повинні разом займатися розробкою таких систем.

2. Стандарт відмовостійкості. Роботизовані системи в процесі роботи потенційно можуть завдати шкоди людям та навколишньому середовищу, тому стандарт для створення ефективних заходів безпеки, які знижують ризик помилок, та безпечного припинення експлуатації скомпрометованих систем встановлює чіткі процедури оцінки, тестування та сертифікації відмовостійкості роботизованих систем.

3. Стандарт для впливу ШІ на добробут суспільства. Творці ШІ – програмісти, інженери, технологи – повинні враховувати, як вироблені ними пристрої змінять добробут людей з погляду продуктивності праці та економічного зростання. У цьому стандарті визначається, які показники людського добробуту необхідно брати до уваги при впровадженні тих чи інших інтелектуальних систем. Це забезпечить підґрунтя для узгодження даних між різними фахівцями. Ще одним прикладом етичних норм для ШІ є корейський статут для

роботів (Korean Robot Ethics Charter). Документ включає 7 статей, які закріплюють наступні етичні стандарти для роботів (приблизний текст):

• стаття 1 (мета): основою етики робота є прагнення спільного процвітання людини і машини;

• стаття 2 (загальний принцип людського буття та робот): людина та робот повинні підтримувати гідність один одного;

• стаття 3 (людська етика): при виробництві робота людина повинна шукати натхнення в найкращих можливих образах;

• стаття 4 (роботизована етика): робот має бути людині другом, помічником та партнером; робот не повинен завдавати шкоди людям;

• стаття 5 (етика виробника): виробництво роботів має бути спрямоване на захист людей та підвищення їхньої гідності;

• стаття 6 (етика користувачів): споживач повинен добре ставитися до роботи, не допускати незаконного обігу роботів;

• стаття 7 (обіцянка виконання): уряд і місцеві органи влади, щоб втілити дух статуту, повинні забезпечити контроль за дотриманням правил етики стосовно робіт.

Найбільш важливою частиною ШІ є алгоритм. Це математичні формули та/або команди програмування, які інформують звичайний неінтелектуальний комп'ютер про те, як вирішити проблеми зі штучним інтелектом.

Алгоритм – правила, які вчать комп'ютери. Штучний інтелект – система, яка може навчитися вчитися. Люди пишуть вихідні алгоритми системи, яка дозволяє комп'ютеру згодом писати власні алгоритми без додаткового контролю чи взаємодії з людиною. Цей процес дозволяє ШІ постійно вчитися і вирішувати нові проблеми всередині середовища, що постійно змінюється, ґрунтуючись на зборі даних, що триває. Виділяють два основні підходи до розробки штучного інтелекту:

• низхідний – створення експертних систем, баз знань та систем логічного висновку, що імітують високорівневі інтелектуальні процеси: мислення, міркування, мовлення, емоції, творчість тощо;

• висхідний – вивчення нейронних мереж та еволюційних обчислень, що моделюють інтелектуальну поведінку, а також створення відповідних обчислювальних систем, таких як нейрокомп'ютер чи біокомп'ютер.

Складно назвати час, коли плоди уяви розробників ШІ знайдуть фізичне втілення. Прогресувати потрібно не лише технологіям, а й людині. Соціум має бути готовим прийняти «залізний» світ та інтелектуальну націю пристроїв. На період адаптації потрібен час. Щоб люди почали довіряти роботизованим поліцейським, лікарям та водіям, їх штучний інтелект має дорівнювати людському. У той же час, чи зможе недосконала людина створити досконалу систему? Чи зможе відстежити ту межу, де штучний інтелект – друг, а не небезпека? І чи зможе уникнути технічної залежності?

4 ІНТЕРНЕТ РЕЧЕЙ (INTERNET OF THINGS)

4.1 Історія Інтернету Речей

Термін «Інтернет речей», зобов'язаний своєю появою Кевіну Ештону, який в 1997 р, працюючи на компанію Proctor and Gamble, застосував технологію радіочастотної ідентифікації (RFID) для керування системою поставок. Завдяки цій роботі в 1999 році його запросили в Масачусетський технологічний інститут, де він з групою однодумців організував дослідний консорціум Auto-ID Center. З тих пір Інтернет речей звершив перехід від простих радіочастотних міток до екосистеми і індустрії. Аж до 2012 р ідея підключення речей до Інтернету переважно відносилася до смартфонів, планшетів, ПК і ноутбуків. По суті, до тих пристроїв, які в усіх відношеннях виступають в якості комп'ютера. До цього, з моменту появи перших боязких зачатків Інтернету (таких як створена в 1969 р мережу ARPANET), більшості технологій, на яких будується Інтернет речей, просто не існувало. До 2000 року більшість пристроїв, які можна було підключити до Інтернету, представляло собою комп'ютери різних розмірів. Нижче показаний поступове підключення речей до Інтернету.

1973 - Маріо У. Кардулло отримує патент на першу радіо-частотну мітку

1982 - Підключений до Інтернету автомат з газованою водою в університеті Карнегі-Меллон

1989 - Підключений до Інтернету тостер на конференції Interop '89

1991 - Компанія HP представила HP LaserJet IIISi: перший підключений до мережі Ethernet мережевий принтер

1993 - Підключена до Інтернету кавоварка в Кембриджському університеті (перша підключена до Інтернету камера)

1996 - Підрозділ General Motors OnStar (дистанційна діагностика 2001)

1998 - Поява організації Bluetooth SIG

1999 - Холодильник LG Internet Digital DIOS

2000 - Перші прояви розробленої компанією HP концепції всепроникної комп'ютеризації (Cooltown): HP Labs, система обчислювальних і комунікаційних технологій, які в поєднанні один з одним створюють підключення до Інтернету для людей, місць і об'єктів

2001 - Випуск першого пристрою, що використовує технологію Bluetooth: мобільний телефон KDDI з підтримкою Bluetooth

2005 - Міжнародний союз електрозв'язку, спеціалізована установа ООН, випустив звіт, в якому вперше були сформульовані прогнози розвитку Інтернету речей

2008 - Поява першого IoT-спільноти IPSO Alliance, метою якого було сприяння підключенню речей до Інтернету

2010 - Успішна розробка напівпровідникових світлодіодних ламп

привела до розвитку концепції розумного освітлення

2014 - Компанія Apple створила протокол iBeacon для маячків

4.2 Огляд архітектури Інтернет речей

Інтернет речей (IP, англ. Internet of Things, IoT) — концепція мережі, що складається із взаємозв'язаних фізичних пристроїв, які мають вбудовані давачі, а також програмне забезпечення, що дозволяє здійснювати передачу і обмін даними між фізичним світом і комп'ютерними системами в автоматичному режимі, за допомогою використання стандартних протоколів зв'язку. Окрім давачів, мережа може мати виконавчі пристрої, вбудовані у фізичні об'єкти і пов'язані між собою через дротові чи бездротові мережі. Ці взаємопов'язані пристрої мають можливість зчитування та приведення в дію, функцію програмування та ідентифікації, а також дозволяють виключити необхідність участі людини, за рахунок використання інтелектуальних інтерфейсів.

За даними Gartner, до 2025 року до Інтернету буде підключено більш 50 мільярдів пристроїв, і ці з'єднання полегшать використання даних для автономного аналізу, попереднього планування, управління та прийняття інтелектуальних рішень. Національна рада з розвідки США (NIC) включила IoT до шести «проривних цивільних технологій» (National Intelligence Council).

У контексті Інтернету речей на даний момент обслуговують декілька секторів, таких як (див. рис 4.1.):

Рисунок 4.1 - Інтернету речей в сферах діяльності людини

- транспорт, розумне місто,
- розумна побутова техніка,
- розумне здоров'я, електронне управління,
- допомога у проживання, електронна освіта,
- роздрібна торгівля, логістика,
- сільське господарство,
- автоматизація, промислове виробництво
- бізнес процеси. керування.

Архітектуру IoT можна розглядати як систему, яка може бути
- фізичною,
- віртуальною
- гібридною,

що складається з безлічі активних фізичних об'єктів, датчиків, приводів, хмарних сервісів, конкретних протоколів IoT, рівнів зв'язку, користувачів, розробників та рівень підприємства.

Архітектура Інтернету речей відрізняється в залежності від реалізації. Тим не менше вона дещо схожа на архітектуру класичних систем АСУТП. Один із прикладів архітектури показаний на рис. 4.2.

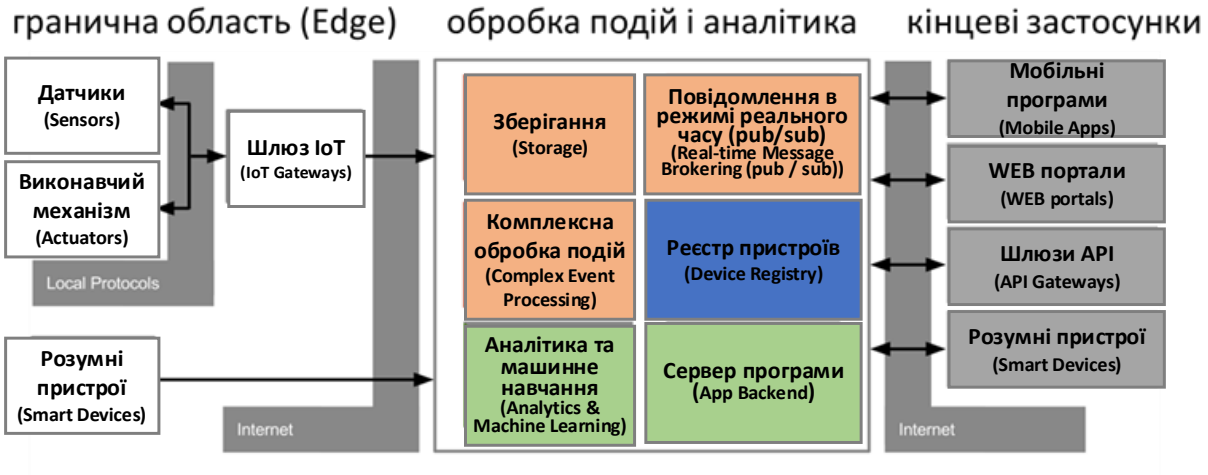


Рисунок 4.2 - Архітектура Інтернету Речей

Конкретні архітектури виступають як основний компонент інфраструктури IoT, сприяючи систематичному підходу до різноманітних компонентів, що призводить до вирішення пов'язаних проблем. Чітко визначена форма архітектури IoT в даний час доступна для інформаційних цілей: «динамічна глобальна мережева інфраструктура з можливостями самоконфігурації на основі стандартних та інтероперабельних протоколів зв'язку, де фізичні та віртуальні «речі» мають ідентифікатори, фізичні атрибути та віртуальні особистості, використовують інтегруються до інформаційної мережі».

Взаємодія з «речами» відбувається через датчики (sensors) та виконавчі механізми (Actuators), аналогічно як це робиться в АСУТП для

будь якого об’єкту керування. Ці датчики разом з усією інфраструктурою для інтеграції з рівнем обробки подій через мережу Internet формують так звану граничну область (Edge).

Події (дані) що поступають з граничної області зберігаються і обробляються відповідно до задачі (рівень обробки подій і аналітики, event processing, Platform). На цьому рівні події(дані) зберігаються (storage), обробляються (Event Processing), перенаправляються потрібним додаткам (Real-Time Message Brokering, Stream Processing). Додатково на цьому рівні відбувається адміністрування та керування пристроями з граничної області (Device Registry, Edge Device Management). Події (дані) обробляються з використанням аналітичних сервісів (Analytics) на основі них проводиться машинне навчання (Machine Learning), що дозволяє зробити певні висновки про об’єкт. Цей рівень як правило реалізований з використанням хмарних (Cloud) або туманних (Fog) обчислень. Якщо провести аналогію с АСУТП, то це рівень контролерів та SCADA (за виключенням функцій HMI). Отримання результатів, контроль, віддалене керування та адміністрування системи проводиться через кінцеві застосунки з використанням Internet. Цей рівень можна умовно порівняти з HMI в АСУТП.

На рис. 4.3. показана подібна наведеній вище архітектура, однак у вигляді сервісів.

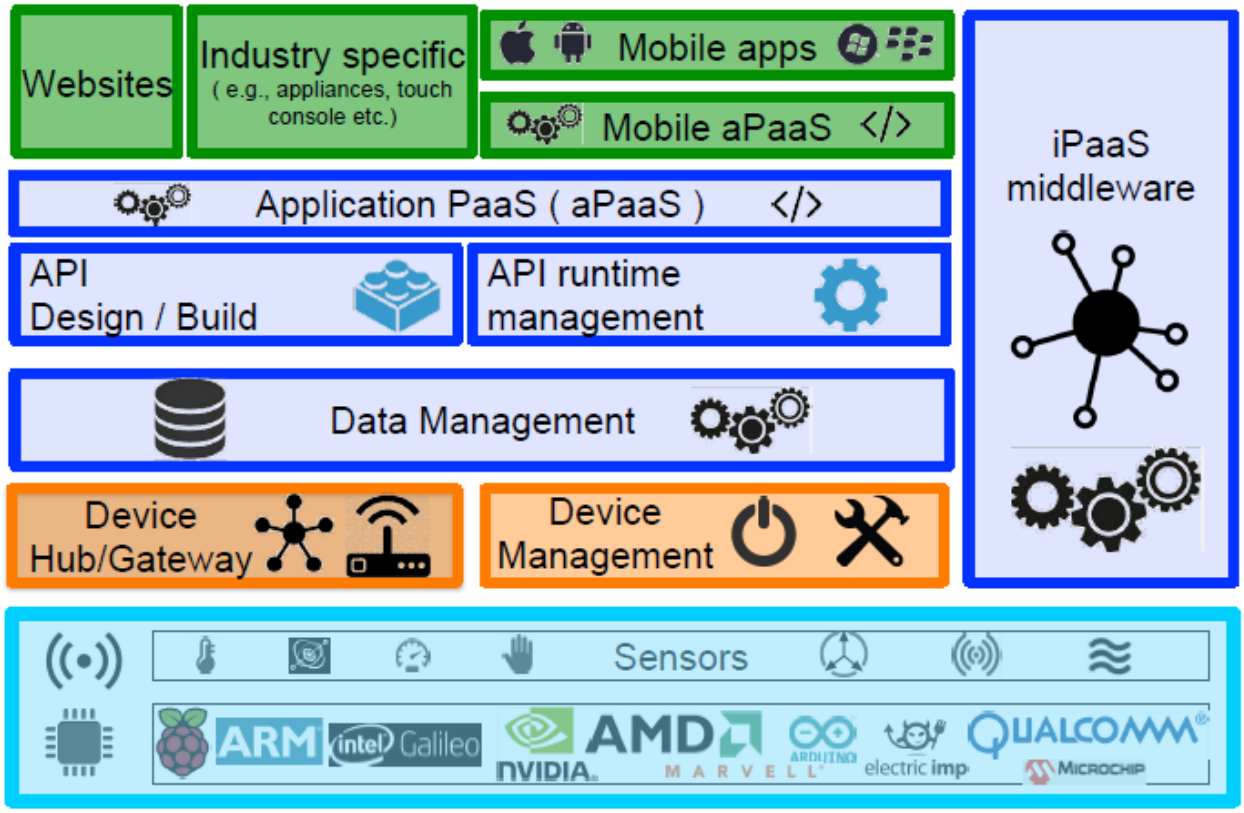


Рисунок 4.3 – Архітектура Інтернет речей у вигляді сервісів

Область Edge представлена у вигляді датчиків (Sensors), Device Hub/Gateway (збір та маршрутизація даних) та Device Management (керування пристроями). Останні частково виконуються як хмарні обчислення так і на граничних пристроях. Усі функції збереження та первинної обробки подій (даних) зведені до Data Management. Усі інші функції обробки, в тому числі аналітичні показані як додатки PaaS, що взаємодіють з сервісами керування даних через API (Application Program Interface).

В Інтернеті речей все можна поділити на три категорії:

- Датчики, які збирають інформацію, а потім надсилають її.
- Комп'ютери, які отримують інформацію, а потім діють відповідно до неї.
- Речі, які роблять і те, й інше.

Приклади Інтернет речей або IoT

Будь-який фізичний об'єкт можна перетворити на пристрій IoT, якщо його можна підключити до Інтернету та керувати ним через Інтернет.

- світильник, який можна увімкнути за допомогою програми для смартфона, є пристроєм IoT
- датчик руху
- інтелектуальний термостат в офісі
- підключений вуличний ліхтар.
- системи розумного будинку (охоронні пристрої, інтелектуальне освітлення, кондиціювання, опалення, підключена побутова техніка)
- медичні пристрої, що носяться як для самостійного відстеження стану здоров'я (пульсоксиметри, глюкометри), так і для моніторингу показників життєдіяльності в клініках
- системи стеження за логістикою (GPS-трекери, датчики рівня, витати палива, системи оповіщення для відстеження поведінки водія)
- автономний транспорт (сільгосптехніка, складські автономні роботи, пасажирські автобуси)
- розумне виробниче обладнання (робототехніка, рішення для профілактичного обслуговування)

Термін "Інтернет речей" в основному використовується для позначення пристроїв, які зазвичай не передбачають підключення до Інтернету, які можуть зв'язуватися з мережею незалежно від дій людини. З цієї причини ПК зазвичай не вважається пристроєм IoT, так само як і смартфон, хоча останній забитий датчиками. Однак, смарт-годинник або фітнес-браслет можуть вважатися пристроєм IoT.

4.3 Типи платформ Інтернету речей

Продукти Інтернету речей складаються з безлічі компонентів:

- Апаратне забезпечення

- Програмне забезпечення
- Комунікаційні технології
- Центральний репозиторій (хмарний чи локальний)
- Програми для кінцевих користувачів

Типи платформ IoT:

- Платформи підключення надають комунікаційні технології для з'єднання фізичних об'єктів із центром обробки даних (локального чи хмарного) та передачі інформації між ними. Серед популярних протоколів підключення та стандартів для Інтернету речей – MQTT, DDS, AMQP, Bluetooth, ZigBee, WiFi, Cellular, LoRaWAN та інші.

Тобто ця платформа виконує маршрутизацію для передачі даних від датчиків в Інтернет-простір. Для цього необхідні дві технології: *маршрутизатор-шлюз і опорні інтернет-протоколи*, що забезпечують ефективність обміну даними. Маршрутизатор особливо важливий в таких аспектах, як безпека, управління і напрям даних. Граничні маршрутизатори (Edge routers) керують і стежать за станом відповідних mesh-мереж, а також вирівнюють і підтримують якість даних. Також велике значення належить конфіденційності та безпеки даних. Маршрутизатор відіграє важливу роль в створенні віртуальних приватних мереж, віртуальних локальних мереж і програмно-визначених глобальних мереж. Вони в буквальному сенсі можуть містити тисячі вузлів, що обслуговуються єдиним граничним маршрутизатором, і в якійсь мірі маршрутизатор служить розширенням для хмари (edge device).

Рисунок 4.4 - Популярні протоколи підключення та стандартів для Інтернету речей

На цьому рівні використовується ряд протоколів, необхідних для обміну даними між вузлами, маршрутизаторами і хмарними сервісами в межах IoT-системи. Інтернет речей відкрив дорогу новим IoT-протоколам, які виходять на один рівень з традиційними протоколами HTTP і SNMP, які застосовуються вже кілька десятків років. Для передачі IoT-даних потрібні ефективні, енергозберігаючі протоколи з малою затримкою, здатні легко і безпечно відправляти дані в хмару і з нього. Зокрема тут використовуються такі протоколи, як усюдисущий MQTT, AMPQ і CoAP.

- Платформи аналітики використовують інтелектуальні алгоритми для аналізу зібраної інформації та перетворення її на корисні ідеї для клієнтів.

- Наскрізні платформи Інтернету речей охоплюють всі аспекти продуктів Інтернету речей, від розробки та підключення до управління даними та візуалізації.

- Платформи розробки устаткування надають фізичні плати розробки створення пристроїв IoT, включаючи мікроконтролери, мікропроцесори, системи на кристалі (SoC), системи на модулі (SoM).

- Платформи розробки додатків служать як інтегроване середовище розробки (IDE) з інструментами та функціями для програмування додатків.

Архітектура Інтернету речей складається з 5 рівнів (см. рис. 4.5):

Рисунок 4.5 – Архітектура рівнів IoT

Перший елемент системи: датчики, детектори, приводи.

Датчики: пристрої або системи, створені для розуміння та виявлення змін у їхньому середовищі, а також для подальшої оптимізації інформації у своїй системі. Датчики мають унікальну здатність визначати фізичні параметри, такі як вологість або температура, а потім перетворювати їх на електронні сигнали.

Приводи: вони є частиною машини, яка дозволяє перетворювати

електричний сигнал на фізичні дії. Ці актуатори грають вирішальну роль компонентів мереж IoT.

Другий елемент системи IoT: інтернет-шлюзи та системи збору даних. Система збору даних (DAS) збирає необроблені дані з датчиків і перетворює їх з аналогового на цифровий формат. Потім DAS збирає та форматує дані перед їх відправкою через шлюз Інтернету бездротовими глобальними мережами (Wi-Fi, стільниковий зв'язок) або провідними глобальними мережами для наступного етапу обробки.

1) Wi-Fi - найпопулярніший і універсальний метод, який використовується в технологіях, керованих даними.

2) Ethernet є обладнанням, яке підтримує фіксовані або постійні пристрої, такі як відеокамери, ігрові консолі та системи безпеки.

3) Bluetooth - технологія, що широко використовується, що підходить в основному для зв'язку між пристроями на невеликій відстані.

4) NFC (Near Field Communications) забезпечує зв'язок на дуже короткій відстані 50 см або менше.

5) LPWAN (глобальна мережа з низьким енергоспоживанням), спроектована та побудована з урахуванням використання Інтернету на великих відстанях. Ці малопотужні пристрої WAN можуть прослужити до 10 років, споживаючи при цьому низьке енергоспоживання.

6) ZigBee – це ще одна передова технологія бездротової мережі, яка споживає невелику кількість енергії та може запропонувати невелику можливість спільного використання даних.

7) *Стільникові мережі* ідеально підходять для зв'язку у глобальному масштабі з великою довірою та надійністю. Для IoT існує два широкі рівні IoT в стільникової мережі:

- LTE-M - це довгостроковий розвиток машин, що забезпечує дуже високошвидкісний обмін даними та безперебійний прямий хмарний зв'язок.

- NB-IoT як вузькосмуговий, який пропонує невеликий обмін даними з використанням низькочастотних каналів.

Також на цьому рівні є певні протоколи передачі даних:

- Служба розподілу даних (DDS) є платформою обміну повідомленнями між машинами в реальному часі в системах Інтернету речей.

- Advanced Message Queuing Protocol (AMQP) надає серверні протоколи для серверів через одноранговий обмін даними.

- Протокол обмеженої програми (CoAP) визначає протоколи для обмежених пристроїв, які використовують мале енергоспоживання та мало пам'яті, наприклад, бездротові датчики.

- Транспортна телеметрія черги повідомлень (MQTT) є стандартами протоколу обміну повідомленнями для пристроїв з низьким енергоспоживанням, що використовують TCP/IP для безперешкодної передачі даних.

Третій елемент системи IoT: попередня обробка та аналітика, після того, як дані Інтернету речей будуть оцифровані та агреговані, їх потрібно обробити, щоб ще більше зменшити обсяг даних, перш ніж вони потраплять до центру обробки даних або в хмару. Прикордонний пристрій може виконувати певну аналітику в рамках попередньої обробки. Машинне навчання може бути дуже корисним на цьому етапі для забезпечення зворотного зв'язку із системою та постійного покращення процесу, не чекаючи отримання інструкцій з корпоративного центру обробки даних чи хмари. Обробка цього типу зазвичай відбувається на пристрої в місці, близькому до того, де знаходяться датчики, наприклад, у комутаційній шафі на місці.

- Накопичення даних

Кожен пристрій надсилає мільйони потоків даних через мережу IoT. Тут дані бувають різних форм, швидкостей та розмірів. Відділення важливих даних від цих великих потоків — першочергове завдання, яке професіонали мають розставити за пріоритетами на цьому рівні.

- Абстракція даних

Після завершення етапу збору даних вибрані дані витягуються з великих даних для програми, щоб оптимізувати свої бізнес-процедури. Тут абстракція даних слідує наступним шляхом:

- Збір усіх даних із усіх систем IoT та не-IoT (CRM, ERP та ERM)
- Використання віртуалізації даних для доступу до даних з одного місця
- Управління необробленими даними у кількох формах

Взаємодія між пристроями та архітектурою відіграє вирішальну роль на рівні обробки.

Четвертий елемент системи IoT: аналіз у хмарі чи центрі обробки даних. На цьому етапі обробку даних починають потужні IT-системи для аналізу, управління та безпечного зберігання даних. Зазвичай це відбувається в корпоративному центрі обробки даних або у хмарі. Компанія може працювати у різних регіонах, і дані IoT можна аналізувати виявлення ключових тенденцій і закономірностей чи виявлення аномалій.

П'ятий елемент системи IoT: на цьому етапі оброблені дані надходять менеджерам системи (користувачам). З цих даних можна дізнатися тенденції, закономірності та аномалії.

Найпопулярніші платформи Інтернету речей у 2021 році:

Google Cloud IoT

Cisco IoT Cloud Connect

Salesforce IoT Cloud

В даний час у всьому світі існує понад 600 публічно відомих платформ Інтернету речей. Лідери - Amazon AWS IoT Core, Microsoft Azure IoT Hub та IBM Watson.

5 БЕЗДРОТОВІ СЕНСОРНІ МЕРЕЖИ

5.1 Основні поняття і принципи сенсорних мереж

Визначимо основні поняття сенсорних мереж.

Сенсор (англ., Sensor) - пристрій, який сприймає контрольований вплив (світло, тиск, температуру і т. п.), вимірює його кількісні та якісні характеристики і перетворює дані вимірювання в сигнал. Сигнал може бути електричний, хімічний або іншого типу.

Датчик (англ., Transducer) - пристрій, який використовується для перетворення одного виду енергії в інший. Отже, сенсор також є датчиком, який перетворює фізичну інформацію в електричну, яка може бути передана обчислювальній системі чи контролеру для обробки.

Актуатор (англ., Actuator) - виконавчий пристрій, що реагує на сигнал, який надійшов, для зміни стану керованого об'єкта. В актуаторі відбувається перетворення типів енергії, наприклад, електрична енергія, або енергія стисненого (розрідженого) повітря (рідини, твердого тіла) перетворюється в механічну.

Сенсорний вузол (англ., Sensor node) - це пристрій, який складається, принаймні, з одного сенсора (може також включати один або декількох актуаторів), і має обчислювальні та дротові або бездротові мережеві можливості.

Сенсорна мережа - система розподілених сенсорних вузлів, взаємодіючих між собою, а також з іншими мережами для запитів, обробки, передачі та надання інформації, отриманої від об'єктів реального фізичного світу з метою вироблення відповідної реакції на цю інформацію. Таким чином, сенсорна мережа включає в себе як мінімум сенсори, актуатори і комунікаційні вузли. Основною областю застосування сенсорної мережі є контроль і моніторинг реальних показників фізичних середовищ і об'єктів та в деяких випадках - управління цими об'єктами (активація в них певних процесів). Приклади сенсорних мереж: всепроникні сенсорні мережі (USN - Ubiquitous Sensor Network), мережі для транспортних засобів (VANET - Vehicular Ad Hoc Network), муніципальні мережі (HANET - Home Ad hoc Network), медичні мережі (MBAN (S) - Medicine Body Area Network (services)) і ін. Основні дії, що виконуються при роботі сенсорних мереж, представлені на рис. 5.1 (пунктиром показані необов'язкові процеси).

Область покриття сенсорної мережі може становити від декількох метрів до декількох кілометрів за рахунок здатності ретрансляції повідомлень від одного елемента мережі до іншого. Сенсорна мережу має здатність до ретрансляції повідомлень по ланцюжку від одного вузла до іншого, що дозволяє в разі виходу з ладу одного з вузлів організувати передачу інформації через сусідні вузли без втрати якості.

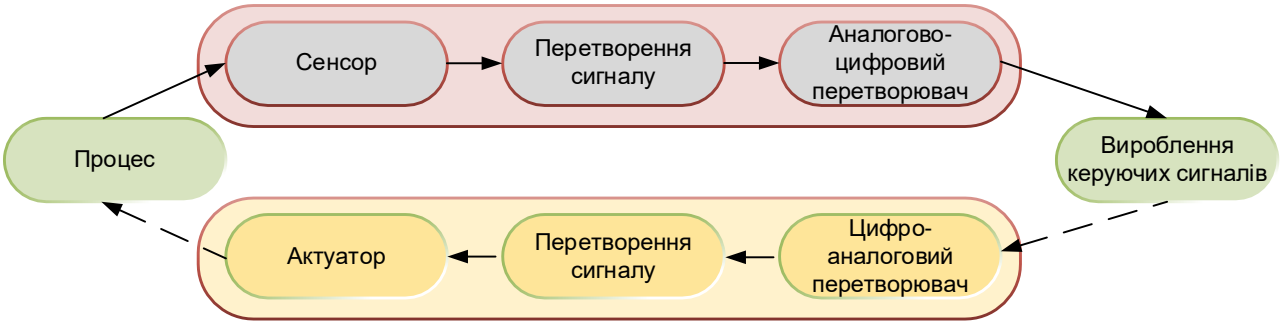


Рисунок 5.1 - Збір даних і управління в сенсорних мережах

Сама мережа визначає оптимальний маршрут руху інформаційних потоків (рис. 5.2).

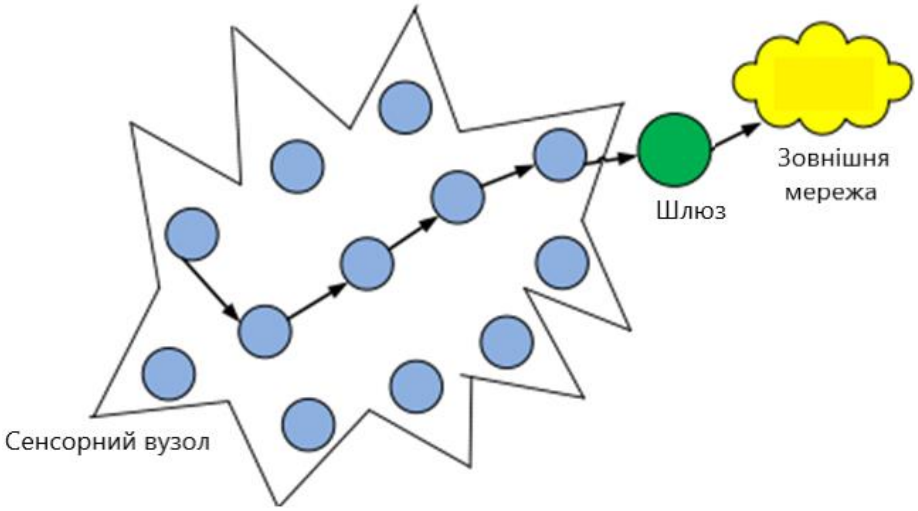


Рисунок 5.2 - Маршрутизація інформації в сенсорній мережі

Самоорганізована (лат. Ad hoc - «за місцем») мережа зв'язку - мережа, в якій число вузлів є випадковою величиною в часі і може змінюватися від 0 до деякого максимального значення. Взаємозв'язки між вузлами в такій мережі також випадкові у часі і утворюються для передачі інформації між подібними вузлами і в зовнішню мережу зв'язку.

Бездротова сенсорна мережа (БСМ) (англ. WSN - Wireless Sensor Network) – розподілена сенсорна мережа множини сенсорів і виконавчих пристроїв, що самоорганізується, об'єднаних між собою за допомогою радіоканалів.

Переваги бездротових сенсорних мереж:

здатність до самовідновлення та самоорганізації;

здатність передавати інформацію на значні відстані при малій потужності передавачів (шляхом ретрансляції);

низька вартість вузлів і їх малий розмір;

низьке енергоспоживання і можливість електроживлення від автономних джерел;

простота установки, відсутність необхідності в прокладанні

кабелів (завдяки бездротовій технології і живленню від батарей);

можливість установки таких мереж на вже існуючий і експлуатується об'єкт без проведення додаткових робіт;

низька вартість технічного обслуговування.

Так як на практиці в найбільшою мірою поширені бездротові сенсорні мережі, тому основна частина матеріалу глави присвячена саме таким мережам.

5.2 Класифікація технологій передачі даних у IoT

Одним із головних питань організації Інтернету речей є реалізація взаємодії між:

a) інтернет-мовами;

b) користувачами та інтернет-мовами;

c) віддаленим сервером та інтернет-мовами.

IoT використовує велику кількість варіантів мереж зв'язку для передачі даних, починаючи від мережі на тілі людини BAN (Body Area Network), яка працює на відстані кілька десятків сантиметрів, аж до всесвітньої мережі інтернет. Комунікації малої дальності використовують такі технології, як RFID, NFC, Bluetooth, Wi-Fi та ін. ін.

По території охоплення телекомунікаційні мережі, що використовуються в Інтернеті речей, можна поділити на 4 основні типи (див. рис. 5.3):

а) персональна мережа PAN (Personal Area Network) – це мережа, побудована «навколо» людини. Дані мережі мають об'єднувати всі персональні пристрої користувача (телефони, смартфони, кишенькові персональні комп'ютери, ноутбуки, гарнітури та ін.). Стосовно IoT така мережа будується «навколо» пристрою («речі»);

б) локальна мережа LAN (Local Area Network) - мережа, що зазвичай покриває відносно невелику територію або невелику групу будівель (будинок, офіс, фірму). До локальних мереж можна віднести і мережу контролерів CAN (Controller Area Network) - промислову мережу, орієнтовану насамперед на об'єднання в єдину мережу різних виконавчих пристроїв і датчиків у рамках окремого підприємства;

в) міська мережа MAN (Metropolitan Area Network) - об'єднує окремих користувачів та локальні мережі в межах міста, являє собою мережу за розмірами більшу, ніж LAN, але меншу, ніж WAN;

г) глобальна мережа WAN (Wide Area Network) - пов'язує користувачів та мережі, розосереджені з відривом сотень і тисяч кілометрів.

Інтернет речей практично не висуває особливих вимог до технологій LAN, MAN і WAN, крім того, вони досить добре освітлені в технічній літературі. Тому в цьому розділі розглянуті лише стандарти та протоколи мереж малого та середнього радіусу дії, які широко використовуються в IoT.

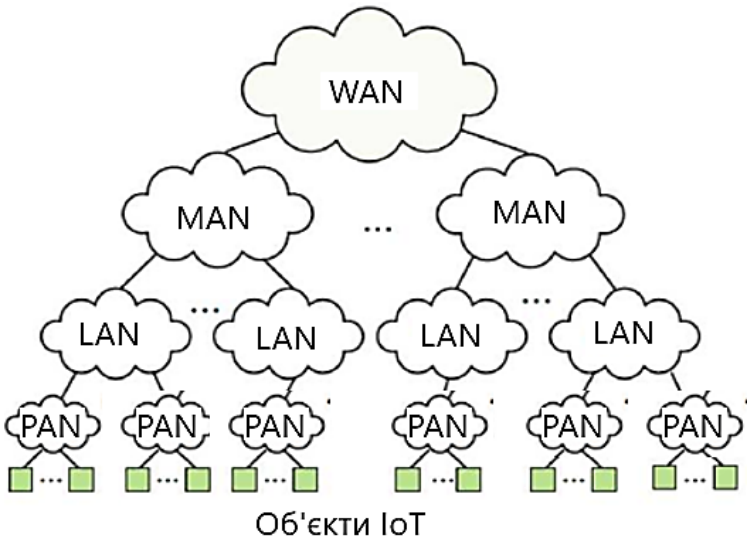


Рисунок 5.3 - Ієрархія мережевих технологій, що використовуються в IoT

Всі технології передачі даних в Інтернеті речей залежно від використовуваного середовища передачі можна розділити на два великі класи: дротові та бездротові.

Дротові технології передачі даних в IoT можуть використовувати металевий (мідний) кабель зв'язку, електропроводку (технологія PLC - Power Line Communication) або волоконно-оптичний кабель. Однак через складності фізичної реалізації ліній зв'язку провідні технології комунікацій інтернет-речей застосовуються меншою мірою, ніж бездротові.

Бездротові мережі малого радіусу дії, що використовуються в IoT, можна розділити на три види:

а) бездротові персональні мережі WPAN (Wireless Personal Area Network). Застосовуються для зв'язку різних пристроїв, включаючи комп'ютерну, побутову та оргтехніку, засоби зв'язку тощо. Фізичний та канальний рівні регламентуються стандартом IEEE 802.15.4. Радіус дії WPAN становить від кількох метрів до кількох десятків сантиметрів. Такі мережі використовуються як для об'єднання окремих пристроїв між собою, так і для зв'язку з мережами вищого рівня, наприклад, глобальною мережею інтернет. WPAN може бути розгорнута з використанням різноманітних мережевих технологій;

б) бездротові сенсорні мережі WSN (Wireless Sensor Network). Розподілені мережі, що самоорганізуються, безлічі датчиків (сенсорів) виконавчих пристроїв, об'єднаних між собою за допомогою радіоканалу. Область покриття подібних мереж може становити від кількох метрів до кількох кілометрів за рахунок можливості ретрансляції повідомлень від одного елемента до іншого;

с) малі локальні мережі TAN (Tiny Area Network). Обчислювальні мережі, що розгортаються в межах невеликого офісу чи окремого

житла. Їх часто називають домашніми мережами, оскільки вони об'єднують комп'ютери, побутову електроніку та сигналізаційні прилади, що належать одній родині. Найчастіше такі мережі будуються з урахуванням технології Wi-Fi.

Для взаємодії безлічі різноманітних пристроїв в IoT потрібні стандартизовані інтерфейси, формати даних та комунікаційні протоколи. У таблиці 5.1 наведено перелік деяких стандартів та протоколів IoT із зазначенням робочої частоти, швидкості передачі даних, підтримки рівнів OSI (фізичного PHY, доступу до середовища MAC, мережевого NWK, транспортного TRP), а також реалізації підрівня підтримки додатків APS (Application Support Sublayer), підтримки списків управління доступом ACЬ (Access Control List) та 128-бітного стандарту шифрування AES (Advanced EncryptionStandart).

Таблиця 5.1 – Стандарти та протоколи IoT

Стандарт	Частота, МГц	Швидкість, кбіт/с	Рівні протоколів						Шифрування
			PHY	MAC	NWK	TRP	APS	ACL	
IEEE 802.15.4	868/915/2400	20/40/250	+	+	-	-	-	+	+
Zigbee	2400	250	-	-	+	+	+	+	+
6L0WPAN	-	50...200	-	-	+	-	-	+	+
WirelessHART	2400	250	+	+	+	+	+	+	+
ISA 100.11a	2400	250	+	+	+	+	+	+	+
Z-Wave	865/915/869	9,6/40	+	+	+	-	+	-	-
Bluetooth	2400	1000	+	+	+	+	+	+	+

Примітка.
PHY - Фізичний рівень
MAC - Рівень керування доступом до мережі
NWK – Рівень мережевої комуникації
TPR – Транспортний рівень
APS – Рівень додатків
ACL – список керування доступом
http://surl.li/eabia – додаткова інформація

5.3 Типи вузлів БСМ

Типова архітектура БСМ включає три типи вузлів (див. рис. 5.4):

1. *Координатор* - здійснює глобальну координацію, організацію та установку параметрів мережі, є найбільш складним пристроєм БСМ, вимагає найбільший об'єм пам'яті і найбільшу потужність джерела живлення. В одній мережі повинен бути присутнім тільки один координатор. З координатора здійснюється вихід в зовнішню мережу (він реалізує функцію шлюзу - gateway). Часто координатор називають базовою станцією (БС).

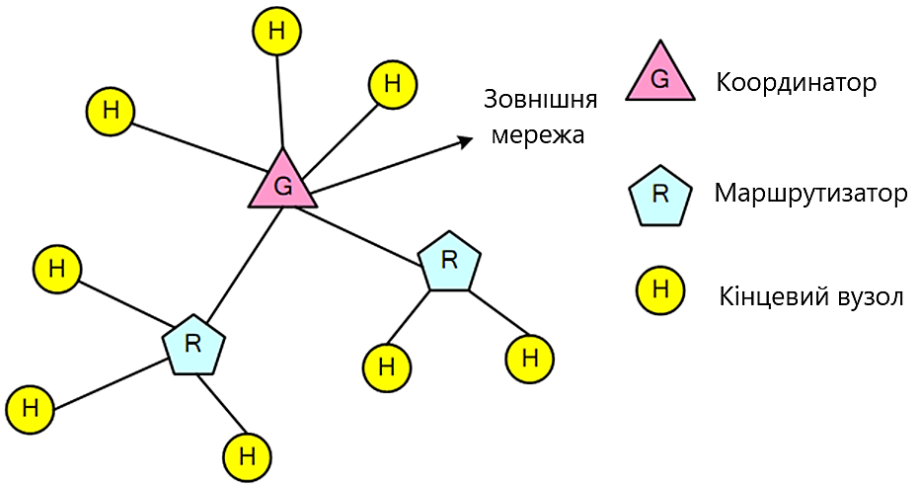


Рисунок 5.4 - Типи вузлів БСМ

Координатор виконує наступні функції:

- визначає незадіяні канали з переліку каналів, доступних для організації мережі і визначаються розробником і організовує мережу;
- передає мережеві сигнальні пакети з інформацією про існуючої мережі;
- управляє мережевими підлеглими пристроями, встановлює параметри мережі - визначає максимальну глибину вкладених підмереж, число мережевих маршрутизаторів і число підлеглих пристроїв;
- забезпечує маршрутизацію інформації між підлеглими пристроями;
- більшу частину часу перебуває в режимі прийому;
- забезпечує організацію таблиць маршрутизації;
- дозволяє маршрутизаторів і кінцевим пристроям входити в мережу.

2. *Маршрутизатор* - приймає, буферизує і передає дані від інших вузлів БСМ, а також визначає напрямок передачі.

Маршрутизатор виконує наступні функції:

- визначає активні канали, підключається до мережі і дозволяє кінцевим пристроям входити в мережу - використовує додаткові, визначені додатком, списки активних каналів;
- ретранслює сигнальні мережеві пакети з параметрами мережі від координатора;
- адмініструє мережеві адреси підключених до маршрутизатора підлеглих пристроїв;
- підтримує наступні класи пристроїв маршрутизації: пристрій з таблицею маршрутизації і з функцією деревовидної маршрутизації, пристрій тільки з функцією деревовидної маршрутизації, підтримка функції аварійної деревовидної маршрутизації;
- підтримує два режими роботи пристроїв: без переходу в «сплячий режим» і з переходом в «сплячий» режим в періоди, які

визначаються координатором мережі і параметрами мережевої синхронізації;

- підтримує функції маршрутизації багато чарункових мереж: створює таблиці сусідніх мережевих вузлів з параметром якості зв'язку з кожним з них, створює таблиці мережевої маршрутизації, ретранслює пакети запиту і підтвердження визначення маршрутів між пристроями;

- підтримує функції маршрутизації по деревовидному принципу – транслює повідомлення вгору і вниз по ієрархічній структурі дерева гілки в залежності від адреси одержувача повідомлення.

3. *Кінцевий пристрій* (сенсорний вузол) – виконує тільки прикладні дії (збір інформації та управління віддаленим об’єктом) і не здійснює ретрансляцію даних.

Сенсорний вузол має такі особливості:

- завжди шукає і намагається увійти в існуючу мережу - використовує додаткові, визначені додатком, списки активних каналів і сигнальні пакети синхронізації існуючої мережі для визначення параметрів мережі та маршрутизатора для входу в мережу;

- живиться від автономного джерела (батареї);

з пакетів синхронізації визначає наявність даних від координатора;

- запрошує дані від координатора;

- здатний знаходитися тривалий час в «сплячому» режимі (до 99,99% від всього часу роботи).

По виконуваним наборам функцій все вузли БСМ можна віднести до двох видів:

1. *Пристрій з повним набором функцій* FFD (Fully Function Device):

- підтримка стандарту IEEE 802.15.4;

- додаткова пам'ять і енергоспоживання дозволяють виконувати роль координатора мережі;

- підтримка всіх типів топологій («точка-точка», «зірка», «дерево», «чарункова мережа»);

- здатність виконувати роль координатора мережі;

- здатність звертатися до інших пристроїв в мережі.

2. *Пристрій з обмеженим набором функцій* RFD (Reduced Function Device):

- підтримує обмежений набір функцій стандарту IEEE 802.15.4;

- підтримка топології «точка-точка», «зірка»;

- не виконує функції координатора;

- звертається до координатора мережі і маршрутизатора.

Координатори та маршрутизатори завжди відносяться до пристроїв FFD, кінцеві пристрої можуть бути FFD або RFD.

6 ОСОБЛИВОСТІ БЕЗДРОТОВИХ СЕНСОРНИХ МЕРЕЖ

6.1 Типові архітектури та топології БСМ (WSN)

Виділяють два типи архітектури бездротових сенсорних мереж: однорідні (однорангові) і ієрархічні (кластерні). Однорідність мережі має на увазі, що всі вузли виконують однакові функції при зборі, обробці та передачі інформації. Цей підхід дозволяє домогтися оптимальної маршрутизації. Пересилання даних відбувається по найефективнішим за деякими критеріями маршрутами, що дозволяє домогтися економії таких важливих ресурсів, як енергія (передача іде по маршруту з найвищим запасом енергії) і час (передача відбувається по найкоротшому маршруту). Для критично важливих даних може бути організована передача по найбільш надійному шляху.

Агрегування даних, якщо необхідно, відбувається у міру проходження повідомлень до координатора. Однак при такій організації мережі формування зв'язків між вузлами відбувається спонтанно, що веде до зіткнень пакетів і виникнення затримок, пов'язаних з виходом із сплячого режиму вузлів, що знаходяться на обраному шляху передачі.

Альтернативним підходом є ієрархічна (деревоподібна) маршрутизація. Вона заснована на поділі мережі на області, які називаються кластерами. кластер утворюють маршрутизатор і кінцеві вузли, у яких він запитує сенсорні дані (див. рис. 6.1).

Усередині кожного кластера маршрутизатор відповідає за збір інформації з усього кластера, її обробку і подальшу передачу. Решта вузли кластера здійснюють тільки збір даних і передачу їх маршрутизатора. Таким чином, вузли в ієрархічній мережі не рівноправні. По-перше, агрегування даних відбувається на маршрутизаторах, і, по-друге, пересилання агрегованих даних далі може проводитися тільки маршрутизаторами. Таким чином, мінімізуються затримки передачі, оскільки маршрутизатори доступні завжди. Зіткнення пакетів виключені завдяки централізованому методу створення посилань. Однак така маршрутизація не надає оптимальних шляхів передачі даних. До того ж сенсорний вузол, що виконує функції маршрутизатора, витрачає значно більше енергії, що призводить до швидкого виснаження його батарей. Існують архітектури, які передбачають використання в якості маршрутизаторів фізично виділених сенсорів, що володіють великими запасами енергії і обчислювальними потужностями, однак цей підхід застосовується лише для вузького ряду додатків. Маршрутизатор кластерів ретранслюють дані один одному і, в кінцевому рахунку, дані передаються координаторові. Координатор зазвичай має зв'язок з IP-мережею, куди і прямують дані для остаточної обробки. У кожній мережі повинно бути, щонайменше, одне повнофункціональний пристрій FFD для роботи в якості координатора.

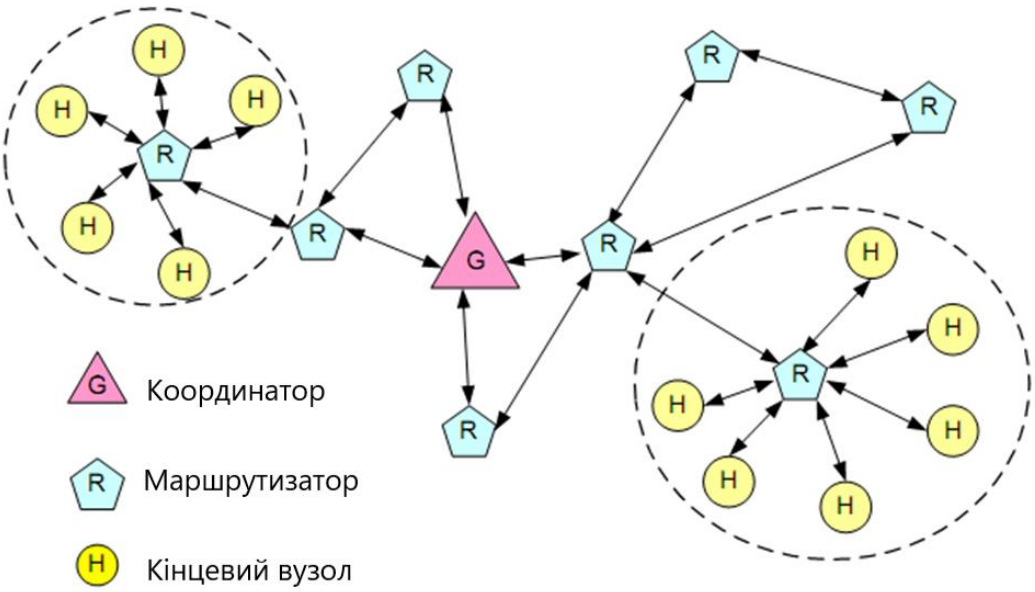


Рисунок 6.1 - Кластерна структура БСМ

Можливо також побудова тимчасових *ніздрюватих мереж* (див. рис. 6.2). У таких мережах функціональні можливості кожного сенсорного вузла однакові. Можливість самоорганізації і самовідновлення мереж комірчастої топології дозволяє в разі виходу частини сенсорів з ладу спонтанно формувати нову структуру мережі. Правда, в будь-якому випадку потрібен центральний функціональний вузол-координатор, який приймає і обробляє всі дані, або шлюз для передачі даних на обробку зовнішньому вузлу.

Спонтанно створювані мережі часто називають латинським терміном Ad Hoc, що означає «для конкретного випадку».

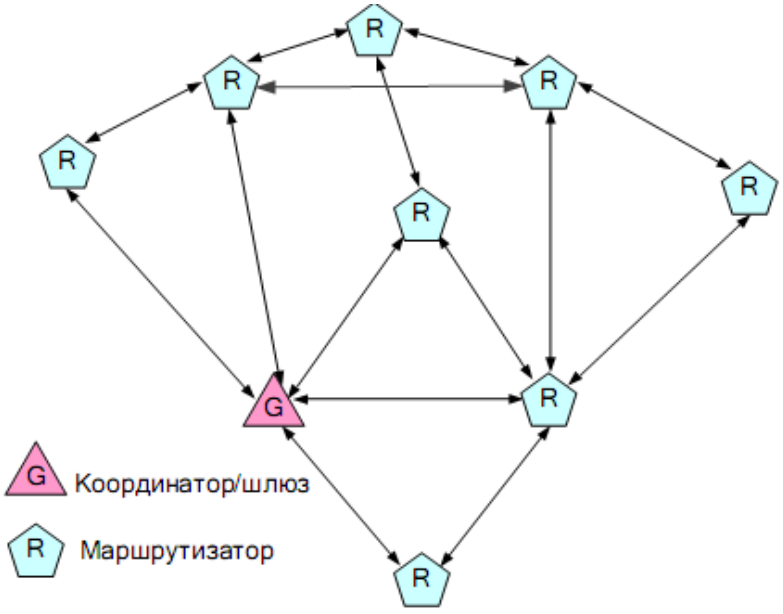


Рисунок 6.2 - Ніздрювата структура БСМ

Можливі топології сенсорної мережі наведені на рис. 6.3. Однорангові мережі можуть формувати довільні топологічні структури (точка-точка, зірка), обмежені тільки дистанцією між кожною парою вузлів. Mesh-мережі (Mesh Topology) - базова повнозв’язна топологія, в якій кожен маршрутизатор мережі з'єднується з декількома іншими маршрутизаторами цієї ж мережі. Характеризується високою стійкістю до відмов, але і більш складною настроюванням.

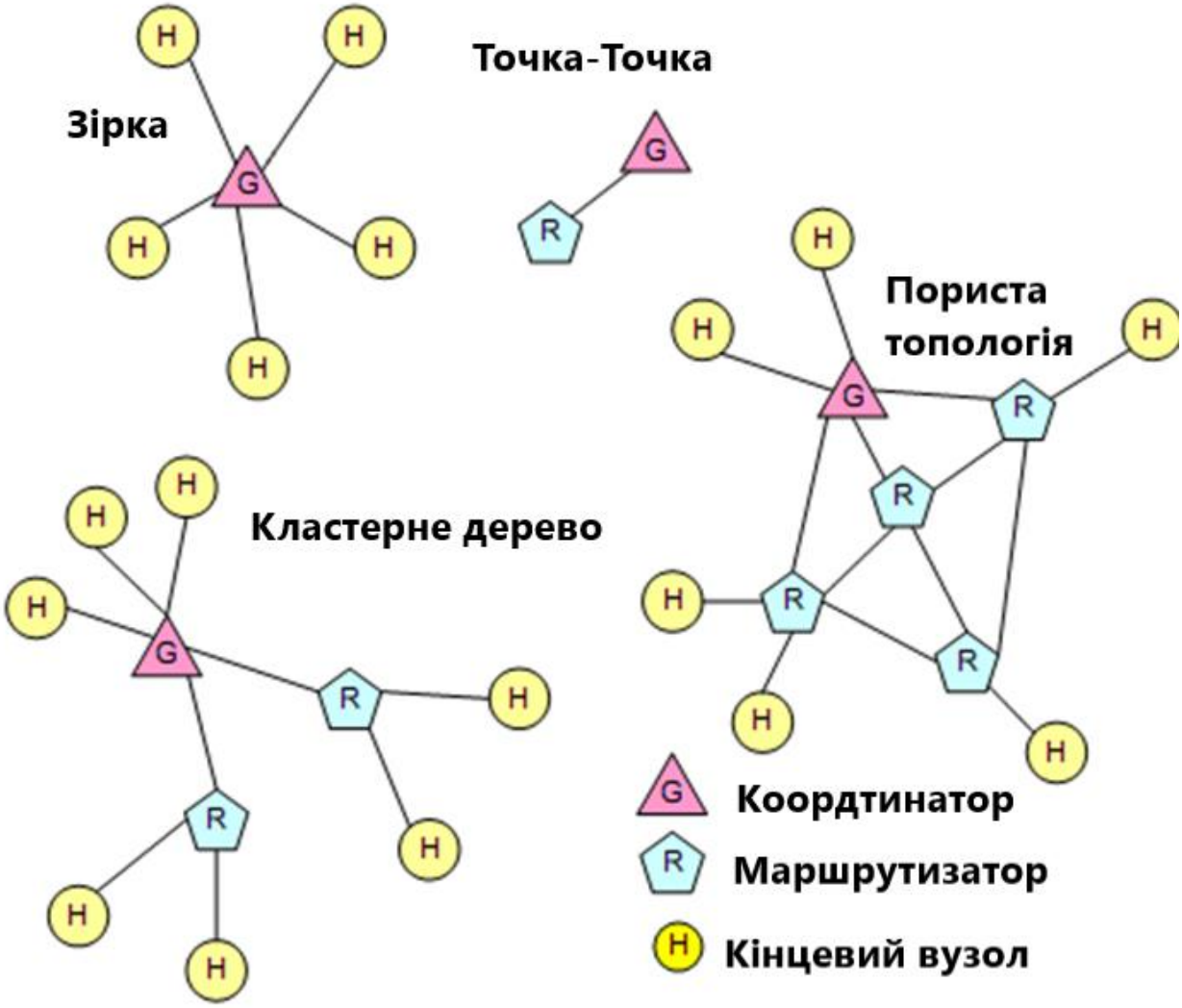


Рисунок 6.3 - Можливі топології сенсорної мережі

Прикладом тимчасової або пирингової мережі (від англ. Peer-to-peer, P2P - рівний до рівного) є кластерне дерево. Мережа типу кластерне дерево є приватним випадком мережі P2P, в якій більшість пристроїв є FFD. Пристрої RFD підключаються до кластеру в якості кінцевих вузлів. Для приєднання до мережі віддалених від координатора нових мережевих пристроїв можуть використовуватися вже під’єднані до мережі FFD в режимі координатора. В цьому режимі вони, як і спочатку координатор PAN, «зазивають» маяками в мережу нові мережеві пристрої. В результаті формується кластер з мережевих

пристроїв, які «чують» свого координатора. Проте, вся інформація про кластер доступна координатору PAN. Подібним чином можуть формуватися мультікластери з мережевих пристроїв.

6.2 Режими роботи БСМ

Самою енерговитратній операцією для сенсорних вузлів є передача даних в бездротове оточення. Тому енергозберігаючі форми передачі є ключовим фактором для продовження терміну служби сенсорів, так як він практично цілком залежить від терміну служби батареї.

Збір даних бездротової сенсорної мережею може здійснюватися різними способами в залежності від цільового призначення конкретної мережі. Приймаючи до уваги різні способи використання мережевих ресурсів, бездротові сенсорні мережі можна розділити на класи в залежності від виду їх функціонування і типу цільового додатки:

1. Проактивні мережі. Вузли такої мережі періодично включають свої сенсори і передавачі, знімають показання і передають їх на базову станцію. Таким чином, вони роблять "моментальну фотографію" свого оточення з певною періодичністю і використовуються зазвичай для додатків, що вимагають регулярного моніторингу деяких значень.

2. Реактивні мережі. Вузли реактивних мереж з певною періодичністю знімають показання, однак залишають поза передачею їх, якщо отримані дані потрапляють в певну область нормальних показань. У той же час відомості про несподівані і різкі зміни в показаннях датчиків або їх виході за діапазон нормальних значень негайно передаються на базову станцію. Цей вид мережі призначений для роботи з додатками реального часу.

3. Гібридні мережі. Це комбінація двох перерахованих вище типів, де сенсорні вузли не тільки періодично відправляють зняті дані, але і реагують на різкі зміни в значеннях.

6.3 Протоколи маршрутизації в БСМ

Для визначення маршруту передачі інформації в БСМ від кінцевого вузла до вузла-координатора, а також між кінцевими вузлами, використовуються спеціальні протоколи маршрутизації. Протоколи маршрутизації в БСМ вирішують наступні завдання:

1. Самоорганізація вузлів мережі (самоконфігурування, самовідновлення та оптимізація).
2. Маршрутизація пакетів даних і адресація вузлів.
3. Мінімізація енергоспоживання вузлів мережі і збільшення загального часу життя всієї мережі.
4. Збір і агрегація даних.
5. Регулювання швидкості передачі і обробки даних в мережі.

6. Максимізація зони покриття мережі.

7. Забезпечення заданої якості обслуговування (QoS).

8. Захист від несанкціонованого доступу.

При виборі шляху передачі інформації в мережі в якості метрик в них можуть бути використані наступні параметри:

- довжина шляху (кількість ділянок переприйому інформації);
- надійність;
- затримка;
- пропускна здатність;
- завантаження;
- вартість передачі трафіку і ін.

Протоколи маршрутизації БСМ відповідають за підтримку маршрутів в мережі і повинні гарантувати надійний зв'язок навіть в жорстких несприятливих умовах. Багато протоколів маршрутизації, управління електроживленням, поширення даних, були спеціально розроблені для БСМ, де енергозбереження є суттєвою проблемою, на вирішення якої спрямовано протокол. Інші ж були розроблені для загального застосування в бездротових мережах, але знайшли своє застосування і в БСМ.

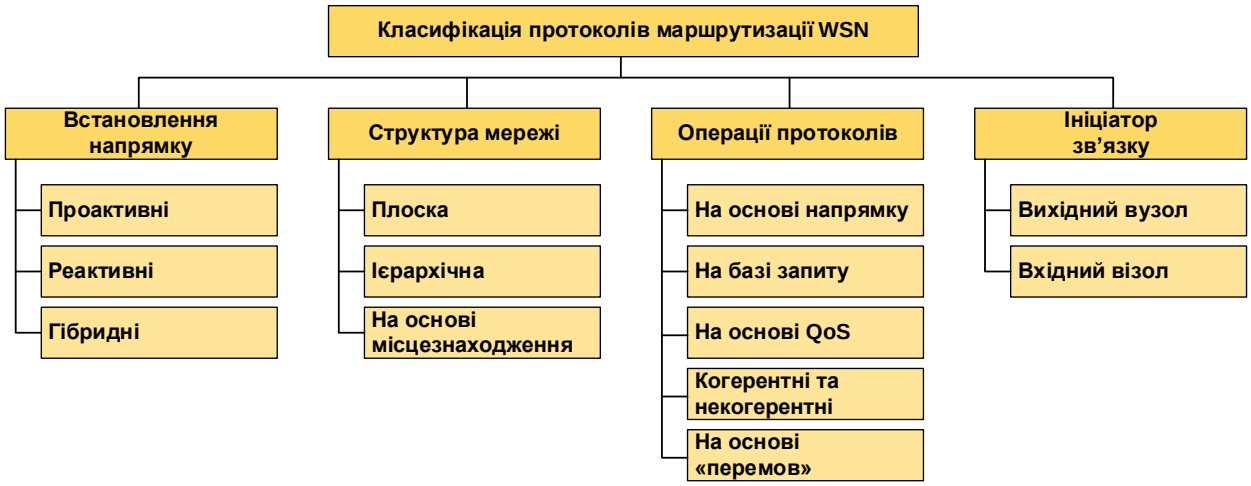


Рисунок 6.4 - Класифікація протоколів маршрутизації БСМ

Існує велика кількість протоколів маршрутизації для БСМ, класифікувати їх можна за різними ознаками (рис. 6.4). Залежно від використовуваного режиму роботи мережі, що обумовлює необхідність передачі інформації від вузлів, всі протоколи маршрутизації можна розділити на проактивні (всі шляхи визначаються заздалегідь, до того як вони будуть потрібні), реактивні (шляху визначаються на вимогу) та гібридні (комбінація перших двох).

Протоколи, що враховують структури мережі, діляться на:

1) протоколи однорівневої (плоскої) (flat-based) маршрутизації - всі вузли БСМ мають однакову функціональність, приклади: SPIN (Sensor

Protocols for Information via Negotiation), Direct Diffusion, Rumor Routing;

2) протоколи ієрархічної (hierarchical-based) маршрутизації - вузли мережі виконують різні функції, вони можуть бути і фізично різними, приклади: LEACH (Low-Energy Adaptive Clustering Hierarchy), PEGASIS (Power-Efficient GAthering in Sensor Information Systems), TEEN і APTEEN (Threshold-sensitive Energy Efficient Protocols), SOP (Self-Organization Protocol);

3) протоколи маршрутизації на основі інформація про місцезнаходження вузла (location-based), приклади протоколів: GAF (Geographic Adaptive Fidelity), GEAR (Geographic and Energy Aware Routing).

Робота протоколу маршрутизації може ґрунтуватися на різних принципах:

1) протоколи маршрутизації з багатьма маршрутами (multipath routing) - використовуються кілька маршрутів від джерела до точки призначення, що підвищує надійність з'єднання, але збільшує накладні витрати і енерговитрати;

2) протоколи маршрутизації «на замовлення» (query-based) - вузол посилає запит на дані в мережу і інший вузол, який має запитувані дані, відповідає на запит;

3) протоколи маршрутизації, засновані на «переговорах» (negotiation routing) між вузлами;

4) протоколи, які враховують якість обслуговування (QoS-based), що дозволяє забезпечити певний рівень послуг в мережі.

У протоколах, спрямованих на агрегацію даних, проміжні вузли, що розташовуються між джерелами інформації та базовою станцією (БС), можуть здійснювати агрегацію даних і посилати БС вже зведені дані. Цей процес дозволяє сенсорним вузлів економити енергію.

Всі протоколи маршрутизації також можна розділити на два види - в одних ініціатором з'єднання є джерело інформації, а в інших - одержувач.

Класифікація протоколів маршрутизації БСМ на основі типів вузлів показана на рис. 6.5.

В останні роки активно впроваджуються бездротові децентралізовані, що самоорганізуються мережі, що складаються з мобільних пристроїв MANET (Mobile Ad hoc NETwork). Кожен пристрій такої мережі може незалежно пересуватися в будь-яких напрямках, і, як наслідок, часто розривати і встановлювати з'єднання с сусідами.

Мережі, що самоорганізуються, MANET мають наступні переваги над бездротовими мережами традиційної архітектури:

- можливість передачі даних на великі відстані без збільшення потужності передавача;

- стійкість до змін в інфраструктурі мережі;

- можливість швидкої реконфігурації в умовах несприятливої завадової обстановки;

- простота і висока швидкість розгортання мережі.

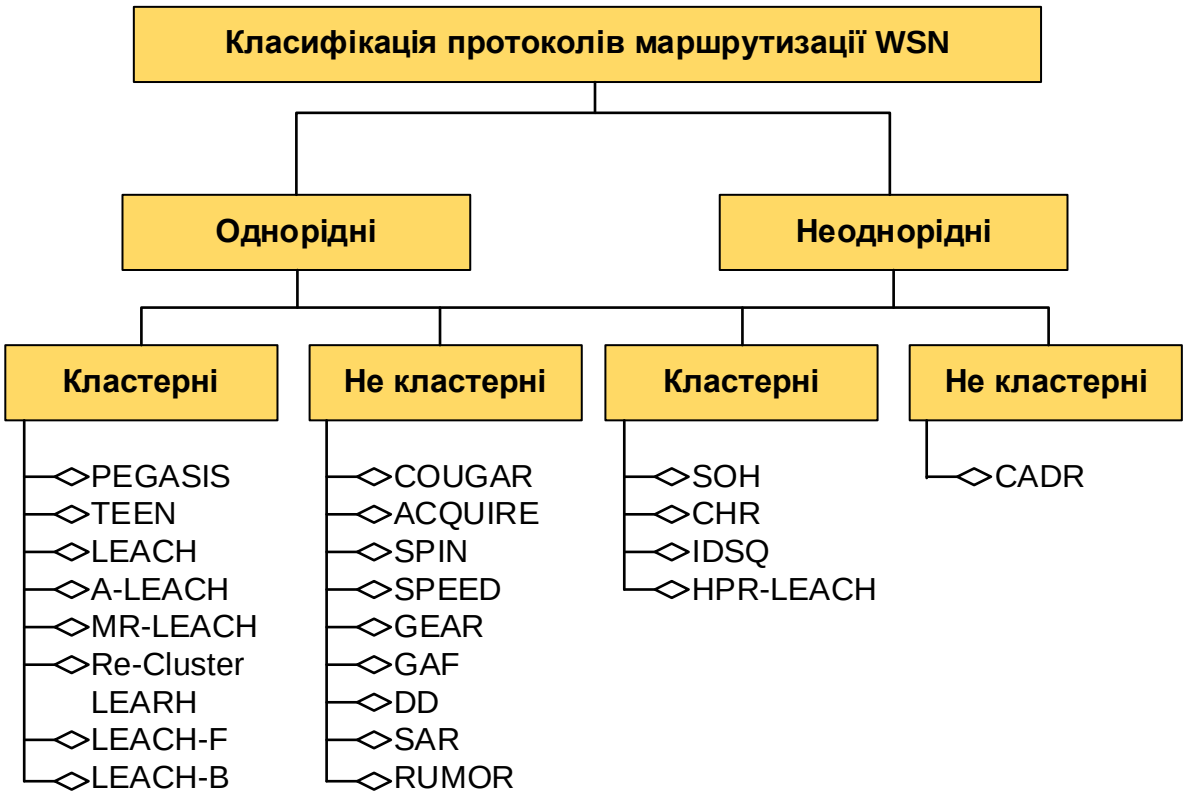


Рисунок 6.5 - Класифікація протоколів маршрутизації БСМ

Однак мобільність вузлів веде до додаткового підвищення динамічності топології мережі і, отже, до можливості обриву зв'язку через перешкоди або включення/вимикання вузла додається ймовірність його переміщення.

Для маршрутизації на мережевому рівні в MANET використовуються спеціальні протоколи, орієнтовані на динамічні мережі (наприклад, підтримувати маршрут, якщо поїхав проміжний вузол, і маршрут зруйнувався):

1) реактивні - знаходять маршрут в тому випадку, коли потрібно передати пакет і для нього немає відомого шляху і намагаються змінити цей шлях, якщо сталася помилка, приклади:

- спеціалізований протокол вектора відстані за запитом AODV (Ad hoc On-Demand Distance Vector),

- протокол динамічної маршрутизації джерела DSR (Dynamic Source Routing) і ін.;

2) проактивні (превентивні) - знаходять маршрут заздалегідь для всіх можливих пар джерело-приймач і періодично оновлюють інформацію про маршрутизації для підтримки шляхів, приклади: протокол оптимізованої маршрутизації стану з'єднання OLSR (Optimized Link-State Routing) і ін.

Перевагу одному або іншому виду протоколів може бути віддано

тільки з урахуванням обстановки і швидкостей руху абонентів. Наприклад, для автомобільної версії MANET має сенс використовувати реактивні протоколи.

Мережі MANET включають Ad hoc мережі для транспортних засобів VANET (Vehicular Ad hoc Network), в яких кожен автомобіль, що бере участь перетворюючись в бездротової маршрутизатор або вузол, що дозволяє автомобілям підключатися один до одного на відстані і створювати мобільну мережу. Стандарт для мереж VANET розробляється в рамках робочої групи IEEE 802.11p. Технічні засоби стандарту IEEE 802.11p повинні функціонувати на швидкості до 200 км/год і на відстані до 1 км. Фізичний рівень і MAC підрівень базуються на стандарті IEEE 802.11a. Частотний діапазон для США включає спектр від 5,859 до 5,925 ГГц, для Європи рекомендується використання двох піддіапазонів шириною по 10 МГц кожен: 5,865 - 5,875 ГГц і 5,885 - 5,895 ГГц.

Можливості по взаємодії транспортних засобів між собою і з мережею зв'язку загального користування в найближчі роки можуть привести до утворення нового, дуже масштабного сегмента Інтернету речей. Уже зараз сучасний автомобіль інтегрує в себе GPS / GLONASS приймач, різні сенсори, бортовий комп'ютер. Однак завдання, яке ставиться при створенні VANET, дещо інша. Архітектура мережі VANET передбачає взаємодію автомобіля, як з іншими автомобілями, так і з придорожньої мережею. При цьому виділяється три групи послуг:

1. Забезпечення безпеки - допомога водію (навігація, запобігання зіткнень і зміна смуг), інформування (про обмеження швидкості або про зону ремонтних робіт), попередження (післяаварійні, про перешкоди або стані дороги).

2. Підвищення ефективності управління автомобільним трафіком - скорочення тривалості поїздки, споживання палива.

3. Підвищення рівня комфорту пасажирів і водіїв - інформація про місцезнаходження автомобіля, про поточний трафіку на дорогах, про погоду, можливість здійснення P2P з'єднань, в тому числі з власним будинком через придорожню мережу, а також інформація від придорожньої мережі про готелі, станціях заправки, меню в ресторанах і так далі.

6.4 Сполучення БСМ з мережами загального користування

В даний час для сполучення БСМ з мережами зв'язку загального користування (ССОП) зазвичай використовується протокол бездротових персональних мереж на базі мережевого протоколу IPv6 з низьким енергоспоживанням 6LoWPAN (IPv6 over Low power Wireless Personal Area Networks), запропонований IETF, який дозволяє інтегрувати сенсорні мережі в існуюче сімейство мереж стека протоколів TCP / IP. Даний протокол дозволяє передавати IP-пакети

поверх стандарту IEEE 802.15.4 способом, що задовольняє відкритим стандартам (протокол IPv6). При цьому забезпечується взаємодія з іншими IP-каналами і пристроями. Протокол 6LoWPAN створений для малопотужних бездротових персональних мереж (LoWPANs) і описаний в документах RFC4919 і RFC4944. В архітектурі мережі 6LoWPAN (рис. 6) визначені три типи логічних пристроїв (крайовий вузол, маршрутизатор і шлюз), а також три види мереж: «Проста LoWPAN», «Розширена LoWPAN» і «Ad hoc LoWPAN». Як видно з малюнка, «Ad hoc LoWPAN» не підключена до ССОП, «Проста LoWPAN» підключена до ССОП через один шлюз, а «Розширена LoWPAN» включає в себе кілька шлюзів, пов'язаних з ССОП і один з одним за допомогою магістральної лінії зв'язку.

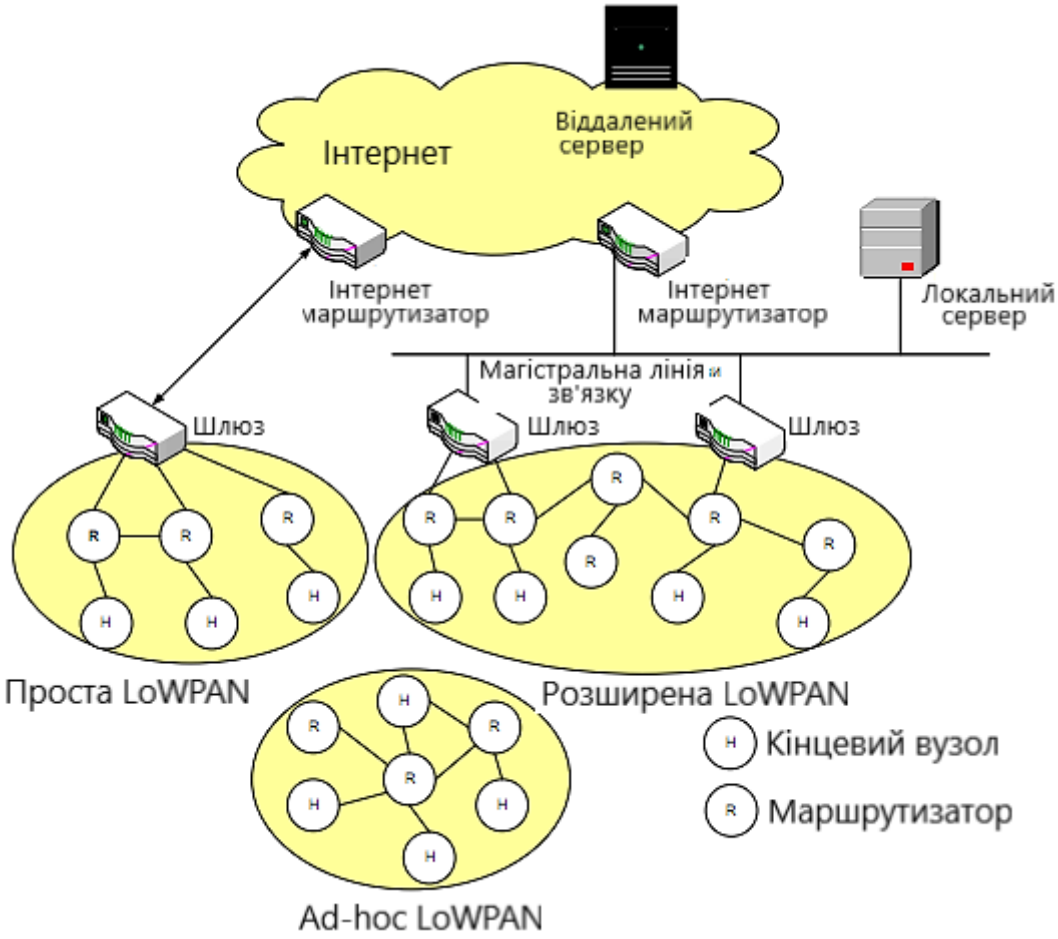


Рисунок 6.6 - Архітектура мережі 6LoWPAN

6.5 Проблеми реалізації БСМ

При практичній реалізації бездротових сенсорних мереж існує ряд проблем:

1. Проблема енергоспоживання.

Обмеження по енергоспоживанню пов'язаний з тим, що сенсори

працюють від джерела живлення з обмеженим лімітом енергії (зазвичай батарейка). Чим рідше вони будуть замінюватися або заряджатися, тим нижчу вартість буде мати їх обслуговування. Також енергоспоживання є важливим обмеженням при використанні сенсорів, доступ до яких ускладнений, отже, джерело живлення не може бути замінений або заряджаючи. Для зменшення енергоспоживання зазвичай передбачається відключення передавачів сенсорних вузлів, коли немає необхідності передачі інформації.

На мережевому рівні використовуються оптимальні шляхи передачі інформації від сенсорного вузла до координатора (базової станції), з огляду на число проміжних вузлів, необхідну енергію і доступну енергію. Крім мережевого протоколу на споживання енергії впливає конструкція вузлів (наприклад, маленький розмір пам'яті, ефективність перемикань між завданнями), програмне забезпечення, механізми захисту і навіть робочі додатки.

2. Проблема самоврядування.

Сенсорні мережі часто повинні працювати у віддалених областях і в жорстких умовах, без можливості їх обслуговування і ремонту. Тому, сенсорні вузли повинні конфігуруватися самостійно, взаємодіяти з іншими вузлами, адаптуватися до поломок змін навколишнього середовища без втручання людини.

3. Проблема бездротового з'єднання.

Вибір бездротового з'єднання накладає ряд обмежень на реалізацію сенсорних мереж. Наприклад, загасання сигналу обмежує відстань передачі інформації. Так зв'язок між потужностями сигналів переданої і прийнятої інформацією описується законом зворотного квадрата відстані:

$$P_{\text{пр}} \sim P_{\text{прд}} / D^2,$$

де $P_{пр}$ - потужність прийнятого сигналу, Вт; $P_{прд}$ - потужність переданого сигналу, Вт; D - відстань між передавачем і приймачем, м.

Отже, збільшення відстані між сенсорним вузлом і маршрутизатором/координатором призводить до збільшення потужності сигналу, що передається.

Тому більш ефективно, з точки зору витрат енергії, розділити великі відстані передачі інформації в сенсорних мережах на кілька невеликих.

4. Проблема децентралізованого управління.

Алгоритми побудови багатьох сенсорних мереж будуються з централізованого принципом. При децентралізованому управлінні сенсорні вузли повинні обмінюватися інформацією з сусідніми вузлами, щоб згенерувати рішенні про комутації вузлів, без глобальної інформації про всю мережі. Внаслідок цього децентралізовані

алгоритми можуть бути неоптимальними, але більш ефективними щодо енергії, ніж централізовані. Наприклад, при централізованому управлінні базова станція може опитувати всі сенсорні вузли, приймати від них інформацію, повідомляти кожному вузлу свій маршрут передачі інформації. При частій зміні мережі втрати будуть значні.

Децентралізований підхід дозволяє кожному вузлу робити власне рішення при наявності невеликої інформації (список сусідніх пристроїв, що включає інформацію про відстані до базової станції). В даному випадку втрати на управління будуть значно зменшені.

5. Проблема конструкції.

Головною метою бездротових сенсорних мереж є створення маленьких, дешевих і ефективних пристроїв. Через вимоги до низького споживання енергії типовий сенсорний вузол має невеликі швидкості виконання операцій і обсяги інформації, що зберігається. Також через це небажано використання деяких пристроїв, таких як GPS-приймачі. Обмеження за розмірами впливає на структуру протоколів і алгоритмів, реалізованих в бездротових сенсорних мережах. Наприклад, таблиця всіх маршрутів в мережі може бути дуже великий і не помістится в пам'яті вузла. Тому тільки невелика частина інформації (наприклад, список сусідніх вузлів) може зберігатися в пам'яті вузла.

6. Проблема безпеки.

Віддалене розташування сенсорів і їх автоматична робота збільшує їх незахищеність до стороннім вторгненням і атакам. При бездротовому з'єднанні досить легко для порушника перехопити пакети, що передаються сенсорним вузлом. Наприклад, найбільш велика загроза здійснення атаки «відмови в обслуговуванні» (denial-of-service), мета даної атаки порушити коректне функціонування сенсорної мережі. Це може бути досягнуто за допомогою різних способів, наприклад, при подачі потужного сигналу, який заважає сенсорним вузлів обмінюватися інформацією («білий шум» або jamming attack). Є різні варіанти захисту систем від зловмисників, але для багатьох з них необхідні високі вимоги до апаратних ресурсів, що важкодосяжно на жорстко обмежених по багатьом вимогам сенсорних вузлах.

Отже, сенсорні бездротові мережі вимагають нових рішень для створення ключів, їх поширення, ідентифікації та захисту вузлів.

6.6 Електроживлення вузлів БСМ від зовнішнього середовища

Одним з основних вимог, що пред'являються до вузлів сенсорної мережі, є тривалий час їх автономної роботи. Завдання зменшення енергоспоживання може вирішуватися за рахунок оптимізації конструкції і режимів роботи аналогових і цифрових схем вузлів, а також за рахунок вилучення енергії, необхідної для роботи цих схем, з довкілля. В даний час в усьому світі ведеться активний пошук нових

екологічних і необмежених ресурсів енергії, які дозволять мережевим пристроям позбутися батареї або дротів і розробити автономні бездротові сенсорні мережі з теоретично необмеженим терміном служби.

В навколишньому середовищу існують чотири основних джерела енергії: механічна енергія (вібрації, деформації), теплова енергія (температурні перепади або зміни), енергія випромінювання (сонце, інфрачервоні промені, радіочастоти) і хімічна енергія (хімія, біохімія). Ці джерела характеризуються різною щільністю потужності (див. рис. 6.7).

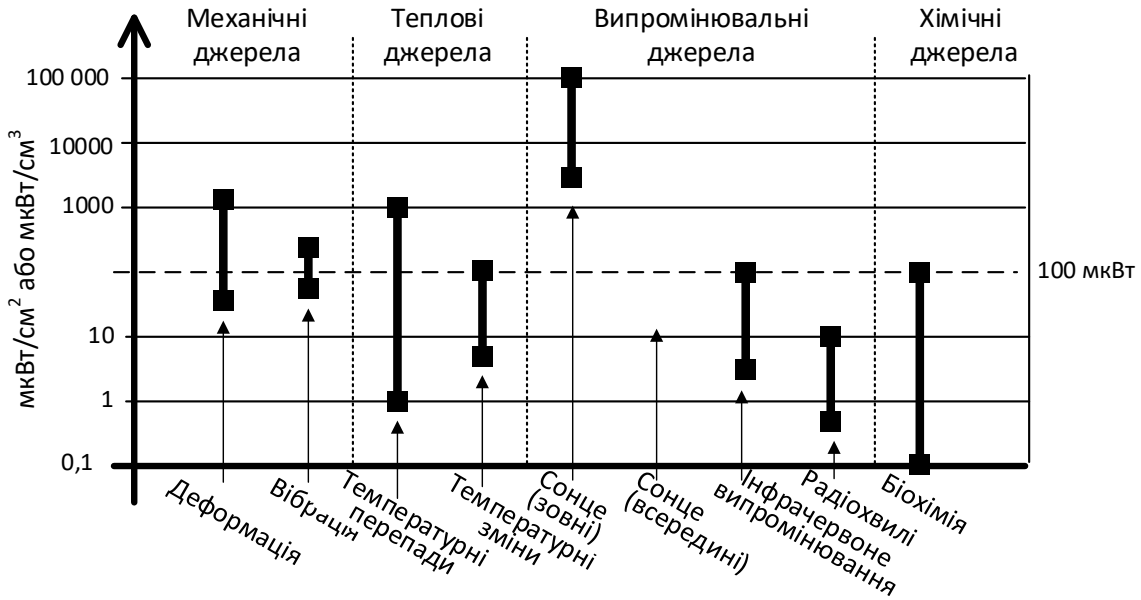


Рисунок 6.7 - Щільність потужності (до перетворення) для різних типів джерел енергії із зовнішнього середовища

Рисунок показує, що значення вихідної потужності 10-100 мкВт є прийнятним при розмірах джерела в 1 см^2 або 1 см^3. Отримання енергії від сонця вважається найбільш потужним (навіть якщо значення, наведені на рис. 6.7, повинні бути помножені на вагові коефіцієнти для перекладу ККД, як правило, перевищують 20% в фотоелементах). На жаль, отримання сонячної енергії неможливо в темних ділянках (наприклад, в приміщеннях). Аналогічно неможливо отримувати енергію від температурних перепадів, якщо цих перепадів немає або від неіснуючих вібрацій. Як наслідок, джерело зовнішньої енергії повинен бути обраний відповідно до місцевого середовища, навколишнього вузла бездротової сенсорної мережі, тобто не існує універсального джерела енергії із зовнішнього середовища.

Для живлення вузлів сенсорної мережі від навколишнього енергії необхідно знизити споживання енергії датчиками (сенсорами/актуаторами), мікроконтролером і радіо-передавачем. В останні роки значний прогрес в цьому напрямку було досягнуто

виробниками мікроконтролерів і радіочастотних чипів (Atmel, Microchip, Texas Instruments і ін.) Як для робочого, так і для холостого режиму. Приклад типового споживання енергії вузлом бездротових сенсорних мереж наведено на рис. 6.8.

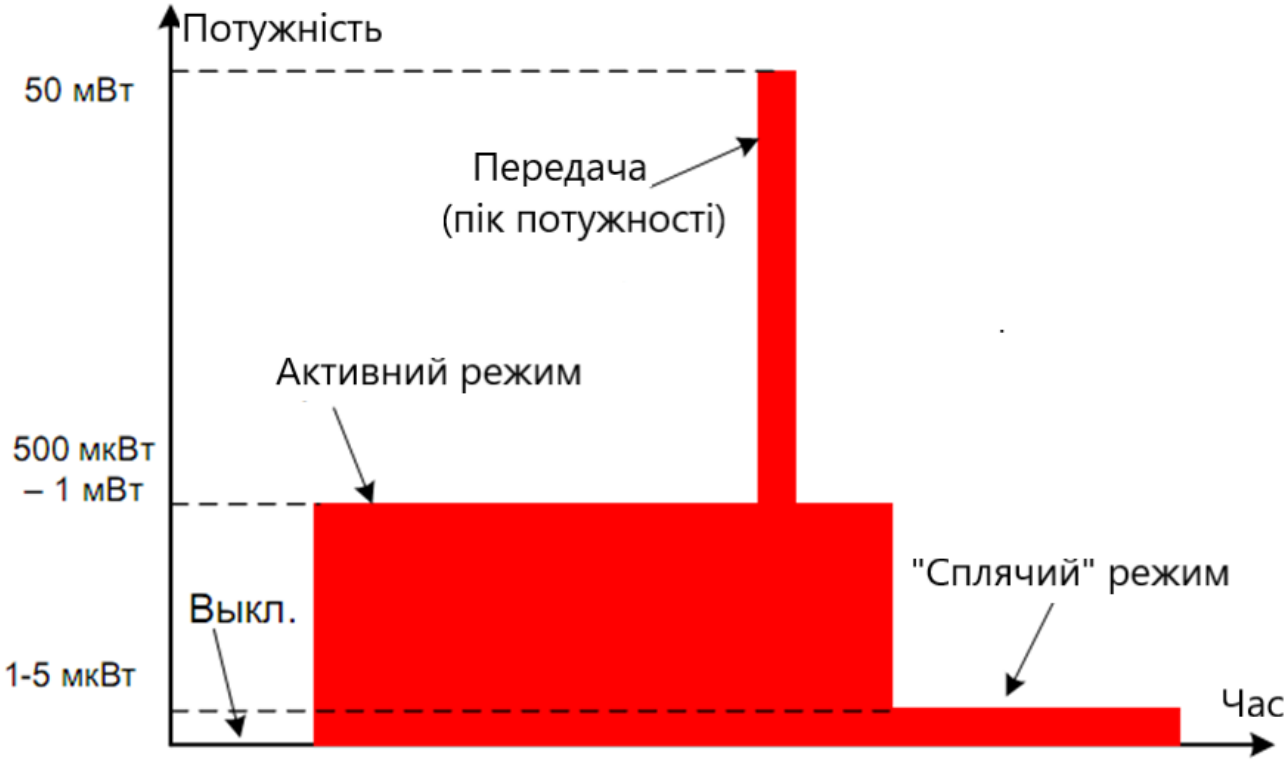


Рисунок 6.8 - Графік споживання енергії вузлом БСМ

Можна виділити три типових значення споживаної потужності:

1) 1-5 мкВт: споживання енергії в «сплячому» режимі;

2) 500 мкВт - 1 мВт: споживання енергії в активному режимі;

3) 50 мВт: пік передачі енергії.

Аналіз наведеної діаграми дозволяє зробити наступні висновки. По-перше, мінімальна потужність джерела енергії із зовнішнього середовища для побудови життєздатних бездротових вузлів повинна бути порядку 1-5 мкВт, що відповідає достатній величині для холостого режиму мікропроцесора і радіочастотного чіпа.

По-друге, сучасні джерела енергії із зовнішнього середовища не можуть забезпечувати бездротові сенсорні мережі енергією, достатньою для активного режиму (споживання енергії в 500 мкВт - 1 мВт проти 10-100 мкВт для вихідної потужності таких джерел). Однак, завдяки ультранизьким споживання енергії в сплячому режимі, бездротові сенсорні мережі, що живляться від зовнішнього середовища, можуть використовувати переривчастий робочий цикл, зображений на рис. 6.9. Енергія зберігається в буфері (а) (конденсатори, батареї) і використовується для виконання вимірювального циклу, як тільки енергії в буфері стає досить (б і в). Далі система знову повертається до

сплячого режиму (г), чекаючи нового вимірювального циклу.

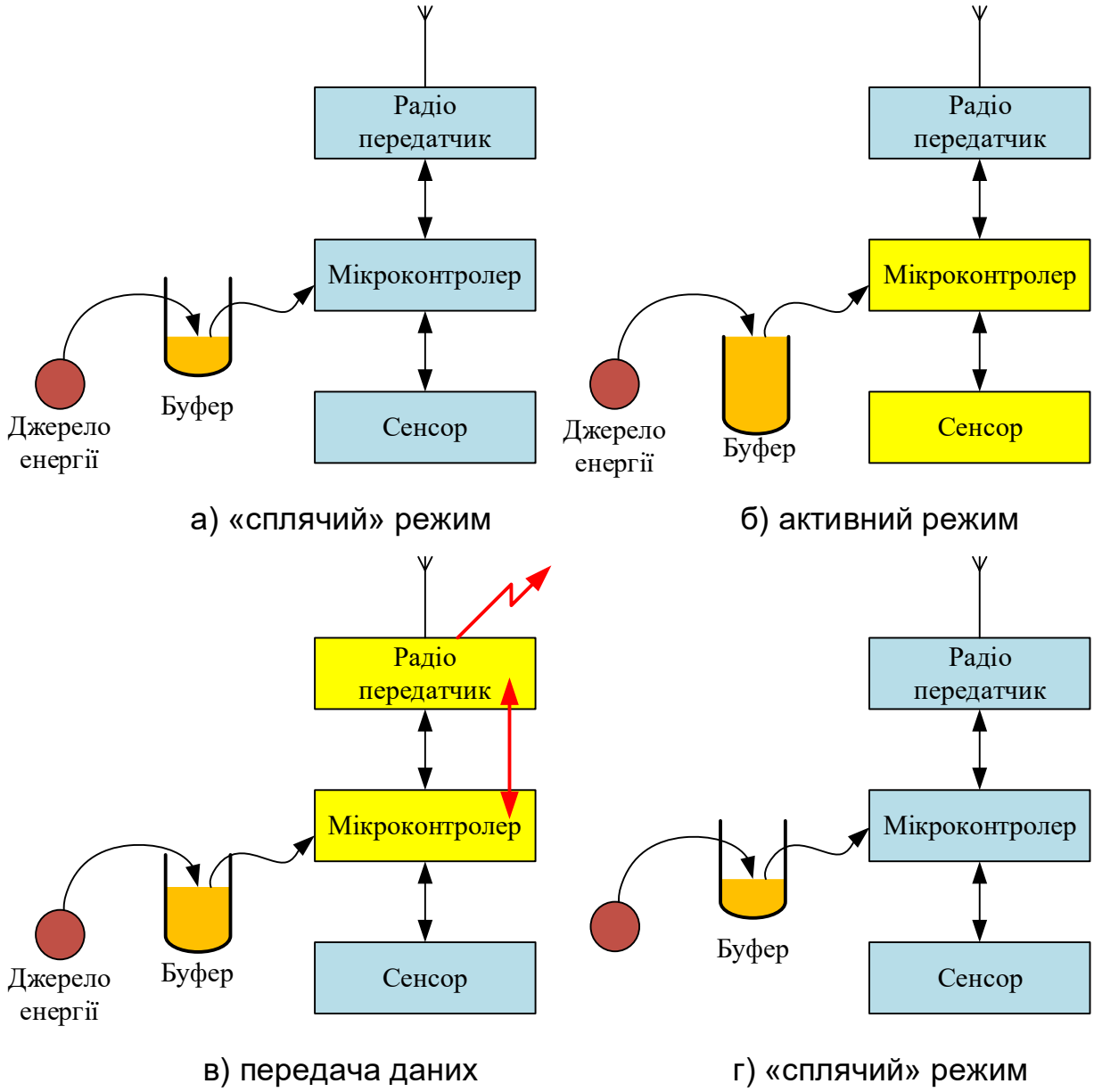


Рисунок 6.9 - Робочий цикл в бездротової сенсорної мережі

Таким чином, використовуючи енергію із зовнішнього середовища можливо живлення будь-яких додатків, навіть самих неощадливих. Основною проблемою є адаптація частоти вимірювального циклу до безперервно вироблюваної енергії. Середнє енергоспоживання сенсорних вузлів (*P*) відповідає загальній кількості енергії, необхідної для одного вимірювального циклу (W), помноженому на частоту цієї дії (*f*):

$$P = W f.$$

Зв'язок між *P*, *W* та *f* проілюстрована на рис. 6.10. Використовуючи

логарифмічні масштаби по осі абсцис (енергія в Джоулях) і по осі ординат (частота вимірювань), середнє енергоспоживання 100 мкВт показано прямою лінією з коефіцієнтом нахилу -1. Наприклад, виконання повного циклу роботи сенсорного вузла (вимір + перетворення + передача) вимагає 250-500 мкДж. Отже, безперервно отримуючи 100 мкВт потужності, можна виконувати повний цикл роботи вузла сенсорної мережі кожні 1-10 секунд (0,1-1 Гц). Це підходить багатьом промисловим потребам, особливо тим, де обслуговування передбачувано.

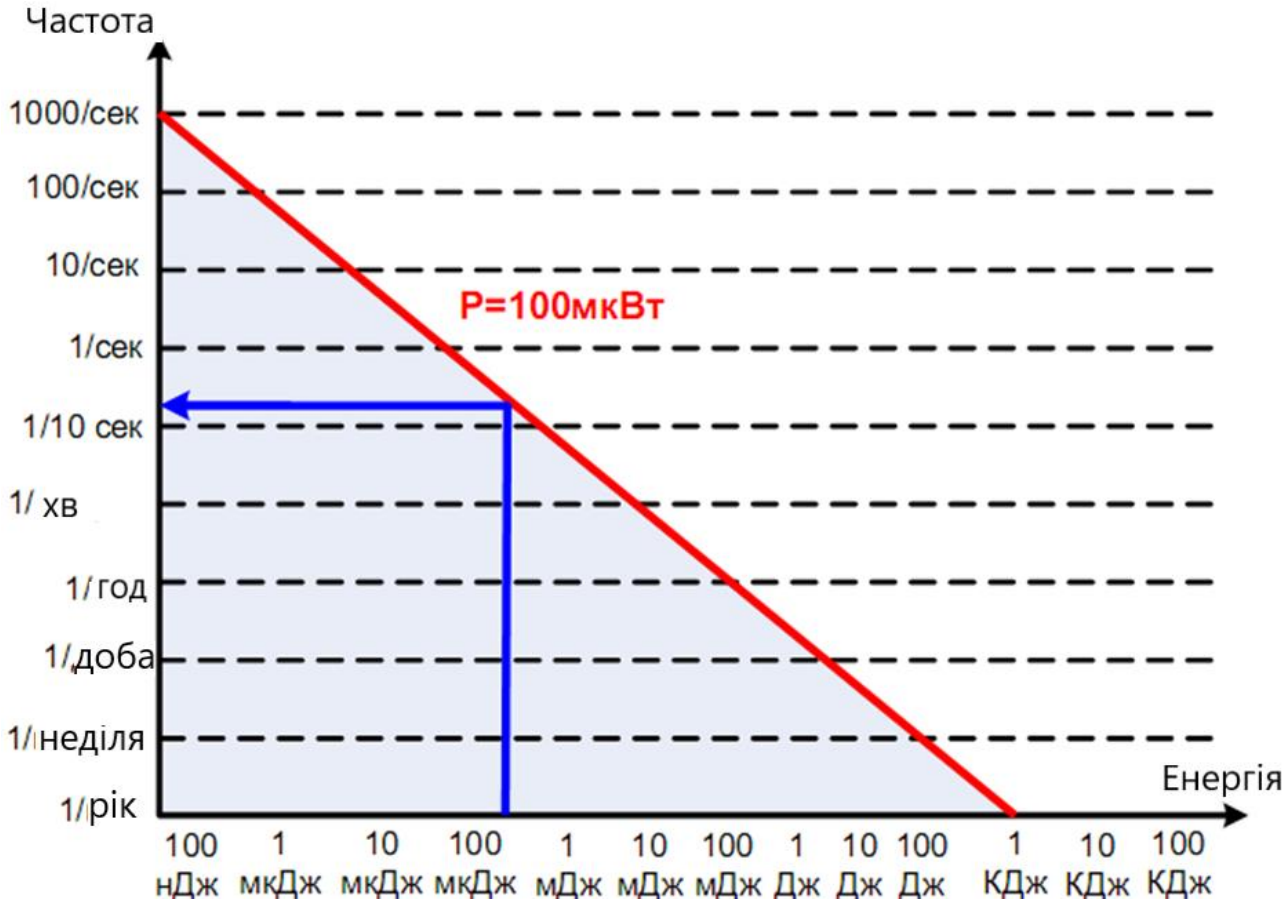


Рисунок 6.10 - Зв'язок потужності, енергії та частоти

В цілому отримання енергії із зовнішнього середовища - за винятком фотоелемента - тільки розвивається галузь, ще не пристосована для промислового застосування. Проте, поліпшення існуючих технологій може привести в перспективі до життєздатним рішенням електроживлення автономних бездротових сенсорних мереж.

6.7 БСМ та Інтернет речей

Завдяки таким характеристикам БСМ, як мініатюрність вузлів, низьке енергоспоживання, вбудований радиоинтерфейс, достатня обчислювальна потужність, порівняно невисока вартість, стало

можливим їх широке використання в багатьох сферах людської діяльності з метою автоматизації процесів збору інформації, моніторингу та контролю характеристик різноманітних технічних і природних об'єктів.

БСМ доцільно застосовувати в наступних предметних областях Інтернету речей:

- моніторинг телекомунікаційної інфраструктури мереж;
- моніторинг транспортних магістралей (залізниць, метрополітену та ін.),

нафто- і газопроводів, інженерних мереж енерго- і теплопостачання;

- контроль і аналіз транспортних вантажопотоків;
- екологічний, біологічний і медичний моніторинг;
- автоматизація систем життєзабезпечення в системах класу Розумний дім;
- виявлення і попередження надзвичайних ситуацій (моніторинг сейсмічної активності і вулканічної діяльності, аналіз атмосфери і прогноз погоди для своєчасного попередження про настання стихійних лих) та інші.

7 СТАНДАРТИ БЕЗДРОТОВИХ СЕНСОРНИХ МЕРЕЖ

7.1 Стандарт IEEE 802.15.4

Стандарт IEEE Std 802.15.4 призначений для реалізації бездротових персональних мереж WPAN великої ємності з низьким енергоспоживанням та низькою швидкістю передачі даних. Він реалізує лише два нижні рівні стека протоколів - фізичний рівень (PHY) та рівень доступу до середовища (MAC). Стандарт 802.15.14 є базовою основою більш високорівневих протоколів, таких як ZigBee, WirelessHART і NWi. Він може бути також використаний спільно зі стандартом 6LoWPAN та стандартними протоколами Інтернету для побудови бездротових сенсорних мереж. Стек протоколів для стандарту зображено рис. 7.1.

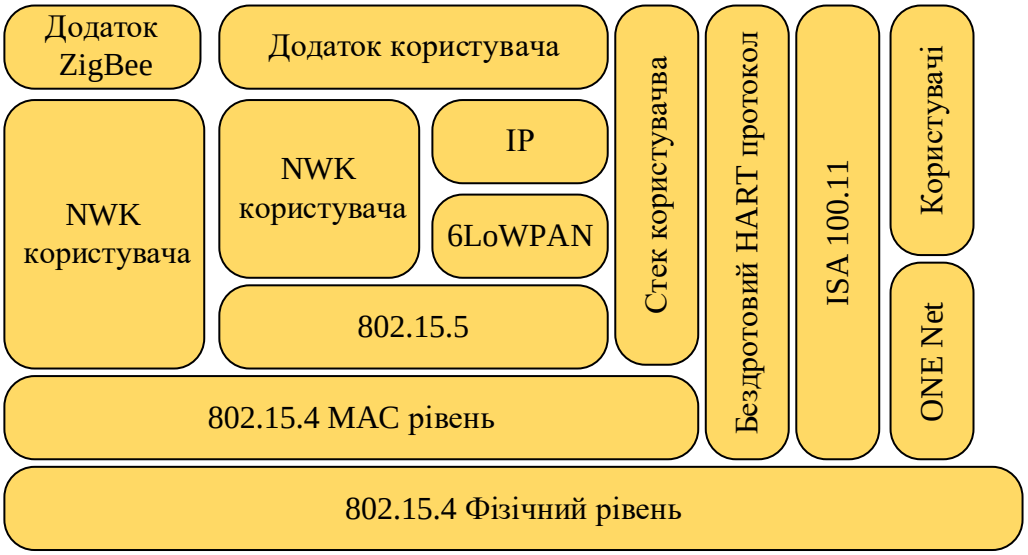


Рисунок 7.1 – Стек протоколів для стандарту IEEE Std 802.15.4

Фізичний рівень 802.15.14 PHY визначає спосіб передачі даних, інтерфейс організації зв'язку, апаратні особливості та параметри, необхідні для побудови мережі. Насправді фізичний рівень управляє роботою трансивера, виконує вибір каналів, сигналів управління та рівня потужності передачі.

На канальному рівні специфікація IEEE 802.15.4 визначає механізми взаємодії елементів мережі фізично для забезпечення формування фрагментів даних (кадрів), перевірки та виправлення помилок, відправки кадрів на мережевий рівень. При цьому підрівень MAC канального рівня регулює множинний доступ до фізичного середовища з поділом за часом, керує зв'язками трансіверів та забезпечує безпеку.

Стандарт визначає два типи вузлів мережі:

а) повнофункціональний пристрій FFD (Fully Function Device), який може реалізувати як функцію координації роботи та встановлення

параметрів мережі, так і працювати в режимі типового вузла;

б) пристрій з обмеженим набором функцій RFD (Reduced Function Device), що має лише можливість підтримки зв'язку з повнофункціональними пристроями.

Варіанти топології мереж стандарту зображені рисунку 7.2

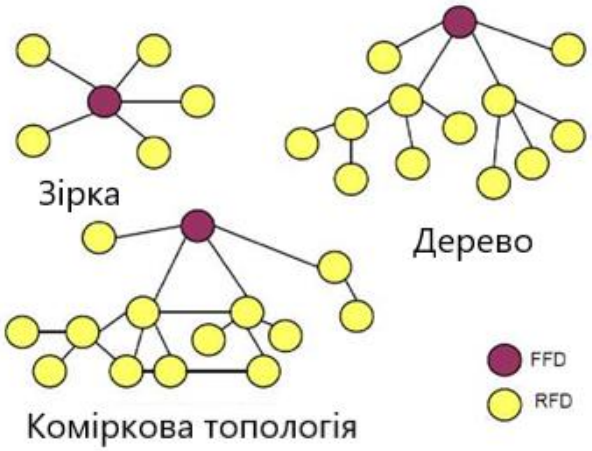


Рисунок 7.2 – Варіанти топології мереж стандарту IEEE Std 802.15.4.

7.2 Стандарт ZigBee

Стандарт IEEE Std 802.15.4 описує два нижні рівні мережевої моделі OSI, при цьому не визначає вимоги до верхніх рівнів та умови їх сумісності. Вирішення цих завдань вимагало розробки спеціальних комунікаційних протоколів. Найбільш відомими є протоколи альянсу ZigBee, якою був створений найбільшими світовими компаніями, що спеціалізуються на розробці програмно-апаратних засобів для інфокомунікаційних систем. Стандарт ZigBee включає повний стік протоколів для бездротових сенсорних мереж. Назва специфікації ZigBee походить від Zig-zag – зигзаг та Веї – бджола. Передбачалося, що топологія мережі нагадуватиме зигзагоподібну траєкторію польоту бджоли від квітки до квітки.

Специфікація ZigBee орієнтована на програми, що вимагають гарантованої безпечної передачі даних при відносно невеликих швидкостях та можливості тривалої роботи мережевих пристроїв від автономних джерел живлення (батарей). Вона забезпечує невисоке споживання енергії та передачу даних зі швидкістю до 250 Кбіт/с на відстань до 75 метрів за умов прямої видимості. Характеристики ZigBee: частотний діапазон 868/915/2400 МГц; бітова швидкість 20/40/250 кбіт/с; тип модуляції сигналу BPSK/BPSK/O-QPSK; метод розширення спектра DSSS; чутливість приймача мінус 92...мінус 85 дБм; вихідна потужність передавача мінус 32...0; Обсяг даних пакета до 127 байт.[3]

Стандарт ZigBee включає опис мережевих процесів керування,

сумісності та профілів пристроїв, а також інформаційної безпеки. На мережевому рівні у ZigBee визначено механізми маршрутизації та формування логічної топології мережі. Конфігурація представлена рисунку 7.3.

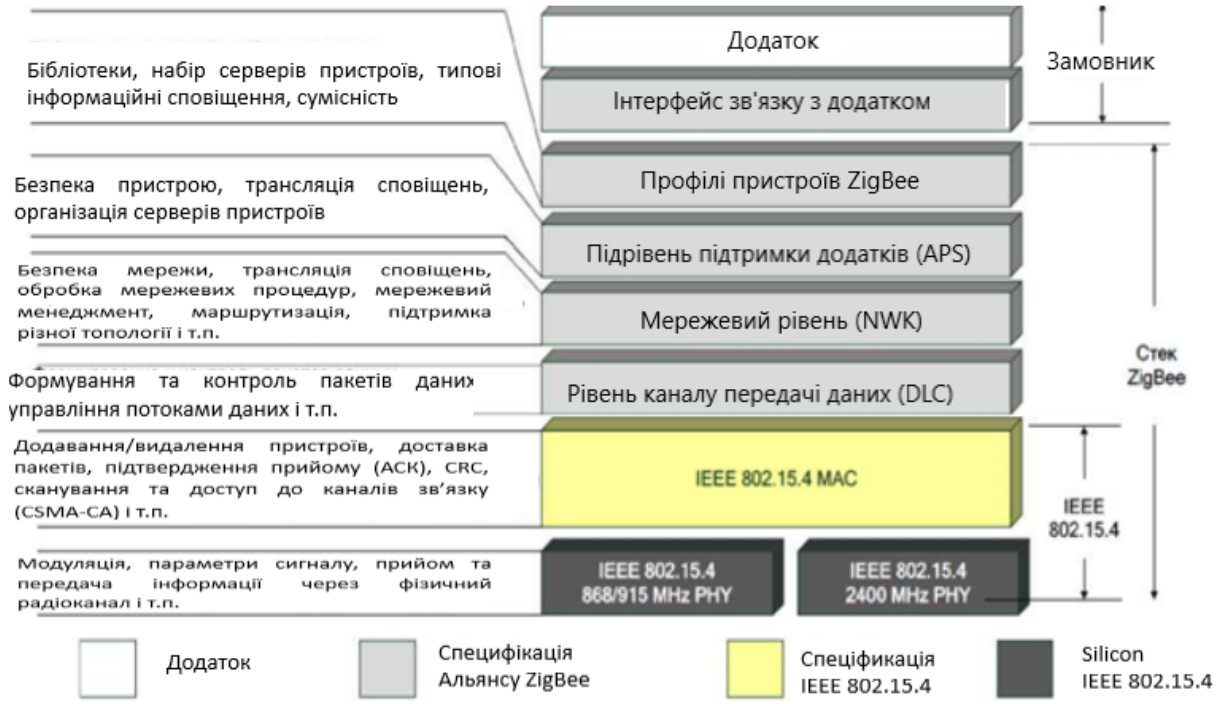


Рисунок 7.3 – Конфігурація стеків протоколу IEEE Std 802.15.4 та ZigBee

Основна особливість технології ZigBee полягає в тому, що вона при малому енергоспоживання підтримує не тільки прості топології мережі («точка-точка», «дерево» і «зірка»), але і осередок (mesh), що самоорганізується і самовідновлюється, з ретрансляцією і маршрутизацією повідомлень . Крім того, специфікація ZigBee містить можливість вибору алгоритму маршрутизації, залежно від вимог програми та стану мережі, механізм стандартизації додатків - профілі додатків, бібліотека стандартних кластерів, кінцеві точки, прив'язки, Гнучкий механізм безпеки, а також забезпечує простоту розгортання, обслуговування та модернізації.

У галузі технологій бездротових сенсорних мереж ZigBee є стандартом, найбільше підкріпленим представленими на ринку повністю сумісними апаратними та програмними засобами. Крім того, протоколи ZigBee дозволяють мережевим пристроям перебувати в сплячому режимі більшу частину часу, що істотно збільшує ресурс роботи вузлів при живленні від батарейних джерел. В БСМ на основі ZigBee підтримується режим профілів пристроїв або профілів для різних датчиків, які сумісні на рівні стека протоколу і можуть об'єднуватися в мережу, передавати, приймати і ретранслювати інформацію. У той же

час «розуміти» цю інформацію буде лише пристрій, для якого вона призначена.

Всі пристрої стандарту ZigBee в залежності від рівня складності поділяються на три класи, вищий з яких - координатор - управляє процесом формування мережі, зберігає дані про її топологію і служить шлюзом для передачі даних, що збираються від усіх сенсорів БСМ, для їх подальшої обробки. У мережі, як правило, використовується лише один PAN-координатор. Середній за складністю пристрій - маршрутизатор - здатний ретранслювати повідомлення, підтримувати всі топології мережі, а також виконувати функції координатора кластера. І, нарешті, найпростіший вузол – кінцевий пристрій – здатний лише передавати дані найближчому маршрутизатору.

Таким чином, стандарт ZigBee підтримує мережу із кластерною архітектурою, сформованою зі звичайних вузлів, об'єднаних у кластери за допомогою маршрутизаторів. Маршрутизатор кластерів запитують сенсорні дані від пристроїв і, ретранслюючи їх один одному, передають координатору, який зазвичай має зв'язок із зовнішньою IP-мережею, куди і відправляє інформацію для накопичення та остаточної обробки. Топологія зображено рисунку 7.4.

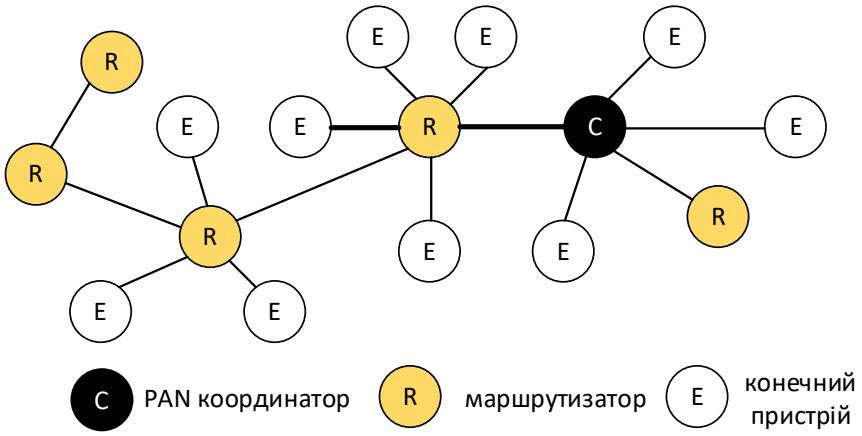


Рисунок 7.4- Типова топологія мережі ZigBee

Мережа ZigBee є самоорганізується, тобто всі вузли здатні самостійно визначати та коригувати маршрути доставки даних. Дані передаються за допомогою радіопередавачів від одних вузлів до інших ланцюжком, і в результаті найближчі до шлюзу вузли скидають всю акумульовану інформацію на шлюз. Ця інформація включає дані, що зчитуються з сенсорних датчиків, а також дані про стан пристроїв та результати процесу передачі інформації. У разі виходу частини пристроїв з ладу робота сенсорної мережі після реконфігурації повинна продовжитися. Бездротові вузли функціонують під керуванням спеціальної програми. Зазвичай всі вузли сенсорної мережі використовують одну і ту ж керуючу програму, що забезпечує їх функціональність і виконання мережевих протоколів. таких як

телевізори, телеприставки та ігрові консолі. Вона має ряд переваг у порівнянні з існуючими технічними рішеннями для дистанційного керування, включаючи керування роботою в зоні непрямої видимості, функціональність маніпулятора типу "миша" та клавіатури, керування з розпізнаванням жестів та сенсорним введенням, двосторонній зв'язок, більш тривалий час роботи від акумулятора.

7.3 Стандарт 6L0WPAN

(IPv6 Low-Power Wireless Personal Area Network) – стандарт, що забезпечує взаємодію малих бездротових мереж з мережами IP за протоколом IPv6 з малим енергоспоживанням. Стандарт розроблений групою IETF і описаний в RFC 4944 та RFC 4919. Технологія використовується в основному для організації мереж датчиків та автоматизації житлового та офісного приміщення з можливістю керування через інтернет, проте може використовуватися і автономно для реалізації простих бездротових мереж датчиків. Передача даних у стандарті 6LoWPAN передбачає використання субгігагерцового діапазону та забезпечує швидкість передачі від 50 до 200 Кбіт/с на відстань до 800 метрів. [3]

Архітектура мереж 6LoWPAN дещо відрізняється від традиційних архітектур IP-мереж (наявність спеціалізованого комутаційного обладнання, маршрутизаторів, медіа-конверторів) і від архітектур бездротових мереж збору даних, що склалися. Найближче до неї знаходиться архітектура WiFi-мереж, хоч і від неї є низка відмінностей.

Виділяють три типи мереж 6LoWPAN, зображені на рисунку 7.5:

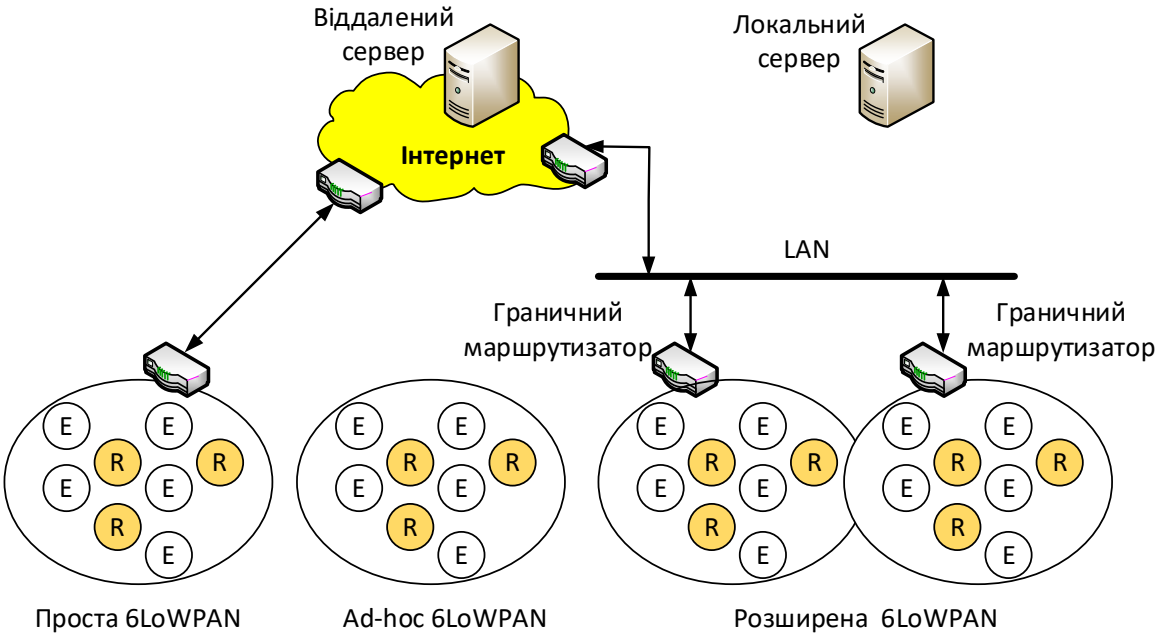


Рисунок 7.5 - Типи мереж 6L0WPAN (R-маршрутизатор, H-хост)

- ad-hoc (самоорганізується, динамічна);
- проста 6LoWPAN-мережа; розширена 6LoWPAN-мережа.

Ad-hoc-мережа є мережею, що самоорганізується, використовує стек протоколів 6loWPAN. Не має граничного маршрутизатора, немає підключення до зовнішньої IP-сети.

Проста 6LoWPAN-мережа підключена до іншої IP-мережі за допомогою одного граничного маршрутизатора. Граничний маршрутизатор може бути підключений безпосередньо до зовнішньої IP-мережі або може входити до складу кампусної мережі (наприклад, мережі організації).

Розширена мережа 6LoWPAN складається з однієї або декількох підмереж, підключених до зовнішньої IP-мережі через кілька граничних маршрутизаторів, підключених до однієї мережі (наприклад, локальна мережа організації). При цьому граничні маршрутизатори в розширеній мережі поділяють той самий мережевий префікс. Вузли розширеної мережі можуть вільно переміщатися в межах мережі та здійснювати обмін із зовнішньою мережею через будь-який граничний маршрутизатор (зазвичай вибирається маршрут із найкращими показниками якості сигналу - рівень помилок, рівень сигналу).

Взаємодія між вузлами в мережі 6L0WPAN, а також взаємодія із зовнішніми вузлами здійснюється так само, як і у звичайній IP-мережі. Кожен вузол має свою унікальну IPv6-адресу і може приймати та передавати пакети IPv6.

Спрощена структура стека протоколів 6L0WPAN у порівнянні зі стеками TCP/IP та ZigBee представлена на рисунку 7.6.

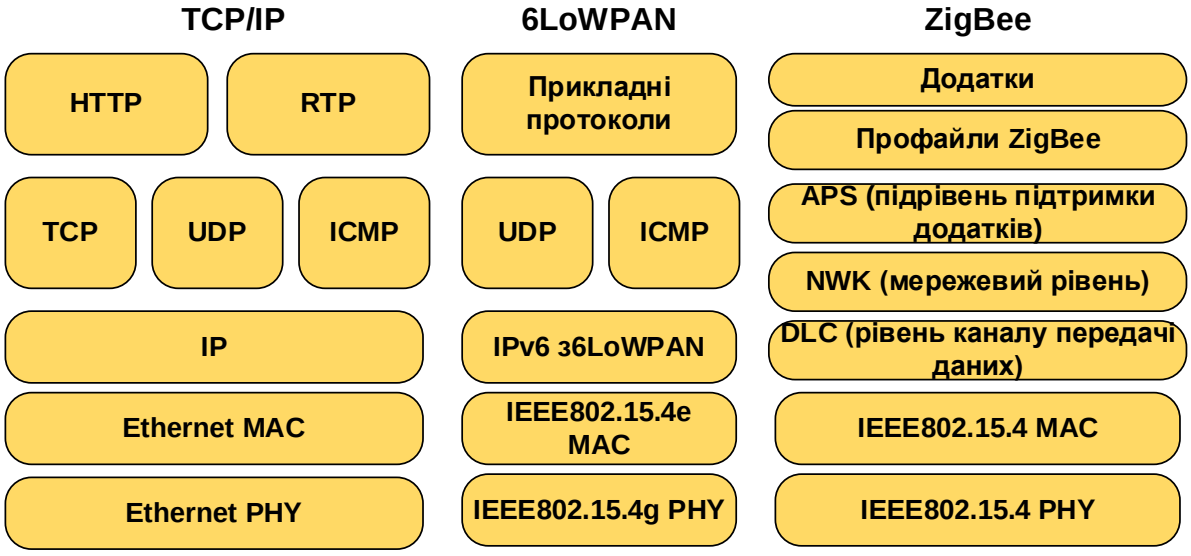


Рисунок 7.6 - Порівняння стеків протоколів TCP/IP, 6L0WPAN та ZigBee

Зазвичай вузли мають підтримку протоколу ICMPv6 та UDP.

Прикладні протоколи найчастіше використовують бінарний формат даних під час роботи з UDP-протоколу в мережах 6LoWPAN. На відміну від TCP/IP-стека, в 6LoWPAN немає підтримки протоколу транспортного рівня TCP - через великі накладні витрати на формування пакетів і через особливості роботи протоколу, які суттєво ускладнюють його застосування в сенсорних бездротових мережах (підтвердження пакетів та встановлення/ розрив з'єднання вимагають частої роботи приймача вузла, і, як наслідок, підвищеного споживання енергії).

Як і мережі ZigBee, мережі 6L0WPAN є самоорганізуються. Для цього використовується стандартна техніка мереж IPv6. На основі заданих параметрів стека автоматично встановлюється оптимальна топологія зв'язків між вузлами в мережі. Оптимальні маршрути визначаються з урахуванням метрик.

На відміну від стандартів ZigBee, 6L0WPAN розширює стандартизацію рівня прикладних завдань, паралельно вирішуючи проблеми з інтеграцією невеликих бездротових вузлів в IP-сети.

Цільові програми стека 6L0WPAN включають досить великі масштабовані мережі з підключенням до IP-мереж (інтернет, інтранет або екстранет). Незважаючи на хорошу масштабованість, потенційно прозоре керування та легкий доступ до вузлів, 6LoWPAN підходить не для всіх застосувань. Зокрема, поточна версія стандарту стека протоколів вимагає постійної активності маршрутизаторів для коректної передачі даних, що у сенсорних бездротових мережах. Тим не менш, ця особливість дозволяє мінімізувати об'єм flash-пам'яті в кінцевому пристрої, що займається стеком 6LoWPAN, і, отже, мінімізувати вартість мережевого процесора.

Основні сфери застосування стандарту 6LoWPAN: інтелектуальні системи обліку; керування вуличним освітленням; промислова автоматика; логістичні системи, відстеження товарів чи об'єктів інвентаризації; комерційні охоронні системи, системи контролю та управління доступом; деякі військові програми.

Деякі області застосувань 6LoWPAN перегукуються з низкою стандартів ZigBee, проте в даному випадку конкуренція відсутня, швидше, - взаємодія та доповнення один одного, особливо в плані інтеграції сервісів, розширення зон дії мережі.

7.4 Стандарти WirelessHART та ISA100.11a

Стандарти бездротових промислових мереж WirelessHART і ISA100.11a, як і розглянуті раніше технології ZigBee і 6L0WPAN, є надбудовами над фізичним рівнем стандарту IEEE 802.15.4. Обидва стандарти мають загальний принцип роботи та конкурують між собою.

WirelessHART - протокол передачі даних бездротової лінії зв'язку, розроблений фондом HART Communication Foundation передачі даних як HART-повідомлень у бездротовому середовищі. Вихідний протокол

обміну даними HART у провідних мережах був призначений для взаємодії з польовими датчиками на основі розширюваного набору простих команд «запросответ», що передаються в цифровому вигляді по двопровідній лінії зі струмом 4...20 мА. Його бездротовий варіант WirelessHART забезпечує передачу даних зі швидкістю до 250 кбіт/с на відстань до 200 м (не більше прямої видимості) при частоті передачі у діапазоні 2,4 ГГц. WirelessHART схвалено міжнародною електротехнічною комісією (МЕК) як перший міжнародний стандарт бездротового зв'язку промислової автоматизації під номером IEC 62591. Порівняння стеків зображено на рисунку 7.7.

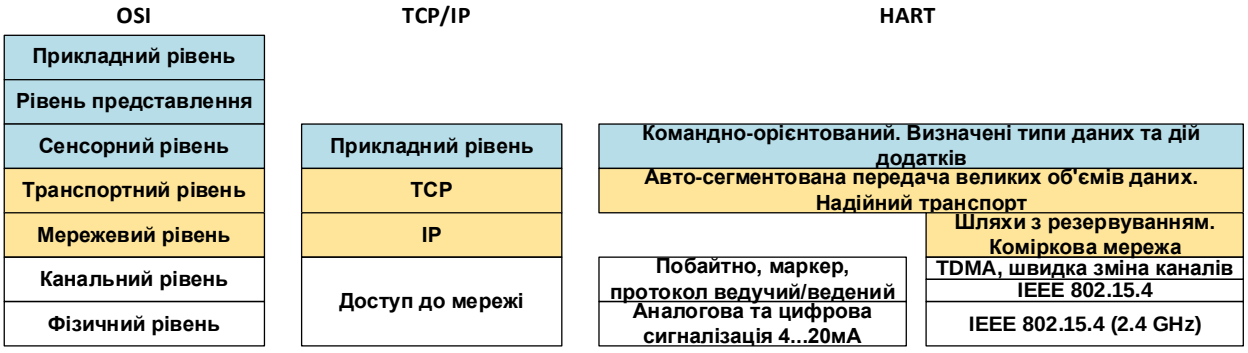


Рисунок 7.7 – Порівняння стеків

Бездротова мережа WirelessHART складається з трьох основних елементів, архітектура представлена на рисунку 7.8:

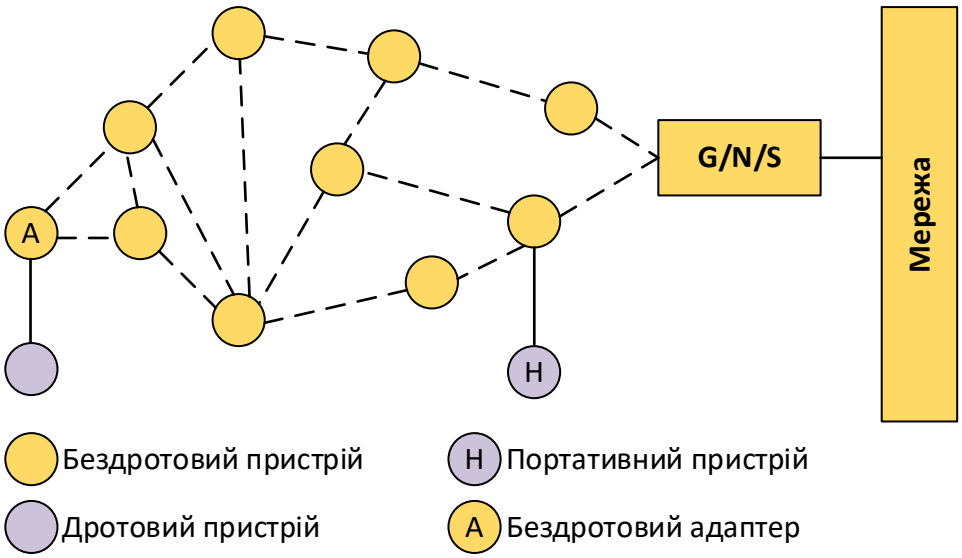


Рисунок 7.8 – Архітектура мережі WirelessHART

а) бездротові польові пристрої, приєднані до промислового обладнання. Це може бути пристрій із вбудованою дротовою технологією WirelessHART або вже наявний встановлений дротовий

HART-пристрій з адаптером WirelessHART;

б) шлюзи - забезпечують обмін даними між польовими пристроями та хост-додатками, приєднаними до високошвидкісної магістральної або іншої наявної на підприємстві комунікаційної мережі;

в) адміністратор мережі/менеджер безпеки - відповідає за конфігурування мережі, планування обміну даними між пристроями, маршрутизацію повідомлень та моніторинг стану мережі. Адміністратор мережі може бути вбудований у шлюз, хост-додаток чи контролер автоматизації технологічного процесу.

Мережа WirelessHART заснована на сумісних з IEEE 802.15.4 радіопередавачі, що працюють в ISM діапазоні 2,4 ГГц. У них використовується технологія широкосмугового сигналу з прямою послідовністю та перемиканням каналів для забезпечення комунікаційної безпеки та надійності, а також технологія синхронізованого багатостанційного доступу з тимчасовим поділом каналів (TDMA) та контрольованою затримкою зв'язку між пристроями в мережі.

Кожен пристрій у мережі може служити маршрутизатором для повідомлень від інших пристроїв. Інакше кажучи, пристрій немає необхідності звертатися безпосередньо до шлюзу; воно просто передає своє повідомлення на найближчий сусідній пристрій. Це розширює масштаб мережі та забезпечує надлишкові канали передачі даних для підвищення надійності.

Адміністратор мережі визначає надлишкові канали на основі часу затримки, ефективності та надійності передачі. Щоб забезпечити відкритість та вільність надлишкових каналів, передача повідомлень поперемінно здійснюється по кожному з них.

Схема мережі WirelessHART також дозволяє легко додавати та переміщувати пристрої. Пристрій завжди залишається на зв'язку, коли він знаходиться в зоні дії інших пристроїв у мережі.

Для забезпечення гнучкості за різних умов застосування стандарт WirelessHART підтримує кілька режимів передачі даних, включаючи однонаправлену публікацію значень параметрів технологічного процесу та управління, миттєве повідомлення з виключення, спеціальний запит/відгук та передача великих наборів даних з автоматичним сегментуванням. Ці можливості дозволяють налаштовувати передачу даних відповідно до виробничих вимог, що знижує енергоспоживання та непродуктивні витрати.

ISA100.11a - стандарт організації промислових сенсорних мереж, мереж датчиків та приводів. Стандарт розроблений Міжнародним товариством з автоматики ISA (International Society of Automation) та схвалений МЕК як загальнодоступна специфікація. Для передачі промислових даних використовується бездротовий низькошвидкісний зв'язок з використанням елементів з низьким енергоспоживанням. Обмін даними здійснюється на частоті в районі 2,4 ГГц та швидкості

близько 250 кбіт/с. В основі архітектури ISA100.11a, як і в протоколі WirelessHART, лежить стандарт IEEE 802.15.4-2006, порівняння стеків показано рисунку 7.9.

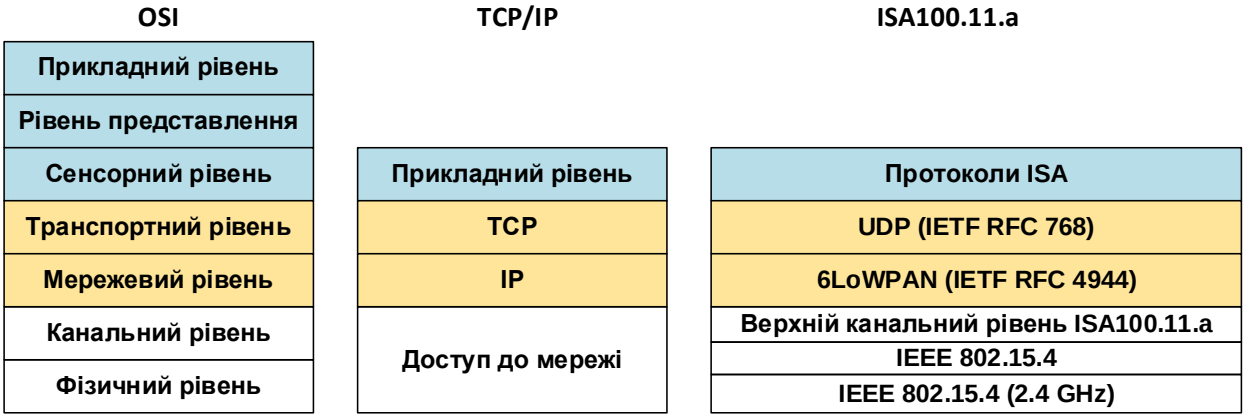


Рисунок 7.9 - Порівняння стеків протоколів OSI, TCP/IP та ISA100.11a

Бездротова мережа стандарту ISA100.11a містить такі компоненти, як показано на рисунку 7.10 - польовий пристрій з функцією маршрутизатора; польовий пристрій без функції маршрутизатора; магістральний маршрутизатор; шлюз; системний менеджер; менеджер безпеки

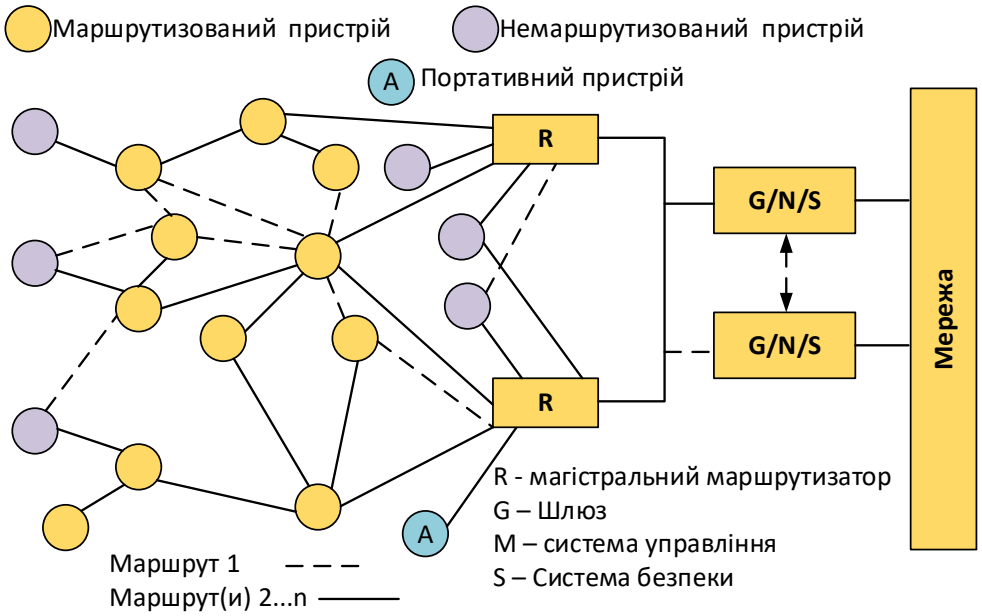


Рисунок 7.10 – Архітектура безпровідної мережі ISA100.11a

ISA100.11a підтримує протоколи Fieldbus Foundation, Profibus-PA та HART, що працюють на рівні додатків. Фактично, він здатний

підтримувати кілька кластерів пристроїв, які працюють із зазначеними протоколами. Він також може підтримувати різні типи датчиків (HART, Profibus та ін.) одного кластері.

Стандарт ISA100.11a використовує топологію мереж датчиків типу «коміркова мережа» або «зірка». Мережі з топологією типу «коміркова мережа», що виконують безліч перемикань, використовують більше заряду батарей, ніж мережі з топологією типу «зірка», але є безпечнішими. Таким чином, у користувача є вибір і він може віддати перевагу тому чи іншому способу побудови мереж, залежно від завдань, що вирішуються.

Протоколи WirelessHART та ISA100.11a мають багато спільного, т.к. за основу взято стандарт IEEE 802.15.4-2006. З метою підвищення надійності бездротових систем для підприємств в обох випадках фізично використовується технологія псевдовипадкової перебудови робочої частоти FHSS (Frequency Hopping Spread Spectrum), а на канальному рівні метод кодового поділу CDMA замінений на метод тимчасового поділу TDMA, порівняння протоколів на рисунку 7.11.

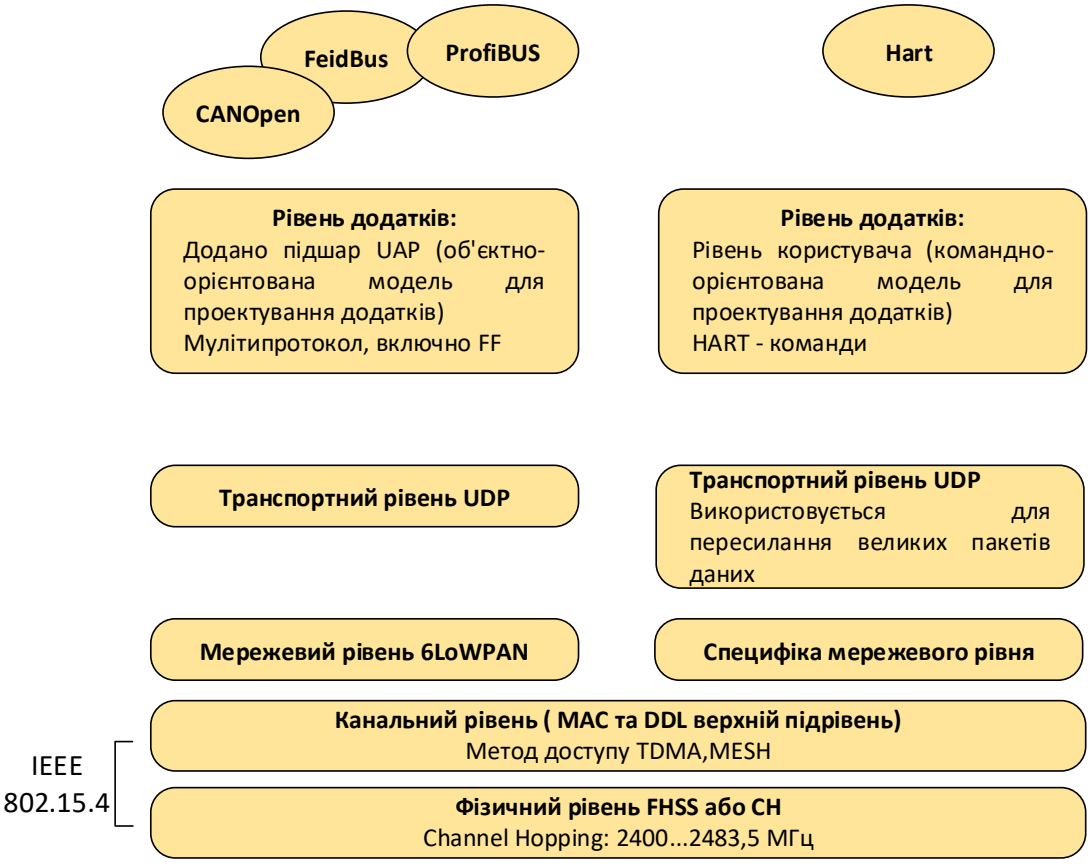


Рисунок 7.11 – Порівняння стеків протоколів стандартів ISA 100.11a та WirelessHart

Обидва стандарти останніх версій підтримують мову опису електронних пристроїв EDDL (Electronic Device Description Language) для забезпечення сумісності польових пристроїв від різних виробників.

Однак є й суттєві відмінності між цими протоколами. У ISA 100.11a використовується мережевий рівень моделі OSI з урахуванням протоколу 6 LoWPAN (RFC4944), тобто. передбачена 128-бітова IPv6-адресація польових пристроїв, яка в основному застосовується на мережевому рівні магістральних маршрутизаторів або шлюзів. Всередині бездротової мережі ISA 100.11a використовується укорочена - 16-бітова адреса EUI (без інкапсуляції та компресії IP-заголовка в рамках однієї бездротової мережі та з інкапсуляцією та компресією IP-заголовка за наявності двох або більше бездротових мереж). У той же час усередині бездротової мережі WirelessHart взагалі відсутня IP-адресація кінцевих пристроїв. Укорочена EUI-адресація та маршрутизація польового бездротового обладнання здійснюється на мережному рівні в рамках однієї бездротової мережі (не передбачена масштабованість мереж).

На прикладному рівні моделі OSI ISA100.11a для проектування додатків використовується концепція об'єктно-орієнтованої моделі, а Wireless Hart - командно-орієнтована. У ISA100.11a на прикладному рівні хоста введено додатковий підшар для управління UAP та між UAP, який за стандартом ISA для польових шин IEC 61158 розглядається окремо від моделі OSI. У Wireless Hart такого поняття немає.

ISA100.11a є повноцінним і перспективним стандартом з технологічної точки зору. Він ґрунтується на відкритих стандартах, а не власних технологіях. Наприклад, він підтримує протокол IPv6 комітету IETF у бездротових персональних мережах низької потужності (6LoWPAN). Адресація пристроїв IPv6 дозволяє використовувати тисячі датчиків і спростити їхнє підключення при переході до інтернету речей.

Хоча бездротова система ISA100.11a повністю усуває необхідність використання WirelessHART, на даний момент більше 15 виробників підтримують стандарт WirelessHART (IEC 62591), тоді як підтримка стандарту ISA100.11a обмежена всього трьома виробниками. Слід також зазначити, що дешевша технологія ZigBee застосовується для домашньої та офісної автоматизації, тоді як дорогі технології WirelessHART та ISA 100.11a призначені для мереж промислової автоматики.

7.5 Стандарт Z-Wave

Z-Wave – це перший відкритий бездротовий стандарт домашньої автоматизації (системи «розумний» будинок), в основі якої лежить пориста (mesh) мережа. Він заснований на специфікації ITU G.9959 та визначає всі аспекти взаємодії пристроїв, що підтримують цей протокол, а також забезпечує їхню сумісність. Технологія використовує малопотужні та мініатюрні радіочастотні модулі, які вбудовуються в побутову електроніку та різні системи, такі як освітлення, опалення, контроль доступу, розважальні системи та побутову техніку. Стек

протоколу Z-Wave представлений рисунку 7.12

Рисунок 7.12 – Стек протоколу Z-Wave

На відміну від Wi-Fi та інших стандартів передачі даних IEEE 802.11, призначених в основному для великих потоків інформації, стандарт Z-Wave працює в діапазоні частот до 1 ГГц і оптимізований для передачі простих керуючих команд (наприклад, увімкнути/вимкнути, змінити гучність, яскравість і т.д.). Вибір низького радіочастотного діапазону для Z-Wave обумовлюється малою кількістю потенційних джерел перешкод (на відміну від завантаженого діапазону 2,4 ГГц, в якому доводиться вдаватися до заходів, що зменшують можливі перешкоди від різних побутових бездротових пристроїв, що працюють - Wi-Fi, ZigBee, Bluetooth) . У Росії її використовується частотний діапазон 869 МГ ц.

Також іншими перевагами стандарту можна відзначити мале споживання енергії, низьку вартість виробництва та вбудовування модулів Z-Wave у різні побутові пристрої.

Швидкість передачі даних у мережі становить 9,6 кбіт/с або 40 кбіт/с із повною сумісністю. Використовується модуляція GFSK. Радіус дії приблизно 30 метрів в умовах прямої видимості, у приміщенні зменшується залежно від форми та матеріалу стін, а також від виду антени.

У мережі Z-Wave вузли поділяються на три типи: контролери (Controllers), виконавчі механізми, що маршрутизують (Routing Slaves) і виконавчі механізми (Slaves). У реальній мережі всі типи пристроїв можуть працювати у будь-якій комбінації.

Z-Wave використовує пористу топологію мережі з маршрутизацією повідомлень від джерела (англ. source routing) і має один основний контролер і нуль або більше вторинних контролерів, які керують маршрутизацією та безпекою. У пористій мережі Z-Wave кожен вузол або пристрій може приймати та передавати керуючі сигнали іншим пристроям мережі, використовуючи сусідні проміжні вузли. Це мережа з

маршрутизацією, що самоорганізується, залежною від зовнішніх факторів - наприклад, при виникненні перешкоди між двома найближчими вузлами мережі, сигнал піде через інші вузли мережі, що знаходяться в радіусі дії.

Z-Wave - це протокол для організації надійного бездротового зв'язку в напівдуплексному режимі з низькою пропускною здатністю. В основному призначений для автоматизації та управління пристроями в рамках концепції IoT та розумний будинок.

Спочатку протокол Z-Wave був розроблений як стартапа групою фахівців із Данії, який згодом був придбаний американською компанією Sigma Designs – провідним постачальником системних рішень для розумного дому. У 2005 році групою провідних фахівців компаній, що займаються домашньою автоматизацією, створила некомерційну організацію Z-Wave Alliance. Метою цієї організації є сертифікація та забезпечення сумісності пристроїв Z-Wave від різних виробників. У Z-Wave Alliance входять понад 540 провідних галузевих компаній, серед яких: ADT, Fibaro, Huawei, Nexia, D-link, Ingersoll Rand, Jasco, LG Up+, Nortek Security & Control, Sigma Designs, Samsung, Bosch та інші.

У жовтні 2013 року вийшов другий стандарт Z-Wave - Z-Wave Plus, що відрізняється покращеними характеристиками та наявністю нових функцій.

Технологія Z-Wave описана у специфікації ITU-T G.9959. Z-Wave має пористу топологію мережі, в якій кожен вузол знає сусідів і може виступати як ретранслятор пакетів для успішного обміну повідомленнями.

Технічні характеристики:

а) відстань: 40...120 м (відстань між пристроями за умови прямої видимості: 10.30 м);

б) максимальна кількість пристроїв: 232;

в) швидкість передачі: 9,6 кбіт/с, 40 кбіт/с або 100 кбіт/с;

г) частота: до 1 ГГц (869,42 МГц у Європі, 908,4 МГц та 916 МГц у США, 922... 926 МГ ц у Японії, 869 МГц).

Пристрої в мережі Z-Wave можуть бути розділені на дві підгрупи: звичайні вузли (перебувають у режимі прийому в будь-який час) і вузли з малою потужністю (велику частину часу знаходяться в режимі сну). Використання вузлів малої потужності продовжує термін служби пристроїв за рахунок низького електроживлення.

Основними перевагами технології Z-Wave є універсальність, тобто. сумісність пристроїв від різних виробників, простота встановлення та надійна передача даних. До недоліків можна віднести неможливість передачі звуку або відео з гарною якістю. Ціна пристроїв, що використовують технологію Z-Wave, варіюється від 10 до 300$.

Технологія Z-Wave активно використовується для зв'язку пристроїв розумної оселі: побутова техніка, штори, освітлення, клімат-контроль, контроль доступу і т.д. На ринку представлено понад 1700 готових рішень, наприклад, повідомлення про відкриття/закриття

дверного замку, коли ви знаходитесь поза домом, голосове управління освітленням та обігрівом приміщення, збір даних із різних сенсорів. [8].

Таким чином, мережа Z-Wave може мати радіус передачі набагато більший, ніж дальність передачі одного вузла. Однак через переприйом (hops) може бути отримана невелика затримка між командою управління та бажаним результатом. Для того щоб Z-Wave пристрої мали можливість маршрутизувати дані, що ними не запитуються, вони не можуть перебувати в сплячому режимі. Таким чином, пристрої з живленням від батарейок не призначені як пристрої ретрансляції. Мережа Z-Wave може містити до 232 пристроїв з можливістю розширення мережі, якщо потрібно ще кілька пристроїв. Додаткові пристрої в мережу можуть бути додані в будь-який час, так само як і кілька контролерів, що управляють.

Хоча технологія Z-Wave є простим та дешевим рішенням, низька швидкість передачі даних виключає передачу зображень, звуку та високошвидкісних даних. Крім того, для рішень, де потрібно понад 30 пристроїв, Z-Wave-система є більш дорогою, ніж кабельні системи. Через свої конструктивні особливості такі системи мають обмежені масштаби та радіус дії, і вимагають використання повторювачів або навіть кабельних з'єднань. У світі налічується понад 200 виробників, що пропонують товари із Z-Wave чіпами чи модулями. Відмінною рисою Z-Wave є те, що всі ці продукти сумісні між собою. Порівняння стека протоколу Z-Wave з іншими технологіями наведено на рисунку 7.13.

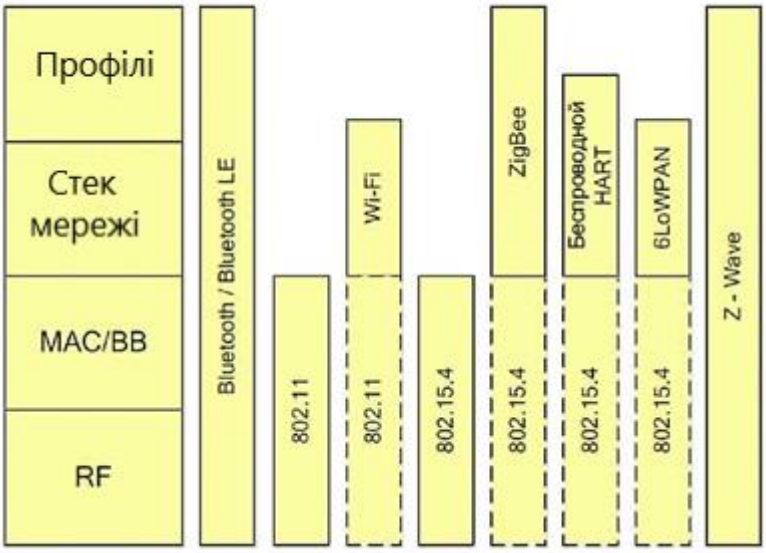


Рисунок 7.13 - Характеристики технології Z-wave

Характеристики Z-wave: частотний діапазон 868/908/2400 МГц; бітова швидкість 9,6/40/200 кбіт/с; тип модуляції сигналу BPSK; чутливість приймача мінус 101 дБм; вихідна потужність передавача мінус 20...0; Обсяг даних пакета до 64 байт.

7.6 Стандарт Bluetooth Low Energy

Технологія Bluetooth Low Energy (BLE) - Bluetooth 4.0 є бездротовою технологією для ближніх комунікацій, розробленою групою Bluetooth Special Interest Group (SIG). Спочатку орієнтований застосування у системах збору даних, моніторингу з автономним харчуванням. BLE споживає в 10-20 разів менше енергії здатний передавати дані в 50 разів швидше, ніж класичні Bluetooth-рішення.

Стандарт Bluetooth Low Energy розрахований на топології типів: точка-точка та зірка. Основні області застосування BLE - це пристрої забезпечення безпеки, управління електроприладами та відображення показань, датчики з батарейним живленням, домашні медичні прилади, спортивні тренажери.

Пристрої BLE працюють у діапазоні 2,4 ГГц. У стандарті визначено 40 частотних каналів з відстанню 2 МГц між каналами. Фізично застосована GFSK-модуляція (Gaussian Frequency Shift Keying) з індексом модуляції в межах від 0,45 до 0,55, що дає змогу зменшити пікове споживання енергії. Швидкість передачі фізично 1 Мбіт/с. У стандарті BLE чутливість приймача визначена як рівень сигналу на приймачі, при якому частота бітових помилок BER (Bit Error Rate) досягає рівня 10". Вона повинна становити мінус 70 дБм або краще. [7]

Технологія адаптивної стрибкоподібної перебудови частоти, що використовується в BLE, дозволяє пристроям швидко змінювати робочу частоту широкому діапазоні робочих частот. Це не тільки дозволяє знизити інтерференцію, а й зменшити або повністю уникнути переповнення робочого частотного діапазону. Поряд із широкомовним режимом, BLE пропонує спосіб передачі даних, орієнтований на встановлене між окремими пристроями з'єднання.

Стек BLE складається з 2х основних частин: контролера (controller) і вузла мережі (host, зображений на малюнку 30. Контролер включає фізичний і канальний рівень і часто реалізується у вигляді системи "на" кристалі з інтегрованим бездротовим трансівером. Частина стека, іменована вузлом мережі, реалізується програмно на мікроконтролері додатків і включає функціональність верхніх рівнів, зображено на рисунку 7.14:

- протокол адаптації L2CAP (Logical Link Control and Adaptation Protocol),
- протокол атрибутів ATT (Attribute Protocol),
- протокол атрибутів профілів пристроїв GATT (Generic Attribute Profile),
- протокол забезпечення безпеки SMP (Security Manager Protocol),
- протокол забезпечення доступу до функцій профілю пристроїв GAP (Generic Access Profile).

Взаємодія між верхньою та нижньою частинами стека здійснюється

через інтерфейс HCI (Host Controller Interface). поверх рівня вузла мережі.

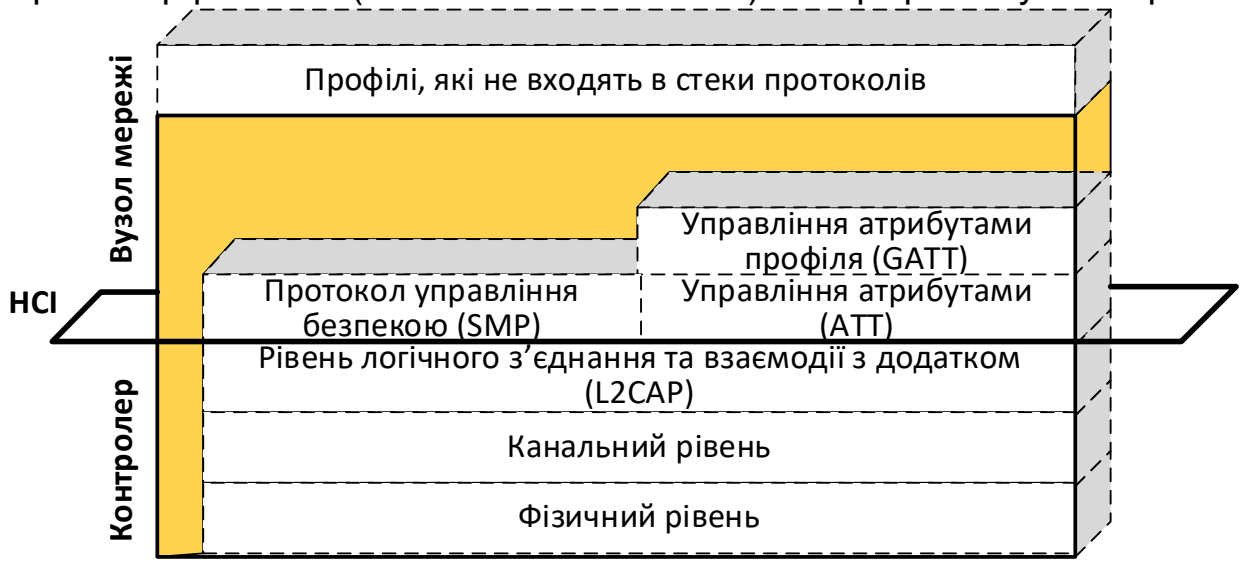


Рисунок 7.14 - Структура стека протоколів Bluetooth Low Energy

Незважаючи на те, що деякі функції контролера BLE запозичені у класичного Bluetooth, вони не сумісні між собою. пристрій, який підтримує лише BLE (однорежимний пристрій - single-mode device) не зможе взаємодіяти з пристроєм, що підтримує лише Bluetooth версій 2.x/3.0. Для здійснення взаємодії між ними хоча б один із пристроїв повинен підтримувати обидва стеки протоколів (дворежимний пристрій - dual-mode device). Однорежимні пристрої мають найменше енергоспоживання і в основному являють собою кінцеві виконавчі пристрої. Дворежимні пристрої передбачають можливість періодичного отримання енергії, розташовуються на різних мобільних пристроях, а також можуть функціонувати і як звичайні пристрої Bluetooth.

Характеристики BLE: частотний діапазон 2400 МГц; бітова швидкість 1000 кбіт/с; тип модуляції сигналу GFSK; метод розширення спектра FNSS; чутливість приймача мінус 70...93 дБм; вихідна потужність передавача мінус 20.0; Обсяг даних пакета до 8.47 байт.

Характеристики Bluetooth – частотний діапазон 2400 МГц; бітова швидкість 1000 кбіт/с; тип модуляції сигналу GFSK; метод розширення спектра FNSS; чутливість приймача мінус 90 дБм; вихідна потужність передавача 20/4/0; Обсяг даних пакета до 358 байт.

Низьке енергоспоживання та стійкіша робота в умовах великої кількості аналогічних пристроїв у ряді випадків дозволяє розглядати BLE як альтернативу пристроям NFC, зокрема RFID-міткам. Але найцікавіший варіант використання BLE разом із NFC. У цьому випадку перші забезпечують більший радіус стійкої роботи і велику кількість пристроїв, що спільно працюють, а другі служать для встановлення логічного з'єднання між парою пристроїв, забезпечуючи більш високий рівень безпеки за рахунок меншого радіусу дії.

8 ОСНОВНІ ПОЛОЖЕННЯ ІНТЕЛЕКТУАЛЬНИХ СИСТЕМ НА БАЗІ НЕЧІТКОЇ ЛОГІКИ

8.1. Основні положення теорії множин

Класичну теорію множин можна розглядати як граничний випадок теорії нечітких множин. Тому розглянемо спочатку основні постулати теорії множин.

Множина – фундаментальне невизначене поняття. Інтуїтивно множина X − це деякий набір об'єктів x, які називаються елементами множини:

$$x \in X.$$

Важливим є поняття рівності множин, для якого справедливі властивості:

− рефлексивність $X = X$;

− симетричність: якщо $X = Y$, то $Y = X$;

− транзитивність: якщо $X = Y$ та $Y = Z$, то $X = Z$.

Множина X є підмножиною множини Y, якщо всі його елементи є одночасно елементами Y:

$X \subset Y$.

Поняття підмножини має такі властивості:

− $X \subset X$ (рефлексивність);

− якщо $X \subset Y$ та $Y \subset Z$, то $X \subset Z$ (транзитивність);

− $\emptyset \subset X$ (де X – будь-яка множина, а ∅ – порожня множина).

Якщо в деякому розгляді беруть участь лише підмножини фіксованої множини U, то цю найбільшу множину називають *базовою* (універсальною) множиною. Базова множина може бути безперервною або дискретною. Наприклад, як базова множина може виступати евклідовий простором R^n, а підмножинами є окремі групи векторів цього простору.

Існують два традиційних способи завдання множини: перерахування та опис.

Перерахування використовується, якщо множина має невелику кількість елементів або ясно, що розуміти під трьома крапками, коли множина нескінченна: {2,4,8...}.

Описовий спосіб завдання множини:

$$A = \{x \in X \,/\, x\text{-властивість}\},$$

тобто A складається з елементів базової множини X, які мають певну властивість.

Множина може бути визначена за допомогою спеціальної функції

– показника приналежності.

Показник приналежності $\eta_A(x)$ елемента *x* до деякої чіткої множини *A*, визначеної на базовій множині *U*, є двійковою функцією:

$$\eta_A(x) = \begin{cases} 1, x \in A \\ 0, x \notin A \end{cases}, \forall x \in U$$

На рис. 8.1 показаний приклад опису чіткої множини *A* = "Висока потужність" за допомогою характеристичної функції.

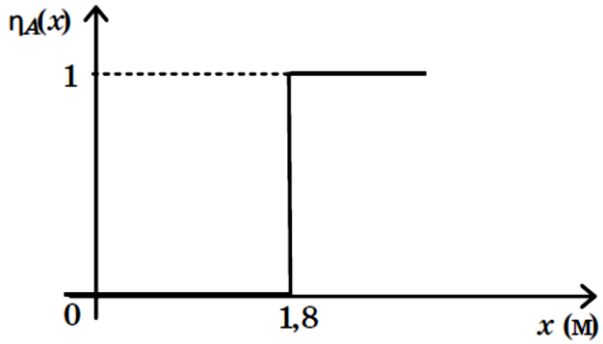


Рисунок 8.1 – Приклад опису множини з допомогою характеристичної функції

Нескінченна множина може бути лічильним або незліченним. Множина лічильна, якщо її можна привести до однозначної відповідності з натуральним рядом чисел. Якщо це неможливо, то множина є незліченною.

Властивістю нескінченної множини є можливість привести його у взаємно однозначну відповідність з його нескінченною підмножиною.

Для кінцевих множин задача знаходження найбільшого та найменшого елемента тривіальна:

$$X = \{9,8,1,4,3\},\ \max\{X\} = 9,\ \min\{X\} = 1.$$

Якщо ж множину задано описовим способом, то тут вводиться поняття верхньої та нижньої межі.

Нехай *S* – множена дійсних чисел. Верхньою межею *S* є число *C* таке, що для будь-якого $x \in S$ має місце $x \leq C$. Чисел, які можуть розглядатися як верхня межа, може бути нескінченно багато або не бути взагалі.

Супремум sup{*S*} - верхня межа, яка не перевищує будь-яку іншу верхню межу.

Інфінум inf{S} – нижня межа, яка не менша за будь-яку іншу нижню межу.

Основною характеристикою кінцевої множини *X* є її *потужність*, тобто кількість його елементів (позначається card{*X*}).

Наприклад, хай X = {5, 6, 7, 20, 99,1}, тоді card{X} = 6.

Дві множини можуть мати одну теж саму потужність (кардинальне число), якщо існує взаємно однозначна відповідність між цими множинами.

Упорядкованою множиною чи *кортежом* називається сукупність елементів, у якій кожен елемент займає певне місце.

Самі елементи називаються при цьому *компонентами кортежу* (перший елемент, другий елемент і так подалі). Тож якщо через h позначити висоту літака, а ще через v – його швидкість, то кортеж x= (h,v) буде описувати стан літака.

Кортеж дійсних чисел (x, y) може розглядатися як точка на площині чи вектор. Число елементів кортежу називається його *довжиною*. Кортеж дійсних чисел довжиною n можна розглядати як n-вимірний вектор або n-вимірний простір (гіперпростір).

Елементами множини можуть бути інші множини, наприклад:

$$X = \{9,8,1,4,3\},\ Y = \{1,3,6,11\},\ Z = \{X, Y\}.$$

Основні операції над множинами

1. *Об'єднання* (або сума) множин X і Y - це множина, яка складається з тих і тільки тих елементів, які належать хоча б одному з множин X або Y

$$X \cup Y \text{ (або X + Y)} = \{ x \,/\, x \in X \text{ або } x \in Y\}.$$

Наприклад:

$$X = \{9,8,1,4,3\},\ Y = \{1,3,6,11\},\ X \vee Y = \{1,3,4,6,8,9,11\}.$$

2. *Перетин* (або добуток) множин X і Y - це множина, що складається з тих і тільки тих елементів, які належать хоча б одній із множині X, так і множині Y:

$$X \cap Y \text{ (або } X \cdot Y) = \{x \,/\, x \in X \text{ та } x \in Y\}.$$

Наприклад:

$$X = \{9,8,1,4,3\},\ Y = \{1,3,6,11\},\ X \wedge Y = \{1,3\}$$

3. *Різниця множин*. Ця операція визначається лише для двох множин. Різницею множин X і Y називається множина, що складається з елементів, які належать X і не належать Y:

$$X - Y = \{x \,/\, x \in X \text{ та } x \notin Y\}.$$

Наприклад:

$$X = \{9,8,1,4,3\},\ Y = \{1,3,6,11\},\ X - Y = \{4,8,9\}.$$

4. *Доповнення*. Множина $\overline{X}$, що визначається із співвідношення

$$\overline{X} = U - X,$$

називають доповненням множини *X* (до універсальної множини *U*):

$$X = \{x \,/\, x \in U \text{ та } x \notin X\}$$

Наприклад:

$$U = \{0,1,2,3,4,5,6,7,8,9\},$$
$$X = \{9,8,1,4,3\},\ \overline{X} = \{0,2,5,6,7\}$$

Очевидно, виконуються такі властивості:

$$\overline{U} = \varnothing,\ \overline{\varnothing} = U,$$

де $\varnothing$ – порожня множина.

Для розглянутих вище операцій мають місце тотожності, наведені у табл. 8.1.

Таблиця 8.1 – Основні закони теорії множин

№ з/п	Закон	Формула
1	Комунікативності	$A \cup B = B \cup A;\ A \cap B = B \cap A$
2	Асоціативності	$A \cup (B \cup C) = (A \cup B) \cup C$ $A \cap (B \cap C) = (A \cap B) \cap C$
3	Дистрибутивності	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
4	Ідемпотентності	$A \cup A = A;\ A \cap A = A$
5	Подвійного заперечення	$\overline{\overline{X}} = X$
6	Де Моргана	$\overline{X \cup Y} = \overline{X} \cap \overline{Y};\ \overline{X \cap Y} = \overline{X} \cup \overline{Y}$
7	Протиріччя	$X \cup \varnothing = X;\ X \cap \varnothing = \varnothing;\ X \cap \overline{X} = \varnothing$

5. *Декартовим (прямим) додатком множин X* та *Y* називають множину, що складається з усіх тих і лише тих упорядкованих пар (кортежів), перша компонента яких належить множині *X*, а друга − множині *Y*. Таким чином, елементами декартового добутку є

двоелементні кортежі виду (*x*, *y*).
Формально це записується так:

$$X \times Y = \{(x, y) / x \in X, y \in Y\}.$$

Наприклад:

$$X = \{1,2,3\},\ Y = \{a,b,c\},$$
$$X \times Y = \{(1, a),(2, a),(3, a),(1, b),(2, b),(3, b),(1, c),(2, c), (3, c)\}.$$

Очевидно:

$$\text{card}\{X \times Y\} = \text{card}\{X\} \cdot \text{card}\{Y\}.$$

Операцію декартового твору множин можна ілюструвати графічно (рис. 8.2).

Операція прямого додатку справедлива і для більшої кількості множин, наприклад для трьох множин:

$$X \times Y \times Z = \{(x, y, z) / x \in X, y \in Y, z \in Z\}.$$

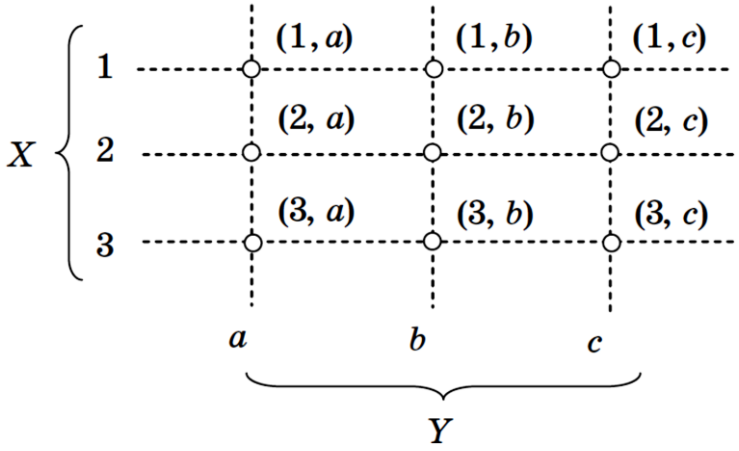


Рисунок 8.2 – Графічна ілюстрація декартового добутку

Ступінь множини *n* – це добуток *n* однакових множин:

$$X \times X \times \cdots \times X = \prod_{i=1}^{n} X^n.$$

6. Відношенням між *n* множинами X_1, $X_2 \ldots X_n$ *називається підмножина прямого добутку* $X_1 \times X_2 \times \ldots \times X_n$, для якого виконується задана властивість. Функція – це окремий випадок відносини.

Найчастіше розглядаються бінарні відносини. Наприклад:

$$R \subset X \times Y = \{(x, y); x \in X, y \in Y, x = y\}.$$

Зауважимо, що відношення є множиною, тому його можна задати шляхом вказівки належності $\eta_R(x, y)$ кожного кортежу декартового добутку до цього відношення:

$$R = \{\mu_R (x, y) / (x, y);\ x \in X,, y \in Y\}.$$

Приклад відношення для дискретної області визначення:

$$X = \{1,2,3,4\},\ Y = \{2,3,8\},$$
$$R_{X=Y} = \{0/(1,2),0/(1,3),0/(1,8),1/(2,2),0/(2,3),0/(2,8),0/(3,2),1/(3,3),$$
$$0/(3,8),0/(4,2),0/(4,3),0/(4,8)\ \}.$$

Приклад відношення для безперервної області визначення:

$$x, y \in R^1;\ c = \text{const};$$

$$R = \begin{cases} 1, & x^2 + y^2 \le c^2 \\ 0, & \text{інше.} \end{cases}$$

Це опис кола радіусом *c* з центром на початку координат.
Відносини відіграють важливу роль в описі знань.
Наприклад, нехай є множина країн та множина міст:

$$X = \{\text{Англія, Україна, Франція}\},$$
$$Y = \{\text{Лондон, Перт, Київ, Львів, Париж, Ліон}\}.$$

За допомогою вказівки приналежності (значення у колі на рис. 8.3) можна описати на $X \times Y$ відношення «Столиця держави».

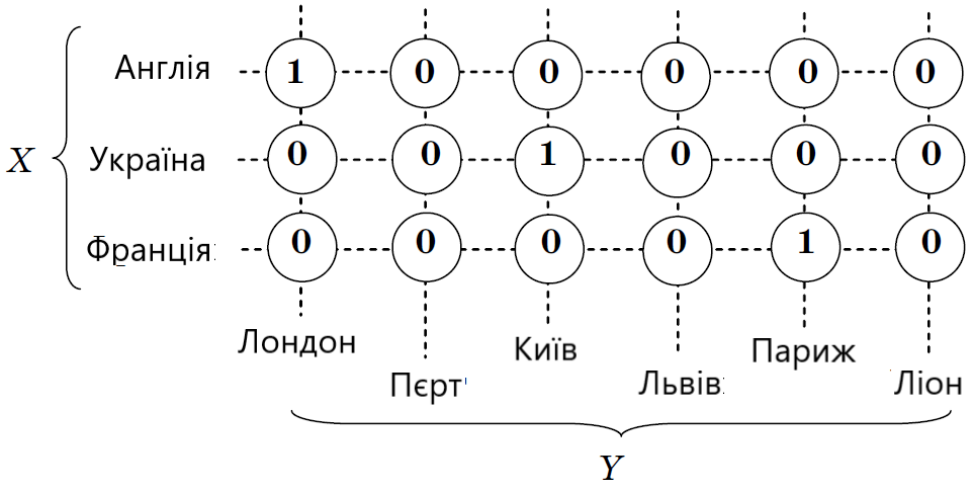


Рисунок 8.3 - Приклад опису відносини «Столиця держави»

Відносна відстань Хеммінгу *d* між двома кінцевими множинами

визначається як відношення кількості елементів, що входять тільки в одну або другу множину, к загальній кількості елементів базової множини.

Наприклад, нехай *U* – універсальна множина, на якій визначені множини *A* і *B*:

$$U=\{a, b, c, d, e\}, A=\{a, d, e\}, B=\{b, d\}.$$
$$d(A,B) = 3/5.$$

Розглянемо поняття розбиття множини.

Нехай *X* – множина, *A* – його підмножини:

$$A_i \subset X, i \in [1, n]$$

Підмножини A_i утворюють розбиття множини *X*, якщо виконуються дві умови:

$$1)\ A_i \cap A_j = \varnothing, i \neq j.$$
$$2)\ \bigcup_{i=1}^{n} A_i = X.$$

Наприклад: $X = \{a, d, v, 2, 5, f\}$, $A_1 = \{2, f\}$, $A_2 = \{d, v, 5\}$, $A_3 = \{a\}$

$A_1 \cap A_2 = \emptyset$, $A_1 \cap A_3 = \emptyset$, $A_2 \cap A_3 = \emptyset$,

$X = A_1 \cup A_2 \cup A_3$.

Поняття функції є одним з основних в математиці.

Нехай *f* – відношення з *A* до *B* таке, що

$$\forall\ a, (a, b) \in f\ \&\ (a, c) \in f \Rightarrow b = c$$

тобто елементу з множини *A* може бути тільки одна пара у відношенні *f*.

Така властивість називається *однозначністю*, а саме відношення – функцією з *A* в B позначається наступним чином:

$$b=f(a),$$

де *a* – аргумент функції; *b* – значення функції.

Область визначення функції

$$f_A = \{a \in A: \exists\ b \in B, b = f(a)\}.$$

Область значень функції

$$f_B = \{b \in B: \exists\ a \in A,\ b = f(a)\}.$$

Функція *f* називається:

− ін'єктивною, якщо $b = f(a1)\ \&\ b = f(a2) \rightarrow a1 = a2$;

− сюр'єктивною, якщо $\forall b \in B,\ \exists a \in A,\ b = f(a)$;

− бієктивною, якщо вона ін'єктивна та сюр'єктивна.

8.2. Поняття нечіткої множини

Поняття нечіткої множини є узагальненням поняття звичайної множини.

Поняття нечіткої множини виникає тоді, коли робляться два припущення:

− ступінь приналежності елемента змінюється в межах від 0 до 1;

− елемент може одночасно належати кільком множинам, але з різними показниками приналежності.

Нечіткою множиною *A* на універсальній (базовій) множині *X* називають сукупність пар елементів виду

$$A = \{\mu_A(x), x\}, x \in X,$$
$$\mu_A(x): X \rightarrow [0,1],$$

де $\mu_A(x)$ – функція приналежності.

Універсальну множину можна описати, як:

$$\mu_X(x) = 1,\ \forall\ x \in X.$$

Значення функції приналежності для конкретного *x* називають *ступенем приналежності*.

Ступінь приналежності $\mu_A(x)$ – це суб'єктивна міра того, наскільки елемент *x* відповідає поняттю, зміст якого формалізується за допомогою нечіткій множині *A*.

Значення $\mu_A(x))$ = 1 означає, що елемент *x* повністю відповідає поняттю *A*, значення $\mu_A(x)$= 0 означає, що елемент зовсім не відповідає поняттю *A*.

Значення $\mu_A(x)$ = 0,5 означає повну невизначеність. Як класичний приклад тут можна навести наполовину наповнену склянку - він є в рівному ступеню "повним" і "порожнім".

Нечітку множину можна розглядати як сукупність складових його *синглетонів* – одноточкових нечітких множин. Залежно від того, безперервна або дискретна множина *X* розглядається, застосовується одна з форм запису:

$$A = \int_X \mu_A(x)dx \quad \text{або} \quad A = \sum_{i=1}^{N} \frac{\mu_A(x_i)}{x_i}.$$

Знак інтеграла чи суми тут означає об'єднання (кон'юнкцію) синглетонів.

При безперервній базовій множині функцію приналежності зручно аналізувати графічно.

Розглянемо приклад. Нехай потрібно описати множину *A* = "Велика купа руди". Вважати окремі фракції неможливо, тому тут треба розглядати безперервну область визначення, розглядаючи, наприклад, *V*(*x*) – об'єм руди у кубометрах (див. рис. 8.4).

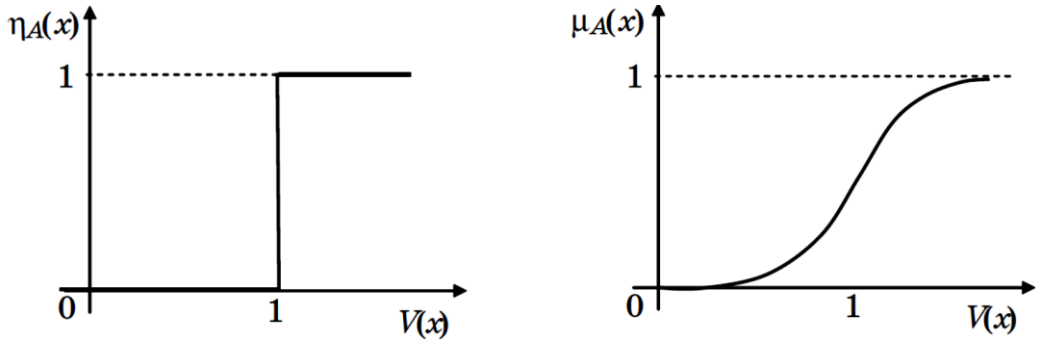


Рисунок 8.4 – Опис множини «Велика куча руди» в чіткому та нечіткому випадку

Розглянемо приклад для дискретної області визначення, що становить кількість шурупів у штуках *N*. Нехай потрібно описати множину *A* = «Багато шурупів» (див. рис. 8.5).

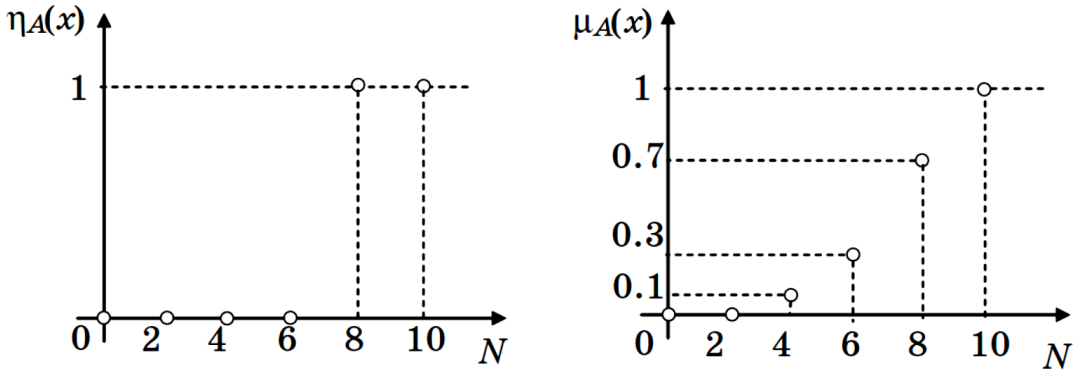


Рисунок 8.5 – Чіткий та нечіткий опис множини «Багато шурупів»

Як свідчать рис. 8.4 та 8.5, приналежність елемента множини приймає будь-які значення всередині діапазону [0, 1].

Наступний приклад показує, що той самий елемент базової множини може належати різним нечітким підмножинам. На рис. 8.6

показано дві нечіткі множини: A_1 – «Невисока температура», A_2 – «Висока температур». Тут будь-який елемент $x \in [1.5, 2]$ входить одночасно до A_1, та A_2.

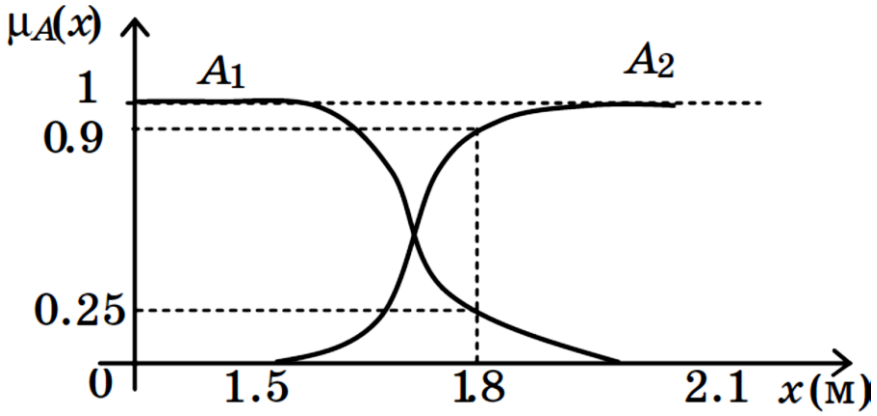


Рисунок 8.6 – Опис двох нечітких множин на одній базовій шкалі

Функція приналежності може бути охарактеризована кількома параметрами, до яких належать: носій, ядро, висота.

У ряді випадків потрібно розглядати поняття звичайної множини $\underline{A}$, найближчої до нечіткої множини A. Воно визначається за формулою:

$$\eta_{\underline{A}}(x) = \begin{cases} 1, & \mu_A(x) > 0{,}5 \\ 0, & \mu_A(x) \le 0{,}5 \end{cases}, \forall\, x \in U$$

Приклад показано на рисунку 8.7.

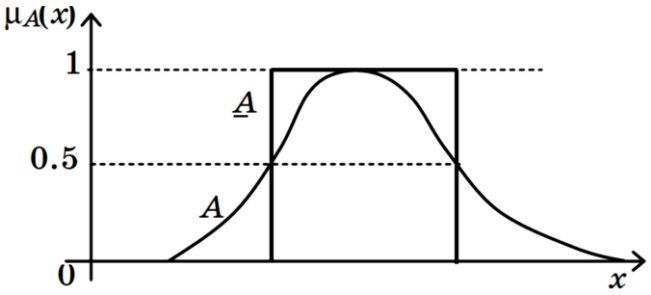


Рисунок 8.7 –Приклад звичайної множини, найближчого до нечіткого

Носієм (*support*) нечіткої множини A з функцією приналежності $\mu_A(x)$ називається звичайна множина виду:

$$\text{supp}(A) = \{x,\ x \in X, \mu_A(x) > 0\}.$$

Якщо носій нечіткої множини – кінцева множина, то таку нечітку множину називається *локальним*, інакше, коли носій – нескінченна множина, нечіткі множини – *глобальні*.

Ядром (*core, kernel* або *nucleus*) нечіткої множини *A* з функцією приналежності $\mu_A(x)$ називається звичайна множина виду:

$$core(A) = \{ x,\ x \in X, \mu_A(x) = 1\}.$$

Висота нечіткої множини (*heigth*):

$$hgt(A)=sup\ \mu_A(x),\ x \in X.$$

Ці поняття пояснює рис. 8.8.

Можна також розглянути поняття центру нечіткої множини, яким є серединою його ядра.

Дві нечіткі множини називаються *сусідніми*, якщо між їхніми центрами не розташовується центр будь-якої третьої нечіткої множини.

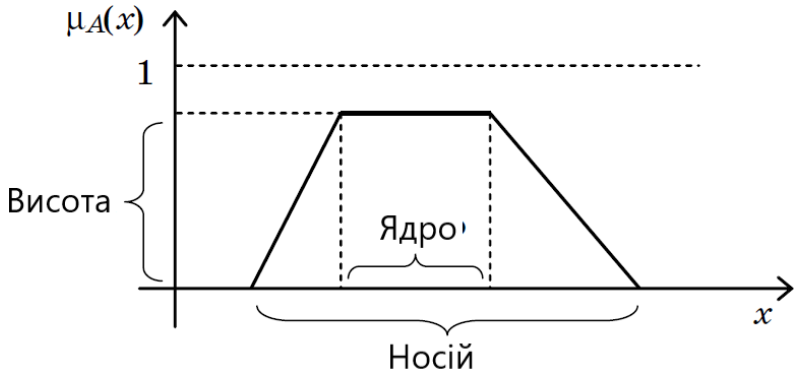


Рисунок 8.8 – Параметри функції приналежності

Нечітка множина називається *опуклою*, якщо виконується умова

$$\forall x_1, x_2, x_3 \in X, x_1 \le x_2 \le x_3 \Longrightarrow \mu_A(x_2) \ge min(\mu_A(x_1), \mu_A(x_3)).$$

Приклад показано на рис. 8.9, де *A* – опукла, а *B* – неопукла нечітка множина.

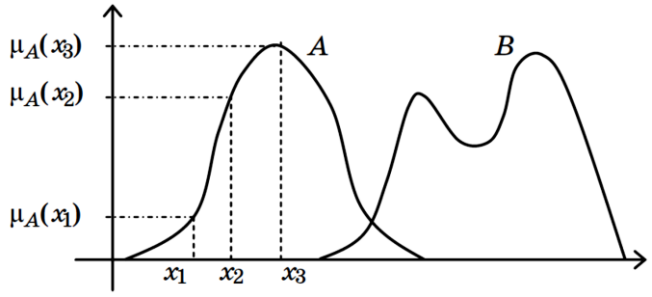


Рисунок 8.9 – Опукла та неопукла нечітка множина

Перетин двох опуклих нечітких множин є опуклою нечіткою множиною.

Нечітка множина називається *порожньою*, якщо:

$$\mu_A(x) = 0, \forall\, x \in X.$$

Вид функції приналежності характеризується кількістю точок перегину та її максимальним значенням.

Якщо $\sup\{\mu_A(x)\} = 1$, $x \in X$, то нечітка множина A називається *нормальним*, а інакше – *субнормальним*.

Якщо функція приналежності має один максимум, то нечітка множина називається *унімодальною*, інакше – *мультимодальною*. Очевидно, мультимодальна нечітка множина неопукла.

На рис. 8.10 функція приналежності $\mu_A(x)$ визначає субнормальну унімодальну нечітку множину, $\mu_B(x)$ – нормальну унімодальну та $\mu_C(x)$ – нормальну мультимодальну.

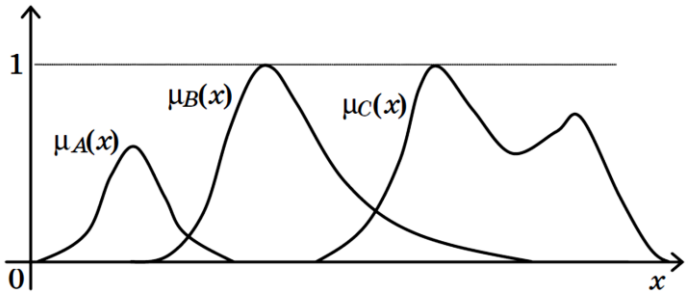


Рисунок 8.10 – Класифікація функцій приналежності

Для нечіткої множини A, визначеної на базовій множині X, можна розглядати поняття *потужності*. Воно вводиться так:

$$card(A) = \sum_{i=1}^{N} \mu_A(x_i), x_i \in X,$$

де сума сприймається не як об'єднання, а як арифметична сума. Як правило, використовуються нормальні та унімодальні нечіткі множини.

Будь-яку непусту нечітку множину можна нормалізувати за формулою

$$\mu_A^H(x) = \frac{\mu_A(x)}{\sup(\mu_A(x))}, x \in X.$$

Множиною рівня α (α-зрізом) нечіткої множини A називається *звичайна* підмножина універсальної множини X, що визначається у вигляді:

$$A_\alpha = \{ x \in X, \mu_A(x) \geq \alpha\}, \text{де } \alpha \leq 1.$$

Справедлива властивість (див. рис. 8.11):
Якщо $\alpha_1 \geq \alpha_2$, то $A_{\alpha 1} \leq A_{\alpha 2}$.

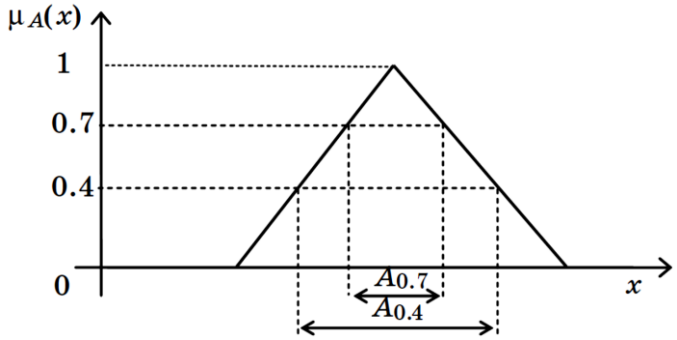


Рисунок 8.11 – Множина α-рівня

Приклад для дискретної області визначення:

$$A = \frac{0.2}{x_1} + \frac{0.6}{x_2} + \frac{1}{x_3} + \frac{0.8}{x_4}$$
$$A_{0.6} = \{x_2, x_3, x_4\}$$

Теорема про декомпозицію

Будь-яку нечітку множину *A* можна розкласти на добуток звичайних підмножин за коефіцієнтами α_i:

$$A = max_{\alpha_i}\{\alpha_1 A_{\alpha_1}, \alpha_2 A_{\alpha_2,} \dots, \alpha_n A_{\alpha_n}\};$$
$$0 < \alpha_i \leq 1, \qquad i = 1,2, \dots, n.$$

Приклад. Вихідна множина:

$$A = \frac{0.2}{x_1} + \frac{0}{x_2} + \frac{0.5}{x_3} + \frac{1}{x_4}.$$

Розкладання за трьома рівнями:

$$A = max_{\alpha_i}\left\{0{,}2\left(\frac{1}{x_1} + \frac{1}{x_3} + \frac{1}{x_4}\right), 0{,}5\left(\frac{1}{x_3} + \frac{1}{x_4}\right), 1\left(\frac{1}{x_4}\right)\right\}.$$

Використання теореми про декомпозицію у випадках виявляється зручним, т.к. дозволяє оперувати не функціями приналежності, а їх множинами рівня, що може економити обчислювальні ресурси.
Сукупність *N* опуклих нечітких множин утворює нечітке розбиття базової множини *U*, якщо виконується умова (приклад показаний на рис. 8.12):

$$\sum_{i=1}^{N} \mu_{A_i}(x) = 1;\ \forall\ x\ \in\ U.$$

Як показано в наступних розділах, цю властивість особливо зручно використовуватиме реалізації стратегії дефазифікації дискретним методом центру тяжкості.

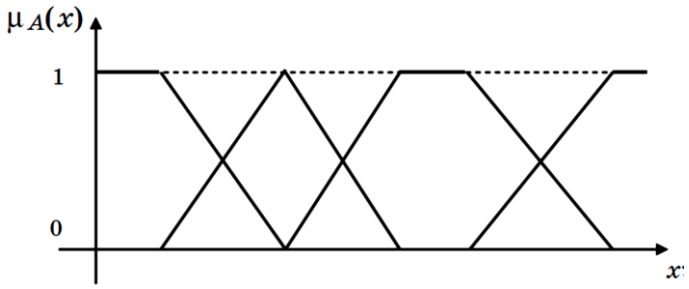


Рисунок 8.12 - Приклад нечіткого розбиття базової множини

Однією з різновидів нечітких множин є нечіткі числа.

Нечітким числом називається визначена на речовій числової нечіткої множини, що має наступні властивості:

− опуклість;

− нормальність;

− ядро нечіткої множини складається лише з однієї точки.

Нечітким інтервалом називається нечітка множина, яка виконує всі вимоги, крім останньої.

8.3 Принцип розширення

Відповідно до принципу розширення, введеного Л. Заде, звичайні математичні операції можуть бути поширені на нечіткі множини.

Нехай є відображення (чітке):

$$f: X \to Y,$$

та дана нечітка множина A, визначена на *X*:

$$A = \frac{\mu_A(x_1)}{x_1} + \frac{\mu_A(x_2)}{x_2} + \cdots + \frac{\mu_A(x_n)}{x_n}.$$

Принцип розширення свідчить, що нечітка множина *B* в області *Y*, що є відображенням *A*, визначається за формулою

$$B = f(A) = f\left(\frac{\mu_A(x_1)}{x_1} + \frac{\mu_A(x_2)}{x_2} + \cdots + \frac{\mu_A(x_n)}{x_n}\right) = \\ = \frac{\mu_A(x_1)}{f(x_1)} + \frac{\mu_A(x_2)}{f(x_2)} + \cdots + \frac{\mu_A(x_n)}{f(x_n)}.$$

Приклади:

Нехай дано A = «близько 3» $= 0{,}5/2 + 1/3 + 0{,}5/4$.

1) $B = A^2 = (\text{«близько 3»})^2 = 0{,}5/4 + 1/9 + 0{,}5/16$.

2) $y = f(x) = 2x + 1$, тоді

$$B = f(A) = \frac{0.5}{5} + \frac{1}{7} + \frac{0.5}{9}.$$

Функція *f* може відображати різні елементи множини *X* в один самий елемент множини *Y*. В результаті одному і тому ж елементу можуть відповідати різні функції приналежності, операція *sup* дозволяє обрати найбільше з цих значень.

$$B = f(A) = \{y, \mu_B(y)\}, y = f(x), x \in X.$$

$$\text{де } \mu_B(y) = \begin{cases} \sup\limits_{x \in f^{-1}(y)} \mu_A(x), \text{якщо } f^{-1}(y) \neq 0, \\ 0, \qquad\qquad \text{якщо } f^{-1}(y) = 0. \end{cases}$$

Наприклад:

$$A = \frac{0.3}{-2} + \frac{0.4}{-1} + \frac{0.8}{0} + \frac{1}{1} + \frac{0.7}{2}.$$

$$(A)^2 = \frac{0.3}{4} + \frac{0.4}{1} + \frac{0.8}{0} + + \frac{1}{1} + \frac{0.7}{4} = \frac{0.8}{0} + \frac{1}{1} + \frac{0.7}{4}.$$

Пояснює цей результат рис. 8.13.

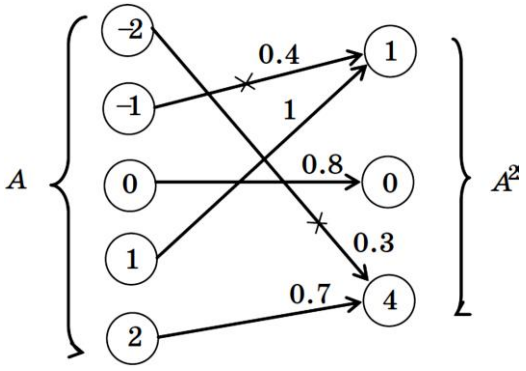


Рисунок 8.13 – Виконання операції возведення нечіткого числа в квадрат

Нехай A_1, A_2, ... A_n - нечіткі множини, визначені на базових множинах X_1, X_2, ... X_n і задано

$$f: X_1 \times X_2 \times \ldots \times X_n \to Y,$$

так що $y = f(x_1, x_2, \ldots, x_n)$.

Тоді принцип розширення дозволяє визначити нечітку множину B, визначену на Y:

$$B = f(A_1, A_2, \ldots, A_n) = \{(y, \mu_B(y))\},$$

$$y = f(x_1, x_2, \ldots, x_n), x_1 \in X_1, x_2 \in X_2, \ldots, x_n \in X_n$$

де

$$\mu_B(y) = \begin{cases} \sup\limits_{x_1, x_2, \ldots, x_n \in f^{-1}(y)} (\mu_{A_1}(x_1), \mu_{A_2}(x_2), \ldots, \mu_{A_n}(x_n)), \text{якщо } f^{-1}(y) \neq 0, \\ 0, \qquad \text{якщо } f^{-1}(y) = 0. \end{cases}$$

При виконанні бінарних операцій формула може бути записана у вигляді:

$$\mu_B(y) = \sup_{y = x_1 + x_2} min\{\mu_{A_1}(x_1), \mu_{A_2}(x_2)\},$$

або

$$\mu_B(y) = \bigcup_{X_1 \times X_2} \left(\mu_{A_1}(x_1) \wedge \mu_{A_2}(x_2)\right).$$

Приклад.

$$A = \frac{0.5}{2} + \frac{0.9}{3} + \frac{0.6}{4};$$
$$B = \frac{0.5}{5} + \frac{1}{6} + \frac{0.6}{7};$$

$$A + B = \frac{0.5}{7} + \frac{0.5}{8} + \frac{0.5}{9} + \frac{0.5}{8} + \frac{0.9}{9} + \frac{0.6}{10} + \frac{0.5}{9} + \frac{0.6}{10} + \frac{0.6}{11} =$$
$$= \frac{0.5}{7} + \frac{0.5}{8} + \frac{0.9}{9} + \frac{0.6}{10} + \frac{0.6}{11}.$$

8.4 Способи побудови функцій приналежності

За визначенням ступінь приналежності $\mu_A(x)$ є суб'єктивна міра відповідності елемента x поняттю, зміст якого формалізується нечіткою

множиною *A*, тому для побудови функції приналежності досить думки одного експерта.

Однак, якщо є множина експертів, які успішно діють у предметній галузі, у процедурі призначення функцій приналежності можна врахувати кілька думок. Кожен експерт отримує коефіцієнт $a_i \in [0, 1]$, який відображає рівень його компетентності, так що для групи з *m* експертів:

$$\mu_A(x) = \frac{1}{m}\sum_{i=1}^{m} p_i a_i,$$

де $p_i = 1$, якщо *i*-й експерт відніс *x* до *A*, або $p_i = 0$ – в противному випадку.

Приклад. Нехай є 6 експертів однаково високої кваліфікації ($a_i = 1$), є множина $X = \{1,2,3,4,5\}$ і необхідно побудувати нечітку множину *A*, що формалізує поняття "*набагато менше* 5".

Результати опитування експертів показано у табл. 8.2.

Таблиця 8.2 - Розрахунок значень приналежності

	x				
	1	2	3	4	5
$\sum_{i=1}^{6} p_i a_i$	6	5	1	0	0
$\mu_A(x_i)$	6/6	5/6	1/6	0/6	0/6

Якщо потрібно отримати узгоджений опис функції приналежності для деякої кількості об'єктів універсальної множини, застосовується *метод кількісного парного порівняння ступенів приналежності*. У цьому методі результатом експертного опитування є матриця

$$M = [m_{ij}],\ i, j = 1,2, \ldots n,$$

де *n* – число точок, у яких порівнюються функції власності.

Кожен елемент матриці m_{ij} показує, у скільки разів $\mu_A(x_i)$ більше $\mu_A(x_j)$. Кількість питань до експерта становить тут не n^2, а лише величину $(n^2 - n)/2$, оскільки $m_{ij} = m_{ji}$ та $m_{ij} = 1/ m_{ji}$. Потім із кожного стовпця матриці визначається одне значення за формулою

$$\mu_A(x_i) = \frac{m_{ij}}{\sum_{i=1,n} m_{ij}},$$

де *i* – рядок матриці, а *j* – стовпець (він може вибиратися довільно

– це мало впливає на результат).

При порівнянні необхідно використовувати кількісні оцінки. Для цього можна, наприклад, оперувати табл. 8.3.

Таблиця 8.3 – Порівняння значень приналежності

Якісна оцінка	Кількісна оцінка
$\mu_A(x_i)$ дорівнює $\mu_A(x_j)$	1
$\mu_A(x_i)$ трохи більше $\mu_A(x_j)$	3
$\mu_A(x_i)$ більше $\mu_A(x_j)$	5
$\mu_A(x_i)$ помітно більше $\mu_A(x_j)$	7
$\mu_A(x_i)$ набагато більше $\mu_A(x_j)$	9
Проміжне значення	2, 4, 6, 8

Розглянемо приклад. Нехай задано базова множина *X* = {40, 55, 70, 100} і необхідно визначити на цій множині нечітку множину «Мала вага». Нехай у результаті опитування експертів є матриця порівнянь:

$$M = \begin{bmatrix} 1 & 5 & 7 & 9 \\ 1/5 & 1 & 4 & 7 \\ 1/7 & 1/4 & 1 & 4 \\ 1/9 & 1/7 & 1/4 & 1 \end{bmatrix}.$$

Наприклад, якщо зафіксувати перший стовпець, то виходить

$$\mu_A(40) = \frac{1}{\sum_{i=1}^{4} \mu_{i1}} = \frac{1}{1.45} = 0.69; \quad \mu_A(55) = \frac{0.2}{\sum_{i=1}^{4} \mu_{i1}} = \frac{0.2}{1.45} = 0.14;$$

$$\mu_A(70) = \frac{0.14}{\sum_{i=1}^{4} \mu_{i1}} = \frac{0.14}{1.45} = 0.1; \quad \mu_A(100) = \frac{0.11}{\sum_{i=1}^{4} \mu_{i1}} = \frac{0.11}{1.45} = 0.08.$$

Якщо зафіксувати другий стовпець, то

$$\mu_A(40) = 0{,}81; \ \mu_A(55) = 0{,}15; \ \mu_A(70) = 0{,}04; \ \mu_A(100) = 0{,}02.$$

Таким чином, можна використати опис:

$$A = \frac{0.69}{40} + \frac{0.14}{55} + \frac{0.1}{70} + \frac{0.08}{100},$$

або близький опис

$$A = \frac{0.81}{40} + \frac{0.15}{55} + \frac{0.04}{70} + \frac{0.02}{100},$$

Метод попарних порівнянь вигідно використовувати, коли розглядаються безмірні, тобто якісні поняття.

8.5 Нечіткість та інші види невизначеності

Людина стикається з двома видами невизначеності - фізіологічної та лінгвістичної (рис. 8.14).

Фізична неточність вивчається теорією вимірів. Невизначеність сенсу фраз вивчається теорією формальних граматик.

Омонімія, як і полісемія, – тип семантичних відносин, який встановлюється між словами, значення яких абсолютно пов'язані друг з одним, але формально ці значення виражаються подібними звуковими оболонками.

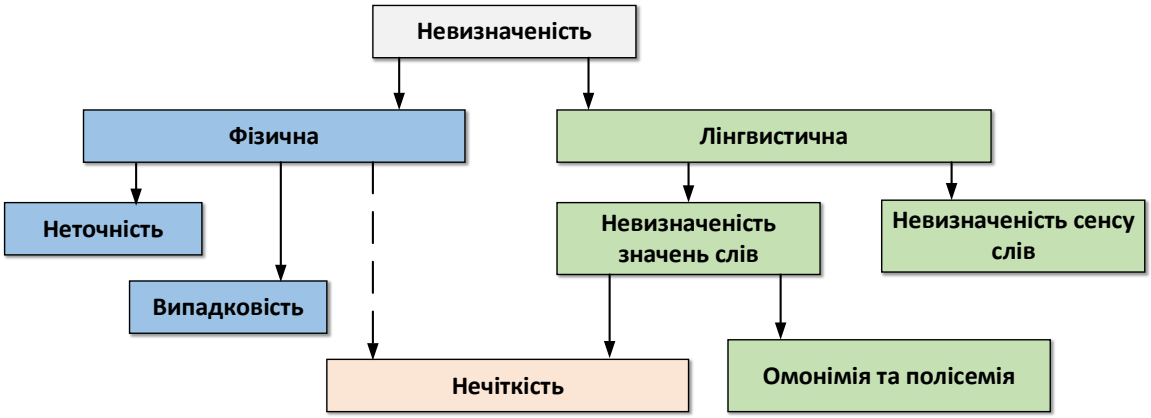


Рисунок 8.14 – Нечіткість та інші види невизначеності

Розглянемо співвідношенні фізичної невизначеності як випадковості і невизначеності як нечіткості.

Нечіткість розуміється (у класичному трактуванні) як лінгвістична невизначеність значень слів. З іншого боку, будь-яка фізична величина, що вимірюється приладом або безпосередньо сприймається, несе певне семантичне навантаження. Поєднання кількох «точних» вимірів часто породжує невизначеність через неможливість віднесення цієї комбінації до чітко визначеного класу. Тож на рис. 8.14 показано, що нечіткість стосується і фізичної невизначеності.

Поняття ймовірності та нечіткості мають спільні риси (табл. 8.4).

Однак застосування теорії ймовірності вимагає виконання низки умов, таких як:

− повторюваність подій;

− незалежність подій;

− спільність подій, що спостерігаються, для всіх явищ подібного роду.

Жорсткість цих вимог дозволяє навіть висловити крайню точку зору, відповідно до якої теорія ймовірності найбільш адекватна тій

галузі діяльності, для опису якої вона і була розроблена – грі в кістки. Проте функція $\mu_A(x)$ може трактуватися як умовна ймовірність спостереження події A при спостереженні x, тому статистичні дані можуть використовуватися при побудові функції приналежності.

Таблиця 8.4 – Загальні риси в опису вірогідності та нечіткості

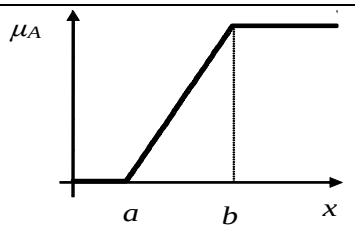
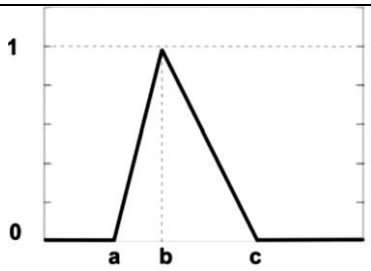
Вірогідність	Ступень приналежності
$0 \le P(A) \le 1$	$0 \le \mu_A(x) \le 1$
$P(\overline{A}) = 1 - P(A)$	$\mu_{\overline{A}}(x) = 1 - \mu_A(x)$
$P(A) = N(A)/N$ де N – загальна кількість подій, $N(A)$ – число наслідків, що приводять до A	$\mu_A(x) = n/m$ де m – загальна кількість експертів, яких n вважає, що x належить до A
Ймовірність несумісних подій $\sum_{i=1}^{N} P(A_i) = 1$	Нечітке розбиття базової множини $\sum_{i=1}^{N} \mu_{A_i}(x) = 1$

8.6 Аналітичний опис функцій приналежності

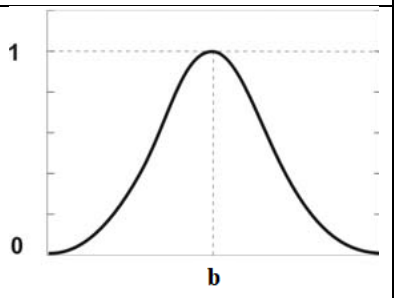
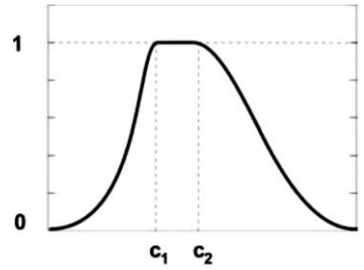
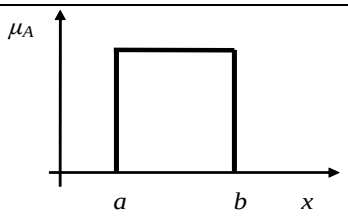
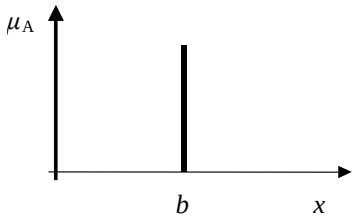
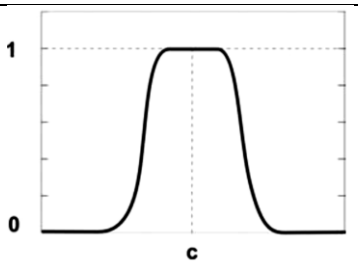
Насправді бажано мати аналітичний опис функції приналежності (ФП), щоб обчислювати ступінь належності для довільного значення області визначення.

Для опису ФП використовують варіанти, наведені в табл. 8.5.

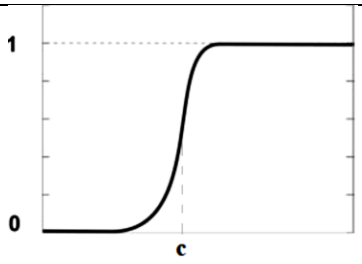
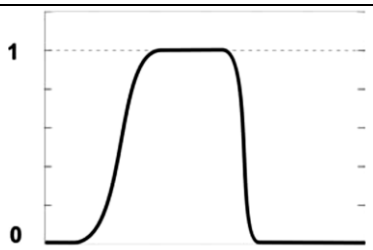
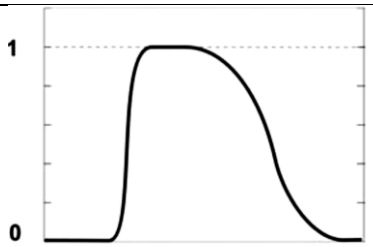
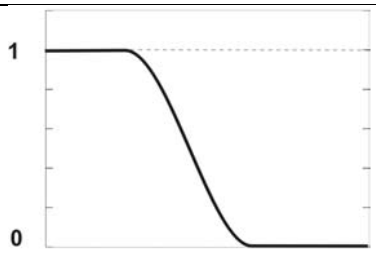
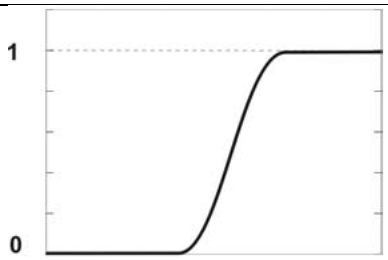
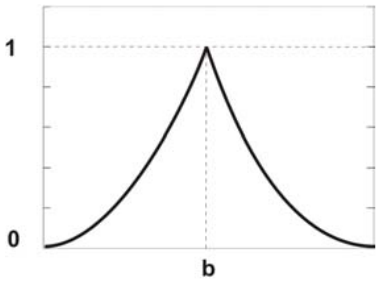
Таблиця 8.5 – Функції приналежності

Тип функції	Математичне визначення	Графічне зображення
Монотонна (лінійна)	$\mu(x) = \begin{cases} 0 & \text{при} \quad x < a \\ \dfrac{x-a}{b-a} & \text{при} \quad a \le x \le b \\ 1 & \text{при} \quad b \le x \end{cases}$	
Трикутна	$\mu(x) = \begin{cases} 0 \quad \text{при} \quad x \le a \text{ або } x \ge c \\ \dfrac{x-a}{b-a} \quad \text{при} \quad a \le x \le b \\ \dfrac{c-x}{c-d} \quad \text{при} \quad b \le x \le c \end{cases}$ де $a \le b \le c$	

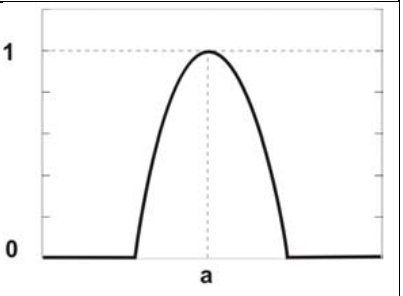
Продовження табл. 8.5

Тип функції	Математичне визначення	Графічне зображення
Симетрична функція Гауса	$\mu(x) = e^{-\frac{(x-b)^2}{2c^2}}, \quad b > 0$	
Двобічна функція Гауса	якщо $c_1 < c_2$, тоді $\mu(x) = \begin{cases} e^{-\frac{(x-c_1)^2}{2a_1^2}}, & \text{при} \quad x < c_1, \\ 1, & \text{при} \quad c_1 < x \le c_2, \\ e^{-\frac{(x-c_2)^2}{2a_2^2}}, & \text{при} \quad x > c_2. \end{cases}$ якщо $c_1 > c_2$, тоді $\mu(x) = \begin{cases} e^{-\frac{(x-c_1)^2}{2a_1^2}}, & \text{при} \quad x < c_2, \\ e^{-\frac{(x-c_1)^2}{2a_1^2}} \cdot e^{-\frac{(x-c_2)^2}{2a_2^2}} & \text{при } c_2 < x \le c_1, \\ e^{-\frac{(x-c_2)^2}{2a_2^2}}, & \text{при} \quad x > c_1. \end{cases}$	
Прямокутна	$\mu(x) = \begin{cases} 0, \text{при } x < a \text{ або } b < x, \\ 1, \quad \text{при } a \le x \le b. \end{cases}$	
Синглетон-функція	$\mu(x) = \begin{cases} 0 & \text{при} \quad x = b, \\ 1 & \text{при} \quad x \ne b. \end{cases}$	
Узагальнена дзвоно-подібна функція	$\mu(x) = \frac{1}{1 + \left\|\frac{x-c}{a}\right\|^{2b}}$ де $a \in (0; +\infty); b \in (-\infty; +\infty)$ $c \in (-\infty; +\infty)$	

Продовження табл. 8.5

Тип функції	Математичне визначення	Графічне зображення
Сигмаїдна функція	$\mu(x) = \frac{1}{1 + e^{-a(x-c)}}$	
Добуток сигмоїдних функцій	$\mu(x) = \frac{1}{1 + e^{-a_1(x-c_1)}} \cdot \frac{1}{1 + e^{-a_2(x-c_2)}}$	
Різниця між сигмоїдними функціями	$\mu(x) = \frac{1}{1 + e^{-a_1(x-c_1)}} - \frac{1}{1 + e^{-a_2(x-c_2)}}$	
Z-подібна функція	$\mu(x) = \begin{cases} 1, & \text{при } x \le a \\ \text{нелінійна, апроксимація,} & \text{при } a < x \le \\ 0, & \text{при } x \ge b. \end{cases}$	
S-подібна функція	$\mu(x) = \begin{cases} 1, & \text{при } x \le a \\ \text{нелінійна, апроксимація,} & \text{при } a < x \le \\ 0, & \text{при } x \ge b. \end{cases}$	
Лапласівська функція	$\mu(x) = e^{-\frac{\lvert x-b \rvert}{d}}$ де $d > 0$	0 

Продовження табл. 8.5

Тип функції	Математичне визначення	Графічне зображення
Квадратична функція	$\mu(x) = \\ = \begin{cases} 1 - \left(\frac{x-a}{b}\right)^2, \text{якщо } \left(\frac{x-a}{b}\right)^2 < 1 \\ 0, \text{інакше} \end{cases}$	

Функція приналежності – це функція, яка дозволяє обчислити ступінь належності будь- якого елемента до нечіткої множини.

8.7 Операції над нечіткими множинами

Над нечіткими множинами виконуються самі операції, як і з звичайними множинами, і навіть ряд спеціальних операцій. При цьому існують альтернативні варіанти опису операцій над нечіткими множинами.

Нехай *A*, *B* і *C* – нечіткі підмножини базової множини *X*. У табл. 8.6 показані основні операції над нечіткими множинами.

Таблиця 8.6 – Основні операції над нечіткими множинами

Операція	Основні варіанти реалізації
Об'єднання $C = A \cup B$	$\mu_C(x) = \max(\mu_A(x), \mu_B(x))$
	$\mu_C(x) = \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x)$
Перетин $C = A \cap B$	$\mu_C(x) = \min(\mu_A(x), \mu_B(x))$
	$\mu_C(x) = \mu_A(x)\mu_B(x)$
Доповнення $C = \overline{A}$	$\mu_C(x) = 1 - \mu_A(x)$
	$\mu_C(x) = \sqrt{1 - (\mu_A(x))^2}$
	$\mu_C(x) = \frac{1 - \mu_A(x)}{1 - k\mu_A(x)}; \\ -1 < k < \infty$
Різність $C = A - B$	$\mu_C(x) = \min(\mu_A(x), 1 - \mu_B(x))$
Симетрична різність $C = A \triangle B$	$\mu_C(x) = max\{\min(\mu_A(x), 1 - \mu_B(x)), \min(1 - \mu_A(x), \mu_B(x))\}$
Алгебраїчний добуток $C = A \cdot B$	$\mu_C(x) = \mu_A(x)\mu_B(x)$
Алгебраїчна сума $C = A + B$	$\mu_C(x) = \mu_A(x) + \mu_B(x) - \mu_A(x)\mu_B(x)$

Продовження табл. 8.6

Операція	Основні варіанти реалізації
Стиснення $C = con(A)$	$\mu_C(x) = \int_X \frac{\left(\mu_A(x)\right)^2}{x}$
Розтягування $C = dil(A)$	$\mu_C(x) = \int_X \frac{\left(\mu_A(x)\right)^{0.5}}{x}$

Для нечітких множин закони протиріччя та виключеного третього не виконуються. Нехай *X* – базова (чітка) множина, і *A* – його нечітка підмножина, тоді

$$\overline{A} \cup A \neq X \text{ та } \overline{A} \cap A \neq 0$$

Покажемо це на прикладі:

$$A = 0.81/40 + 0.15/55 + 0.04/70 + 0.02/100;$$
$$\overline{A} = 0.19/40 + 0.85/55 + 0.96/70 + 0.98/100;$$

$$A \cap \overline{A} = 0.19/40 + 0.15/55 + 0.04/70 + 0.02/100 \neq \emptyset;$$
$$A \cap \overline{A} = 0.81/40 + 0.85/55 + 0.96/70 + 0.98/100 \neq U.$$

За реалізації об'єднання з допомогою функції *max*, а перетину – з допомогою функції *min* справедливо:

$$card(A) + card(B) = card(A \cup B) + card(A \cap B);$$
$$card(A) + card(\overline{A}) = card(X).$$

Операції рівності та включення визначаються для нечітких множин наступним чином:

$$A = B \text{ якщо } \mu_A(x) = \mu_B(x);$$
$$A \subset B, \text{якщо } \mu_A(x) \leq \mu_B(x)).$$

Операцію включення $A \subset B$ можна як ставлення домінування *B* над *A*.

Розглянемо приклади виконання операцій для нечіткими множинами, заданих на дискретній ділянці визначення.

Нехай дана базова множина *X* і визначено дві нечіткі множини *A* та *B*:

$$X = \{x_1, x_2, x_3, x_4, x_5\};$$

$$A = \{0.1/x_1, 0.2/x_2, 0.3/x_3, 0.4/x_4, 0.5/x_5\};$$
$$\text{B} = \{1/x_1, 0/x_2, 0.8/x_3, 0.6/x_4, 0.1/x_5\}.$$

При використанні операторів *max* та *min*:

$$C = A \cup B = \{1/x_1, 0.2/x_2, 0.8/x_3, 0.6/x_4, 0.5/x_5\};$$

$$\text{C} = A \cap B = \{0.1/x_1, 0/x_2, 0.3/x_3, 0.4/x_4, 0.1/x_5\}.$$

При використанні операторів обмежена сума та добуток:

$$C = A \cup B = \{1/x_1, 0.2/x_2, 0.86/x_3, 0.76/x_4, 0.55/x_5\};$$

$$\text{C} = A \cap B = \{0.1/x_1, 0/x_2, 0.24/x_3, 0.24/x_4, 0.05/x_5\}.$$

Таким чином, вибір варіанта реалізації операцій об'єднання та перетину нечітких множин надає помітний вплив на результат операції.

8.8 Трикутна норма та конорма

З метою оптимального вибору варіанта реалізації операцій над нечіткими множинами в теорії нечітких множин розглядаються узагальнені операції об'єднання і перетину, які називаються *T*-норма (від triangular norm) та *T*-конорма (*S*-норма).

T-норма узагальнює поняття перетину НМ та є бінарним відображенням

$$T: [0,1] \times [0,1] \to [0,1],$$

для якого мають виконуватися такі властивості наведені в табл. 8.7.

Таблиця 8.7 - Властивості Т-норми

$T(a,b) = T(b,a)$	Комутативність
$T\big(T(a,b)\big) = T(a,T(b,c))$	Асоціативність
$T(a,b) \geq T(c,d)$, якщо $a \geq c$ та $b \geq d$	Монотонність
$T(a,1) = a$	Гранична умова

Варіанти різних Т-норм наведені у таблиці 8.8.

Для Т-норм справедливо обмеження:

$$T_W(a,b) \leq T(a,b) \leq \min(a,b)$$

Це обмеження випливає з властивостей монотонності, комутативності та граничної умови:

$$T(a,b) \le T(a,1) \le a;$$
$$T(a,b) = T(b,a) \le T(1,b) \le b.$$

Таблиця 8.8 – Основні варіанти Т-норм

Назва	Опис
Т-норма Заде	$T(a,b) = \min(a,b)$
Добуток	$T(a,b) = ab$
Квазілінійна	$T(a,b) = \max(0, a+b-1)$
Т-норма Вебера або драстичний добуток (drastic product)	$T_W(a,b) = \begin{cases} \min(a,b), & \text{якщо } \max(a,b) = 1 \\ 0, & \text{інакше} \end{cases}$
Параметричне сімейство Т-норм	$T_p(a,b) = 1 - \min\left(1, ((1-a)^p + (1-b)^p)^{\frac{1}{p}}\right)$ $p \ge 0$

Звідки $T(a,b) \le \min(a,b)$, крім цього

$$T_W(a,b) \le T(a,b), \text{ якщо } a \ne 1 \text{ або } b \ne 1;$$
$$T_W(a,b) \le T(1,b), \text{ та } T_W(a,b) \le T(a,1).$$

Тому наведене обмеження є справедливим.

S-норма (Т-конорма) узагальнює поняття об'єднання нечіткої множини.

S-норма також є відображенням

$$S\colon [0,1] \times [0,1] \to [0,1].$$

Для S-норми також виконуються описані вище властивості комутативності, асоціативності та монотонності, але гранична умова записується інакше: $S(a,0) = a$.

Для опису S-норми використовуються зазвичай варіанти, наведені в таблиці 8.9.

Таблиця 8.9 – Основні варіанти S-норм

Назва	Опис
S-норма Заде	$S(a,b) = \min(a,b)$
Обмежена сума	$S(a,b) = a + b - ab$
Квазілінійна	$S(a,b) = \min(1, a+b)$
S-норма Вебера або драстичний добуток (drastic product)	$S_W(a,b) = \begin{cases} \max(a,b), & \text{якщо } \min(a,b) = 0 \\ 1, & \text{інакше} \end{cases}$
Параметричне сімейство S-норм	$S_p(a,b) = \min\left(1, ((a)^p + (b)^p)^{\frac{1}{p}}\right)$ $p \ge 0$

Для S-норм справедливо обмеження:

$$max(a,b) \le S(a,b) \le S_w(a,b)$$

T-норма та S-норма дуальні

$$T(a,b) = \overline{S(\overline{a},\overline{b})}$$

Кожна з T-норм та S-норм може бути записана для більшій кількості елементів.

Наприклад, якщо для двох аргументів

$$T(a,b) = \max(0, a+b-1)$$

тоді для *n* аргументів

$$T(a_1,a_2,\dots,a_n) = \max\left\{0, \sum_{i=1}^{n} a_i - n + 1\right\}.$$

8.9 Заходи подібності нечітких множин

Для двох нечітких множин *A* і *B*, що мають однакові базові множини з *N* елементів, може бути обчислена відстань Хеммінга за формулою

$$\rho(A,B) = \sum_{i=1}^{N} |\mu_A(x_i) - \mu_B(x_i)|$$

Можна також розглядати Евклідову (квадратичну) відстань

$$\varepsilon(A,B) = \sqrt{\sum_{i=1}^{N} (\mu_A(x_i) - \mu_B(x_i))^2}$$

Наприклад

$$A = 0.3/-2 + 0.4/-1 + 0.8/0 + 1/1 + 0.7/2;$$
$$B = 0.1/-2 + 0.8/-1 + 1/0 + 0.7/1 + 0.4/2;$$
$$\rho(A,B) = 1.4;\ \varepsilon(A,B) = 0.69.$$

Відносна відстань Хеммінга та відносна Евклідова відстань виходять за формулами:

$$\acute{\rho}(A,B) = \frac{1}{N}\sum_{i=1}^{N}|\mu_A(x_i) - \mu_B(x_i)|$$

$$\acute{\varepsilon}(A,B) = \frac{1}{N}\sqrt{\sum_{i=1}^{N}(\mu_A(x_i) - \mu_B(x_i))^2}$$

Очевидно, $\rho(A,B)$, $\varepsilon(A,B) \in [0, 1]$.

На підставі однієї з функцій відстані та введеного поняття звичайної множини, найближчої до нечіткої, можна ввести в розгляд величину, яка називається *індексом нечіткості нечіткої множини*.

Лінійний індекс нечіткості:

$$d_p(A) = \frac{2}{N}\rho\left(A, \underline{A}\right).$$

Квадратичний індекс нечіткості:

$$d_\varepsilon(A) = \frac{2}{\sqrt{N}}\rho\left(A, \underline{A}\right).$$

Наприклад

$$A = 0.3/-2 + 0.4/-1 + 0.8/0 + 1/1 + 0.7/2;$$
$$\underline{A} = 0/-2 + 0/-1 + 1/0 + 1/1 + 1/2;$$

$$\rho\left(A, \underline{A}\right) = \sum_{i=1}^{5}|\mu_A(x_i) - \mu_B(x_i)| = 1{,}2, d_p(A) = \frac{2}{5}\rho\left(A, \underline{A}\right) = 0{,}48;$$

$$\varepsilon\left(A, \underline{A}\right) = \sqrt{\sum_{i=1}^{N}(\mu_A(x_i) - \mu_B(x_i))^2} = 0{,}62, d_\varepsilon(A) = \frac{2}{\sqrt{5}}\rho\left(A, \underline{A}\right) = 0{,}55$$

Відстань Хеммінга та квадратична відстань не цілком придатні для оцінювання подібності нечіткої множин, оскільки вони «усереднюють» оцінку близькості, тоді як при оцінюванні особливо важливо розглядати екстремальні значення функції приналежності. Тому як заходи подібності нечіткої множин зазвичай використовуються інші *методи: міра подібності Заде, міра подібності Лукасевича, і навіть міра подібності площини*.

Міра подібності Заде між нечіткими множинами *A* і *B* описується формулою

$$Z(A,B) = \min\left(\max\left(1-\mu_A(x_i), \mu_B(x_i)\right), \max\left(\mu_A(x_i), 1-\mu_B(x_i)\right)\right),$$
$$i = \overline{1,N}.$$

Міра подібності Лукасевича розраховується за формулою

$$L(A,B) = 1 - \max|\mu_A(x_i) - \mu_B(x_i)|, i = \overline{1,N}.$$

Міра подібності по площини може бути розрахована по формулі

$$P(A,B) = 1 - \frac{\sum_{i=1}^{N}|\mu_A(x_i) - \mu_B(x_i)|}{\sum_{i=1}^{N}\mu_A(x_i) - \mu_B(x_i)}.$$

Розглянемо приклад

$$A = 0.3/-2 + 0.4/-1 + 0.8/0 + 1/1 + 0.7/2;$$
$$B = 0.1/-2 + 0.8/-1 + 1/0 + 0.7/1 + 0.4/2;$$
$$Z(A,B) = 0{,}4.$$
$$L(A,B) = 0{,}6.$$
$$P(A,B) = 0{,}77.$$

8.10 Нечіткі відношення та нечітка композиція

Вище було розглянуто основні операції над одновимірними нечіткими множинами. Можуть бути також використані багатовимірні нечіткі множини. Вони виходять під час розгляду нечітких відносин. Нечітке відношення *R* двох чітких множин *X* і *Y* є нечіткою підмножиною декартового твору *X* і *Y*.

$$R = \{\mu_R(x,y)/(x/y); x \in X, y \in Y\}$$

Для *n* множин нечітке відношення описується формулою:

$$R = \{\mu_R(x_1, x_2, \dots, x_n)/(x/y); x_1 \in X, x_2 \in X, \dots, x_n \in X\}$$

Для безперервних областей для визначення часто використовується запис:

$$R = \int_{X\times Y} \mu_R(x,y)/(x/y).$$

Функція приналежності тут є деякою поверхнею в тривимірному просторі. Якщо у нечіткому відношенні беруть участь більше двох множин, то функція приналежності перетворюється на гіперповерхню.

Розглянемо приклад:

$$x, y \in R^1; c = const;$$
$$R(x,y) = \begin{cases} 1, x^2 + y^2 \le c^2 \\ exp\left(\frac{x^2 + y^2}{0{,}5}\right), \text{інакше.} \end{cases}$$

Це відношення описує «нечітке коло» із розмитими кордонами. Для дискретних областей визначення:

$$R(x,y) = \sum_X \sum_Y \mu_R(x,y)/(x/y).$$

Розглянемо приклад. Нехай є дві множини: *X* - "Автомобілі", *Y* - "Ціна (у.о.)".

$$X = \{\text{Мерседес}, \text{Ланос}, \text{Бентлі}\},$$
$$Y = \{6000, 10000, 30000, 100000, 500000, 2000000\}.$$

За допомогою вказівки ступеня належності (значення в колі на рис. 8.15) можна описати на X × Y відношення R – «Прийнятна вартість».

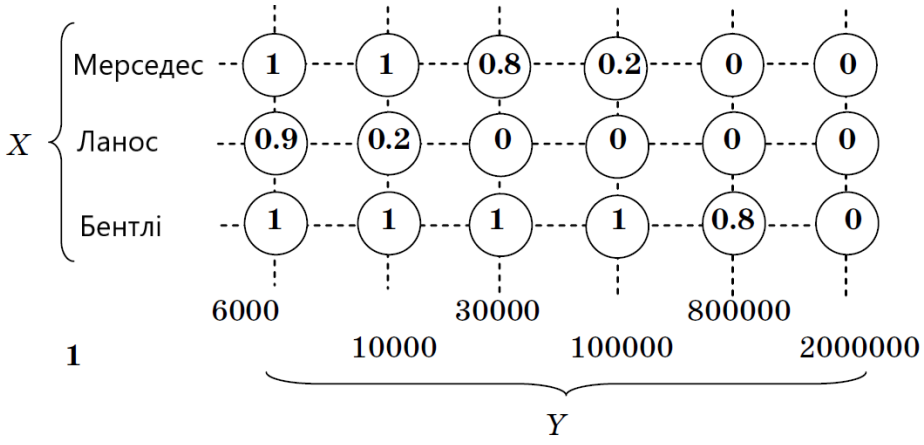


Рисунок 8.15 - Приклад опису нечіткого відношення «Прийнятна ціна»

Як свідчить рис. 8.15, нечітке відношення зручно описувати матрицею

$$R = \begin{vmatrix} 1 & 1 & 0{,}8 & 0{,}2 & 0 & 0 \\ 0{,}9 & 0{,}2 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0{,}8 & 0 \end{vmatrix}$$

Тривимірне нечітке відношення можна отримати, наприклад, якщо розглядати три нечіткі множини: *X* - "Автомобілі", *Y* - "Ціна (у.о.)", *Z* - "Рік

випуску". Відповідно можна точніше побудувати нечітке ставлення *R* – «Прийнятна ціна»:

R = 1/(Ланос, 3000, 2005) + 0.7/(Бентлі, 400000,1999) + …

Нечітке відношення є нечіткою множиною, тому йому справедливі всі поняття та операції, описані для звичайних нечітких множин.

Проте є й специфічні операції, запроваджувані лише для нечітких відносин. До них відносяться операція проекції та циліндричного розширення, а також операція композиції. Якщо поставлено бінарне нечітке відношення

$$R: (x, y) \to [0,1],$$

то операція проекції може бути записана у вигляді

$$B = proj(R; Y) = \int_Y \sup_y \mu_R(x, y)/y.$$

Приклад для безперервної області визначення показаний на рис. 8.16.

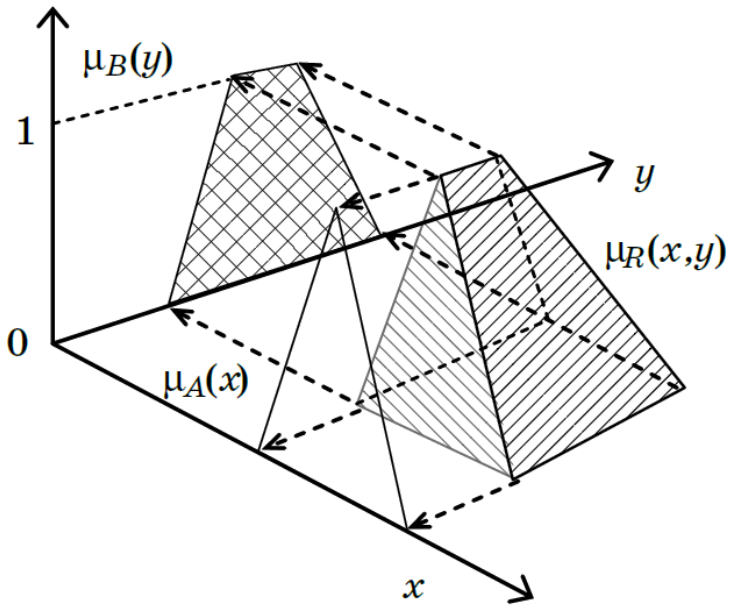


Рисунок 8.16 – Проекція бінарного нечіткого відношення

Очевидно, бінарне нечітке відношення може мати дві проекції.

Першою проекцією бінарного нечіткого відношення (проекцією на *X*) називається нечітка множина, заданий на X з функцією приналежності:

$$\mu_{R1}(x) = \bigcup_{y} \mu_R(x, y).$$

Аналогічно, другою проекцією (проекцією на *Y*) називається нечітка множина, задана на *Y* з функцією приналежності:

$$\mu_{R1}(y) = \bigcup_{x} \mu_R(x, y).$$

Глобальною проекцією бінарного нечіткого відношення *R* називається величина:

$$h(R) = \bigcup_{x} \mu_{R1}(x, y) = \bigcup_{y} \mu_{R1}(x, y).$$

Якщо *h*(*R*)=1, то ставлення нормальне. Приклад для дискретних множин *X* та *Y* показаний на рис. 8.17.

$R=$

	y_1	y_2	y_3	y_4	
x_1	0.2	0.4	0	0.9	0.9
x_2	0.6	0.3	1	0.1	1
x_3	0.2	0.8	0.7	0	0.8

$=R_1$

$R_2=$

0.6	0.8	1	0.9	1

$=h(R)$

Рисунок 8.17 – Проекції бінарного нечіткого відношення

Циліндричне розширення нечіткого відношення R є зворотною операцією щодо проекції, і описується формулою

$$R = cext(A; X \times Y) = \int_{X \times Y} \mu_A(x)/(x, y).$$

Виконання циліндричної проекції ілюструє рис. 8.18. Розглянемо приклад для дискретної області визначення.

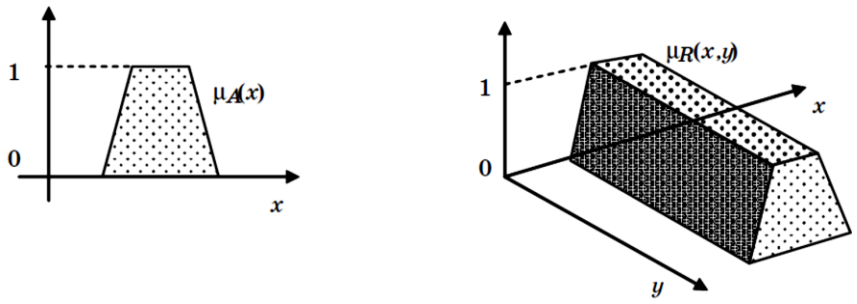


Рисунок 8.18 - Циліндричне розширення нечіткої множини

Нехай дані дві базові множини:

$$X = \{x_1, x_2, x_3\}, \qquad Y = \{y_1, y_2, y_3, y_4\}$$

та дана нечітка множина *A* на *X*

$$A = \frac{0.2}{x_1} + \frac{0.6}{x_2} + \frac{0.2}{x_3}$$

Рисунок 8.19 ілюструє операцію циліндричного розширення.

$A=$

x_1	0.2
x_2	0.6
x_3	0.2

$\mathrm{cext}(A, X\times Y)=$

	y_1	y_2	y_3	y_4
x_1	0.2	0.2	0.2	0.2
x_2	0.6	0.6	0.6	0.6
x_3	0.2	0.2	0.2	0.2

Рисунок 8.19 - Циліндричне розширення при дискретній області визначення

Оперуючись на операцій проекції і циліндричного розширення вводиться операція нечіткої композиції.

Нехай є нечітке відношення *R* на *X*×*Y* і нечітка множина *A* на *X*, тоді нечітка множина на *Y* виходить як композиція *A* та *R* відповідно до виразу:

$$B = proj(R \cap cext(A; X \times Y); Y)$$

Операція ∩ зазвичай трактується як *min* (Т-норма) і для безперервних, і для дискретних множин. Операція *proj* для безперервних множин трактується як *sup*, а дискретних – як *max*.

Таким чином, для безперервної області визначення композиція описується формулою

$$\mu_B(y) = \sup_x(\min(\mu_A(x), \mu_R(x,y)))$$

а для дискретних областей визначається формулою:

$$\mu_B(y) = \max_x(\min(\mu_A(x), \mu_R(x,y)))$$

Скорчено операція композиції записується у вигляді:

$$B = A \circ R.$$

Нечіткі відношення, що виконуються послідовно, допускають композицію. Наприклад:

$$R \circ S = \begin{bmatrix} 0.4 & 0.6 \\ 0.5 & 0.2 \\ 0.1 & 1 \end{bmatrix} \circ \begin{bmatrix} 0.8 & 1 \\ 0.5 & 0 \end{bmatrix} = \begin{bmatrix} 0.5 & 0.4 \\ 0.5 & 0.5 \\ 0.5 & 0.1 \end{bmatrix}$$

Композицію нечітких відносин зручно ілюструвати з допомогою графа. Для останнього прикладу виходить граф, показаний на рис. 8.20.

8.11 Робота з нечіткими множинами в MatLab

У системі MatLab реалізовано набір команд для опису функцій власності (ФП) (див. табл. 8.10).

Таблиця 8.10 – Команди для опису функцій приналежності MatLab

Команда	Функція
dsigmf	ФП у вигляді різниці між двома сигмоїдами
evalmf	Обчислення значень довільної ФП
evalmmf	Розрахунок ступенів приладдя для кількох ФП
gauss2mf	Двостороння гауссівська ФП
gaussmf	Гауссівська ФП
gbellmf	Узагальнена Дзвоноподібна ФП
pimf	π-подібна ФП
psigmf	Добуток двох сигмоїдних ФП
sigmf	Сигмоїдна ФП
smf	*s*-подібна ФП
trapmf	Трапецієподібна ФП
trimf	Трикутна ФП
zmf	*z*-подібна ФП
fuzarith	Калькулятор для виконання арифметичних операцій складання, віднімання, множення та поділу над НЧ

Розглянемо приклади опису функцій власності (ФП) у MatLab.

Для опису трикутної ФП використовується команда

```
>> y = trimf(x, [ABC]);
```

де *x* – базова шкала, де описується ФП; *A*, *B* та *C* – координати вершин трикутника на базовій шкалі. Наприклад:

```
x = (0:0.2:10);
y = trimf(x, [3 4 5]);
plot(x, y);
grid;
ylabel('y');
xlabel('x')
```

Отриманий графік показано на рис. 8. 21.

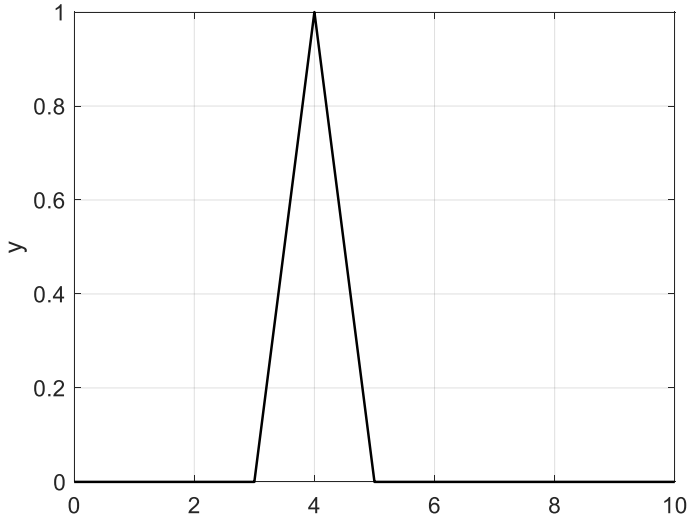


Рисунок 8.21 – Трикутна функція приналежності

Для опису трапецеподібної ФП використовується команда виду

```
>> y = trapmf(x, [A B C D]);
```

де x - базова шкала; A, B, C та D – координати вершин трапеції на базовій шкалі.

Наприклад:

```
x = (0:0.2:10);
y = trapmf(x, [3 4 6 8]); plot(x, y)
grid;
ylabel('y');
xlabel('x')
```

Отриманий графік показано на рис. 8.22.

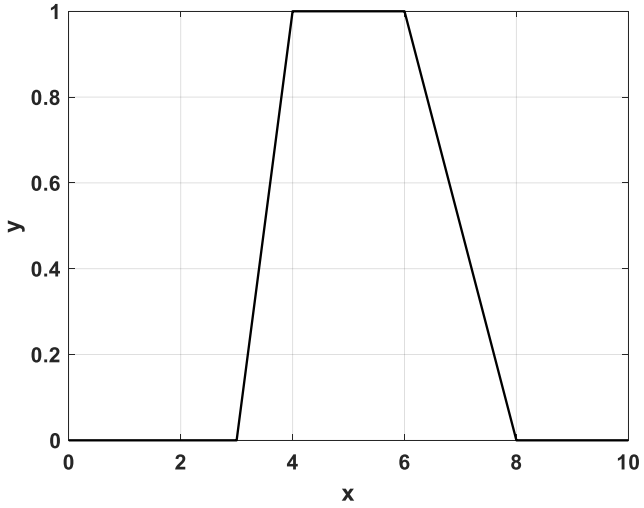


Рисунок 8.22 - Трапецієподібна функція приналежності

Для опису гаусової ФП використовується команда виду

```
>> y = gaussmf(x, [AB]);
```

Наприклад:

```
x = (0:0.2:10);
y = gaussmf(x, [1 5]); plot(x, y)
grid;
ylabel('y');
xlabel('x')
```

Отриманий графік показано на рис. 8.23.

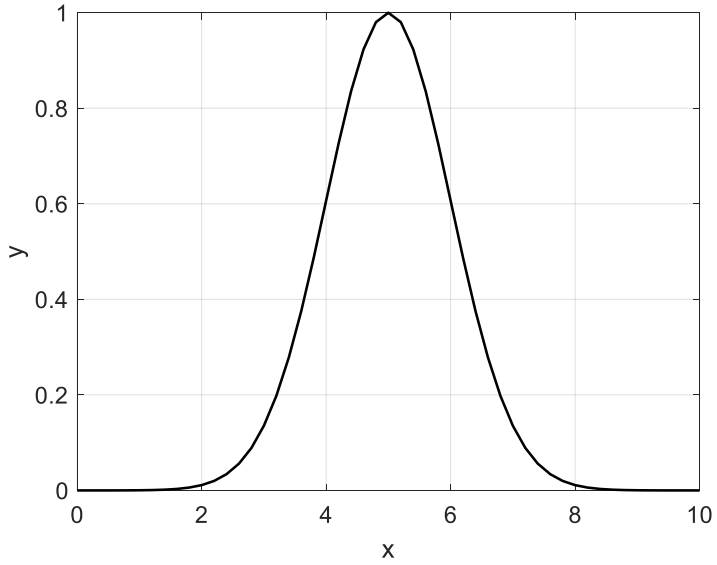


Рисунок 8.23 – Гаусовська функція приналежності

У MatLab передбачено варіант комбінації двох гаусових функцій, так що в проміжку між їхніми центрами значення ФП дорівнює 1.

```
>> y = gauss2mf(x,[B1 A1 B2 A2]);
```

Наприклад:

```
x = (0:0.2:10);
y = gauss2mf(x, [1 5 1.5 6]);
plot(x, y);
grid;
ylabel('y');
xlabel('x')
```

Отриманий графік показано на рис. 8.24.

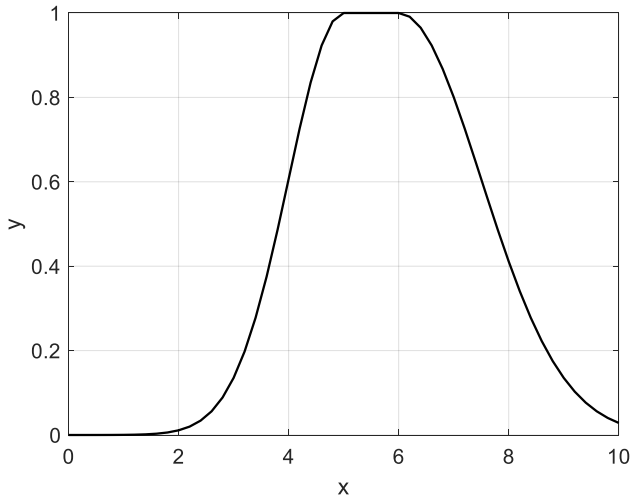


Рисунок 8.24 – Двостороння Гаусовська функція приналежності

Дзвоноподібна ФП описується командою:

```
>> y = gbellmf(x, [ABC]);
```

де A і B - параметри; C – центр ФП.

Розглянемо вплив параметра B:

```
x = (0:0.2:10);
y1 = gbellmf(x, [1 1 5]);
y2 = gbellmf(x, [1 2 5]);
y3 = gbellmf(x, [1 3 5]);
plot (x, y1, x, y2, x, y3);
grid
ylabel('y');
xlabel('x')
```

Сімейство графіків, що вийшло, показано на рис. 8.25.

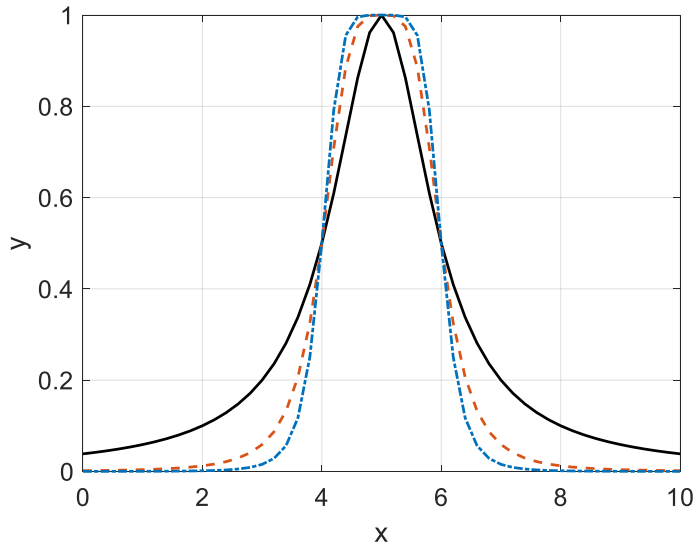


Рисунок 8.25 - Зміна «різкості» дзвоноподібної функції приналежності

Розглянимо вплив параметра *A*:

```
x = (0:0.2:10);
y1 = gbellmf(x, [1 1 5]);
y2 = gbellmf(x, [2 1 5]);
y3 = gbellmf(x, [3 1 5]);
plot(x,y1,x,y2,x,y3);
grid;
ylabel('y');
xlabel('x')
```

Сімейство графіків, що вийшло, показано на рис. 8.26.

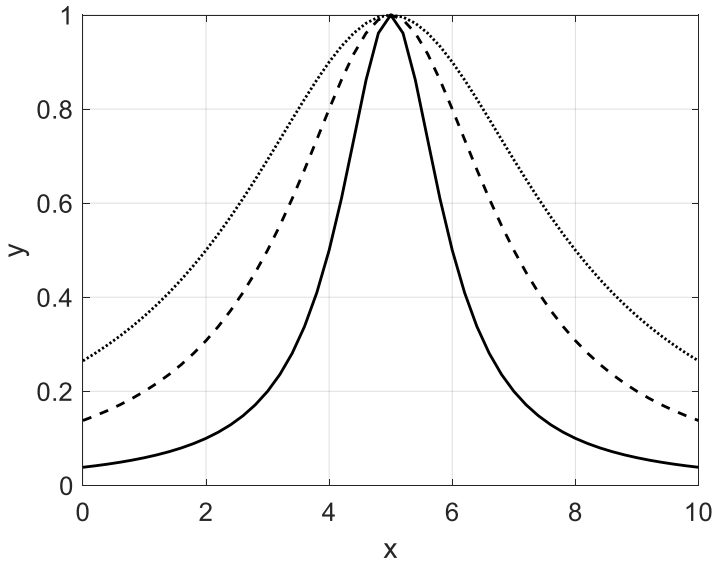


Рисунок 8.26 - Зміна «ширини» дзвоноподібної функції приналежності

Для визначення сигмоїдної ФП використовується команда виду

```
>> y = sigmf(x, [AB]);
```

Для з'ясування впливу параметрів A та B побудуємо сімейство графіків:

```
x = (0:0.2:10);
y1 = sigmf(x, [3 5]);
y3 = sigmf(x, [1 5]);
y2 = sigmf(x, [2 5]);
plot(x,y1,x,y2,x,y3)
grid;
ylabel('y');
xlabel('x')
```

Як свідчить рис. 8.27, параметр *B* визначає координату x, в якою функція приналежності перетинає значення 0,5. Параметр *A* визначає "розмитість" сигмоїдної функції.

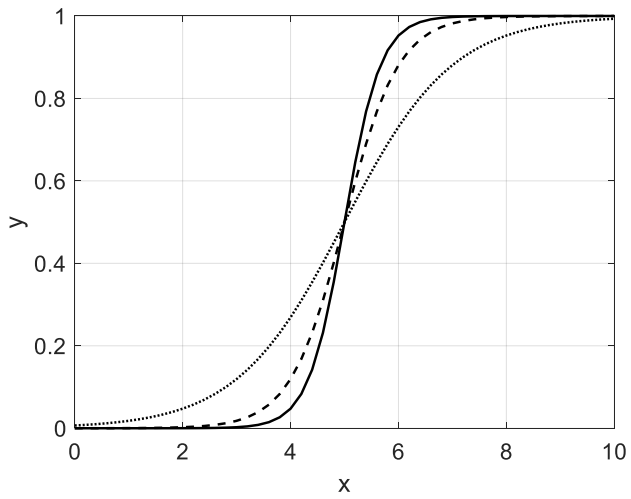


Рисунок 8.27 – Вплив параметрів на опис сигмоїдної функції

Вибираючи негативне значення *A*, можна отримати правосторонню сигмоїдну функцію (див. рис. 8.28):

```
x = (0:0.2:10);
y1 = sigmf(x, [-3 5]);
y2 = sigmf(x, [-4 5]);
y3 = sigmf(x, [-1 5]);
plot(x,y1,x,y2,x,y3)
grid;
ylabel('y');
xlabel('x')
```

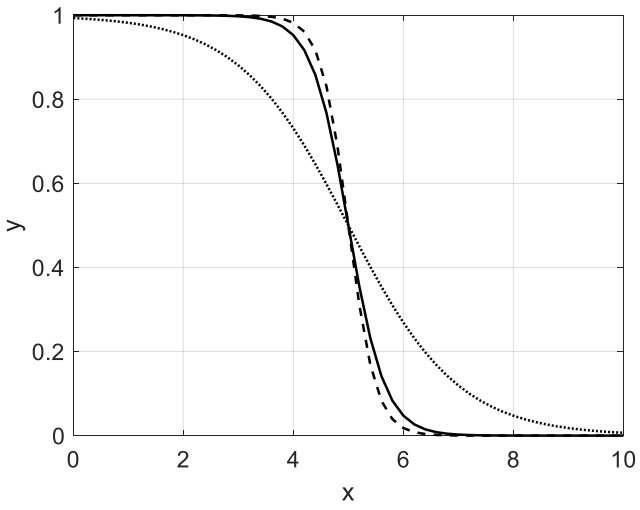


Рисунок 8.28 – Правостороння сигмоїдна функція

За допомогою функції *psigmf* може бути використаний добуток «лівосторонньої» та «правосторонньої» сигмоїдних функцій, наприклад:

```
x = (0:0.2:10);
params1 = [2 3];
params2 = [-5 8];
y = psigmf(x, [params1 params2]);
plot(x,y);
grid;
ylabel('y');
xlabel('x')
```

Графік добутоку сигмоїдних функцій показаний на рис. 8.29.

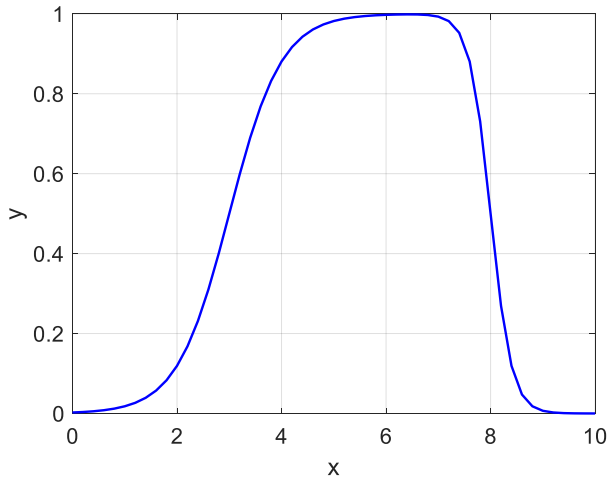


Рисунок 8.29 – Добуток сигмоїдних функцій

Різниця сигмоїдних функцій задається командою виду:

```
>> dsigmf(x,[A1 B1 A2 B2])
```

Наприклад:

```
x = (0:0.2:10);
y=dsigmf(x,[5 2 5 7]);
plot(x,y);
grid;
ylabel('y');
xlabel('x')
```

Ця команда відповідає наступній групі команд:

```
x = (0:0.2:10);
y1 = sigmf(x, [5 2]);
y2 = sigmf(x, [5 7]);
y3 = y1-y2;
plot(x,y3);
grid;
ylabel('y');
xlabel('x')
```

Графік ФП показано на рис. 8.30.

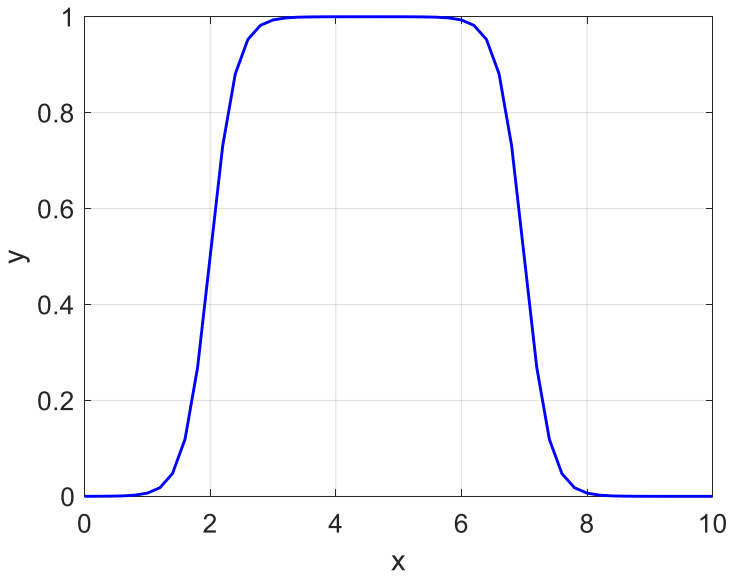


Рисунок 8.30 – Різниця сигмоїдних функцій

На основі сплайн-апроксимації побудована *Z*-подібна ФП, яка дозволяє описати плавне зменшення приналежності від *A* до *B*:

```
>> zmf(x,[a b])
```

Наприклад:

```
x = (0:0.2:10);
plot(x, zmf(x, [2 8]));
grid;
ylabel('y');
```

```
xlabel('x')
```

Графік цієї функції показано на рис. 8.31.

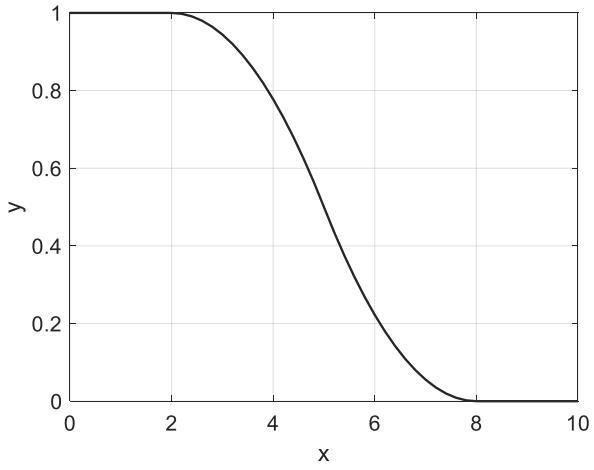


Рисунок 8.31 – Графік Z-подібна функції приналежності

S-подібна ФП є «парною» до Z-подібної функції:

```
>> smf(x,[a b])
```

Наприклад:

```
x = (0:0.2:10);
y=smf(x,[1 8]);
plot(x,y);
grid;
ylabel('y');
xlabel('x')
```

Графік Z-подібної функції показано на рис.8.32.

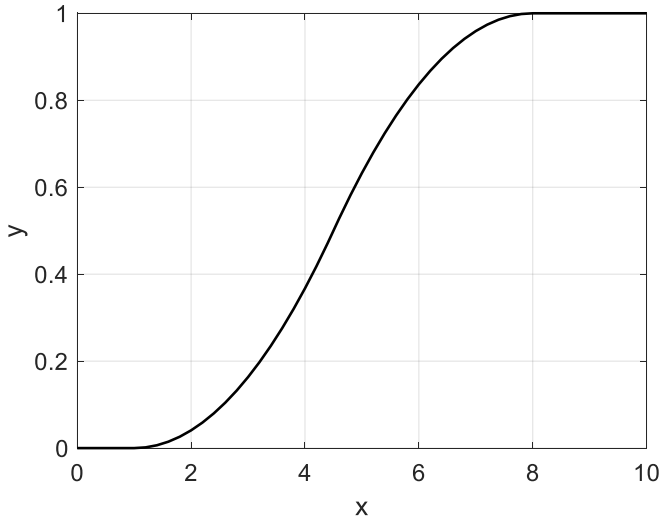


Рисунок 8.32 – Графік Z-подібної функції приналежності

Для опису π-подібної функції приналежності використовуються чотири параметри:

```
x = (0:0.2:10);
y = pimf(x, [2 5 8 9]);
plot(x,y);
grid;
ylabel('y');
xlabel('x')
```

Графік, що вийшов, показаний на рис. 8.33.

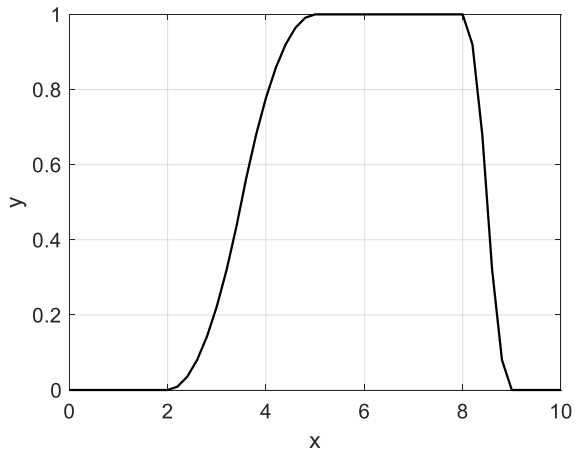


Рисунок 8.33 – Графік π-подібної функції приналежності

За допомогою функції evalmf можна оцінити ступінь приналежності елементів заданого вхідного вектора до нечіткої множини. Функція має формат:

y = evalmf (x, params, type),

де *x* - вектор, для координат якого необхідно розрахувати ступеня належності; *params* - вектор параметрів функції приналежності, порядок завдання яких визначається її типом; *type* – тип функції приналежності, який може бути заданий ім'ям функції або її кодом: 1 – *trimf*; 2 - *'trapmf'*; 3 - *'gaussmf'*; 4 - *'gauss2mf'*; 5 - *'sigmf'*; 6 - *'dsigmf'*; 7 - *'psigmf'*; 8 - *'gbellmf'*; 9 - *'smf'*; 10 - *'zmf'*; 11 - 'pimf'.

При заданні іншого типу функції приналежності передбачається, що її визначено користувачем і задана відповідним m-файлом.

Розглянемо приклад

```
x = [3 4 5];
y = evalmf (x, [2 5 8 9], 'pimf')
y =
0.2222 0.7778 1.0000
```

Обчислити значення кількох функцій приналежності нечітких множин, заданих на одній й тій же універсальній множині, можна з допомогою функції

y = evalmmf(x, params, types),

де *x* - вектор, для координат якого необхідно розрахувати ступеня належності; *params* – матриця властивостей функції приналежності. Перший рядок матриці визначає параметри першої функції приладдя, другий рядок – параметри другої функції приналежності тощо; *types* – матриця типів функції власності. Перший рядок матриці визначає тип першої функції приналежності, другий рядок – тип другої функції приналежності (див. опис команди *evalmf*).
Розглянемо приклад.

```
mf = [fismf(@gaussmf,[1.5 5]) fismf(@trapmf,[3 4 6 7])];
x = (-2:0.1:12)';
y = evalmmf(mf,x);
plot(x,y); grid;
xlabel('Universe of discourse (x)'),ylabel('Membership value (y)')
legend('gaussmf, P=[1.5 5]','trapmf, P=[3 4 6 7]')
```

Результат обчислення функцій приналежності наведено на рисунку 8.34

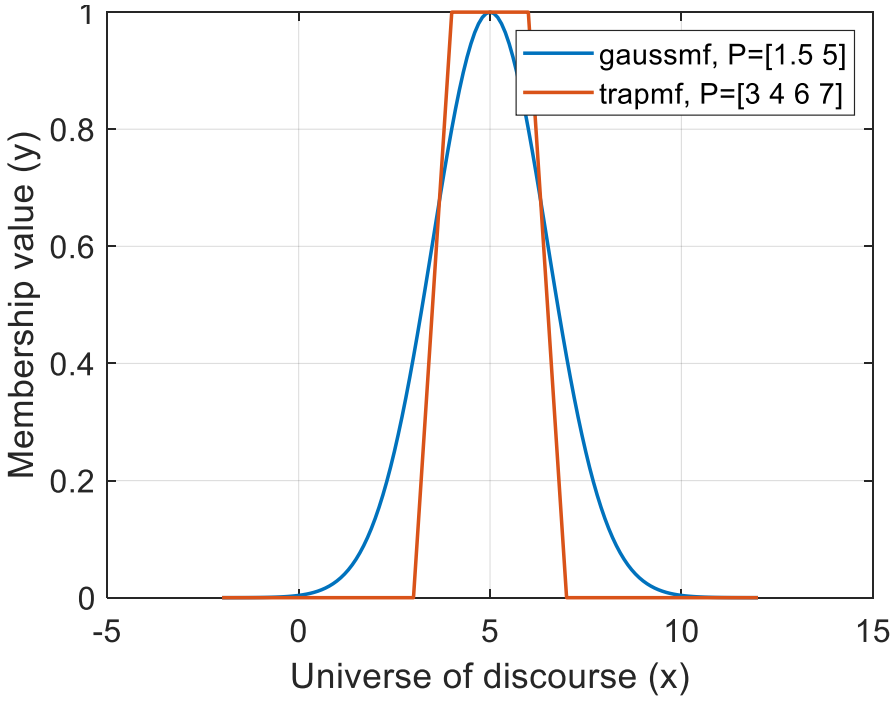


Рисунок 8.34 – Графіки обчислення функцій приналежності гаусовської та трапецеїдальної

Розглянемо приклади виконання операцій над нечіткими множинами MatLab. Нехай дано базова множина X = [-5,20]:

```
N = 501;
minX = -5;
maxX = 20;
x = linspace(minX,maxX,N);
mf1 = gaussmf(x,[1 5]);
mf2 = gaussmf(x,[1 7]);
% обчислення max та побудова графіку обчислення
mf = max(mf1,mf2);
figure(1)
plot(x,mf,'LineWidth',3)
% обчислення min та побудова графіку обчислення
figure(2)
mf3 = min(mf1,mf2);
plot(x,mf3,'LineWidth',3)
% обчислення об’єднання та побудова графіку обчислення
figure(3)
mf4 = probor([mf1;mf2])% об’єднання
plot(x,mf4,'LineWidth',3)
% обчислення перетин та побудова графіку обчислення
figure(4)
mf4 = prod([mf1;mf2])
plot(x,mf4,'LineWidth',3)% перетин
```

На рис. 8.35 та 8.36 показано виконання операцій об'єднання та перетину при використанні відповідно операторів *max* та *min*.

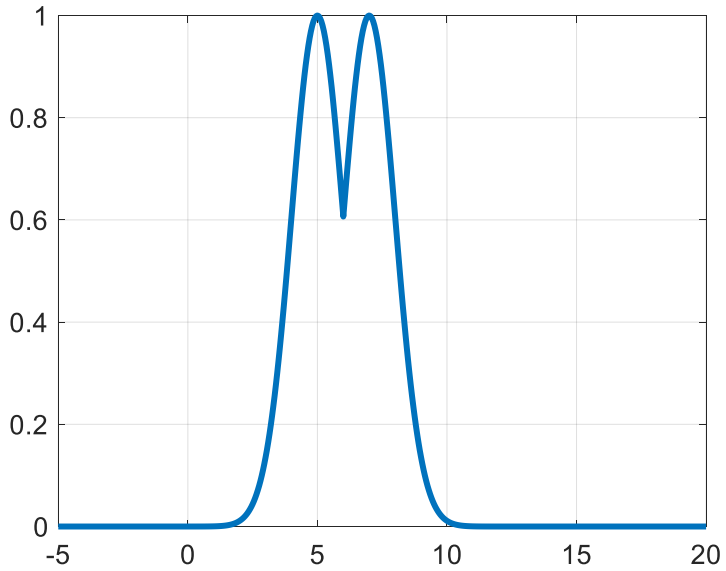


Рисунок 8.35 – Графіки об'єднання нечітких множин (операція max)

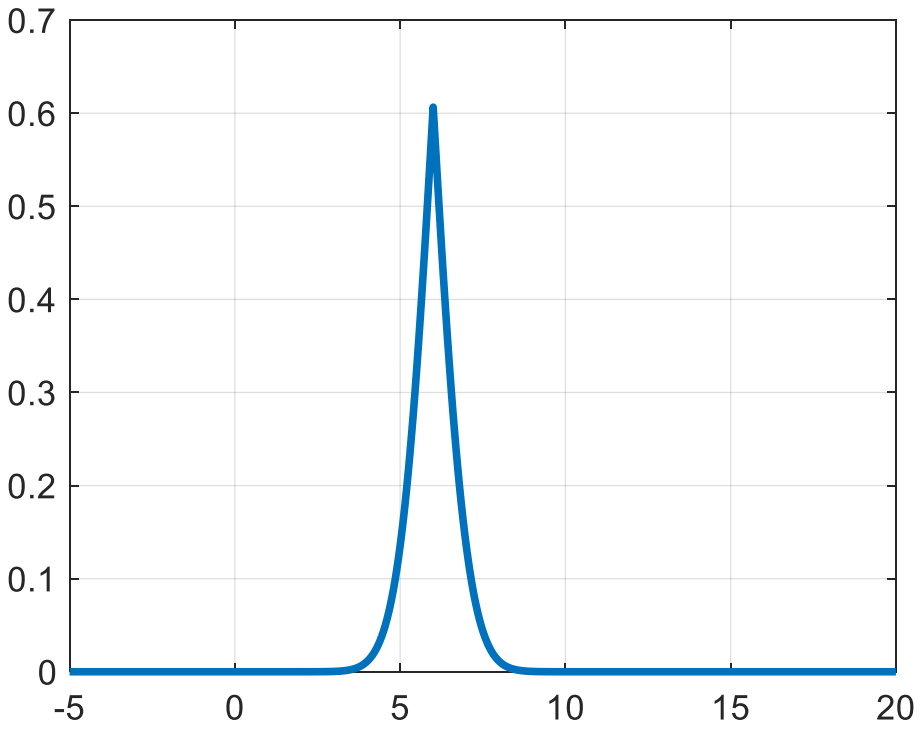


Рисунок 8.36 – Графіки перетину нечітких множин (операція min)

На рис. 8.37 та 8.38 показано виконання операцій об'єднання та перетину при використанні відповідно операторів обмеженої суми та додатку (див. табл. 8.6).

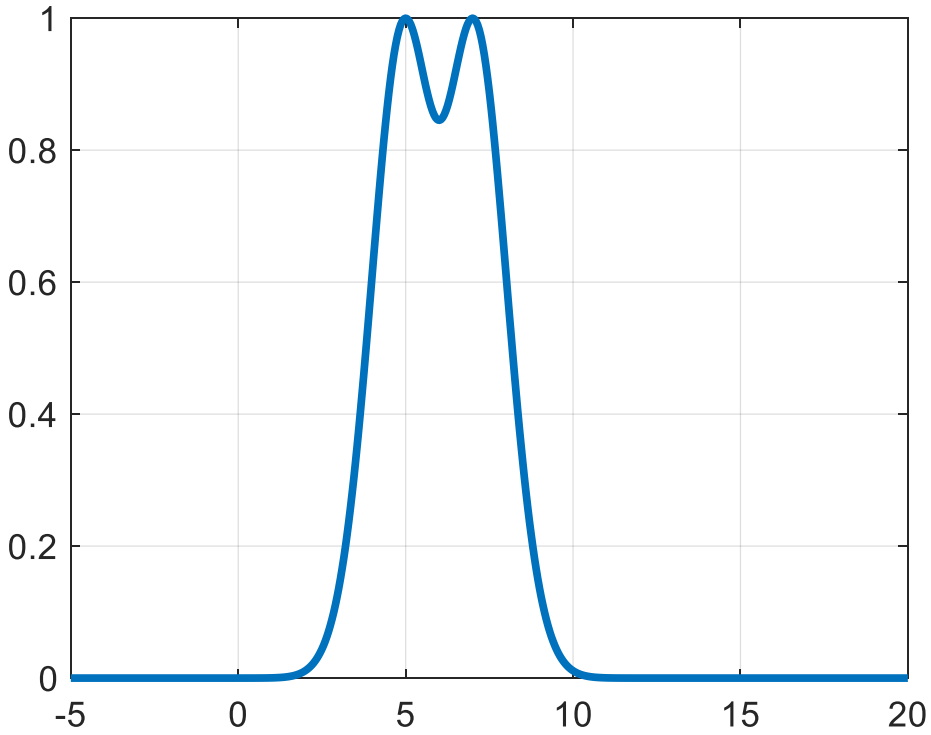


Рисунок 8.37 – Графіки об'єднання нечітких множин (операція обмежена *сума)*

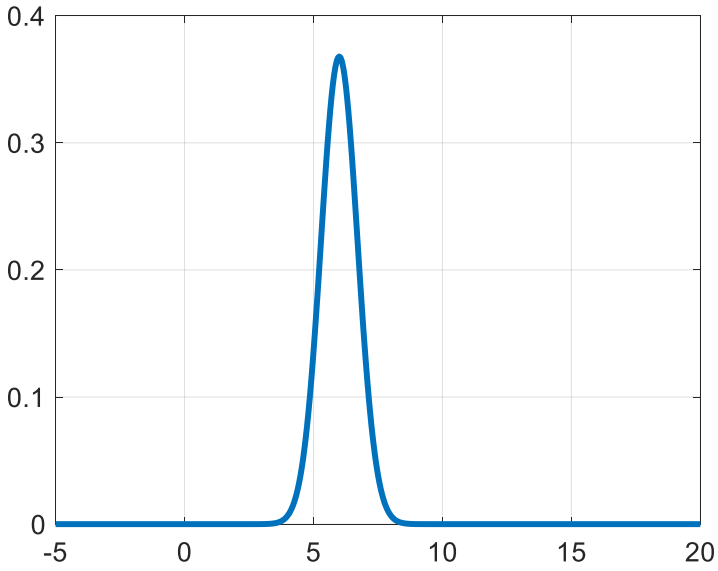


Рисунок 8.38 – Графіки перетину нечітких множин (операція добуток)

Для виконання арифметичних операцій складання, віднімання, множення та поділу над нечіткими числами нечіткої множини у складі MatLab є калькулятор fuzarith. Для його використання необхідно задати рядок виду

c = fuzarith(x, a, b, operator),

де *x* – універсальна множина, на якій задані нечіткі числа; *a* – вектор ступенів приналежності елементів універсальної множини першої нечіткої множини; *b* – вектор ступенів приналежності елементів універсальної множини другої нечіткої множини; operator - одна з допустимих арифметичних операцій: *'sum'* - додавання; *'sub'* - віднімання; *'prod'* – множення; *'div'* – розподіл.

Вихідною змінною функцією *fuzarith* є вектор ступенів приналежності елементів універсальної множини *x* результату виконання нечіткої арифметичної операції. Розмірності векторів *x*, *a*, *b* та *c* повинні бути однаковими.

Приклад виконання арифметичних операцій складання, віднімання, множення та поділу над нечіткими числами нечіткої множини у MatLab:

```
N = 501;
minX = -20;
maxX = 20;
x = linspace(minX,maxX,N);

A = trapmf(x,[-10 -2 1 3]);
B = gaussmf(x,[2 5]);
```

```
%Evaluate the sum, difference, product, and quotient of A and B.
Csum = fuzarith(x,A,B,'sum');
Csub = fuzarith(x,A,B,'sub');
Cprod = fuzarith(x,A,B,'prod');
Cdiv = fuzarith(x,A,B,'div');

%Plot the addition and subtraction results.
figure(5)
subplot(2,1,1)
plot(x,A,'--',x,B,':',x,Csum,'c')
title('Fuzzy Addition, A+B')
legend('A','B','A+B')
subplot(2,1,2)
plot(x,A,'--',x,B,':',x,Csub,'c')
title('Fuzzy Subtraction, A-B')
legend('A','B','A-B')
figure(6)
subplot(2,1,1)
plot(x,A,'--',x,B,':',x,Cprod,'c')
title('Fuzzy Multiplication, A*B')
legend('A','B','A*B')
subplot(2,1,2)
plot(x,A,'--',x,B,':',x,Cdiv,'c')
title('Fuzzy Division, A/B')
legend('A','B','A/B')
```

На рис. 8.39 та показаний результат арифметичного обчислення нечітких множин

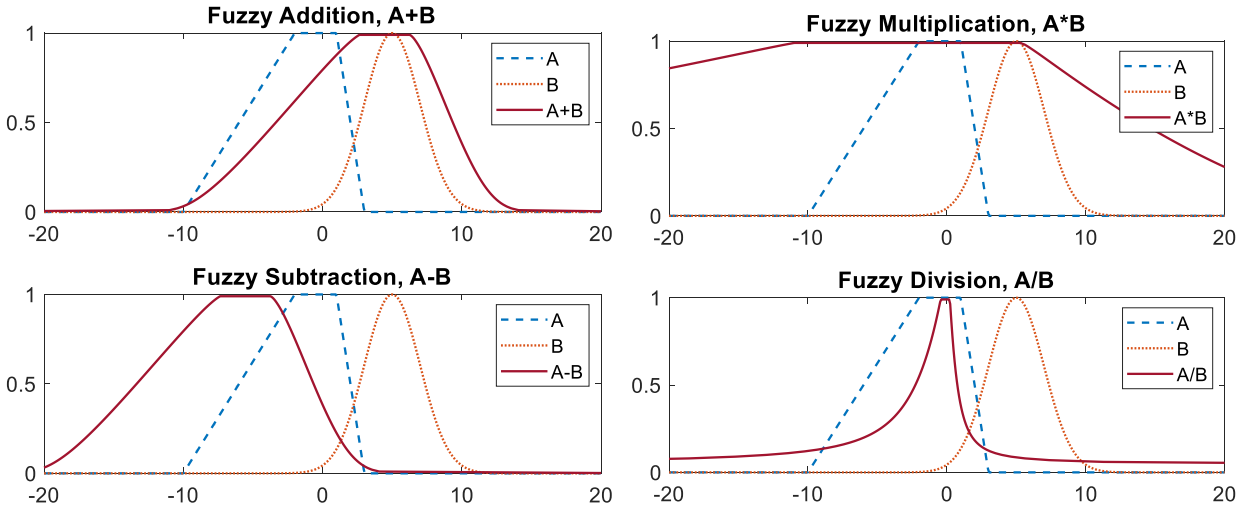


Рисунок 8.39 – Графіки результатів арифметичного обчислення нечітких множин

Нечіткі арифметичні операції виконуються за таким алгоритмом:

– перетворення нечітких чисел-операндів в α-рівневі нечіткі множини;

– виконання арифметичної операції для кожного α-рівня відповідно до принципу розширення;

– перетворення результуючого нечіткого числа з α-рівневого уявлення до традиційного виду.

Кількість α-рівнів може вибиратися користувачем.

9 ІНТЕЛЕКТУАЛЬНІ ДИНАМІЧНІ СИСТЕМИ З ВИКОРИСТАННЯМ НЕЧІТКИХ ЛОГІКИ

9.1 Двійкова логіка, висловлювання та предикати

Логіка досліджує методи міркувань, які застосовуються для переконливого обґрунтування тверджень. Предметом вивчення у логіки є поняття *істини*, *неправдиво* та *протиріччя*, а також причини істинності чи протиріччя висновків, отриманих із справжніх посилок.

Логіка висловлювань розглядає речення мови не з погляду точки зору їх змісту, а лише з погляду їхньої істинності чи протиріччя. Дж. Буль у середині 19 століття запропонував алгебру логіки (булеву алгебру), що дозволило перетворити логіку на математичну науку.

Булева алгебра називається також двійковою або перемикальною алгеброю, оскільки можна кодувати значення "протиріччя " логічним нулем, а значення "істина" - логічною одиницею. Це дозволяє використовувати булеву алгебру як основний інструмент для опису поведінки цифрових електронних схем.

Висловлювання – це твердження, яке може бути істинним або протирічимо (TRUE або FALSE). Логічні формули виходять за допомогою основних логічних зв'язок «І» та «АБО», що поєднують окремі (атомарні) висловлювання, а також заперечення «НІ». Зв'язка «І» носить назву *кон'юнкції* (її позначають також символами & або ∧ або *), зв'язка АБО - диз'юнкції (символи ∨ або +), заперечення «НІ» позначається за допомогою символу ¯. Закони булевої алгебри дозволяють перетворювати і визначати значення логічних формул.

1. Закон комутативності (переміщень):

$$x_1 x_2 = x_2 x_1; \quad x_1 \vee x_2 = x_2 \vee x_1.$$

(результат не залежить від порядку проходження змінних).

2. Закон асоціативності (сполучний):

$$x_1(x_2 x_3) = (x_1 x_2)x_3 = x_1 x_2 x_3;$$

$$x_1 \vee (x_2 \vee x_3) = (x_1 \vee x_2) \vee x_3 = x_1 \vee x_2 \vee x_3.$$

3 Закон дистрибутивності (розподільні):

$$x_1(x_2 \vee x_3) = x_1 x_2 \vee x_1 x_3;$$

$$x_1 \vee x_2 x_3 = (x_1 \vee x_2)(x_1 \vee x_3);$$

$$x_1 \vee \overline{x_1} x_2 = x_1 \vee x_2.$$

4. Закон додатковості:

$$x\overline{x} = 0; \quad x \vee \overline{x} = 1.$$

5. Закон повторення (тавтології) змінної:

$$x_1 = x; \; xx = x; \; \; x \vee x = x; \; \; x \vee 0 = x.$$

6. Закон повторення (тавтології) константи

$$x \wedge 0 = 0; \quad x \vee 1 = 1.$$

7. Закон подвійного заперечення (інверсії):

$$\bar{\bar{x}} = x$$

8. Закон ідемпотентності:

$$xx = x; \quad x \vee x = x.$$

9. Закони інвертування (правила де Моргана):

$\overline{x_1 x_2} = \overline{x_1} \vee \overline{x_2}$. (інверсія добутку дорівнює сумі інверсій);

$\overline{x_1 \vee x_2} = \bar{x}_1 \bar{x}_2$. (інверсія суми дорівнює добутку інверсій).

10. Закони склеювання:

$$x_1 x_2 \vee x_1 \overline{x_2} = x_1; \quad x_1 x_2 \vee \overline{x_1} x_3 \vee x_2 x_3 = x_1 x_2 \vee \overline{x_1} x_3;$$

$$(x_1 \vee x_2)(x_1 \vee \overline{x_2}) = x_1; \quad (x_1 \vee x_2)(\overline{x_1} \vee x_3) = x_1 x_3 \vee \overline{x_1} x_2.$$

11. Закони поглинання:

$$x_1(x_1 \vee x_2) = x_1; \qquad x_1 \vee x_1 x_2 = x_1;$$

$$x_1(\overline{x_1} \vee x_2) = x_1 x_2; \qquad x_1 \vee x_1 x_2 \vee x_1 x_3 = x_1;$$

$$x_1(x_1 \vee x_2)(x_1 \vee x_3) = x_1; \qquad x_1 \vee \overline{x_1} x_2 = x_1 \vee x_2.$$

12. Закони заміщення:

$$x_1 \vee \overline{x_1} x_2 = x_1 \vee x_2; \qquad x_1(\overline{x_1} \vee x_2) = x_1 x_2.$$

13. Закони виявлення:

$$x_1x_2 \vee \overline{x_1}x_3 = x_1x_2 \vee \overline{x_1}x_3 \vee x_2x_3;$$

$$(x_1 \vee x_2)(\overline{x_1} \vee x_3) = (x_1 \vee x_2)(\overline{x_1} \vee x_3)(x_2 \vee x_3).$$

Легко помітити, що закони булевої логіки мають той самий вигляд, як і закони математичної теорії множин, якщо замінити ∧ на ∩, а ∨ на ∪.

Логічний висновок у вирахуванні висловлювань реалізується використанням затвердження

Якщо A, то *B*,

тобто якщо *A* істинно, то теж *B* істинно. Якщо ж *A* неправдиво, то значення довільно. Якщо ж *B* неправдиво, а *A* істинно, то неправдивим виявляється твердження "Якщо *A*, то *B*". Це збігається з формулою:

$$\overline{A} \vee B.$$

Зважаючи на важливість формули "Якщо *A*, то *B*" для неї введено спеціальне позначення «→» (імплікація) і запис $A \rightarrow B$ читається, як "*A* тягне *B*". Стрілка показує напрямок міркувань – з *A* випливає *B*, але не навпаки. Таким чином:

$$\mathrm{A} \rightarrow \mathrm{B} = \overline{A} \vee B.$$

У логіки потрібно виводити істинні речення з інших істинних речень. Якщо деякі посилки істинні, то не можна довести одночасно істинність деякої пропозиції та істинність її заперечення.

Відомо класичне правило логічного висновку, яке називається правилом *скорочення посилки* (modus ponens):

$$A \wedge (A \rightarrow B) = B .$$

Під обчисленням (або дедуктивною системою) розуміється система, в якій існує кілька вихідних об'єктів (аксіом) і правила виведення нових об'єктів з вихідних.

Істотно, що при цьому порядок правил виводу може бути довільним (на відміну алгоритмів).

Розв'язувана задача представляється як сукупності тверджень (аксіом), мета завдання також записується як твердження. Тоді рішення задачі є з'ясування логічного послідовності з аксіом

$$(\mathrm{A}_1 \wedge \mathrm{A}_2 \wedge \mathrm{A}_3 \wedge \ldots \wedge \mathrm{A}_n) \rightarrow \mathrm{B}.$$

Логічна формула від однієї чи кількох аргументів називається *предикатом*. Введення предикатів у логічні формули значно розширює можливості та зручність логічного опису, дозволяючи формалізувати

досить складні описи предметної галузі.

Елементарні компоненти мови предикатів:

− предикатні символи;

− символи змінних та констант;

− функціональні символи.

Кожному предикатному символу приписується відношення предметної області, кожному символу константи елемент області, кожному функціональному символу – функцію у цій області.

Предикат $F(x)$, визначений на предметній області M, є істинним, якщо об'єкт x має властивість F, і помилковим – якщо цієї властивості немає.

Наприклад, нехай є предикат «*Висока швидкість*», визначений на шкалі швидкостей в км/год:

Висока швидкість (5) = 0, *Висока швидкість* (500) = 1.

Предикат $F(x_1, x_2, x_2,..., x_n)$ задає відношення між елементами $x_1, x_2, x_2,..., x_n$ та інтерпретується як позначення висловлювання «*Елементи* x1, $x_1, x_2, x_2,..., x_n$ *знаходяться між собою щодо F*».
Наприклад:

Столиця (Україна, Київ) = 1, *Столиця (Польща, Київ)* = 0.

Предикати, які не можна розбити на окремі компоненти називають *атомарними*. Складні формули будуються шляхом комбінування атомарних предикатів логічними зв'язками І, АБО та НІ.

Допустимі вирази у мові предикатів називаються *правильно побудованими формулами* (ППФ).

У логікі предикатів широко використовуються *квантори існування* ∃ та *спільності* ∀.

Запис $(\exists x)F(x)$ означає, що існує хоча б один об'єкт x, для якого $F(x)$ є істинним.

Запис $(\forall x)\ F(x)$ означає, що вислів істинно при всіх $x \in M$ і неправдиво – у протилежному випадку.

Символ ∀ може бути визначений у термінах ∃ і навпаки:

$$(\exists x)F(x) = \overline{(\forall x)\ \overline{F(x)}};$$

$$\overline{(\exists x)F(x)} = (\forall x)\ \overline{F(x)}.$$

Якщо змінна пов'язані з будь-яким квантором, її називають пов'язаної, інакше вона вільна.

Для того, щоб висловлювання про істинність твердження мало сенс, всі змінні, що входять до нього, мають бути пов'язані.

Наприклад, наступний вираз є ППФ:

$$(\forall x)F(x) \to (\exists x)\big(F(x,y) \wedge F(x)\big).$$

Таким чином, формула в мові предикатів може бути побудована з інших формул, які пов'язані один з одним операціями кон'юнкції, диз'юнкції, еквівалентності або імплікації з можливим використанням операції заперечення. Формулам повинен передувати один або кілька кванторів. Зазвичай використовуються предикати першого порядку, тобто такі, для яких у ППФ заборонено використання кванторів по відношенню до предикатних або функціональних символів.

Для того щоб записати мовою обчислення предикатів знання про деяку предметну область, потрібно знайти атомарні висловлювання та існуючі між ними логічні взаємозв'язки. Потім формулюються узагальнюючі та приватні відомості, використовуючи слова "будь-який" і "деякий" або їх еквіваленти.

Розглянемо найпростіший приклад. Нехай треба описати мовою предикатів пропозицію «Всякий мисливець хоче знати, де сидить фазан». Це можна зробити за допомогою формули:

$$(\forall\, X, Y, Z)\ \text{Мисливець}(X), \text{Фазан}(Y), \text{Місце}(Z) \to$$
$$\to\ \text{Сидить}\ (Y, Z)\ \wedge\ \text{Бажає знати}\ (X, Z).$$

Таким чином, використання предикатів дозволяє розглядати семантику (зміст) деякої предметної галузі. На використанні предикатів засновано роботу мови штучного інтелекту PROLOG.

Зауважимо, що геометрія, наприклад, відноситься до дедуктивних систем. Її постулати можна описати мовою предикатів. Доказ нового твердження (теореми) зводиться до пошуку протиріччя з існуючими аксіомами та твердженнями. Якщо такого протиріччя немає, теорема доведена.

Однак логіка предикатів має очевидні обмеження – і посилки, і висновки правил мають бути або неправдивими, або істинними. У реальному житті достовірність окремих фактів, а також правил є величиною, що набуває проміжних значень між повною істинністю і повною помилковістю.

9.2 Продукційні системи

Логіка предикатів є лише однією з можливих форм уявлення знань. Відмінність логіки предикатів у тому, що окреме правило не є повністю автономним, а розглядається як частина єдиного знання предметної області.

Продукційні системи є альтернативним способом опису знань,

який часто використовують у експертних системах, системах автоматичного керування.

Продукційними називають правила "*умова-дія*". У найпростішому випадку продукційне правило має вигляд

Якщо *A*, то *B*,

де *A* – посилка правила (*антецедент*), а B – висновок правила (*консеквент*).

Антецедент є зразком, призначеним для розпізнавання ситуації, за якої правило спрацьовує. *Консеквент* складається з дії, яка повинна виконуватися в цьому випадку.

Неважко бачити, що набір продукційних правил реалізує табличний спосіб завдання функції, який застосовується у випадках, коли невідома аналітична залежність $y=f(x)$ (див. рис.9.1).

Для продукційних правил, як моделі представлення знань, характерні такі переваги:

− модульність системи продукційних правил, ця властивість дозволяє видаляти окремі продукції або додавати нові без змін інших;

− однаковість правил та їх природність.

x	$f(x)$
x_1	y_1
x_2	y_2
…	…
x_N	y_N

$\Leftrightarrow$

Якщо $x=x_1$, то $y=y_1$
Якщо $x=x_2$, то $y=y_2$
…
Якщо $x=x_N$, то $y=y_N$

Рисунок 9.1 – Система продукційних правил

До основних недоліків системи звичайних продукційних правил слід зарахувати:

− змагальність продукційних правил;

− дискретність функціонування системи продукційних правил.

Змагальність системи продукційних правил означає, що з активації будь-якої однієї продукції необхідно зіставити поточну і еталонні ситуації всіх правил.

Властивість змагальності передбачає, що можливі ситуації, коли зіставлення виявляється успішним для кількох правил. Але оскільки за визначенням функції одному аргументу неспроможна відповідати більше значення функції, у разі утворюється конфліктна множина, з якої у тому чи іншому принципу вибирає лише одне правило. При великій кількісті правил існує можливість втрати працездатності системи, коли окремі правила входять у непримиренні протиріччя.

Дискретність продукційних правил означає, що значення функції може змінюватися різко, стрибками під час переходу від ситуації до

ситуації.

Наприклад, нехай розглядаються правила керування автомобілем при наближенні до опущеного шлагбауму:

"Якщо відстань до шлагбауму = велика, то швидкість = велика";
"Якщо відстань до шлагбауму = середня, то швидкість = середня";
"Якщо відстань до шлагбауму = мала, то швидкість = мала".

Перевірка посилки кожного правила означає перевірку належності поточної відстані до заданого діапазону. Висновок є деякою константою (рис. 9.2).

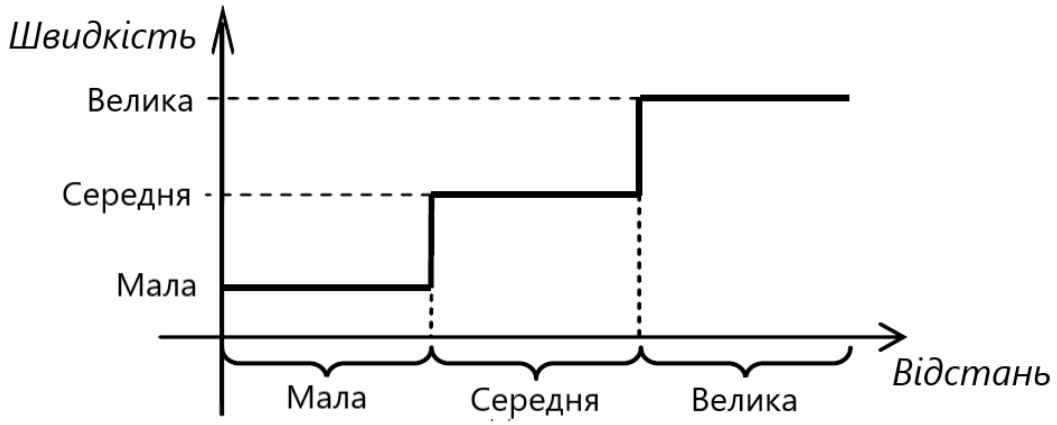


Рисунок 9.2 – Дискретність функціонування продукційної системи

Використання нечітких продукційних правил (як це показано нижче) дозволяє виконувати інтерполяцію між правилами, так що дія плавно змінюється під час переходу від ситуації до ситуації.

Ще одна проблема використання звичайних продукційних систем пов'язана з тим, що для активізації правила потрібен суворий збіг його посилок та вхідної інформації.

У реальному житті істинність посилок майже завжди відрізняється від 100%, тобто в одному факті ми впевнені на 50%, в іншому на 90% і т.д.

Таким чином, характерною є ситуація, коли потрібно мати кількісну оцінку достовірності посилок правила та потрібно обчислювати ступінь достовірності висновку щодо достовірності посилок, яких може бути декілька.

Для вирішення цих проблем можливе використання апарату теорії ймовірностей. У цьому випадку з посиланнями правил пов'язуються деякі ймовірності, а потім обчислюється ймовірність висновку.

Розглянемо базові формули теорії ймовірностей, які цього можливо використовувати.

Нехай є дві події *A* і *B*. Вони можуть відбуватися окремо, відбуватися спільно або взагалі не відбуватися. Наприклад, якщо

виймати з мішка кульки, позначені або літерою *A* або літерою *B* або обома літерами *A* і *B*, ми отримаємо таку модель подій.

Якщо *N* – загальне число подій (спостережень), то $N(A \wedge \overline{B})$ – число спостережень тільки події *A*; $N(\overline{A} \wedge B)$ – число спостережень тільки події *B*; $N(A \wedge B)$ – число спільних появ *A* та *B*; $N(\overline{A} \wedge \overline{B})$ – число спостережень, коли події не відбувалися. Очевидно:

$$P(A) = \frac{N\left(A \wedge \overline{B}\right) + N(A \wedge B)}{N}, P(B) = \frac{N\left(\overline{A} \wedge B\right) + N(A \wedge B)}{N}$$

Вірогідність спільної появи *A* та *B*

$$P(A \wedge B)) = \frac{N(A \wedge B)}{N}$$

Якщо розглянути загальну кількість спостережень *B* і виділити в ньому спостереження, де також було і *A*, то отримаємо умовну ймовірність

$$P(A/B)) = \frac{N(A \wedge B)}{N(B)} = \frac{P(A \wedge B)}{P(B)}.$$

Аналогічно, для загального числа спостережень *A* можна виділити події, коли відбувалося також *B*:

$$P(B/A)) = \frac{N(A \wedge B)}{N(A)} = \frac{P(A \wedge B)}{P(A)}.$$

З останніх двох формул випливає правило Байєса:

$$P(A)P(B/A)) = P(B)P(A/B).$$

Для обчислення ймовірності події справедлива ще одна формула, яка виходить із використанням формули Байєса. Так як:

$$P(B) = \frac{N(B \wedge A) + N(B \wedge \overline{A})}{N} = P(B \wedge A) + P\left(B \wedge \overline{A}\right).$$

тоді можливо записати вираження:

$$P(B) = P(B/A)P(A) + P(B/\overline{A})P(\overline{A})$$

яке називається правилом композиції. Якщо подія *B* відбувається лише за умови *A*, правило композиції спрощується:

$$P(B) = P(B/A)P(A).$$

Цю формулу можна використовуватиме обчислення ймовірності (істинності) укладання продукційного правила. Наприклад, якщо істинність правила «*Якщо сьогодні буде дощ*», *то й завтра буде дощ*» = 0.5, а ймовірність, що «*сьогодні буде дощ*» = 0.1, то ймовірність, що «*завтра буде дощ*» = 0.05.

Якщо правило має безліч посилок A_i, формула композиції природним чином узагальнюється:

$$P(B) = \sum_{i=1}^{N} P(B/A_i)P(A_i),$$

тобто ймовірність події *B* залежить від набору ситуацій A_i, коли *B* може з'явитися, помноженою на ймовірність кожної ситуації A_i. Необхідно підкреслити, що достовірність побудов теорії ймовірностей базується на законі великих чисел – значення ймовірності події може вважатися достовірним лише за відомого результату великої кількості дослідів. На практиці така інформація зазвичай не може бути отримана, тому в технології експертних систем, які використовують правила для опису знань, застосовуються «квазіймовірні» евристичні обчислення.

9.3 Нечітка логіка та лінгвістичні змінні

Значення істинності нечіткого логічного висловлювання *f* знаходиться у діапазоні від 0 до 1.

$$f: [0,1]^n \to [0,1],$$

тобто *n*−арне нечітке відношення, визначене на *n* нечітких множинах, відображається у відрізок [0, 1].

Базові нечіткі логічні операції мають бути визначені таким чином, щоб кон'юнкції, диз'юнкції та заперечення нечітких висловлювань також були нечіткими висловлюваннями.

У нечіткій логіки, як і класичної логіки, розглядаються властивості об'єктів реального світу. З кожною властивістю пов'язується нечітка множина, яку можна назвати нечітким предикатом.

Повна істинність логічного висловлювання кодується числом 1, повна неправдивість числом 0, але існує безліч градацій істинності в діапазоні [0, 1]. Це дозволяє виконувати логічний висновок на знаннях, у якому достовірність висновку змінюється залежно від достовірності посилок і правил які використовуються.

Істинність нечіткого предикату з його базовій множині задається

обраної функцією приналежності.
Наприклад:

Висока швидкість (50) = 0.5; *Висока швидкість* (80) = 0.8;
Мегаполіс (Пекін) = 1, *Мегаполіс (Луцьк)* = 0.1.

Таким чином, у нечіткій логіки термінологічна відмінність між операціями над значеннями приналежності та операціями над нечіткими множинами певною мірою стирається. Операції над нечіткими множинами визначаються по-елементно за допомогою операцій над ступенями приналежності, а лінгвістичні зв'язки можуть інтерпретуватися як операції над значеннями приналежності, і як операції над нечіткими множинами.

Наприклад розглянемо висловлювання:

Висока швидкість (50) = 0.5

Тут величина 0.5 може інтерпретуватися як істинність предикату «*Висока швидкість*» значення 50. Одночасно це приналежність значення 50 до нечіткої множини «*Висока швидкість*».

Якщо ж розглянути висловлювання (*Висока швидкість*(*x*)) І (*Мала висота*(*y*)), то тут логічна зв'язка "І" може інтерпретуватися як кон'юнкція значень істинності нечітких предикатів (*Висока швидкість*(*x*)) І (*Мала висота*(*y*)) при певних значеннях *x* та *y*. Ця зв'язка може також інтерпретуватися і як операція перетину нечітких (циліндричних) розширень, що визначаються нечіткими множинами "*Висока швидкість*" і "*Мала висота*" в декартовому добутку множин значень швидкостей і висот.

В даний час в нечіткій логіки як операції кон'юнкції та диз'юнкції широко використовуються *t*-норми і *t*-конорми.

Людині зручніше формулювати, і навіть сприймати знання, використовуючи якісні характеристики, тобто словесну форму описи. Для полегшення процесу обробки знань у нечіткій логіки смисл розглянутого завдання описується за допомогою лінгвістичних змінних.

Лінгвістичною змінною називається змінна, задана на деякій базовій шкалі і яка приймає значення, що є словами природної мови, які описуються нечіткими множинами.

Лінгвістичну змінну можна описати за допомогою набору

$$\{x, T(x), X, G, M\},$$

де *x* - найменування лінгвістичної змінної; *T*(*x*) – множина значень лінгвістичної змінної (терм-множина); *X* – базова шкала; *G* – синтаксичне правило генерації імен термів; *M* – семантичне правило зв'язування імені терму та його значення.

Людина ефективно сприймає та запам'ятовує максимум 7±2 градації властивостей будь-якого об'єкта. Це саме стосується і наборів фактів та зв'язків між ними. Тому при описі лінгвістичної змінної зазвичай використовують від 3 до 7 термів.

У багатьох додатках кількість термів лінгвістичних змінних вважається фіксованою, і кожному терму заздалегідь надається значення. У цьому випадку лінгвістичні зміни описуються трійкою:

$$\{x, T(x), X\}.$$

Наприклад, швидкість руху об'єкта можна описати за допомогою трійки:

x = *"Швидкість"*, *T*(*x*) = {*"низька"*, *"середня"*, *"висока"*}, *X* = [0,150].

Зазвичай опис термів лінгвістичної змінної вводиться таким чином, щоб забезпечити нечітке розбиття базової множини X (рис. 9.3).

Таким чином, лінгвістична змінна набуває значення на множині предикатів, але кожне значення має певну істинність. Наприклад, значення *x*=40 км/год (див. рис. 9.3) набувають такі значення істинності:

"*Швидкість*" є "*Низька*" з істинністю 0.4;
"*Швидкість*" є "*Середня*" з істинністю 0.6.

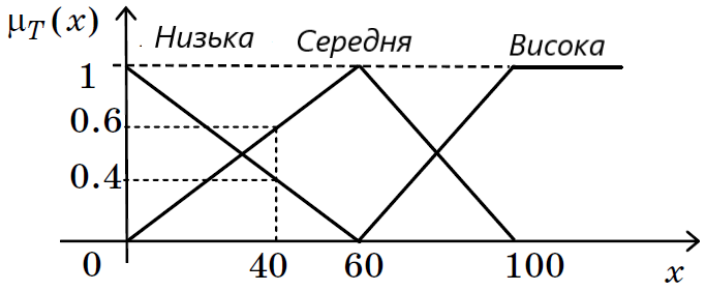


Рисунок 9.3 – Опис лінгвістичної змінної «Швидкість»

Можливі такі види нечітких лінгвістичних висловлювань:

− висловлювання «*A* є *B*», де *A* – найменування лінгвістичної змінної, *B* – її значення, якому відповідає терм із базової терм-множини;

− висловлювання «*A* є *mB*», де *m* – модифікатор значення лінгвістичної змінної, який може бути отриманий з використанням заданих процедур *G* та *M*;

− складові висловлювання, утворені з висловлювань видів 1 і 2 та нечітких логічних операцій у формі зв'язок І, АБО, ЯКЩО-ТО, НІ.

Модифікаторам відповідають такі слова природної мови, як «*дуже*», «не *дуже*», «*більш-менш*» тощо. Їх можна описати, наприклад, за допомогою операції розтягування і стиснення нечіткої множини.

Наприклад, якщо описано функцію приналежності терму «*Деталь*» для лінгвістичної змінної «*Вага*», то модифікацію терму «*Маленька деталь*» можна отримати, виконуючи стиснення вихідного терму (рис. 9.4):

```
x = (0:1:30);
y = sigmf(x, [-0.5 15]);
y1=y.*y;
plot(x,y,x,y1);
grid
ylabel('y');
xlabel('x')
```

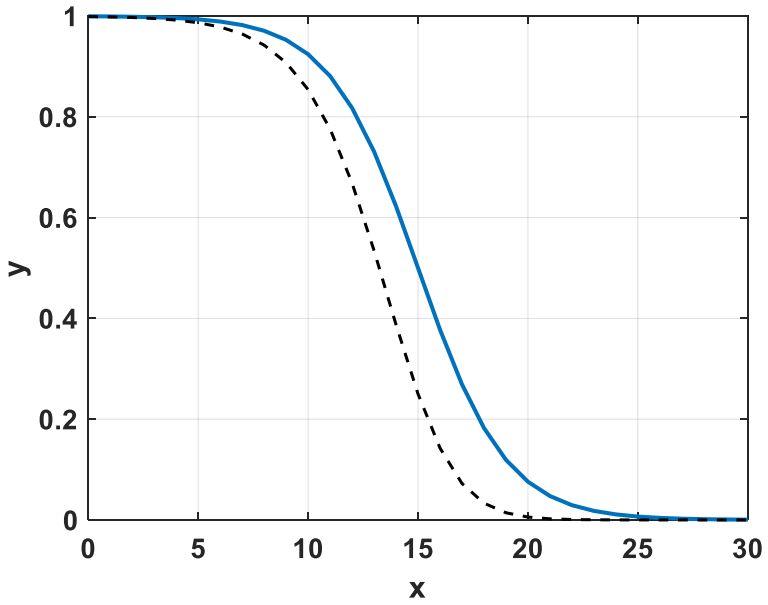


Рисунок 9.4 – Приклад лінгвістичної модифікації терма

9.4 Нечітка імплікація та нечіткі правила

У нечіткому продукційному правилі “Якщо *A*, то *B*” і посилка, і висновок є нечіткими логічними змінними. Позначимо найпростіше нечітке правило *R* наступним чином:

$$R = I(A, B),$$

де *I* – нечітка імплікація.

Нечітку імплікацію можна реалізувати за допомогою нечіткого відношення, що дозволяє описати необхідне відображення з *A* в *B*.

Існують два основні способи опису нечіткої імплікації. При першому імплікація описується як узагальнена кон'юнкція:

$$I(A, B) = T(A, B).$$

При другому імплікація реалізується подібно до класичної

імплікації:

$$I(A,B) = S(\overline{A},B).$$

Відомо багато варіантів нечіткої імплікації, деякі варіанти показані в таблиці 9.1.

Таблиця 9.1 – Варіанти опису нечіткої імплікації

Назва імплікації	Формула
Mamdani	$I(A,B) = min(A,B)$
Larsen	$I(A,B) = A \cdot B$
Yager	$I(A,B) = B^A$
Kleene-Dienes	$I(A,B) = max(1-A,B)$
Gaines	$I(A,B) = \begin{cases} 1, A \le B, \\ 0, A > B. \end{cases}$
Goguen	$I(A,B) = \begin{cases} 1, A = 0, \\ \min(\frac{B}{A}, 1), A \ne B. \end{cases}$

Розглянемо приклад. Нехай є базове безліч X і два його нечіткі підмножини A і B:

$$X = \{1,2,3\};$$
$$A = \frac{1}{1} + \frac{0.8}{2} + \frac{0}{3};$$
$$B = \frac{0.6}{1} + \frac{0.9}{2} + \frac{1}{3}.$$

При використанні нечіткої імплікації *Mamdani* нечітке правило «Якщо *A*, то *B*» описується наступним чином:

$$R_{AB} = I(A,B) = = \frac{0.6}{1,1} + \frac{0.9}{1,2} + \frac{1}{1,3} + \frac{0.6}{2,1} + \frac{0.8}{2,2} + \frac{0.8}{2,3} + \frac{0}{3,1} + \frac{0}{3,2} + \frac{0}{3,3}$$

У матричній формі:

$$R_{AB} = \begin{vmatrix} 0,6 & 0,9 & 1 \\ 0,6 & 0,8 & 0,8 \\ 0 & 0 & 0 \end{vmatrix}.$$

Якщо *A* та *B* визначені за допомогою безперервних функцій приналежності, тоді нечітке правило описується деякою поверхнею. Розглянемо приклад із використанням функцій MatLab. Нехай задані дві нечіткі множини за допомогою функцій приналежності:

N = 50;

```
minX = 0;
maxX = 10;
x = linspace(minX,maxX,N);
N1 = 50;
minY = 0;
maxY = 10;
y = linspace(minY,maxY,N1);
[X1,Y1] = meshgrid([x,y]);
% формування трикутної ф-ції приналежності для кожного
стовбця матриці
mf1 = trimf(X1(:),[1 2 3]);
% формування гаусовської ф-ції приналежності для кожного
стовбця матриці
mf2 = gaussmf(Y1(:),[1 3]);
Z = min(mf1,mf2);
figure(1)
plot3(X1(:),Y1(:),Z)
```

Поверхня, що описує нечітке відношення, показано на рис. 9.5.

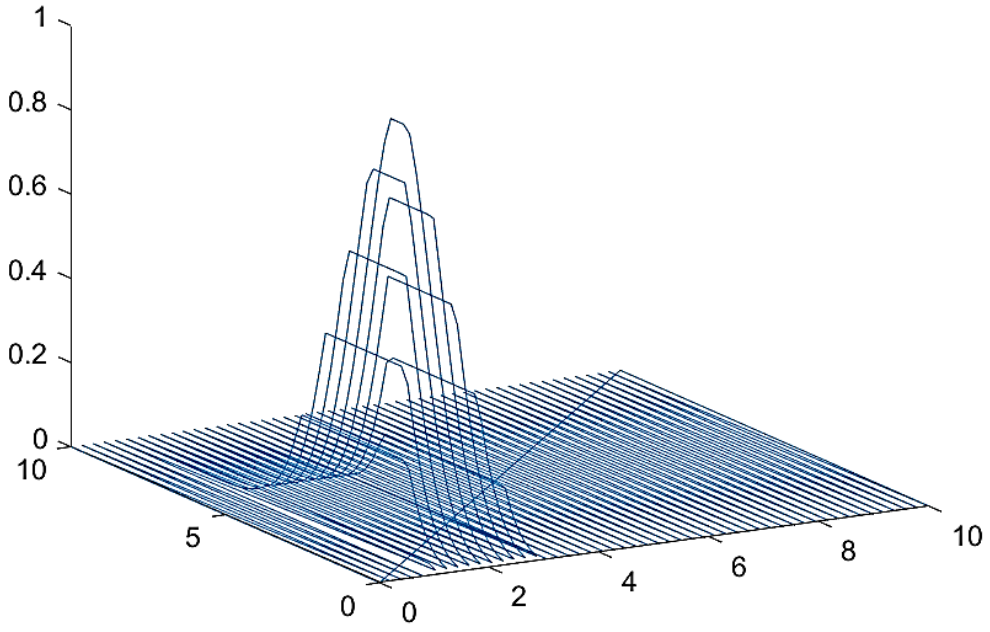


Рисунок 9.5 – Графічна інтерпретація нечіткого відношення

У нечіткій логіки значення істинності логічних змінних лежать у діапазоні [0, 1], тому тут потрібна модернізація правил логічного виводу. Узагальненням правила *modus ponens* для нечітких множин є композиційне правило виводу:

$$B' = A' \circ (A \rightarrow B),$$

де A' – нечітка посилка; B' – нечітке висновок.

На значеннєвому рівні правило композиції означає, що близькі посилки викликають близькі слідства.

Розглянемо приклад для дискретної області визначення.

Нехай дано базова множина

$$X = \{1,2,3,4\},$$

на якому визначено дві нечіткі множини:

$$\mathrm{A} = \frac{1}{1} + \frac{0.8}{2} + \frac{0.3}{3} + \frac{0.1}{4},$$
$$\mathrm{B} = \frac{0}{1} + \frac{0.1}{2} + \frac{0.6}{3} + \frac{1}{4}.$$

Розглянемо нечітке відношення

$$R = \mathrm{A} \to \mathrm{B}.$$

При використанні імплікації Mamdani отримуємо:

$$R = 0/(1,1) + 0.1/(1,2) + 0.6/(1,3) + 1(1,4) + 0/(2,1) + 0.1/(2,2) + 0.6/(2,3) + 0.8/(2,4) + 0/(3,1) + 0.1/(3,2) + 0.3/(3,3) + 0.3/(3,4) + + 0/(4,1) + 0.1/(4,2) + 0.1/(4,3) + 0.1/(4,4).$$

У матричній формі запису

$$R = \begin{vmatrix} 0 & 0.1 & 0.6 & 1 \\ 0 & 0.1 & 0.6 & 0.8 \\ 0 & 0.1 & 0.3 & 0.8 \\ 0 & 0.1 & 0.1 & 0.3 \end{vmatrix}$$

Припустимо, що є НМ - посилання виду:

$$A' = \frac{0}{1} + \frac{0.5}{2} + \frac{1}{3} + \frac{0.3}{4},$$

тоді нечіткий висновок виходить у вигляді

$$B' = A' \circ R = [0 \;\; 0.5 \;\; 1 \;\; 0.3] \circ \begin{vmatrix} 0 & 0.1 & 0.6 & 1 \\ 0 & 0.1 & 0.6 & 0.8 \\ 0 & 0.1 & 0.3 & 0.8 \\ 0 & 0.1 & 0.1 & 0.3 \end{vmatrix} = [0 \;\; 0.1 \;\; 0.5 \;\; 0.5].$$

Розглянемо приклад нечіткого висновку для безперервної області

визначення. Нехай дані дві базові множини *X* та *Y*, на яких задані нечіткі множини за допомогою трикутних функцій приналежності (див. рис. 9.6).

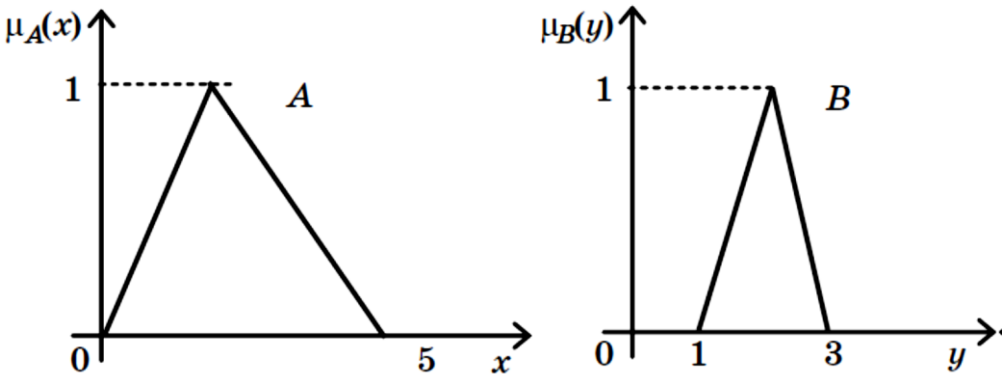


Рисунок 9.6 – Опис посилки та висновку нечіткого правила

Нечітке відношення R: A → B ілюструє рис. 9.7.

```
x2 = (0:0.2:6);
x1 = (0:0.2:6);
[X, Y] = meshgrid(x1, x2);
Z = min(trimf(X(:),[0 2.5 5]),trimf(Y(:),[1 2 3]));
plot3(X(:),Y(:),Z)
```

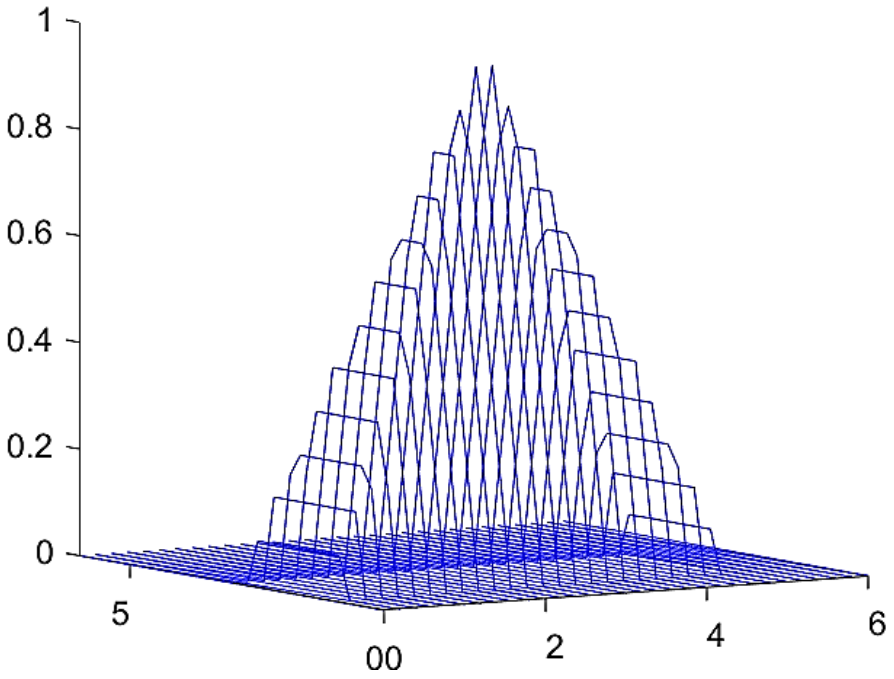


Рисунок 9.7 – Опис нечіткого правила

Нехай є нечітка посилка *A'*. Варіант графічної інтерпретації нечіткого висновку (у разі використання операції *min*) показаний на рис. 9.8.

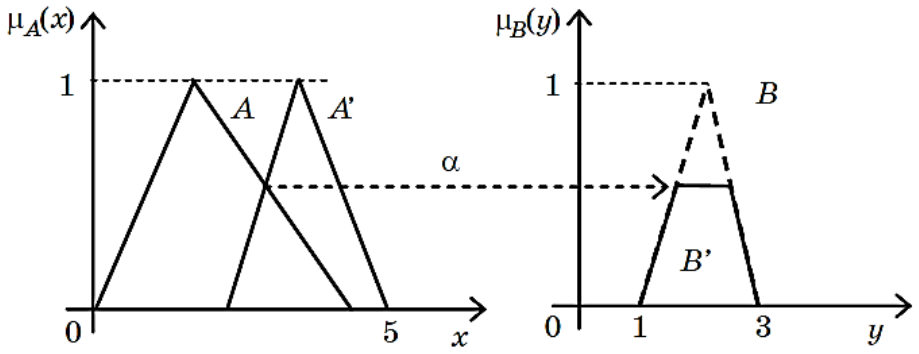


Рисунок 9.8 – Опис нечіткої посилки

Нечіткий висновок описується формулою

$$\mu_{B'} = \sup_x T\mu_{A'}(x), I\big(\mu_A(x), \mu_B(x)\big) = \sup_x \Big(\mu_{A'}(x) \wedge \big(\mu_A(x) \wedge \mu_B(x)\big)\Big) = \\ = \alpha \wedge \mu_B(x).$$

де $\alpha = sup(\mu_{A'}(x) \wedge \mu_A(x)$

Той самий результат у прикладі можна отримати, якщо описати вираз

$$T\big(\mu_{A'}(x), I\mu_A(x), \mu_B(x)\big),$$

командами:

```
x2 = (0:0.2:6);
x1 = (0:0.2:6);
[X, Y] = meshgrid(x1, x2);
Z = min(trimf(X(:),[0 2.5 5]),trimf(Y(:),[1 2 3]));
Z1 = min(trimf(X(:),[3 4 5]),Z);
plot3(X(:),Y(:),Z1)
```

Розглядаючи проекцію просторової фігури (див. рис. 9.9) на площину YZ можна отримати результат, що збігається із правою частиною рис. 9.8.

Нечітка посилка може бути кон'юнкцією, диз'юнкцією або запереченням двох або більшої кількості нечітких множин.

Розглянемо нечіткий висновок для продукційного правила із двома посилками, пов'язаними з допомогою кон'юнкції. Нехай є правило:

$$R_i\text{: Якщо } (x_1 = A_{1,i}) \text{ і } (x_2 = A_{2,i}), \text{то } y = B$$

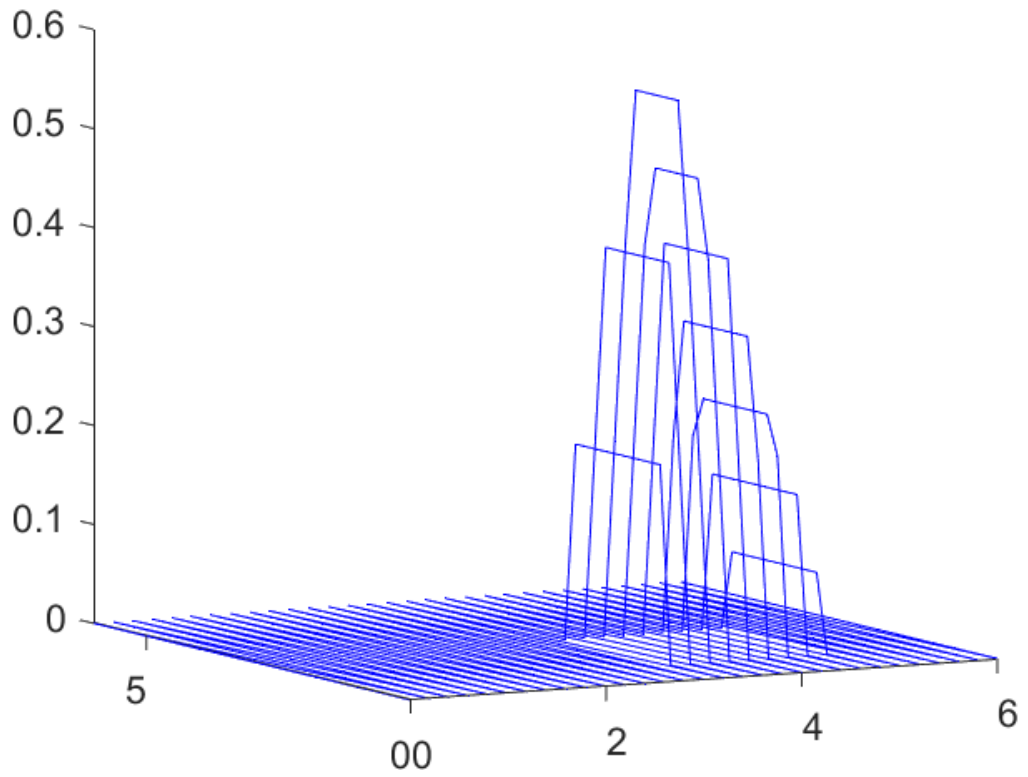


Рисунок 9.9 - Перетин нечіткої посилки та нечіткого відношення

та нечіткі вихідні дані:

$$x_1 = \text{A}'_1 \text{ та } x_2 = \text{A}'_2.$$

Нечіткий висновок виходить за формулою:

$$B' = T(\text{A}'_1, \text{A}'_2) \circ R_i$$

де функція приналежності *B'* описується композицією sup−min:

$$\begin{aligned}\mu_{B'}(y) &= \sup_{x_1,x_2}\left\{\left(\mu_{A'_1}(x_1) \wedge \mu_{A'_2}(x_2)\right) \wedge \mu_R(x_1, x_2, y)\right\} \\ &= \sup_{x_1,x_2}\left\{\left(\mu_{A'_1}(x_1) \wedge \mu_{A'_2}(x_2)\right) \wedge \left[\mu_{A_1}(x_1) \wedge \mu_{A_2}(x_2) \wedge \mu_B(y)\right]\right\} = \\ &= \sup_{x_1}\left(\mu_{A'_1}(x_1) \wedge \mu_{A_1}(x_1)\right) \wedge \sup_{x_2}\left(\mu_{A'_2}(x_2) \wedge \mu_{A_2}(x_2)\right) \wedge \mu_B(y) \\ &= \alpha \wedge \mu_B(y).\end{aligned}$$

де $\alpha = \sup_{x_1}\left(\mu_{A'_1}(x_1) \wedge \mu_{A_1}(x_1)\right) \wedge \sup_{x_2}\left(\mu_{A'_2}(x_2) \wedge \mu_{A_2}(x_2)\right)$.

Розмір α характеризує ступінь відповідності вихідних даних посилкам правила, тобто визначає рівень запуску правила (ступінь спрацьовування).

У випадку нечітке продукційне правило може мати *N* посилок. Рівень запуску у разі описується формулою:

$$\alpha_i = \bigcap_{j=1}^{N} \sup_{x_j} \left(T\left(\mu_{A'_j}(x_j), \mu_{A'_{j,i}}(x_j) \right) \right)$$

9.5 Нечіткий висновок на базі правил

Нечітка логічна система може складатися з багатьох правил. У будь-який момент можливе спрацювання будь-якого з правил, тому можна вважати, що вони пов'язані логічним зв'язком АБО :

$$R = R_1 \vee R_2 \vee R_3 \ldots \vee R_N.$$

Розглянемо приклад для дискретної області визначення. Нехай нечітка система складається з двох правил:

$$R_1: \text{Якщо } x = A_1, y = B_1;$$
$$R_2: \text{Якщо } x = A_2, y = B_2.$$

Посилки правил описуються за допомогою нечітких множин, визначених на дискретній базовій множині:

$$A_1 = \frac{0.7}{1} + \frac{1}{2} + \frac{0.7}{3} + \frac{0.2}{4};\ B_1 = \frac{0}{1} + \frac{0.1}{2} + \frac{0.7}{3} + \frac{1}{4};$$
$$A_2 = \frac{0}{1} + \frac{0.1}{2} + \frac{0.7}{3} + \frac{1}{4};\ B_2 = \frac{0.7}{1} + \frac{1}{2} + \frac{0.7}{3} + \frac{0.2}{4}.$$

Нечіткі відношення, що відповідають правилам, можна описати за допомогою матриць

$$R_1 = A_1 \to B_1 = \begin{bmatrix} 0 & 0.1 & 0.7 & 0.7 \\ 0 & 0.1 & 0.7 & 1 \\ 0 & 0.1 & 0.7 & 0.7 \\ 0 & 0.1 & 0.2 & 0.2 \end{bmatrix},$$

$$R_2 = A_2 \to B_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0.1 & 0.1 & 0.1 & 0.1 \\ 0.7 & 0.7 & 0.7 & 0.2 \\ 0.7 & 1 & 0.7 & 0.2 \end{bmatrix},$$

система нечітких правил описується

$$R = R_1 \vee R_2 = \begin{bmatrix} 0 & 0.1 & 0.7 & 0.7 \\ 0.1 & 0.1 & 0.7 & 1 \\ 0.7 & 0.7 & 0.7 & 0.7 \\ 0.7 & 1 & 0.7 & 0.2 \end{bmatrix}.$$

Якщо базові множини безперервні, то кожному нечіткому відношенню відповідає деяка поверхня, а система правил також є поверхнею, так що кожна її точка – верхня межа всіх відповідних точок окремих поверхонь.

При двох і більше посилках правила матричне уявлення стає скрутним.

Розглянемо приклад для безперервної області визначення. Нехай є система двох правил, кожне з яких має по дві посилки:

$$R_1\text{: Якщо } (x = A_1) \text{ та } (y = B_1), \text{то } (z = C_1);$$
$$R_2\text{: Якщо } (x = A_2) \text{ та } (y = B_2), \text{то } (z = C_2).$$

де A_i, B_i та C_i - нечіткі множини, що входять в *i*-е правило.

Нехай A' та B' − нечіткі посилки, тоді рівень запуску α_i кожного правила можна описати виразом

$$\alpha_i = hgt(A' \cap A_i) \wedge hgt(B' \cap B_i),$$

та вихідний сигнал системи правил виходить за формулою:

$$R(A', B') = R_1 \vee R_2 = (\alpha_1 \wedge C_1) \vee (\alpha_2 \wedge C_2)$$

У багатьох випадках вхідні змінні нечіткої системи є числами (тобто чіткими значеннями). Для виконання нечітких операцій необхідно виконати їх перетворення на нечітку форму (фаззифікацію). Найпростіше розглядати вхідну змінну x_0, як однеточкова нечітка множина (синглетон):

$$\mu_{A'}(x) = \begin{cases} 1, & x = x_0; \\ 0, & x \neq x_0. \end{cases}$$

При двох посилках *x* та *y* кожне правило можна описати так

$$R_i(x, y, z) = [A_i(x) \wedge B_i(y)] \rightarrow C_i(z).$$

При вхідних даних $x=x_0$ та $y=y_0$ вихід *i*-го правила виходить за формулою

$$A'_i = [A_i(x_0) \wedge B_i(y_0)] \rightarrow C_i(z),$$

де $[A_i(x_0) \wedge B_i(y_0)]$ – величина, яка визначає силу запуска правила.

Вихідний сигнал системи нечітких правил описується формулою

$$C = \bigcup_{i=1}^{N} C'_i.$$

Відомо кілька класичних схем нечіткого висновку – варіанти *Mamdani*, *Larsen*, *Tsukamoto* та *Sugeno*.

Розглянемо варіант *Mamdani*. Нечітка імплікація тут моделюється за допомогою оператора *min*.

Покладемо для простоти викладу, що база правил містить лише два правила R_1 та R_2. Сила запуску кожного з них:

$$\alpha_1 = A_1(x_0) \wedge B_1(y_0);$$
$$\alpha_2 = A_2(x_0) \wedge B_2(y_0).$$

Індивідуальний вихід кожного правила виходить за формулами

$$C'_1(z) = \alpha_1 \wedge C_1(z);$$
$$C'_2(z) = \alpha_2 \wedge C_2(z);$$

Графічна ілюстрація схеми *Mamdani* наведена рис. 9.10.

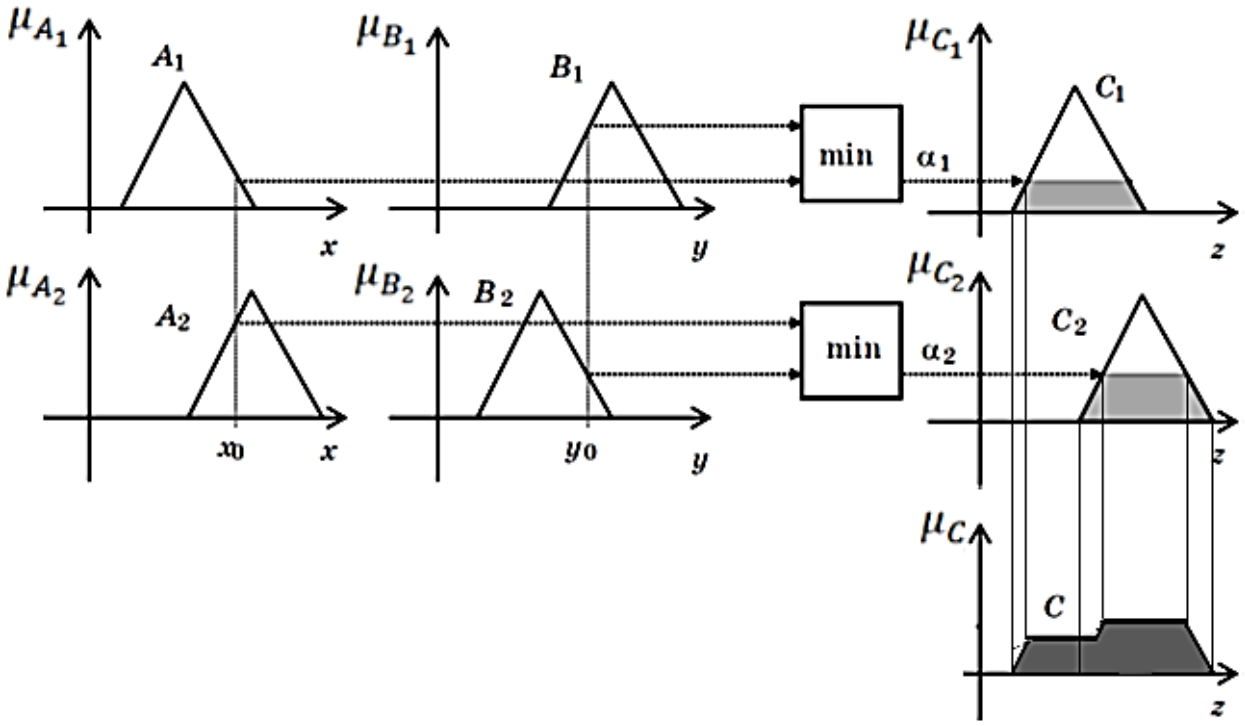


Рисунок 9.10 – Нечіткий висновок Mamdani

Варіант *Larsen* відрізняється від варіанта *Mamdani* лише тим, що оператор імплікації тут описується за допомогою множення алгебри.

$$\alpha_i = A_i(x_0)B_i(y_0);$$
$$C'_i(z) = \alpha_i C_i(z);$$

Графічна ілюстрація схеми *Larsen* наведена рис. 9.11.

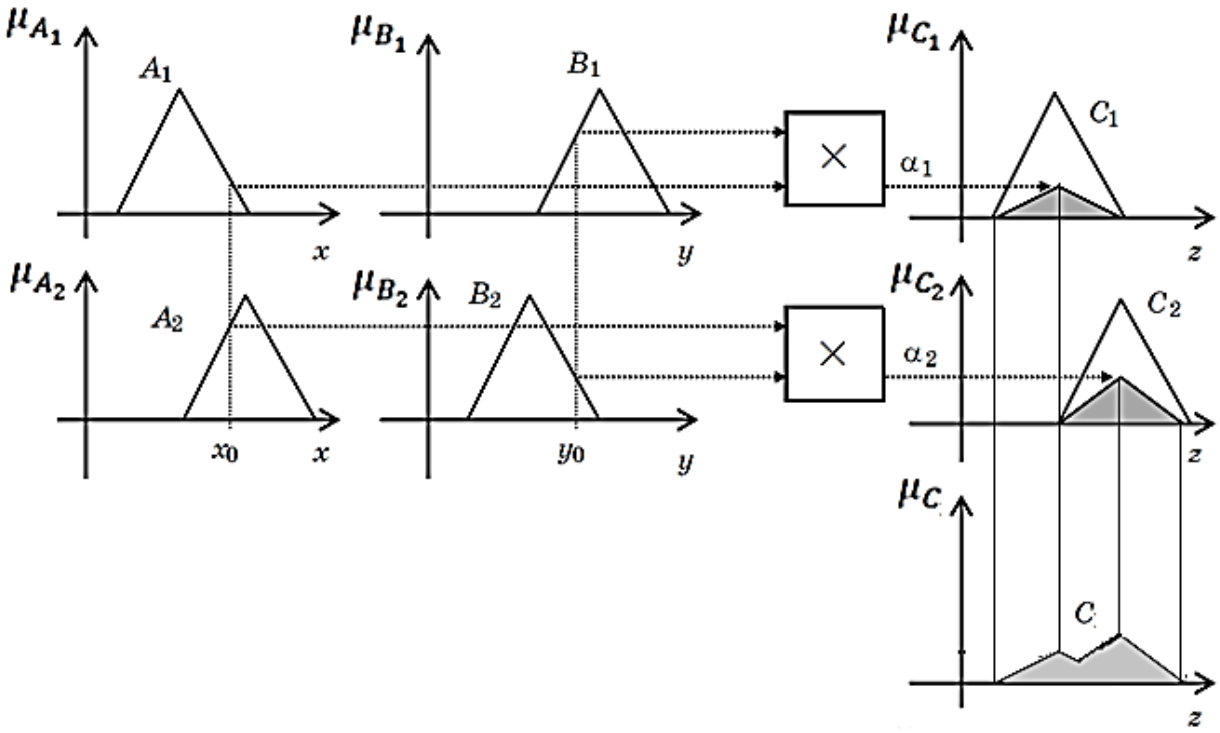


Рисунок 9.11 – Нечіткий висновок Larsen

В схемі *Mamdani*, і в схемі *Larsen* зв'язка *ІНАКШЕ* описується за допомогою оператора *max*:

$$C(z) = C'_1(z) \vee C'_2(z)$$

Нечіткий висновок, розглянутий *Tsukamoto*, відрізняється тим, що всі терми лінгвістичної змінної тут мають монотонні функції приналежності.

Якщо деяке правило має рівень запуску α_i, його вихідний сигнал може бути розрахований за формулою:

$$\alpha_i = C(z_i), \text{тобто } z_i = C^{-1}(\alpha).$$

На рисунку 9.12 наведено схему Tsukamoto для двох правил.

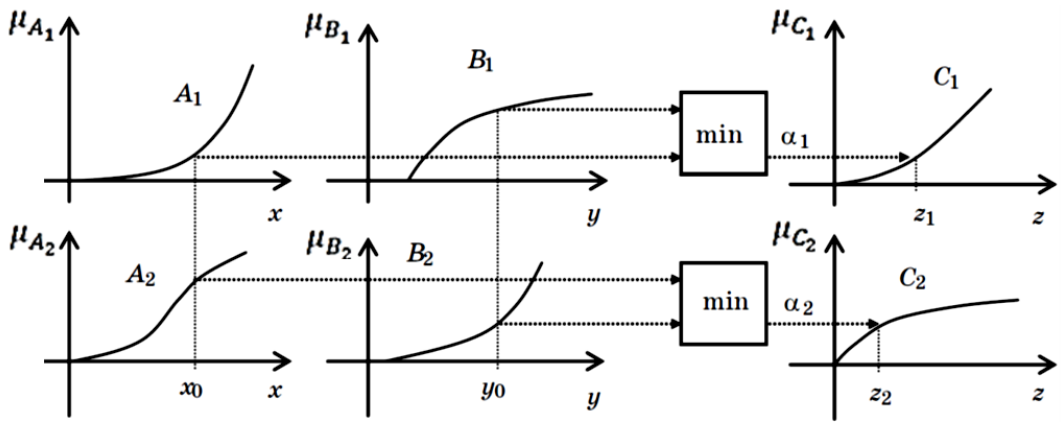


Рисунок 9.12 – Схема нечіткого висновку Tsukamoto

Схема нечіткого висновку, запропонована Sugeno, відрізняється від схеми *Tsukamoto* тільки тим, що тут вихідні значення кожного правила є функціями вхідних значень:

$$z_i = a_i x_0 + b_i y_0.$$

Роботу схеми *Sugeno* для двох правил зображено рис. 9.13.

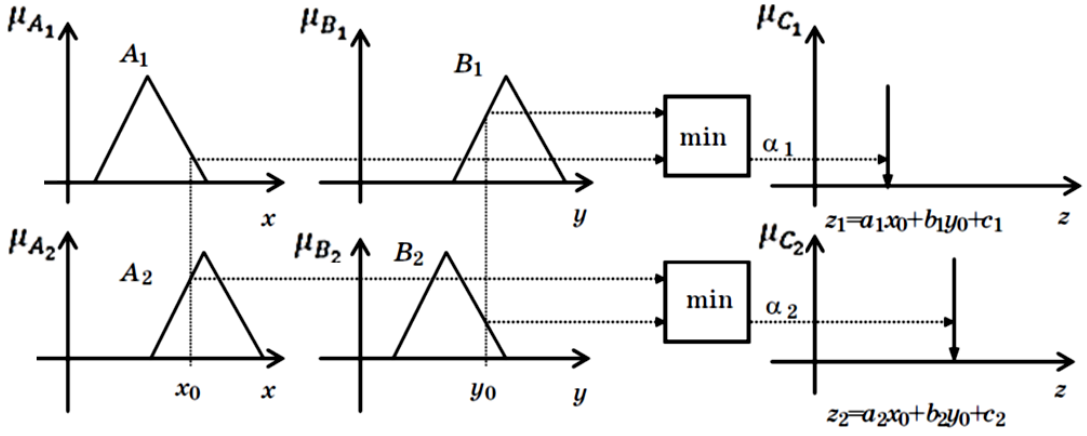


Рисунок 9.13 – Схема нечіткого висновку Sugeno

У схемах *Sugeno* та *Tsukamoto* для кожного правила виходить чітке вихідне значення, тому для отримання результуючого сигналу бази правил тут використовується формула (дискретний варіант методу центру тяжіння):

$$u = \frac{\sum_{i=1}^{N} \alpha_i z_i}{\sum_{i=1}^{N} \alpha_i}$$

У варіантах *Mamdani* та *Larsen* виходом кожного правила є нечітка множина, тому тут потрібно виконати операцію агрегування - об'єднання

безлічі модифікованих висновків правил в одну нечітку множину.
Тут можливі два варіанти:

а) розглядається логічна сума функцій приналежності висновків окремих правил, якій відповідає оператор *max*. Це варіант, що найчастіше використовується (рис. 9.14, а).

$$\mu_{\Sigma}(z) = max\left[\mu_{c_1}(z), \mu_{c_2}(z), \dots, \mu_{c_n}(z)\right], \qquad z \in Z.$$

При цьому можуть «поглинатися» укладання правил із низькими α;
б) розглядається обмежена арифметична сума функцій приналежності (рис. 9.14, б). Цей спосіб дозволяє врахувати внесок висновків правил із низькими значеннями α.

$$\mu_{\Sigma}(z) = min\left[1, \mu_{c_1}(z) + \mu_{c_2}(z) + \dots + \mu_{c_n}(z)\right], \qquad z \in Z.$$

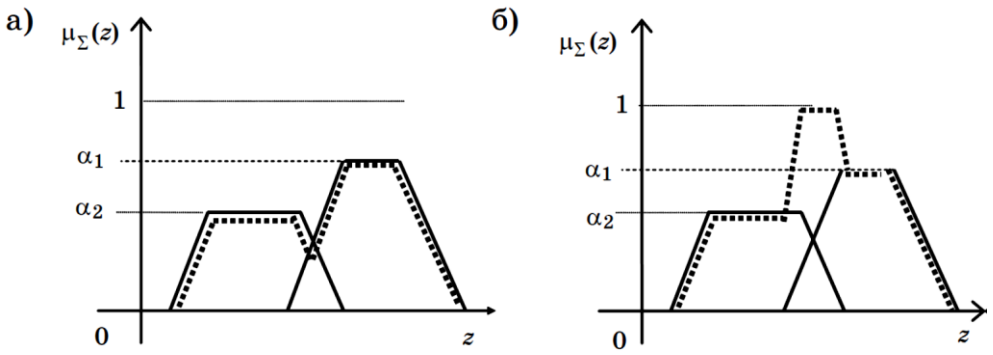


Рисунок 9.14- Варіанти агрегування висновків

Після виконання агрегування потрібно виконати операцію дефазифікації – отримати точне значення виходу всієї бази правил.

9.6 Метод дефазифікації

Дефаззифікацією (англ. defuzzification) називається процедура перетворення нечіткої множини в чітке число.
Існує велика кількість методів дефазифікації, але на практиці найчастіше застосовують кілька методів, описаних нижче.

Нехай є нечітка множина *A*, визначена на базовій шкалі *X*, з функцією $\mu_A(x)$. Тоді для перетворення їх у число можна використовувати такі методи.

1. *Метод центру тяжкості* (англ. centroid або center-of-gravity (COG)). Цей метод, як випливає з його назви, полягає у обчисленні центру тяжіння плоскої фігури, обмеженою функцією приналежності нечіткої множини та віссю координат:

$$COG(A) = \frac{\int_X (\mu_A(x)x)dx}{\int_X \mu_A(x)dx}.$$

2. *Дискретний метод центру тяжкості* (або індексний – англ. discret center-of-gravity (DCOG)) полягає у тому, що універсальна множина – дискретна. Відповідно формула для розрахунку центру ваги має вигляд

$$DCOG(A) = \frac{\sum_{i=1}^{N} \mu_A(x_i)x_i}{\sum_{i=1}^{N} \mu_A(x_i)}.$$

3. *Метод медіани* (bisector) *чи центру області* (center-of-area (COA)) полягає у тому, вибирається точка в області визначення вихідної змінної *x* така, що виявляються рівними інтеграли:

$$\int_{inf x}^{COA(A)} \mu_A(x)dx = \int_{COA(A)}^{\sup x} \mu_A(x)dx.$$

Ілюстрація даного методу наведена на рисунку 9.5.

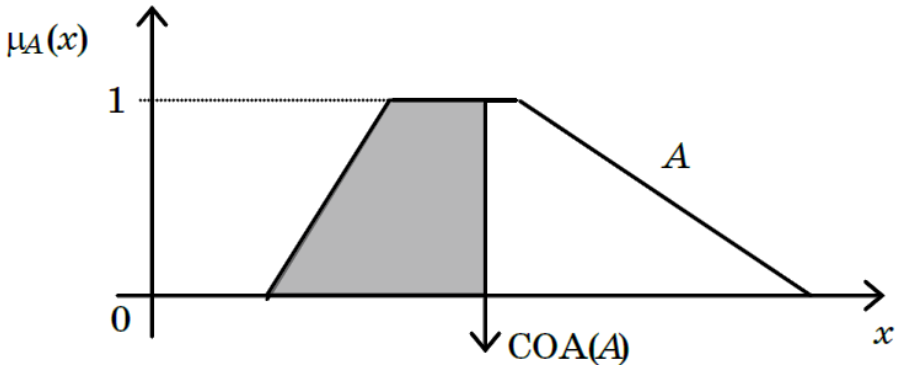


Рисунок 9.15 – Метод медіани

4. Дискретний *метод медіани* використовує формулу

$$a = \min(j)\,;\ \forall j\colon \sum_{i=1}^{j} \mu_A(x_i) \geq \frac{1}{2}\sum_{i=1}^{N} \mu_A(x_i).$$

5. *Метод центру максимумів* (center-of-maximum (COM)). Відмінність від методу центру тяжкості полягає в тому, що розглядається тільки та частина області визначення нечіткої множини, для якої значення належності максимально

$$COM(A) = \frac{\int_M (\mu_A(x)x)dx}{\int_M \mu_A(x)dx}.$$

$$M \subset X, x \in M \Rightarrow \mu_A(x) = \max\big(\mu_A(x)\big);\ \forall x \in X.$$

6. *Дискретний метод центру максимумів* (DCOM) знаходить середнє арифметичне елементів універсальної множини, що мають максимальні ступені приналежності:

$$DCOM(A) = \frac{\sum_{x_i \in M} x_i}{|M|}.$$

7. *Індексний (пороговий) метод центру тяжкості* (ICOG). Даний метод центру тяжкості застосовується не до всій нечіткий множини, а тільки до тієї його частини, в якій значення приналежності перевищує заданий поріг β:

$$COM(A) = \frac{\int_{x_i \in M} (\mu_A(x)x)dx}{\int_{x_i \in M} \mu_A(x)dx}.$$

$$M \subset X, x \in M \Rightarrow \mu_A(x) > \mu.$$

На рисунку 9.6 наведено приклад ICOG при β=0,5.

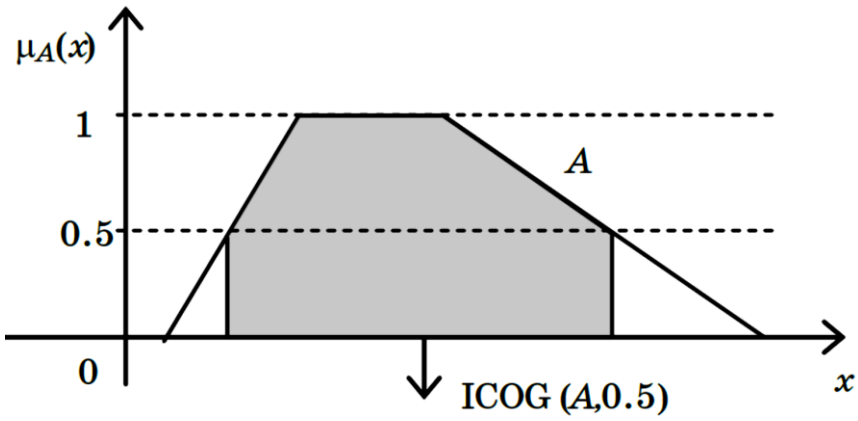


Рисунок 9.16 – Індексний метод центру тяжіння

8. *Метод середини максимуму* (mean-of-maximum (MOM)). Це пороговий метод центру ваги, у якому поріг відповідає максимуму функції приналежності (рис. 9.17):

$$MOM(A) = ICOG\big(A, hgt(A)\big).$$

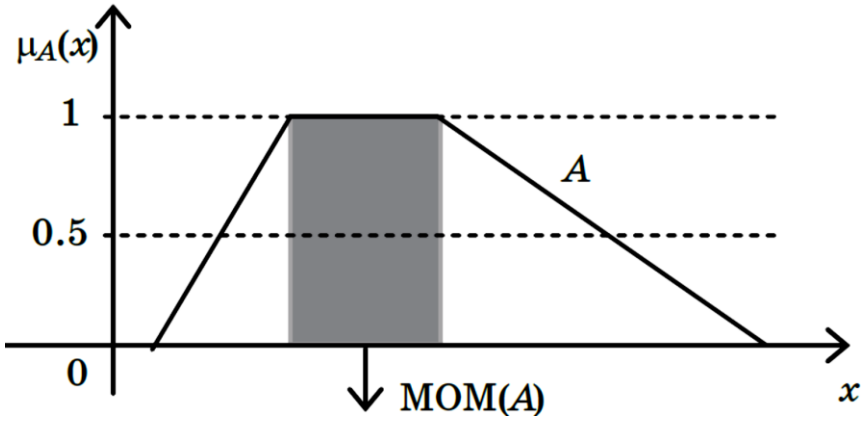


Рисунок 9.17 –Метод середини максимуму

9. *Метод найбільшого з максимумів* LOM (Largest Of Maximums):

$$LOM(A) = max(x);\ \forall x \in X;\ \mu_A(x) = max(\mu_A(x)).$$

9. *Метод найменшого з максимумів* SOM (Smallest Of Maximums):

$$LOM(A) = min(x);\ \forall x \in X;\ \mu_A(x) = max(\mu_A(x)).$$

Вочевидь, що з унімодальної нечіткої множини методи MOM, LOM і SOM збігаються.

Розглянемо реалізацію деяких методів дефазифікації у MatLab.

```
x = 0:0.1:20;
mf1 = trapmf(x,[0 2 8 12]);% трапеціподібна ф-ція приналежності 1
mf2 = trapmf(x,[5 7 12 14]);% трапеціподібна ф-ція приналежності 2
mf3 = trapmf(x,[12 13 18 19]);% трапеціподібна ф-ція приналежності 3
mf = max(0.5*mf2,max(0.9*mf1,0.1*mf3)); % агрегування висновків
figure('Tag','defuzz')
plot(x,mf,'LineWidth',3); grid;
h_gca = gca;
h_gca.YTick = [0 .5 1] ;
ylim([-1 1])
%
% Обчислення центроїда нечіткої множини.

xCentroid = defuzz(x,mf,'centroid');

% Результат дефазифікації центроїда на вихідному графіку..

hCentroid = line([xCentroid xCentroid],[-0.2 1.2],'Color','k');
tCentroid = text(xCentroid,-0.2,' centroid','FontWeight','bold');
```

```
% Метод бісектриси знаходить вертикальну лінію,
% яка розділяє нечітку множину на дві підобласті однакової площі.
% Іноді, але не завжди, вона збігається з центроїдною лінією.

xBisector = defuzz(x,mf,'bisector');
% Результат бісектриси на початковому графіку
hBisector = line([xBisector xBisector],[-0.4 1.2],'Color','k');
tBisector = text(xBisector,-0.4,' bisector','FontWeight','bold');
gray = 0.7*[1 1 1];
hCentroid.Color = gray;
tCentroid.Color = gray;

%Середній (MOM), найменший (SOM) і найбільший від максимуму
(LOM)
%
% MOM, SOM і LOM означають середнє,
% найменше та найбільше з максимуму відповідно.
% У цьому прикладі, оскільки сукупний нечітка множина має плато
% при максимальному значенні, результати дефазифікації
% MOM, SOM і LOM мають різні значення.
% Якщо сукупний нечіткий набір має унікальний максимум,
% то всі MOM, SOM і LOM дають однакове значення.

xMOM = defuzz(x,mf,'mom');
xSOM = defuzz(x,mf,'som');
xLOM = defuzz(x,mf,'lom');

% Результати MOM, SOM і LOM на початковому графіку
hMOM = line([xMOM xMOM],[-0.7 1.2],'Color','k');
tMOM = text(xMOM,-0.7,' MOM','FontWeight','bold');
hSOM = line([xSOM xSOM],[-0.7 1.2],'Color','k');
tSOM = text(xSOM,-0.7,' SOM','FontWeight','bold');
hLOM = line([xLOM xLOM],[-0.7 1.2],'Color','k');
tLOM = text(xLOM,-0.7,' LOM','FontWeight','bold');
hBisector.Color = gray;
tBisector.Color = gray;
hCentroid.Color = 'red';
tCentroid.Color = 'red';
hMOM.Color = gray;
tMOM.Color = gray;
hSOM.Color = gray;
tSOM.Color = gray;
hLOM.Color = gray;
tLOM.Color = gray;
```

На рисунку 9.18 наведено результати програмної реаілзації

деяких методів дефазифікації у MatLab

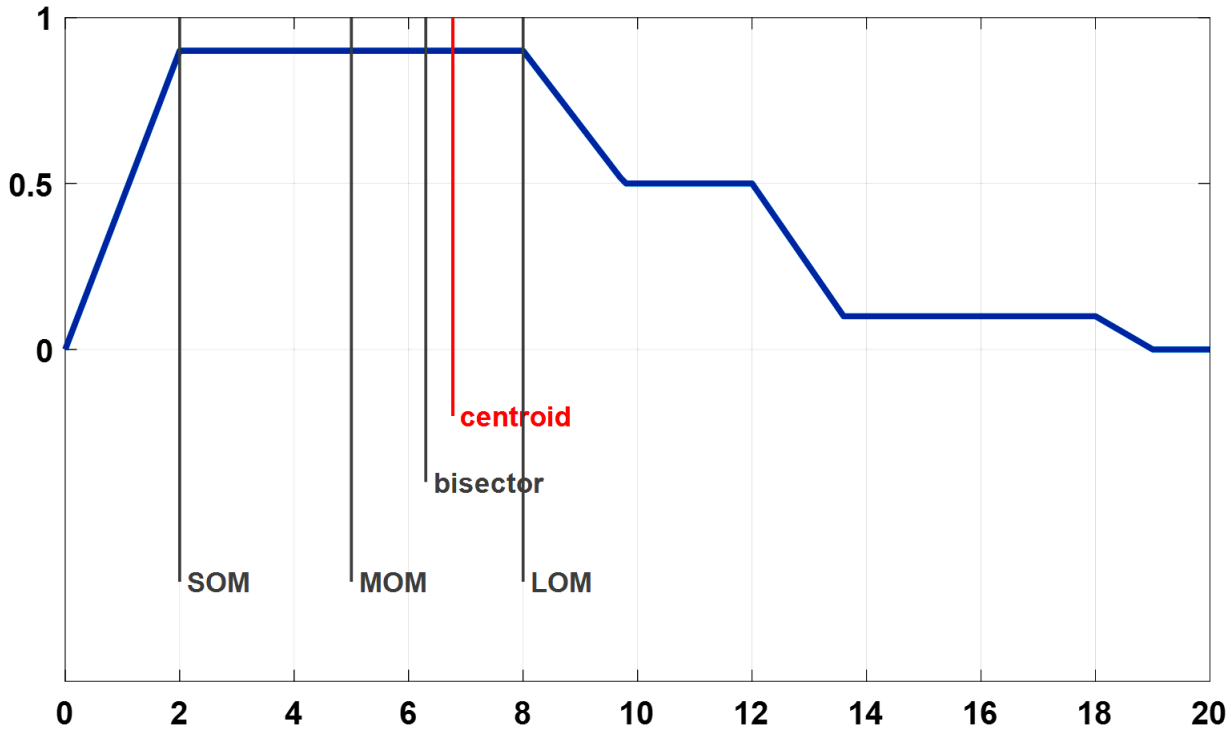


Рисунок 9.18 –Метод середини максимуму

9.7 Табличне подання основи правил

На практиці часто використовується варіант, при якому нечіткі правила мають дві посилки.

Наприклад, нехай посилки x_1, x_2 і висновок у є лінгвістичні змінні, кожна з яких описується набором з 5 термів

$$T(x_1) = T(x_2) = T(y) = \{\text{НВ}, \text{НМ}, \text{Н}, \text{ПМ}, \text{ПБ}\},$$

де НВ – «негативне велике», НМ – «негативне мале», Н – «нульове», ПВ – «позитивне велике», ПМ – позитивне мале».

Таким чином, правила мають вигляд:

$$\text{Якщо } (x_1 = \text{Н}) \text{ та } (x_2 = \text{Н}), \text{то } y = \text{ПМ}.$$

Максимальна кількість можливих правил дорівнює добутку потужностей терм-множин посилок. У цьому прикладі

$$card(\text{Т}(x_1)) \times card(\text{Т}(x_2)) = 25.$$

Якщо всі правила відомі, то вміст бази правил зручно відображати у вигляді так званої *таблиці лінгвістичних правил* (ТЛП) (рис. 9.19).

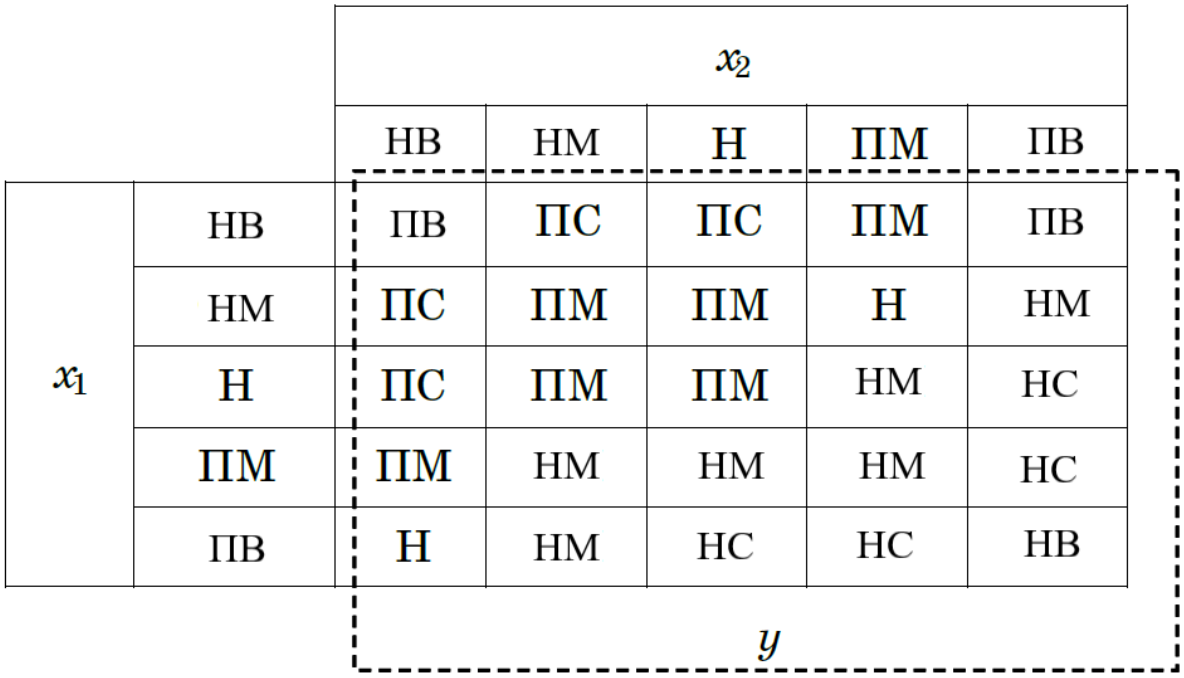


Рисунок 9.19 – Приклад таблиці лінгвістичних правил

Залежно від розв'язуваної задачі ТЛП може описувати, наприклад, динаміку об'єкта управління (x_1 – вхід об'єкта, x_2 – вихід об'єкта, y – зміна виходу об'єкта), або поведінка регулятора (x_1 – помилка, x_2 – зміна помилки, y – сигнал керування).

Механізм функціонування ТЛП ілюструє приклад, показаний на рис. 9.20 де для опису терм-множин посилок використовуються трапецеподібні функції приналежності.

Висновок різних нечітких правил можуть збігатися. Зафарбовані ділянки на рис. 9.20 означають зони взаємодії (конкуренції) сусідніх правил. У кожен час тут можуть спрацьовувати трохи більше 4 правил. При виборі іншого способу опису функцій приналежності термів (трикутні або гаусові функції) зона взаємодії правил розшириться, і поведінка нечіткої системи зміниться.

Проте найбільше значення функціонування нечіткої системи має вибір правил.

9.8 Вимоги до бази правил

Хороша база правил має задовольняти вимогам несуперечності, повноти та безперервності.

1. *Несуперечність* (англ. consistency) означає, що в базі не повинно бути правил, які мали б при подібних посилках суттєво різні висновки. Розглянемо правила з однією посилкою та одним висновком:

$$R_j: \text{Якщо } A_j, \text{то } B_j,$$

де *A* та *B* – нечіткі множини, визначені на універсальних множинах *X* та *Y*.

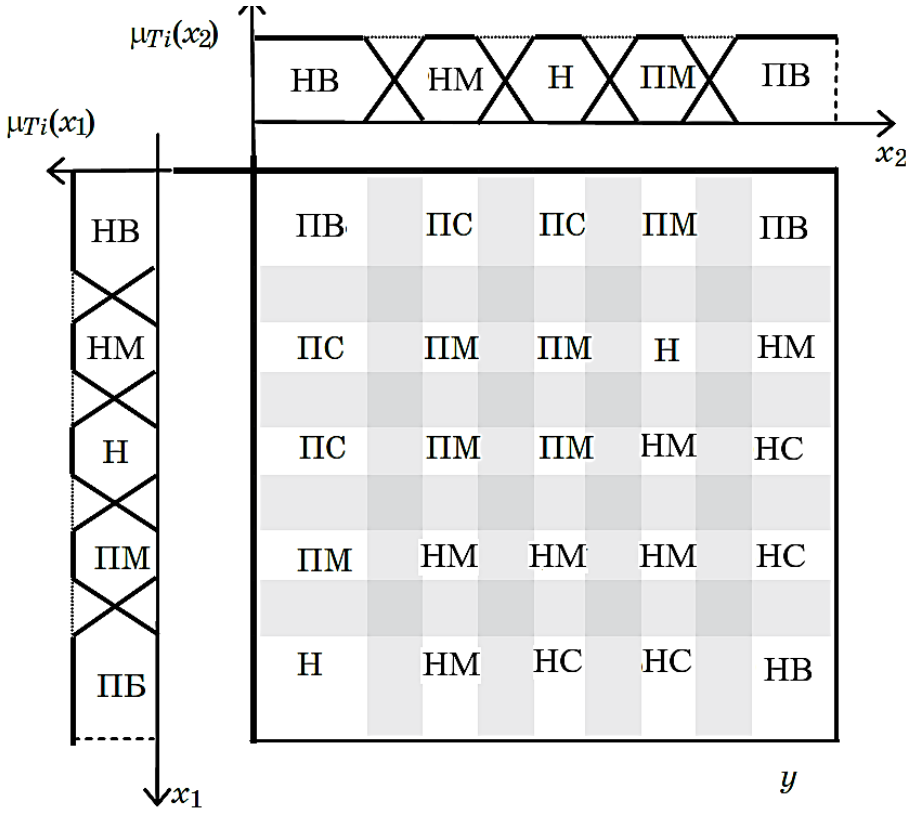


Рисунок 9.20 - Графічне подання функціонування ТЛП

Ступінь несуперечності двох правил R_j і R_k може бути оцінена за формулою

$$C(R_j, R_k) = \left| \max_x \left(\mu_{A_j}(x) \wedge \mu_{A_k}(x) \right) - \max_y \left(\mu_{B_j}(y) \wedge \mu_{B_k}(y) \right) \right|.$$

Суперечливість правила R_j стосовно всієї бази з N правил виражається формулою

$$C(R_j) = \sum_{i=1}^{N} C(R_j, R_k), i \neq j.$$

Таким чином, можна виявити «погані» правила, які не узгоджуються з рештою бази правил.

2. *Повнота* (англ. completeness) бази правил означає, що має бути «білих плям» у вхідному просторі правил. Для кожного поточного стану *X* має існувати хоча б одне керуюче правило, посилки якого мають ненульову приналежність для *X*.

Повнота бази правил може бути оцінена за формулою

$$CM(X) = \sum_{k=1}^{N_R} \left\{ \prod_{i=1}^{N_X} \mu_{A_{i,k}}(x_i) \right\},$$

де N_r – кількість правил; N_x – кількість посилок кожного правила.

Можливі такі показники повноти бази правил:

CM(X) = 0 – неповна база правил (є «біла пляма»);

CM (X) = 1 - суворо повна;

CM(X) <1 – не зовсім повна;

CM(X)> 1 - надлишкова база правил.

3. *Безперервність основи правил*. Наприклад, нехай посилки є нечіткими множинами, що утворюють нечітке розбиття базової шкали:

$$A_1 < A_2 < \ldots A_{i-1} < A_i < A_{i+1} < \ldots A_N.$$

Нехай у базі правил є два правила:

$$R_n: \text{Якщо } x_2 = A_{1,n} \text{ та } x_2 = A_{2,n}, \text{то } y = B_n$$

$$R_m: \text{Якщо } x_2 = A_{1,m} \text{ та } x_2 = A_{2,m}, \text{то } y = B_m$$

У безперервній базі правил виконуються умови:

$$\text{Якщо } A_{1,n} = A_{1,m}, \text{то } A_{2,n} \cap A_{2,m} \neq 0$$

$$\text{Якщо } A_{2,n} = A_{2,m}, \text{то } A_{1,n} \cap A_{1,m} \neq 0$$

Відповідно, при виконанні цих умов має виконуватись:

$$B_n \cap B_m \neq 0.$$

Таким чином, безперервність бази правил означає, що близьким посилкам повинні відповідати близькі висновки.

9.9. Нечітка система як універсальний апроксиматор

Нечіткі продукційні системи позбавлені недоліків традиційних продукційних систем. Система нечітких правил дозволяє апроксимувати будь-яку функцію $Y = F(X)$, задану на обмеженій безперервній множині значень U, тобто виконується

$$\sup_{X \in U} \left\| F(X - \overline{F}(X)) \right\| \leq \varepsilon,$$

де X - вектор аргументів функції; $\overline{F}(X)$ – вихід нечіткої системи, ε –

довільна мала константа.

Нечітка система, що використовує набір *N* правил

$$R_i\text{: Якщо } (x_i \in A_i)\text{і } (y_i \in B_i), \text{то } (z_i \in c_i);\ i = 1,2,\dots,N,$$

є універсальним апроксиматором за таких умов:

1) для опису посилок та висновків використовуються гауссові функції приналежності

$$A_i(x) = exp\left(-\frac{1}{2}\left(\frac{x-\alpha_{i1}}{\beta_{i1}}\right)^2\right);\ B(x) = exp\left(-\frac{1}{2}\left(\frac{y-\alpha_{i2}}{\beta_{i2}}\right)^2\right);$$

$$C_i(x) = exp\left(-\frac{1}{2}\left(\frac{z-\alpha_i}{\beta_i}\right)^2\right).$$

2) логічна операція & описується як добуток

$$[A_i(x)\&B_i(y)] = A_i(x)B_i(y)$$

3) імплікація використовується у вигляді добутку

$$[A_i(x)\&B_i(y)] \to C_i(z) = A_i(x)B_i(y)C_i(z).$$

4) для дефазифікації використовується дискретний метод центру тяжкості:

$$z = \frac{\sum_{i=1}^{N} c_i A_i B_i}{\sum_{i=1}^{N} A_i B_i}.$$

Нечітка логічна система може бути універсальним апроксиматором і за інших умов:

1) для опису посилок та висновків використовуються симетричні трикутні функції:

$$A_i(x) = \begin{cases} 1 - \dfrac{|a_i - x|}{\alpha_i}, \text{якщо } |a_i - x| \le \alpha_i, \\ 0, \quad \text{інакше;} \end{cases}$$

$$B_i(x) = \begin{cases} 1 - \dfrac{|b_i - y|}{\beta_i}, \text{якщо } |b_i - y| \le \beta_i, \\ 0, \quad \text{інакше;} \end{cases}$$

$$C_i(x) = \begin{cases} 1 - \dfrac{|c_i - z|}{\gamma_i}, \text{якщо } |c_i - z| \le \gamma_i, \\ 0, \qquad \text{інакше.} \end{cases}$$

2) логічна операція & описується як *min*:

$$[A_i(x)\&B_i(y)] = min\big(A_i(x)B_i(y)\big)$$

3) використовується імплікація *Mamdani*

$$[A_i(x)\&B_i(y)] \to C_i(z) = min\left(min\big(A_i(x)B_i(y)\big), C_i(z)\right).$$

4) для дефазифікації застосовується дискретний метод центру тяжкості:

$$z = \frac{\sum_{i=1}^{N} min\left(min\big(A_i(x)B_i(y)\big), C_i(z)\right)}{\sum_{i=1}^{N} min\big(A_i(x)B_i(y)\big)}.$$

На практиці абсолютно точна апроксимація не потрібна, достатньо наближення з деякою припустимою помилкою.

9.10 Нечіткий висновок у матричній формі

Процедура нечіткого висновку допускає компактний запис у матричній формі за таких умов:

1) для опису термів посилок використовуються трикутні функції приналежності, і терми утворюють нечітке розбиття відповідних базових множин;

2) висновки описуються синглетонами, для дефазифікації використовується дискретний метод центру тяжкості.

Розглянемо спочатку систему нечітких правил з однією посилкою та одним висновком, що реалізує відображення *f*: *X* → *Y* (див. рис. 9.21, де *F* та *DF* – позначення операцій фазифікації та дефазифікації).

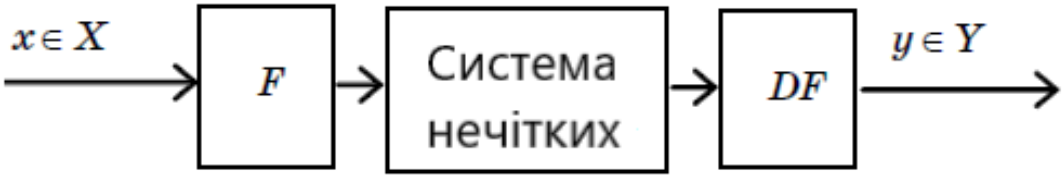


Рисунок 9.21 - Нечітка система з однією посилкою та одним висновком

При допущеннях формула дискретного методу центру тяжкості спрощується:

$$y = \frac{\sum_{i=1}^{N} \alpha_i y_i}{\sum_{i=1}^{N} \alpha_i} = \frac{\sum_{i=1}^{N} \mu_{T_i}(x) y_i}{\sum_{i=1}^{N} \mu_{T_i}(x)} = \sum_{i=1}^{N} \mu_{T_i}(x) y_i = XY.$$

Таким чином, вихідний сигнал нечіткої системи виходить як скалярний добуток двох векторів: *X* − вектора ступенів приналежності вхідного значення до різних терм вхідної змінної та *Y* − вектора центрів термів вихідної змінної.

Розглянемо приклад. Нехай для опису вхідний та вихідний змінних використовуються по 5 термів (рис. 9.22 та 9.23).

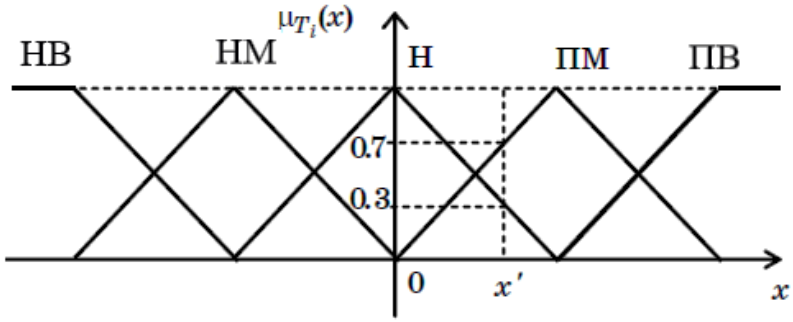


Рисунок 9.22 - Обчислення ступенів приналежності вхідної змінної

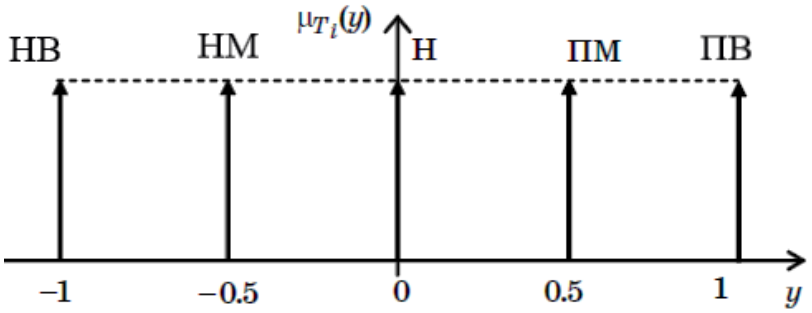


Рисунок 9.23 - Опис висновків правил

Вектор ступенів приналежності посилки (рис. 9.22) має вигляд:

$$X = \{\mu_{T_i}(x')\} = [\mu_{\text{НВ}}(x') \;\; \mu_{\text{НМ}}(x') \;\; \mu_{\text{Н}}(x') \;\; \mu_{\text{ПМ}}(x') \;\; \mu_{\text{ПВ}}(x')] = \\ = [0 \quad 0 \quad 0{,}3 \quad 0{,}7 \quad 0].$$

Нечітка система виводу використовую 5 правил

$$\text{Якщо } x = \text{НВ}, \text{то } y = \text{НВ}, \text{інакше}$$
$$\text{Якщо } x = \text{НМ}, \text{то } y = \text{НМ}, \text{інакше}$$

$$\text{Якщо } x = \text{Н}, \text{то } y = \text{Н}, \text{інакше}$$
$$\text{Якщо } x = \text{ПМ}, \text{то } y = \text{ПМ}, \text{інакше}$$
$$\text{Якщо } x = \text{ПБ}, \text{то } y = \text{ПБ}, \text{інакше}$$

Нечітким висновкам відповідає вектор

$$Y = [-1 \quad -0{,}5 \quad 0 \quad 0{,}5 \quad 1]^T$$

так що результат нечіткого висновку виходить за такою формулою:

$$y = XY == [0 \quad 0 \quad 0{,}3 \quad 0{,}7 \quad 0] \begin{bmatrix} -1 \\ -0{,}5 \\ 0 \\ -0{,}5 \\ 1 \end{bmatrix} = 0{,}35.$$

Розглянемо далі систему нечітких правил із двома посилками та одним висновком, що реалізує відображення *f*: $X_1 \times X_2 \rightarrow Y$ (рис. 9.24).

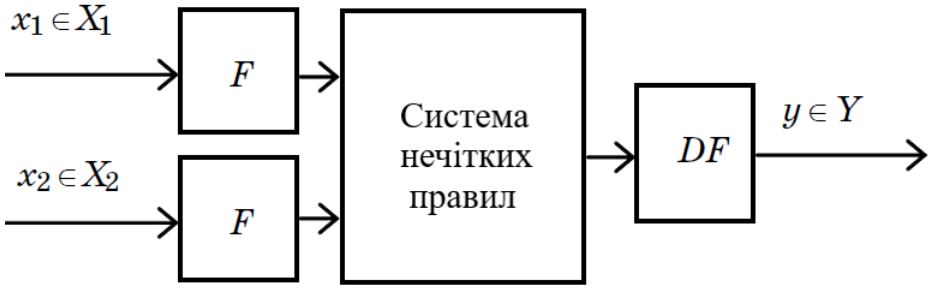


Рисунок 9.24 – Нечітка система з двома посилками та одним висновком

Нехай перша посилка та висновок описуються відповідно до рис. 9.22 та рис. 9.23, а друга посилка відповідно до рис. 9.25.

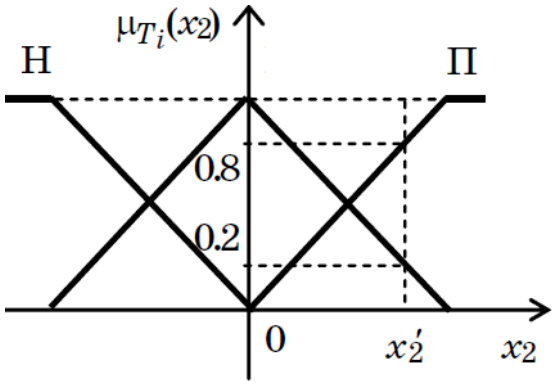


Рисунок 9.25 - Опис посилки за допомогою трьох термів

Нехай база правил має вигляд, показаний на рис. 9.26.

		x_2		
		Н	О	П
x_1	НВ	ПВ	ПВ	ПМ
	НМ	ПМ	ПМ	О
	О	ПМ	О	НМ
	ПМ	О	НМ	НМ
	ПВ	НМ	НВ	НВ

y

Рисунок 9.26 - Система з 15 нечітких правил

Оскільки висновки представлені синглетонами (рис. 9.23), праві частини правил можна описати за допомогою матриці

$$R = \begin{bmatrix} 1 & 1 & 0{,}5 \\ 0{,}5 & 0{,}5 & 0 \\ 0{,}5 & 0 & -0{,}5 \\ 0 & -0{,}5 & -0{,}5 \\ -0{,}5 & -1 & -1 \end{bmatrix}$$

Припустимо, що в певний момент часу вхідна змінна x_1 має наступний вектор ступенів приналежності

$$X_1 = \{\mu_{T_i}(x_1)\} = [\mu_{\text{НВ}}(x_1)\ \ \mu_{\text{НМ}}(x_1)\ \ \mu_0(x_1)\ \ \mu_{\text{ПМ}}(x_1)\ \ \mu_{\text{ПВ}}(x_1)] = \\ = [0 \quad 0 \quad 0{,}3 \quad 0{,}7 \quad 0].$$

Це означає, що в матриці правил *R* запускатимуться лише правила, що входять до третього та четвертого рядків.
Нехай для другої вхідної змінної вектор ступенів приналежності відповідно до рис. 9.25 має вигляд

$$X_2 = \left\{\mu_{T_j}(x_2)\right\} = [\mu_{\text{Н}}(x_2)\ \ \mu_0(x_2)\ \ \mu_{\text{П}}(x_2)] = [0 \quad 0{,}2 \quad 0{,}8].$$

Це означає, що *R* будуть запускатися тільки правила, що входять у другий і третій стовпець. Таким чином:

$$y = \frac{\sum_{i=1}^{5}\sum_{i=1}^{3}\left(\mu_{T_i}(x_1)\mu_{T_j}(x_2)\right)y_{ij}}{\sum_{i=1}^{5}\sum_{i=1}^{3}\left(\mu_{T_i}(x_1)\mu_{T_j}(x_2)\right)} =$$

$$= \frac{(0{,}3 \cdot 0{,}2) \cdot 0 + (0{,}3 \cdot 0{,}8)(-0{,}5) + (0{,}7 \cdot 0{,}2)(-0{,}5) + (0{,}7 \cdot 0{,}8)(-0{,}5)}{0{,}3 \cdot 0{,}2 + 0{,}3 \cdot 0{,}8 + 0{,}7 \cdot 0{,}2 + 0{,}7 \cdot 0{,}8} =$$

$$= \frac{-0{,}47}{1} = -0{,}47.$$

Такий же результат можна отримати, використовуючи матричний запис:

```
>> x1=[0 0 0.3 0.7 0];
>> x2=[0 0.2 0.8]';
>> R=[1 1 0.5; 0.5 0.5 0; 0.5 0 -0.5; 0 -0,5 -0,5; -0.5-1-1];
>> x1*R*x2
        ans =
              -0.47
```

Таким чином, за двох вхідних змінних результат нечіткого логічного висновку описується формулою:

$$y = X_1 R X_2.$$

9.11 Нечітка динамічна система

Алгоритм нечіткого висновку оперує правилами нечітких продукцій, у яких умови та висновки записані у формі нечітких лінгвістичних змінних.

Залежно від сенсу розв'язуваної задачі, вхідні та вихідні змінні нечіткої системи виводу можуть інтерпретуватися по-різному. Якщо вважати, що на вході нечіткої системи подається поточний стан *X*, а на виході виходить похідна цього стану *dX*/*dt*, то можна говорити про нечітку динамічну систему, один з варіантів представлення якої показаний на рис. 9.27.

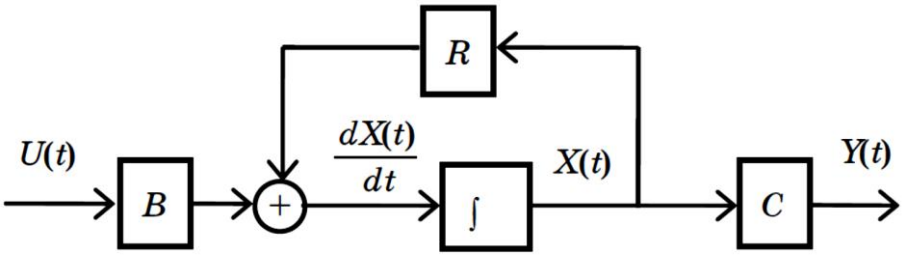


Рисунок 9.27 – Динамічна система

На рис. 9.27 оператори *R*, *B* та *C* позначають операції перетворення вхідного сигналу *U*(*t*), стану *X*(*t*) та вихідного сигналу *Y*(*t*). Ці оператори можуть бути задані за допомогою системи нечітких правил.

Допустимо, що всі змінні системи визначені на кінцевих дискретних множинах, тоді нечітка динамічна система стає дискретною.

Розглянемо вільний рух динамічної системи – за *U*(*t*) = 0. Нехай також виконується умова *Y*(*t*) = *X*(*t*). У даному випадку нечітку динамічну систему можна описати з допомогою дискретних рівнянь:

$$X_{t+1} = X_t \circ R,$$

де *t* та *t*+1 – поточний та наступний моменти часу; *R* – нечітке відношення, що визначає дію системи правил і може бути задано за допомогою матриці.

Якщо розглянути початковий стан системи X_0, будь-який інший стан може бути описаний виразом:

$$X_{t+1} = X_t \circ R_1 \circ R_2 \circ \ldots \circ R_t = X_0 \circ R^t.$$

Аналіз нечіткої динамічної системи передбачає передусім дослідження її стійкості. Для цієї мети може бути введено на розгляд поняття енергії нечіткої множини та енергії нечіткого відношення.

Енергія нечіткої множини *X* з дискретною областю визначення описується формулою

$$P(X) = \frac{1}{n}\sum_{i=1}^{n} X_i \mu_X(X_i).$$

Енергії нечіткого відношення *R* відповідає формула

$$P(R) = \frac{1}{nm}\sum_{i=1}^{n}\sum_{j=1}^{m} X_i Y_j \mu_R(X_i, Y_j).$$

Умови стійкості нечіткої динамічної системи визначаються властивостями характеристичної функції:

$$\Delta P = P(X_i) - P(X_{i-1}) = P(X_0 \circ R^t) - P(X_0 \circ R^{t-1}).$$

Оскільки $P(X_0)$ – константа, характеристичну функцію можна описати як

$$\Delta P_C(R, t) = P(R^t) - P(R^{t-1}).$$

Відповідно, система є стійкою якщо: $\Delta P_C(R,t) \le 0, t \to \infty$ і система нестійка, якщо $\Delta P_C(R,t) > 0, t \to \infty$
Система також може бути коливальною при умові:

$$|\Delta P_C(R,t)| = |\Delta P_C(R,t+\tau)|, t \to \infty$$

де τ - період коливань.

Приклад. Нехай дано базове безліч *X* = {1,2} потрібно оцінити стійкість нечіткої системи з нечітким відношенням:

$$R = \begin{vmatrix} 0{,}1 & 0{,}5 \\ 0{,}3 & 0{,}2 \end{vmatrix}.$$

Для розв'язання задачі потрібно розглянути R^t за різних *t*:

$$R^2 = R \circ R = \begin{vmatrix} 0{,}1 & 0{,}5 \\ 0{,}3 & 0{,}2 \end{vmatrix} \circ \begin{vmatrix} 0{,}1 & 0{,}5 \\ 0{,}3 & 0{,}2 \end{vmatrix} = \begin{vmatrix} 0{,}3 & 0{,}2 \\ 0{,}2 & 0{,}3 \end{vmatrix},$$
$$R^3 = R^2 \circ R = \begin{vmatrix} 0{,}3 & 0{,}2 \\ 0{,}2 & 0{,}3 \end{vmatrix} \circ \begin{vmatrix} 0{,}1 & 0{,}5 \\ 0{,}3 & 0{,}2 \end{vmatrix} = \begin{vmatrix} 0{,}2 & 0{,}3 \\ 0{,}3 & 0{,}2 \end{vmatrix},$$
$$R^4 = R^3 \circ R = \begin{vmatrix} 0{,}2 & 0{,}3 \\ 0{,}3 & 0{,}2 \end{vmatrix} \circ \begin{vmatrix} 0{,}1 & 0{,}5 \\ 0{,}3 & 0{,}2 \end{vmatrix} = \begin{vmatrix} 0{,}3 & 0{,}2 \\ 0{,}2 & 0{,}3 \end{vmatrix},$$
$$R^5 = R^4 \circ R = \begin{vmatrix} 0{,}3 & 0{,}2 \\ 0{,}2 & 0{,}3 \end{vmatrix} \circ \begin{vmatrix} 0{,}1 & 0{,}5 \\ 0{,}3 & 0{,}2 \end{vmatrix} = \begin{vmatrix} 0{,}2 & 0{,}3 \\ 0{,}3 & 0{,}2 \end{vmatrix}.$$

Визначається енергія цих ступенів нечіткого відношення:

$$P(R) = \frac{1}{2 \cdot 2}(1 \cdot 1 \cdot 0{,}1 + 1 \cdot 2 \cdot 0{,}5 + 2 \cdot 1 \cdot 0{,}3 + 2 \cdot 2 \cdot 0{,}2) = 0{,}625;$$
$$P(R^2) = \frac{1}{2 \cdot 2}(1 \cdot 1 \cdot 0{,}3 + 1 \cdot 2 \cdot 0{,}2 + 2 \cdot 1 \cdot 0{,}2 + 2 \cdot 2 \cdot 0{,}3) = 0{,}575;$$
$$P(R^3) = \frac{1}{2 \cdot 2}(1 \cdot 1 \cdot 0{,}2 + 1 \cdot 2 \cdot 0{,}3 + 2 \cdot 1 \cdot 0{,}3 + 2 \cdot 2 \cdot 0{,}2) = 0{,}55;$$
$$P(R^4) = \frac{1}{2 \cdot 2}(1 \cdot 1 \cdot 0{,}3 + 1 \cdot 2 \cdot 0{,}2 + 2 \cdot 1 \cdot 0{,}2 + 2 \cdot 2 \cdot 0{,}3) = 0{,}575;$$
$$P(R^5) = \frac{1}{2 \cdot 2}(1 \cdot 1 \cdot 0{,}2 + 1 \cdot 2 \cdot 0{,}3 + 2 \cdot 1 \cdot 0{,}3 + 2 \cdot 2 \cdot 0{,}2) = 0{,}55.$$

Визначаються характеристичні функції:

$$\Delta R_C(R,2) = P(R^2) - P(R) = 0{,}575 - 0{,}625 = -0{,}05;$$
$$\Delta R_C(R,3) = P(R^3) - P(R^2) = 0{,}55 - 0{,}575 = -0{,}025;$$
$$\Delta R_C(R,4) = P(R^4) - P(R^3) = 0{,}575 - 0{,}55 = 0{,}025;$$
$$\Delta R_C(R,5) = P(R^5) - P(R^4) = 0{,}55 - 0{,}575 = -0{,}025.$$

Таким чином, система входить у режим коливань із періодом *τ*=2.

Енергетичний підхід до оцінки стійкості нечіткої динамічної системи може бути використаний і тоді, коли розглядається вимушений рух системи, який можна описати формулою

$$X_{t+1} = X_t \circ U_t \circ R.$$

В цьому випадку нечітке відношення *R* має бути тривимірним. Проте, якщо вхідний сигнал постійний, можна розглядати вираз

$$X_{t+1} = X_t \circ (U_t \circ R) = X_t \circ P.$$

де *P* - двовимірне нечітке відношення.

Описаний підхід може бути корисним під час аналізу поведінки нечітких регуляторів.

9.12 Використання MatLab

Основний редактор системи нечіткого виведення (fuzzy inference system – FIS) викликається у MatLab командою

>> fuzzy

На рис. 9.28 показано головне вікно редактора нечіткої логічної системи.

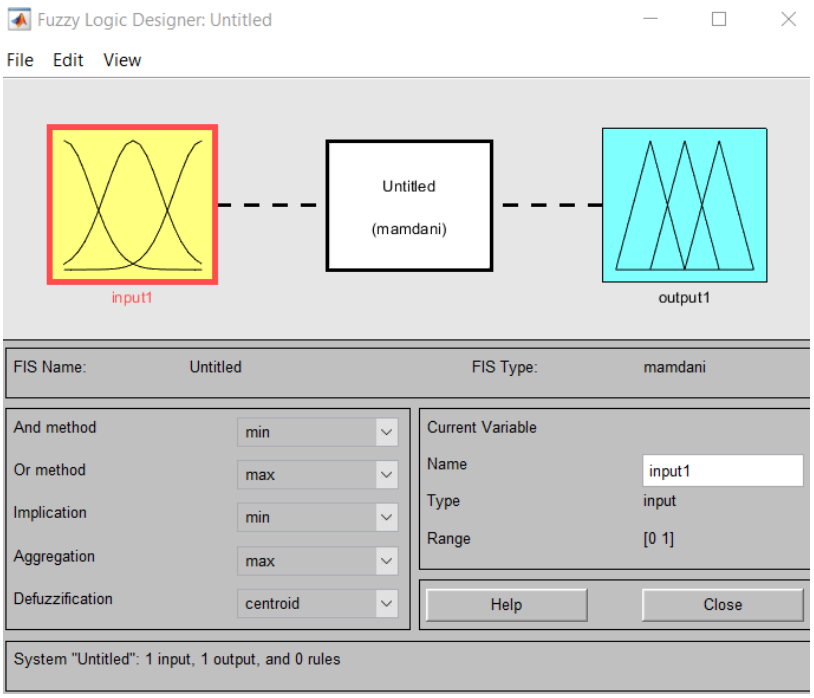


Рисунок 9.28 - Інтерфейс FIS editor

FIS editor дозволяє описати систему нечіткого логічного висновку *Mamdani* або *Sugeno*. Для створюваної системи можна вибрати вид

логічних зв'язок (*And method*) та (*Or method*), вид імплікації (*Implication*), спосіб агрегування висновків правил (*Aggregation*) та метод дефазифікації (*Defuzzification*). Система, що створюється в *FIS editor*, може бути записана в постійну або оперативну пам'ять (*Save to disk* або *Save to workspace*). Об'єкт, що запам'ятовується, має розширення **.fis*. У меню *Edit* можна додати або видалити вхідні або вихідні змінні, які беруть участь у нечітких правилах.

Для опису вхід-вихідних лінгвістичних змінних, введення правил та дослідження поведінки нечіткої логічної системи *FIS editor* викликає окремі інтерфейси, доступ до яких можливий окремими командами. Для опису логічних змінних передбачено редактора функцій приналежності, що запускається командою (або подвійним кілком лівої кнопки миші по іконці «input 1» або «output 1»)

>> mfedit

Головне вікно редактора *mfedit* показано на рис. 9.29.

Меню *File* містить звичайні команди роботи із файлами.

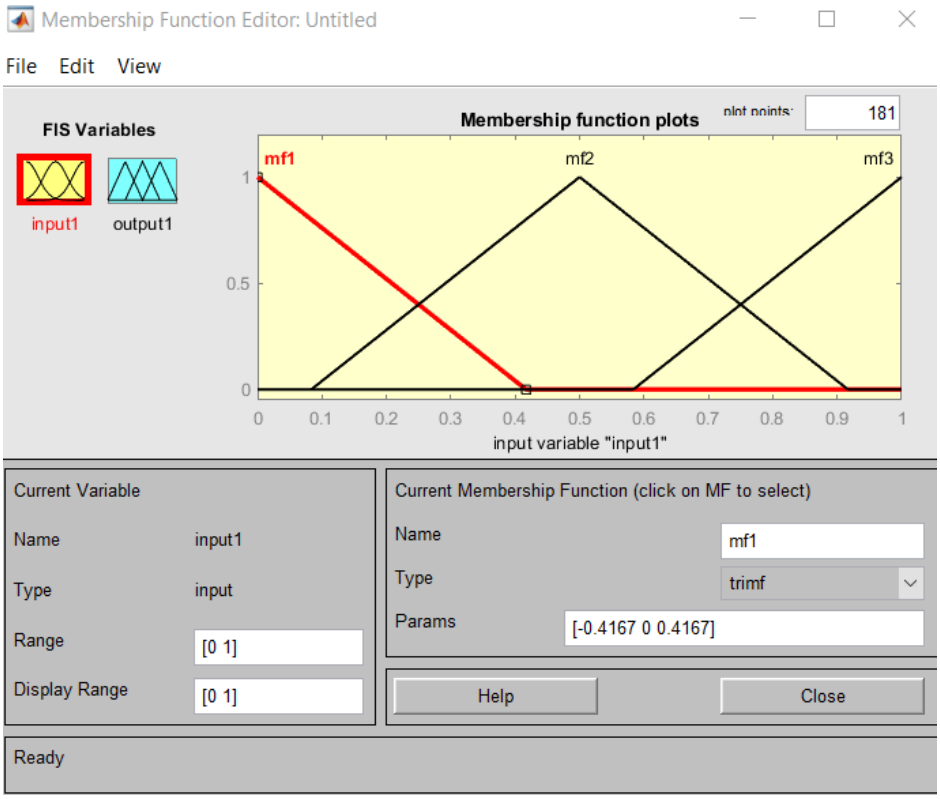


Рисунок 9.29 - Інтерфейс mfedit

Меню *Edit* дозволяє додати або видалити лінгвістичну змінну на вході або виході системи.

Ліва нижня частина екрану містить опис редагованої лінгвістичної змінної: її ім'я (*Name*), тип (*Type*, вхідна – *input*, або вихідна – *output*), розмір базової шкали (*Range*) і розмір ділянки базової шкали (*Display Range*), що відображається.

Права нижня частина екрану містить опис редагованого терма лінгвістичної змінної: його ім'я (*Name* – значення може вводитися користувачем), тип (*Type* – тип функції приналежності, значення вибирається зі списку, що випадає), параметри (*Params* – відповідають обраному типу функції приналежності, вводяться вручну за допомогою миші).

Якщо користувач описав вхідні та вихідні лінгвістичні змінні, то він може розпочати опис правил (рис. 9.30).

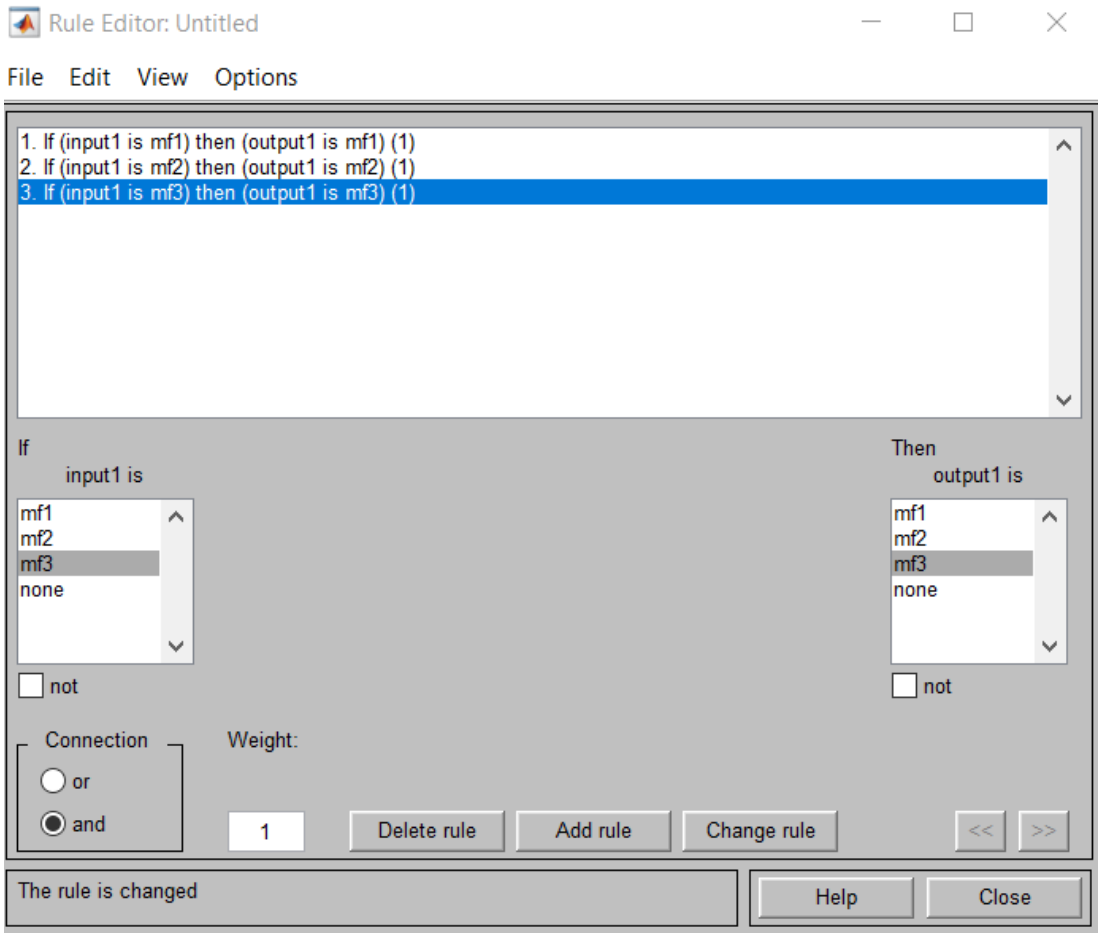


Рисунок 9.30 – Інтерфейс ruleedit

Для виклику інтерфейсу редагування правил можна використовувати команду

```
>> ruleedit
```

або рядок *Rules*… у меню *Edit* інтерфейсу *mfedit*.

Редагування правил полягає у зв'язуванні терма лінгвістичної змінної – посилки та терма лінгвістичної змінної – висновку.

У верхній частині екрана розміщено вікно, в якому відображаються введені правила. Нижче ліворуч і праворуч показані назви наявних термів посилки (*If* частина) та укладання (*Then* частина). Посилка може входити в правило з запереченням (*not*). Посилки можуть бути пов'язані (connection) за допомогою операції *or* або *and*. Висновок може входити

в правило з запереченням.

Для видалення, додавання або зміни правила використовуються відповідно кнопки *Delete rule, Add rule, Change rule*.

Введене правило може бути забезпечене ваговим коефіцієнтом (*Weight*), за допомогою якого можна описати істинність (важливість) правила у поточній базі правил.

Меню *View* інтерфейсу *ruleedit* дозволяє вирішити дві задачі:

− за допомогою пункту Rules переглянути роботу системи нечіткого логічного висновку за різних вхідних даних (рис. 9.31);

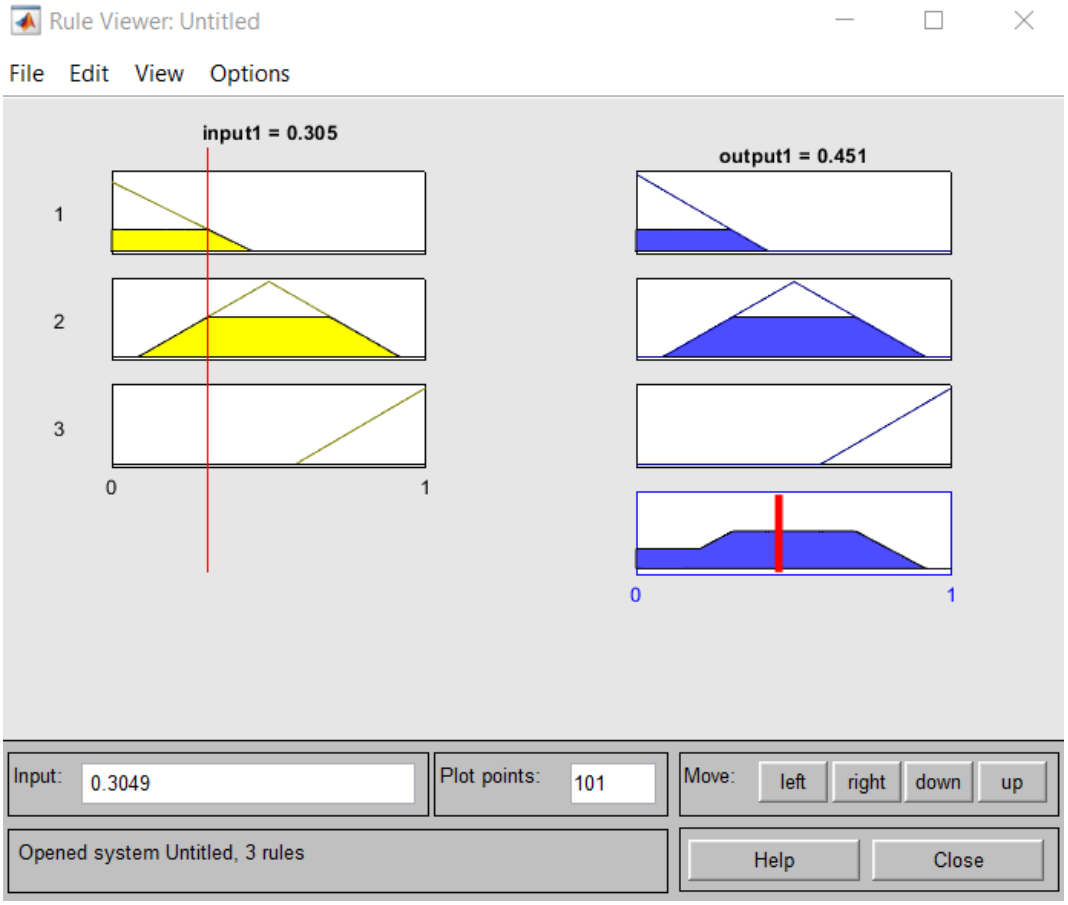


Рисунок 9.31 - Робота системи нечіткого логічного висновку

− за допомогою пункту *Surface* можна переглянути керуючу поверхню нечіткої логічної системи, яка виходить при подачі на вхід системи різних допустимих значень (рис. 9.32).

На рис. 9.31 зліва показано вхідне (чітке) значення, задане вертикальною лінією. Справа показаний вихід системи (жирна вертикальна лінія). Вихідне значення визначається правилами, термами та використовуваним методом дефазифікації.

Інтерфейс візуалізації процедури нечіткого логічного висновку можна також викликати одній з команд, які звертаються відповідно до постійної або оперативної пам'яті:

>> ruleview(fis)
>> ruleview('filename')

Показана на рис. 9.32 керуюча поверхня є плоскою кривою, оскільки правила мають одну посилку. При двох посилках з'являється керуюча поверхня, а при більшій кількості посилок можна розглядати лише проекції керуючої поверхні. Форма керуючої поверхні також визначається правилами, термами та методом дефазифікації, що використовується.

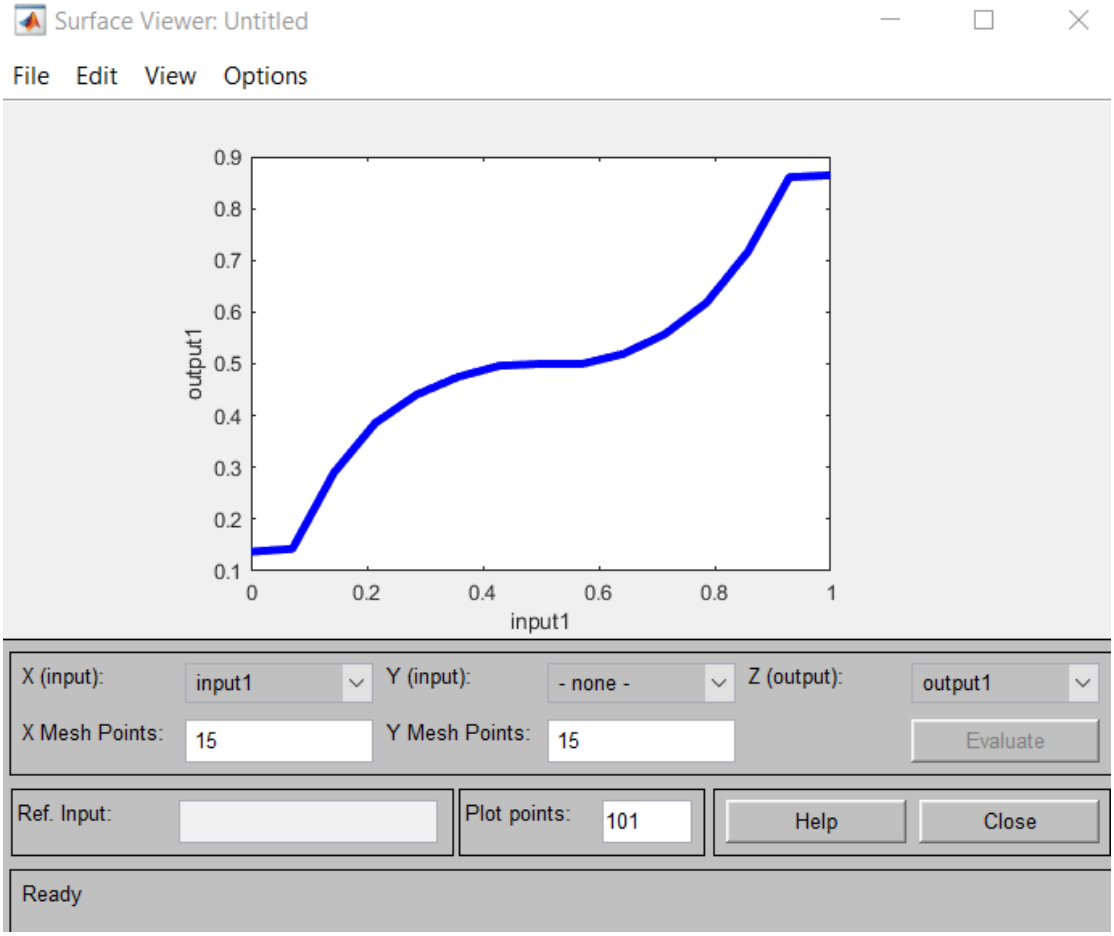


Рисунок 9.32 - Керуюча поверхня системи нечіткого логічного висновку

За допомогою команди

>> gensurf(fis)

Можна отримати керуючу поверхню у графічному форматі, зручному для запам'ятовування.

У системі MatLab до виконання операції дефазифікації служить команда defuzz. Вона має формат:

>> out = defuzz(x,mf,type)

де out - Вихідна величина (число); x - базова множина; mf –

використовувана функція приналежності; type – метод дефазифікації (centroid, bisector, mom, som, lom).
Наприклад:

```
>> x = -10: 0.1: 10;
mf = trapmf(x,[-10 -8 -4 7]);
xx = defuzz(x,mf,'centroid')
xx =
-3.2857
```

За допомогою команди

```
>> defuzzdm
```

у MatLab можна отримати графічну ілюстрацію, що показує як змінюється вихідна (чітка) величина залежно від методу дефазифікації, що використовується (рис. 9.18).

Приклад. Розглянемо опис системи нечіткого логічного висновку. Нехай потрібно описати правила керування вентилятором, що зв'язують поточну температуру (*t*, градуси) у кімнаті та швидкість обертання лопат вентилятора (*N*, оберти за секунду).

1. Якщо холодно (cold), вентилятор не працює (stop).
2. Якщо прохолодно (cool), вентилятор працює повільно (slow).
3. Якщо комфорт (normal), вентилятор працює середньо (medium).
4. Якщо жарко (warm), то вентилятор працює швидко (fast).
5. Якщо спекотна (hot), то вентилятор працює дуже швидко (blast).

Таким чином, нечітка логічна система матиме одну вхідну та одну вихідну змінні, її поведінка описуватиме 5 правил.

За допомогою *FIS editor* зробимо опис термів та правил, та завантажимо їх на диск та в робочу пам'ять FIS об'єкта з ім'ям vent.fis.

При налаштуванні опису вхідної логічної змінної прийнято:

- діапазон змін температур у приміщенні *x* від 0 до 50 °C;
- для змінної «холодно» (COOD) визначено діапазон температур від 0 до 15 °C з описом у вигляді z-подібної функції приналежності, яка набуває значення $\mu(x_{cood})=1$ при 0 °C;
- для змінної «прохолодно» (COOL) визначено діапазон температур від 15 до 25 °C з описом у вигляді гаусовської функції приналежності, яка набуває значення $\mu(x_{cool})=1$ при 15 °C;
- для змінної «комфорт» (NORMAL) визначено діапазон температур від 12 до 32 °C з описом у вигляді гаусовської функції приналежності, яка набуває значення $\mu(x_{normal})=1$ при 21 °C;
- для змінної «жарко» (WARM) визначено діапазон температур від 20 до 40 °C з описом у вигляді гаусовської функції приналежності, яка набуває значення $\mu(x_{warm})=1$ при 28 °C;

- для змінної «спекотна» (HOT) визначено діапазон температур від 28 до 50 °С з описом у вигляді s-подібної функції приналежності, яка набуває значення $\mu(x_{hot})=1$ від 40 °С.

Результати налаштуванні опису вхідної лінгвістичної змінної наведено на рисунку 9.33.

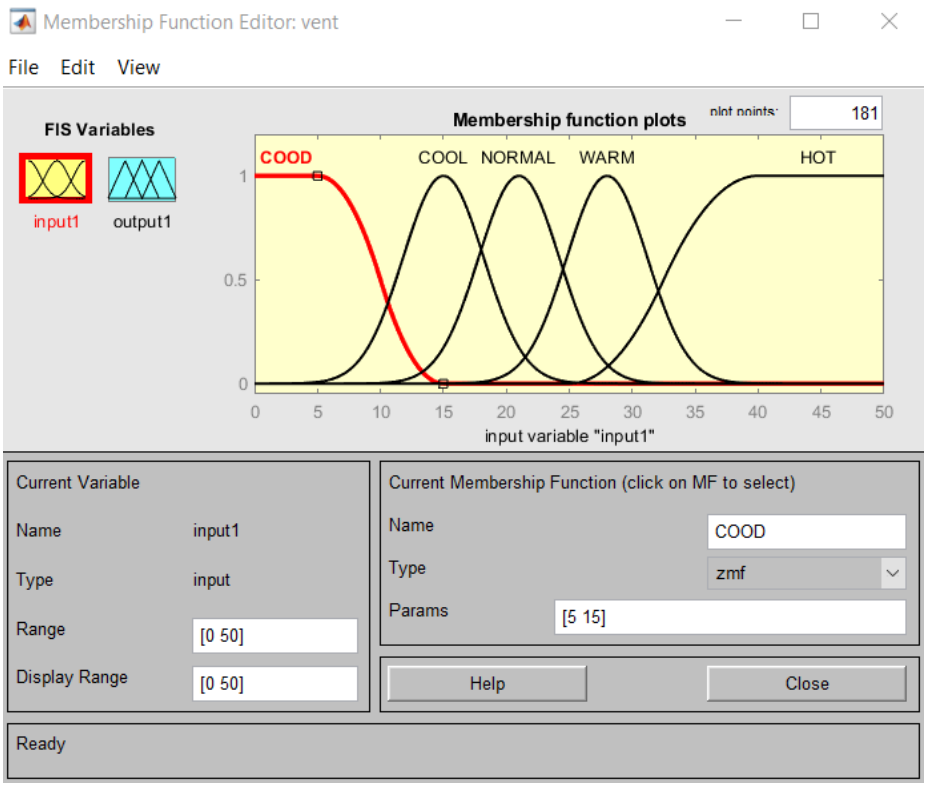


Рисунок 9.33 - Результати налаштуванні опису вхідної лінгвістичної змінної x

Отримати графічний опис вхідної лінгвістичної змінної можна за допомогою команд

```
fis = readfis('vent');
figure(1)
plotmf(fis,'input',1); grid
```

На рис. 9.34 показаний графік функцій приналежності вхідної лінгвістичної змінної.

При налаштуванні опису вихідної лінгвістичної змінної прийнято гаусовські функції приналежності:

- діапазон змін швидкості обертання вентилятора *y* від 0 до 100 $хв^{1}$;
- для змінної «STOP» значення $\mu(x_{stop})=1$ при 0 ° $хв^{-1}$;
- для змінної «SLOW» значення $\mu(x_{slow})=1$ при 25 ° $хв^{-1}$;
- для змінної «MEDIUM» значення $\mu(x_{medium})=1$ при 50 ° $хв^{-1}$;

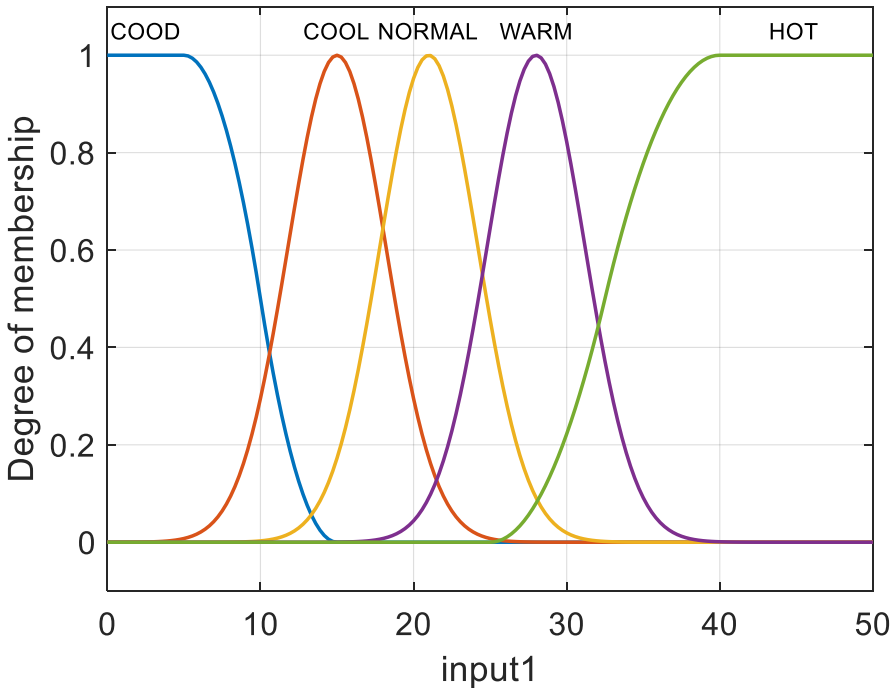


Рисунок 9.34 – Графіки лінгвістичного опису вхідної змінної

- для змінної «FAST» значення $\mu(x_{fast})=1$ при 75 ° $хв^{-1}$;
- для змінної «BFAST» значення $\mu(x_{bfast})=1$ при 100 ° $хв^{-1}$;

Результати налаштування вихідної лінгвістичної змінної наведено на рисунку 9.35.

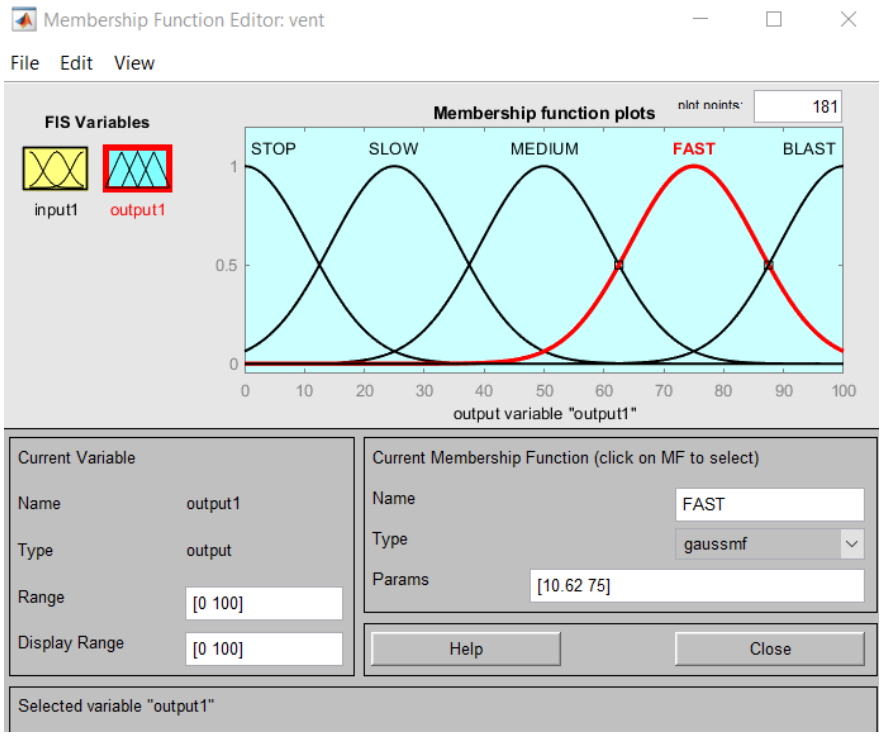


Рисунок 9.35 - Результати налаштуванні опису вихідної лінгвістичної змінної y

Отримати графічний опис вихідної змінної за допомогою команд (рис. 9.36)

```
fis = readfis('vent');
figure(2)
plotmf(fis,'output',1);grid
```

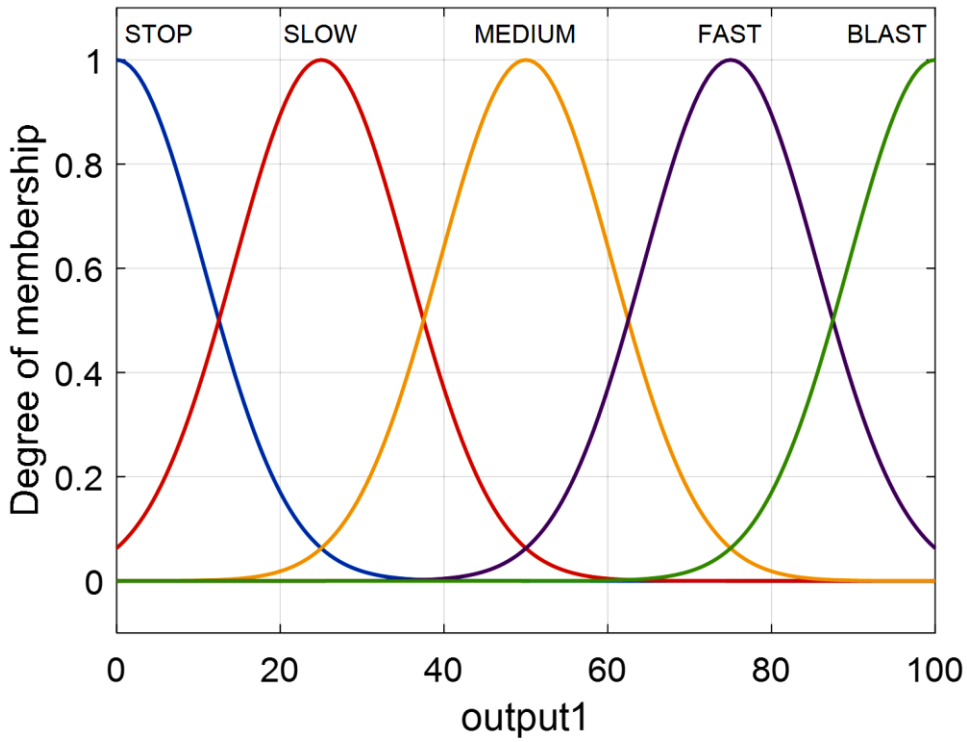


Рисунок 9.36 – Графіки лінгвістичного опису вихідної змінної

Для отримання опису правил нечіткої логічної системи можна скористатися командою

```
fis = readfis('vent');
showrule(fis,[1 2 3 4 5],'symbolic')
```

```
ans =
  5×43 char array
    '1. (input1==COOD) => (output1=STOP) (1)    '
    '2. (input1==COOL) => (output1=SLOW) (1)    '
    '3. (input1==NORMAL) => (output1=MEDIUM) (1)'
    '4. (input1==WARM) => (output1=FAST) (1)    '
    '5. (input1==HOT) => (output1=BLAST) (1)    '
```

Опис поверхні, що керує, можна отримати за допомогою команд (рис. 9.37):

```
fis = readfis('vent');
figure(3)
gensurf(fis)
```

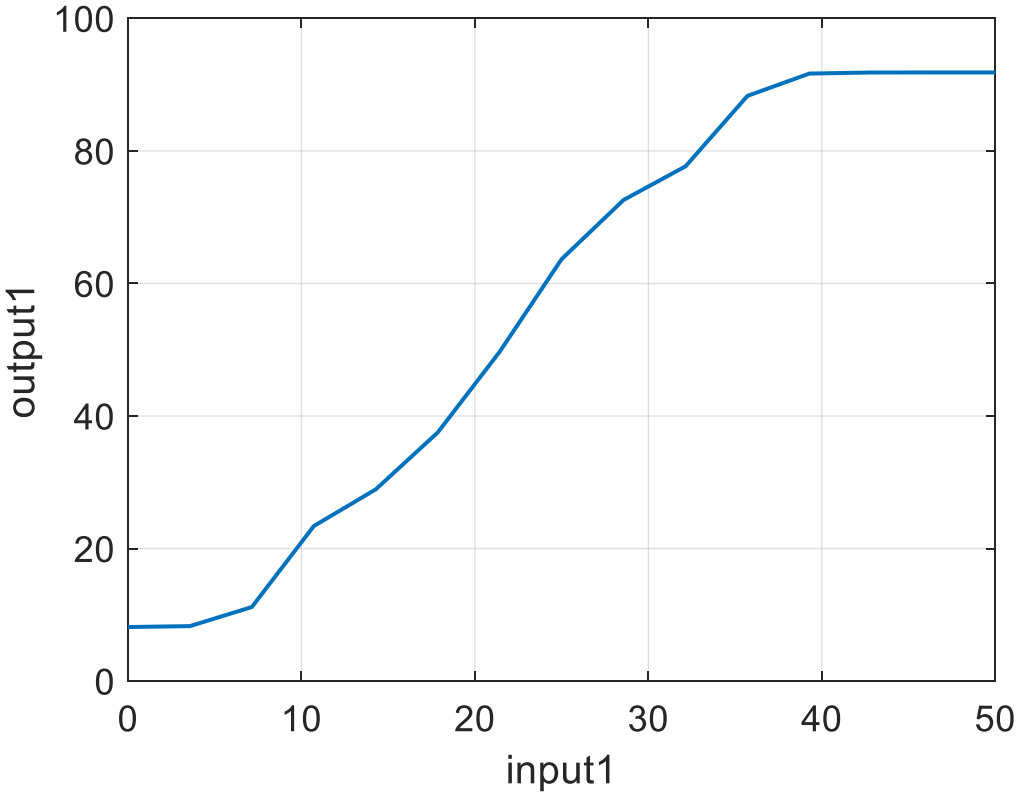


Рисунок 9.37 – Закон керування вентилятором

Графічний опис основних параметрів нечіткої системи виведення можна отримати за допомогою команд (рис. 9.38)

```
fis = readfis('vent');
figure(4)
plotfis(fis)
```

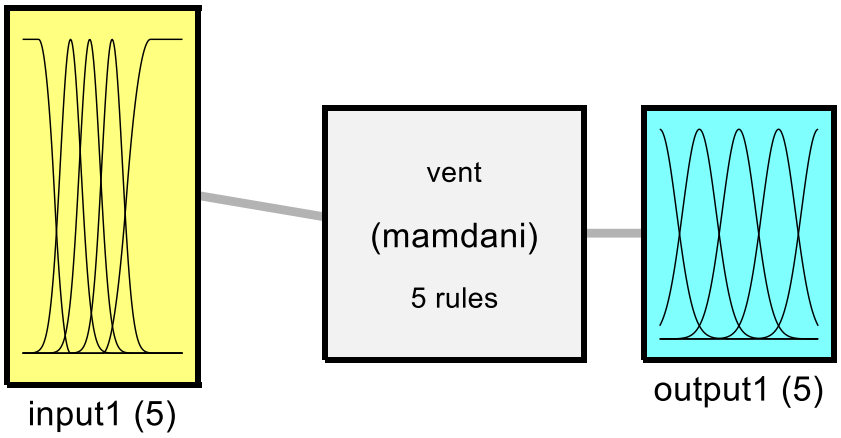


Рисунок 9.38 – Струкрура нечіткої системи для керування вентилятором

Повний опис усіх властивостей *FIS* об'єкта в текстовому форматі можна отримати за допомогою команд

```
fis = readfis('vent');
showfis (fis);
```

Зауважимо, що для опису нечіткої логічної системи не обов'язково використовувати *FIS Editor* з інтерфейсами, що входять до нього. Усі необхідні операції можна виконати за допомогою командного рядка або програми *MatLab*.

Приклад. Нехай потрібно вирішити, як залежить ціна виробу (Profit) в залежності від кваліфікації робітника (qualification) та від якості матеріалу продукції (product material).

а) Додаємо та описуємо лінгвістичні змінні:

1. Описуються вхідні лінгвістичні змінні:

Кваліфікації робітника (qualification) = [низька (low), середня (average), висока (high)]

Матеріал продукції (product material) = [низької якості (low quality), високої якості (high quality)]

2. Описується вихідна лінгвістична змінна:

Ціна продукції (Profit) = [дешева (cheap), середня (average), дорога (expensive)]

б) Налаштовуємо лінгвістичні змінні:

1) При налаштуванні опису вхідної лінгвістичної змінної «product material» прийнято трапецієподібні функції приналежності:

- діапазон лінгвістичної змінної «product material» від 0 до 10 балів;
- для змінної «low» значення $\mu(x_{low})$=1 при 0 балах;
- для змінної «average» значення $\mu(x_{average})$=1 при 5 балах;
- для змінної «high» значення $\mu(x_{high})$=1 при 10 балах.

2) При налаштуванні опису вхідної лінгвістичної змінної «qualification» прийнято гаусовські функції приналежності:

- діапазон лінгвістичної змінної «qualification» від 0 до 10 балів;
- для змінної «low quality» значення $\mu(x_{low\ q})$=1 від 0 до 1 балах, $\mu(x_{low\ q})$=0 при 3 балах;
- для змінної «high quality» значення $\mu(x_{high\ q})$=1 від 9 до 10 балах, $\mu(x_{low\ q})$=0 при 7 балах;

3) При налаштуванні опису вихідної лінгвістичної змінної «Profit» прийнято трикутна функції приналежності:

- діапазон лінгвістичної змінної «Profit» від 0 до 30 у.о.;
- для змінної «cheap» значення $\mu(x_{cheap})$=1 при 5 у.о., діапазон визначення від 0 до 10 у.о.;
- для змінної «average» значення $\mu(x_{average})$=1 при 15 у.о., діапазон визначення від 10 до 20 у.о.;
- для змінної «expensive» значення $\mu(x_{expensive})$=1 при 25 у.о., діапазон визначення від 20 до 30 у.о.

в) Описуються лінгвістичні правила роботи системи.

Нехай роботу системи описуються наступні правила:

1. Якщо (qualification==low) або (product material==low quality) то (profit=cheap)'
2. Якщо (qualification==low) або (product material==high quality) то (profit=average)
3. Якщо (qualification==average) або (product material==low quality) то (profit=average)
4. Якщо (qualification==average) або (product material==high quality) то (profit=expensive)
5. Якщо (qualification==high) або (product material==low quality) то (profit=average)
6. Якщо (qualification==high) або (product material==high quality) то (profit=expensive)

Створимо нечітку систему з використання командних строк MatLab.

Варіант програми 1.

Перша команда створює новий *FIS* – об'єкт (це короткий формат команди, багато параметрів прийнято за замовчуванням):

```
fis = mamfis('Name','Prof');

% Додається перша вхідна змінна
fis = addInput(fis,[0 10],'Name','qualification');
fis = addMF(fis,'qualification','gaussmf',[1.5 0],'Name','low');
fis = addMF(fis,'qualification','gaussmf',[1.5 5],'Name','average');
fis = addMF(fis,'qualification','gaussmf',[1.5 10],'Name','high');

% Додається друга вхідна змінна
fis = addInput(fis,[0 10],'Name','product material');
fis = addMF(fis,'product material','trapmf',[-2 0 1 3],'Name','low quality');
fis = addMF(fis,'product material','trapmf',[7 9 10 12],'Name','high quality');

% Додається вихідна змінна
fis = addOutput(fis,[0 30],'Name','profit');
fis = addMF(fis,'profit','trimf',[0 5 10],'Name','cheap');
fis = addMF(fis,'profit','trimf',[10 15 20],'Name','average');
fis = addMF(fis,'profit','trimf',[20 25 30],'Name','expensive');

% Додаються правила
% Кожен рядок масиву містить одне правило в такому форматі.
% Стовпець 1 - Індекс функції належності для першого введення
% Стовпець 2 - Індекс функції належності для другого входу
% Стовпець 3 - Індекс функції належності для виведення
% Стовпець 4 - Вага правила (від 0 до 1)
% Стовпець 5 - Нечіткий оператор (1 для "І", 2 для "АБО")
```

```
ruleList = [ ...
1 1 1 1 2
1 2 2 1 2
2 1 2 1 2
2 2 3 1 2
3 1 2 1 2
3 2 3 1 2];
fis = addRule(fis,ruleList);
showrule(fis,[1 2 3 4 5 6],'symbolic')
gensurf(fis)
```

Варіант програми 2.

```
fis = mamfis('Name','Proff2');

% Додається та налаштується перша вхідна змінна.
% У цьому випадку необхідно створить стандартний об'єкт fisvar
% і вказати його властивості за допомогою покрокової нотації.

fis.Inputs(1) = fisvar;
fis.Inputs(1).Name = "qualification";
fis.Inputs(1).Range = [0 10];

% Задаються функції приналежності для першої вхідної змінної.
% Для кожного MF створіть об’єкт fismf і встановлюються властивості
% за допомогою покрокової нотації.

fis.Inputs(1).MembershipFunctions(1) = fismf;
fis.Inputs(1).MembershipFunctions(1).Name = "low";
fis.Inputs(1).MembershipFunctions(1).Type = "gaussmf";
fis.Inputs(1).MembershipFunctions(1).Parameters = [1.5 0];
fis.Inputs(1).MembershipFunctions(2) = fismf;
fis.Inputs(1).MembershipFunctions(2).Name = "average";
fis.Inputs(1).MembershipFunctions(2).Type = "gaussmf";
fis.Inputs(1).MembershipFunctions(2).Parameters = [1.5 5];
fis.Inputs(1).MembershipFunctions(3) = fismf;
fis.Inputs(1).MembershipFunctions(3).Name = "high";
fis.Inputs(1).MembershipFunctions(3).Type = "gaussmf";
fis.Inputs(1).MembershipFunctions(3).Parameters = [1.5 10];

% Додається та налаштується друга вхідна змінна.
% Для цієї змінної вкажіть назву та
% діапазон під час створення об’єкта fisvar.
fis.Inputs(2) = fisvar([0 10],'Name',"product material");
```

```
% Вказується функції приналежності для другого входу.
% Для кожного MF вказується назва, тип і параметри
% під час створення об'єкта fismf.

fis.Inputs(2).MembershipFunctions(1) = fismf("trapmf",[-2 0 1 3],...
                                    'Name',"low quality");
fis.Inputs(2).MembershipFunctions(2) = fismf("trapmf",[7 9 10 12],...
                                    'Name',"high quality");

% Так само додається та налаштовується вихідна змінна
% та її функція приналежності.

fis.Outputs(1) = fisvar([0 30],'Name',"profit");

% Вказується вихідні функції приналежності
% за допомогою вектора об’єктів fismf.

mf1 = fismf("trimf",[0 5 10],'Name',"cheap");
mf2 = fismf("trimf",[10 15 20],'Name',"average");
mf3 = fismf("trimf",[20 25 30],'Name',"generous");
fis.Outputs(1).MembershipFunctions = [mf1 mf2 mf3];

% Створюються правила для нечіткої системи.
% Для кожного правила добудується об'єкт fisrule.
% Потім вказуються правила за допомогою вектора цих об'єктів.
% Під час створення об'єкта fisrule з використанням числових значень
% необхідно вказати кількість вхідних змінних.

rule1 = fisrule([1 1 1 1 2],2);
rule2 = fisrule([1 2 2 1 2],2);
rule3 = fisrule([2 1 2 1 2],2);
rule4 = fisrule([2 2 3 1 2],2);
rule5 = fisrule([3 1 2 1 2],2);
rule6 = fisrule([3 2 3 1 2],2);

rules = [rule1 rule2 rule3 rule4 rule5 rule6];

% Перш ніж додати свої правила до нечіткої системи,
% необхідно повинні оновити їх, використовуючи дані в об'єкті FIS.
% Правила оновлюються за допомогою функції оновлення та
% додавання їх до нечіткої системи.

rules = update(rules,fis);
fis.Rules = rules;
```

```
% Оцінюється система нечіткого логічного висновку
% Для цього результат нечіткої системи  для
% заданої комбінації введення, скористаємось командою evalfis.
% Наприклад, обчисліть fis, використовуючи значення вхідних змінних
1 і 2.

evalfis(fis,[1 2])

% Можливо оцінити кілька вхідних комбінацій за допомогою масиву,
% де кожен рядок представляє одну вхідну комбінацію.

inputs = [3 5;
          2 7;
          3 1];
evalfis(fis,inputs)

% Перевіримо введені правила командою
showrule(fis,[1 2 3 4 5 6],'symbolic')
% будуємо поверхню керування системи (див. рис 9.39)
gensurf(fis)
% Графічний опис властивості описаної нечіткої системи (див. рис. 9.
40.)
plotfis(fis)
```

На рисунку 9.39 наведена поверхня керування за результатами програмної побудови нечіткої системи. На рисунку 9.40 зображено графічний опис властивості описаної нечіткої системи.

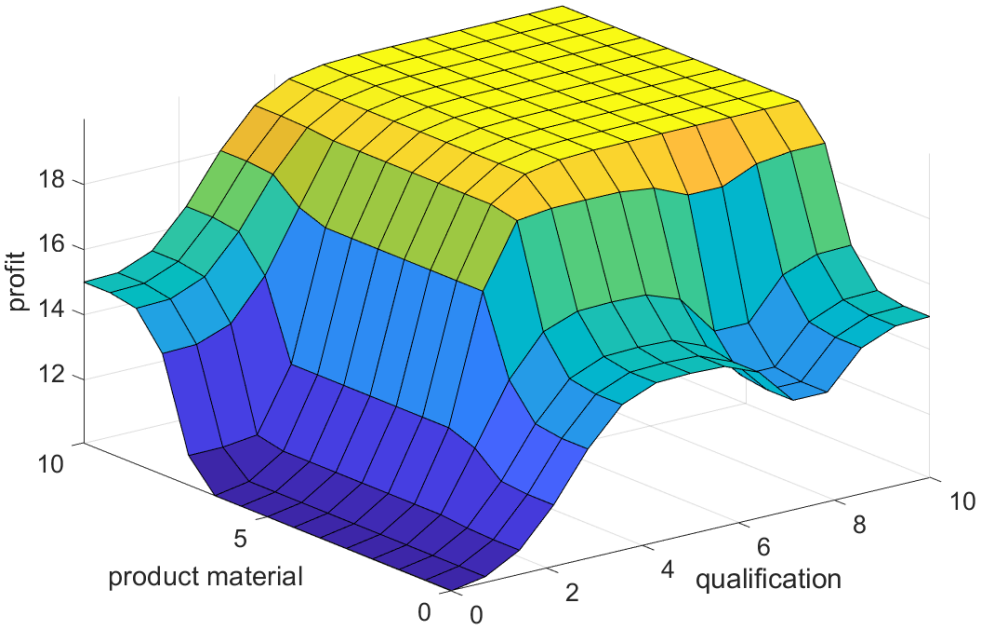


Рисунок 9.39 - Поверхня керування за результатами програмної

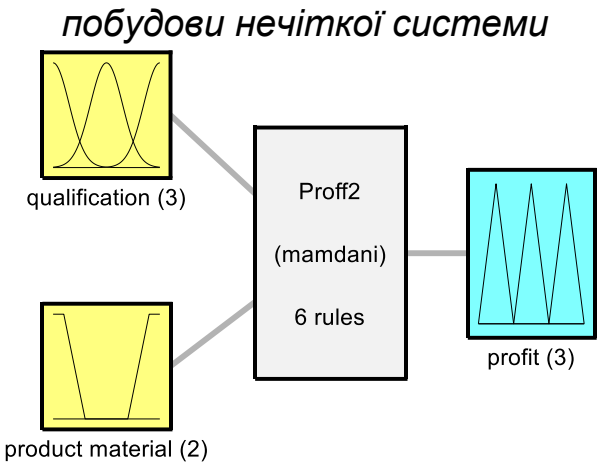


Рисунок 9.40 - Графічний опис властивості описаної нечіткої системи

Примітка. Реалізація даної нечіткої системи з використовувати *FIS Editor* з інтерфейсами, що входять до нього, наведено у файлі *Proff2.fis*.

Зазначимо, що керуюча крива (див. рис. 9.32 або рис. 9.37) може розглядатися як деяка функція, аргументами якої є входи нечіткої логічної системи.

У розглянутих прикладах ця функція була невідома, вона виходила відповідно до параметрів FIS. Можлива й інша постановка завдання – коли відома бажана форма кривої управління, і потрібно підібрати параметри FIS, при яких виходить апроксимація цієї кривої.

Приклад. Допустимо, потрібно підібрати такі параметри FIS, при яких виходить апроксимація заданої кривої (рис. 9.41).

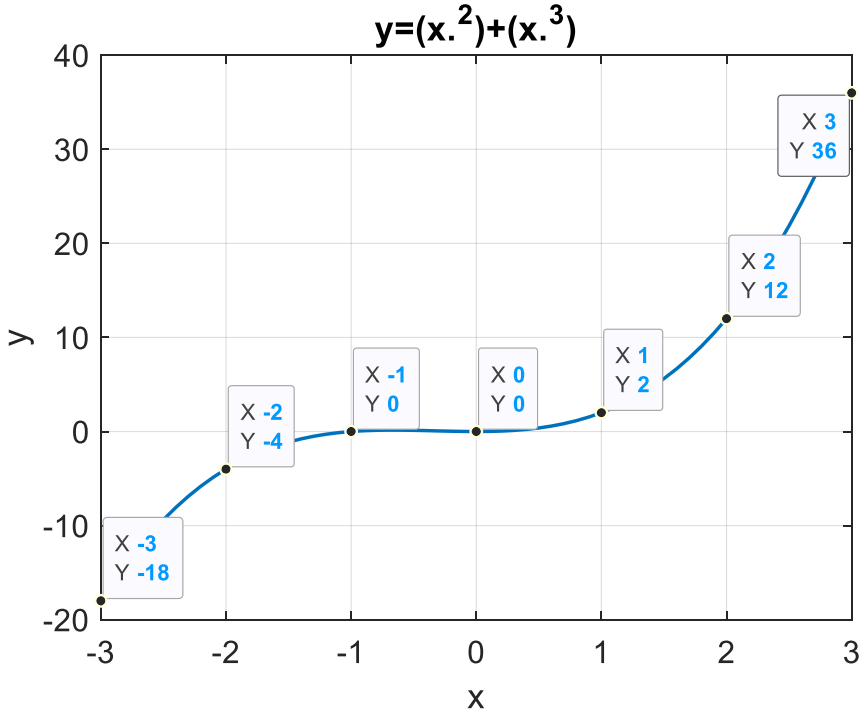


Рисунок 9.41 – Графік функції $y = x^2 + x^3$

Для виконання апроксимації потрібно вибрати набір точок (табл. 9.2).

Таблиця 9.2 – Набір точок для апроксимації функції

№ з/п	1	2	3	4	5	6	7
x	-3	-2	-1	0	1	2	3
y	-17	-4	0	0	3	12	36

Програмна реалізація для апроксимації функції з використанням нечіткої системи:

```
% Побудова графіку залежності, що завдана
xx = -3:0.1:3;
yy=(xx.^2)+(xx.^3);
figure (1)
plot(xx,yy); grid;
title('y=(x.^2)+(x.^3)');
xlabel('x');
ylabel('y');

fis = mamfis('Name','Graffuzzy');
% Додається та налаштується вхідна змінна.
fis.Inputs = fisvar;
fis.Inputs.Name = "x";
fis.Inputs.Range = [-3 3];
% Задаються функції приналежності вхідної змінної.
% Для кожного MF створюється об'єкт fismf і встановлюються
властивості
% за допомогою покрокового опису.
% Виберемо центри термів X, що описуються гаусовими функціями,
відповідно до табл. 9.2.

fis.Inputs.MembershipFunctions = fismf;
fis.Inputs.MembershipFunctions.Name = "x1";
fis.Inputs.MembershipFunctions.Type = "gaussmf";
fis.Inputs.MembershipFunctions.Parameters = [0.4247 -3];
fis.Inputs.MembershipFunctions(2) = fismf;
fis.Inputs.MembershipFunctions(2).Name = "x2";
fis.Inputs.MembershipFunctions(2).Type = "gaussmf";
fis.Inputs.MembershipFunctions(2).Parameters = [0.4247 -2];
fis.Inputs.MembershipFunctions(3) = fismf;
fis.Inputs.MembershipFunctions(3).Name = "x3";
fis.Inputs.MembershipFunctions(3).Type = "gaussmf";
fis.Inputs.MembershipFunctions(3).Parameters = [0.4247 -1];
```

```
fis.Inputs.MembershipFunctions(4) = fismf;
fis.Inputs.MembershipFunctions(4).Name = "x4";
fis.Inputs.MembershipFunctions(4).Type = "gaussmf";
fis.Inputs.MembershipFunctions(4).Parameters = [0.4247 0];
fis.Inputs.MembershipFunctions(5) = fismf;
fis.Inputs.MembershipFunctions(5).Name = "x5";
fis.Inputs.MembershipFunctions(5).Type = "gaussmf";
fis.Inputs.MembershipFunctions(5).Parameters = [0.4247 1];
fis.Inputs.MembershipFunctions(6) = fismf;
fis.Inputs.MembershipFunctions(6).Name = "x6";
fis.Inputs.MembershipFunctions(6).Type = "gaussmf";
fis.Inputs.MembershipFunctions(6).Parameters = [0.4247 2];
fis.Inputs.MembershipFunctions(7) = fismf;
fis.Inputs.MembershipFunctions(7).Name = "x7";
fis.Inputs.MembershipFunctions(7).Type = "gaussmf";
fis.Inputs.MembershipFunctions(7).Parameters = [0.4247 3];
% Додається та налаштовується вихідна змінна
% та її функція приналежності.
fis.Outputs = fisvar([-17 36],'Name',"y");
% Вказується вихідні функції приналежності
% за допомогою вектора об’єктів fismf.
% Виберемо центри термів У, що описуються гаусовими функціями,
відповідно до табл. 9.2
mf1 = fismf("gaussmf",[3.751 -17],'Name',"y1");
mf2 = fismf("gaussmf",[3.751 -4],'Name',"y2");
mf3 = fismf("gaussmf",[3.751 0],'Name',"y3");
mf4 = fismf("gaussmf",[3.751 0],'Name',"y4");
mf5 = fismf("gaussmf",[3.751 3],'Name',"y5");
mf6 = fismf("gaussmf",[3.751 12],'Name',"y6");
mf7 = fismf("gaussmf",[3.751 36],'Name',"y7");

fis.Outputs.MembershipFunctions = [mf1 mf2 mf3 mf4 mf5 mf6 mf7];

% Створюються правила для нечіткої системи.
rule1 = fisrule([1 1 1 1],1);
rule2 = fisrule([2 2 1 1],1);
rule3 = fisrule([3 3 1 1],1);
rule4 = fisrule([4 4 1 1],1);
rule5 = fisrule([5 5 1 1],1);
rule6 = fisrule([6 6 1 1],1);
rule7 = fisrule([7 7 1 1],1);

rules = [rule1 rule2 rule3 rule4 rule5 rule6 rule7];
rules = update(rules,fis);
fis.Rules = rules;
```

```
% Перевірка введених правил
showrule(fis,[1 2 3 4 5 6 7],'symbolic')

% Графіки опису термів
figure(2)
plotmf(fis,'input',1); grid
figure(3)
plotmf(fis,'output',1);grid

% Перевірка правил
showrule(fis,[1 2 3 4 5 6 7],'symbolic')

% Побудова графіку апроксимації ф-ції
figure(4);
gensurf(fis); grid

% Графічний опис властивості описаної нечіткої системи
figure(5)
plotfis(fis);
```

Графічне уявлення опису термів вхідної змінної наведено на рисунку 9.42, а вихідної змінної на рису. 9.43. Терми описано гаусовським функціями приналежності, центри термів співпадають з значеннями наведеними у таблиці 9.2.

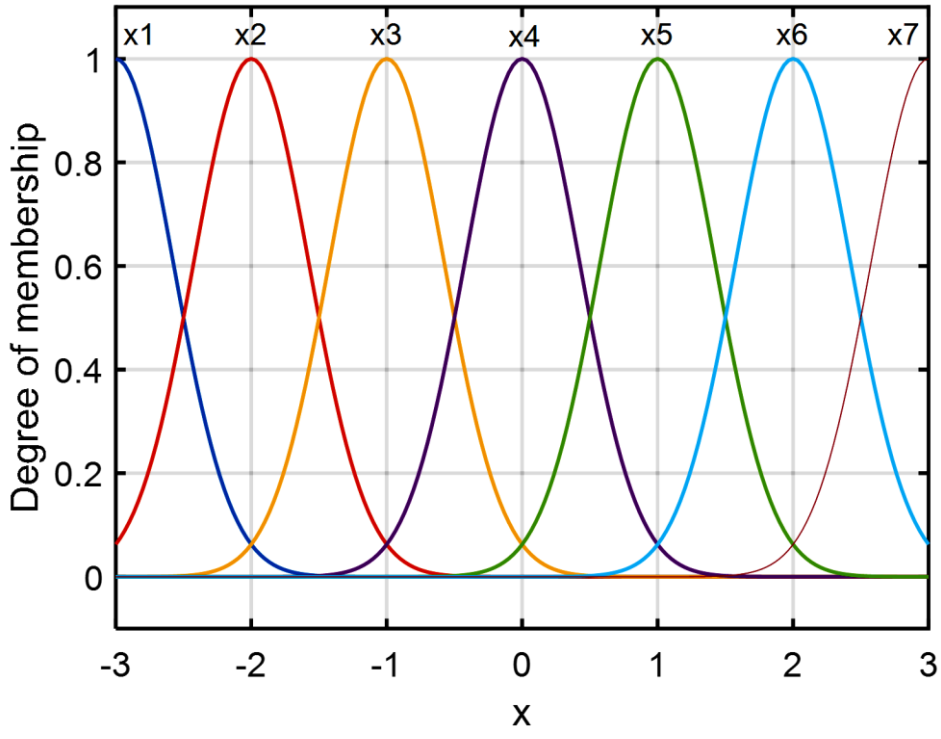


Рисунок 9.42 - Графічне уявлення опису термів вхідної змінної

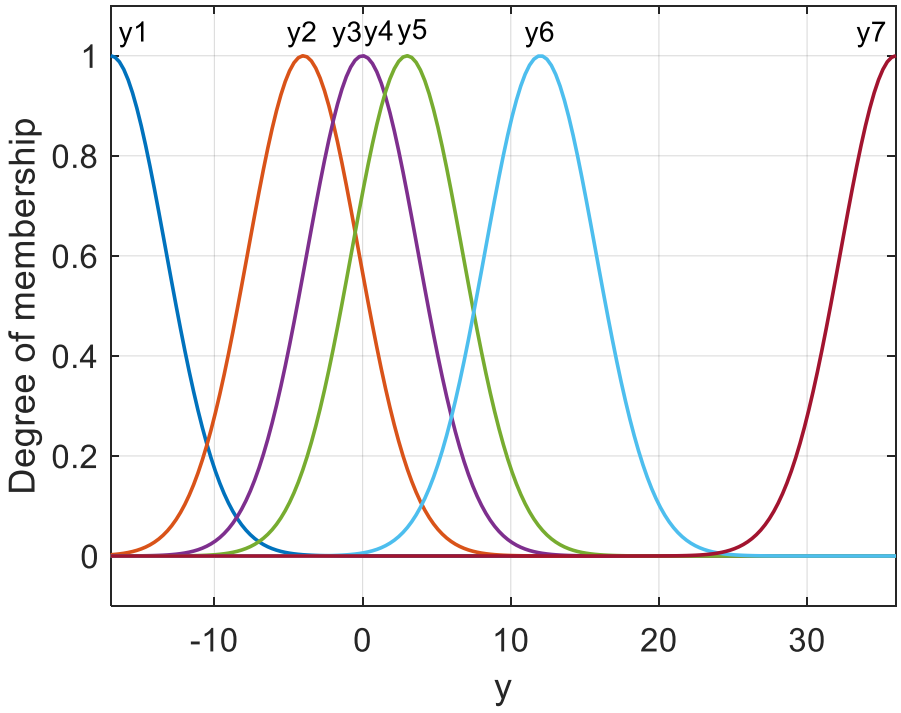


Рисунок 9.43 - Графічне уявлення опису термів вихідної змінної

На рис. 9.44 показаний результат апроксимації вихідної функції за допомогою системи нечітких правил.

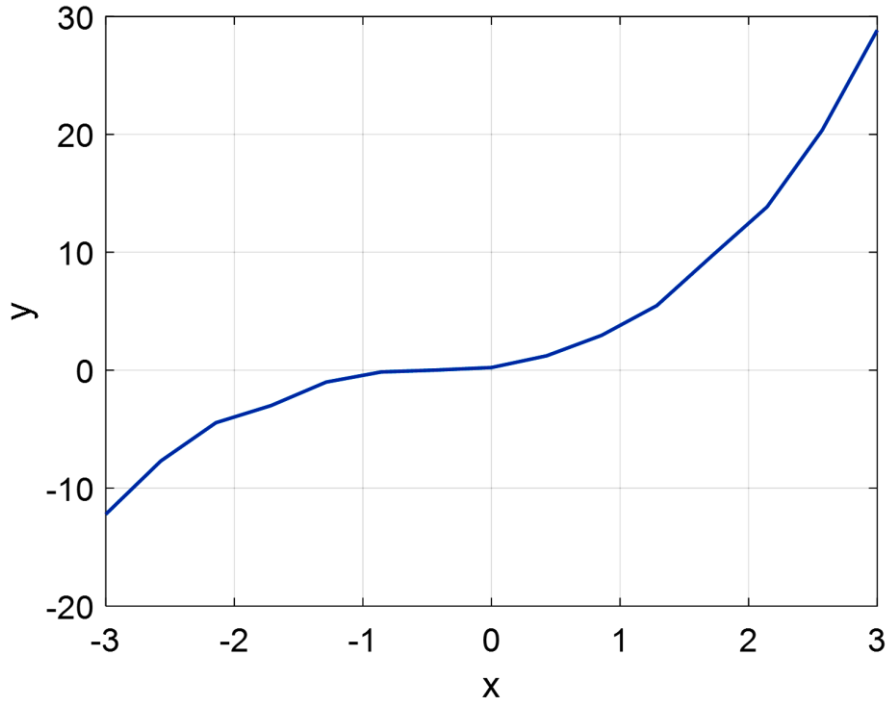


Рисунок 9. 44 – Результат апроксимації вихідної функції за допомогою системи нечітких правил

На рисунку 9.45 зображено графічний опис властивості описаної нечіткої системи.

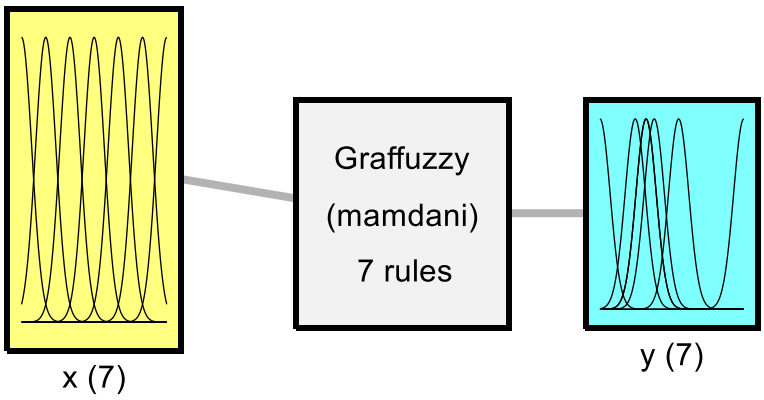


Рисунок 9.45 – Графічний опис властивості описаної нечіткої системи

Вирішимо дане завдання з використанням редактора системи нечіткого висновку.

Викликається редактор системи нечіткого виведення (Fuzzy Inference System – FIS) у MatLab командою

>> fuzzy

Дале проводиться налаштовування головне вікно редактора нечіткої логічної системи (див. рис. 9.46.).

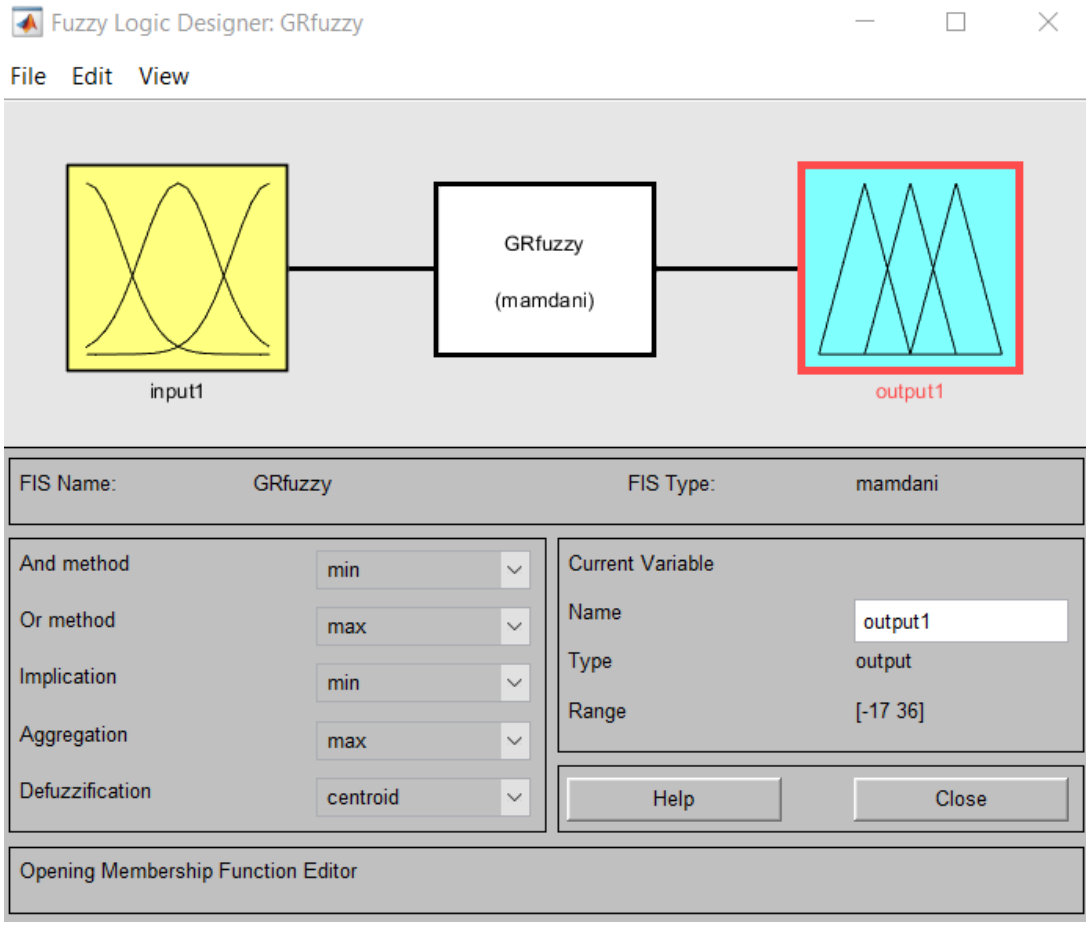


Рисунок 9.46 - Інтерфейс FIS editor

У *FIS editor* задається опис систему нечіткого логічного висновку *Mamdani*. Для створюваної системи обирається вид логічного зв'язоку (*And method – min*) та (*Or method – max*), вид імплікації (*Implication – min*), спосіб агрегування висновків правил (*Aggregation – max*) та метод дефазифікації (*Defuzzification – centroid*).

У меню *Edit* послідовно додаємо 7 вхідних з розміром базової шкали (*Range*= [-3 3]) та 7 вихідних змінних з розміром базової шкали (*Range*= [-17 36]).

Для опису вхідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної змінної *x* гаусовську функцію приналежності. З параметрами:

Name = "x1"; Type = "gaussmf"; Param = [0.4247 -3];
Name = "x2"; Type = "gaussmf"; Param = [0.4247 -2];
Name = "x3"; Type = "gaussmf"; Param = [0.4247 -1];
Name = "x4"; Type = "gaussmf"; Param = [0.4247 0];
Name = "x5"; Type = "gaussmf"; Param = [0.4247 1];
Name = "x6"; Type = "gaussmf"; Param = [0.4247 2];
Name = "x7"; Type = "gaussmf"; Param = [0.4247 3];

Для опису вихідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної *y* гаусовську функцію приналежності. З параметрами:

Name = "y1"; Type = "gaussmf"; Param = [3.751 -17];
Name = "y2"; Type = "gaussmf"; Param = [3.751 -4];
Name = "y3"; Type = "gaussmf"; Param = [3.751 0];
Name = "y4"; Type = "gaussmf"; Param = [3.751 0];
Name = "y5"; Type = "gaussmf"; Param = [3.751 3];
Name = "y6"; Type = "gaussmf"; Param = [3.751 12];
Name = "y7"; Type = "gaussmf"; Param = [3.751 36];

Примітка. У якості центрів термів, що описуються гаусовими функціями, обрані значення відповідно до табл. 9.2.

Приклад налаштування головного вікна редактора *Membership Function Editor* показано на рис. 9.47.

Після опису вхідні та вихідні лінгвістичні змінні, здійснюється опис правил у вікні редактора Rule Editor. Налаштовування правил для даного випадку наведено на рисунку (рис. 9.48).

Посилки у правилах можуть бути пов'язані (connection) за допомогою операції *and*. Введене правило забезпечене ваговим коефіцієнтом (*Weight*=1).

Після опису правил можливо за допомогою:

- інтерфейсу *Rule Viewer* можливо переглянути роботу системи нечіткого логічного висновку за різних вхідних даних (див. рис. 9.49);

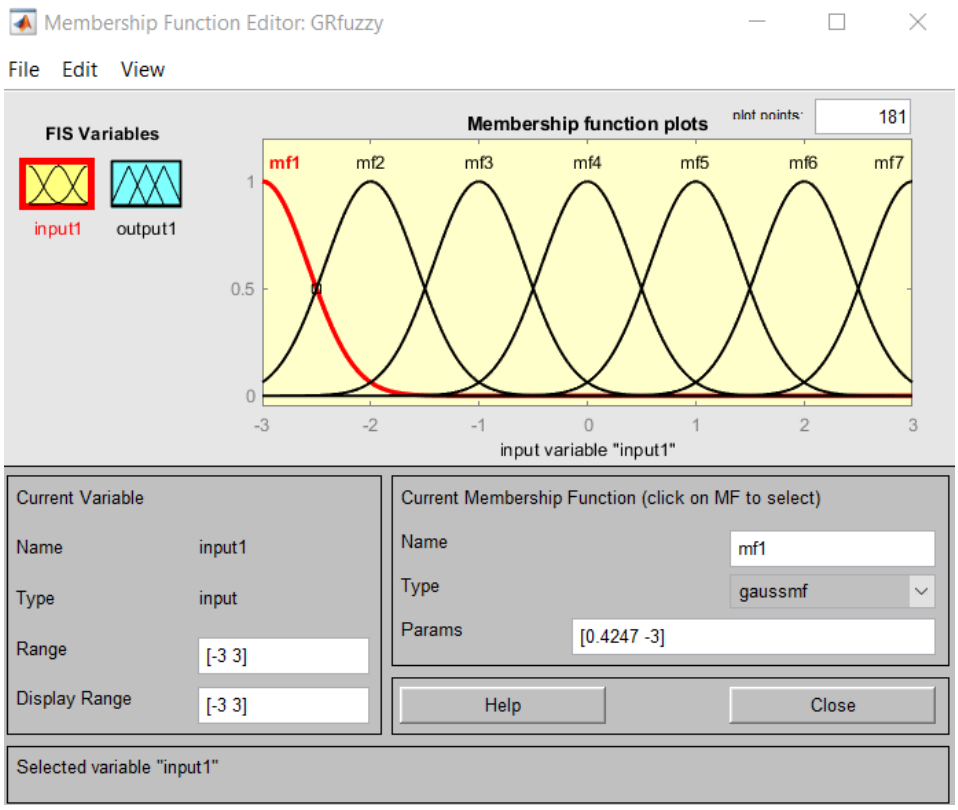


Рисунок 9.47 - Приклад налаштування головного вікна редактора *Membership Function Editor*

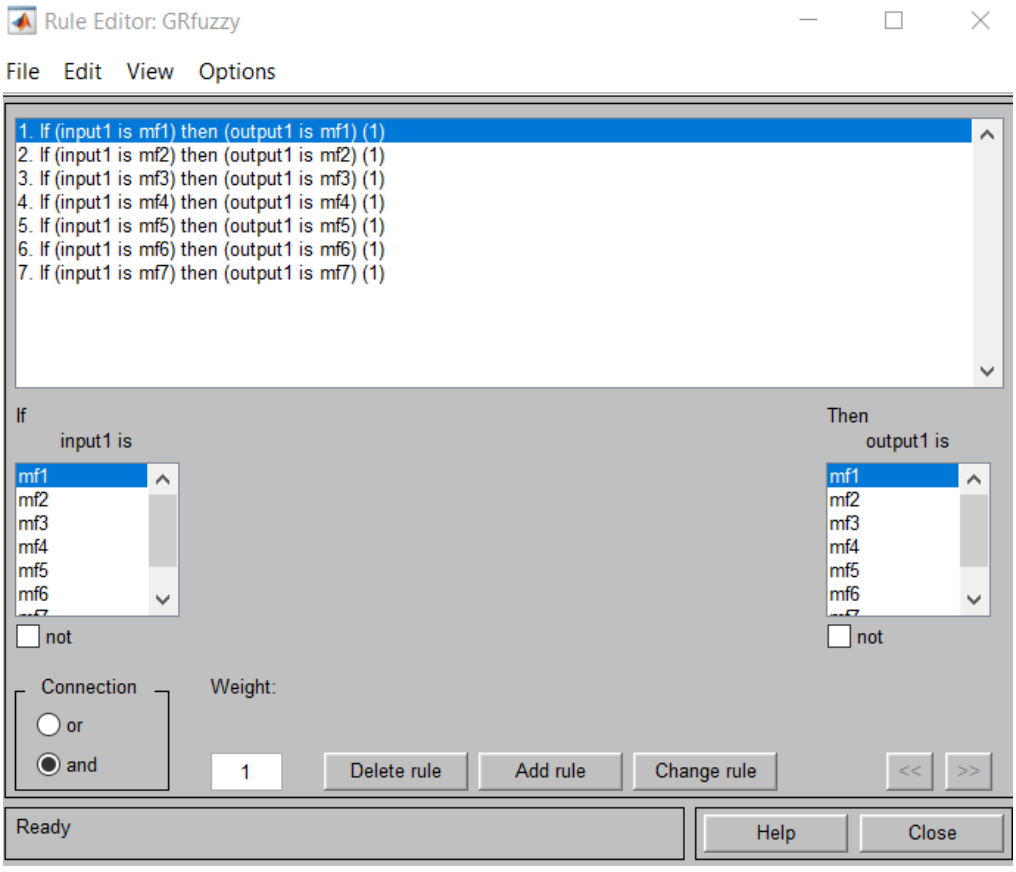


Рисунок 9.48 – Інтерфейс ruleedit

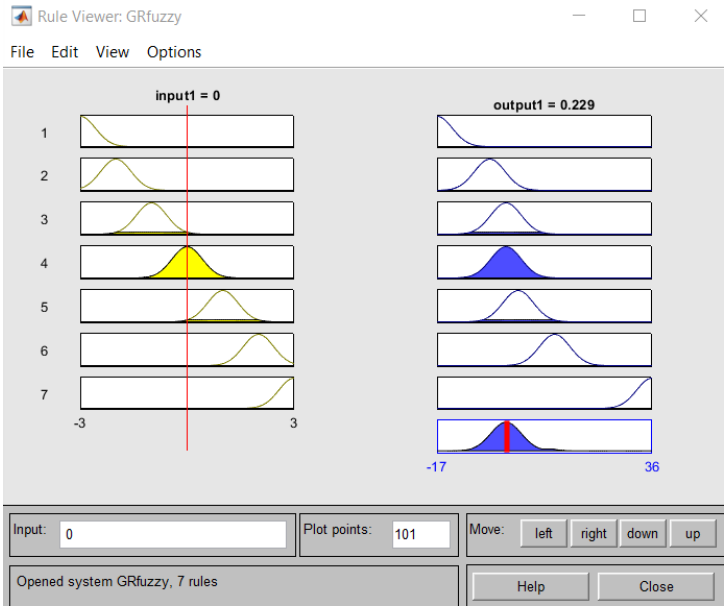


Рисунок 9.49 - Робота системи нечіткого логічного висновку

– інтерфейсу *Surface Viewer* можливо переглянути керуючу поверхню нечіткої логічної системи, яка виходить при подачі на вхід системи різних допустимих значень (рис. 9.50).

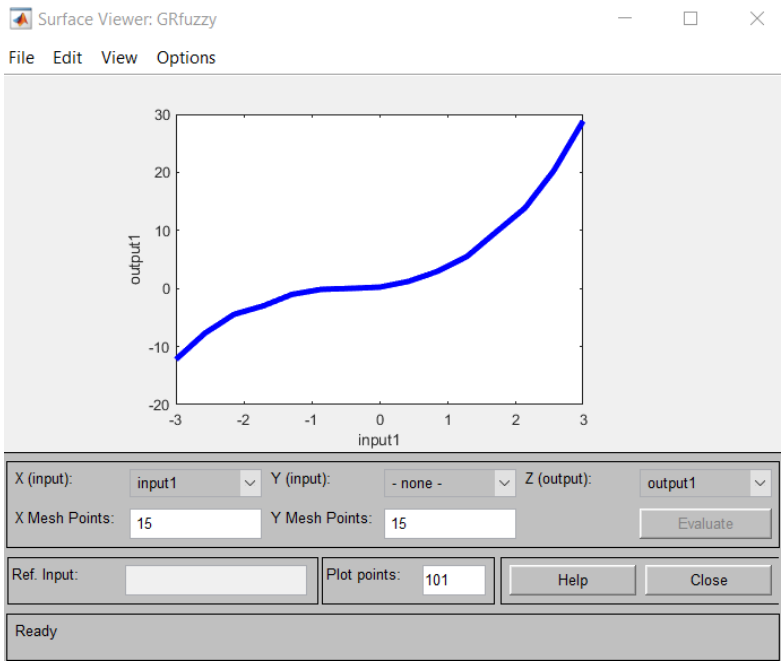


Рисунок 9.50 – Апроксимаційна крива побудована системою нечіткого логічного висновку

Отримана крива не цілком відповідає графіку вихідної функції $y=x^2 + x^3$, проте якість апроксимації можна покращити, якщо змінити форму термів, що описують входи та виходи нечіткої системи, або додати нові правила.

10 ІНТЕЛЕКТУАЛЬНІ РЕГУЛЯТОРИ З ВИКОРИСТАННЯМ НЕЧІТКИХ ЛОГІКИ

10.1. Управління зі зворотним зв'язком

Розглянемо традиційну систему управління з негативним зворотним зв'язком (рис.10.1).

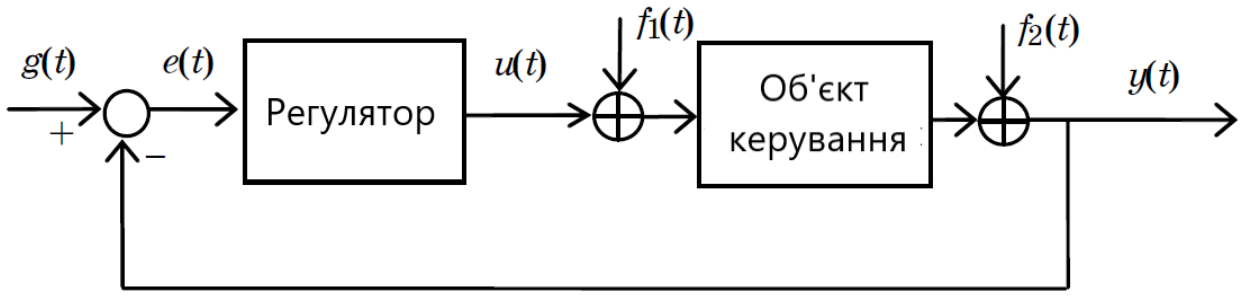


Рисунок 10.1 - Система управління із зворотним зв'язком

Вплив $g(t)$ визначає бажане значення виходу об'єкта управління $y(t)$. Помилка керування $e(t) = g(t) – y(t)$ використовується регулятором для формування сигналу керування $u(t)$. Об'єкт являє собою динамічну систему, яка може бути нестійкою або мати погані динамічні властивості. Регулятор служить поліпшення динамічного поведінки об'єкта. Шуми на вході та виході об'єкта $f_1(t)$ та $f_2(t)$ можуть бути одним із джерел неточності (нечіткості) завдання управління (при синтезі регулятора ними часто нехтують).

Класична теорія управління орієнтована переважно на синтез лінійних регуляторів для таких об'єктів. Властивість лінійності дозволяє отримати зручні алгоритми оцінки таких важливих системних властивостей, як стійкість, керованість, спостережуваність тощо.

Проте всі реальні об'єкти є нелінійними, і лінеаризація їхнього математичного опису вимагає розгляду певної робочої точки, щодо якої розглядаються малі відхилення.

Нелінійність математичної моделі виявляється у присутності нелінійних блоків: насичення, сухе тертя, гістерезис тощо. Зокрема, введення у структуру рис. 10.1 обмеження значення сигналу управління робить її нелінійною. Нелінійність також виникає, якщо в математичному описі є нелінійні функції (квадрат, квадратний корінь, тригонометричні функції тощо).

Нечіткі логічні регулятори (НЛР), нелінійні за своєю сутю, можуть керувати лінійними об'єктами краще, ніж класичні регулятори, і навіть керувати нелінійними об'єктами, котрим лінійні регулятори непридатні.

Для лінійних скалярних об'єктів (з одним входом та одним виходом) при описі перехідних процесів традиційно розглядається

реакція $y(t)$ на стрибкоподібний вхідний вплив $g(t)$, і використовуються такі параметри, як (рис. 10.2):

− час наростання $t_н$, тобто час, за який змінна $y(t)$ зростає з 0.1 до 0.9 значення $y_{вст}(t)$, що встановилося;

− перерегулювання $d = (y_{max}(t) – y_{вст}(t))/100$ %;

− помилка, що встановилася $e = g(t) − y_{вст}(t)$;

− час перехідного процесу $t_{пп}$ (час від початку перехідного процесу до моменту, коли $y(t)$ не залишає інтервал 1 ± 0.05;

− $t_з$ – час загасання перехідного процесу (час між моментом першого досягнення сигналом $y(t)$ одиничного рівня та моментом, починаючи з якого значення y(t) залишаються всередині інтервалу [1±0.05].

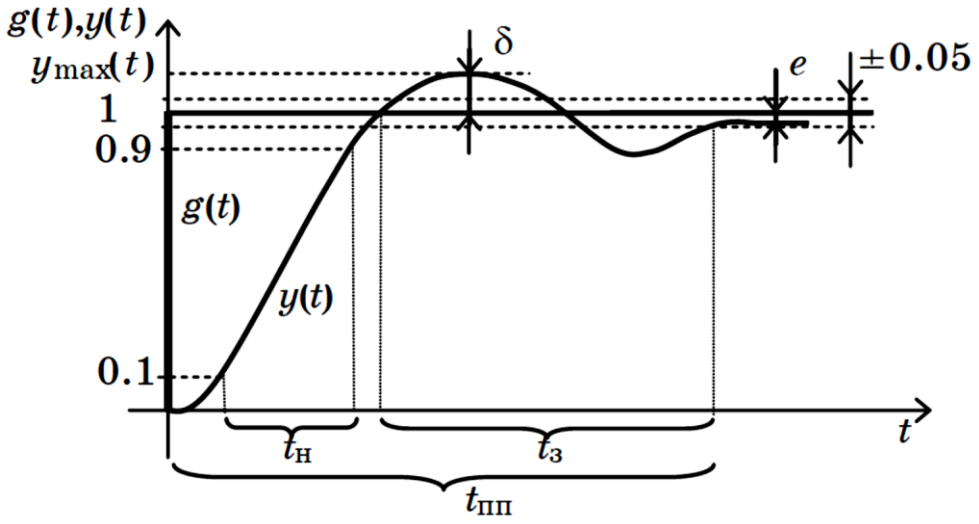


Рисунок 10.2 – параметри перехідного процесу

Реакція об'єкта на ступінчасту вхідну дію називається *перехідною функцією або кривою розгону*.

За характером значення вихідної величини об'єкта (при подачі на вхід одиничного стрибка) всі об'єкти діляться на дві групи: з самовирівнюванням і без самовирівнювання.

Самовирівнюванням називається властивість регульованого об'єкта після зміни вхідного сигналу самостійно повернутися до нового стану (див. рис. 10.2). Самовирівнювання полегшує роботу регулятора. Стійке функціонування об'єкта управління без самовирівнювання неможливе без регулятора.

10.2. Моделі об'єктів керування

Як показує багаторічний досвід конструювання промислових систем управління, більшість реальних динамічних об'єктів або описується передаточними функціями першого або другого порядку

(можливо – із запізненням), або їх динамічні характеристики можуть бути апроксимовані цими функціями. Таким чином, типовий опис об'єкта управління має вигляд:

$$W(p) = \frac{ke^{-\tau p}}{(T_1 p + 1)(T_2 p + 1)};\ T_1 > 2T_2. \qquad (10.1)$$

За наявності ланки запізнення ($e^{(-\tau p)}$) порядок передаточної функції зростає на k разів, де k – порядок розкладання експоненційної функції ряд Паде.

Для об'єктів керування з явно вираженою постійною часу передатна функція спрощується:

$$W(p) = \frac{ke^{-\tau p}}{Tp + 1}, \qquad (10.2)$$

де k, T, τ– коефіцієнт посилення, постійна часу та запізнення, які мають бути визначені в околиці номінального режиму роботи об'єкта.

Для об'єкта керування без самовирівнювання передаточна функція має вигляд

$$W(p) = \frac{ke^{-\tau p}}{p}. \qquad (10.3)$$

Виникає завдання визначення параметрів динамічної моделі об'єкта керування (ідентифікації). Це можна зробити експериментально на реальному об'єкті управління. Можуть розглядатися часові або частотні характеристики об'єкта управління.

Розглянемо експериментальні методи визначення часових динамічних характеристик об'єкта управління. Вони поділяються на два класи: активні та пасивні.

Активні методи передбачають подачу на вхід об'єкта пробних сигналів, таких як ступінчастий або прямокутний імпульс.

Залежно від виду пробного сигналу вибирають відповідні способи обробки вихідного сигналу об'єкта управління.

Так, наприклад, при подачі ступінчастого керуючого сигналу знімають криву розгону об'єкта, а при подачі прямокутного імпульсного сигналу знімають криву відгуку. Крива відгуку знімається для об'єктів, які не допускають подачі на вхід об'єкта ступеневих сигналів.

Пасивні методи застосовуються тоді, коли неможливе порушення нормального перебігу технологічного процесу.

Розглянемо визначення динамічних характеристик об'єкта виду 10.2) щодо його кривої розгону при подачі ступінчастого пробного сигналу.

У початковий момент необхідно, щоб система управління перебувала в спокої, тобто регульована величина *y*(*t*) і вплив *g*(*t*), що задає вплив, не змінювалися, а зовнішні обурення були відсутні. Потім на вхід виконавчого механізму подається ступінчаста дія, і стан об'єкта починає змінюватися.

При знятті кривої розгону необхідно виконати низку умов:

1) якщо проектується система стабілізації, то крива розгону має зніматися на околиці робочої точки процесу;

2) криві розгону необхідно знімати як за позитивних, і негативних стрибках управляючого сигналу. По виду кривих можна будувати висновки про ступеня асиметрії об'єкта;

3) за наявності зашумленого виходу бажано знімати кілька кривих розгону з їх подальшим накладенням один на одного та отриманням усередненої кривої.

Знявши криву розгону і оцінивши характер об'єкта управління (з самовирівнюванням або без) можна визначити параметри відповідної передаточної функції. Зокрема, з цією метою можна використовувати метод дотику до точки перегину кривої розгону.

Розглянемо приклад використання методу для об'єкта з самовирівнюванням (рис. 10.3).

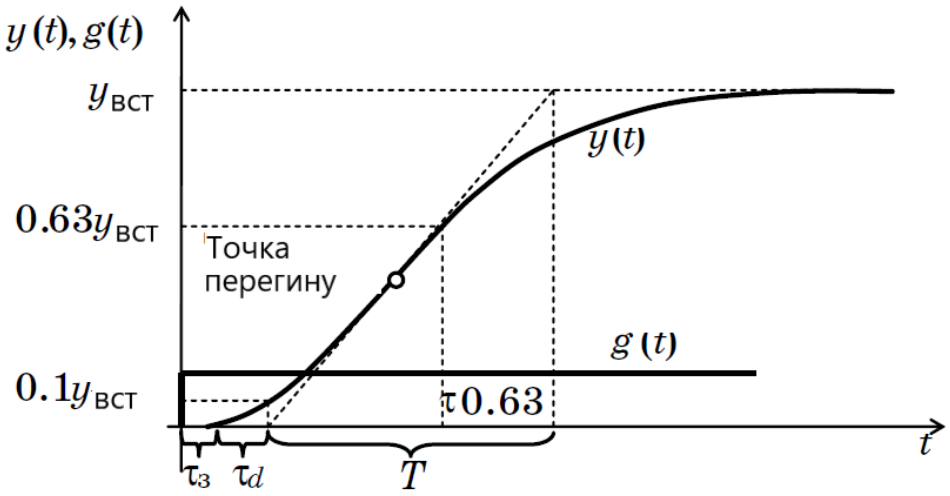


Рисунок 10.3 – Графік кривої розгону

Точка перегину відповідає переходу кривої від режиму прискорення до режиму уповільнення темпу наростання вихідного сигналу. Постійна часу *T* визначається відповідно до графіка рис. 10.3.

Досить точною вважається також оцінка виду

$$T = \tau_{0.63} - \tau_з$$

Динамічне запізнення τ також визначається за графіком, і складається з двох компонентів:

$$\tau = \tau_з + \tau_d,$$

де $\tau_з$ - чисте (ємнісне) запізнення, τ_d – транспортне запізнення.

Для об'єкта з самовирівнюванням динамічний коефіцієнт посилення *K* показує, у скільки разів ця ланка посилює вхідний сигнал (див. рис. 10.3):

$$K = \frac{y_{\text{вст}}}{g}.$$

Для об'єкта без самовирівнювання виду (10.3) динамічний коефіцієнт посилення *K* визначається як відношення швидкості зміни вихідної величини у до величини стрибка вхідного сигналу. При одиничному стрибку

$$K = \frac{\Delta y}{\Delta t}.$$

Для визначення динамічних показників об'єкта виду (10.1) можна використовувати метод Орманса.

Цей метод дозволяє за нормованою кривою розгону визначити дві домінуючі постійні об'єкти управління.

Перед початком обробки перехідної характеристики потрібно її пронормувати (діапазон зміни нормованої кривої 0-1) і виділити з початкової ділянки величину чистого тимчасового запізнення.

Нормована крива перехідна характеристика визначається формулою

$$h(t) = \frac{y(t) - y_{min}}{y_{max} - y_{min}}.$$

Потім визначається час, що відповідає значенню y_H = 0.7 і позначається t_7. Отриманий інтервал поділяється на три частини. Піднімається перпендикуляр до кривої розгону та визначається величина y_{H4}. Аналітично доведено зв'язок між точками кривої розгону та параметрами моделі, а саме

$$t_7 = 1.2(T_1 + T_2);\ \ t_4 = \frac{t_7}{3}.$$

Постійні часу об'єкта управління T_1 та T_2 визначаються за допомогою допоміжної величини Z^2, для якої використовується номограма (рис. 10.4).

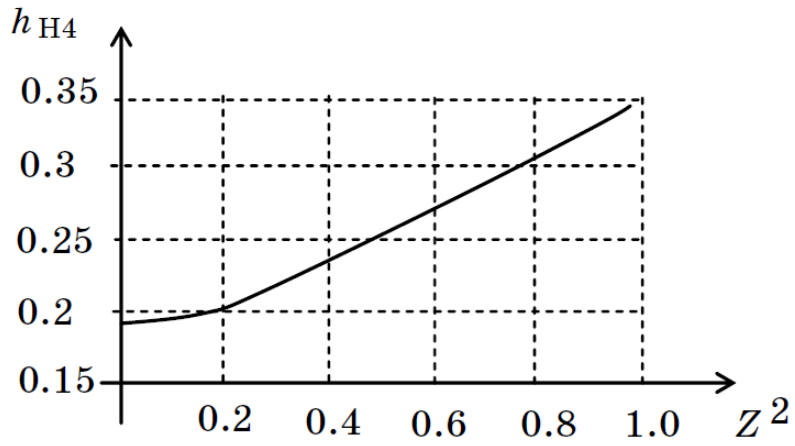


Рисунок 10.4 – Номограма для визначення величини

Постійні часу об'єкта управління T_1 і T_2 визначаються за такими формулами:

$$T_1 = \frac{t_7}{2.4}(1+z);\ T_2 = \frac{t_7}{2.4}(1-z)$$

Якщо $h_{H4} < 0.19$, то визначення динаміки об'єкта використовують метод площини. Якщо $T_1 >> T_2$, можна перейти до моделі першого порядку.

Найбільш універсальним алгоритмом визначення параметрів моделі є метод найменших квадратів (МНК). З його допомогою можна побудувати модель не лише другого, а й вищих порядків.

Для використання МНК необхідні масиви значень вхідних та вихідних сигналів об'єкта, знятих через деякий інтервал часу T_k – період квантування. У вхідному сигналі об'єкта повинна бути як постійна, так і пробна складові. Постійна складова визначає положення робочої точки процесу, в околиці якої проводиться визначення параметрів динамічної моделі об'єкта. Таким чином, робота відбувається з цифровою (дискретною) моделлю об'єкта.

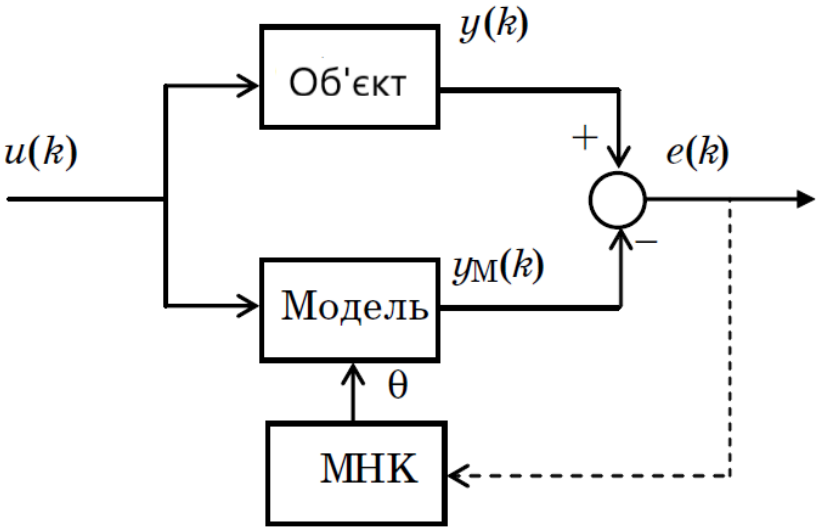
Нехай цифрова модель першого порядку задана у вигляді

$$y_M(k) = ay(k-1) + bu(k-1).$$

Схема експерименту показано на рис. 10.5.

Нехай накопичено (N + 1) точок вимірювання вхідного та вихідного сигналів об'єкта. У методі найменших квадратів узагальнена помилка ідентифікації має бути мінімальною:

$$E = \sum_{k=1}^{N+1} e(k)^2 \to \min_{\Theta}$$



u(*k*), *y*(*k*), y_M(k); *θ* – вхідний та вихідний сигнали об'єкта, вихідний сигнал моделі та вектор оцінки параметрів; *e*(*k*) – поточна помилка ідентифікації

Рисунок 10.5 – Структурна схема експерименту

Введемо позначення

$$\mathrm{A} = \mathrm{y(k)} - \mathrm{ay(k-1)}; \mathrm{B} = \mathrm{bu(k-1)},$$

тоді

$$E = \sum_{k=1}^{N+1} [A - B]^2.$$

Після розкриття дужок та приведення подібних отримаємо

$$E = S_1 - 2aS_2 + a^2S_3 - 2bS_4 + 2abS_5 + b^2S_6,$$

де

$$S_1 = \sum y^2(k);$$
$$S_2 = \sum y(k)y(k-1);$$
$$S_3 = \sum y^2(k-1);$$
$$S_4 = \sum y(k)u(k-1);$$
$$S_5 = \sum y(k)u(k-1);$$

$$S_6 = \sum u^2(k-1).$$

Умова мінімуму функції *E* передбачає рівність нуля приватних похідних за параметрами *a* та *b*:

$$\frac{\partial E}{\partial a} = 0;\ \frac{\partial E}{\partial b} = 0;$$

$$\begin{cases} \frac{\partial E}{\partial a} = -S_2 + aS_3 + bS_5 = 0 \\ \frac{\partial E}{\partial b} = -S_4 + aS_5 + bS_6 = 0 \end{cases} \Rightarrow \begin{cases} S_2 = aS_3 + bS_5 \\ S_4 = aS_5 + bS_6 \end{cases}.$$

Тоді можна записати:

$$A\theta = B$$

Отже, для обчислення оцінок векторе параметрів об'єкта управління методом найменших квадратів отримано формулу

$$\theta = A^{-1}B.$$

Зворотна матриця A^{-1} завжди існує, оскільки вихідна матриця *A* симетрична та позитивно визначена.

Знаючи параметри дискретної моделі, можна визначити параметри передаточної функції об'єкта.

$$W(p) = \frac{K}{Tp+1} = \frac{y(p)}{u(p)}.$$

Зв'язок між параметрами дискретної моделі та передаточної функції визначається формулами:

$$a = e^{-\frac{T_k}{T}};\ b = K(1-a).$$

Звідки випливає, що

$$T = -\frac{T_k}{\ln(a)};\ K = \frac{b}{1-a}.$$

У пакеті MatLab подібні перетворення виконуються однаково для моделей будь-якого порядку.

При використанні МНК оцінки, що одержуються, обчислюються з

деякими помилками, які називаються зміщенням оцінок. Для отримання хороших результатів необхідно виконати ряд умов, головні з яких полягають у тому, щоб сигнал, що тестує, був досить різноманітним, а обсяг вибірки був досить великим.

Розглянемо приклади ідентифікації.

Приклад. Дослідження поведінки об'єкта за умов обурень. Необхідно ідентифікувати постійні часу об'єкта за перехідною функцією і помилку в процесі, що встановився.

Вихідні дані:

1. Структурну схему об'єкта представлено на рис. 10.6.

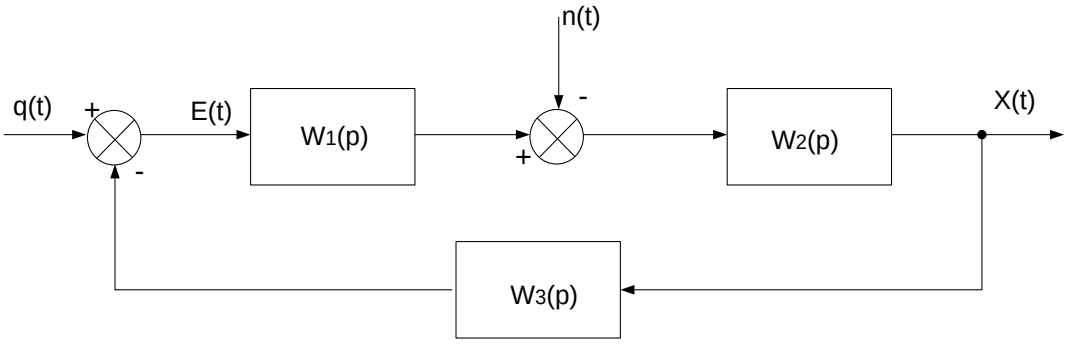


Рисунок 10.6 – Структурна схема об'єкта

2. Передаточні функції структурних блоків:

$$W_1(p)=\frac{K_1}{T_1p+1};\quad W_2(p)=\frac{K_2}{T_2p+1};\quad W_3=K_3.$$

3. Значення коефіцієнтів та постійних часу: K_1=5; K_2=1; K_3=0,2; T_1=0,05*с*; T_2=0,25*с*.

4. Моделювання перехідного процесу зробити із застосуванням пакета MATLAB (Simulink).

Обчислення параметрів перехідної функції виконати із застосуванням пакета MathCAD.

Рішення:

1. Визначимо передатну функцію замкнутої системи, для подальшого порівняння результатів ідентифікації:

$$W_3(p)=\frac{W_1(p)W_2(p)}{1+W_1(p)W_2(p)W_3(p)}=\frac{\frac{5}{0.05p+1}\cdot\frac{1}{0.25p+1}}{1+\frac{5}{0.05p+1}\cdot\frac{1}{0.25p+1}\cdot 0.2}=$$

$$= \frac{\frac{5}{(0.05p+1)(0.25p+1)}}{1+\frac{1}{(0.05p+1)(0.25p+1)}} = \frac{5}{(0.05p+1)(0.25p+1)+1} \cdot$$

$$\cdot \frac{(0.05p+1)(0.25p+1)}{1+(0.05p+1)(0.25p+1)} = \frac{5}{1+0.05\cdot 0.25p^2+0.25p+0.05p+1} =$$

$$= \frac{5}{0.0125p^2+0.3p+2} = \frac{2.5}{0.00625p^2+0.15p+1}$$

2. Здійснюємо математичне моделювання в середовищі MATLAB. Схема математичної моделі представлена рис. 10.7.

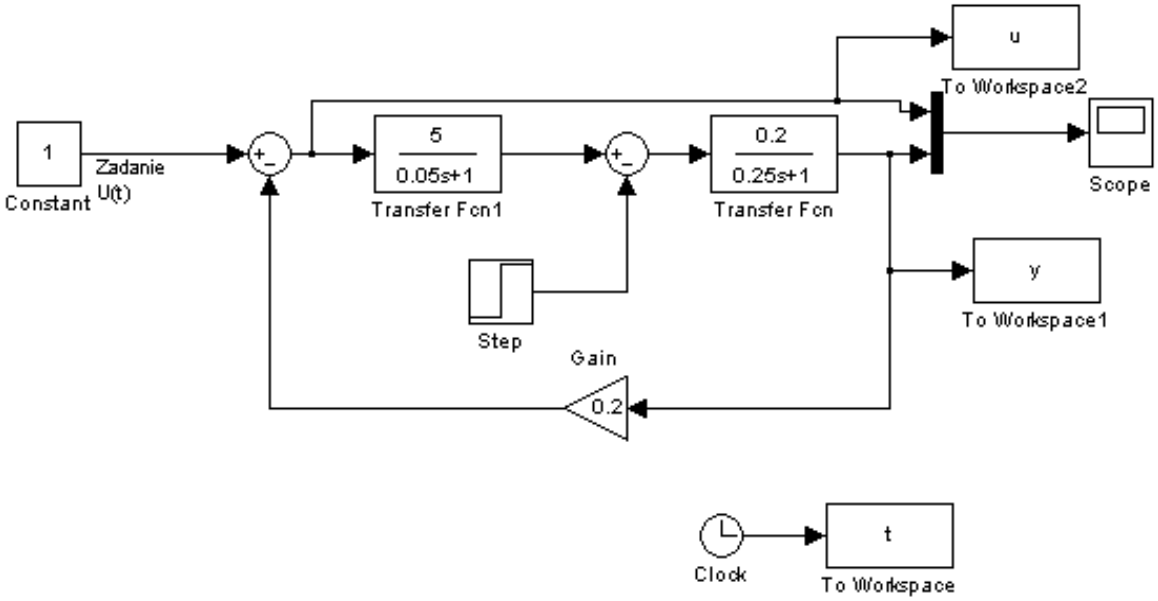


Рисунок 10.7 – Структурна схема математичної моделі об'єкта

Параметри моделювання:

- початок моделювання - 0,0

– кінець моделювання – 4,0

– спосіб моделювання – з фіксованим кроком (Fixed-Step)

– величина кроку – auto (встановлена автоматично).

Для блоку Step, що моделює зовнішнє обурення, встановлюємо такі параметри:

– час настання перепаду сигналу – 2,0 (після закінчення перехідного процесу);

- Початкове значення сигналу - 0,0

- Кінцеве значення сигналу - 4,0 (приймаємо обурення на рівні 20% керуючого впливу:

$1(t)\frac{20\%}{100\%}\cdot$, тобто 0,2·(1·5) = 1,0;

Для блоку *Scope* встановлюємо такі значення:
- координата Y - від 0 до 2,5 (коефіцієнт передачі системи).
Результати моделювання подано на рис. 10.8.

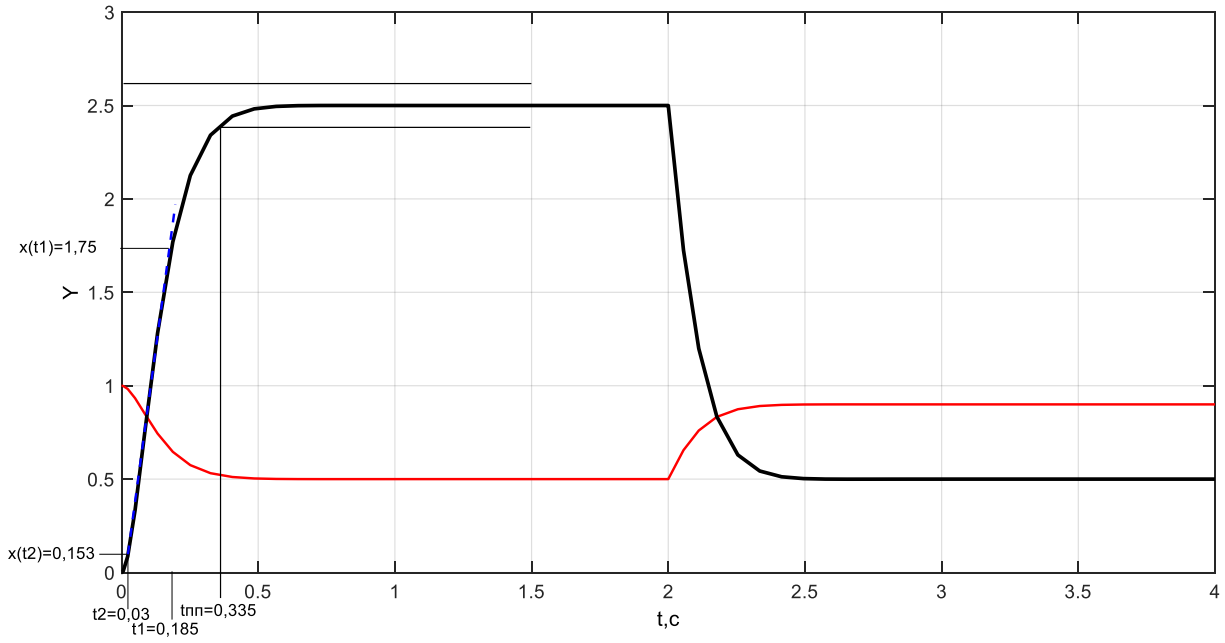


Рисунок 10.8 – Графік перехідного процесу

З графіка видно, що:
- Тривалість перехідного процесу складає $t_{пп}$ = 0,335 *с*;
– система має аперіодичний перехідний процес, стійка;
– встановлене значення помилки $E_{вст}$ = 0,525.
– при обуренні помилка збільшується до 0,9.

3. Постійні часу об'єкта T_1 та T_2 визначаємо із графіка перехідної функції.

Так як $T_2 > 4T_1$ і перехідний процес аперіодичний, то відносна величина вихідного сигналу $x(t)_{вітн}$ описується рівнянням:

$$x(t)_{відн} = \frac{x(t)}{x_{вст}} = 1 - 0.5\left[\frac{1+\eta}{\eta} e^{\frac{-2t^*}{1+\eta}} + \frac{1-\eta}{\eta} e^{\frac{-2t^*}{1-\eta}}\right],$$

де $t^* = \frac{t}{T_2}$ – відносний час; $\eta = \sqrt{1 - \frac{4T_1}{T_2}}$ – коефіцієнт, який характеризує відношення T_1 та T_2; $\zeta = \sqrt{\frac{4T_1}{T_2}}$ – коефіцієнт демпфірування системи; $x(t)_{отн} = 0...1$.

Визначаємо коефіцієнт, що характеризує співвідношення постійних часу.

$$\eta = \sqrt{1 - \frac{4T_1}{T_2}} = \sqrt{1 - \frac{4 \cdot 0{,}05}{0{,}25}} = 0{,}447.$$

Використовуючи відомий висновок, що величина η практично не впливає на β при $x(t)_{отн} = 0{,}7$, для якої $t^*_{0{,}7} \approx 1{,}2$.

Визначаємо значення $x(t_1)$ при $x(t)_{отн} = 0{,}7$:

$$x(t_1) = 0{,}7 x_{уст} = 0{,}7 \cdot 2{,}5 = 1{,}75.$$

З графіка перехідного процесу (див. рис. 10.8) визначаємо час встановлення значення $x(t_1) = 1{,}75$, яке дорівнює $t_{0{,}7} = 0{,}185$.

Так як $t^*_{0{,}7} = 1{,}2$, постійна асу T_2 для $t_{0{,}7} = 0{,}185$ становить:

$$T_2 = \frac{t_{0{,}7}}{t^*_{0{,}7}} = \frac{0{,}185}{1{,}2} = 0{,}154.$$

Задаючи інший відносний час $t^* = 0.2$ знаходим

$$t_2 = 0{,}2 \cdot T_2 = 0{,}2 \cdot 0{,}154 = 0{,}0308.$$

З графіка перехідного процесу визначається $x(t_2) = 0{,}153$

$$x_{отн}(t_2) = \frac{0{,}153}{2{,}5} = 0{,}0617.$$

4. Використовуючи середовище математичної прикладної програми *MathCAD*, за допомогою функції *Given* розв'язуємо рівняння $x(t)$ щодо η:

$$XX := 0.0617$$

Given $\qquad \eta := 0.477$

$$1 - \left(\frac{1 + \eta}{2 \cdot \eta} \cdot e^{\frac{-2 \cdot 0.4}{1+\eta}} \right) - \left(\frac{1 - \eta}{\eta \cdot 2} \cdot e^{\frac{-2 \cdot 0.4}{1-\eta}} \right) = XX$$

$$\eta 2 := \text{Find}(\eta) \qquad \eta 2 = 0.529$$

$$\eta := 0.526$$

$$\hat{T}_1 = \frac{1}{4} \cdot (1 - \eta^2) \cdot \hat{T}_2 = \frac{1}{4} \cdot (1 - 0{,}526^2) \cdot 0{,}154 = 0{,}028\ c.$$

Визначимо помилку ідентифікації:

$$\Delta T_1 = \hat{T}_{z1} - T_1 = 0.05 - 0.028 = 0.022;$$
$$\Delta T_2 = \hat{T}_{z2} - T_2 = 0.25 - 0.154 = 0.096.$$

Визначаємо умову аперіодичного процесу

$$\xi = \frac{T_2}{4 \cdot T_1} \geq 1, \qquad \xi = \frac{0{,}154}{4 \cdot 0{,}028} = 1.38 \rangle 1.$$

Висновок: Постійні часу T_1 та T_2 можуть бути ідентифіковані з достатньою точністю за графіком аперіодичного перехідного процесу (у тимчасовій області).

Приклад. Необхідно дослідити поведінку об'єкта та ідентифікувати його параметри у частотній області

Вихідні дані:

1. Передатна функція об'єкта:

$$W_{0(p)} = \frac{k}{a_0 p^2 + a_1 p + 1}.$$

2. Значення коефіцієнту передачі та постійних часу: $k = 2$, $a_0 = 0{,}2$, $a_1 = 0{,}3$.

Рішення:

1. Складаємо структурну схему дослідження моделі з використанням пакета *Simulink* у середовищі *MATLAB* (рис. 10.9).

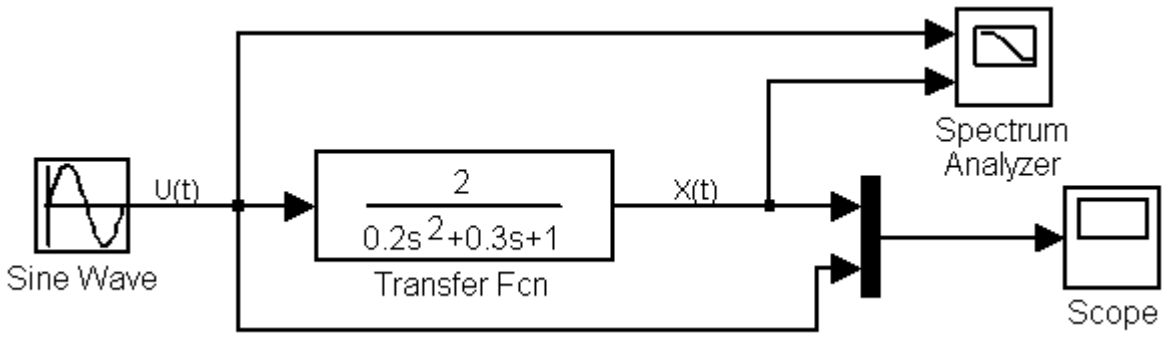


Рисунок 10.9 – Схема дослідження математичної моделі у пакеті MatLab Simulink

Параметри налаштування моделі:

- Час моделювання - 0,0 ÷ 6,0 с;
- Амплітуда синусоїдального сигналу - 1,0;
- Постійна складова (Bias) - 0,0;
- Частота (Frequency) - 1,0; 5,0; 10,0 с-1;
- Мультиплексор - 2 входи, спосіб відображення - bar;
- Осцилограф - координата *Y* - від -15 до 15;

Результати досліджень ідеальної та реальної перехідної характеристики, а також амплітудно-частотної та фазо-частотної характеристик вихідного сигналу при різних частотах керуючого сигналу наведено на рис. 10.10–10.12.

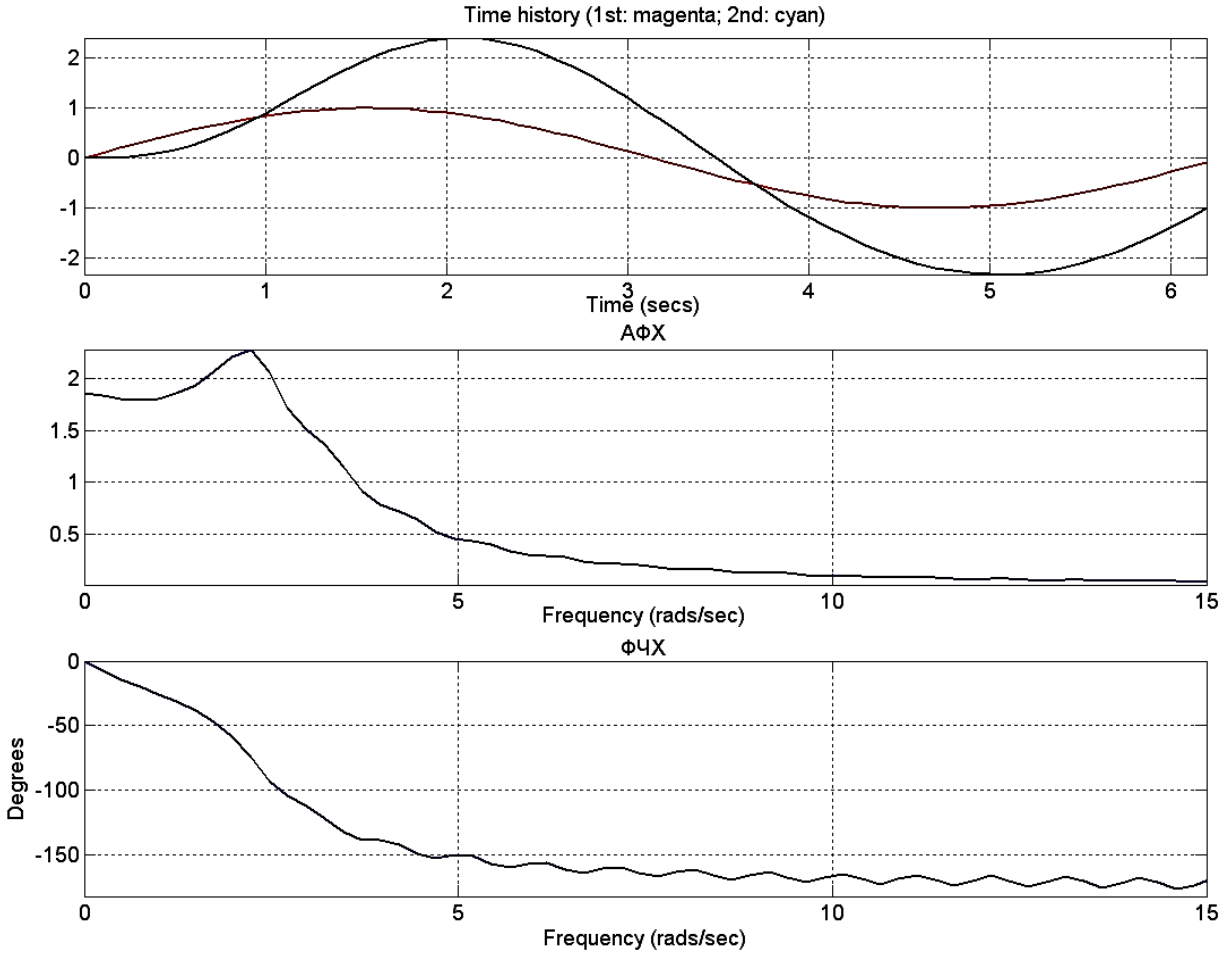


Рисунок 10.10 – Ідеальна та реальна перехідна, амплітудно-частотна та фазо-частотна характеристики вихідного сигналу при частоті $\omega = 1c^{-1}$

Обробка результатів математичного моделювання проводиться у наступній послідовності.

Вхідний сигнал описується виразом

$$U_{i(t)} = 1 \cdot \sin \omega_i t \,,$$

де $\omega_i \in [1;5;10] c^{-1}$.

Вихідний сигнал *X(t)*:

$$X_{i(t)} = A_i \, sin[\, \omega_i t + \phi_{(\omega_i)}].$$

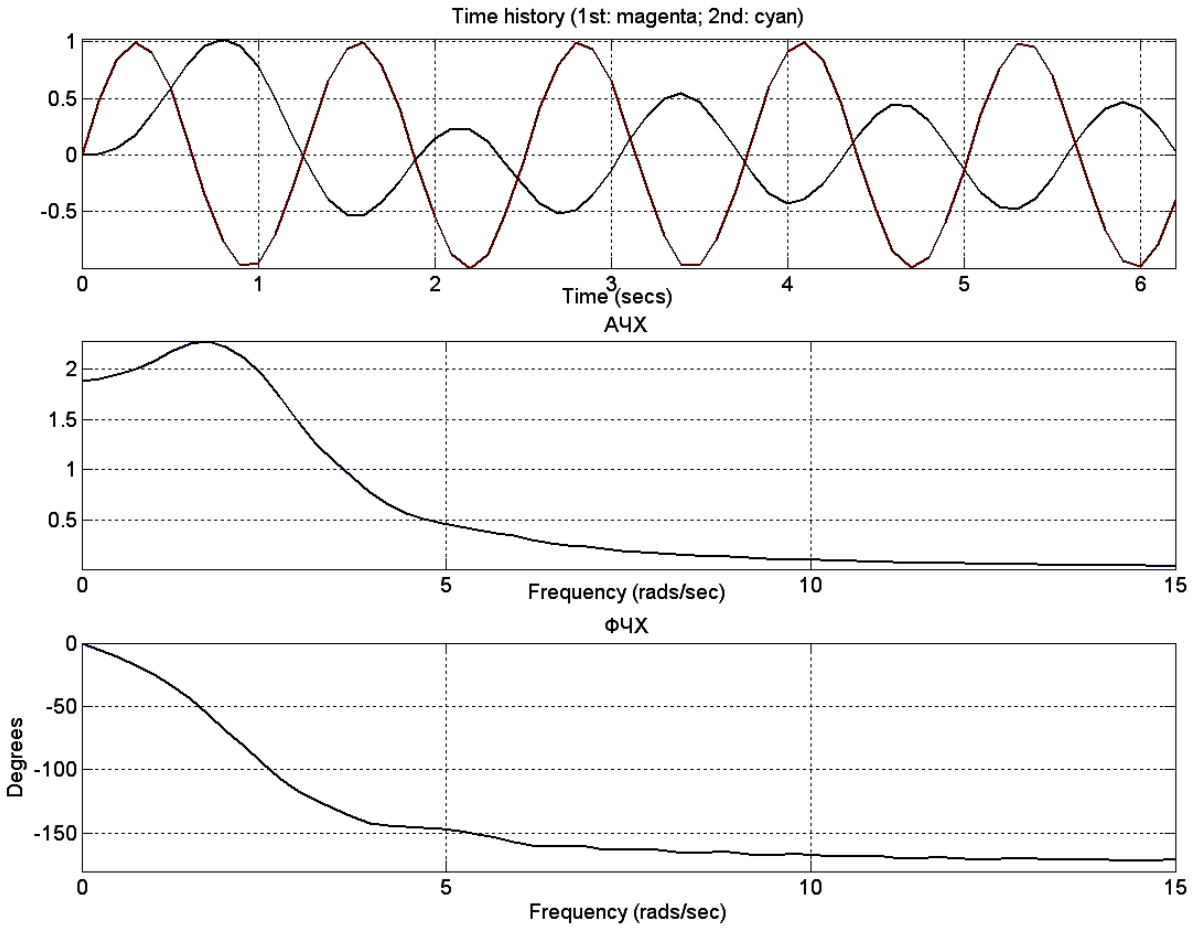


Рисунок 10.11 – Ідеальна та реальна перехідна, амплітудно-частотна та фазо-частотна характеристики вихідного сигналу при частоті $\omega = 5c^{-1}$

Відношення цих сигналів визначається передаточною функцією об'єкта:

$$W_0(p) = \frac{X(t)}{U(t)} = \frac{2}{0{,}2p^2 + 0{,}3p + 1}$$

У загальному випадку передатна функція при переході до полярних координат (амплітудно-фазової характеристики) описується ставленням комплексних функцій чисельника (*R*) і знаменника (*Q*) або

подається в комплексній формі наступним чином:

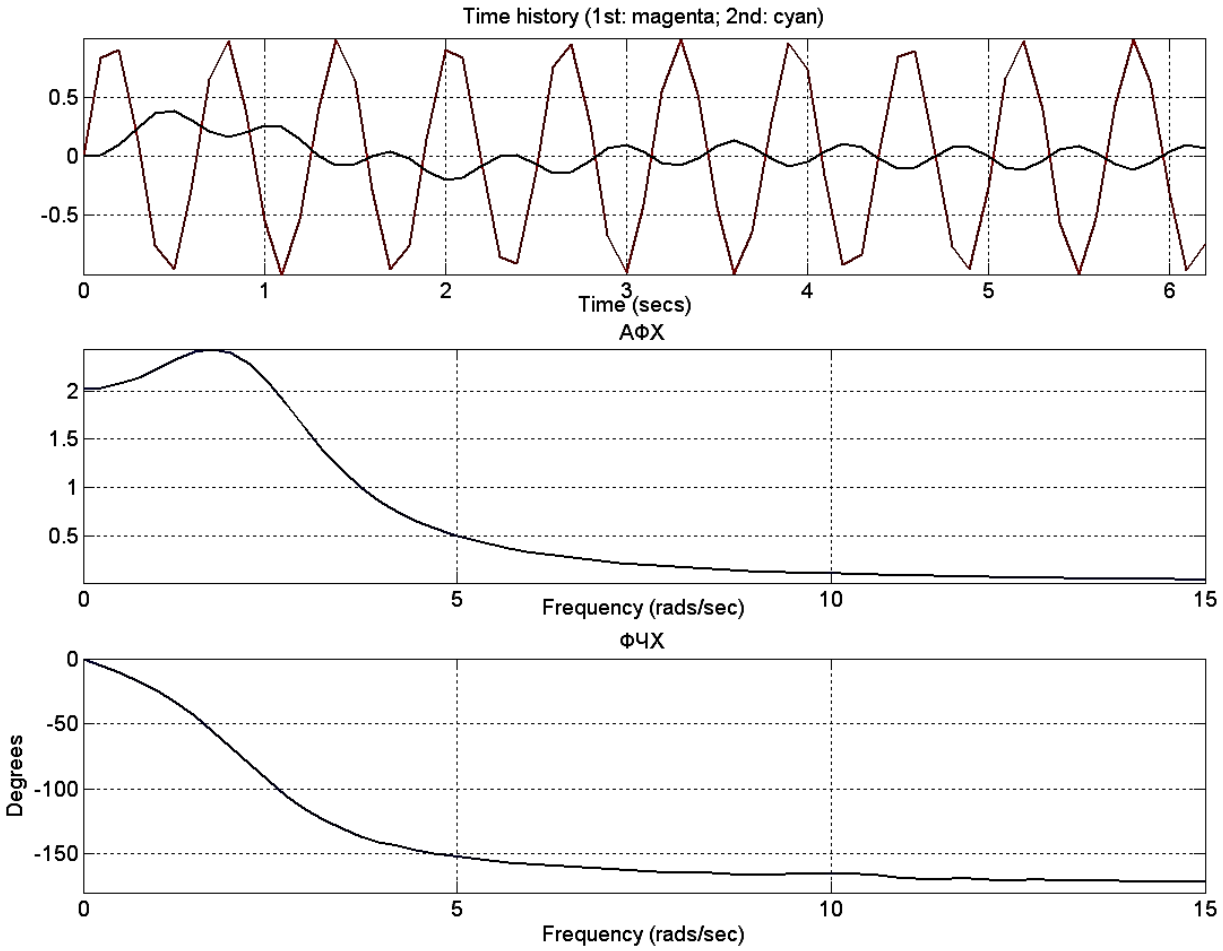


Рисунок 10.12 – Ідеальна та реальна перехідна, амплітудно-частотна та фазо-частотна характеристики вихідного сигналу при частоті $\omega = 10c^{-1}$

$$W_0(j\omega) = \frac{U_R(\omega) + jV_R(\omega)}{U_Q(\omega) + jV_Q(\omega)} = U(\omega) + jV(\omega)$$

Звідси чисельник:

$$U_R(\omega) + jV_R(\omega) = U(\omega)U_Q(\omega) + jV_Q(\omega)U(\omega) + U_Q(\omega)jV(\omega) - V_Q(\omega)V(\omega).$$

Після групування речовинної та уявної складових отримаємо:

$$\begin{cases} U_R(\omega) = U(\omega)U_Q(\omega) - V_Q(\omega)V(\omega), \\ V_R(\omega) = U(\omega)V_Q(\omega) + U_Q(\omega)V(\omega) \end{cases}$$

З графіків на рис. 10.10–10.12 визначаємо:

1. ω_1= 1 c^{-1}, A_1=1,83, φ_1= –25,9;

2. ω_2= 5 c^{-1}, A_2=0,46, φ_2= –147;
3. ω_3= 10 c^{-1}, A_3=0,105, φ_3= –164,5.

Визначаємо речові та уявні характеристики об'єкта в цих точках:

1. ω_1= 1 c^{-1}:

$$U_1 = A_1\, cos\left(\phi_1 \cdot \frac{\pi}{180}\right) = 1{,}83\, cos\left(-25.9\frac{\pi}{180}\right) = 1.646,$$
$$V_1 = A_1\, sin\left(\phi_1 \cdot \frac{\pi}{180}\right) = 1.83 \cdot sin\left(-25.9\frac{\pi}{180}\right) = -0.799.$$

2. ω_2= 5 c^{-1}:

$$U_2 = A_2\, cos\left(\phi_2 \cdot \frac{\pi}{180}\right) = 0.46\, cos\left(-147\frac{\pi}{180}\right) = -0.386,$$
$$V_2 = A_2\, sin\left(\phi_2 \cdot \frac{\pi}{180}\right) = 0.46 \cdot sin\left(-147\frac{\pi}{180}\right) = -0.251.$$

3. ω_3= 10 c^{-1}:

$$U_3 = A_3\, cos\left(\phi_3 \cdot \frac{\pi}{180}\right) = 0.105\, cos\left(-164.5\frac{\pi}{180}\right) = -0.101,$$
$$V_3 = A_3\, sin\left(\phi_3 \cdot \frac{\pi}{180}\right) = 0.105 \cdot sin\left(-164.5\frac{\pi}{180}\right) = -0.028.$$

Визначаємо речові та уявні характеристики чисельника та знаменника АФХ об'єкта:

$$W_0(j\omega) = \frac{k}{a_0(j\omega)^2 + a_1(j\omega) + 1} = \frac{U_R(\omega)}{U_Q(\omega) + jV_Q(\omega)}.$$

де:

$$U_R(\omega) = k;$$
$$jV_R(\omega) = 0;$$
$$U_Q(\omega) = -a_0\omega^2 + 1;$$
$$jV_Q(\omega) = j(a_1\omega).$$

Складаємо систему рівнянь для кожного із трьох експериментальних результатів:

ω_1= 1 c^{-1}:

$$k = 1.646 \cdot (1 - a_0 \cdot 1^2) - a_1 \cdot 1 \cdot (-0.799);$$
$$0 = 1.646 \cdot a_1 \cdot 1 + (1 - a_0 \cdot 1^2) \cdot (-0.799).$$

ω_2= 5 c^{-1}:

$$k = (-0.386) \cdot (1 - a_0 \cdot 5^2) - a_1 \cdot 5 \cdot (-0{,}251);$$
$$0 = (-0{,}386) \cdot a_1 \cdot 5 + (1 - a_0 \cdot 5^2) \cdot (-0{,}251).$$

ω_3= 10 c^{-1}:

$$k = (-0{,}101) \cdot (1 - a_0 \cdot 10^2) - a_1 \cdot 10 \cdot (-0{,}028);$$
$$0 = (-0{,}101) \cdot a_1 \cdot 10 + (1 - a_0 \cdot 10^2) \cdot (-0{,}028).$$

Із застосуванням пакету MathCad визначаємо рішення системи рівнянь при ω_1= 1 c^{-1} та ω_3= 10 c^{-1} відностно k, a_0, a_1:

Given

$$U1 \cdot a1 \cdot \omega1 + V1 \cdot \left[1 - a0 \cdot (\omega1)^2\right] = 0$$

$$U3 \cdot a1 \cdot \omega3 + V3 \cdot \left[1 - a0 \cdot (\omega3)^2\right] = 0$$

$a := \text{Find}(a0, a1)$

$a_0 = 0.158$

$a_1 = 0.409$

$k := U1 \cdot \left(1 - a_0 \cdot \omega1^2\right) - V1 \cdot a_1 \cdot \omega1$ $\quad k = 1.714$

$k := U2 \cdot \left(1 - a_0 \cdot \omega2^2\right) - V2 \cdot a_1 \cdot \omega2$ $\quad k = 1.646$

$k := U3 \cdot \left(1 - a_0 \cdot \omega3^2\right) - V3 \cdot a_1 \cdot \omega3$ $\quad k = 1.607$

Для побудови амплітудно-фазової частотної характеристики у командному вікні середовища MatLab скористаємося командою nyquist(sys), а саме

```
H=tf([2],[0.2 0.3 1])
nyquist(H)
```

Результати виконання команди наведені на рисунку 10.13

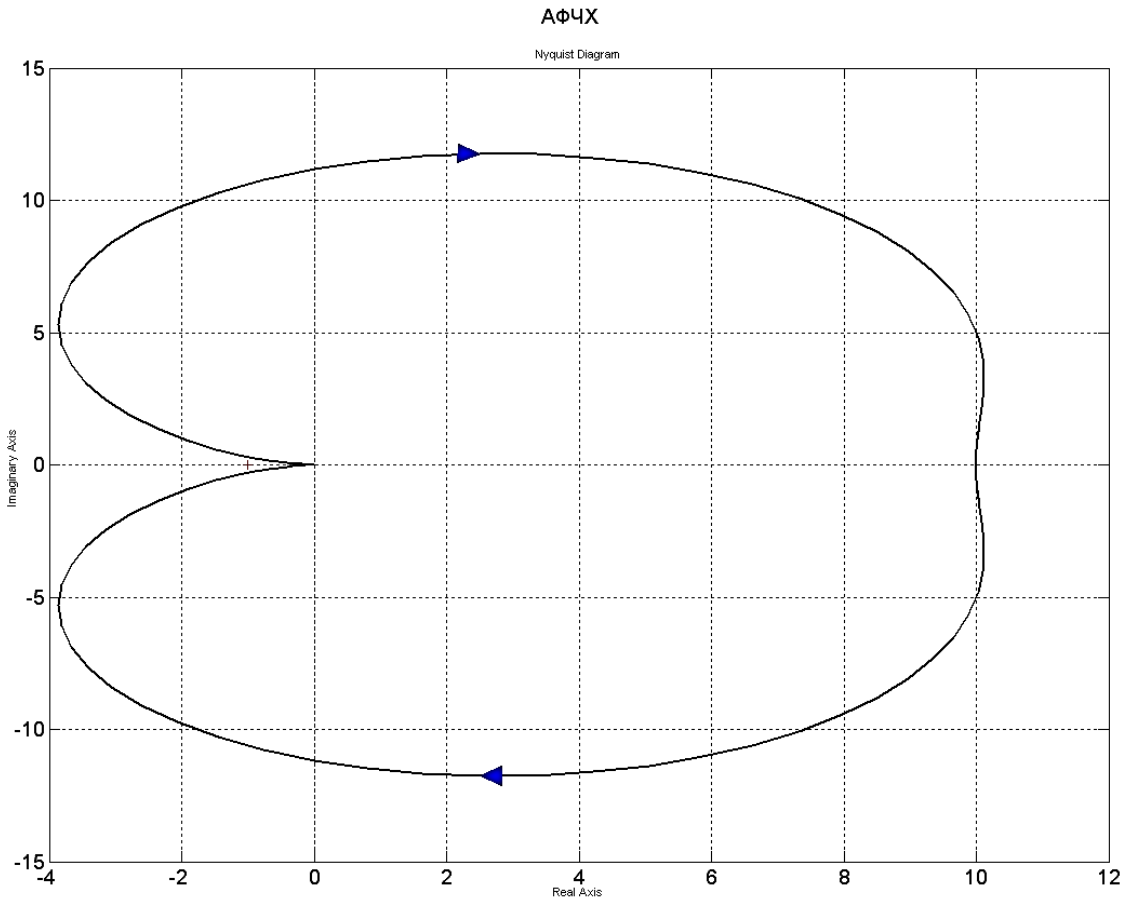


Рисунок 10.13 – Амплітудно-фазова частотна характеристика математичної моделі

10.3 ПІД-регулятори

У промисловості найчастіше використовується пропорційно-інтегро-диференціальний закон управління, що описується формулою

$$u(t) = k_p e(t) + k_d \frac{de(t)}{dt} + k_i \int e(t)dt, \qquad (10.4)$$

де k_p, k_d та k_i – параметри закону управління.

Закон (10.4) реалізується за допомогою цифрових чи аналогових ПІД-регуляторів (рис. 10.14), а також їх спрощених модифікацій: П, ПД, ПІ та І-регуляторів, що виходять при різних комбінаціях доданків, що входять до формули (10.4).

Керуюча поверхня ПІД-регулятора є гіперплощиною, ПД і ПІ-регуляторів - площиною, поведінка П та І-регуляторів описується прямою, тобто всі ці регулятори є лінійними.

Компоненти ПІД-регулятора мають різний вплив на перехідний процес (табл. 10.1).

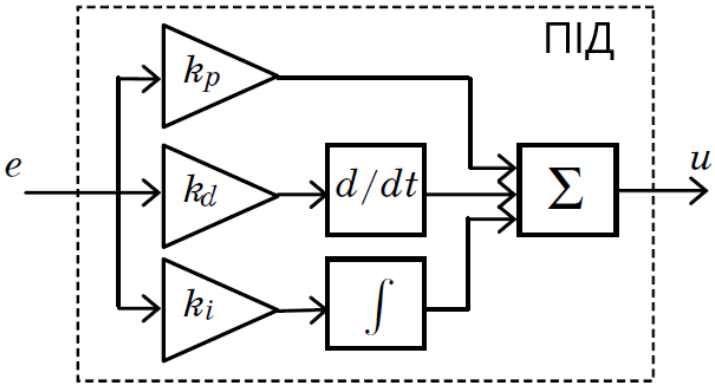


Рисунок 10.14 – Структура ПІД-регулятора

Таблиця 10.1 - Ефекти зі збільшенням коефіцієнтів ПІД-регулятора

Коефіцієнт	Час наростання (швидкодія)	Встановлена помилка	Перерегулювання
Пропорційний	зменшується	зменшується	збільшується
Диференційний	не впливає	не впливає	зменшується
Інтегральний	зменшується	усуває	збільшується

Таким чином, у формулі (10.4) перший доданок відповідає за швидкодію системи (час наростання), а також за величину помилки, що встановилася. Друге доданок дозволяє збільшити демпфування системи, тобто придушити небажані коливання та зменшити перерегулювання. Третій доданок дозволяє зменшити до нуля помилку, що встановилася в системі.

Якщо математична модель об'єкта відома досить точно, параметри ПІД-регулятора можна визначити в результаті обраної процедури оптимізації, наприклад за допомогою генетичного алгоритму.

Загалом ПІД-регулятори можна віднести до категорії експертних регуляторів, оскільки вони можуть налаштовуватися шляхом безпосередніх експериментів з об'єктом відповідно до відомих методів (Зіглера – Ніколса та ін.).

Метод Зіглера – Ніколса може бути формульовано у двох варіантах – для замкнутої та розімкнутої системи. Розглядаються П, ПІ та ПІД-регулятори.

Перепишемо закон управління ПІД-регулятора у вигляді передаточної функції

$$H(p) = K_p\left(1 + T_d p + \frac{1}{T_i p}\right).$$

Розглянемо перший варіант (замкнута система):

1) коефіцієнти k_d і k_i встановлюються рівними нулю, а коефіцієнт

k_p збільшується до тих, доки в системі не виникнуть автоколивання.

2) позначимо граничне значення k_p як P, а період автоколивань як T.

3) значення коефіцієнтів регулятора розраховуються відповідно до табл. 10.2.

Таблиця 10.2 – Розрахунок коефіцієнтів регулятора (варіант 1)

	k_p	T_i	T_d
П	0,5P		
ПІ	0,45P	T/1,2	
ПІД	0,6P	$T/2$	$T/8$

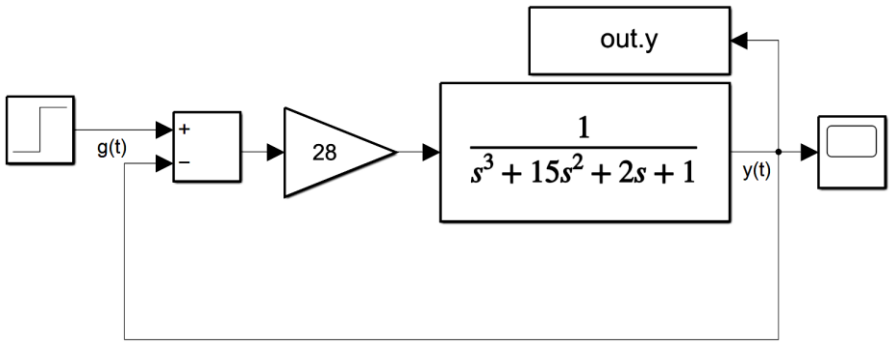
Нехай об'єкт управління описується передаточною функцією

$$W(p) = \frac{1}{p^3 + 15p^2 + 2p + 1}.$$

Потрібно розрахувати параметри ПІД-регулятора.

Схема моделювання у Simulink MatLab показана на рис. 10.15.

а)



б)

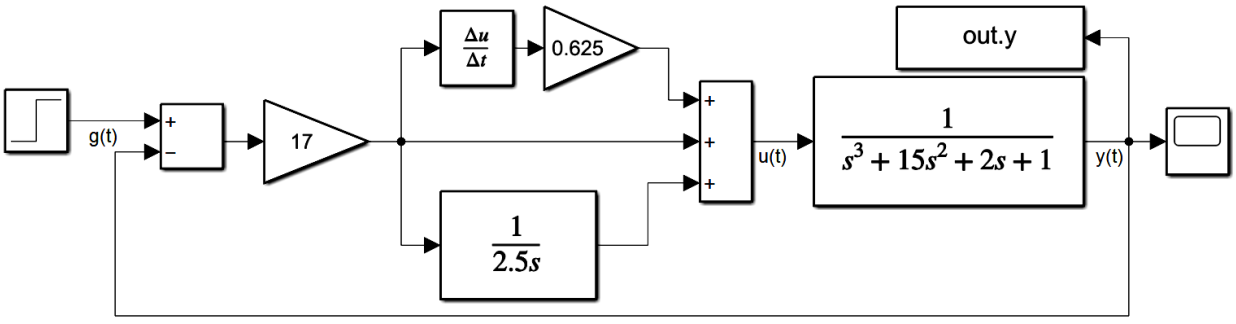


Рисунок 10.15 – Математична модель роботи а) П-регулятора та б) ПІД-регулятора

Автоколивання у системі виникають при k_p =28 (рис. 10.16).

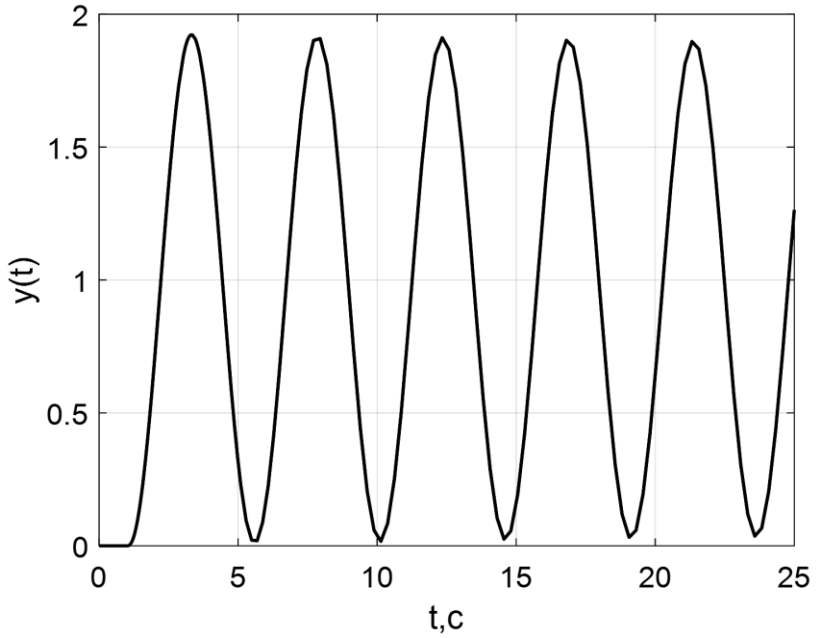


Рисунок 10.16 - Незагасні коливання в системі управління

Таким чином, $P = 28$, $T \approx 5$. Відповідно до табл. 10.2 отримуємо:

$$K_P \approx 17;\ T_d \approx 0.625;\ T_i \approx 2.5.$$

Перехідний процес для ПІД-регулятора показано на рис. 10.17.

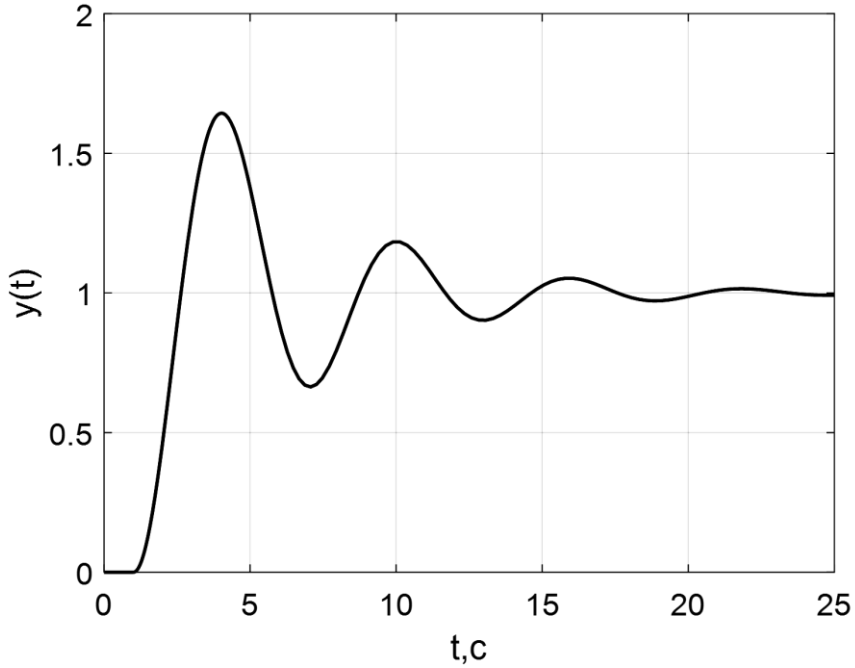


Рисунок 10.17 – Перехідний процес у системі з ПІД-регулятором

Розглянемо другий варіант (розімкнута система).
На вхід системи подається одиничний стрибок. Незалежно від

того, чи є система стійкою чи нестійкою, будується дотична до точки перегину кривої перехідного процесу, і визначаються два параметри: *R* та *L* (рис. 10.18). Далі коефіцієнти обраного типу регулятора розраховуються відповідно до табл. 10.3.

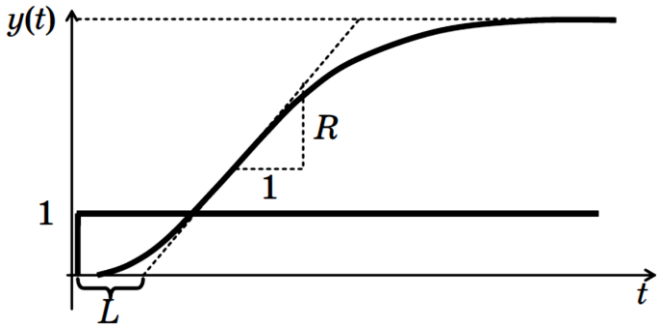


Рисунок 10.18 – Визначення параметрів для стійкої системи

Таблиця 10.3 – Розрахунок коефіцієнтів регуляторів (варіант 2)

	k_p	T_i	T_d
П	1/(*RL*)		
ПІ	0,9/(*RL*)	3*L*	
ПІД	1,2/(*RL*)	3*L*	0,5*L*

Нехай об'єкт управління описується передаточною функцією

$$W(p) = \frac{1}{15p^2 + 8p + 1}.$$

Необхідно розрахувати параметри ПІД-регулятора. Математична модель об’єкта наведена на рисунку 10.19.

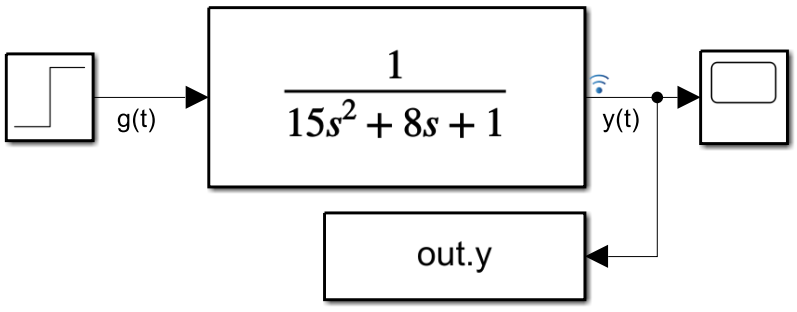


Рисунок 10.19 – Математична модель об’єкта

Крива перехідного процесу для розімкнутої системи показано на рис. 10.20.

За кривою перехідного процесу, показаної на рис. 10.20, визначаємо

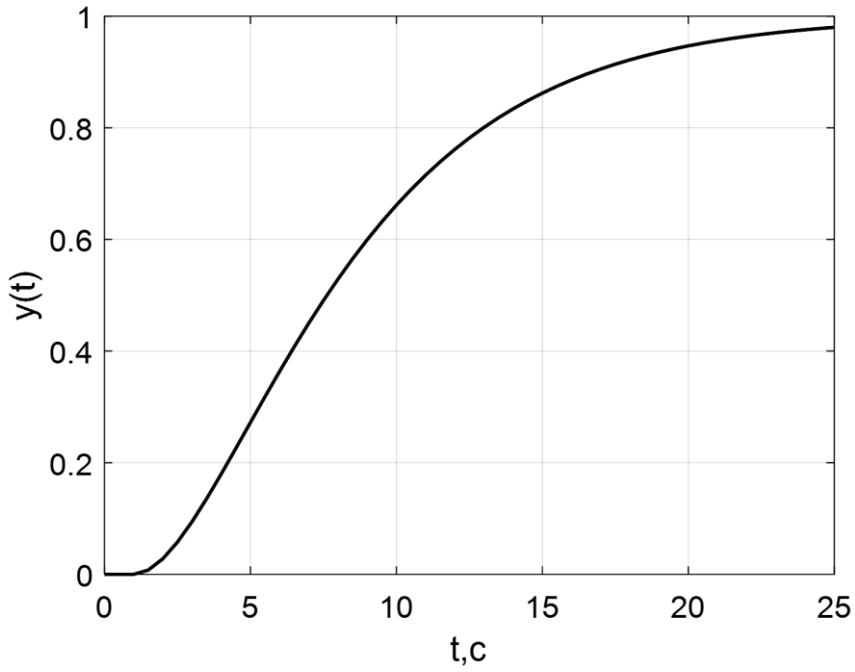


Рисунок 10.20 – Графік перехідного процесу для системи без регулятора

$L \approx 1$; $R \approx 0.1$.

Потім за формулами табл. 10.3 маємо

$K_P \approx 12$; $T_d \approx 0.5$; $T_i \approx 2$.

Математична модель роботи замкненої системи управління з ПІД-регулятором наведена на рис.10.21.

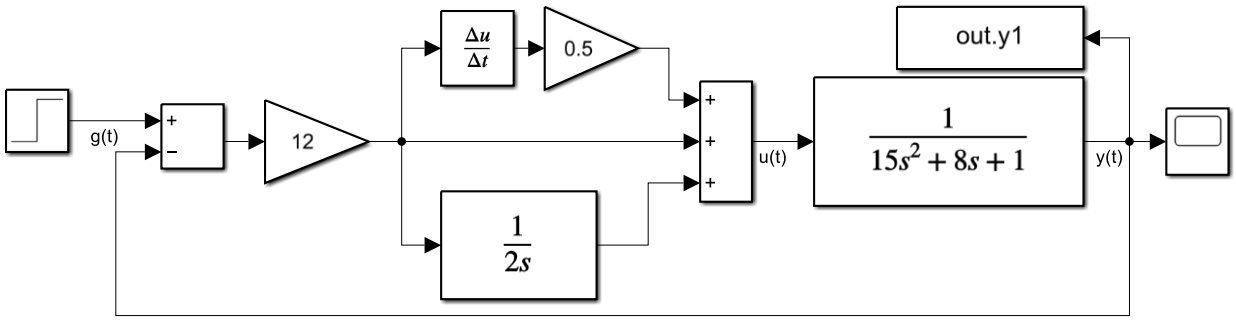


Рисунок 10.21 - Математична модель роботи замкненої системи управління з ПІД-регулятором

Перехідний процес для замкнутої системи показано на рис. 10.22.

Як свідчать рис. 10.17 та 10.22, для ПІД-регуляторів, розрахованих за методикою Зіглера – Ніколса, характерне велике перерегулювання (порядку 60%).

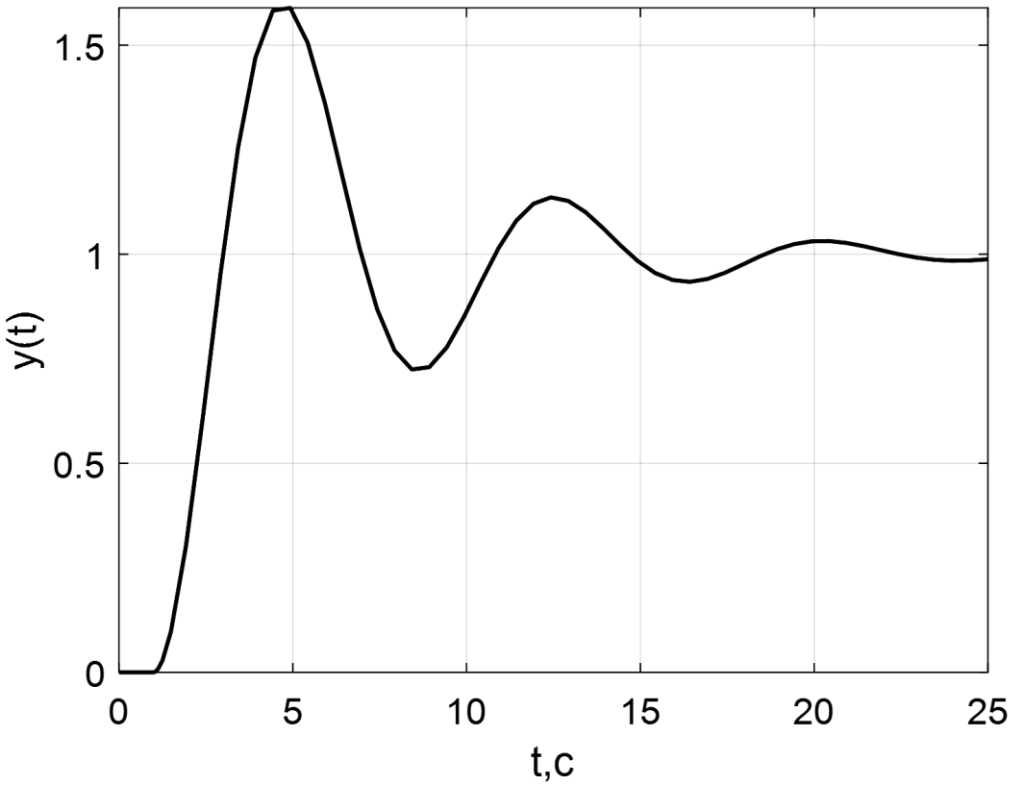


Рисунок 10.22 – Графік перехідного процесу в замкненій системі з ПІД-регулятором (варіант 2)

Нечіткі регулятори можуть розглядатися як природний розвиток ідей ПІД-управління, при правильному проектуванні вони забезпечують кращі експлуатаційні показники, тобто можуть зменшити перерегулювання, час перехідного процесу і помилку, що встановилася. Для НЛР, як буде показано нижче, можна запропонувати досить прості експертні алгоритми налаштування.

10.4 Структури нечітких регуляторів

Зазвичай НЛР включаються послідовно з об'єктом управління, подібно до традиційних регуляторів (див. рис. 10.1). Вхідні та вихідні сигнали НЛР піддаються перетворенню (див. рис. 10.23).

У загальному випадку передобробка вхідного сигналу може включати такі операції, як:

- квантифікація (округлення);
- нелінійне шкалювання;
- фільтрація та видалення шумів;
- диференціювання та (або) інтегрування;
- усереднення на певному інтервалі часу, екстраполяція, інтерполяція;
- обчислення довільних функцій над комбінаціями вхідних змінних отримання семантично значимих значень.

Рисунок 10.23 – Обробка сигналів на вході та виході НЛР

Однак найчастіше передобробка зводиться до нормалізації (*N*) вхідного сигналу - приведення його до діапазону [-1, 1]. Тоді постобробка зводиться до денормалізації (*DN*) вихідного сигналу НЛР (рис. 10.24).

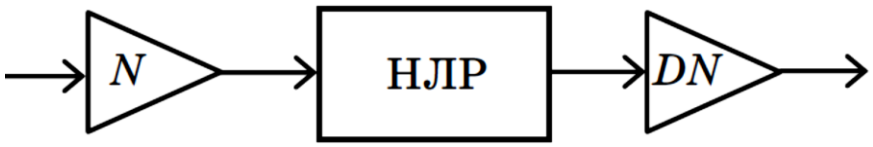


Рисунок 10.24 – Нормалізація та денормалізація сигналів на вході та виході НЛР

Якщо *x* - вхідна змінна НЛР, то нормалізація відбувається шляхом множення *x* на коефіцієнт, що масштабує:

$$x_N = xN = \frac{x}{|x_{max}|}.$$

Денормалізація вихідної величини y зводиться до зворотної операції:

$$x_N = xDN = y_N y_{max},$$

де y_{max} - максимальне значення керування, що подається на об'єкт.

Кількість і вид вхідних і вихідних сигналів НЛР залежать від задачі, що вирішується, проте найчастіше входами НЛР служать (у різних комбінаціях) помилка управління, її похідна та інтеграл.

Відповідно розрізняють НЛР П-типу, ПД-типу, ПІ-типу або ПІД-типу (рис. 10.25-10.28).

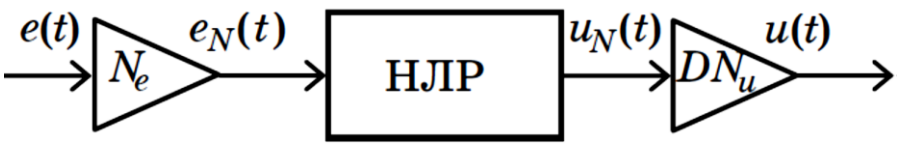


Рисунок 10.25 – НЛР П-типу (НЛР_П)

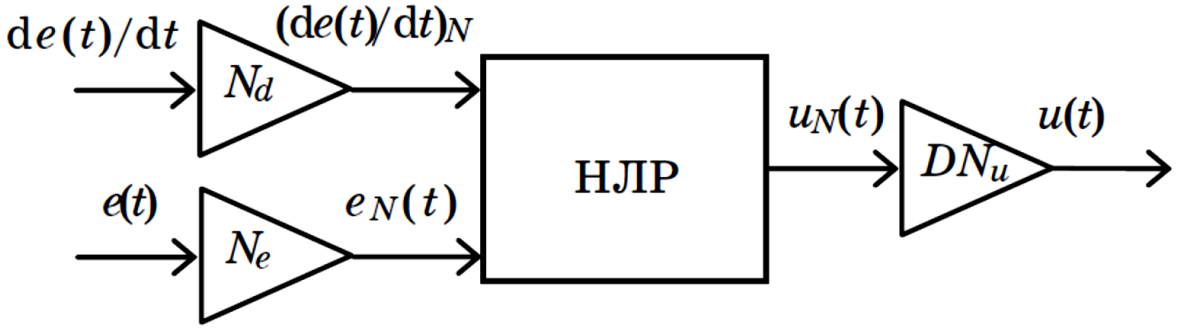


Рисунок 10.26 – НЛР ПД-типу (НЛР_ПД)

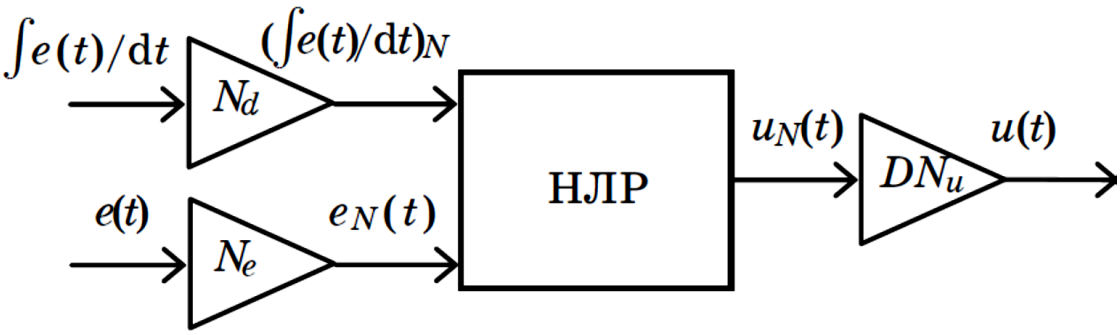


Рисунок 10.27 – НЛР ПІ-типу (НЛР_ПІ)

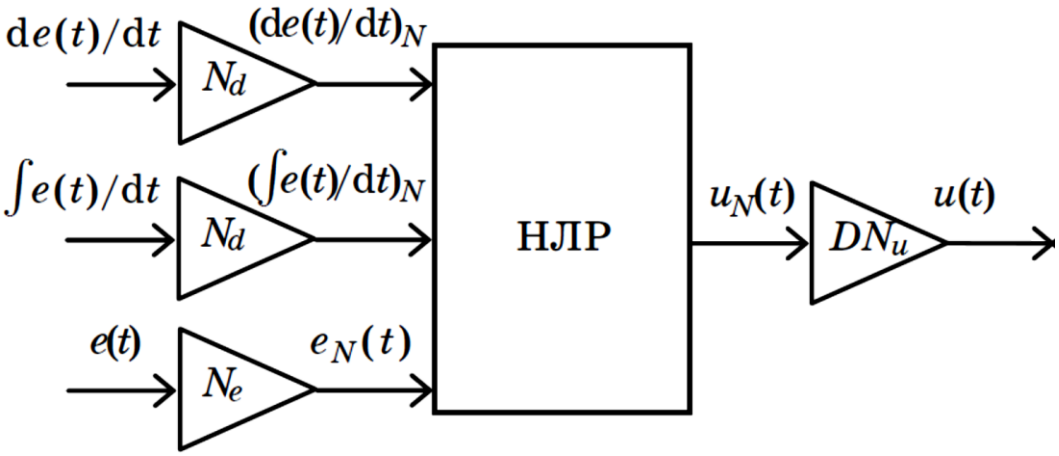


Рисунок 10.28 – НЛР ПІД-типу (НЛР_ПІД)

Позначимо через *E* вхідний вектор НЛР (рис. 10.29), він перетворюється на нечітку форму *E** (блок фазифікації *F*), потім виконується нечіткий логічний висновок у основі правил, у результаті виходить нечітка вихідна змінна *u**. Після дефазифікації (блок *DF*) на об'єкт управління надходить чіткий сигнал управління *u*.

Операції фазифікації та дефазифікації залежать від параметрів термів, що описують вхідні та вихідні змінні. Таким чином, можна сказати, що, крім механізму виведення в НЛР, існує база знань, що включає опис термів і правил (рис. 10.30).

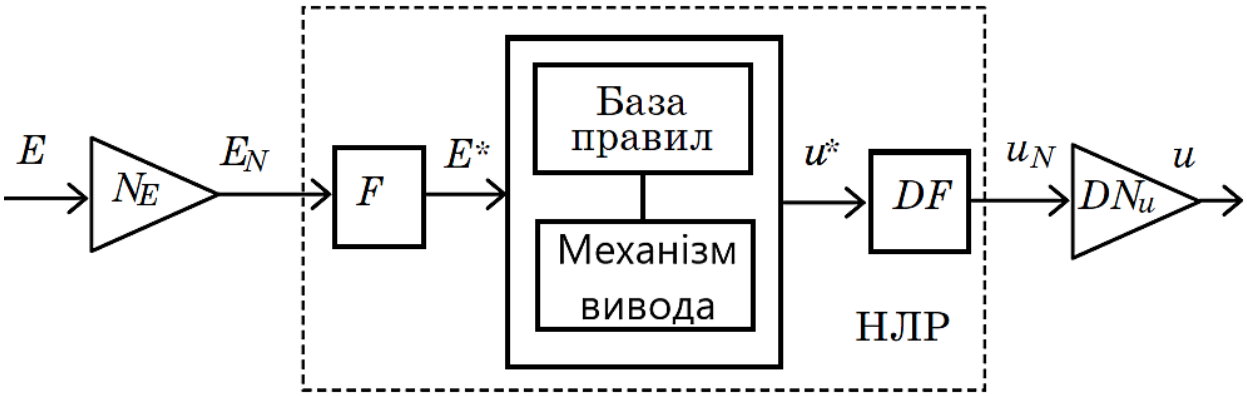


Рисунок 10.29 – НЛР П-типу

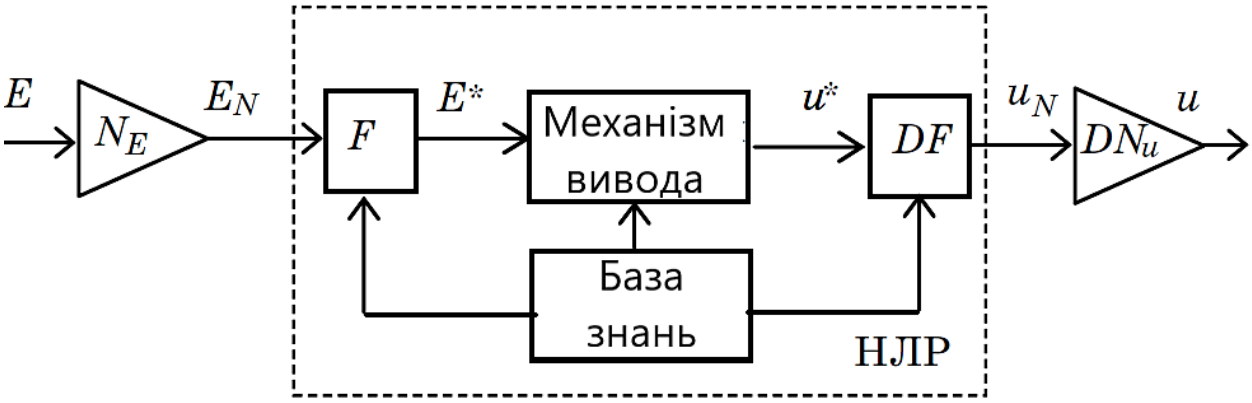


Рисунок 10.30 – Уточнений варіант опису НЛР П-типу

Запишемо *k*-е правило управління НЛР _ПІД як

$$\text{Ящко } (x_1 = A_1^k) \text{ та } (x_2 = A_2^k) \text{ та } (x_3 = A_3^k), \text{то } y = B_k,$$

Нечіткий висновок реалізується у декілька кілька кроків:

1. Виконується фазифікація – обчислюються значення $\alpha_{i,k}$ – ступінь відповідності *i* вхідної змінної *i* посилки *k*-го правила:

$$\alpha_{i,k} = x_i^* \cap A_i^*$$

де x_i^* - нечітка множина (синглетон), утворена з x_i.

2. Визначається ступінь відповідності всіх посилок *k*-го правила поточним вхідним даним:

$$\beta_k = T_{i=1}^N \alpha_{i,k},$$

де *N* – кількість посилок правила; *T* - використовувана *T*-норма; β_k - ступінь спрацьовування (запуску) правила.

3. Визначається вихідний сигнал кожного правила:

$$\mu_{B_k'}(y) = I\left(\beta_k, \mu_{B_k}(y)\right),$$

де *I* - використовувана імплікація; $\mu_{B_k}(y)$ – функція приналежності висновку *k*-го правила.

4. Загальний вихідний сигнал системи правил *B*' визначається шляхом агрегування вихідних сигналів окремих правил:

$$\mu_{B`}(y) = \bigcup_k \mu_{B_k'}(y).$$

Таким чином, закон управління реалізується як сукупна дія всіх правил, записаних у основі правил:

$$R = R_1 \vee R_2 \vee R_3 \vee \ldots \vee R_k.$$

Відповідно до рис. 10.30 синтез НЛР пов'язаний із визначенням наступних параметрів:

1) коефіцієнтів нормалізації та деноралізації;

2) функцій належності термів лінгвістичних змінних;

3) керуючих правил НЛР.

Можливі різні постановки задачі синтезу НЛР, але для надання їй реалістичного характеру бажано мінімізувати кількість параметрів.

В залежності від кількості входів змінюється кількість посилок нечітких керуючих правил.

Для НЛР _П кількість правил дорівнює кількості термів, що описують лінгвістичні змінні «Помилка управління». Для інших типів НЛР потенційна кількість правил дорівнює декартову добутку потужностей терм-множин вхідних змінних. Якщо вважати, що кожна вхідна змінна має однакову кількість термів, число правил можна оцінити відповідно до табл. 10.4.

Таблиця 10.4 - Потенційна кількість правил для різних типів НЛР

Кількість термів	Кількість правил		
	НЛР_П	НЛР_ПД(ПІ)	НЛР_(ПІД)
3	3	9	27
5	5	25	125
7	7	49	343

Для спрощення завдання проектування НЛР ПІД використовуються різні варіанти запису рівняння (10.4). Перепишемо рівняння ПІД-регулятора (10.4) у вигляді:

$$u(t) = \left(\frac{1}{2}k_p e(t) + k_d \frac{de(t)}{dt}\right) + \left(\frac{1}{2}k_p e(t) + k_i \int e(t)dt\right).$$

Перший доданок описує ПД-регулятор, другий доданок - ПІ-регулятор, тобто ПІД-регулятор можна подати у вигляді паралельного з'єднання цих регуляторів. Відповідно, НЛР _ПІД можна як структури, показаної на рис. 10.31.

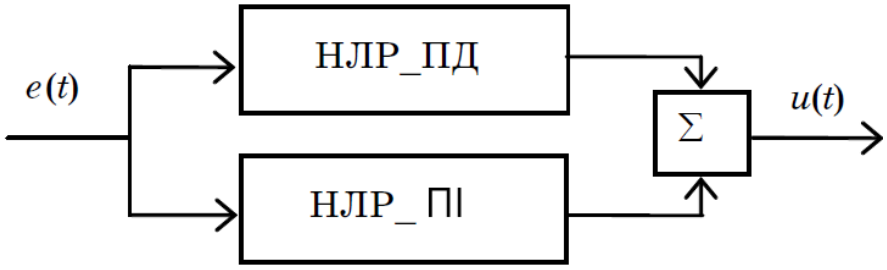


Рисунок 10.31 – Спрощена структура НЛР_ПІД

Інший варіант реалізації ґрунтується на поданні (10.4) у вигляді

$$u(t) = k_p\left(e(t) + \frac{k_d}{k_p}\frac{de(t)}{dt}\right) + k_i \int e(t)dt.$$

Тут перший доданок описує ПД-регулятор, що відповідає за час наростання і перерегулювання, а другий доданок - І-регулятор, який служить для зменшення статичної помилки. Таким чином, ПД-регулятор повинен передавати управління І-регулятору, коли $e(t)$ та $\frac{de(t)}{dt}$ стають малими (рис. 10.32).

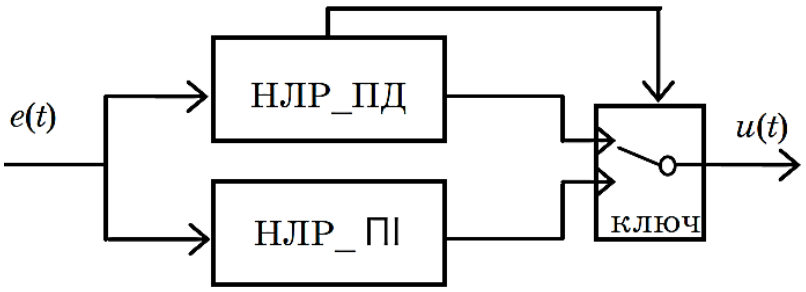


Рисунок 10.32 – Варіант структури НЛР_ПІД

Нечіткі логічні регулятори реалізуються зазвичай з урахуванням мікроконтролерів. У дискретному НЛР замість похідної помилки управління розглядається її збільшення $\Delta e(t) = e(t) - e(t-1)$.

Значення коефіцієнтів, що масштабують, не змінюються.

Поведінку НЛР _ПД можна описати виразом

$$u(t) = F_л(e(t), \Delta e(t)),$$

де $F_л$ – лінгвістичний закон управління, заданий набором правил (рис. 10.33, де z^{-1} означає операцію затримки однією такт).

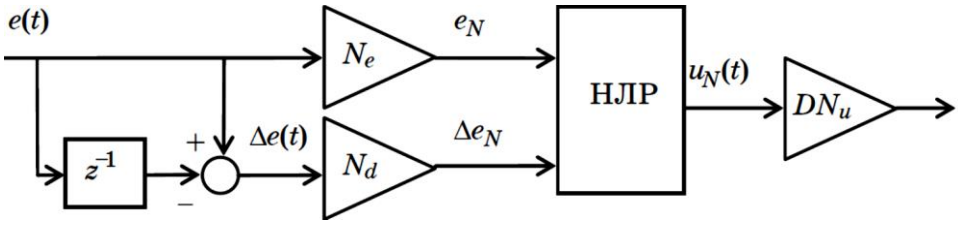


Рисунок 10.33 – Дискретний опис НЛР_ПД

Нехай k – номер кроку квантування за часом, тоді:

$$\Delta e = e(k) - e(k-1) = (1 - z^{-1})e(k)$$

та цифровий НЛР _ПД можна подати у вигляді, показаному на рис. 10.34.

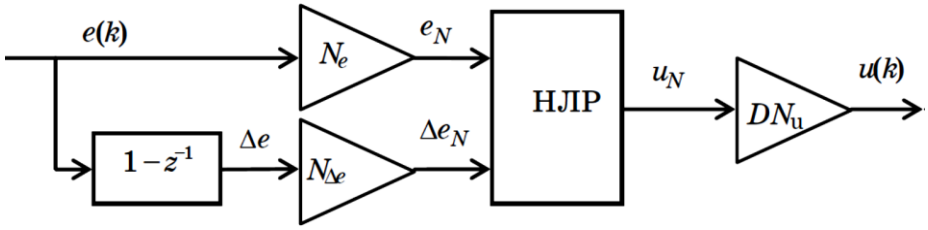


Рисунок 10.34 – Варіант дискретного опису НЛР_ПД

Пропорційно-інтегральний контролер для кожної пари значень помилки та її інтеграла обчислює значення управління:

$$u(t) = k_p e(t) + k_i \int e(t)dt. \qquad (10.5)$$

Розглянемо похідну рівняння (10.5):

$$\frac{du(t)}{dt} = k_p \frac{de(t)}{dt} + k_i e(t)$$

У дискретній формі замість похідної розглядається збільшення, а замість інтеграла – сума:

$$\Delta u(t) = k_p \Delta e(t) + k_i e(t),$$

де $\Delta u(t) = u(t) - u(t-1)$.

Таким чином, поведінка НЛР _ПІ описується системою лінгвістичних правил виду:

$$\Delta u(t) = F_{\text{Л}}(e(t), \Delta e(t))$$

і структура НЛР _ПІ збігається із структурою НЛР _ПД (рис. 10.33 і 10.34) за винятком того, що тут розглядається не сигнал управління, а його прирощення (рис. 10.35).

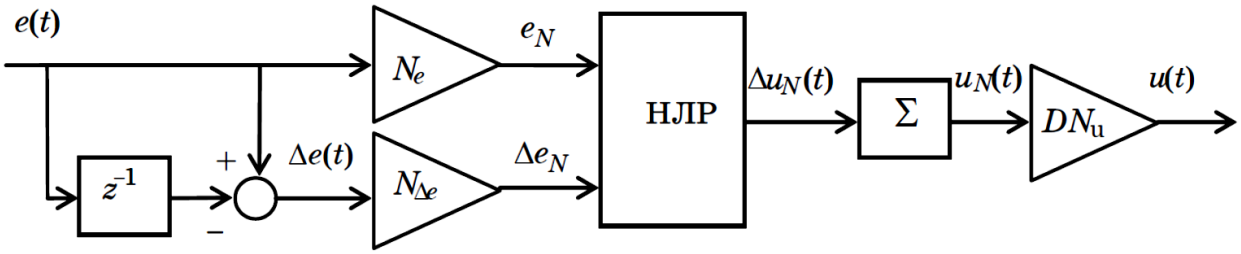


Рисунок 10.35 – Дискретний опис НЛР_ПІ

Блок підсумовування зазвичай описується як, показаному на рис. 10.36.

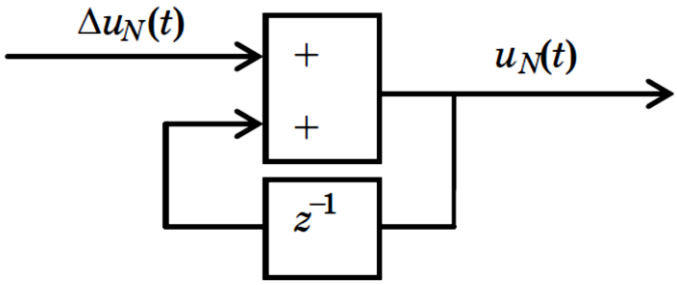


Рисунок 10.36 – Дискретна реалізація інтеграла

Розглядаючи прирощення сигналу управління

$$\Delta u = u(k) - u(k-1) = (1 - z^{-1})u(k)$$

можливо записати

$$u(k) = \frac{\Delta u}{1 - z^{-1}},$$

і тоді дискретний НЛР _ПІ можна подати у вигляді, показаному на рис. 10.37.

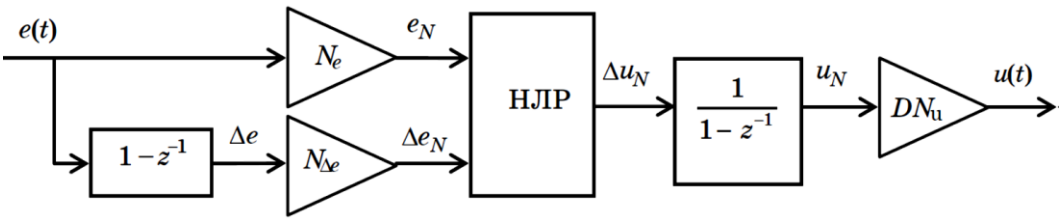


Рисунок 10.37 – Варіант дискретного опису НЛР_ПІ

Крім описаних типових структур НЛР, можливі інші варіанти використання НЛР в системі управління. Наприклад, НЛР може бути компенсації зовнішніх шумів (рис. 10.38).

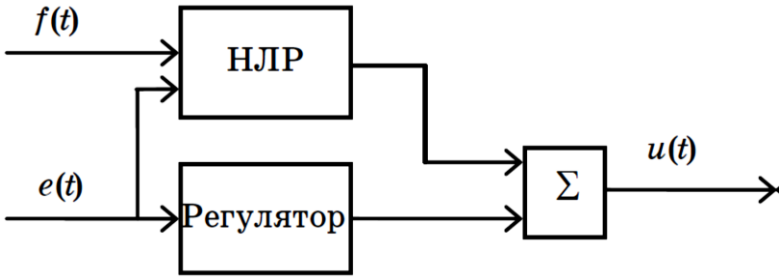


Рисунок 10.38 – Нечітка компенсація зовнішніх обурень

Можливе використання НЛР у складі адаптивної системи для зміни параметрів Δ*p* основного регулятора при погіршенні якості функціонування (рис. 10.39).

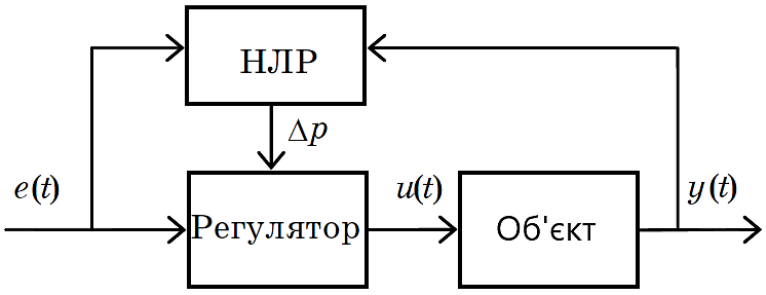


Рисунок 10.39 - Включення НЛР у контур адаптації

У ряді випадків ефективною також може бути ієрархічна структура з НЛР, в якій регулятори нижнього рівня здійснюють первинну обробку інформації.

10.5 Методи синтезу нечітких регуляторів

Існують різні підходи до синтезу основи знань НЛР. Історично першими були варіанти, пов'язані з формалізацією знань експерта, який керує об'єктом вручну. Знання можуть бути формалізовані

безпосередньо шляхом інтерв'ювання експерта або шляхом спостереження за його діями з управління.

Якщо ж експерт відсутній, то НЛР може налаштовуватись у процесі експериментів з об'єктом. У цьому випадку процес налаштування НЛР нагадує процес налаштування лінійного ПІД-регулятора.

Перелічені підходи не вимагають використання математичного опису об'єкта управління, але потребує експериментів із реальним об'єктом управління, що у більшості випадків небажано. Найвигідніше використовувати комп'ютерну (імітаційну) модель.

Якщо є досить достовірна комп'ютерна модель об'єкта управління, з її допомогою можна реалізувати безліч варіантів налаштування НЛР, зокрема – виходячи з ідентифікації, адаптації, еволюційної самоорганізації.

У найпростіших випадках при синтезі НЛР для однозв'язкових об'єктів можна використовувати евристичні міркування, які спираються аналіз вимог до протікання перехідного процесу.

Ефективним інструментом для представлення НЛР є штучні нейронні мережі (НМ), у тому числі мережі прямого поширення та RBF-мережі (радіальні базисні функції). Для них розроблено перевірені методи навчання.

Методи навчання нейронної мережі зазвичай поділяють на дві групи – залежно від того, чи задана у явному вигляді навчальна інформація. Для опису варіантів синтезу НЛР можна використовувати таку класифікацію (рис. 10.40).

Рисунок 10.40 – Класифікація методів синтезу НЛР

Варіант синтезу шляхом кластеризації виникає, коли є вхід-вихідні залежності (навчальні пари), що пов'язують ситуацію на об'єкті та реакцію регулятора (або оператора) у цій ситуації. Мета кластеризації полягає в стисканні великого масиву навчальної інформації в компактну множину нечітких правил. Нижче буде розглянуто реалізовані в MatLab методи С–середніх та вершинна (гірська) кластеризація. До цих методів примикає і нейронне подання НЛР за допомогою нейронної мережі Anfis, параметри якої налаштовуються за навчальними даними і можуть

бути легко описані за допомогою системи нечітких правил.

Пошукова оптимізація НЛР передбачає використання певної цільової функції, яка описує бажану поведінку системи управління. Тут може бути використана еталонна модель, тоді цільовою функцією буде функція нев'язки між виходами системи та моделі. Багатоекстремальний характер цільової функції вимагає використання методів глобальної оптимізації, таких як генетичний алгоритм, метод відпалу металу, метод мурашиних колоній або рійний інтелект.

Завдання синтезу НЛР може бути спрощена, якщо враховувати модульність системи продукційних правил. Окремі правила відповідають за локальні області простору станів системи, тому кожне правило породжує свій власний елементарний вихідний сигнал. Загальний вихідний сигнал є об'єднанням (сумою) цих елементарних сигналів. Таким чином, систему нечітких правил можна подати у вигляді, показаному на рис. 10.41.

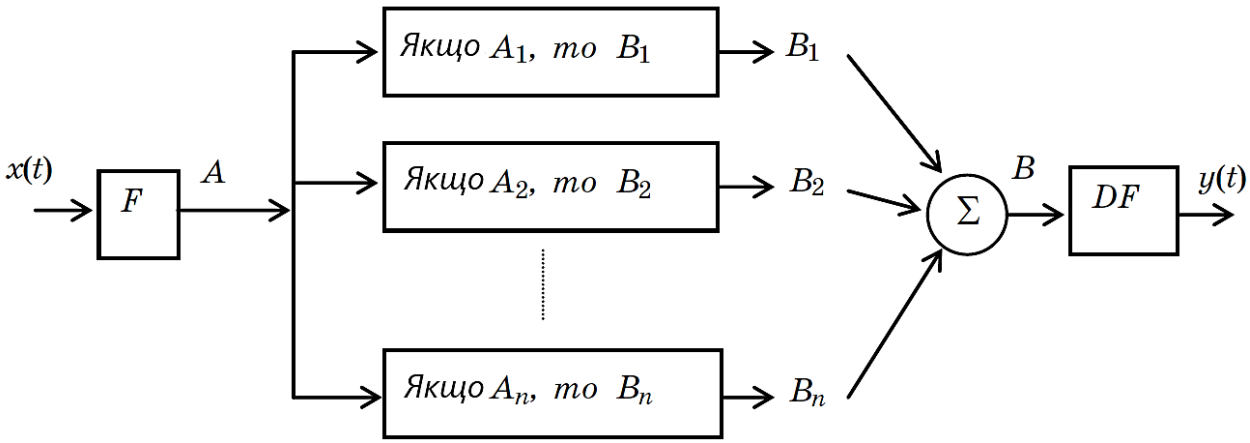


Рисунок 10.41 – Адитивна нечітка система

Розглядаючи локальні області простору станів системи, можна синтезувати окремі керуючі правила, та був використовувати їх разом отримання глобального закону управління.

10.6 Евристичний синтез нечіткого регулятора П-типу

Класичний регулятор П-типу (тобто пропорційний) є лінійним регулятором, в якому сигнал управління розраховується шляхом множення вхідної помилки на задану константу (коефіцієнт пропорційності):

$$u(t) = k_p\big(g(t) - y(t)\big) = k_p e(t). \qquad (10.6)$$

Структура П-регулятора показано на рис. 10.42.

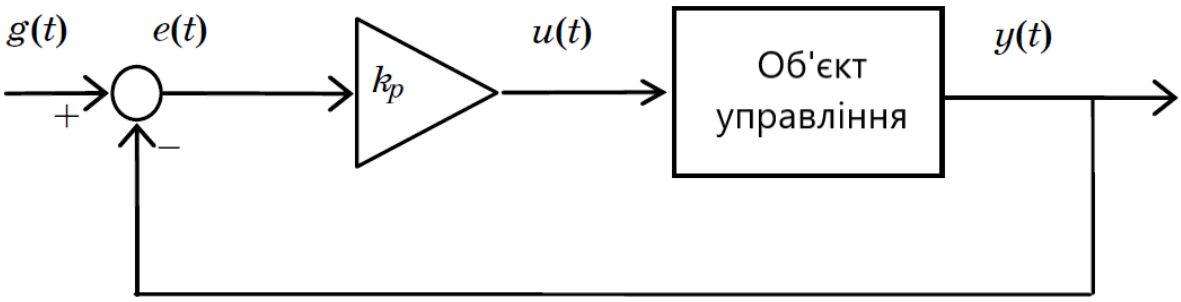


Рисунок 10.42 – Управління, пропорційне похибки

Розрахувати значення коефіцієнта k_p можна, використовуючи метод кореневого годографа або за допомогою оптимізаційного пошуку, що мінімізує задану помилку

Графічно закон пропорційного управління є пряму, яка проходить через початок координат під кутом α. Так що

$$k_p = tg\alpha.$$

У реальній системі завжди існують фізичні обмеження значення сигналу управління, отже пропорційний закон управління має вигляд, показаний на рис. 10.43.

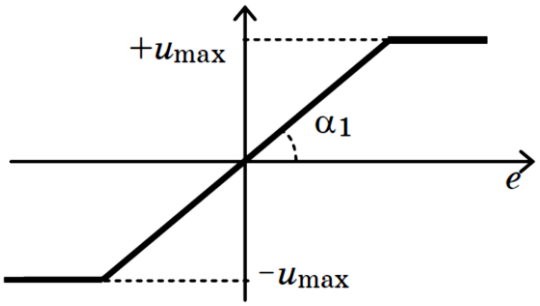


Рисунок 10.43 – Графік пропорційного закону управління

Як відомо, найбільша швидкодія забезпечується граничним варіантом П-регулятора – регулятором релейного типу, який працює за ознакою помилки:

$$u = Ksign(e).$$

Однак їх використання викликає велике перерегулювання.

Якщо нормалізувати (привести до діапазону [-1, 1]) вхід та вихід П-регулятора, то пропорційний закон управління набуває вигляду, показаного на рис. 10.44.

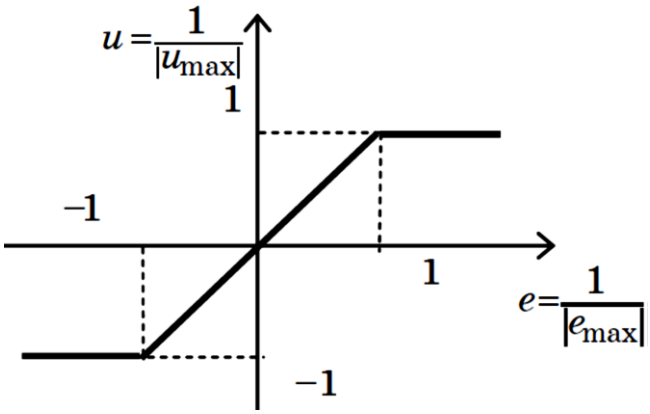


Рисунок 10.44 – Нормування вхід-вихідних змінних регулятора

Відповідно до (3.6) та рис. 10.44 можна записати:

$$u=e. \qquad (10.7)$$

Пропорційне управління може викликати конфлікт між вимогами до перехідного процесу: велике значення k_p зменшує час наростання, але одночасно збільшує перерегулювання. Через це принцип управління помилково часто не дозволяє отримати прийнятне рішення. Наприклад, нехай об'єкт управління описується за допомогою передаточної функції

$$W(p)=\frac{50}{p^2+20p+50}$$

На рис. 10.45 показано схему моделювання системи з П-регулятором.

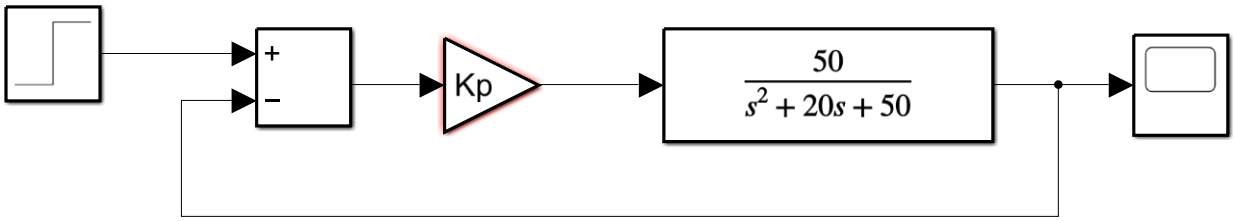


Рисунок 10.45 – Структурна схема П-регулятора в Simulink MatLab

На рис. 10.46 показані перехідні процеси при k_p=5 та k_p =30. У першому випадку (при маленькому коефіцієнті посилення) виходить занадто велика статична помилка (≈16%). При великому коефіцієнті посилення статична помилка зменшується до 4%, але виходить неприйнятно велике перерегулювання (близько 40%).

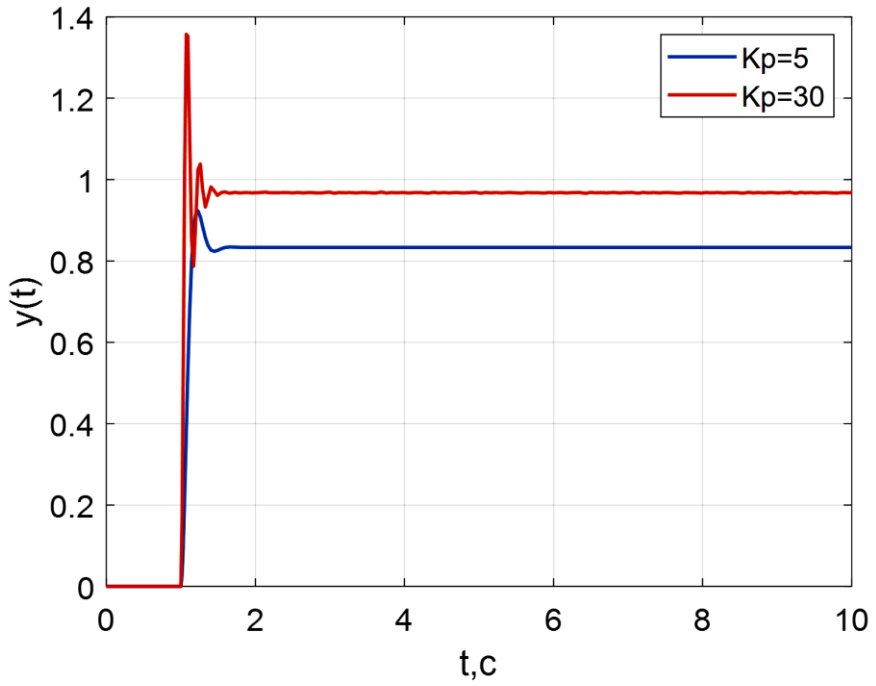


Рисунок 10.46 – Графік перехідного процесу при різних коефіцієнтах підсилення

Нечіткий логічний регулятор П-типу (НЛР_П) відрізняється від класичного П-регулятора тим, що може реалізовувати нелінійний закон управління виду

$$u(t) = F_{л}\big(e(t)\big),$$

де $F_л$ - функція, що управляє, задана набором лінгвістичних правил.

Нехай базові шкали лінгвістичної змінної «помилка» та лінгвістичної змінної «сигнал управління» нормалізовані. Допустимо також, що кожна лінгвістична змінна має по 3 терми з назвами: «негативний», «нульовий», «позитивний» (скорочено Н, 0 і П, рис. 10.47).

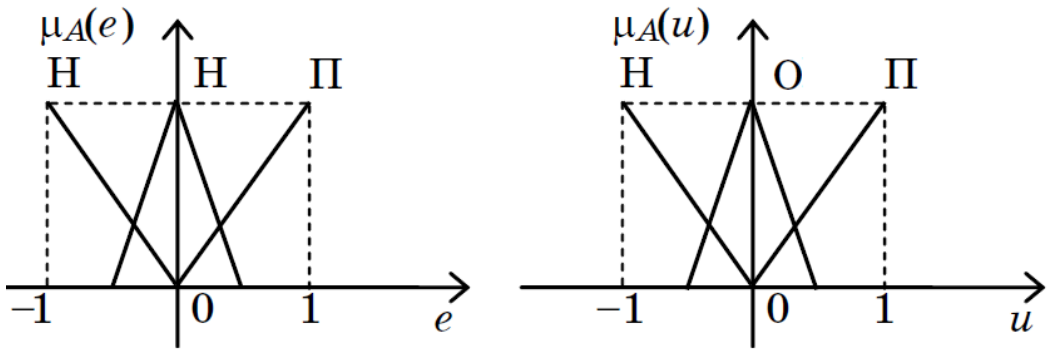


Рисунок 10.47 – Лінгвістичний опис входу та виходу НЛР_П

Тоді, відповідно (10.7), нечіткий закон управління може бути описаний за допомогою правил:

$$\text{Якщо } e = \text{"Н"}, \text{то } u = \text{"Н"},$$
$$\text{Якщо } e = \text{"0"}, \text{то } u = \text{"0"},$$
$$\text{Якщо } e = \text{"П"}, \text{то } u = \text{"П"}.$$

Цей евристичний закон управління універсальний. Для конкретного об'єкта змінюються лише коефіцієнти нормалізації та денормалізації.

Якщо вибрати більше термів для вхідної та вихідної змінної, то автоматично збільшується кількість керуючих правил.

Крім кількості термів, якість управління значною мірою може залежати від форми термів лінгвістичної змінної та від їх розташування на базовій шкалі.

Слід відмітити, що за певних умов НЛР_П може бути еквівалентний лінійному П-регулятору, але завдання синтезу нелінійного НЛР_П полягає у виконанні вимог, з якими лінійний регулятор не справляється (див. рис. 10.46).

Для того, щоб виявити переваги НЛР_П стосовно лінійного П-регулятора, розглянемо умови еквівалентності цих двох регуляторів.

10.7 Умови лінійності нечіткого регулятора П-типу

Нехай описи термів нормалізованих вхідний та вихідний змінних НЛР_П обрані у вигляді, показаному на рис. 10.48 та 10.49.

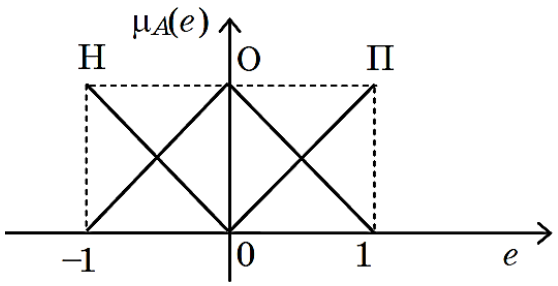


Рисунок 10.48 – Опис вхідної змінної регулятора - похибки

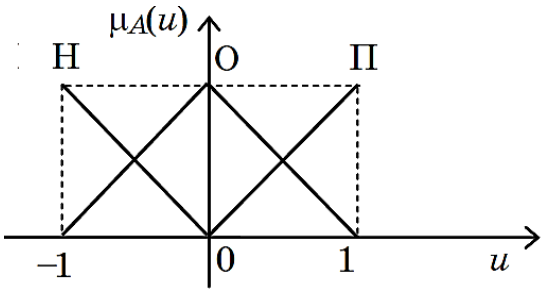


Рисунок 10.49 – Опис виходу регулятора – сигнал управління

Зауважимо, що при цьому забезпечується нечітке розбиття базової множини:

− вхідне значення належить не більше ніж двом термам;

− сума ступенів належності у кожній точці дорівнює 1.

Тоді, використовуючи для дефазифікації дискретний метод центру тяжіння, для будь-якого значення помилки праворуч від нуля маємо таке значення сигналу управління:

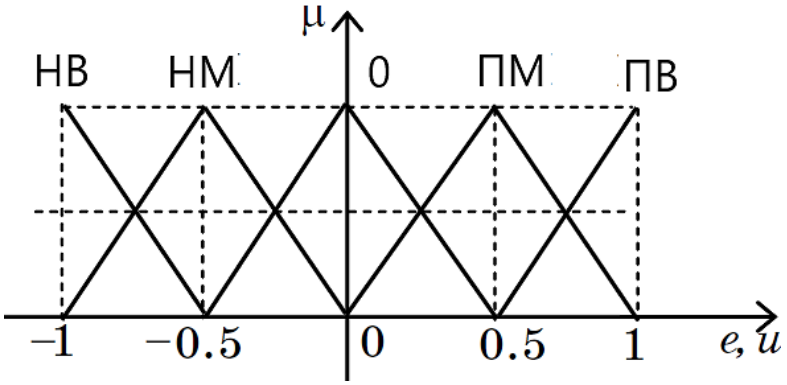
$$u^+ = \frac{\sum_{i=1}^{2} \mu_i(e) u_i}{\sum_{i=1}^{2} \mu_i(e)} = \frac{\mu_0(e) \cdot 0 + \mu_{\text{П}}(e) \cdot 1}{1} = \mu_{\text{П}}(e) = e.$$

Аналогічно для будь-якого значення помилки зліва від нуля:

$$u^- = \frac{\sum_{i=1}^{2} \mu_i(e) u_i}{\sum_{i=1}^{2} \mu_i(e)} = \frac{\mu_0(e) \cdot 0 + \mu_{\text{Н}}(e) \cdot 1}{1} = \mu_{\text{Н}}(e) = e.$$

Таким чином, у будь-якому випадку виконується умова (10.7), і НЛР_П еквівалентний лінійному П-регулятору. Відповідно, використання НЛР_П не дає жодних переваг по відношенню до П-регулятора.

Властивість лінійності може виконуватися за будь-якої кількості термів. Наприклад, нехай помилка керування *e* та сигнал керування *u* описуються за допомогою 5 термів (рис. 10.50).



НВ – негативне велике; НМ – негативне мале; 0 – нульове; ПМ – позитивно мале; ПВ – позитивно велике

Рисунок 10.50 – Опис входу та виходу регулятора за допомогою п'яти термів

Сигнал управління у правій напівплощині виходить за формулою

$$u^+ = \frac{\sum_{i=1}^{3} \mu_i(e) u_i}{\sum_{i=1}^{2} \mu_i(e)} = \frac{\mu_0(e) \cdot 0 + \mu_{\text{ПМ}}(e) \cdot 0{,}5 + \mu_{\text{ПВ}}(e) \cdot 1}{1}$$

В інтервалі $0 \le e \le 0{,}5$ маємо $\mu_{\text{ПМ}} = 2e$, тобто

$$u^+ = \mu_{\text{ПМ}}(e) \cdot 0{,}5 + \mu_{\text{ПВ}}(e) \cdot 1 = (1 - 2e) \cdot 0.5 + 2e - 1 = e.$$

Аналогічно можна розглянути ліву напівплощину, тому у всіх точках вхідного простору регулятора виконується (10.7). Таким чином,

можна констатувати, що зі збільшенням числа термів (і числа правил) закон управління залишається лінійним, якщо терми мають трикутну форму і утворюють нечітке розбиття базової множини, а для дефазифікації використовується дискретний метод центру тяжіння.

Як було показано попередньому розділі, для такого лінійного НЛР_П допускається запис процедури нечіткого виведення в матричній формі.

10.8 Нелінійна поведінка нечіткого регулятора П-типу

Розглянемо далі умови, у яких поведінка НЛР_П стає нелінійним. Цього можна досягти шляхом зміни опису термів вхідної змінної. Наприклад, можна вибрати опис, показаний на рис. 10.51 де терми «згущуються» в області нуля.

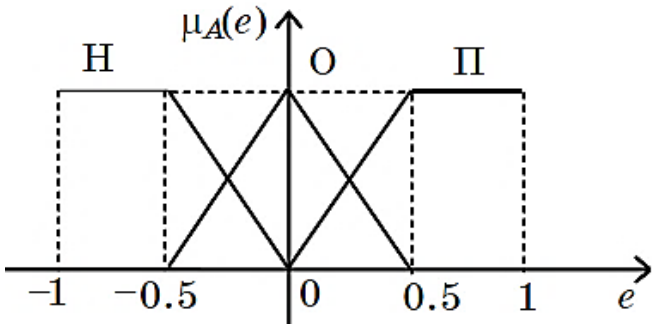


Рисунок 10.51 – Масштабування термів вхідної змінної

Оскільки опис термів керування не змінився, у лінійній зоні (–0.5 ≤ *e* ≤ 0.5) виконується:

$$u^{+} = \mu_{\text{П}}(e) = 2e \text{ та } u^{-} = \mu_{\text{Н}}(e) = 2e.$$

При умові $0{,}5 \leq |e| \leq 1$ виконується *u*=1.

Отже, нелінійний закон управління описується графіком, показаним на рис. 10.52.

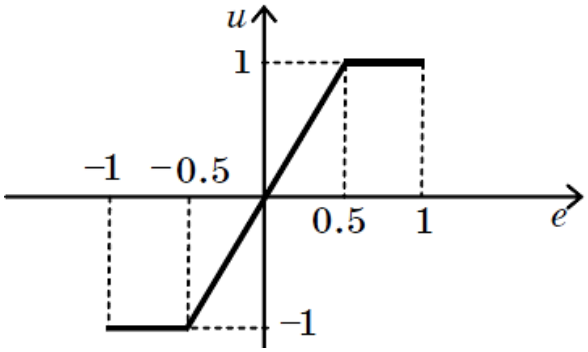


Рисунок 10.52 – Зміна нахилу лінії управління

Перевага НЛР_П по відношенню до звичайного П-регулятора полягає в тому, що коефіцієнт посилення може змінюватися залежно від поточної помилки управління.

Наприклад, нехай потрібно, щоб при малих значеннях помилки (до 0.3) був невеликий коефіцієнт посилення, а при великих значеннях помилки коефіцієнт посилення збільшувався. Щоб забезпечити бажану нелінійну поведінку регулятора, можна залишити опис помилки таким, як показано на рис. 10.50, а опис сигналу управління слід змінити, згустивши терми в області малих значень і розрядивши в області великих значень помилки. Приклад показано на рис. 10.53.

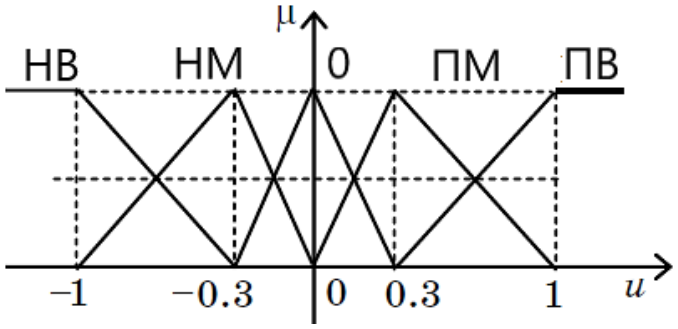


Рисунок 10.53 - Згущення термів в області малих значень помилки

У цьому випадку у правій напівплощині

$$u^+ = \frac{\sum_{i=1}^{3} \mu_i(e) u_i}{\sum_{i=1}^{2} \mu_i(e)} = \frac{\mu_0(e) \cdot 0 + \mu_{\text{ПМ}}(e) \cdot 0{,}3 + \mu_{\text{ПВ}}(e) \cdot 1}{1}$$

До точки e = 0.5 (див. рис. 10.50): $\mu_{ПМ} = 2e$ та $\mu_{ПВ} = 0$, отже:

$$u^+ = \mu_{\text{ПМ}}(e) \cdot 0{,}3 + \mu_{\text{ПВ}}(e) \cdot 1 = (1 - 2e) \cdot 0.3 + 2e - 1 = 1{,}4e - 0{,}4.$$

Аналогічно можна розглянути ліву напівплощину. На рис. 10.54 показаний підсумковий графік $u(e)$ побудований за програмою:

```
% Задаються діапазони помилки
e1=0:0.1:0.5;
e2=0.5:0.1:1;
e3=-0.5:0.1:0;
e4=-1:0.1:-0.5;

% Задаються діапазони управління
u1=0.6*e1;
u2=1.4*e2-0.4;
u3=0.6*e3;
u4=1.4*e4+0.4;
```

```
% Побудова графіка управління
plot(e1,u1,e2,u2,e3,u3,e4,u4);
grid;
xlabel('e');
ylabel('u');
```

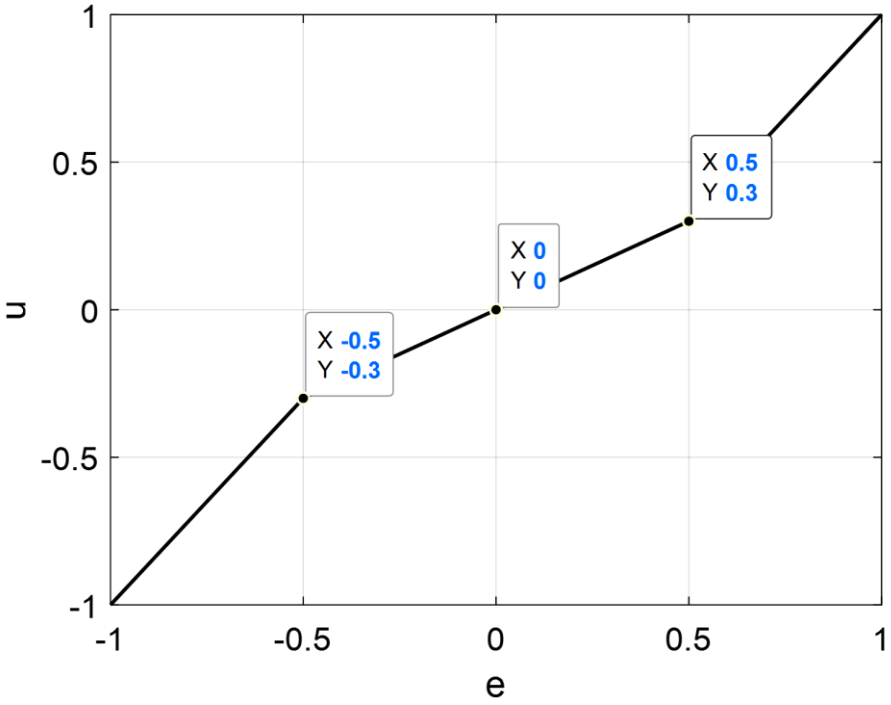


Рисунок 10.54 – Нелінійний закон управління НЛР

Отже, нелінійні ефекти у роботі НЛР_П виникають при нерівномірному розподілі термів лінгвістичних змінних.

За рахунок змінного коефіцієнта посилення НЛР_П можна зі-

голосувати суперечливі вимоги, з якими неспроможна впоратися класичний П-регулятор.

Рисунок 10.54 показує, що закон управління НЛР_П можна синтезувати поетапно, розглядаючи лінійні окремі ділянки. На останньому етапі локальні стратегії управління поєднуються, утворюючи нелінійний закон управління.

10.9 Умови еквівалентності НЛР та П-регулятора

Сформулюємо умови лінійної поведінки НЛР_П:

1) використовуються трикутні функції приналежності;

2) терми утворюють нечітке розбиття відповідних базових множин;

3) для дефазифікації використовується дискретний метод центру тяжкості.

При цих припущеннях кожне значення e має ненульове значення приналежності до двох сусідніх терм T_i та T_{i+1} з центрами e_i та e_{i+1} (див.

рис. 10.55).

Значення приналежності для точки $e_i < e < e_{i+1}$ розраховуються за формулами:

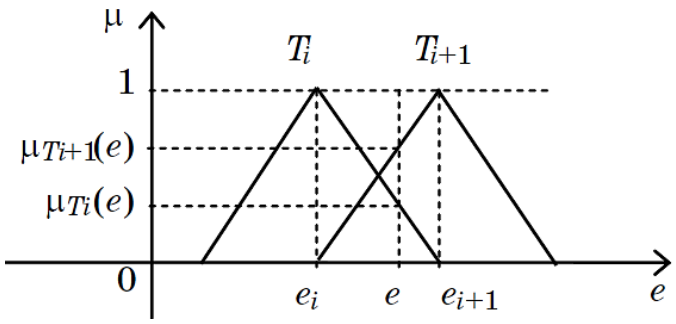


Рисунок 10.55 – Перехідні процеси при різних коефіцієнтах підсилення

$$\mu_{T_i}(e) = \frac{e_{i+1} - e}{e_{i+1} - e_i}, \mu_{T_{i+1}}(e) = \frac{e - e}{e_{i+1} - e_i}.$$

У такій ситуації спрацьовують два правила:

R_1: Якщо $e = T_i(e)$, то $u = T_i(u)$,
R_2: Якщо $e = T_{i+1}(e)$, $u = T_{i+1}(u)$.

При використанні дискретного методу центру тяжіння має значення тільки u_i − центральна точка терму $T_i(u)$ і результат дефазифікації описується формулою:

$$u = \frac{\mu_{T_i}(e)u_i + \mu_{T_{i+1}}(e)u_i}{\mu_{T_i}(e) + \mu_{T_{i+1}}(e)} = \mu_{T_i}(e)u_i + \mu_{T_{i+1}}(e)u_{i+1=}$$
$$= \frac{e_{i+1} - e}{e_{i+1} - e_i}u_i + \frac{e - e}{e_{i+1} - e_i}u_{i+1}.$$

Цей вираз можна переписати у вигляді

$$u = \frac{u_{i+1} - u_i}{e_{i+1} - e_i}e + \frac{e_{i+1}u_i - e_iu_{i+1}}{e_{i+1} - e_i} = Ke + C, \qquad (10.8)$$

де K – коефіцієнт підсилення; C - зміщення

Зауважимо, що (10.8) стосується лише постійного діапазону між e_i та e_{i+1}. Якщо відстань між термами змінюється, змінюватимуться і константи K і C.

Формула (10.8) показує, що для еквівалентності поведінки НЛР_П і звичайного П-регулятора необхідно виконання двох умов:

1) значення K має бути однаковим у всьому діапазоні зміни помилки e:

$$\frac{u_{i+1}-u_i}{e_{i+1}-e_i}=K, \qquad \forall i \in \{1, N-1\} \tag{10.9}$$

2) значення зсуву С повинно бути нульовим, що виконується при:

$$\frac{e_{i+1}}{e_{i+1}}=\frac{u_{i+1}}{u_i}, \quad \forall i \in \{1, N-1\} \tag{10.10}$$

де N – кількість термів.

Таким чином, НЛР _П, для якого справедливо (10.9) та (10.10), описується формулою

$$u = Ke.$$

Оскільки для лінійного НЛР_П кількість термів не має значення, розглянемо регулятор із трьома правилами.

У разі використання дискретного методу центру тяжкості вихід регулятора можна описати з допомогою рис. 10.56.

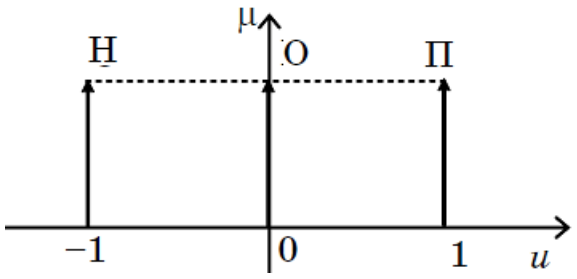


Рисунок 10.56 – Опис виходу регулятора з трьома правилами

Відповідно до рис. 3.46 виконується умова:

$$u_{i+1} - u_i = 1.$$

Нехай базову шкалу вхідної змінної описано відповідно до рис. 10.57.

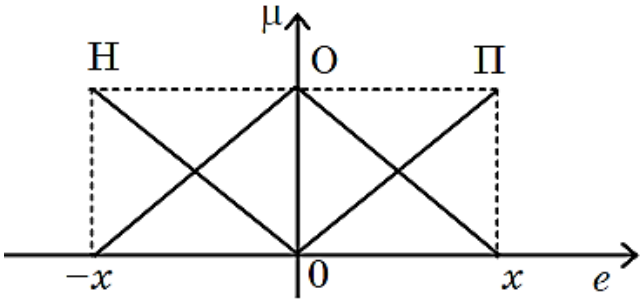


Рисунок 10.57 – Опис входу регулятора з трьома правилами

Тоді для вхідної змінної відстань між центрами термів:

$$e_{i+1} - e_i = x, \quad x \in [0,1]$$

Тоді

$$K = \frac{\Delta u}{\Delta e} = \frac{1}{x}. \qquad (10.11)$$

і що менше x, то більше вписувалося коефіцієнт посилення. Закон управління можна описати з допомогою табл. 10.5.

Таблиця 10.5 - Закон управління НЛР_П із трьома правилами

	Н	0	П
e	-*x*	0	*x*
u	-1	0	1

Вибір параметра x визначає нахил кривої, що описує закон керування (рис. 10.58). Чим менше значення x, тим ближче виявляється закон керування до релейного (при x → 0 регулятор працює за знаком помилки).

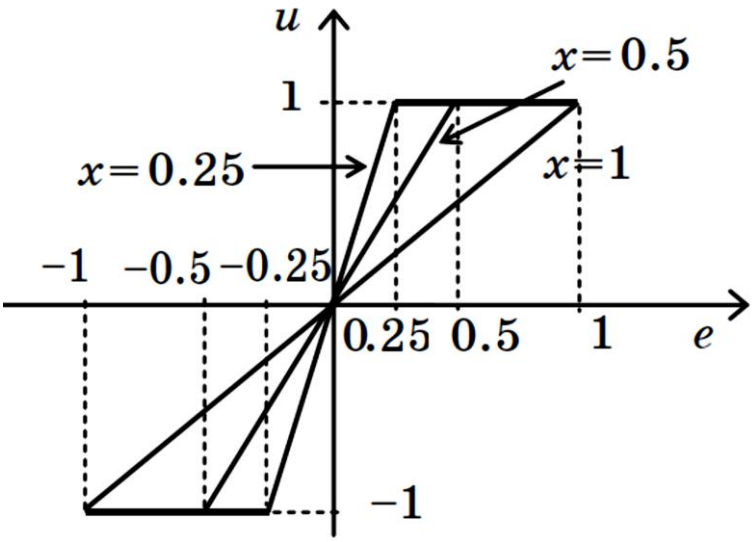


Рисунок 10.58 – Варіанти закону управління для НЛР_П з трьома правилами

Як показав приклад на рис. 10.46 коефіцієнт посилення повинен бути великим наприкінці перехідного процесу – для зменшення статичної помилки. На початку перехідного процесу потрібно мати менший коефіцієнт посилення, щоб уникнути надмірного перерегулювання.

10.10 Синтез нелінійного нечіткого регулятора П-типу

Формула (10.11) дозволяє легко налаштовувати значення змінного коефіцієнта посилення для різних ділянок базової шкали вхідної змінної.
Наприклад, розглянемо НЛР_П із п'ятьма термами, де дві пари термів розташовані симетрично щодо нуля (рис. 10.59).

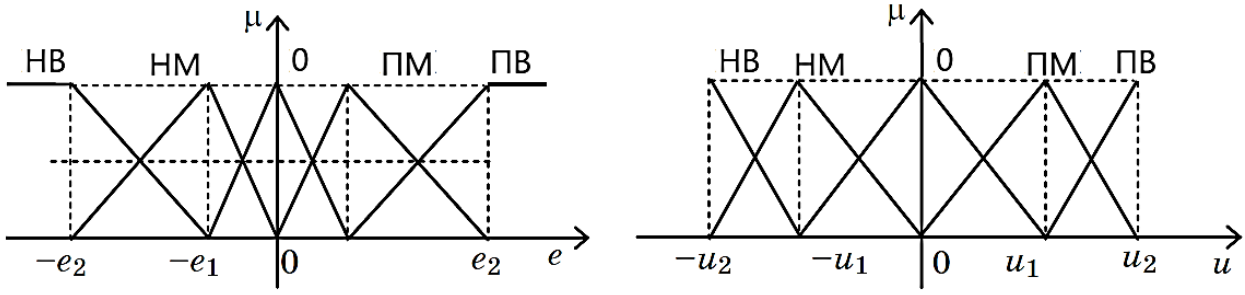


Рисунок 10.59 – Варіант опису термів НЛР_П з п'ятьма правилами

На підставі (10.7), закон управління визначає табл. 10.6.

Таблиця 10.6 – Типовий закон управління НЛР_П із п'ятьма правилами

	НВ	НМ	0	ПМ	ПВ
e	$-e_2$	$-e_1$	0	e_1	e_2
u	$-u_2$	$-u_1$	0	u_1	u_2

На ділянці між центрами термів 0 і НМ, а також 0 і ПМ коефіцієнт посилення

$$K_1 = \frac{u_1}{e_1} = -\frac{u_1}{e_1}.$$

На ділянках між центрами термів ПРО та ОМ, а також ПМ та ПВ коефіцієнт посилення

$$K_2 = \frac{u_2 - u_1}{e_2 - e_1} = \frac{-u_2 + u_1}{-e_2 + e_1}.$$

Наприклад, нехай вибрано значення центрів термів, показані в табл. 10.7.
Тоді

$$K_1 = \frac{0{,}3}{0{,}5} = 0{,}6; \quad K_2 = \frac{1 - 0{,}3}{1 - 0{,}5} = 1{,}4.$$

Таблиця 10.7 - Варіант закону управління НЛР_П із п'ятьма правилами

	НВ	НМ	0	ПМ	ПВ
e	-1	-0.5	0	0.5	1
u	-1	-0.3	0	0.3	1

Графічне уявлення закону управління показано на рис. 10.60.

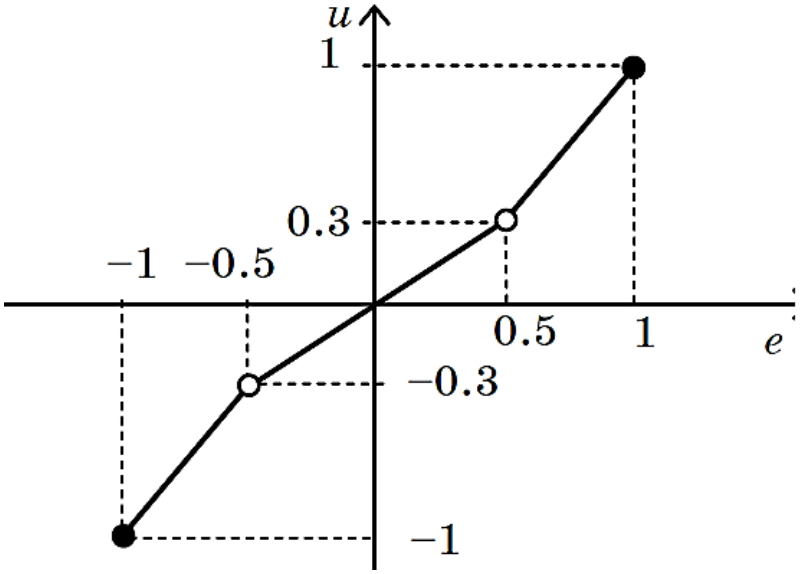


Рисунок 10.60 – Закон управління для регулятора з п’ятьома правилами

Як свідчить рис. 10.60, конфігурація кривої управління залежить від положення точок перегину (білі кружки), координати яких відповідають центрам термів НМ та ПМ вхідний та вихідний змінної НЛР_П (див. табл. 10.7). Оскільки ці центри симетричні, процес конструювання НЛР_П з п'ятьма правилами зводиться до вибору двох параметрів.

Якщо розглядати сім термів, то вибору підлягають 4 параметри, якщо дев'ять - то 6 і т. д. Таким чином, при N термах кількість параметрів, що настроюються $K = N - 3$.

У загальному випадку кількість термів (і, відповідно, правил) має бути такою, щоб задовольнити всі поставлені вимоги.

Розглянемо задачу конструювання нелінійного НЛР_П зі змінним коефіцієнтом посилення K_f як завдання корекції лінійного П-регулятора, що описується коефіцієнтом k_p. Тут можливі два варіанти корекції (рис. 10.61 та рис. 10.62).

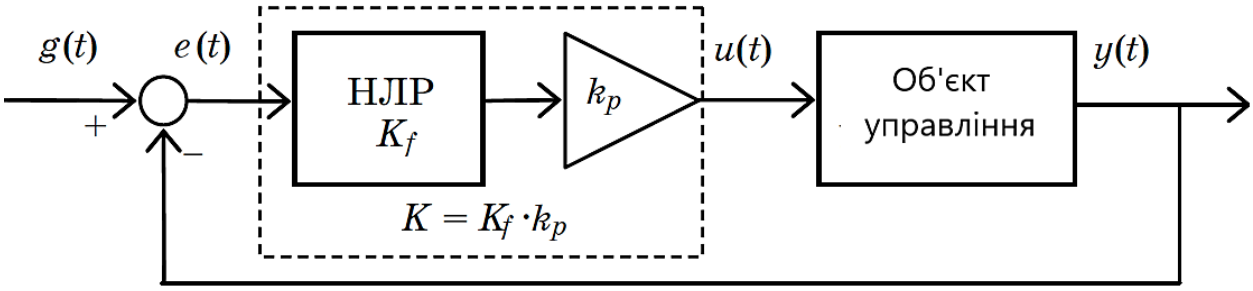


Рисунок 10.61 – Послідовна корекція П-регулятора

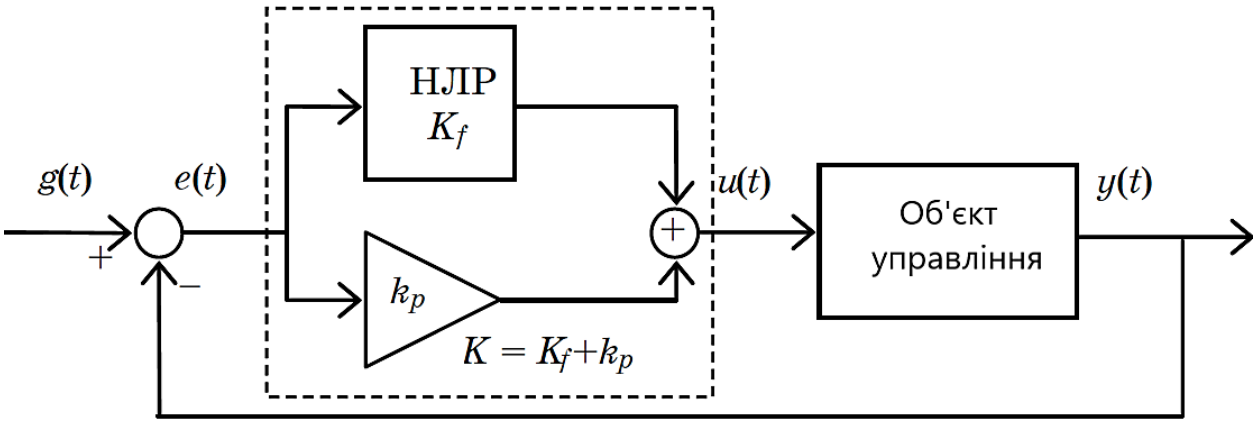


Рисунок 10.62 – Паралельна корекція П-регулятора

У цих структурах k_p визначає базовий коефіцієнт посилення. Постає питання – яким слід вибрати k_p? Оскільки НЛР_П має забезпечувати додаткове посилення, можна зажадати, щоб базовий П-регулятор забезпечував умову y_{max}=1 (див. рис. 10.42). Тим самим гарантується, що помилка прийматиме всі значення в межах шкали [0, 1]. Інші параметри перехідного процесу можуть бути покращені за рахунок використання НЛР_П.

Таким чином, перший крок синтезу полягає у налаштуванні такого k_p, при якому забезпечується нульове перерегулювання (K_f = 1 (див. рис. 10.61) або K_f = 0 (див. рис. 10.62)).

Базовий коефіцієнт посилення є дійсним числом, яке може набувати широкого діапазону значень залежно від об'єкта управління. Таким чином, можна або давати збільшення базовому коефіцієнту (див. рис. 10.62), або масштабувати його (див. рис. 10.61).

Далі розглядатимемо варіант послідовної корекції (див. рис.10.61) при використанні НЛР _П з сімома правилами.

Відповідно до (10.11) маємо

$$K_f = \frac{\Delta u}{\Delta e},$$

таким чином для отримання бажаного коефіцієнта посилення не

обов'язково міняти обидва коефіцієнти – можна міняти тільки Δ*u* або Δ*e*, проте |Δ*u*| ≤ 1, тому далі оперуватимемо величиною Δ*e*.
Припустимо, що вихідні терми змінної НЛР_П рівномірно розподілені за базовою шкалою (див. рис. 10.63), тобто |Δ*u*|= 0,33.

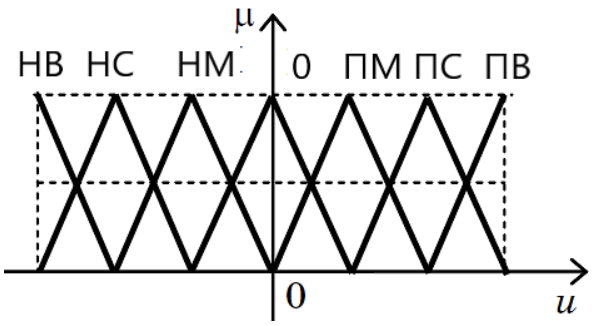


Рисунок 10.63 – Терми вихідної змінної НЛР_П із сімома правилами

Якщо терми вхідної змінної також рівномірно розташувати за базовою шкалою, то K_f = 1 будь-якого значення помилки (рис. 10.64).

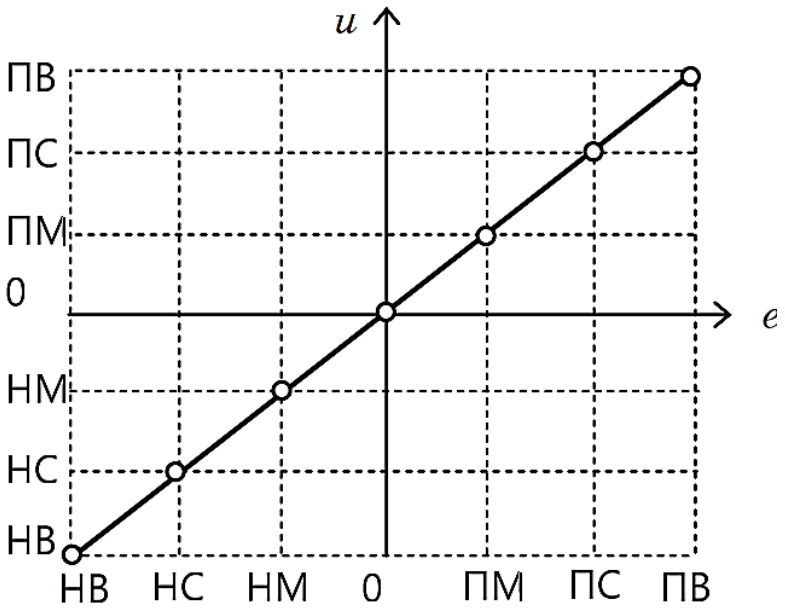


Рисунок 10.64 - Управляюча крива при рівномірному розподілі термів

На рис. 10.64 білими кружками позначені точки, у яких спрацьовує лише одне правило. Ці точки перебувають на перетині ліній, проведених із центрів однойменних термів. При переміщенні *e* між центрами термів керування лінійно змінюється. Таким чином, при семи керуючих правилах виходить п'ять лінійних ділянок закону управління.

Змінивши розташування центрів проміжних термів на базовій шкалі вхідної змінної можна отримати нелінійний закон управління (див. рис. 10.65).

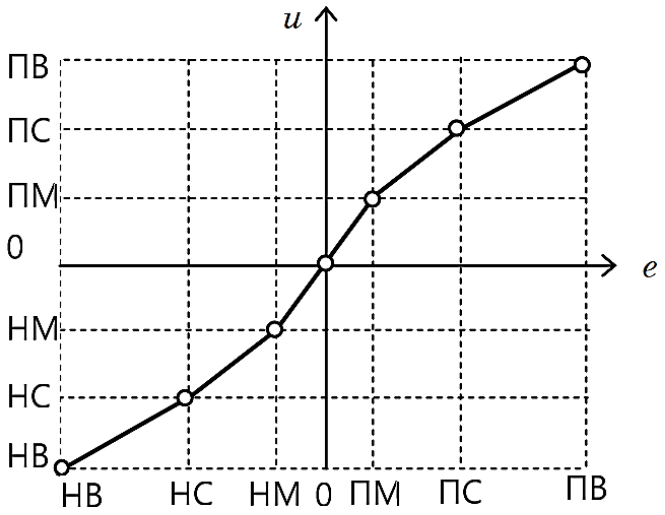


Рисунок 10.65 - Управляюча крива при нерівномірному розподілі термів

Центри термів НВ та ПВ для вхідної та вихідної змінної знаходяться у крайніх становища – у точках –1 та +1.

Розглянемо завдання вибору положення центру ПС (НС). За визначенням помилка *e* має в околиці цього терму середнє значення, тому вибір цього терму впливає на час наростання перехідного процесу: чим ближче ПС до ПВ, тим меншим буде час наростання, але при цьому можливе зростання перерегулювання.

При виборі положення центру ПМ (НМ) слід враховувати, що в околиці цього терма помилка має мале значення, отже, від вибору центру цього терму залежить значення помилки, що встановилася: чим менше Δ*e* (тобто чим ближче ПМ і 0), тим більше K_f, і тим менше встановилася помилка.

Узагальнюючи всі вищенаведені міркування, можна остаточно сформулювати алгоритм синтезу нелінійного НЛР_П у межах структури, показаної на рис. 10.61.

1. Терми вхідних і вихідних змінних розподіляються рівномірно за базовими шкалами, отже K_f = 1. Вибирається значення k_p, у якому виконується умова y_{max}=1.

2. Розглядається регулятор із трьома правилами (див. рис. 10.56, 10.57). Вибирається таке значення x_1 (див. рис. 10.57), за якого забезпечується мінімальний час наростання при максимально допустимому перерегулюванні.

3. Знову розглядається регулятор із трьома правилами. Вибирається таке значення x_2, при якому забезпечується мінімальна помилка, що встановилася.

4. Розглядаються графіки трьох законів управління. Терми вихідної змінної регулятора можна рівномірно розташувати за відповідною базовою шкалою (див. рис. 10.63).

Для визначення положення центрів термів ПМ та ПС по осі *e*

розглядаються точки перетину ліній, проведених із центрів термів ПМ та ПС по осі *u* та прямих, що описують закони управління x_1 та x_2 (див. рис. 10.66). Координати точок перетину по осі *e* дають положення центрів термів ПМ та ПС для вхідної змінної регулятора.

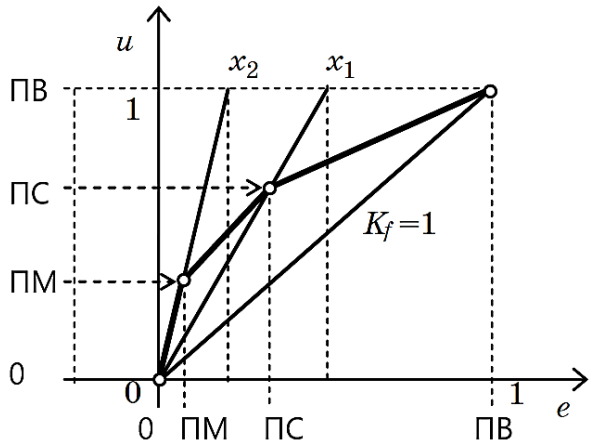


Рисунок 10.66 – Визначення положення термів НЛР_П

Очевидні геометричні розрахунки дозволяють описати закон управління як табл.10.8.

Таблиця 10.8 – Закон управління НЛР_П

	НВ (негативне велике)	НС (негативне середнє)	НМ (негативне мале)	Н (нульове)	ПМ (позитивне мале)	ПС (позитивне середнє)	ПВ (позитивне велике)
e	-1	$-0{,}66x_2$	$-0{,}33x_1$	0	$0{,}33x_1$	$0{,}66x_2$	1
u	-1	-0,66	0,33	0	0,33	0,66	1

Ширина всіх термів (включаючи терм 0) вибирається таким чином, щоб забезпечити нечітке розбиття базової шкали вхідної змінної (див. рис. 10.67).

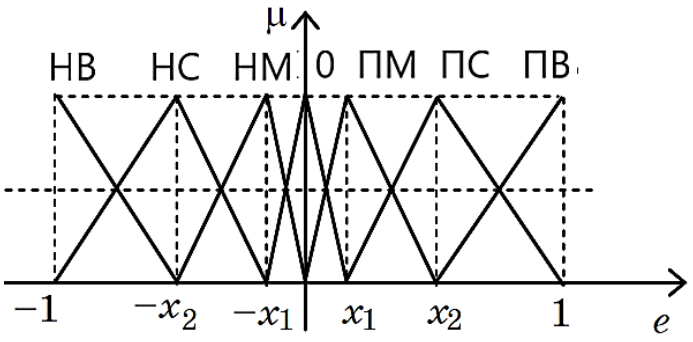


Рисунок 10.67 – Результати налаштування нечіткого регулятора П-типу

Нечіткий регулятор П-типу є основою розробки НЛР_ПД, НЛР_ПІ та НЛР_ПІД, оскільки використання у законі управління похідної помилки може зменшити перерегулювання, а використання інтеграла помилки – зменшити встановлену помилку.

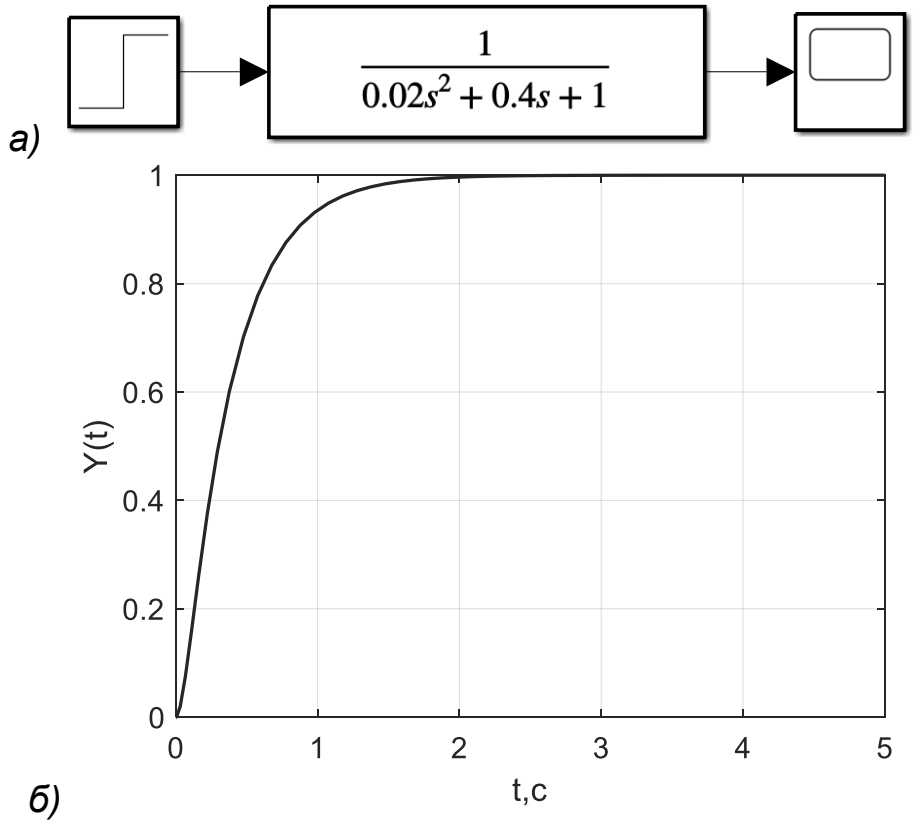
Таким чином, розробка НЛР_П є першим кроком при проектуванні складніших нечітких регуляторів.

10.11 Приклад синтезу нечіткого регулятора П-типу

У Simulink MatLab контейнером для системи нечіткого логічного висновку є блок Fuzzy Logic Controller (пакет Fuzzy Logic ToolBox). Синтезуємо нелінійний НЛР_П відповідно до методики, викладеної в п. 10.10, для об'єкта, що описується передаточною функцією:

$$W(p) = \frac{1}{0.02p^2 + 0.4p + 1}.$$

Цей об'єкт є стійким, але час перехідного процесу дуже великий (див. рис. 10.68).



а) математична модель; б) графік перехідного процесу

Рисунок 10.68 – Математичне моделювання об’єкта управління

Потрібно забезпечити перехідний процес з малим часом наростання, що встановилася помилкою Δ ≤ 3% та перерегулюванням d ≤ 10%.

Використовуватимемо описаний вище 3-кроковий алгоритм проектування.

1. Розглянемо НЛР_П з із сімома правилами, що має одиничний коефіцієнт посилення (див. рис. 10.63 та 10.67). Такий регулятор не впливає процес управління. Виберемо базовий коефіцієнт посилення k_p=6.5, при цьому забезпечується умова y_{max} = 1 (див. рис. 10.69 та 10.70).

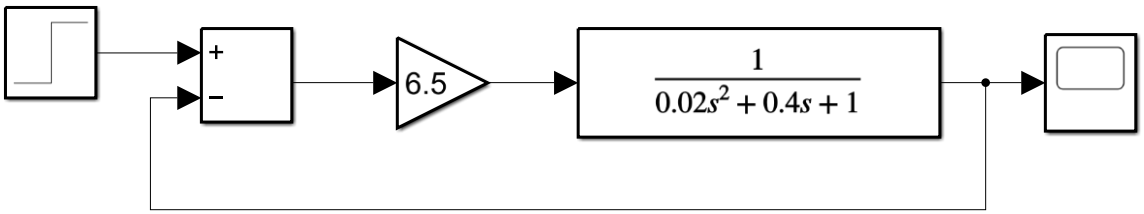


Рисунок 10.69 – Математична модель САУ з базовим П-регулятором

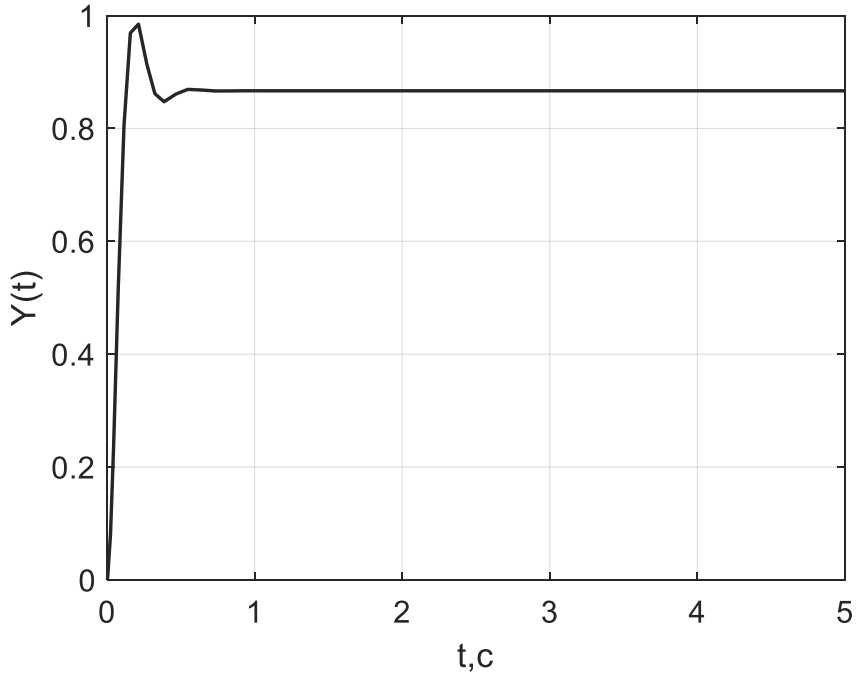


Рисунок 10.70 – Графік перехідного процесу при базовому коефіцієнте підсилення

Таким ‘чином, центрами термів НВ та ПВ будуть точки –1 та 1.

2. Розглянемо НЛР_П із сімома правилами (рис. 10.71).

Блок Saturation використовується для приведення вхідної змінної до заданого діапазону [-1;1]. Терми вихідний змінної регулятора розташовуються за базовою шкалою відповідно до рис. 10.63, терми

вхідної змінної – відповідно до рис. 10.67. Методом спроб та помилок вибирається значення x_1 = 0.5 (рис. 1.67), при цьому забезпечується мінімальний час наростання при заданому перерегулюванні.

Центрами термів ОС та ПС будуть точки –0.5 та 0.5.

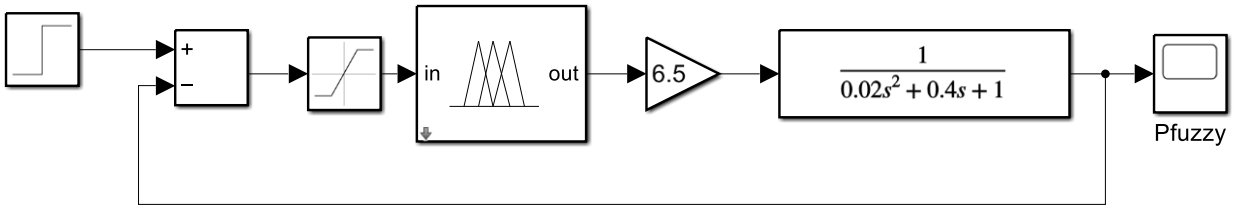


Рисунок 10.71 – Математична модель системи управління з нечітким регулятором

3. На третьому кроці знову розглядається НЛР_П із сім'ю правилами. Вибирається коефіцієнт посилення регулятора x_2 такий, щоб Δ<3%.

Центрами термів НМ та ПМ будуть точки –0.1 та 0.1.

4. На останньому етапі необхідно отримати нелінійний закон управління НЛР_П. Центри термів вхідної (*e*) вихідний змінної (*u*) розраховуються відповідно до табл. 10.8. У таблиці 10.9 наведено закон управління проектованим НЛР_П.

Таблиця 10.9 – Закон управління проектованим НЛР_П

	НВ (негативне велике)	НС (негативне середнє)	НМ (негативне мале)	Н (нульове)	ПМ (позитивне мале)	ПС (позитивне середнє)	ПВ (позитивне велике)
e	-1	-0,33	-0,033	0	0,033	0,33	1
u	-1	-0,66	0,33	0	0,33	0,66	1

Вирішимо дане завдання згідно описаного алгоритму з використанням редактора системи нечіткого висновку.

Будуємо у MatLab Simulink математичну модель об’єкту керування з послідовним включення Fuzzy Logic Controller до класичного регулятора (див. рис. 10.71). Fuzzy Logic Controller надаємо Fis name: Pfis7 (ім’я може буди будь-яке).

Викликається редактор системи нечіткого висновку (Fuzzy Inference System – FIS) у MatLab командою

```
>> fuzzy
```

Далі проводиться налаштовування головного вікна редактора нечіткої логічної системи (див. рис. 10.72.).

У *FIS editor* задається опис систему нечіткого логічного висновку *Mamdani*. Для створюваної системи обирається вид логічного зв'язку (*And method – min*) та (*Or method – max*), вид імплікації (*Implication – min*), спосіб агрегування висновків правил (*Aggregation – max*) та метод дефазифікації (*Defuzzification – centroid*).

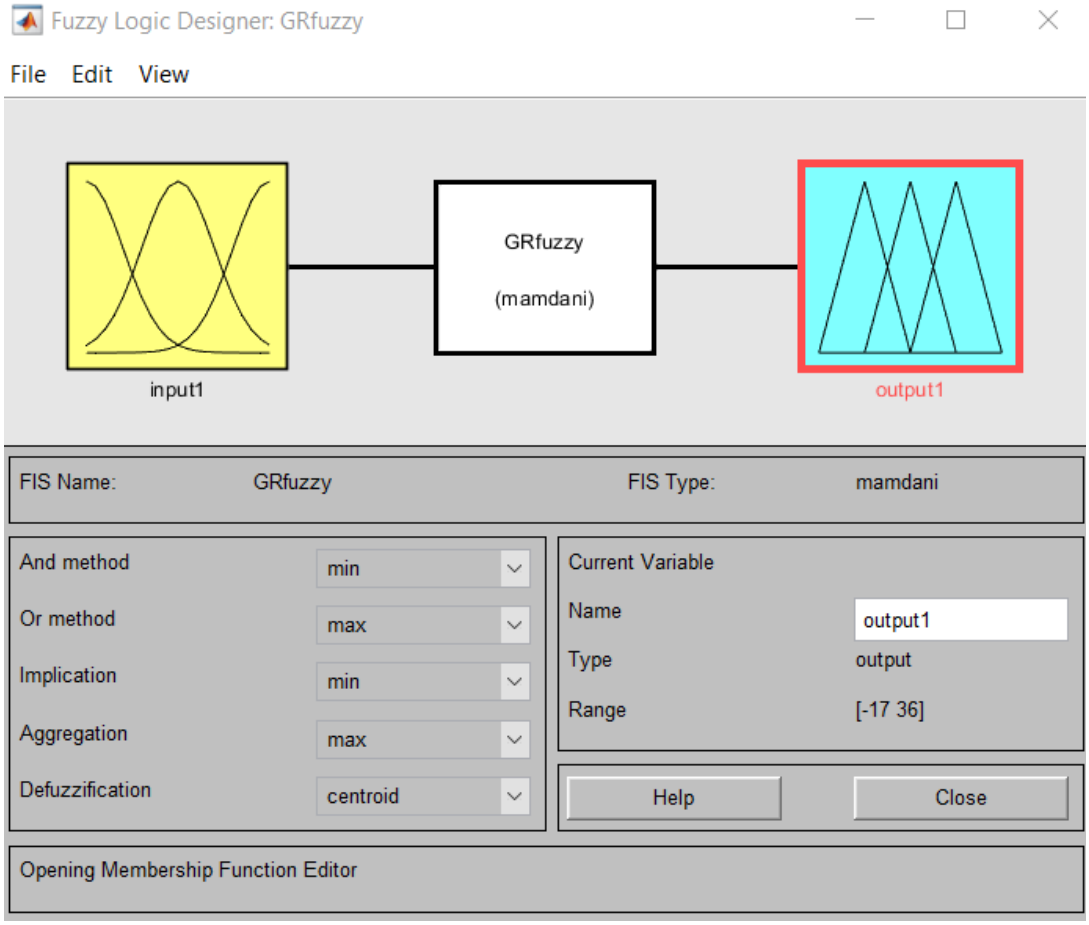


Рисунок 10.72 – Налаштування інтерфейсу FIS editor

У меню *Edit* послідовно додаємо вхідну змінну з розміром базової шкали (Range= [-1 1]) та вихідну змінну з розміром базової шкали (*Range*= [-1 1]).

Для опису вхідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної змінної трикутну функцію приналежності. 3 параметрами:

Name = "NB"; Type = "trimf"; Param = [-1.33 -1 -0.33];
Name = "NM"; Type = " trimf"; Param = [-1 -0.33 -0.03];
Name = "NS"; Type = " trimf"; Param = [-0.33 -0.033 0];
Name = "Z"; Type = " trimf"; Param = [-0.1 0 0.1];
Name = "PS"; Type = " trimf"; Param = [0 0.033 0.33];
Name = "PM"; Type = " trimf"; Param = [0.033 0.33 1];
Name = "PB"; Type = "trimf"; Param = [0.33 1 1.33].

Результат налаштування функцій приналежності вхідних змінних наведено на рис. 10.73.

Для опису вихідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної трикутну функцію приналежності. 3 параметрами:

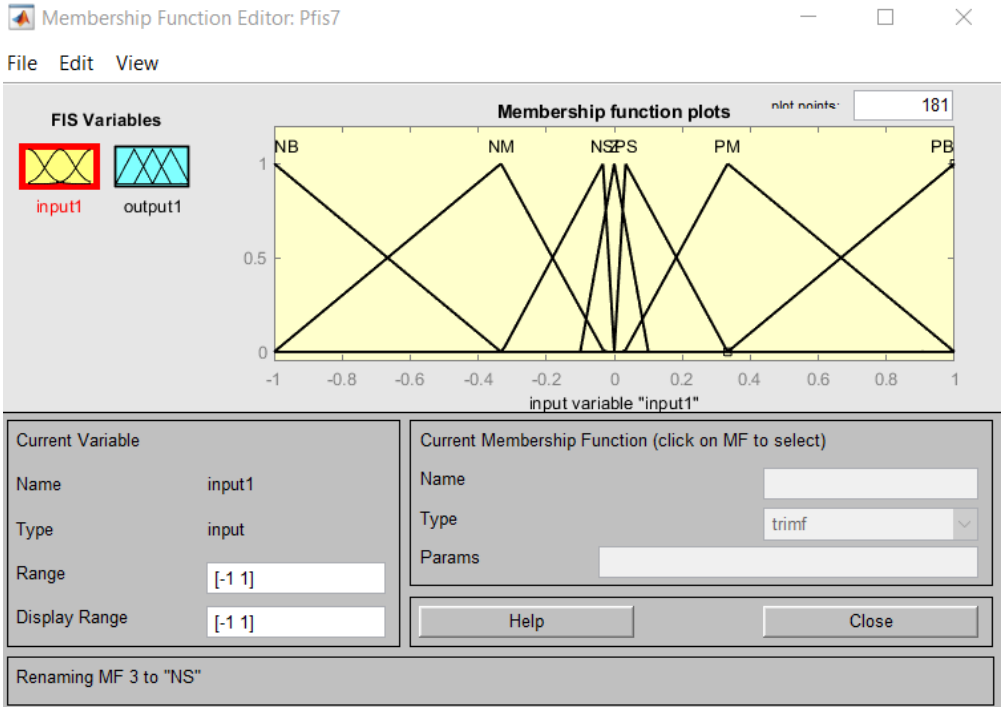


Рисунок 10.73 – Результат налаштування функцій приналежності вхідних змінних

Name = "NBu"; Type = "trimf"; Param = [-1.33 -1 -0.66];
Name = "NMu"; Type = " trimf"; Param = [-1 -0.66 -0.33];
Name = "NSu"; Type = " trimf"; Param = [-0.66 -0.33 0];
Name = "Zu"; Type = " trimf"; Param = [-0.33 0 0.33];
Name = "PSu"; Type = " trimf"; Param = [0 0.33 0.66];
Name = "PMu"; Type = " trimf"; Param = [0.33 0.66 1];
Name = "PBu"; Type = "trimf"; Param = [0.66 1 1.33].

Результат налаштування функцій приналежності вихідних змінних наведено на рис. 10.74.

Примітка. У якості центрів термів, що описуються функціями приналежності, обрані значення відповідно до табл. 10.9.

Після опису вхідні та вихідні лінгвістичні змінні, здійснюється опис правил у вікні редактора Rule Editor, кількість правил для даного випадку згідно табл. 10.4 дорівнює 7.

1. if (input1 is NB) then (output1 is NBu) (1)
2. if (input1 is NM) then (output1 is NMu) (1)
3. if (input1 is NS) then (output1 is NSu) (1)
4. if (input1 is Z) then (output1 is Zu) (1)
5. if (input1 is PS) then (output1 is PSu) (1)
6. if (input1 is PM) then (output1 is NMu) (1)

7. if (input1 is PB) then (output1 is PBu) (1)

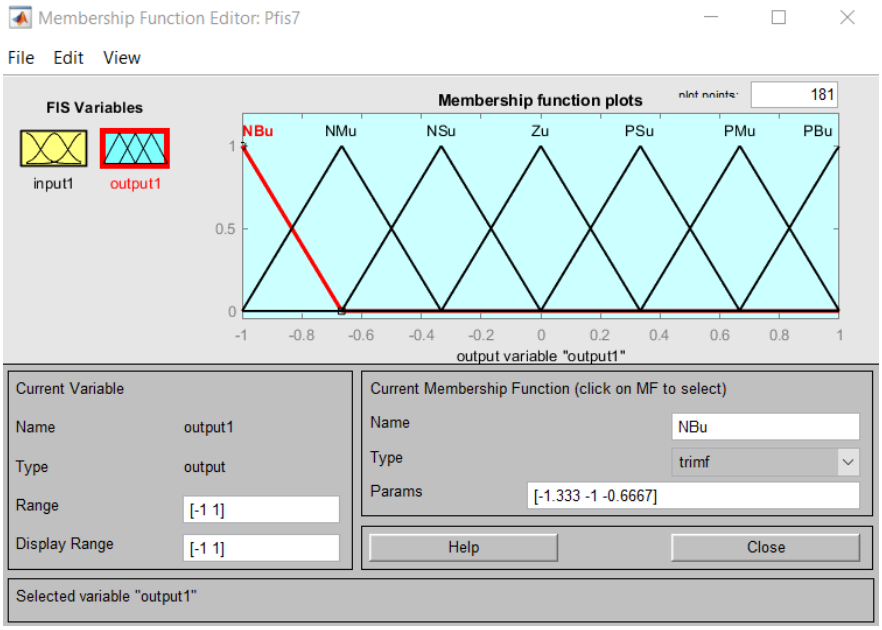


Рисунок 10.74 – Результат налаштування функцій приналежності вихідних змінних

Налаштовування правил для даного випадку наведено на рисунку (див. рис. 10.75).

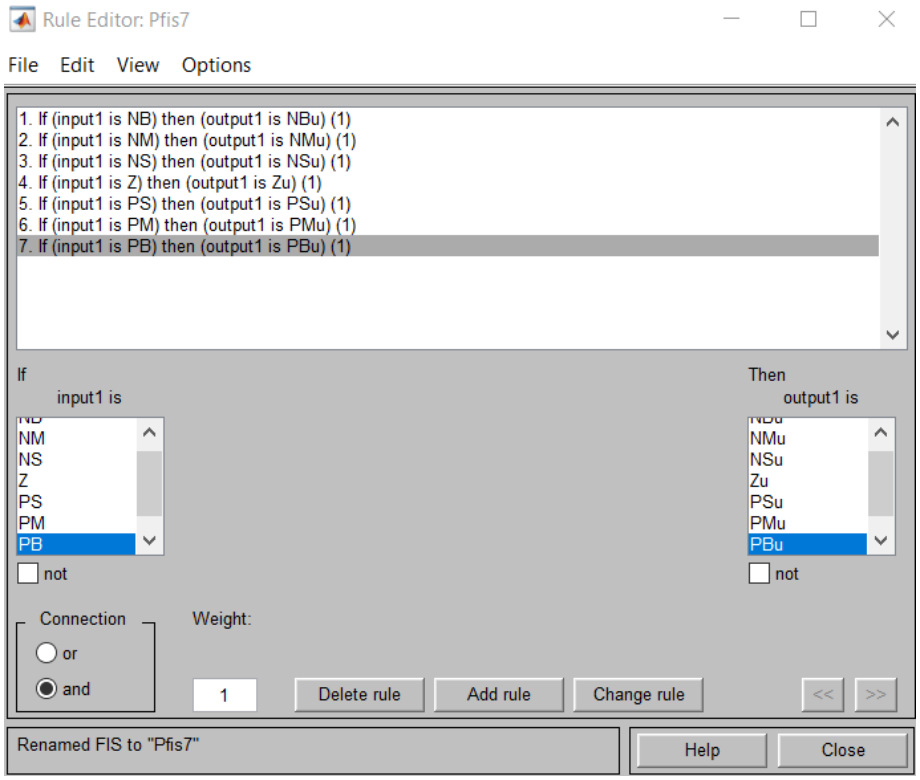


Рисунок 10.75 - Налаштовування опису правил функціонування НЛР_П

Посилки у правилах пов'язані (connection) за допомогою операції *and*. Введене правило забезпечене ваговим коефіцієнтом (*Weight*=1).

Після опису правил можливо за допомогою

- інтерфейсу *Rule Viewer* можливо переглянути роботу системи нечіткого висновку за різних вхідних даних (див. рис. 10.76);

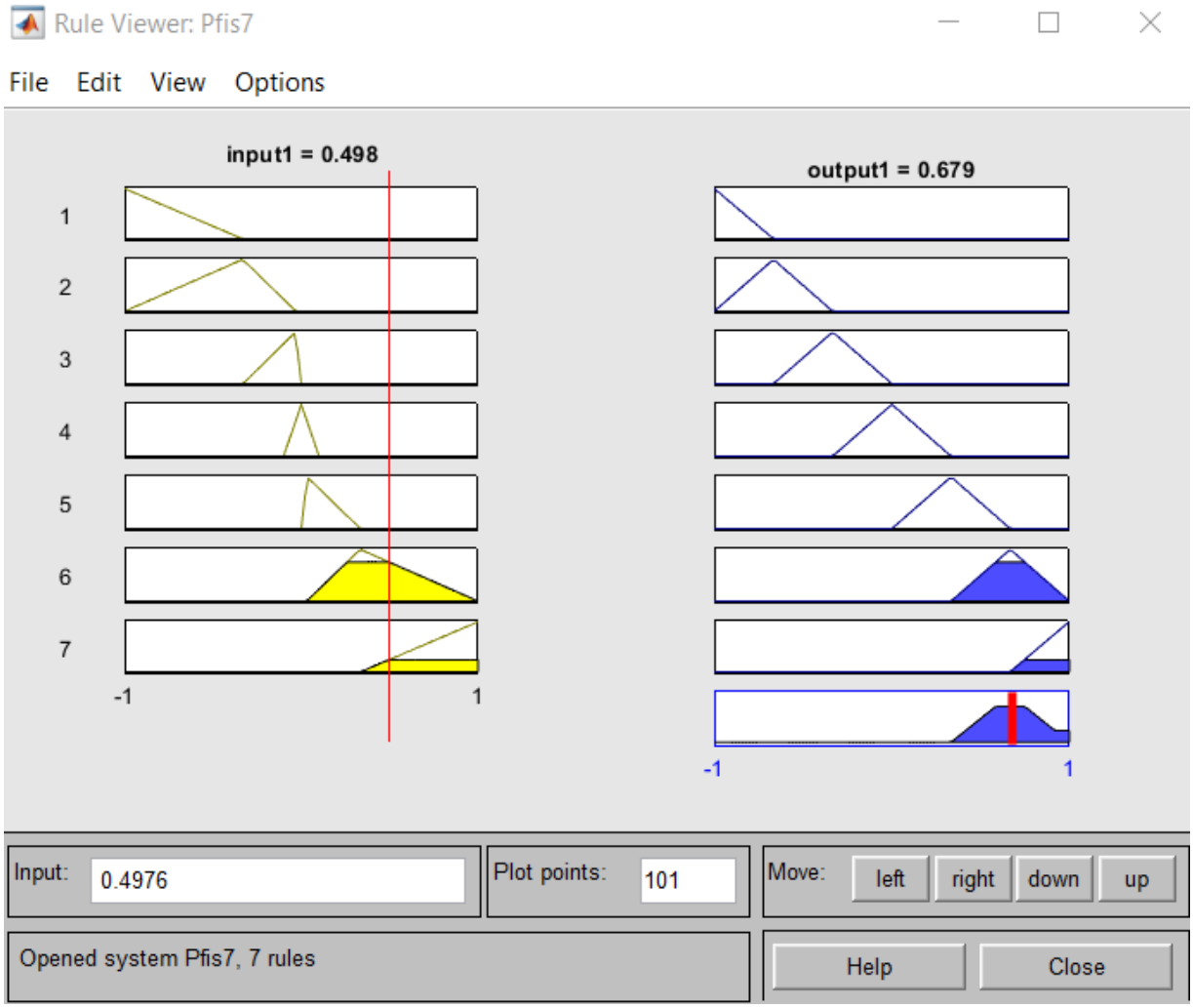


Рисунок 10.76 - Робота системи НЛР_П

− інтерфейсу Surface Viewer можливо переглянути графічне подання закону управління (рис. 10.77).

Після процедур налаштування НЛР_П у редакторі системи нечіткого висновку здійснює запис результатів налаштування у Fuzzy Logic Controller. Для цього у вкладці File здійснюємо Export в математичну модуль нечіткого контролера From Workspase вказавши ім’я Fuzzy Logic Controller.

Після експорту закону функціонування НЛР_П на математичної моделі системи управління (див. рис. 10.71) получено перехідний процес для синтезованого НЛР_П, який задовольняє поставленому завданню проектування

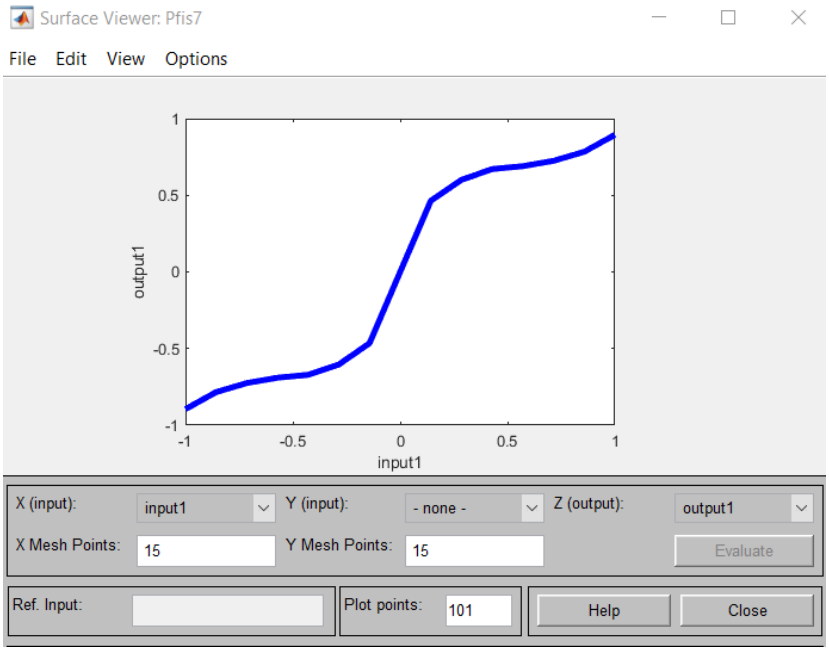


Рисунок 10.77 – Крива управління системою нечіткого висновку

На рис. 10.78 показаний перехідний процес для синтезованого НЛР_П.

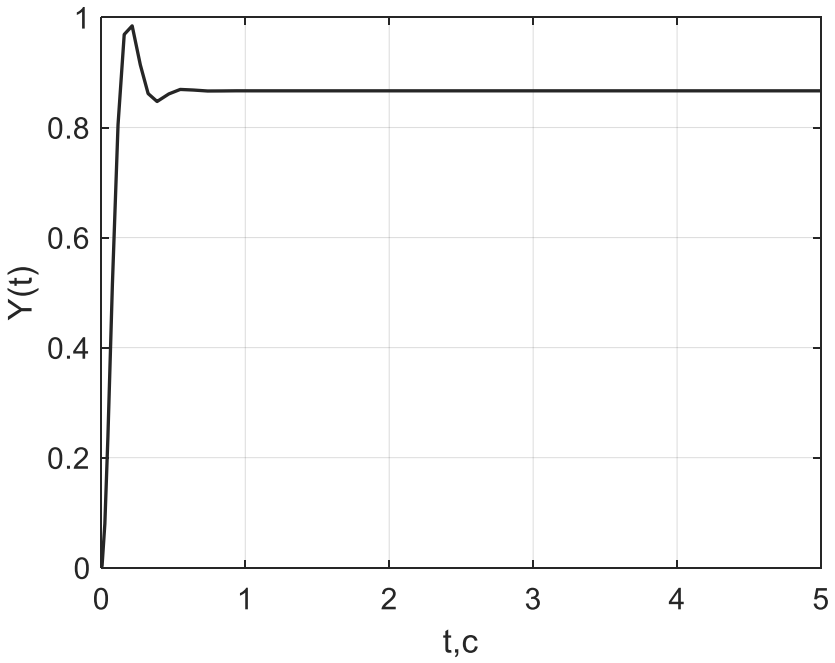


Рисунок 10.78 - Перехідний процес для НЛР_П із сімома правилами

10.12 Аналітичний опис нечіткого логічного регулятора ПД-типу

Розглянемо при зроблених вище припущеннях аналітичний опис НЛР_ПД.

Відповідно до рис. 10.55 значення приналежності помилки до терм із номерами *i* та *i* +1 (E_j та E_{j+1}) розраховуються за формулами:

$$\mu_i(e) = \frac{e_{i+1} - e}{e_{i+1} - e_i}, \mu_{i+1}(e) = \frac{e - e_i}{e_{i+1} - e_i} = 1 - \mu_i(e).$$

Аналогічно, якщо збільшення помилки Δ*e* знаходиться між термами з номерами *k* та *k*1 (ΔE_k та ΔE_{k+1}), то значенню приналежності до цих терм відповідає:

$$\mu_i(\Delta e) = \frac{\Delta e_{k+1} - \Delta e}{\Delta e_{k+1} - \Delta e_k}, \mu_{i+1}(\Delta e) = \frac{\Delta e - \Delta e_k}{\Delta e_{k+1} - \Delta e_k} = 1 - \mu_k(e\Delta).$$

Нехай закон управління НЛР _ПД сформований як набору правил:

$$R_{i,k}: \text{Якщо } (e = E_i) \text{ і } (\Delta e = \Delta E_k), \text{то } u = U_{i,k}.$$

Нехай для реалізації логічної зв'язки «І» використовується множення, тоді ступінь запуску правила $R_{i,k}$ можна оцінити за формулою:

$$\mu_{i,k} = \left(\frac{e_{i+1} - e}{e_{i+1} - e_i}\right) \cdot \left(\frac{\Delta e_{k+1} - \Delta e}{\Delta e_{k+1} - \Delta e_k}\right).$$

Оскільки терми помилки та збільшення помилки описані за допомогою трикутних функцій приналежності, що утворюють нечітке розбиття відповідних базових областей, спрацювати можуть ще три правила зі ступенями запуску:

$$\mu_{i,k+1} = \left(\frac{e_{i+1} - e}{e_{i+1} - e_i}\right) \cdot \left(\frac{\Delta e - \Delta e_k}{\Delta e_{k+1} - \Delta e_k}\right);$$

$$\mu_{i+1,k} = \left(\frac{e - e_i}{e_{i+1} - e_i}\right) \cdot \left(\frac{\Delta e_{k+1} - \Delta e}{\Delta e_{k+1} - \Delta e_k}\right);$$

$$\mu_{i+1,k+1} = \left(\frac{e - e_i}{e_{i+1} - e_i}\right) \cdot \left(\frac{\Delta e - \Delta e_k}{\Delta e_{k+1} - \Delta e_k}\right).$$

Дефаззифікація виконується за формулою:

$$u = \frac{U_{i,k}\mu_{i,k} + U_{i+1,k}\mu_{i+1,k} + U_{i,k+1}\mu_{i,k+1} + U_{i+1,k+1}\mu_{i+1,k+1}}{\mu_{i,k} + \mu_{i+1,k} + \mu_{i,k+1} + \mu_{i+1,k+1}}$$

Як можна помітити, знаменник цієї формули завжди дорівнює

одиниці:

$$\mu_{i,k} + \mu_{i+1,k} + \mu_{i,k+1} + \mu_{i+1,k+1} == \mu_i(e)\mu_k(\Delta e) + \big(1 - \mu_i(e)\big)\mu_k(\Delta e) + \\ + \mu_i(e)(1 - \mu_k(\Delta e)) + (1 - \mu_i(e))(1 - \mu_k(\Delta e)) = 1$$

Тому сигнал керування можна записати у вигляді

$$u = U_{i,k}\left(\frac{e_{i+1} - e}{e_{i+1} - e_i}\right) \cdot \left(\frac{\Delta e_{k+1} - \Delta e}{\Delta e_{k+1} - \Delta e_k}\right) + \\ + U_{i+1,k}\left(\frac{e - e_i}{e_{i+1} - e_i}\right) \cdot \left(\frac{\Delta e_{k+1} - \Delta e}{\Delta e_{k+1} - \Delta e_k}\right) + \\ + U_{i,k+1}\left(\frac{e_{i+1} - e}{e_{i+1} - e_i}\right) \cdot \left(\frac{\Delta e - \Delta e_k}{\Delta e_{k+1} - \Delta e_k}\right) + U_{i+1,k+1}\left(\frac{e - e_i}{e_{i+1} - e_i}\right) \cdot \left(\frac{\Delta e - \Delta e_k}{\Delta e_{k+1} - \Delta e_k}\right).$$

Виконуючи елементарні перетворення, отримуємо

$$\begin{aligned} u(e_{i+1} - e_i)(\Delta e_{k+1} - \Delta e_k) \\ &= U_{i,k}(e_{i+1} - e)(\Delta e_{k+1} - \Delta e) + U_{i+1,k}(e - e_i)(\Delta e_{k+1} - \Delta e) \\ &+ U_{i,k+1}(e_{i+1} - e)(\Delta e - \Delta e_k) + U_{i+1,k+1}(e - e_i)(\Delta e - \Delta e_k) = \\ &= U_{i,k}(e_{i+1}\Delta e_{k+1} - e_{i+1}\Delta e - e\Delta e_{k+1} + e\Delta e) \\ &+ U_{i+1,k}(ee_{k+1} - e\Delta e - e_i\Delta e_{k+1} + e_i\Delta e) \\ &+ U_{i,k+1}(e_{i+1}\Delta e - e_{i+1}\Delta e_k - e\Delta e + e\Delta e_k) \\ &+ U_{i+1,k+1}(e\Delta e - e\Delta e_k - e_i\Delta e + e_i\Delta e_k) \\ &= e\left(\Delta e_{k+1}\big(U_{i+1,k} - U_{i,k}\big) + \Delta e_k\big(U_{i,k+1} - U_{i+1,k+1}\big)\right) \\ &+ \Delta e\left(e_{i+1}\big(U_{i,k+1} - U_{i,k}\big) + e_i\big(U_{i+1,k} - U_{i+1,k+1}\big)\right) \\ &+ e\Delta e\big(U_{i,k} - U_{i+1,k} - U_{i,k+1} + U_{i+1,k+1}\big) + U_{i,k}e_{i+1}\Delta e_{k+1} \\ &- U_{i+1,k}e_i\Delta e_{k+1} - U_{i,k+1}e_{i+1}\Delta e_k + U_{i+1,k+1}e_i\Delta e_k. \end{aligned}$$

Остаточно

$$\begin{aligned} u = e\left(\frac{\Delta e_{k+1}\big(U_{i+1,k} - U_{i,k}\big) + \Delta e_k\big(U_{i,k+1} - U_{i+1,k+1}\big)}{(e_{i+1} - e_i)(\Delta e_{k+1} - \Delta e_k)}\right) \\ + \Delta e\left(\frac{e_{i+1}\big(U_{i,k+1} - U_{i,k}\big) + e_i\big(U_{i+1,k} - U_{i+1,k+1}\big)}{(e_{i+1} - e_i)(\Delta e_{k+1} - \Delta e_k)}\right) \\ + e\Delta e\left(\frac{U_{i,k} - U_{i+1,k} - U_{i,k+1} + U_{i+1,k+1}}{(e_{i+1} - e_i)(\Delta e_{k+1} - \Delta e_k)}\right) \\ + \left(\frac{e_{i+1}\big(U_{i,k}\Delta e_{k+1} - U_{i,k+1}\Delta e\big) - e_i\big(U_{i+1,k}\Delta e_{i+1,k} + U_{i+1,k+1}\Delta e_k\big)}{(e_{i+1} - e_i)(\Delta e_{k+1} - \Delta e_k)}\right). \end{aligned}$$

Якщо e та Δe – нормовані величини, їх добуток можна вважати малою величиною. Тоді закон управління НЛР_ПД описується

формулою

$$u = k_p e + k_d \Delta e + const. \qquad (10.12)$$

Перші два доданки відповідають класичному ПД-регулятору, однак у НЛР _ПД коефіцієнти k_p та k_d є змінними. Третій доданок (const) може компенсувати помилку, що встановилася.

Запишемо закон управління ПД-регулятора у вигляді:

$$u = k_p \left(e(t) + \frac{k_d}{k_p} \frac{de(t)}{dt} \right)$$

Тоді класичний ПД-регулятор можна зобразити у вигляді показаному на рис. 10.79.

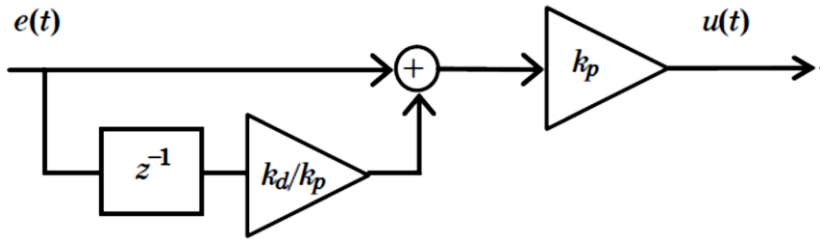


Рисунок 10.79 – Варіант опису ПД-регулятора

Використання похідної дозволяє зменшити перерегулювання, що може виникати при пропорційному управлінні. Однак при цьому можливе небажане збільшення часу перехідного процесу. Методика проектування має узгодити ці два суперечливі ефекти. Тому процес налаштування k_d та k_p передбачає такі кроки:

1) коефіцієнт k_d отримує по можливості невелике значення для досягнення найбільш швидкої реакції системи;

2) пропорційний коефіцієнт k_p повинен бути завжди більшим, ніж k_d.

10.13 Синтез нечіткого регулятора ПД-типу

НЛР_ПД відрізняється від класичного ПД-регулятора тим, що може реалізувати нелінійний закон управління, який описуватиметься деякою поверхнею в тривимірному просторі.

При цифровій реалізації НЛР_ПД замість похідної помилки управління часто розглядається її збільшення $\Delta e(t)$, так що поведінку НЛР_ПД можна описати виразом

$$u(t) = F_{Л}\big(e(t), \Delta e(t)\big),$$

де $F_л$ - лінгвістичний закон управління, заданий набором правил. Структура НЛР _ПД наведено на рис. 10.34.

Як показує формула (10.12), за певних умов НЛР_ПД може бути еквівалентний звичайному ПД-регулятору. Для забезпечення вищої якості роботи НЛР_ПД по відношенню до класичного регулятора потрібно правильно описати керуючі правила та правильно вибрати розташування термів лінгвістичних змінних.

Управляючі правила можна вибрати на основі якісного аналізу перехідного процесу.

Відповідно до (10.1) помилка управління обчислюється за формулою

$$e(t) = g(t) - y(t).$$

Негативне значення помилки $e(t)$ означає, що вихідний сигнал $y(t)$ більше уставки $g(t)$, і його потрібно зменшувати, позитивне значення $e(t)$) означає, що вихідний сигнал менший за уставку, і його потрібно збільшувати. Нульове значення $e(t)$ відповідає, очевидно, рівності $g(t)$ та $y(t)$.

Розглянемо збільшення помилки управління:

$$\Delta e(t) = e(t) - e(t-1) = g(t) - y(t) - g(t) + y(t-1) = y(t-1) - y(t) = -\Delta y(t)$$

Позитивний знак $\Delta e(t)$ означає, що вихідний сигнал зменшується. Негативне значення $\Delta e(t)$ означає, що вихідний сигнал збільшується. Нульове значення $\Delta e(t)$ відповідає постійному вихідному сигналу.

На рис. 10.80 показано поведінку помилки та збільшення помилки при нормальному перебігу перехідного процесу. Як показує рисунок, наближення вихідного сигналу системи до заданого значення відбувається тоді, коли $e(t)$ та $\Delta e(t)$ мають різні знаки. У цьому випадку керування на вхід об'єкта можна не подавати. В інших ситуаціях потрібна корекція.

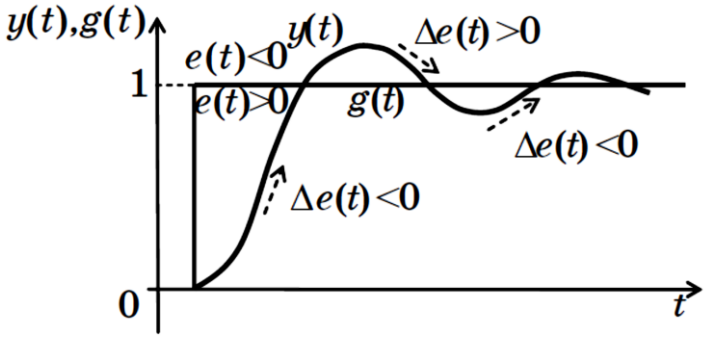


Рисунок 10.80 – Зміна помилки при нормальному перебігу перехідного процесу

Зроблені зауваження дозволяють класифікувати всі, що виникають при управлінні ситуації (табл. 10.10).

Таблиця 10.10 – Семантичний опис процесу управління

Ситуація		Опис ситуації	Характеристика ситуації	Управління
Знак $e(t)$	Знак $\Delta e(t)$			
-	+	$y(t) > g(t), \Delta y(t) < 0$	Норма	Нульове (0)
-	-	$y(t) > g(t), \Delta y(t) > 0$	Потрібна корекція	Негативне (-)
-	0	$y(t) > g(t), \Delta y(t) = 0$	Потрібна корекція	Негативне (-)
0	-	$y(t) = g(t), \Delta y(t) > 0$	Потрібна корекція	Негативне (-)
0	+	$y(t) = g(t), \Delta y(t) < 0$	Потрібна корекція	Позитивне (+)
0	0	$y(t) = g(t), \Delta y(t) = 0$	Норма	Нульове (0)
+	+	$y(t) < g(t), \Delta y(t) < 0$	Потрібна корекція	Позитивне (+)
+	-	$y(t) < g(t), \Delta y(t) > 0$	Норма	Нульове (0)
+	0	$y(t) < g(t), \Delta y(t) = 0$	Потрібна корекція	Позитивне (+)

У табл. 10.10 розглядаються лише знаки помилки управління та її збільшення, тому вважатимуться, що кількість термів для лінгвістичних змінних на входах і виході НЛР_ПД дорівнює 3, і закон управління можна сформулювати як таблиці лінгвістичних правил (ТЛП) (табл.10.11).

Табл. 10.11 містить 9 правил управління (лінгвістичне значення управління знаходиться на перетині рядка та стовпця). Ці правила мають універсальний характер, і можуть бути використані для будь-якого об'єкта управління невисокого порядку.

Таблиця 10.11 – Лінгвістичні правила НЛР_ПД

Таблиця правил		e*			
		Н	0	П	
Δ e*	Н	Н	Н	0	u*
	0	Н	0	П	
	П	0	П	П	

Відповідно до (10.7), для П-регулятора знак управління збігається зі знаком помилки. Наприклад, при 5 термах справедлива табл. 10.12.

Таблиця. 10.12 – Лінгвістичний закон управління П-регулятора

*e**	НВ	НМ	0	ПМ	ПВ
*u**	НВ	НМ	Н	ПМ	ПВ

Входи та виходи ПД-регулятора нормалізовані, тому виконується:

$$u^* = e^* + \Delta e^*.$$

Припустимо, що *u**, *e** та Δ*e** мають однакові множини термів, наприклад: НВ, НС, НМ, 0, ПМ, ПС, ПВ. Тоді, враховуючи обмежені розміри базової шкали, можна записати 49 варіантів обмеженої суми:

НВ+НВ=НВ; НВ+НС=НВ; НВ+НМ=НВ; НВ+0=НВ;
НВ+ПМ=НС; НВ+ПС=НМ; НВ+НВ=0;
НС+НВ=НВ; НС+НС=НВ;НС+НМ=НВ;НС+0=НС;
НС+ПМ=НМ;НС+ПС=0;НС+ПВ=ПМ;
…
ПВ+НВ=0; ПВ+НС=ПМ; ПВ+НМ=ПС; ПВ+0=ПВ;
ПВ+ПМ=ПВ; ПВ+ПС=ПВ; ПВ+ПВ=ПВ

Лінгвістичний закон управління набуває вигляду, показано у табл. 10.13. Табл. 10.13 можна модифікувати за допомогою додаткових евристичних міркувань. Наприклад, можна вимагати, щоб при великому значенні помилки сигнал керування також був великим незалежно від значення збільшення помилки (табл. 10.14).

Таблиця 10.13 – Табличні лінгвістичні правила

Таблиця правил		e*						
		НВ	НС	НМ	0	ПМ	ПС	ПВ
Δ e*	НВ	НВ	НВ	НВ	НВ	НС	НМ	0
	НС	НВ	НВ	НВ	НС	НМ	0	ПМ
	НМ	НВ	НВ	НС	НМ	0	ПМ	ПС
	Н	НВ	НС	НМ	0	ПМ	ПС	ПВ
	ПМ	НС	НМ	0	ПМ	ПС	ПВ	ПВ
	ПС	НМ	0	ПМ	ПС	ПВ	ПВ	ПВ
	ПВ	0	ПМ	ПС	ПВ	ПВ	ПВ	ПВ

u*

Таблиця 10.14 – Модифіковані табличні лінгвістичні правила

Таблиця правил		e*						
		НВ	НС	НМ	0	ПМ	ПС	ПВ
Δ e*	НВ	НВ	НВ	НВ	НВ	НС	НМ	ПВ
	НС	НВ	НВ	НВ	НС	НМ	ПМ	ПВ
	НМ	НВ	НВ	НС	НМ	0	ПС	ПВ
	Н	НВ	НС	НМ	0	ПМ	ПС	ПВ
	ПМ	НВ	НС	0	ПМ	ПС	ПВ	ПВ
	ПС	НВ	НМ	ПМ	ПС	ПВ	ПВ	ПВ
	ПВ	НВ	Н	ПС	ПВ	ПВ	ПВ	ПВ

u*

Використання модифікованої таблиці лінгвістичних правил має зменшити час перехідного процесу у системі.

Евристичний закон керування, заданий табл. 10.13 та 10.14, є універсальним для об'єктів з одним входом та одним виходом. Налаштування закону управління для конкретного об'єкта передбачає вибір коефіцієнтів нормалізації та денормалізації, а також центрів термів лінгвістичних змінних на відповідних базових шкалах.

Приклад. Нехай як базова використовується структура, показана на рис. 10.71. При заміні регулятора на НЛР_ПД, вона набуває вигляду, показаного на рис. 10.81.

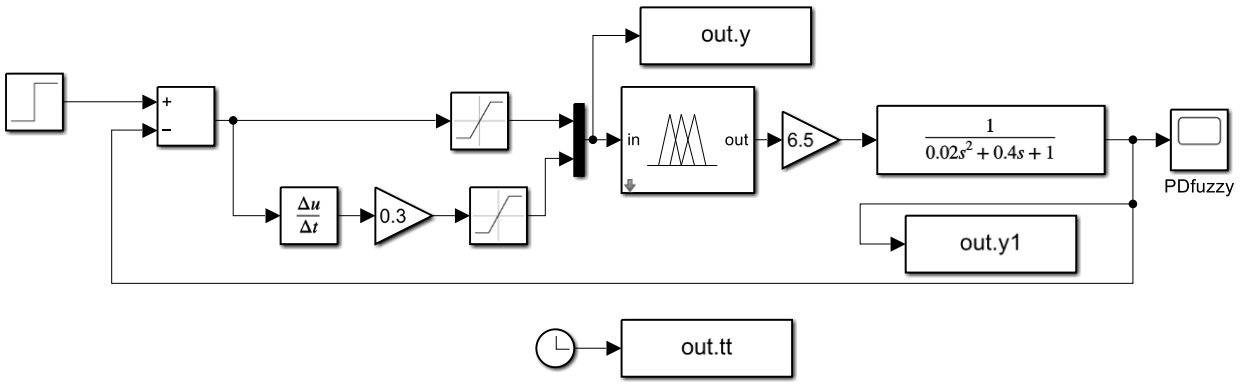


Рисунок 10.81 – Математична модель системи управління з НЛР_ПД

Вирішимо дане завдання згідно описаного алгоритму з використанням редактора системи нечіткого висновку.

Будуємо у MatLab Simulink математичну модель об'єкту керування з послідовним включення Fuzzy Logic Controller до класичного регулятора (див. рис. 10.71). Fuzzy Logic Controller надаємо Fis name: PDfis7 (ім'я може буди будь-яке).

Викликається редактор системи нечіткого висновку (Fuzzy Inference System – FIS) у MatLab командою

>> fuzzy

Дале проводиться налаштовування головного вікна редактора нечіткої логічної системи (див. рис. 10.82.).

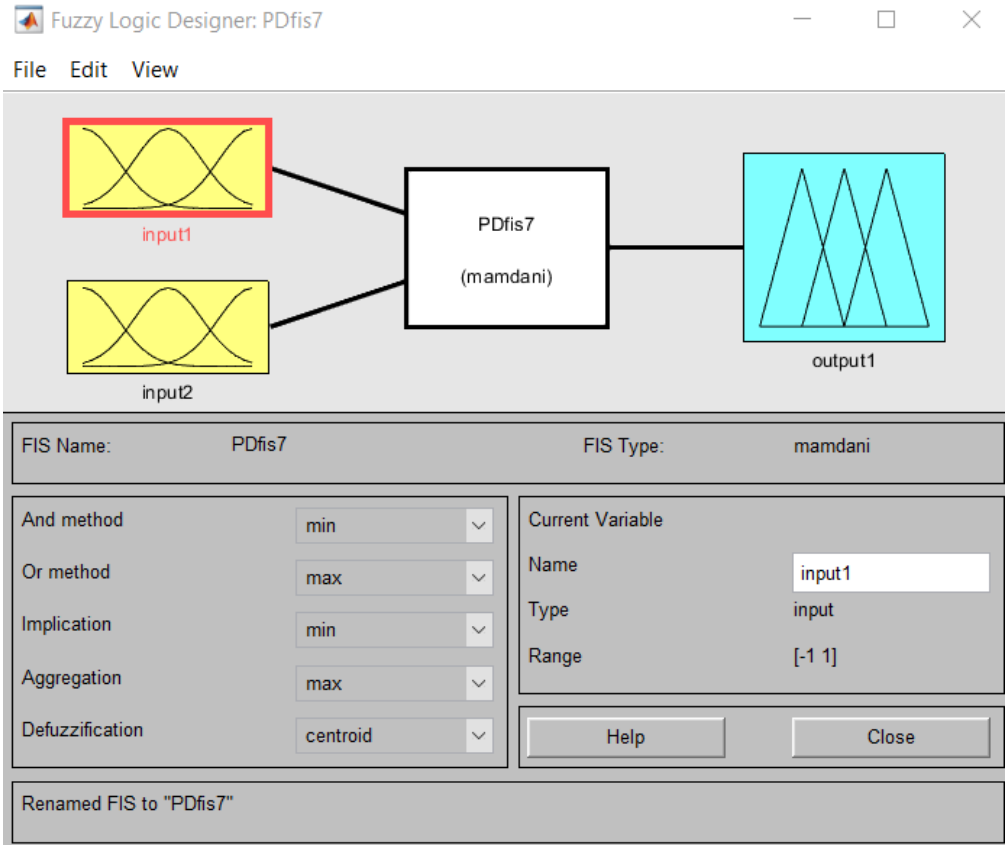


Рисунок 10.82 – Налаштування інтерфейсу FIS editor

У *FIS editor* задається опис систему нечіткого логічного висновку *Mamdani*. Для створюваної системи обирається вид логічного зв'язку (*And method – min*) та (*Or method – max*), вид імплікації (*Implication – min*), спосіб агрегування висновків правил (*Aggregation – max*) та метод дефазифікації (*Defuzzification – centroid*).

У меню *Edit* послідовно додаємо дві вхідних змінних з розміром базової шкали (Range= [-1 1]) та вихідну змінну з розміром базової шкали (*Range*= [-1 1]).

Для опису вхідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної змінної

трикутну функцію приналежності.

Нехай терми лінгвістичної змінної «Помилка управління» розміщуються відповідно до рис. 10.73 згідно налаштуванням НЛР_П регулятора з параметрами:

Name = "NB"; Type = "trimf"; Param = [-1.33 -1 -0.33];
Name = "NM"; Type = " trimf"; Param = [-1 -0.33 -0.03];
Name = "NS"; Type = " trimf"; Param = [-0.33 -0.033 0];
Name = "Z"; Type = " trimf"; Param = [-0.1 0 0.1];
Name = "PS"; Type = " trimf"; Param = [0 0.033 0.33];
Name = "PM"; Type = " trimf"; Param = [0.033 0.33 1];
Name = "PB"; Type = "trimf"; Param = [0.33 1 1.33].

Результат налаштування функцій приналежності вхідних змінних «Помилка управління» наведено на рис. 10.83.

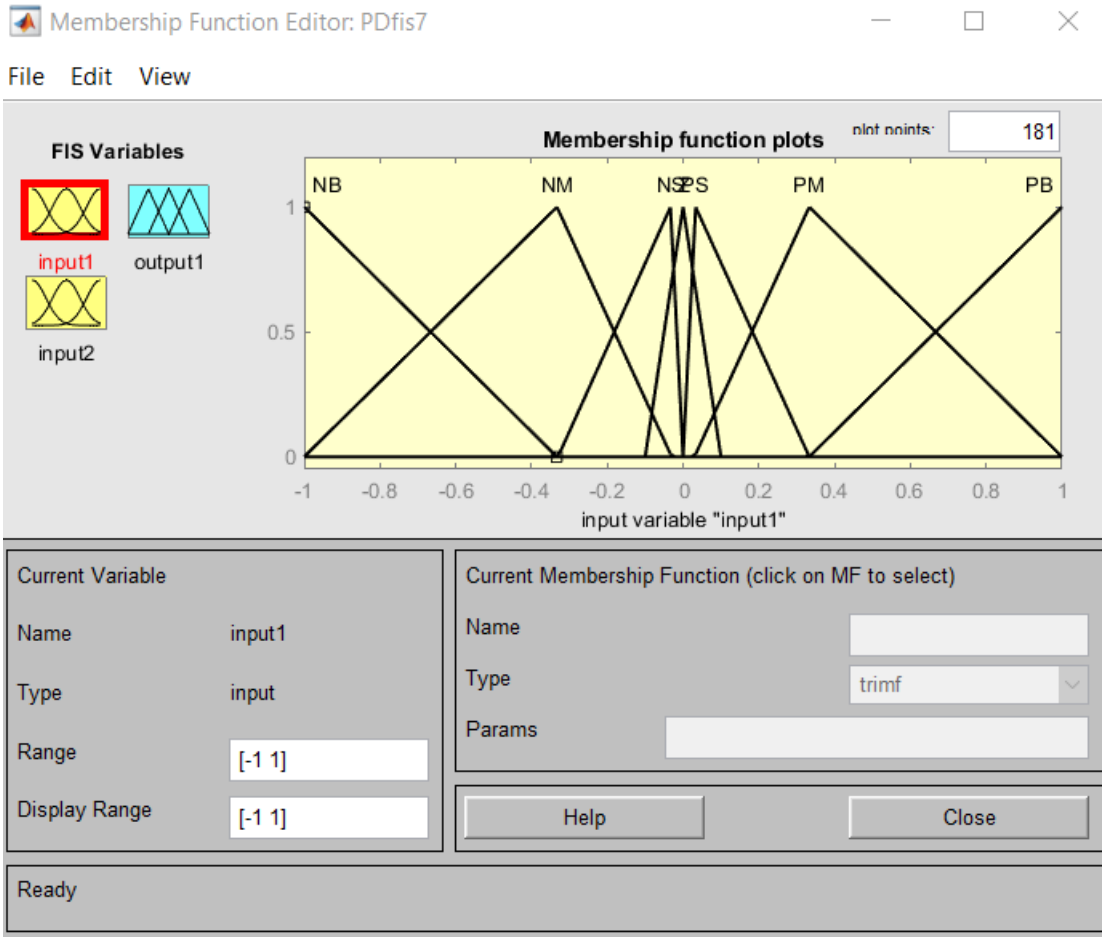


*Рисунок 10.83 – Результат налаштування функцій приналежності вхідних змінних «*Помилка управління»

Терми вхідної лінгвістичної змінної «Похідна помилки управління» формуємо рівномірно. Для опису вхідних логічних змінних «Похідна помилки управління» у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної трикутну функцію

приналежності. 3 параметрами:

Name = "NBe"; Type = "trimf"; Param = [-1.33 -1 -0.66];
Name = "NMe"; Type = " trimf"; Param = [-1 -0.66 -0.33];
Name = "NSe"; Type = " trimf"; Param = [-0.66 -0.33 0];
Name = "Ze"; Type = " trimf"; Param = [-0.33 0 0.33];
Name = "PSe"; Type = " trimf"; Param = [0 0.33 0.66];
Name = "PMe"; Type = " trimf"; Param = [0.33 0.66 1];
Name = "PBe"; Type = "trimf"; Param = [0.66 1 1.33].

Результат налаштування функцій приналежності вхідних змінних «Похідна помилки управління» на рис. 10.84.

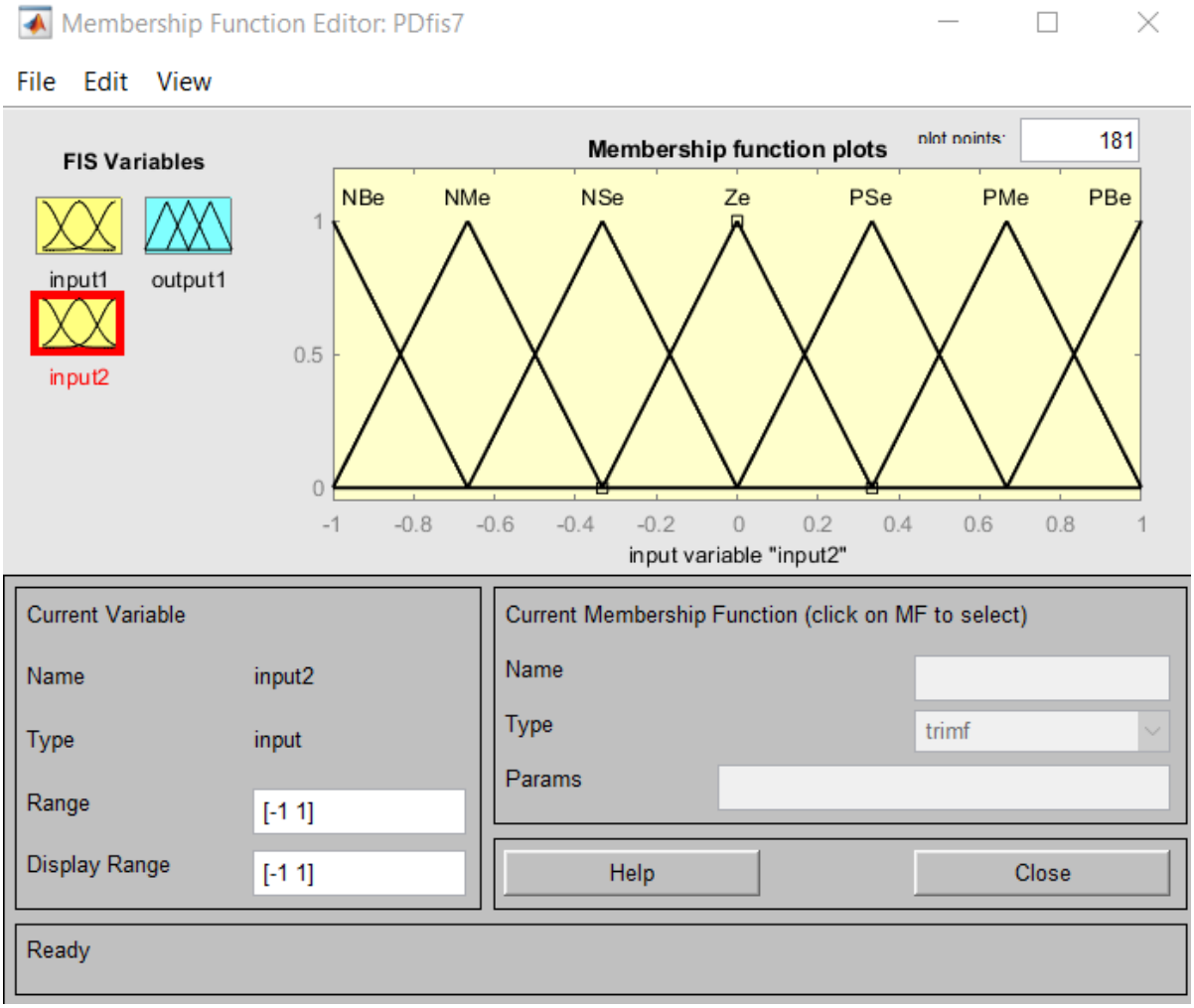


Рисунок 10.84 – Результат налаштування функцій приналежності вхідних змінних «Похідна помилки управління»

Для опису вихідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної трикутну функцію приналежності. 3 параметрами:

Name = "NBu"; Type = "trimf"; Param = [-1.33 -1 -0.66];
Name = "NMu"; Type = " trimf"; Param = [-1 -0.66 -0.33];
Name = "NSu"; Type = " trimf"; Param = [-0.66 -0.33 0];
Name = "Zu"; Type = " trimf"; Param = [-0.33 0 0.33];
Name = "PSu"; Type = " trimf"; Param = [0 0.33 0.66];
Name = "PMu"; Type = " trimf"; Param = [0.33 0.66 1];
Name = "PBu"; Type = "trimf"; Param = [0.66 1 1.33].

Результат налаштування функцій приналежності вхідних змінних наведено на рис. 10.85.

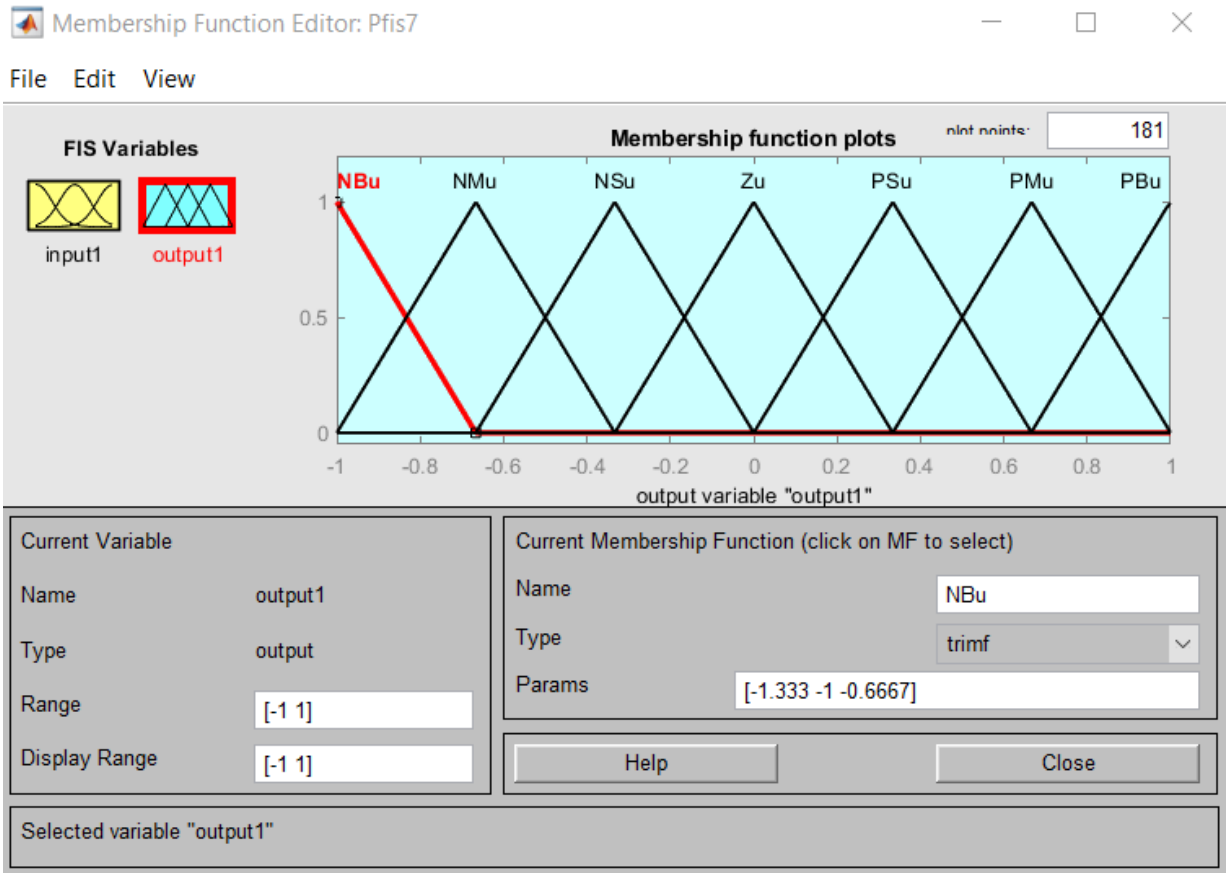


Рисунок 10.85 – Результат налаштування функцій приналежності вихідних змінних

Примітка. У якості центрів термів, що описуються функціями приналежності, обрані значення відповідно до табл. 10.9.

Після опису вхідні та вихідні лінгвістичні змінні, здійснюється опис правил у вікні редактора Rule Editor, кількість правил для даного випадку згідно табл. 10.4 дорівнює 49. Управляючі правила сформуємо згідно з табл. 10.14.:

1. if (input1 is NB) and (input2 is NBe) then (output1 is NBu) (1)
2. if (input1 is NB) and (input2 is NMe) then (output1 is NBu) (1)

3. if (input1 is NB) and (input2 is NSe) then (output1 is NBu) (1)
4. if (input1 is NB) and (input2 is Ze) then (output1 is NBu) (1)
5. if (input1 is NB) and (input2 is PSe) then (output1 is NBu) (1)
6. if (input1 is NB) and (input2 is PMe) then (output1 is NBu) (1)
7. if (input1 is NB) and (input2 is PBe) then (output1 is NBu) (1)
8. if (input1 is NM) and (input2 is NBe) then (output1 is NBu) (1)
9. if (input1 is NM) and (input2 is NMe) then (output1 is NBu) (1)
10. if (input1 is NM) and (input2 is NSe) then (output1 is NBu) (1)
11. if (input1 is NM) and (input2 is Ze) then (output1 is NMu) (1)
12. if (input1 is NM) and (input2 is PSe) then (output1 is NMu) (1)
13. if (input1 is NM) and (input2 is PMe) then (output1 is NSu) (1)
14. if (input1 is NM) and (input2 is PBe) then (output1 is Zu) (1)
15. if (input1 is NS) and (input2 is NBe) then (output1 is NBu) (1)
16. if (input1 is NS) and (input2 is NMe) then (output1 is NBu) (1)
17. if (input1 is NS) and (input2 is NSe) then (output1 is NMu) (1)
18. if (input1 is NS) and (input2 is Ze) then (output1 is NSu) (1)
19. if (input1 is NS) and (input2 is PSe) then (output1 is Zu) (1)
20. if (input1 is NS) and (input2 is PMe) then (output1 is PSu) (1)
21. if (input1 is NS) and (input2 is PBe) then (output1 is PMu) (1)
22. if (input1 is Z) and (input2 is NBe) then (output1 is NBu) (1)
23. if (input1 is Z) and (input2 is NMe) then (output1 is NMu) (1)
24. if (input1 is Z) and (input2 is NSe) then (output1 is NSu) (1)
25. if (input1 is Z) and (input2 is Ze) then (output1 is Zu) (1)
26. if (input1 is Z) and (input2 is PSe) then (output1 is PSu) (1)
27. if (input1 is Z) and (input2 is PMe) then (output1 is PMu) (1)
28. if (input1 is Z) and (input2 is PBe) then (output1 is PBu) (1)
29. if (input1 is PS) and (input2 is NBe) then (output1 is NMu) (1)
30. if (input1 is PS) and (input2 is NMe) then (output1 is NSu) (1)
31. if (input1 is PS) and (input2 is NSe) then (output1 is Zu) (1)
32. if (input1 is PS) and (input2 is Ze) then (output1 is PSu) (1)
33. if (input1 is PS) and (input2 is PSe) then (output1 is PMu) (1)
34. if (input1 is PS) and (input2 is PMe) then (output1 is PBu) (1)
35. if (input1 is PS) and (input2 is PBe) then (output1 is PBu) (1)
36. if (input1 is PM) and (input2 is NBe) then (output1 is NMu) (1)
37. if (input1 is PM) and (input2 is NMe) then (output1 is PSu) (1)
38. if (input1 is PM) and (input2 is NSe) then (output1 is PMu) (1)
39. if (input1 is PM) and (input2 is Ze) then (output1 is PMu) (1)
40. if (input1 is PM) and (input2 is PSe) then (output1 is PBu) (1)
41. if (input1 is PM) and (input2 is PMe) then (output1 is PBu) (1)
42. if (input1 is PM) and (input2 is PBe) then (output1 is PBu) (1)
43. if (input1 is PB) and (input2 is NBe) then (output1 is PBu) (1)
44. if (input1 is PB) and (input2 is NMe) then (output1 is PBu) (1)
45. if (input1 is PB) and (input2 is NSe) then (output1 is PBu) (1)
46. if (input1 is PB) and (input2 is Ze) then (output1 is PBu) (1)
47. if (input1 is PB) and (input2 is PSe) then (output1 is PBu) (1)

48. if (input1 is PB) and (input2 is PMe) then (output1 is PBu) (1)
49. if (input1 is PB) and (input2 is PBe) then (output1 is PBu) (1)

Налаштовування управляючих правил для даного випадку наведено на рисунку (див. рис. 10.86).

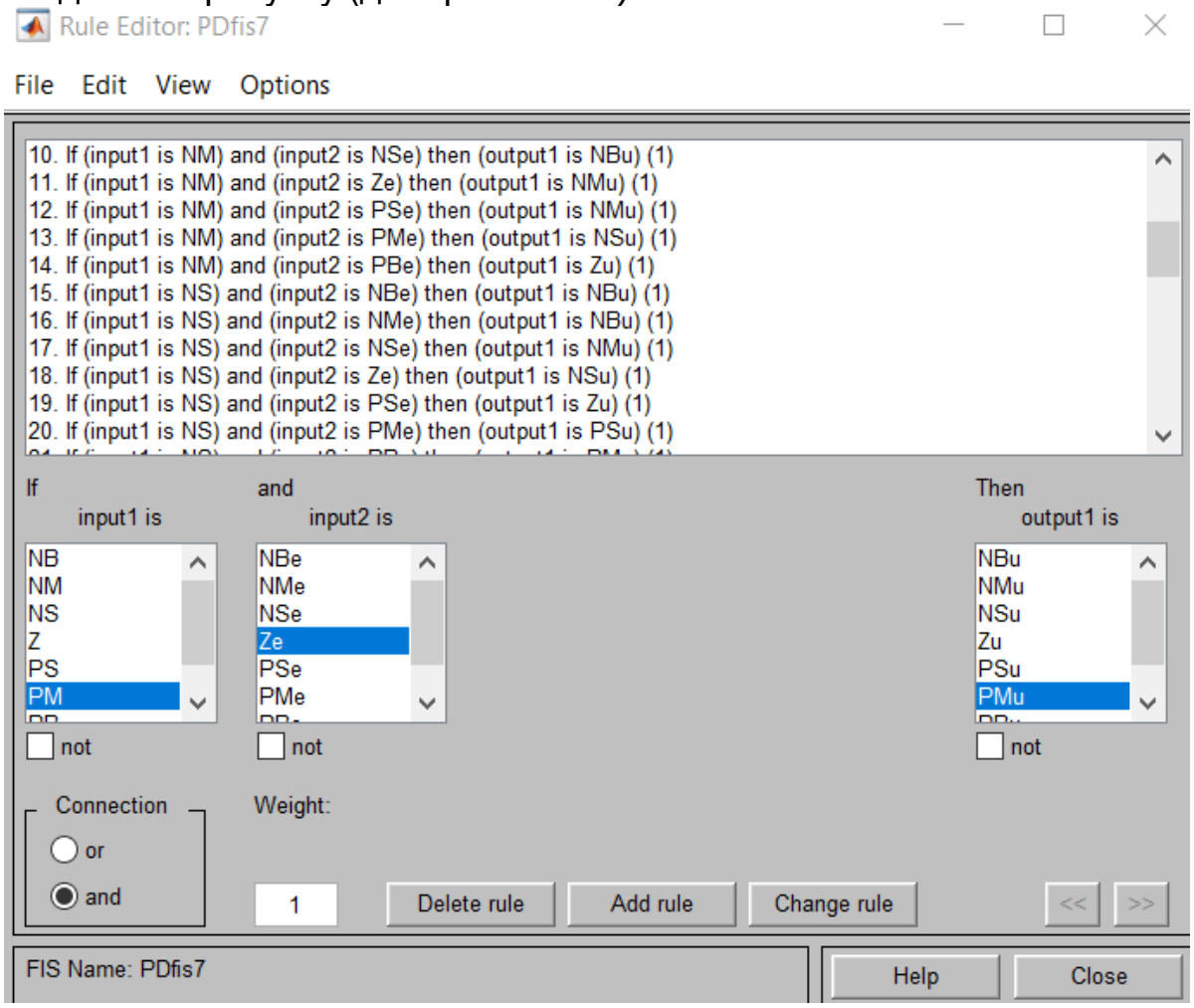


Рисунок 10.86 - Налаштовування опису правил функціонування НЛР_ПД

Посилки у правилах пов'язані (connection) за допомогою операції *and*. Введене правило забезпечене ваговим коефіцієнтом (*Weight*=1).

Після опису правил можливо за допомогою

- інтерфейсу *Rule Viewer* можливо переглянути роботу системи нечіткого висновку за різних вхідних даних (див. рис. 10.87);

− інтерфейсу Surface Viewer можливо переглянути графічне подання закону управління (рис. 10.88).

Після процедур налаштування НЛР_ПД у редакторі системи нечіткого висновку здійснює запис результатів налаштування у Fuzzy Logic Controller. Для цього у вкладці File здійснюємо Export в математичну модуль нечіткого контролера From Workspase вказавши ім’я Fuzzy Logic Controller.

Після експорту закону функціонування НЛР_ПД на математичної моделі системи управління (див. рис. 10.81) получено перехідний процес для синтезованого НЛР_ПД, який задовольняє поставленому завданню проектування

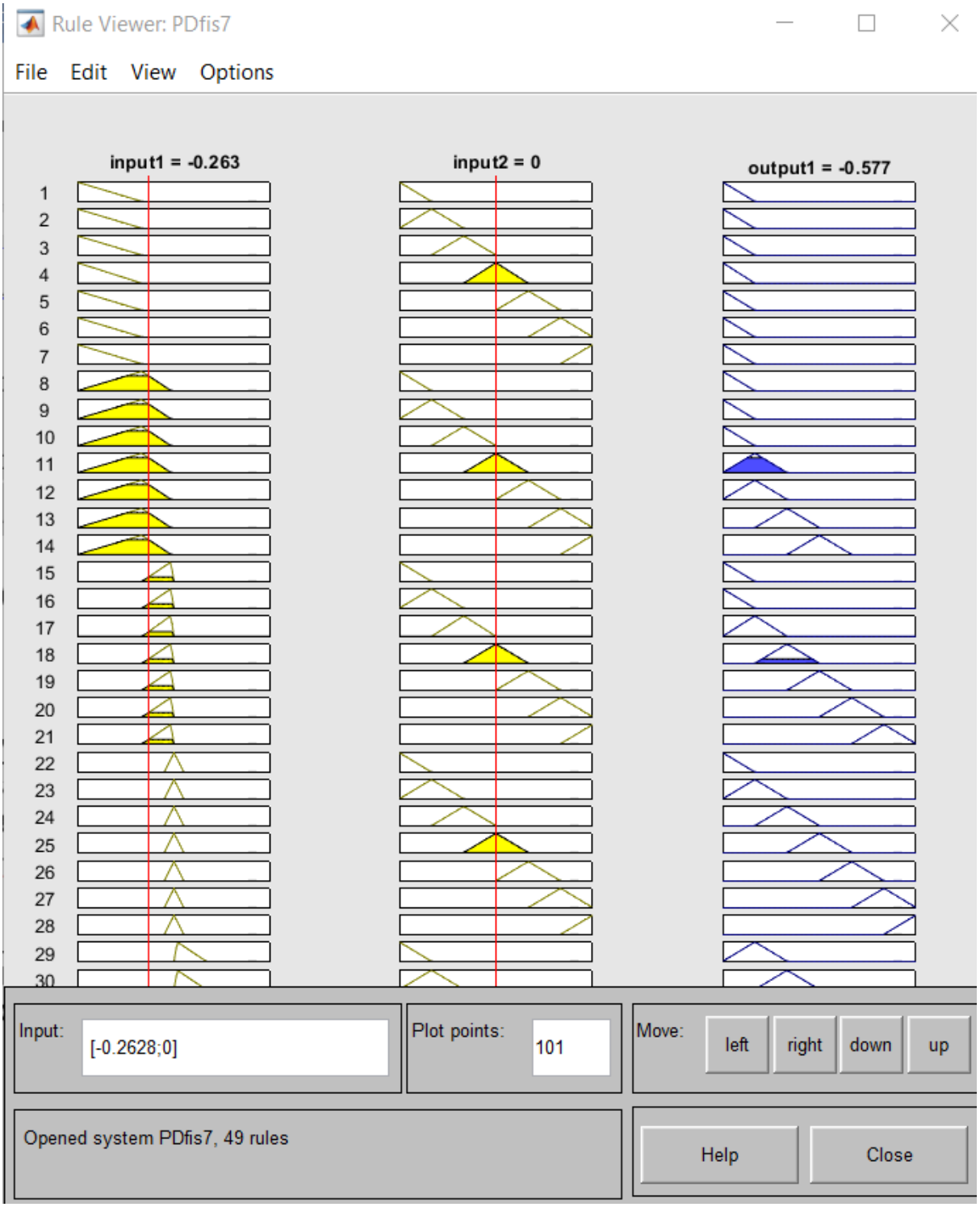


Рисунок 10.87 - Робота системи НЛР_ПД

На рис. 10.89 показаний перехідний процес для синтезованого НЛР_ПД. Як свідчить рис. 10.89, перерегулювання зведено до мінімуму.

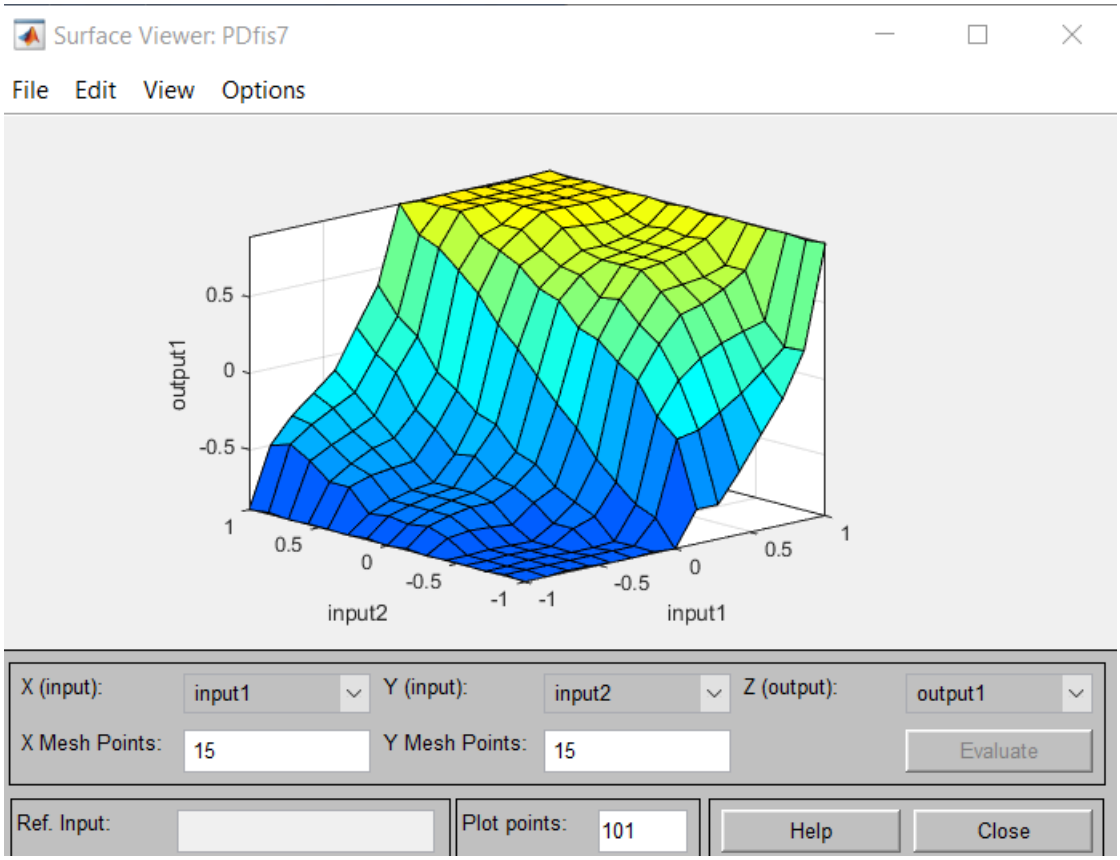


Рисунок 10.88 – Поверхня управління системою нечіткого висновку

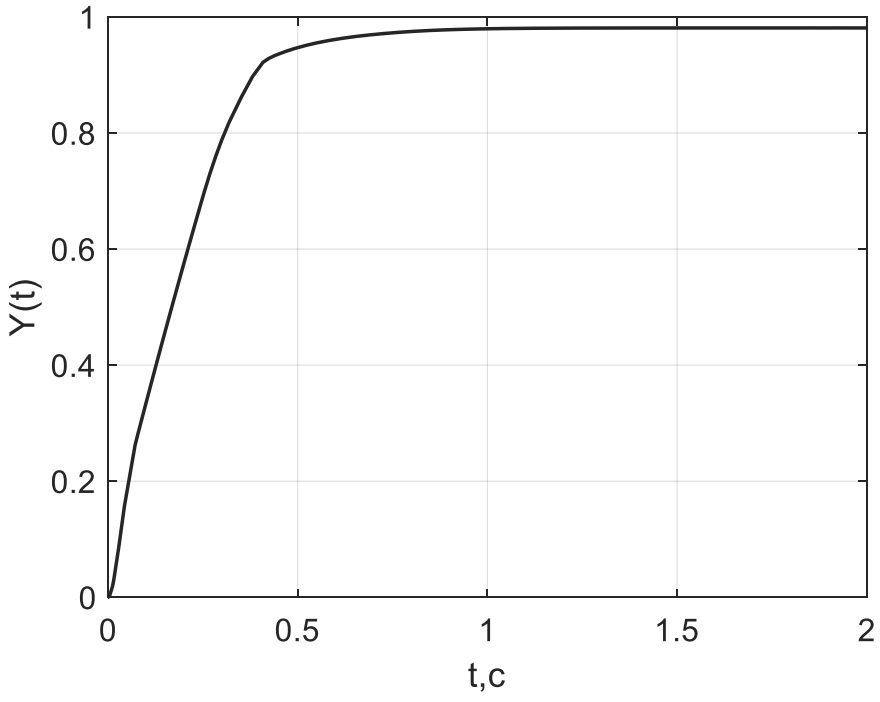


Рисунок 10.89 - Перехідний процес для НЛР_ПД із сімома правилами

Таким чином, вихідними даними для синтезу НЛР_ПД є опис об'єкта управління та вимоги до перехідного процесу: час наростання, перерегулювання, помилка, що встановилася.

Процес проектування НЛР_ПД відбувається у два кроки.

1. Синтезується НЛР_П, що відповідає заданим вимогам до часу наростання та помилки, яка встановилася, відповідно до методики, викладеної в п. 10.10. Перерегулювання прицьому перевищує допустиме значення. Після цього кроку виявляється відомим положення термів лінгвістичних змінних «Помилка» та «Сигнал керування», а також значення коефіцієнта денормалізації k_p. Визначається за графіком перехідного процесу максимальна величина $\Delta e(t)$.

Терми Δe^* розташовуються рівномірно за базовою шкалою.

2. Розглядається закон керування, заданий табл. 10.14. Відповідно до графіка перехідного процесу вибирається коефіцієнт нормалізації для Δe.

$$N_{\Delta e} = \frac{1}{max|\Delta e(t)|}$$

так що $\Delta e_N \in [-1,1]$.

10.14 Синтез нечіткого регулятору ПІ-типу

Можна вважати, що нечіткий регулятор ПІ-типу має такі самі входи, як і НЛР_ПД (див. рис. 10.35 та 10.37). Тому йому справедливі закони управління, показані в табл. 10.11–10.14. Основна відмінність полягає в способі формування сигналу, що управляє, використання якого дозволяє звести до мінімуму статичну помилку в системі.

Найпростіший алгоритм нечіткого ПІ управління можна сформулювати, розглядаючи рис. 10.80. Управління потрібно змінювати лише тоді, коли помилка управління та її збільшення мають однаковий знак. Таким чином, потрібні лише два правила:

− якщо $e(k)$ позитивна та $\Delta e(k)$ позитивна, то $\Delta u(k)$ позитивна;

− якщо $e(k)$ негативне та $\Delta e(k)$ негативне, то $\Delta u(k)$ негативне.

Терми лінгвістичних змінних можуть бути описані способом, показаним на рис. 10.90.

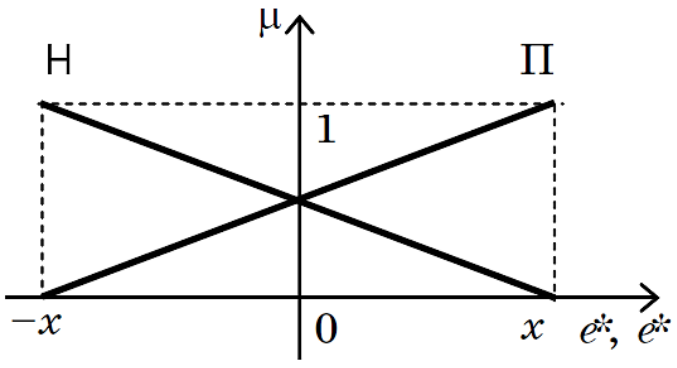


Рисунок 10.90 – Лінгвістичний опис входів та виходів НЛР_ПІ

Таблиця лінгвістичних правил НЛР_ПІ при двох термах для опису $e(k)$ та $\Delta e(k)$ і трьох термах для $u(k)$ (див. рис. 10.56) набуває наступного вигляду (табл. 10.15).

Таблиця 10.15 – Лінгвістичні правила НЛР_ПІ

Таблиця правил		e*(k)	
		Н	П
Δe*(k)	Н	Н	0
	П	П	П

Δu*(t)

Отже, універсальний закон управління, заданий табл. 10.15 вимагає для конкретного об'єкта налаштування лише одного параметра x (див. рис. 10.89).

Приклад. Відповідно до рис. 10.35 сформуємо в Simulink MatLab модель системи управління з НЛР ПІ при x = 6.5 (рис. 10.91).

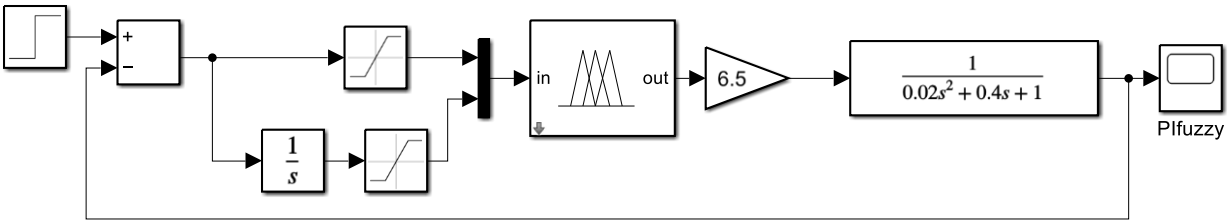


Рисунок 10.91 – Математична модель системи управління з НЛР_ПІ у в Simulink MatLab

Fuzzy Logic Controller надаємо Fis name: PIfis7 (ім’я може буди будь-яке).

Налаштування Fuzzy Logic Controller для НЛР_ПІ аналогічно налаштуванню НЛР_ПД.

Викликається редактор системи нечіткого висновку (Fuzzy Inference System – FIS) у MatLab командою

>> fuzzy

Дале проводиться налаштовування головного вікна редактора нечіткої логічної системи (див. рис. 10.92).

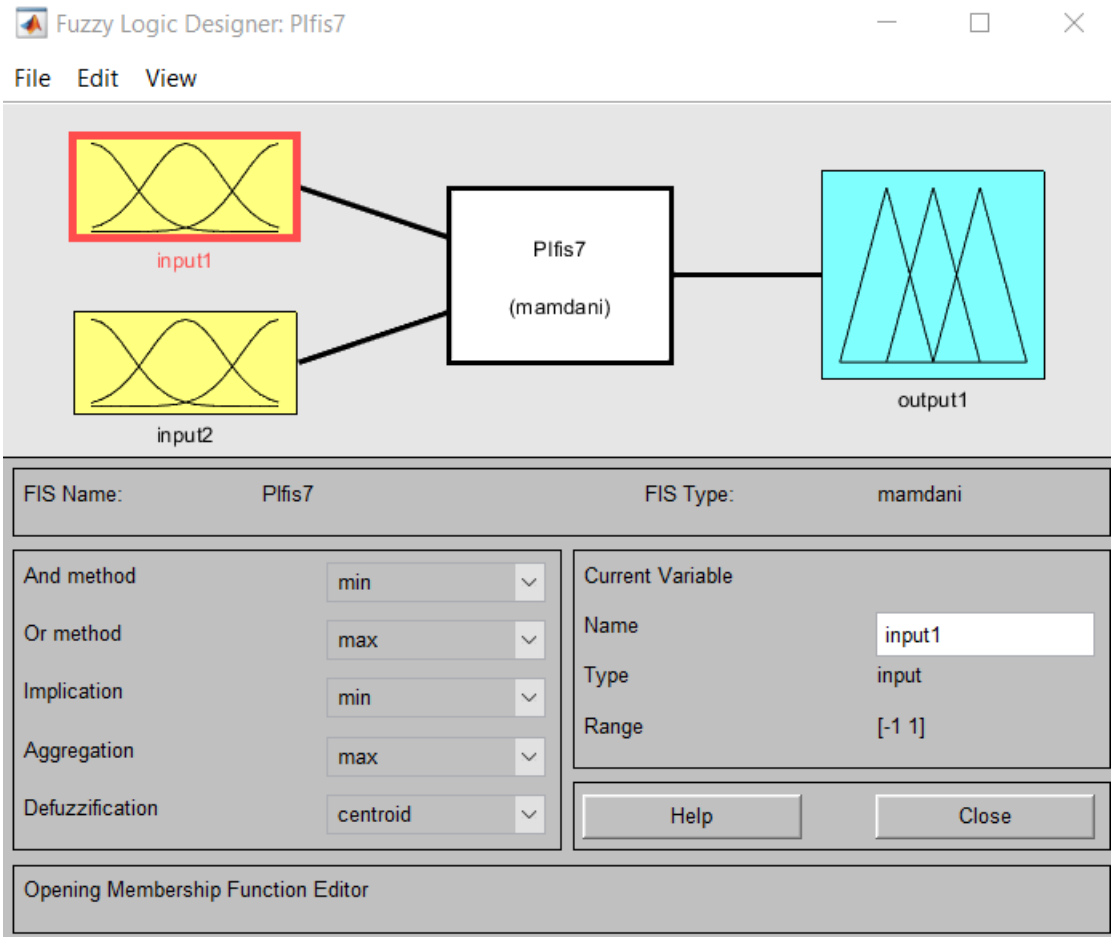


Рисунок 10.92 – Налаштування інтерфейсу FIS editor

У *FIS editor* задається опис систему нечіткого логічного висновку *Mamdani*. Для створюваної системи обирається вид логічного зв'язку (*And method – min*) та (*Or method – max*), вид імплікації (*Implication – min*), спосіб агрегування висновків правил (*Aggregation – max*) та метод дефазифікації (*Defuzzification – centroid*).

У меню *Edit* послідовно додаємо 2 вхідні змінні з розміром базової шкали (Range= [-1 1]) та вихідну змінну з розміром базової шкали (*Range*= [-1 1]).

Для опису вхідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної змінної трикутну функцію приналежності.

Нехай терми лінгвістичної змінної «Помилка управління» розміщуються відповідно до рис. 10.73 згідно налаштуванням НЛР_П регулятора з параметрами:

Name = "NB"; Type = "trimf"; Param = [-1.33 -1 -0.33];
Name = "NM"; Type = " trimf"; Param = [-1 -0.33 -0.03];
Name = "NS"; Type = " trimf"; Param = [-0.33 -0.033 0];
Name = "Z"; Type = " trimf"; Param = [-0.1 0 0.1];
Name = "PS"; Type = " trimf"; Param = [0 0.033 0.33];
Name = "PM"; Type = " trimf"; Param = [0.033 0.33 1];

Name = "PB"; Type = "trimf"; Param = [0.33 1 1.33].

Результат налаштування функцій приналежності вхідних змінних «Помилка управління» наведено на рис. 10.93.

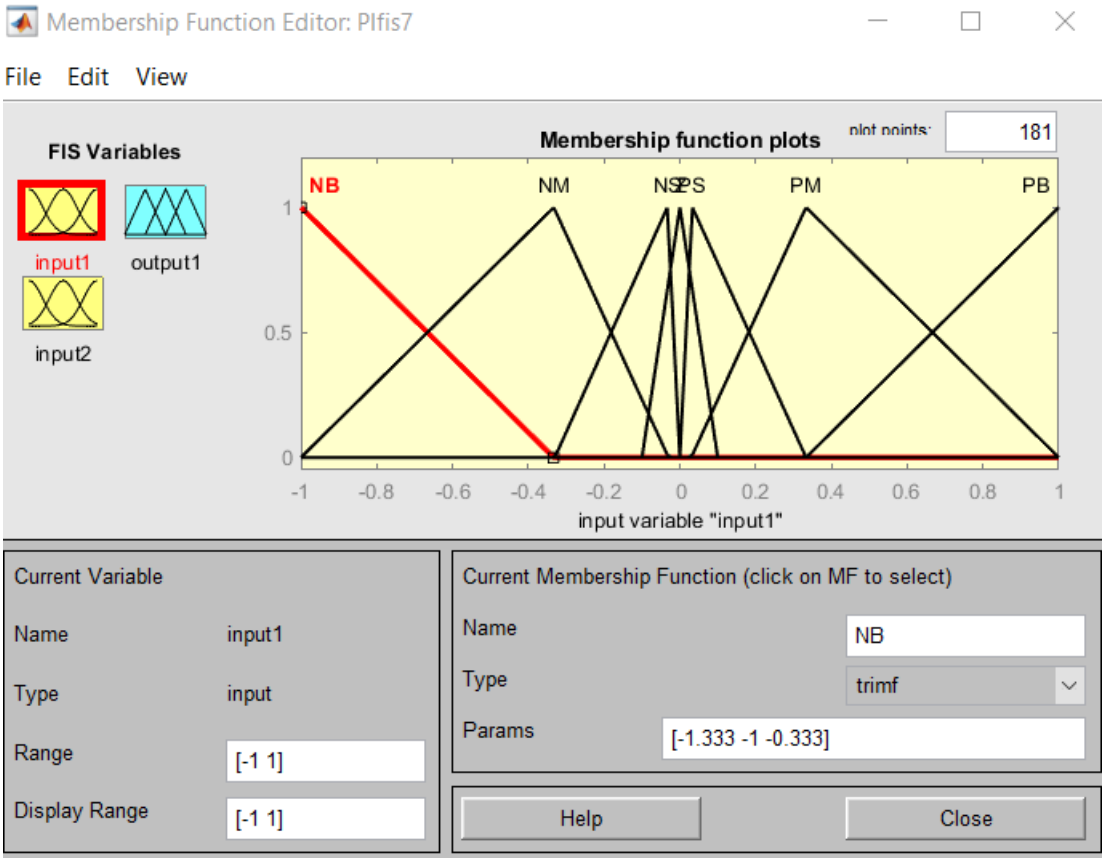


*Рисунок 10.93 – Результат налаштування функцій приналежності вхідних змінних «*Помилка управління»

Терми вхідної лінгвістичної змінної «Інтеграл помилки управління» формуємо рівномірно. Для опису вхідних логічних змінних «Інтеграл помилки управління» у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної трикутну функцію приналежності. З параметрами:

Name = "NBe"; Type = "trimf"; Param = [-1.33 -1 -0.66];
Name = "NMe"; Type = " trimf"; Param = [-1 -0.66 -0.33];
Name = "NSe"; Type = " trimf"; Param = [-0.66 -0.33 0];
Name = "Ze"; Type = " trimf"; Param = [-0.33 0 0.33];
Name = "PSe"; Type = " trimf"; Param = [0 0.33 0.66];
Name = "PMe"; Type = " trimf"; Param = [0.33 0.66 1];
Name = "PBe"; Type = "trimf"; Param = [0.66 1 1.33].

Результат налаштування функцій приналежності вхідних змінних «Інтеграл помилки управління» на рис. 10.94.

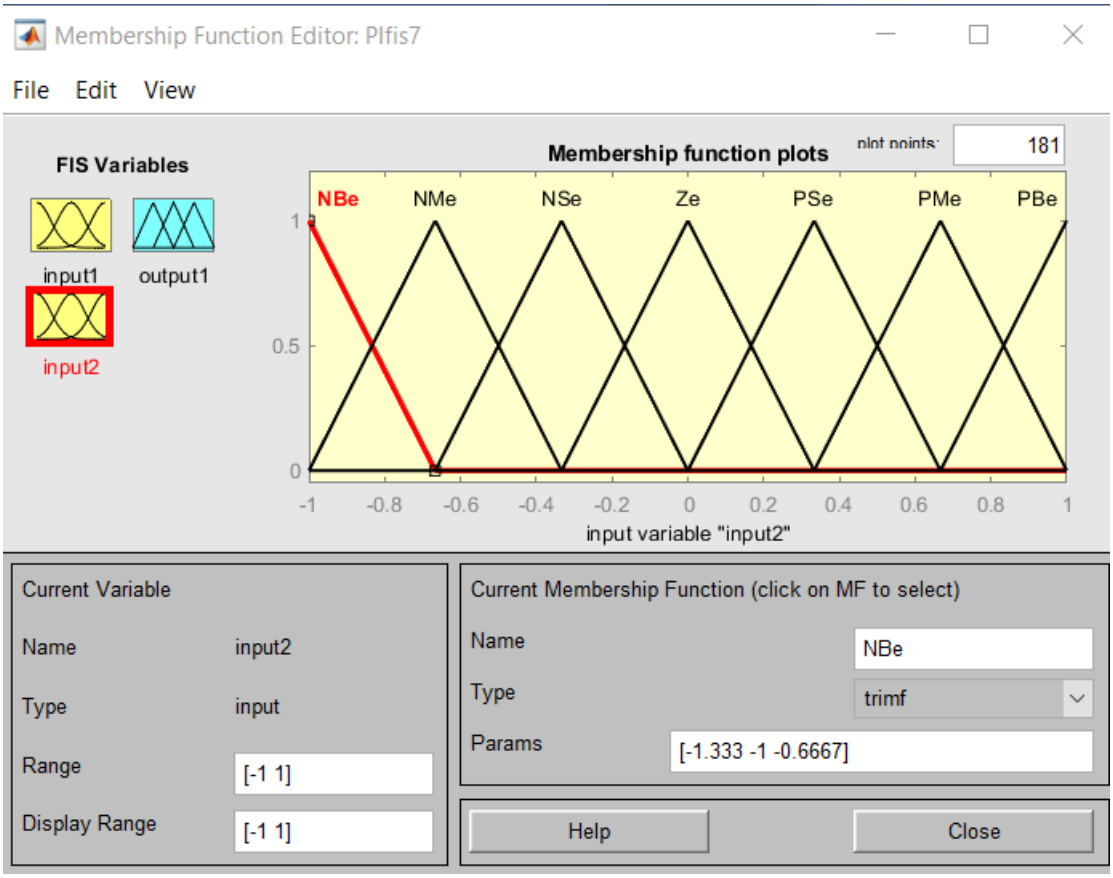


*Рисунок 10.94 – Результат налаштування функцій приналежності вхідних змінних «Інтеграл п*омилки управління»

Для опису вихідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної трикутну функцію приналежності. З параметрами:

Name = "NBu"; Type = "trimf"; Param = [-1.33 -1 -0.66];
Name = "NMu"; Type = " trimf"; Param = [-1 -0.66 -0.33];
Name = "NSu"; Type = " trimf"; Param = [-0.66 -0.33 0];
Name = "Zu"; Type = " trimf"; Param = [-0.33 0 0.33];
Name = "PSu"; Type = " trimf"; Param = [0 0.33 0.66];
Name = "PMu"; Type = " trimf"; Param = [0.33 0.66 1];
Name = "PBu"; Type = "trimf"; Param = [0.66 1 1.33].

Результат налаштування функцій приналежності вхідних змінних наведено на рис. 10.95.

Примітка. У якості центрів термів, що описуються функціями приналежності, обрані значення відповідно до табл. 10.9.

Після опису вхідні та вихідні лінгвістичні змінні, здійснюється опис правил у вікні редактора Rule Editor, кількість правил для даного випадку згідно табл. 10.4 дорівнює 49. Управляючі правила сформуємо згідно з табл. 10.14:

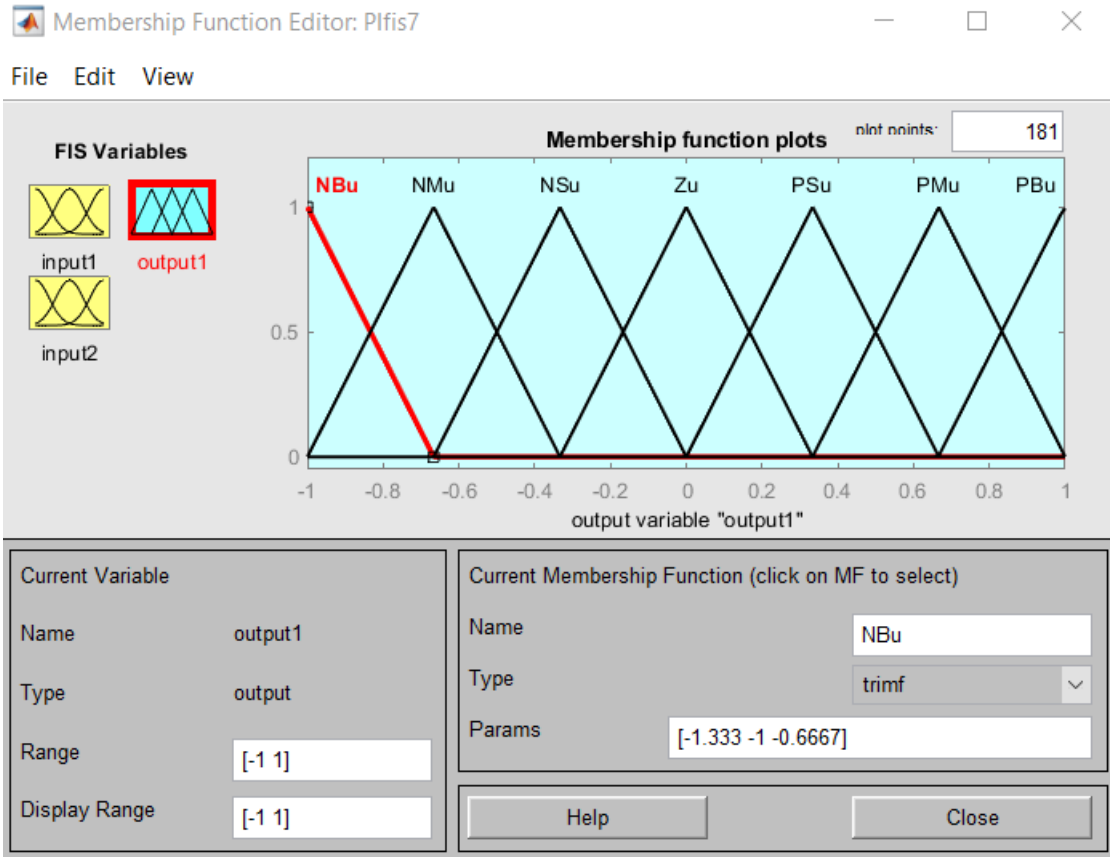


Рисунок 10.95 – Результат налаштування функцій приналежності вихідних змінних

1. if (input1 is NB) and (input2 is NBe) then (output1 is NBu) (1)
2. if (input1 is NB) and (input2 is NMe) then (output1 is NBu) (1)
3. if (input1 is NB) and (input2 is NSe) then (output1 is NBu) (1)
4. if (input1 is NB) and (input2 is Ze) then (output1 is NBu) (1)
5. if (input1 is NB) and (input2 is PSe) then (output1 is NBu) (1)
6. if (input1 is NB) and (input2 is PMe) then (output1 is NBu) (1)
7. if (input1 is NB) and (input2 is PBe) then (output1 is NBu) (1)
8. if (input1 is NM) and (input2 is NBe) then (output1 is NBu) (1)
9. if (input1 is NM) and (input2 is NMe) then (output1 is NBu) (1)
10. if (input1 is NM) and (input2 is NSe) then (output1 is NBu) (1)
11. if (input1 is NM) and (input2 is Ze) then (output1 is NMu) (1)
12. if (input1 is NM) and (input2 is PSe) then (output1 is NMu) (1)
13. if (input1 is NM) and (input2 is PMe) then (output1 is NSu) (1)
14. if (input1 is NM) and (input2 is PBe) then (output1 is Zu) (1)
15. if (input1 is NS) and (input2 is NBe) then (output1 is NBu) (1)
16. if (input1 is NS) and (input2 is NMe) then (output1 is NBu) (1)
17. if (input1 is NS) and (input2 is NSe) then (output1 is NMu) (1)
18. if (input1 is NS) and (input2 is Ze) then (output1 is NSu) (1)
19. if (input1 is NS) and (input2 is PSe) then (output1 is Zu) (1)
20. if (input1 is NS) and (input2 is PMe) then (output1 is PSu) (1)

21. if (input1 is NS) and (input2 is PBe) then (output1 is PMu) (1)
22. if (input1 is Z) and (input2 is NBe) then (output1 is NBu) (1)
23. if (input1 is Z) and (input2 is NMe) then (output1 is NMu) (1)
24. if (input1 is Z) and (input2 is NSe) then (output1 is NSu) (1)
25. if (input1 is Z) and (input2 is Ze) then (output1 is Zu) (1)
26. if (input1 is Z) and (input2 is PSe) then (output1 is PSu) (1)
27. if (input1 is Z) and (input2 is PMe) then (output1 is PMu) (1)
28. if (input1 is Z) and (input2 is PBe) then (output1 is PBu) (1)
29. if (input1 is PS) and (input2 is NBe) then (output1 is NMu) (1)
30. if (input1 is PS) and (input2 is NMe) then (output1 is NSu) (1)
31. if (input1 is PS) and (input2 is NSe) then (output1 is Zu) (1)
32. if (input1 is PS) and (input2 is Ze) then (output1 is PSu) (1)
33. if (input1 is PS) and (input2 is PSe) then (output1 is PMu) (1)
34. if (input1 is PS) and (input2 is PMe) then (output1 is PBu) (1)
35. if (input1 is PS) and (input2 is PBe) then (output1 is PBu) (1)
36. if (input1 is PM) and (input2 is NBe) then (output1 is NMu) (1)
37. if (input1 is PM) and (input2 is NMe) then (output1 is PSu) (1)
38. if (input1 is PM) and (input2 is NSe) then (output1 is PMu) (1)
39. if (input1 is PM) and (input2 is Ze) then (output1 is PMu) (1)
40. if (input1 is PM) and (input2 is PSe) then (output1 is PBu) (1)
41. if (input1 is PM) and (input2 is PMe) then (output1 is PBu) (1)
42. if (input1 is PM) and (input2 is PBe) then (output1 is PBu) (1)
43. if (input1 is PB) and (input2 is NBe) then (output1 is PBu) (1)
44. if (input1 is PB) and (input2 is NMe) then (output1 is PBu) (1)
45. if (input1 is PB) and (input2 is NSe) then (output1 is PBu) (1)
46. if (input1 is PB) and (input2 is Ze) then (output1 is PBu) (1)
47. if (input1 is PB) and (input2 is PSe) then (output1 is PBu) (1)
48. if (input1 is PB) and (input2 is PMe) then (output1 is PBu) (1)
49. if (input1 is PB) and (input2 is PBe) then (output1 is PBu) (1)

Налаштовування управляючих правил для даного випадку наведено на рисунку (див. рис. 10.96).

Посилки у правилах пов'язані (connection) за допомогою операції *and*. Введене правило забезпечене ваговим коефіцієнтом (*Weight*=1).

Після опису правил можливо за допомогою

- інтерфейсу *Rule Viewer* можливо переглянути роботу системи нечіткого висновку за різних вхідних даних (див. рис. 10.97);

− інтерфейсу Surface Viewer можливо переглянути графічне подання закону управління (рис. 10.98).

Після процедур налаштування НЛР_ПІ у редакторі системи нечіткого висновку здійснює запис результатів налаштування у Fuzzy Logic Controller. Для цього у вкладці File здійснюємо Export в математичну модуль нечіткого контролера From Workspase вказавши ім’я Fuzzy Logic Controller.

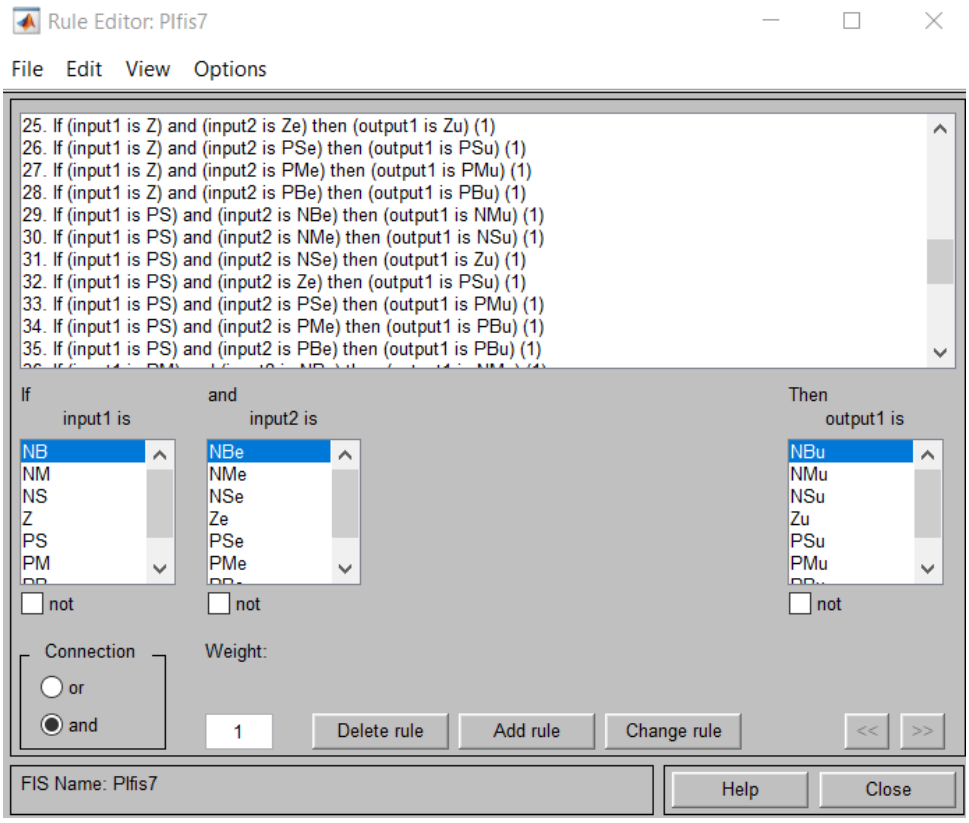


Рисунок 10.96 - Налаштовування опису правил функціонування НЛР_ПІ

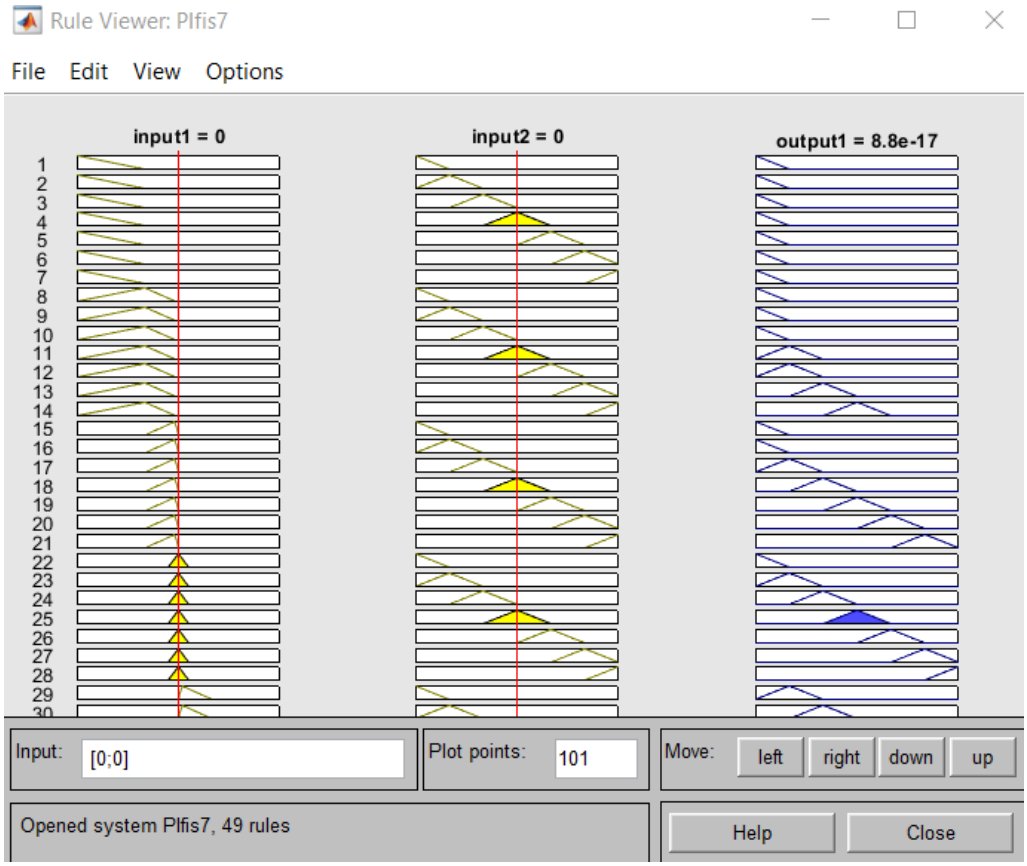


Рисунок 10.97 - Робота системи НЛР_ПІ

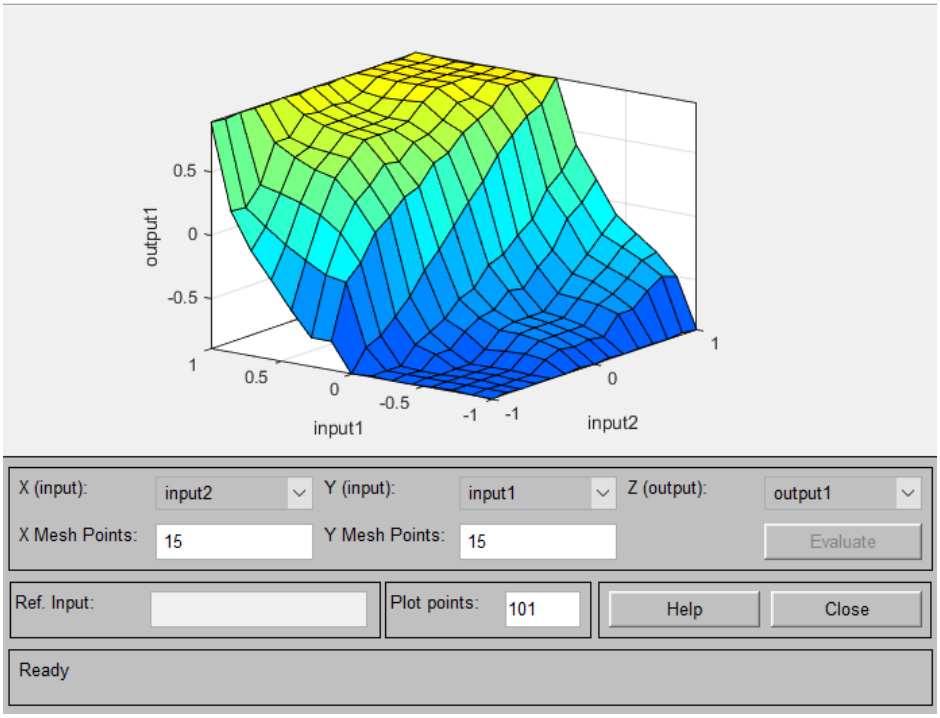


Рисунок 10.98 – Поверхня управління системою нечіткого висновку

Після експорту закону функціонування НЛР_ПІ на математичної моделі системи управління (див. рис. 10.91) получено перехідний процес для синтезованого НЛР_ПІ, який задовольняє поставленому завданню проектування

На рис. 10.99 показаний перехідний процес для синтезованого НЛР_ПІ.

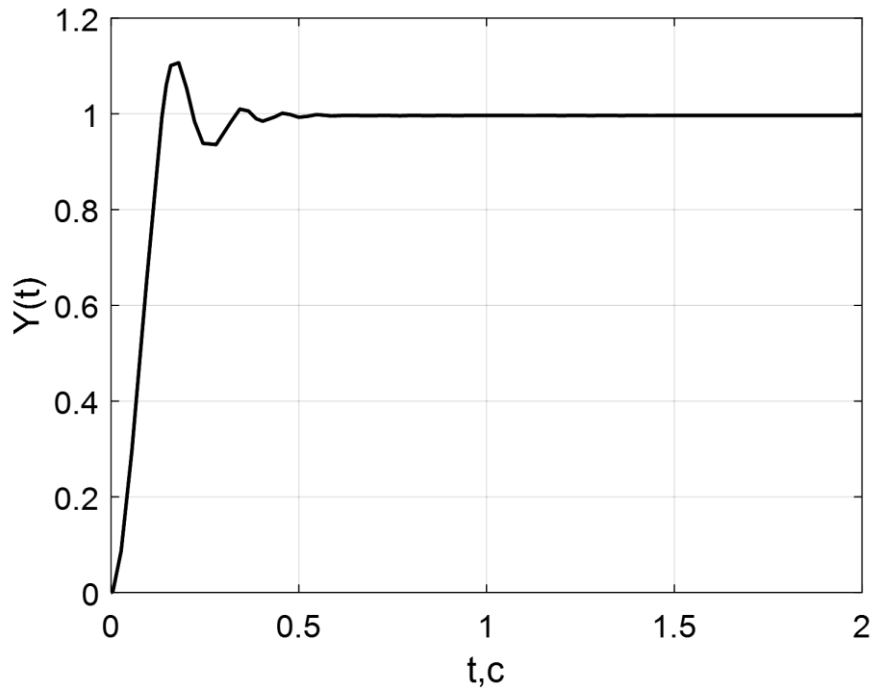


Рисунок 10.99 - Перехідний процес для НЛР_ПІ із сімома правилами

Розглянемо алгоритм покрокового налаштування НЛР _ПІ, який (як і у разі НЛР _ПД) починається з конструювання НЛР_П.
Алгоритм включає такі кроки.

1. Синтезується НЛР _П, що відповідає заданим вимогам до перерегулювання, часу наростання і помилці, що встановилася. Після цього кроку виявляється відомим положення термів лінгвістіческіх пеерменных «Помилка» і «Сигнал управління», а також значення коефіцієнта денормалізації.

$$N_{\Delta e} = \frac{1}{max|\int e(t)dt|}$$

2. Терми лінгвістичної змінної «Інтеграл помилки управління» розташовуються рівномірно за базовою шкалою.

3. Розглядається база правил (табл. 10.14, де замість Δe* використовується ∫e*).

На рис. 10.99 показано результат моделювання. Очевидно, час наростання зменшився майже на порядок при допустимому перерегулюванні і нульовій помилці.

10.15 Синтез нечіткого регулятора ПІД-типу

Розглянемо реалізацію структури НЛР_ПІД рис. 10.32 з використанням синтезованих вище НЛР_ПД та НЛР_ПІ (див. рис. 10.100).

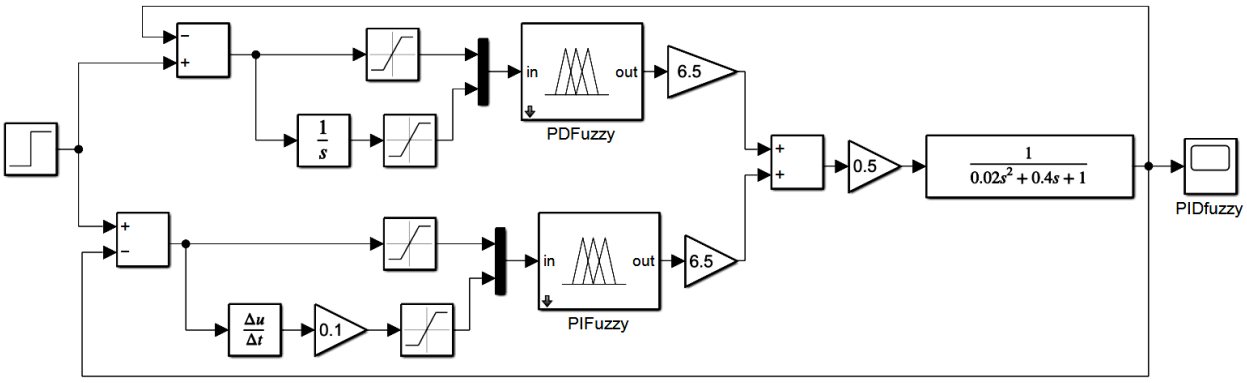


Рисунок 10.100 – Математична модель системи управління з НЛР_ПІД в Simulink MatLab

Налаштування НЛР_ПД та НЛР_ПІ здійснюється згідно методики викладеної вище

На рис. 10.101 показаний перехідний процес у системі.

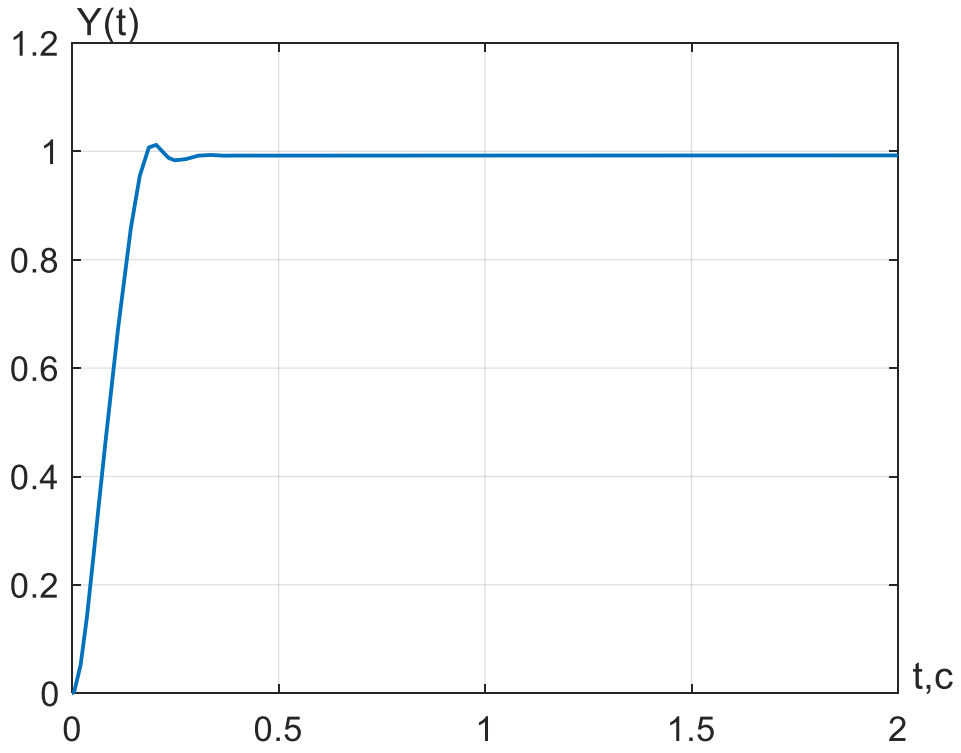


Рисунок 10.101 - Перехідний процес під керуванням НЛР_ПІД

Існують також альтернативні варіанти синтезу НЛР_ПІД, розглянемо деякі з них.

Розглянемо далі реалізацію структури, яка наведена на рис. 10.28, тобто безпосередньо НЛР_ПІД, що отримує на вхід помилку, її похідну та інтеграл.

З метою мінімізації кількості правил будемо для вхідних змінних використовувати по 3 терми (що дає 27 керуючих правил). При цьому кожної змінної відповідає власний масштаб базової шкали *x* (рис. 10.102).

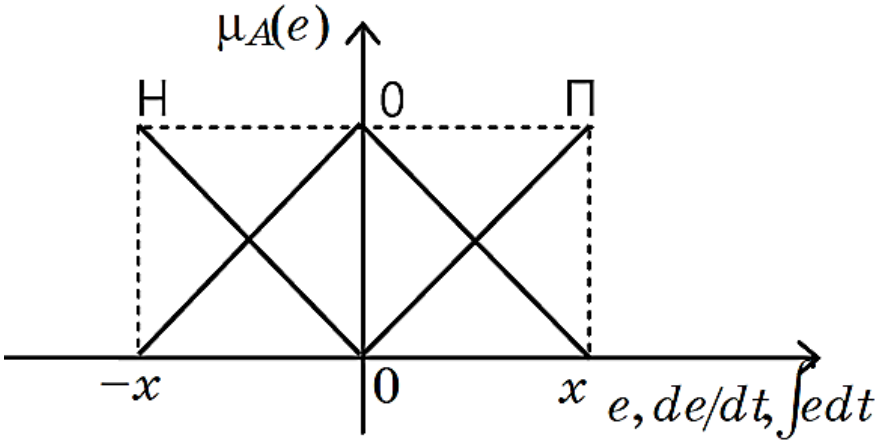


Рисунок 10.102 - Лінгвістичний опис вхідних змінний НЛР_ПІД

Для опису сигналу управління використовуватимемо 5 термів (рис. 1.103).

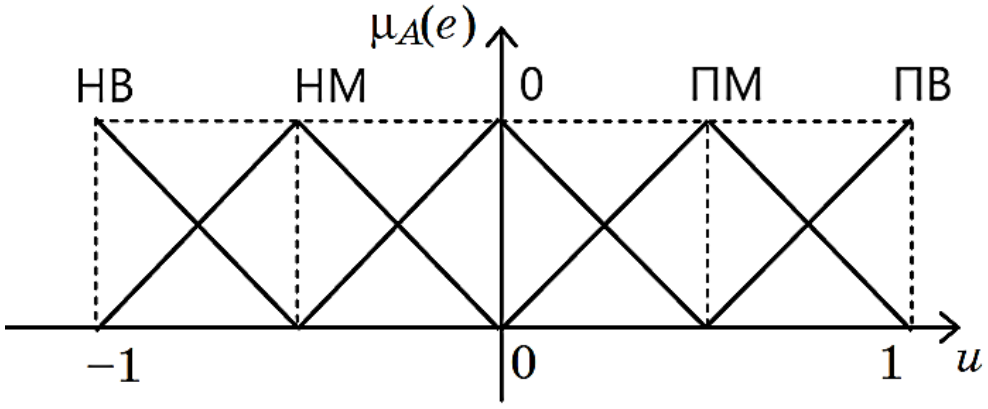


Рисунок 10.103 - Лінгвістичний опис вихідний змінної НЛР_ПІД

Розглянемо проекції поверхні управління НЛР _ПІД на площину *e*×*de*/*dt* за різних лінгвістичних значеннях ∫*edt*.

Варіант 1: "інтеграл помилки" = "0". У цій ситуації коливання відбуваються в області нульової помилки і може бути використана симетрична таблиця лінгвістичних змінних (табл. 10.16).

Таблиця 10.16 – Правила НЛР_ПІД при нульовому інтегралі помилки

Таблиця правил		e*			
		Н	0	П	
Δ e*	Н	НВ	НМ	0	u*
	0	НМ	0	ПМ	
	П	0	ПМ	ПВ	

Варіант 2: "інтеграл помилки" = "П". У цій ситуації керована змінна може бути в області позитивної помилки, тоді потрібен сигнал корекції, величина якого пропорційна зростанню помилки. Якщо ж інтеграл помилки позитивний, а помилка негативна і зменшується, то корекція нульова (табл. 10.17).

Таблиця 10.17 - Правила НЛР_ПІД при позитивному інтегралі помилки

Таблиця правил		e*		
		Н	0	П
Δ e*	Н	0	ПМ	ПВ
	0	ПМ	ПВ	ПВ
	П	ПВ	ПВ	ПВ

u*

Варіант 3: "інтеграл помилки" = "Н". Цей варіант виявляється симетричним до попереднього варіанту (табл. 10.18).

Таблиця 10.18 - Правила НЛР_ПІД при негативному інтегралі помилки

Таблиця правил		e*		
		Н	0	П
Δ e*	Н	НВ	НВ	НВ
	0	НВ	НВ	НМ
	П	НВ	НВ	0

u*

Загальна таблиця правил набуває наступного вигляду (табл. 10.19), вона містить 27 правил, і кожне правило має три посилання.

На рис. 10.104 показана структурна схема НЛР_ПІД Simulink MatLab,

У схемі використаний інтегратор з насиченням, інакше НЛР_ПІД вироджується в НЛР_ПД.

Таблиця 10.19 – Правила управління НЛР_ПІД

№	∫*edt*	e	*de/dt*	*u*	№	∫*edt*	e	*de/dt*	*u*
1	0	Н	Н	НВ	15	П	0	П	ПВ
2	0	Н	0	НМ	16	П	П	0	ПВ
3	0	Н	П	0	17	П	П	Н	ПВ
4	0	0	Н	НМ	18	П	П	П	ПВ
5	0	0	0	0	19	Н	Н	0	НВ
6	0	0	П	ПМ	20	Н	Н	Н	НВ
7	0	П	Н	0	21	Н	Н	П	НВ
8	0	П	0	ПМ	22	Н	0	0	НВ
9	0	П	П	ПВ	23	Н	0	Н	НВ
10	П	Н	Н	0	24	Н	0	П	НИ
11	П	Н	0	ПМ	25	Н	П	0	НВ
12	П	Н	П	ПВ	26	Н	П	Н	НМ
13	П	0	Н	ПМ	27	Н	П	П	0
14	П	0	0	ПВ					

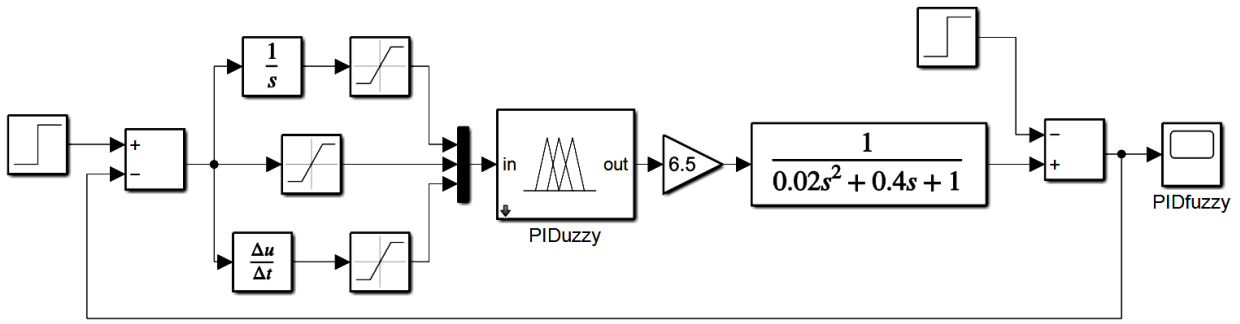


Рисунок 10.104 – Структурна схема математичної моделі об'єкта управління з НЛР_ПІД в Simulink MatLab

Приклад. Відповідно до рис. 10.104 сформуємо в Simulink MatLab модель системи управління з НЛР ПІД при *x* = 6.5 (рис. 11.90).

Fuzzy Logic Controller надаємо Fis name: PID2fis (ім'я може буди будь-яке).

Викликається редактор системи нечіткого висновку (Fuzzy Inference System – FIS) у MatLab командою

>> fuzzy

Дале проводиться налаштовування головного вікна редактора нечіткої логічної системи (див. рис. 10.105.).

У *FIS editor* задається опис систему нечіткого логічного висновку *Mamdani*. Для створюваної системи обирається вид логічного зв'язку (*And method – min*) та (*Or method – max*), вид імплікації (*Implication – min*), спосіб агрегування висновків правил (*Aggregation – max*) та метод дефазифікації (*Defuzzification – centroid*).

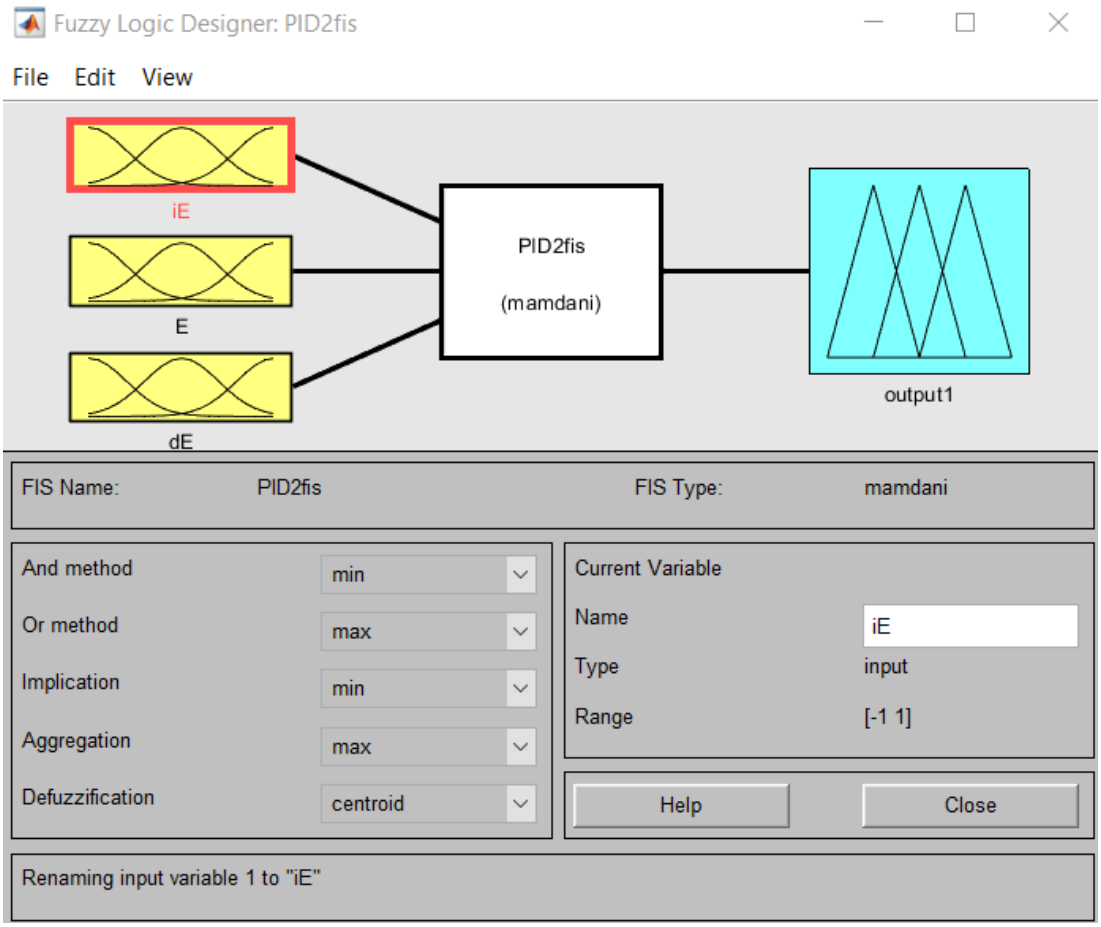


Рисунок 10.105 – Налаштування інтерфейсу FIS editor

У меню *Edit* послідовно додаємо 3 вхідних змінних з розміром базової шкали (Range= [-1 1]) та вихідну змінну з розміром базової шкали (*Range*= [-1 1]).

Для опису вхідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної змінної трикутну функцію приналежності.

Терми лінгвістичної змінної «Інтеграл помилки управління» розміщуються відповідно до рис. 11.116 з параметрами:

Name = "NiE"; Type = "trimf"; Param = [-2 -1 0];
Name = "ZiE"; Type = " trimf"; Param = [-1 0 1];
Name = "PiE"; Type = "trimf"; Param = [0 1 2].

Результат налаштування функцій приналежності вхідних змінних «Інтеграл помилки управління» наведено на рис. 10.106.

Терми вхідної лінгвістичної змінної «Помилка управління» формуємо рівномірно. Для опису вхідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної трикутну функцію приналежності. З параметрами:

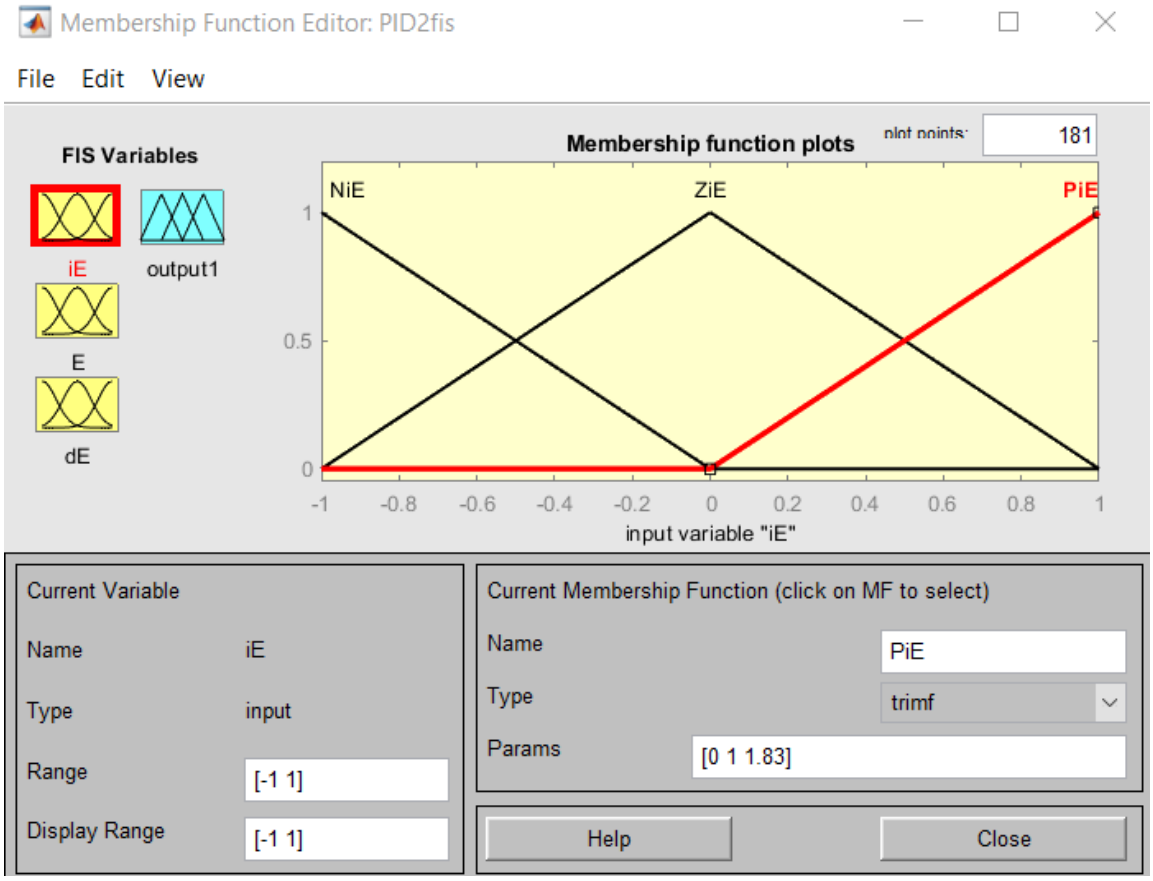


Рисунок 10.106 – Результат налаштування функцій приналежності вхідних змінних «Інтеграл помилки управління»

Name = "NE"; Type = "trimf"; Param = [-2 -1 0];
Name = "ZE"; Type = " trimf"; Param = [-1 0 1];
Name = "PE"; Type = "trimf"; Param = [0 1 2].

Результат налаштування функцій приналежності вхідних змінних «Помилка управління» на рис. 10.107.

Терми вхідної лінгвістичної змінної «Похідна помилки управління» формуємо рівномірно. Для опису вхідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної трикутну функцію приналежності. З параметрами:

Name = "NdE"; Type = "trimf"; Param = [-2 -1 0];
Name = "ZdE"; Type = " trimf"; Param = [-1 0 1];
Name = "PdE"; Type = "trimf"; Param = [0 1 2].

Результат налаштування функцій приналежності вхідних змінних «Похідна помилки управління» на рис. 10.108.

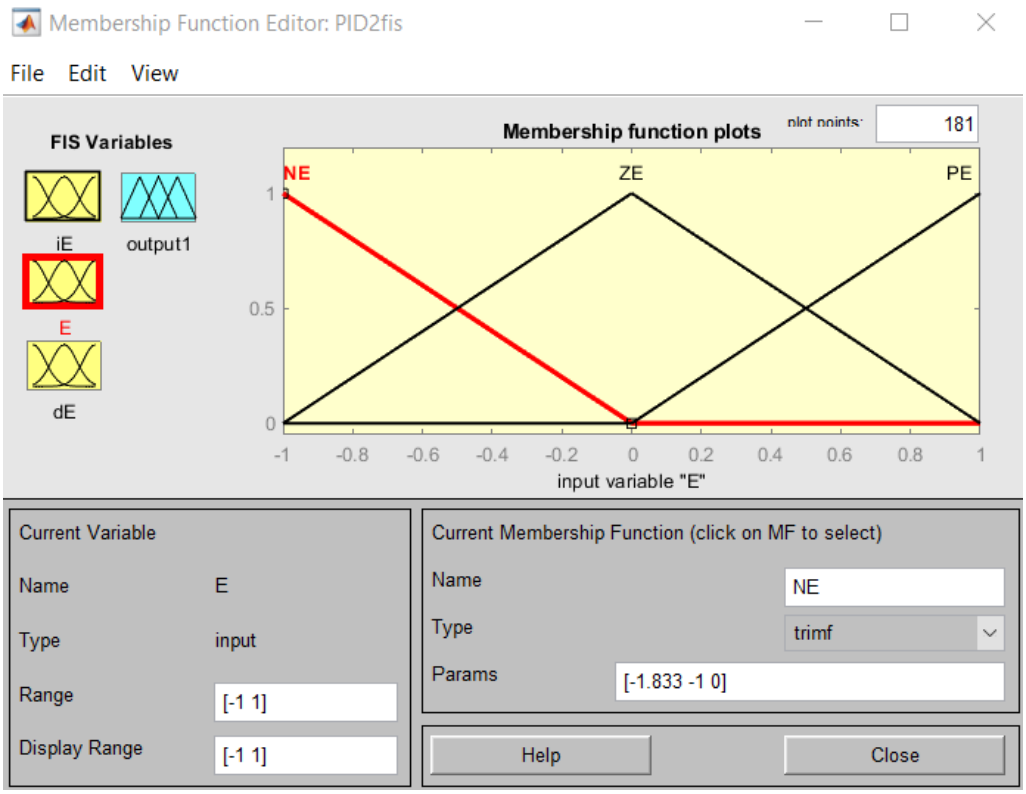


Рисунок 10.107 – Результат налаштування функцій приналежності вхідних змінних «Помилка управління»

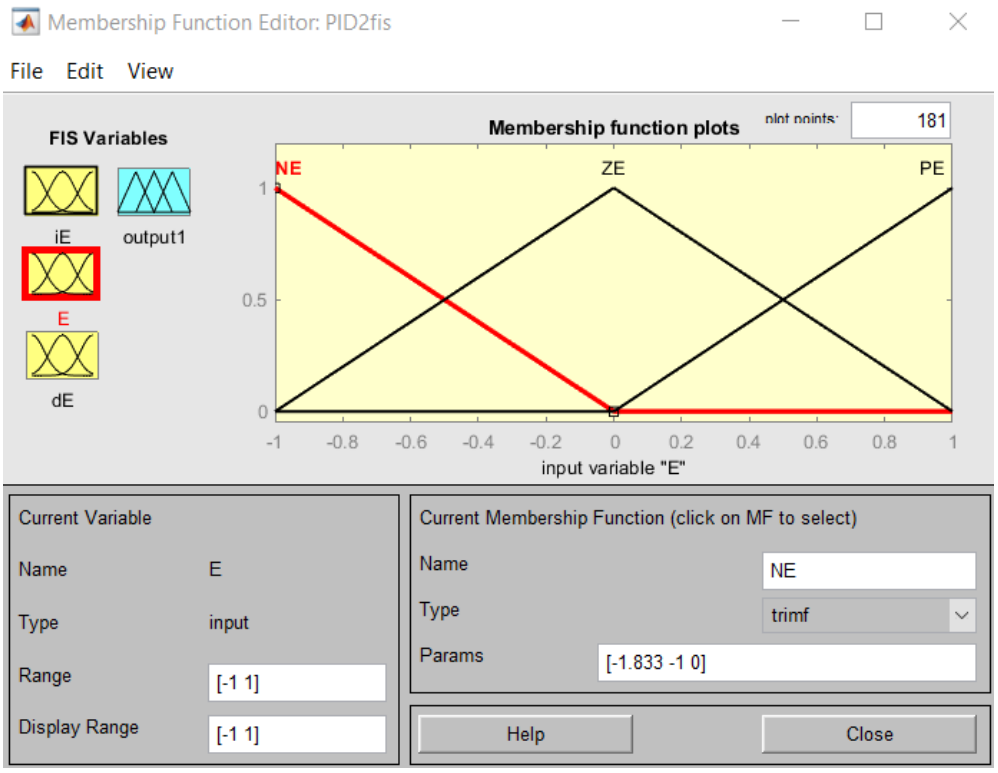


Рисунок 10.108 – Результат налаштування функцій приналежності вхідних змінних «Похідна помилки управління»

Для опису вихідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної вихідної змінної трикутну функцію приналежності. 3 параметрами:

Name = "NBu"; Type = "trimf"; Param = [-1.5 -1 -0.5];
Name = "NSu"; Type = " trimf"; Param = [0 -0.5 0];
Name = "Zu"; Type = " trimf"; Param = [-0.5 0 0.5];
Name = "PSu"; Type = " trimf"; Param = [0 0.5 0.1];
Name = "PBu"; Type = "trimf"; Param = [0.5 1.5 1.5].

Результат налаштування функцій приналежності вхідних змінних наведено на рис. 10.109.

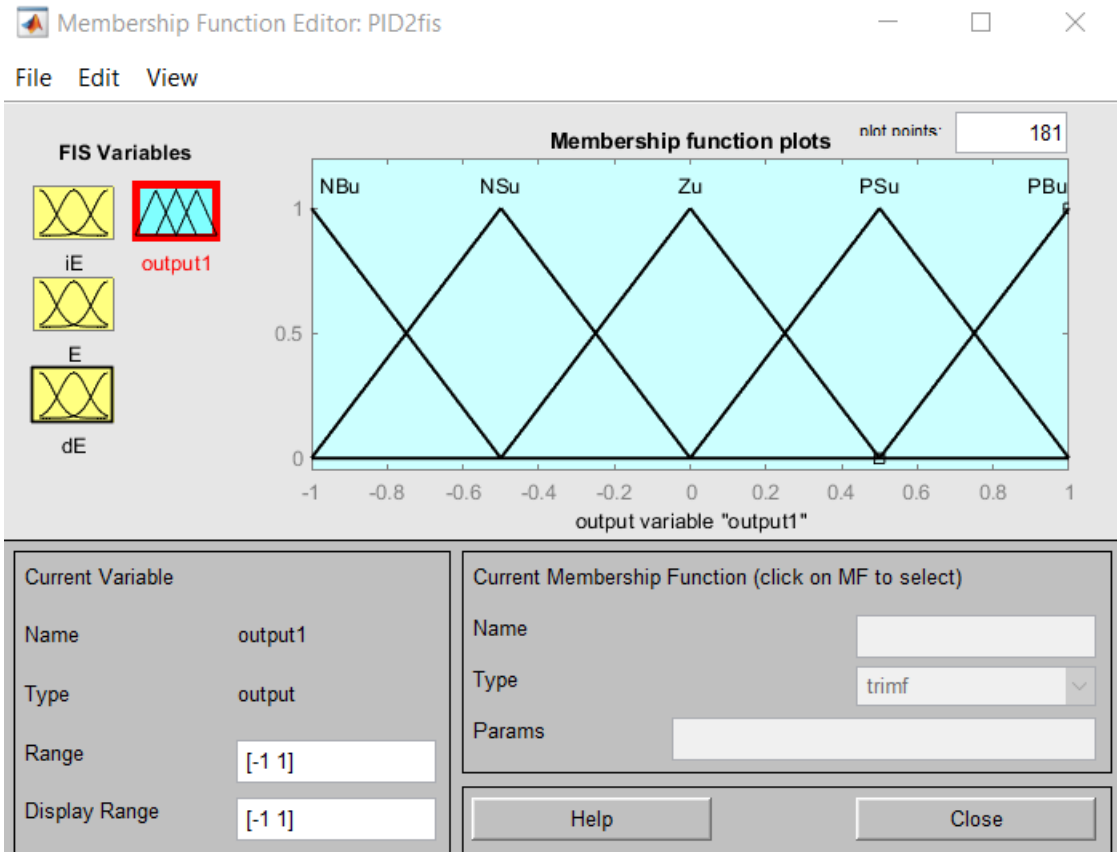


Рисунок 10.109 – Результат налаштування функцій приналежності вихідних змінних

Після опису вхідних та вихідних лінгвістичних змінних, здійснюється опис правил у вікні редактора Rule Editor, кількість правил для даного випадку дорівнює 27. Управляючі правила сформуємо згідно з табл. 10.19:

1. if (iE is ZiE) and (E is NE) and (dE is NdE) then (output1 is NBu) (1)
2. if (iE is ZiE) and (E is NE) and (dE is ZdE) then (output1 is NSu) (1)
3. if (iE is ZiE) and (E is NE) and (dE is PdE) then (output1 is Zu) (1)

4. if (iE is ZiE) and (E is ZE) and (dE is NdE) then (output1 is NSu) (1)
5. if (iE is ZiE) and (E is ZE) and (dE is ZdE) then (output1 is Zu) (1)
6. if (iE is ZiE) and (E is ZE) and (dE is PdE) then (output1 is PSu) (1)
7. if (iE is ZiE) and (E is PE) and (dE is NdE) then (output1 is Zu) (1)
8. if (iE is ZiE) and (E is PE) and (dE is ZdE) then (output1 is PSu) (1)
9. if (iE is ZiE) and (E is PE) and (dE is PdE) then (output1 is PBu) (1)
10. if (iE is PiE) and (E is NE) and (dE is NdE) then (output1 is Zu) (1)
11. if (iE is PiE) and (E is NE) and (dE is ZdE) then (output1 is PSu) (1)
12. if (iE is PiE) and (E is NE) and (dE is PdE) then (output1 is PBu) (1)
13. if (iE is PiE) and (E is ZE) and (dE is NdE) then (output1 is PSu) (1)
14. if (iE is PiE) and (E is ZE) and (dE is ZdE) then (output1 is PBu) (1)
15. if (iE is PiE) and (E is ZE) and (dE is PdE) then (output1 is PBu) (1)
16. if (iE is PiE) and (E is PE) and (dE is NdE) then (output1 is PBu) (1)
17. if (iE is PiE) and (E is PE) and (dE is ZdE) then (output1 is PBu) (1)
18. if (iE is PiE) and (E is PE) and (dE is PdE) then (output1 is PBu) (1)
19. if (iE is NiE) and (E is NE) and (dE is NdE) then (output1 is NBu) (1)
20. if (iE is NiE) and (E is NE) and (dE is ZdE) then (output1 is NBu) (1)
21. if (iE is NiE) and (E is NE) and (dE is PdE) then (output1 is NBu) (1)
22. if (iE is NiE) and (E is ZE) and (dE is NdE) then (output1 is NBu) (1)
23. if (iE is NiE) and (E is ZE) and (dE is ZdE) then (output1 is NBu) (1)
24. if (iE is NiE) and (E is ZE) and (dE is PdE) then (output1 is NSu) (1)
25. if (iE is NiE) and (E is PE) and (dE is NdE) then (output1 is NBu) (1)
26. if (iE is NiE) and (E is PE) and (dE is ZdE) then (output1 is NSu) (1)
27. if (iE is NiE) and (E is PE) and (dE is PdE) then (output1 is Zu) (1)

Налаштовування управляючих правил для даного випадку наведено на рисунку (див. рис. 10.110).

Посилки у правилах пов'язані (connection) за допомогою операції *and*. Введене правило забезпечене ваговим коефіцієнтом (*Weight*=1). Після опису правил можливо за допомогою

- інтерфейсу *Rule Viewer* можливо переглянути роботу системи нечіткого висновку за різних вхідних даних (див. рис. 10.111).

Після процедур налаштування НЛР_ПІД у редакторі системи нечіткого висновку здійснює запис результатів налаштування у Fuzzy Logic Controller. Для цього у вкладці File здійснюємо Export в математичну модуль нечіткого контролера From Workspase вказавши ім’я Fuzzy Logic Controller.

Після експорту закону функціонування НЛР_ПІД на математичної моделі системи управління (див. рис. 10.104) получено перехідний процес для системи управління з НЛР_ПІД, який задовольняє поставленому завданню проектування

На рис. 10.112 показаний перехідний процес для синтезованого НЛР_ПІД з додавання збурення на 10с. Згідно графіку перехідного процесу можливо зробити висновок, що спроектована система управління повністю компенсує збурюючи впливи.

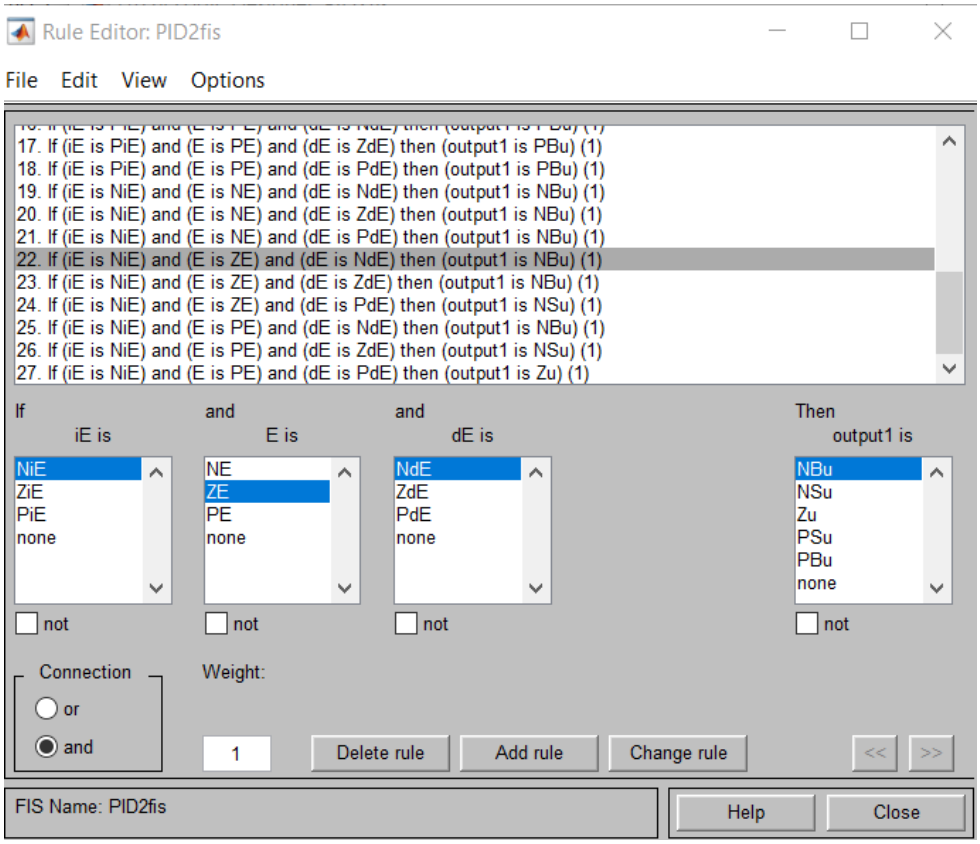


Рисунок 10.110 - Налаштовування опису правил функціонування НЛР_ПІД

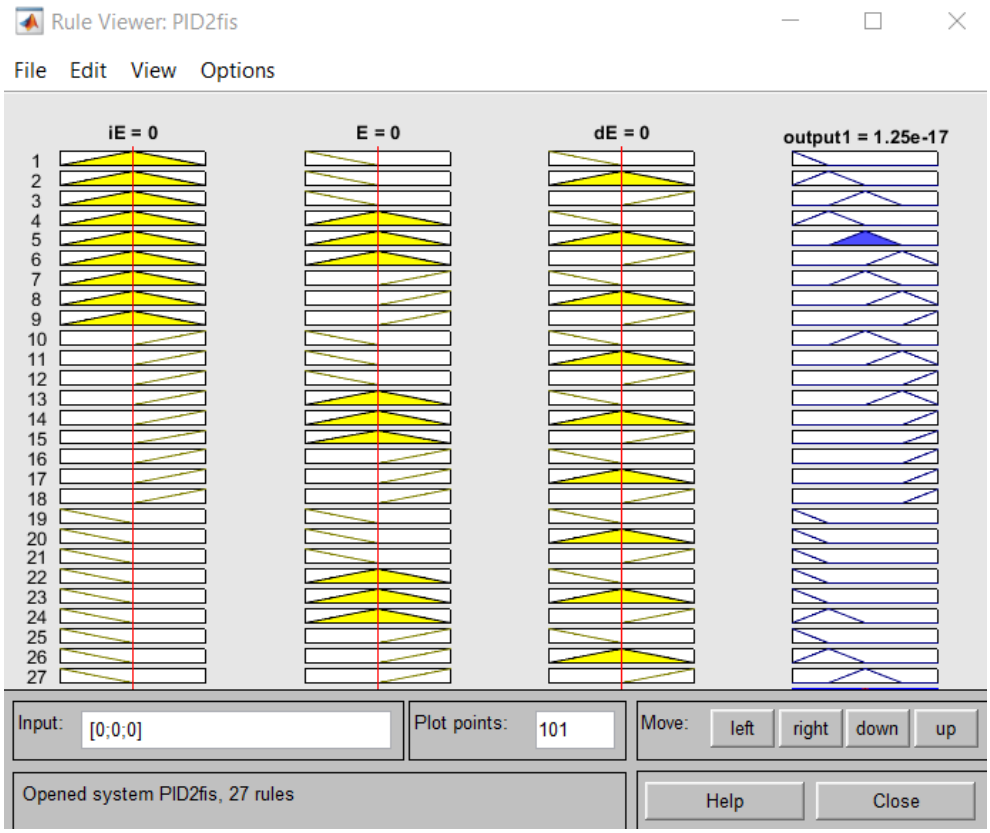


Рисунок 10.111 - Робота системи НЛР_ПІД

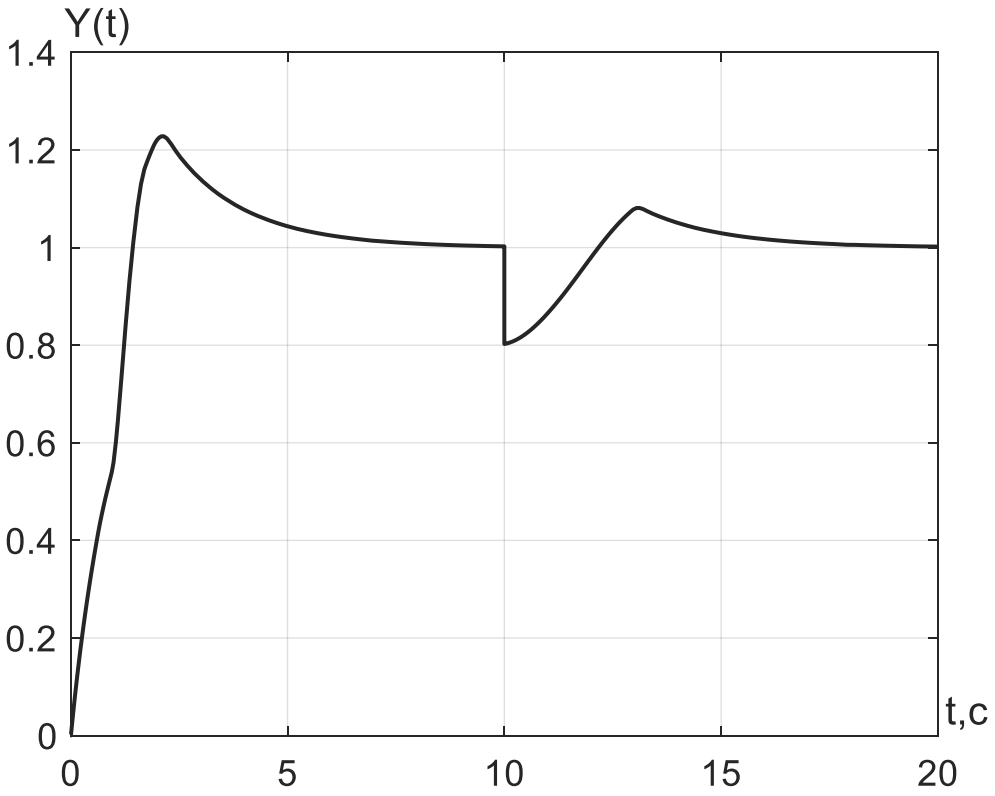


Рисунок 10.112 - Перехідний процес системи управління з НЛР_ПІД

11 СИНТЕЗ НЕЧІТКИХ ПРАВИЛ З ДАНИХ ПРО ПРОЦЕС УПРАВЛІННЯ

11.1 Постановка задачі синтезу

Основою нечіткого логічного регулятора є нечіткі керуючі правила, які пов'язують поточний стан об'єкта та управління, що відповідає даному стану. Залежно від типу нечіткого логічного регулятора нечіткі правила мають одну, дві чи три посилки.

Для синтезу нечіткого логічного регулятора необхідно вирішити дві основні задачі:

− описати лінгвістичні змінні, що відповідають входам та виходам нечіткого логічного регулятора (цей опис включає вибір коефіцієнтів масштабування);

− вибрати правила управління.

Ці завдання можуть бути вирішені методом спроб та помилок в результаті процедури інтерактивного моделювання. Вибирається деякий початковий опис лінгвістичних змінних і початковий набір керуючих правил, потім відбувається моделювання перехідного процесу корекція вихідного опису залежно від помилок. Однак цей шлях може бути досить трудомістким через існуючу на початковому етапі значну невизначеність про величину базових множин лінгвістичних змінних, про необхідну кількість правил, і про те, які терми повинні входити в те чи інше правило.

Полегшити вирішення завдання налаштування нечіткого логічного регулятора може підхід, заснований на використанні еталонної траєкторії руху об'єкта.

Сутність цієї методики полягає у переході від представлення процесу управління в часі, до подання у фазовому просторі з подальшою апроксимацією фазової траєкторії набором нечітких правил.

Припустимо, що є набір еталонних траєкторій руху об'єкта, тобто набір числових значень, що описують входи та виходи регулятора у різні моменти часу. Цей набір можна отримати при керуванні за допомогою класичного регулятора (ПІД, модального тощо). Обробляючи стандартні траєкторії, можна отримати набір керуючих нечітких правил.

Виникає природне питання: навіщо синтезувати нечіткий логічний регулятор, якщо вже є регулятор? При переході від лінійного регулятора до нечіткого логічного регулятора можна досягти таких позитивних ефектів:

− використання нечіткого логічного регулятора може підвищити завадостійкість системи управління;

− застосування нечіткого логічного регулятора може розширити діапазон регулювання за рахунок використання нелінійної керуючої функції;

− отриманий набір правил може бути розширений для надання регулятору нелінійних властивостей.

Сформульована задача отримання керуючих правил даних, що описують процес управління, належить data mining (розробка даних) – розділу технічного штучного інтелекту, присвяченому вилученню семантично значимих закономірностей з необробленої «сирою» інформації.

11.2 Поняття кластеризації

Кластеризація - розбиття об'єктів на групи, так що відстань між об'єктами кожної групи менша, ніж відстань від будь-якого з об'єктів до будь-якої іншої групи.

Зауважимо, що завдання кластеризації відрізняється від завдання класифікації. Класифікація передбачає віднесення нового об'єкта до одного з вже відомих класів, а кластеризація передбачає формування класів (кластерів) з багатьох відомих об'єктів.

Завдання кластеризації може ставитися як заздалегідь заданому, і заздалегідь невідомому кількості кластерів.

Елементи одного кластера повинні бути так близькі один одному, як це тільки можливо, і, одночасно, кластери повинні бути на найбільшому віддаленні один від одного. Для забезпечення керованості процесу кластеризації необхідно використовувати міру близькості, якою зазвичай визначають відстань d між двома об'єктами (точками в n-мірному просторі) X_k та X_l у вигляді речової функції

$$d: X \times X \to R^+$$

для якої виконується умова

$$d(X_k, X_l) \geq 0;$$
$$d(X_k, X_l) = 0 \Rightarrow X_k = X_l;$$
$$d(X_k, X_l) = d(X_l, X_k);$$

Якщо додатково виконується нерівність трикутника:

$$d(X_k, X_l) \leq d\left(X_k, X_j\right) + d\left(X_j, X_l\right),$$

то функція відстані є метрикою (хоча це не завжди необхідно для завдання кластеризації).

Завдання кластеризації формулюється в такий спосіб.

Дано N об'єктів, що підлягають кластеризації, кожен об'єкт володіє n ознаками, так що величина

$$x_{ij}; i = \overline{1,N}; j = \overline{1,n}$$

являє значення *j*-ї ознаки *i*-го об'єкта. Таким чином, кожен *i*-й об'єкт являє собою точку в *n*-вимірному просторі ознак, задану набором координат (вектором):

$$X_i = [x_{i1} \quad x_{i2} \quad \cdots \quad x_{in}].$$

Уся сукупність об'єктів, що підлягають кластеризації, може бути описана матрицею

$$X = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{N1} & x_{N2} & \ldots & x_{Nn} \end{bmatrix}.$$

Нехай завдано *C* – кількість кластерів. Для кожного кластера *A*і повинні виконуватись умови:

$$A_i, i = \overline{1,C}; 2 \le C \le N,$$
$$\bigcup_{i=\overline{1,C}} A_i = X,$$
$$A_i \cap A_j = 0,, i, j = \overline{1,C}, i \ne j.$$

При цьому жоден кластер не може бути порожнім або містити всі об'єкти.

$$0 \ne A_i \ne X,, i = \overline{1,C}.$$

Розглянемо «чітку» кластеризацію. Тут потрібно використовувати бінарну функцію приналежності

$$b_{ki} = \begin{cases} 1, X_k \in A_i, \\ 1, X_k \notin A_i. \end{cases} \qquad (11.1)$$

Для значення приналежності виконуються умови:

$$\sum_{i=1}^{C} b_{ki} = 1, k = \overline{1,N},$$
$$\sum_{k=1}^{N} b_{ki} \in [0, N], i = \overline{1,C}.$$

Перша умова означає, що кожен об'єкт належить лише одному

кластеру. Друга умова означає, що кількість об'єктів, що входять до кластера, змінюється від 0 до *N*.

11.3 Алгоритм С-середніх

Використовуючи функцію приналежності, можливо вирішити задачу кластеризації за допомогою опису кластерів матрицею розбиття:

$$B = [b_{ki}], b_{ki} \in \{0,1\}, k = \overline{1,N}, i = \overline{1,C}. \qquad (11.2)$$

Для того, щоб кластер не був порожнім і не містив усі об'єкти, можна поставити умову:

$$0 < \sum_{k=1}^{N} b_{ki} < N; \;\; i = \overline{1,C}$$

Для оцінки якості розбиття використовують критерій розкиду, що показує суму відстаней від об'єктів до центру свого кластера. Для евклідового простору цей критерій записується так

$$\sum_{i=\overline{1,C}} \sum_{X_k \in A_i} \|V_i - X_k\|^2 \to min, \qquad (11.3)$$

де X_k – об’єкт кластеризації; A_i – *i*-й кластер; V_i – центр кластера;

$$A_i = \{X_k\}, b_{ki} = 1,, k = \overline{1,N},$$

$$V_i = \frac{1}{|A_i|} \sum_{X_k \in A_i} X_k,$$

де $|A_i|$ - потужність множини, так що компоненти V_i виходять як усереднення по всіх елементах кластера.

Кластеризацію об'єктів за допомогою алгоритму *С*-середніх можна сформулювати як задачу оптимізації: знайти матрицю виду (11.2), що мінімізує значення критерію (11.3), тобто потрібно вибрати центри кластерів таким чином, щоб мінімізувати відхилення елементів кластеру від його центру.

Дискретний характер чіткого розбиття призводить до труднощів знаходження оптимальної кластеризації через негладкість цільової функції.

При нечіткому С-розбиття будь-який об'єкт одночасно може належати до різних кластерів, але з різним ступенем.

Для цього бінарна функція приналежності виду (11.1) замінюється на безперервну функцію приналежності

$$\mu_{ki} \in [0,1]$$

так що нечітким С-розбиттям (або матрицею ступенів приналежності) називається матриця

$$M = [\mu_{ki}], \qquad k = \overline{1,N}, i = \overline{1,C}.$$

Таким чином, єдиною відмінністю матриць *B* та *M* є те, що при нечіткому розбитті ступінь приналежності об'єкта до кластера приймає значення з інтервалу [0,1], а при чіткому - тільки з множини {0,1}.

На матрицю *M* можуть бути накладені обмеження:

1) сума приналежності об'єкта до різних кластерів дорівнює 1

$$\sum_{i=\overline{1,C}} \mu_{ki} = 1, k = \overline{1,N}; \qquad (11.4)$$

2) кластер не може бути пустим або містити усі об’єкти

$$0 < \sum_{k=1}^{N} \mu_{ki} < N; \;\; i = \overline{1,C}. \qquad (11.5)$$

Перша умова визначає нечітке розбиття базової множини об'єктів на безліч кластерів.

Якщо об'єкти розташовані межі двох кластерів, їм призначають ступень приналежності рівний 0.5. Нестача нечіткого розбиття проявляється під час роботи з об'єктами, віддаленими від центрів всіх кластерів. Видалені об'єкти мають мало спільного з будь-яким із кластерів, але за умовою (11.4) сума їх ступенів приналежності така сама, як і для об'єктів, близьких до центрів кластерів, тобто. дорівнює 1.

Для усунення цього недоліку замість (11.4) можна використовувати іншу умову, яка вимагає, щоб довільний об'єкт належав хоча б одному кластеру:

$$\exists i, \mu_{ki} > 0, \forall k.$$

Позначимо центри кластерів

$$V_i = (v_{i1}, v_{i2}, \dots, v_{in}), \qquad i = \overline{1,C}.$$

При використанні евклідової відстані завдання нечіткої

кластеризації полягає у знаходженні такої матриці ступенів приналежності *M* та таких координат центрів кластерів $V = (V_1, V_2, \dots, V_c)$, які забезпечують мінімум наступного критерію:

$$\sum_{i=1}^{C}\sum_{k=1}^{N}(\mu_{ki})^m \|X_k - V_i\|^2 \to min, \qquad (11.6)$$

$$V_i = \frac{1}{\sum_{k=1}^{N}\mu_{ki}}\sum_{k=1}^{N}((\mu_{ki})^m X_k), \qquad (11.7)$$

де *m* - експонентна вага, яка встановлюється до початку кластеризації.

Експонентна вага дозволяє при формуванні координат центрів кластерів посилити вплив об'єктів з великими значеннями ступенів приналежності та зменшити вплив об'єктів з малими значеннями ступенів приналежності. Зазвичай встановлюють *m* = 2.

Критерій (11.6) визначає суму всіх зважених відстаней від об'єктів до центрів кластерів. Центр кластера (11.7) виходить шляхом усереднення компонентів векторів, що входять до кластеру, з урахуванням значень приналежності та експоненційної ваги.

Знаходження матриці нечіткого розбиття *M* з мінімальним значенням критерію (11.6) є завданням нелінійної оптимізації, яка може бути вирішена різними методами.

Найбільш відомий і найчастіше застосовуваний метод розв'язання цього завдання алгоритм нечітких *C*-середніх, основою якого покладено метод невизначених множників Лагранжа. Він дозволяє знайти локальний оптимум, тому виконання алгоритму різних початкових точок може призвести до різних результатів.

Алгоритм нечітких *C*-середніх зводиться до наступної послідовності кроків.

1. Встановити параметри алгоритму: *C* – кількість кластерів; *m* – експоненційна вага; *e* – параметр зупинки алгоритму.

2. Випадково згенерувати матрицю *M*.

3. Розрахувати центри кластерів:

$$V_i = \frac{\sum_{k=\overline{1,N}}^{N}((\mu_{ki})^m X_k)}{\sum_{k=\overline{1,N}}^{N}(\mu_{ki})^m}, i = \overline{1, C}.$$

4. Розрахувати відстані між об'єктами з *X* та центрами кластерів:

$$d_{ki} = \sqrt{\|X_k - V_i\|^2}; k = \overline{1, N};\ i = \overline{1, C}.$$

5. Перерахувати елементи матриці нечіткого розбиття

$$k = \overline{1,N};\ i = \overline{1,C};$$

$$d_{ki} > 0 \Rightarrow \mu_{ki} = \frac{1}{\left(d_{ki}^2 \sum_{j=\overline{1,C}} \frac{1}{d_{ki}^2}\right)^{\frac{1}{m-1}}};$$

$$d_{ki} = 0 \Rightarrow \mu_{ki} = \begin{cases} 1, & j = i \\ 0, j \neq i, j = \overline{1,C} \end{cases}$$

Таким чином, нове значення приналежності зворотно пропорційно відношенню квадрата відстані від об'єкта до центру *i*-го кластера к суми квадратів відстаней від цього об'єкта до інших кластерів.

6. Перевірити умову

$$\|M - M^*\| < \varepsilon,$$

де *M** - матриця нечіткого розбиття на попередній ітерації.

Якщо умова виконано, робота алгоритму завершується, інакше – повернення до кроку 3.

У наведеному алгоритмі найважливішим параметром є кількість кластерів. Правильно обрати кількість кластерів для реальних завдань без будь-якої апріорної інформації про структури даних досить складно.

Для вирішення цієї проблеми можна починати кластеризацію при досить великій кількості кластерів, а потім послідовно поєднувати схожі суміжні кластери. При цьому використовуються різні формальні критерії схожості кластерів.

11.4 Субтрактивна кластеризація

Метод гірничої кластеризації (інші назви - гірська або вершинна кластеризація) не вимагає завдання кількості кластерів. Кластеризація за гірським методом не є нечіткою, проте її можна ефективно використовувати при синтезі нечітких правил даних.

На першому етапі гірської кластеризації визначають точки, які можуть бути центрами кластерів. На другому кроці для кожної такої точки розраховується значення потенціалу, що показує можливість формування кластера у її околиці. Чим щільніше розташовані об'єкти на околиці потенційного центру кластера, тим вище значення його потенціалу. Після цього ітераційно вибираються центри кластерів серед точок із максимальними потенціалами.

На першому кроці необхідно сформувати потенційні центри кластерів *Q*. Тут можливі два способи.

Перший спосіб вибору *Q* передбачає, що центрами кластерів може бути об'єкти кластеризації (рядки матриці *X*), тоді *Q* = *N*.

Другий спосіб вибору потенційних центрів кластерів полягає в дискретизації простору вхідних ознак. Для цього діапазони зміни вхідних ознак розбивають декілька інтервалів. Проводячи через точки розбиття прямі, паралельні координатним осям, отримуємо "решітчастий" гіперкуб. Вузли цих решіток і відповідатимуть центрам потенційних кластерів.

Позначимо через q_r – кількість значень, які можуть набувати центри кластерів по *r*-й координаті. Тоді кількість можливих кластерів

$$Q = \prod_{i=\overline{1,n}} q_r.$$

Введемо позначення для потенційного центру *h*-го кластера:

$$Z_h = (z_{1h}, z_{2h}, \dots, z_{nh}), h = \overline{1, Q}$$

На другому кроці алгоритму розраховується потенціал центрів кластерів за такою формулою:

$$P(Z_h) = \sum_{k=\overline{1,N}} e^{(-\alpha D(Z_h, X_k))}; h = \overline{1, Q},$$

де α - позитивна константа; $D(Z_h, X_k)$ - відстань між потенційним центром кластера Z_h та об'єктом кластеризації X_k.

У евклідовому просторі ця відстань розраховується за формулою

$$D(Z_h, X_k) = \sqrt{\|Z_h - X_k\|^2}.$$

У разі коли об'єкти кластеризації задані двома ознаками *n*=2, графічне зображення розподілу потенціалу буде поверхнею, що нагадує гірський рельєф. Звідси й назва – гірський метод кластеризації. На третьому кроці алгоритму як центри кластерів вибирають координати «гірських» вершин. Для цього центром першого кластера V_1 призначають точку з найбільшим потенціалом:

$$V_1 = \underset{Z_1, Z_2, \dots, Z_Q}{arg\ max}\big(P_1(Z_1), P_1(Z_2), \dots, P_1(Z_Q)\big).$$

Зазвичай найвища вершина оточена кількома досить високими піками. Тому призначення центром наступного кластера точки з максимальним потенціалом серед вершин, що залишилися, призвело б

до виділення великої кількості близько розташованих центрів кластерів. Щоб вибрати наступний центр кластера, необхідно спочатку виключити вплив щойно знайденого кластера. Для цього значення потенціалу для можливих центрів кластерів, що залишилися, перераховується наступним чином: від поточних значень потенціалу віднімають внесок центру щойно знайденого кластера (тому кластеризацію за цим методом іноді називають субтрактивною). Перерахунок потенціалу відбувається за формулою

$$P_2(Z_h) = P_1(Z_h) - P_1(V_1)e^{\left(-\beta D(Z_h, V_1)\right)},$$

де $P_1(.)$ - потенціал на 1-й ітерації; $P_2(.)$ – потенціал на 2-й ітерації; β - позитивна константа, тобто чим ближче точка до першого знайденого центру, тим більше вона послаблюється.

Центр другого кластера визначається за максимальним значенням оновленого потенціалу:

$$V_2 = \underset{Z_1, Z_2, \ldots, Z_Q}{arg\ max}\left(P_2(Z_1), P_2(Z_2), \ldots, P_2(Z_Q)\right).$$

Потім знову перераховується значення потенціалів:

$$P_3(Z_h) = P_2(Z_h) - P_2(V_2)e^{\left(-\beta D(Z_h, V_2)\right)},$$

Ітераційна процедура перерахунку потенціалів та виділення центрів кластерів триває доти, доки максимальне значення потенціалу перевищує певний поріг.

11.5 Кластеризація за допомогою нейронної мережі

Нейронні мережі прямого поширення складаються з безлічі нейронів і ваг, що чергуються, при цьому кожен шар нейронної мережі може мати довільну кількість нейронів. Мережі прямого поширення відрізняються тим, що нейрони кожного шару пов'язані між собою. Нейрони вхідного шару розподіляють сигнали між нейронами прихованого першого шару. Вихідний сигнал кожного нейрона надходить на входи всіх нейронів наступного шару.

Багатошарова нейронна мережа є універсальним апроксиматором – при відповідному виборі числа шарів та параметрів може бути реалізована будь-яка задана нелінійна функція. З іншого боку, є регулярні алгоритми навчання нейронної мережі, тобто такого підбору її параметрів, у якому мінімізується помилка представлення заданої функції.

Нейронні мережі прямого поширення використовують алгоритм

навчання з учителем, що вимагає використання пар вхід-вихід, що описують еталонну поведінку мережі. Після навчання нейронна мережа виконує інтерполяцію (екстраполяцію), генеруючи для нових вхідних даних такий вихід, який є «правильним» стосовно навчальної інформації.

Властивість навчання дозволяє застосовувати нейронні мережі для вирішення безлічі прикладних завдань, у тому числі – завдання нечіткої кластеризації, тобто стиснення великого масиву експериментальної інформації в компактну множину нечітких правил.

Одним із варіантів подання нечітких правил за допомогою нейронної мережі є архітектура ANFIS (Adaptive Neuro-Fuzzy Inference System).

Нейронна мережа ANFIS реалізує систему нечіткого виведення Сугено, в якій нечіткі правила мають вигляд:

$$R_i\text{: Якщо } (x_1 = A_{1i}) \,\&\, (x_2 = A_{2i}) \,\&\, \dots (x_n = A_{ni}), \text{тоді}$$
$$y_i = b_{0i} + x_1 b_{1i} + \cdots + x_n b_{ni}$$

де *i* – номер правила; *x* – входи; *y* – вихід; A_i – терми лінгвістичних змінних; b_i – коефіцієнти полінома, що описує висновок.

Нейронна мережа ANFIS є п'ятишарової нейронною мережою прямого поширення. Призначення шарів таке:

− перший шар – обчислення належності вхідних змінних до терм відповідних лінгвістичних змінних;

− другий шар – обчислення ступеня запуску нечітких правил;

− третій шар – нормалізація ступенів запуску;

− четвертий шар – опис висновків правил;

− п'ятий шар – агрегування результату, отриманого за різними правилами.

Входи мережі до окремого шару не виділяються.

Приклад структури ANFIS представлений на рис. 11.1.

На рис. 11.1 показана нейронна мережа, що має два входи x_1, x_2 та один вихід *y*.

Нейронна мережа описує 4 правила, кожне з яких має дві посилки. Висновок правил описується константою. Літери B, C, M - лінгвістичні мітки термів ("великий", "маленький", "середній").

Розглянемо функціонування шарів ANFIS докладніше.

Шар 1. Кожен вузол першого шару становить один терм. Входи мережі з'єднані лише зі своїми термами. Таким чином, кількість вузлів першого шару дорівнює сумі потужностей терм-множин вхідних змінних. Виходом вузла є ступінь належності значення вхідної змінної відповідного нечіткого терму. Наприклад, при використанні гаусової функції приналежності вихід верхнього вузла першого шару:

$$\mu_{\text{В}}(x_1) = exp\left[-\frac{(x_1 - C)^2}{2\delta^2}\right],$$

де *C* та δ – параметри гаусової функції.

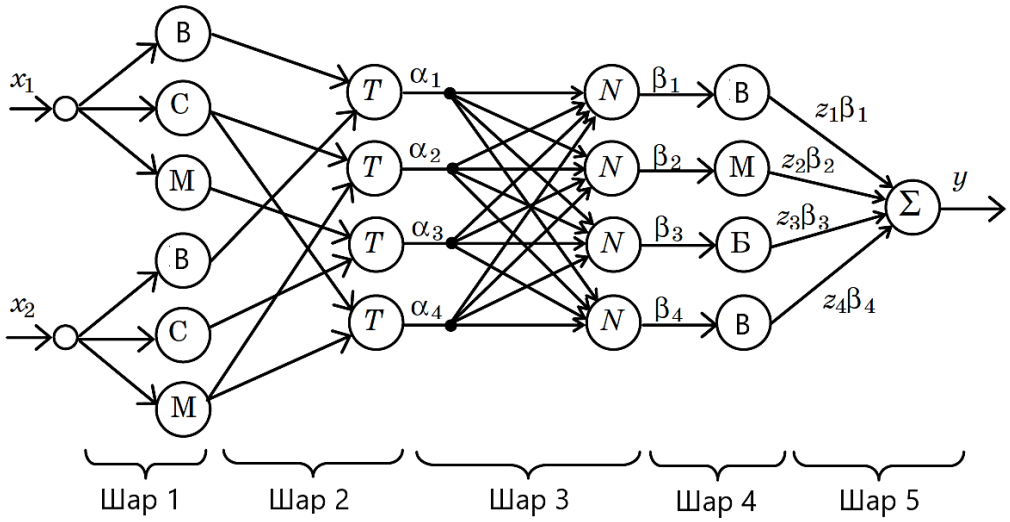


Рисунок 11.1 - Структура мережі ANFIS

Шар 2. Реалізує нечітку *T*-норму, на його виході з'являється величина α_i, що є результатом застосування нечіткої операції AND щодо посилок відповідного правила. Вузол другого шару з'єднаний із тими вузлами першого шару, які формують антецеденти відповідного правила. Отже, кожен вузол другого шару може приймати від 1 до *n* вхідних сигналів. Кількість вузлів цього шару дорівнює кількості правил.

Виходом вузла є ступінь запуску правила, яка розраховується як добуток ступенів належності посилок правила. Наприклад, для верхнього вузла другого шару (рис. 11.1) маємо:

$$\alpha_1 = \mu_{\text{В}}(x_1)\mu_{\text{В}}(x_2).$$

Шар 3. Нейрони 3 шару обчислюють значення нормованого ступеня запуску:

$$\beta_i = \frac{\alpha_i}{\sum_{i=1}^{k} \alpha_i}$$

де *k* – кількість правил.

Шар 4. Кількість вузлів четвертого шару також дорівнює кількості правил. Кожен вузол з'єднаний з одним вузлом третього шару, а також з усіма входами мережі (на рис.11.1 зв'язку з входами не показано).

Якщо деяке правило має нормований рівень запуску β_i, його вихідний сигнал може бути розрахований за формулою

$$z_i = F^{-1}(\beta_i)$$

де $F(z)$ – функція приналежності, що описує термін висновку z_i.

На рисунку 11.2. наведена ілюстрація процесу виведення висновку.

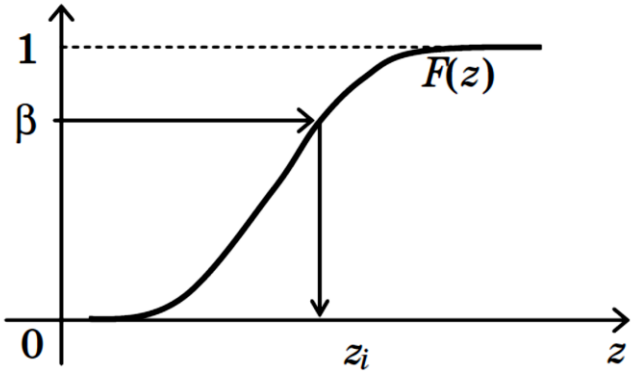


Рисунок 11.2 - Обчислення висновку

Шар 5. Виконує операцію дефазифікації:

$$z_0 = \beta_1 z_1 + \beta_2 z_2 + \cdots + \beta_m z_m.$$

Для навчання мережі ANFIS застосовується комбінація алгоритму зворотного розповсюдження помилки та методу найменших квадратів.

Алгоритм зворотного поширення помилки настроює параметри посилок правил (функцій приналежності).

Метод найменших квадратів служить з метою оцінки коефіцієнтів висновків правил.

Кожен крок процедури налаштування виконується у два етапи. Yf першому етапі на входи подається навчальна вибірка, і з нев'язці між бажаним і дійсним поведінкою мережі ітераційним методом найменших квадратів знаходять оптимальні параметри вузлів четвертого шару.

На другому етапі залишкова нев'язка передається з виходу мережі на входи, і шляхом зворотного поширення помилки модифікуються параметри вузлів першого шару. При цьому знайдені на першому етапі коефіцієнти висновків правил не змінюються.

Ітераційна процедура налаштування триває допоки нев'язка перевищує заздалегідь встановлене значення.

11.6 Кластеризація в MatLab

У системі MatLab реалізовані всі вищеописані інструменти кластеризації:

- нечіткий алгоритм С-середніх,
- субтрактивна кластеризація;

- нейронечітка система ANFIS.

Нечіткий алгоритм С-середніх реалізований у MatLab у функції *fcm*, яка може бути викликана в командному рядку в одному з наступних варіантів:

```
>> [V, M, obj_fcn] = fcm(X, C)
>> [V, M, obj_fcn] = fcm(X, C, options)
```

Функція *fcm* може мати три вхідні аргументи:

− *X* – матриця, яка представляє дані, що підлягають кластеризації. Кожен рядок матриці відповідає одному об'єкту;

− *C* – кількість кластерів, яка має бути отримана в результаті виконання функції *fcm*. Кількість кластерів має бути більшою 1 і менше числа образів, заданих матрицею *X*;

− *Options* – необов'язковий аргумент, що встановлює параметри алгоритму кластеризації: *options*(1) – значення експоненційної ваги (значення за умовчанням – 2.0); *options*(2) – максимальна кількість ітерацій алгоритму кластеризації (значення за умовчанням – 100); *options*(3) – мінімально допустиме значення поліпшення цільової функції за одну ітерацію алгоритму (за замовчуванням – 0.00001); *options*(4) – виведення проміжних результатів під час роботи функції *fcm* (значення за промовчанням – 1).

Для використання значень за умовчанням можна вести *NaN* як значення відповідної координати вектора *options*.

Алгоритм кластеризації зупиняється, коли виконано максимальну кількість ітерацій або коли покращення значення цільової функції за одну ітерацію менше вказаного мінімально допустимого значення.

− *V* – матриця координат центрів кластерів, отриманих у результаті кластеризації. Кожен рядок матриці відповідає центру одного кластера;

− *M* – матриця ступенів приналежності образів до кластерів. Кожен рядок матриці відповідає функції приналежності одного кластера;

− *obj_fcn* – вектор значень цільової функції на кожній ітерації алгоритму кластеризації.

Метод субтрактивної кластеризації реалізований Matlab за допомогою функції *subclust*.

Алгоритм виконує такі основні дії:

1) вибирає точку даних із максимальним потенціалом для представлення центру першого кластера;

2) видаляє всі точки даних на околиці центру першого кластера, розмір якої задається параметром *radii*, щоб визначити наступний нечіткий кластер і координати його центру.

Ці операції продовжуються до тих пір, поки всі точки даних не виявляться всередині околиць радіусу *radii* шуканих центрів кластерів. Функція *subclust* має такий загальний формат:

>> [C, S] = subclust(X, radii, xBounds, options),

Вхідні параметри функції:

− *X* – матриця, що містить дані кластеризації (кожен рядок відповідає окремій точці даних);

− *radii* – вектор, компоненти якого приймають значення з відрізка [0,1], і задають діапазон розрахунку центрів кластерів за кожною ознакою вимірювань. У цьому передбачається, що всі дані містяться у деякому одиничному гіперкубе. Якщо значення *radii* велике, то буде отримано малу кількість кластерів. Рекомендовані значення *radii* перебувають у діапазоні від 0.2 до 0.5;

− *xBounds* – матриця розмірності 2×*q* визначає спосіб відображення матриці даних *X* у деякому одиничному гіперкубі. Тут *q* - кількість ознак, що розглядаються в безлічі даних. Цей параметр необов'язковий, якщо матриця вже нормалізована. Перший рядок цієї матриці містить мінімальні значення інтервалу вимірювання кожної ознаки, а другий рядок – максимальні значення вимірювання кожної ознаки;

− *Options* – цей вектор може бути використаний для зміни заданих за умовчанням значень параметрів алгоритму кластеризації. Цей вектор має такі компоненти:

options(1) – *squashFactor* – параметр, що використовується як коефіцієнт для множення значень *radii*, які визначають околицю центру кластера. Це здійснюється з метою зменшення впливу потенціалу граничних точок, що розглядаються як частина нечіткого кластера (за умовчанням це значення дорівнює 1.25);

options(2) – *acceptRatio* – параметр, що встановлює потенціал як частина потенціалу першого кластера, вище якого інша точка даних може розглядатися як центр іншого кластера (за умовчанням це значення дорівнює 0.5);

options(3) – *rejectRatio* – параметр, що встановлює потенціал як частину потенціалу першого кластера, нижче якого інша точка даних неспроможна розглядатися як центр іншого кластера (за умовчанням це значення дорівнює 0.15);

options(4) – *verbose* – якщо значення цього параметра не дорівнює нулю, то на екран монітора виводиться інформація про виконання процесу кластеризації (за умовчанням це значення дорівнює 0).

Вихідні параметри функція *subclust*:

− *C* – матриця значень координат центрів нечітких кластерів. Кожен рядок містить координати одного центру.

− *S* – вектор, компоненти якого є σ-значення, що визначають діапазон впливу центрів кластерів за кожною з ознак, що розглядаються. Всі центри кластерів мають однакову кількість σ-значень.

Приклад використання функції у скороченому форматі:

```
>> [C,S] = subclust(X,0.5)
```

Крім командного рядка в MatLab існує можливість використання спеціального графічного модуля, що викликається командою

```
>> findcluster
```

Модуль Findcluster дозволяє автоматично знаходити центри багатомірних кластерів даних за допомогою нечіткого алгоритму *C*-середніх або алгоритму субтрактивної кластеризації.

Основне графічне вікно модуля Findcluster показано на рис. 11.3.

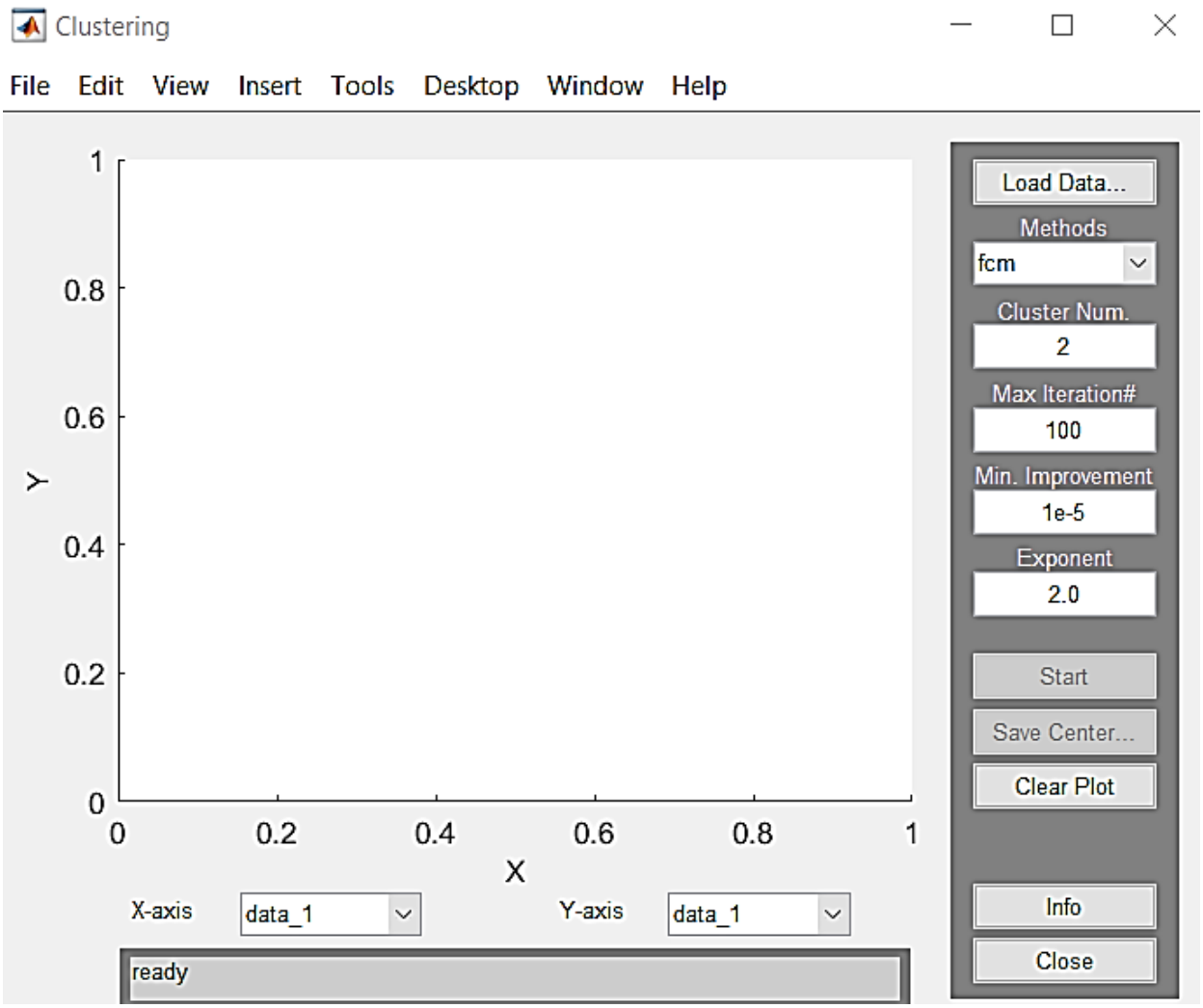


Рисунок 11.3 – Графічне вікно модуля Findcluster

Верхня частина графічного вікна містить типові функції меню: File, Edit, View, Insert, Tools, Windows та Help.

Центральна частина графічного вікна являє собою область візуалізації. У цій області у двовимірному просторі виводяться експериментальні дані (образи) та знайдені центри кластерів. Для

образів використовується маркер у вигляді червоного кола, а центрів кластерів – маркер як чорної точки.

Нижче області візуалізації розташоване меню вибору координатних осей *X* та *Y*, що дозволяють асоціювати ознаки образів з осями абсцис та ординат відповідно.

Область виведення поточної інформації розташована внизу графічного вікна, до вікна виводиться, наприклад, стан модуля, номер ітерації алгоритму кластеризації, значення цільової функції тощо.

Кнопка завантаження даних *Load Data* розташована зверху праворуч. Після її натискання з'являється типове вікно відкриття файлу. У файлі дані повинні бути записані рядково, тобто кожному образу повинен відповідати один рядок файлу даних.

Область кластеризації розташована праворуч нижче кнопки завантаження даних. У цій області можливо вибрати алгоритм кластеризації, встановити параметри алгоритму кластеризації, провести кластеризацію та зберегти координати центрів кластерів у вигляді файлу.

Меню *Method* дозволяє вибрати один із двох алгоритмів кластеризації: *subtractiv* – алгоритм субтрактивної кластеризації; *fcm* – нечіткий алгоритм *C*-середніх. При виборі алгоритму субтрактивної кластеризації графічне вікно змінюється.

При виборі *subtractiv* можна встановити такі параметри: *Influence Range, Squash, Accept Ratio і Reject Ratio*.

При виборі *fcm* алгоритму можна встановити значення параметрів: **Cluster Num** – кількість кластерів; *Max* Iteration - максимальна кількість ітерацій алгоритму; *Min* – мінімально допустиме значення покращення цільової функції за одну ітерацію алгоритму; *Exponent* – значення експонентної ваги.

Кнопка *Start* запускає кластеризацію. Кнопка *Save Center* дозволяє зберегти координати знайдених центрів кластерів.

Кнопка *Clear Plot* дає змогу очистити поле виведення даних.

На рисунку 11.4 наведено графічне вікно модуля Findcluster після застосування алгоритм субтрактивної кластеризації.

У системі MatLab реалізований ANFIS-редактор, який дозволяє за допомогою нейронечіткого опису автоматично синтезувати з експериментальних даних нечіткі правила.

Як було показано вище, мережа ANFIS визначає систему нечіткого логічного висновку типу Сугено. Параметри мережі після навчання налаштовуються так, щоб мінімізувати відхилення між результатами моделювання та експериментальними даними.

Завантаження ANFIS-редактора здійснюється за командою

```
>> anfisedit
```

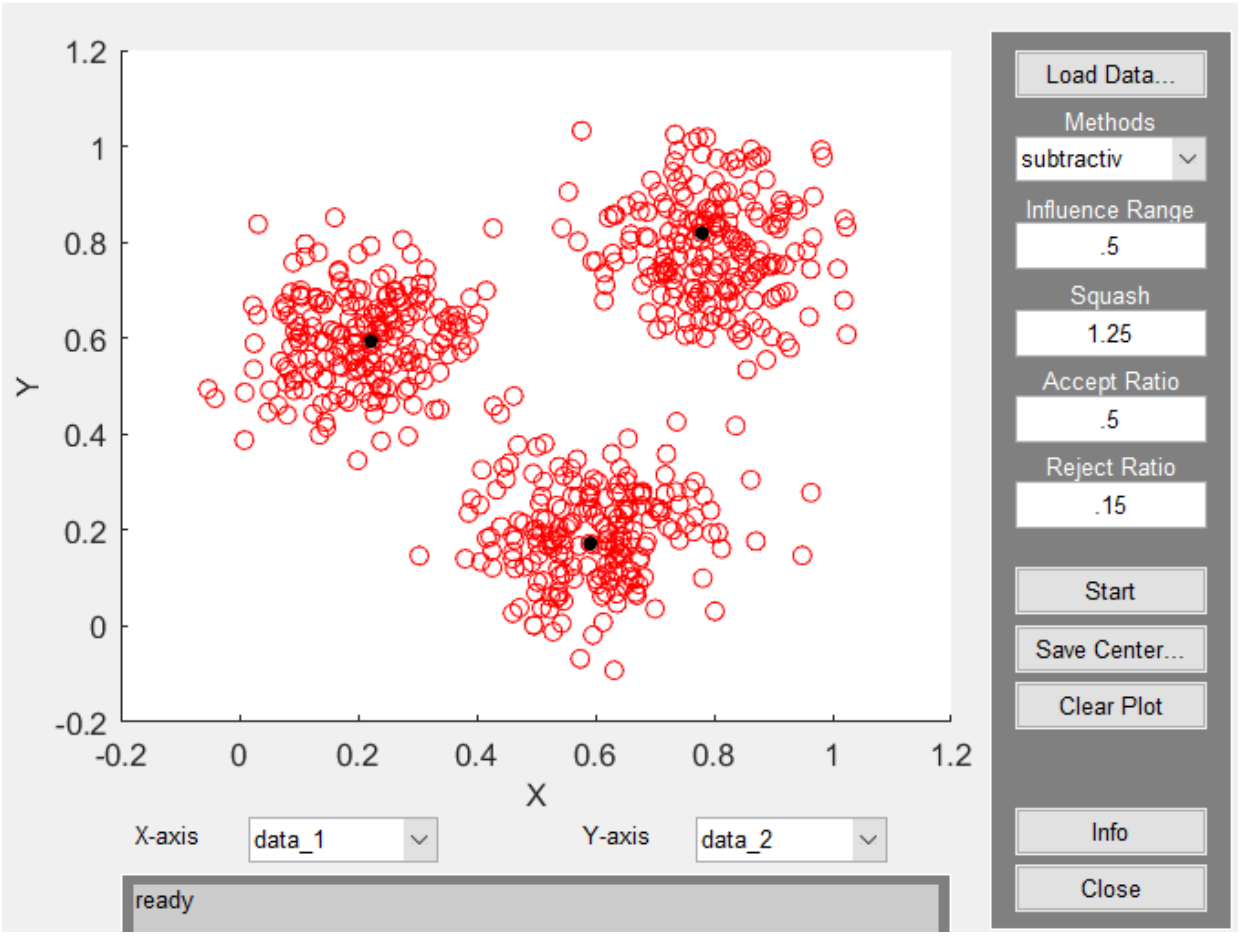


Рисунок 11.4 – Графічне вікно модуля Findcluster після застосування алгоритм субтрактивної кластеризації

Графічне вікно Anfis Editor показано на рис. 11.5 воно містить кілька областей.

У центрі вікна розташована область візуалізації. У цій області виводиться два типи інформації:

– під час навчання системи – графік залежності помилки навчання від порядкового номера ітерації;

– при завантаженні даних та тестуванні системи – експериментальні дані та результати моделювання. При цьому по осі абсцис відкладається порядковий номер рядка даних у вибірці (навчальної, тестуючої чи контрольної), а, по осі ординат – значення вихідний змінної для цього рядка вибірки. Використовуються такі маркери: блакитна точка (.) – вибірка, що тестує; блакитне коло (o) – навчальна вибірка; блакитний плюс (+) – контрольна вибірка; червона зірочка (*) – результати моделювання

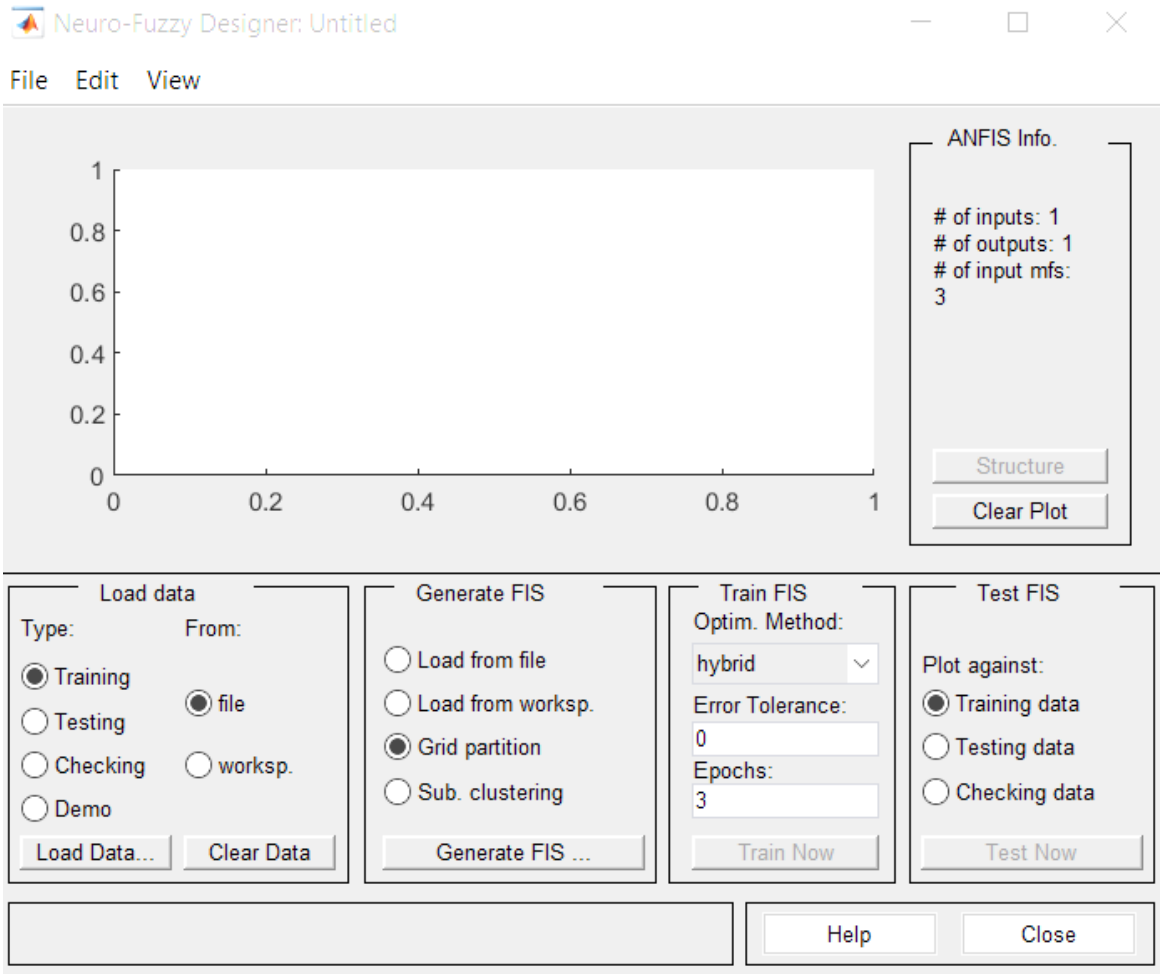


Рисунок 11.5 – Графічне вікно модуля ANFIS

Правіше області візуалізації знаходиться область властивостей *ANFIS info*.

В області властивостей виводиться інформація про кількість вхідних та вихідних змінних, про кількість функцій приналежності для кожної вхідної змінної, а також про кількість рядків у вибірках. У цій області розташовані дві кнопки *Structure* та *Clear Plot*.

Натискання кнопки *Structure* відкриває нове графічне вікно, в якому система нечіткого логічного виводу представляє у вигляді нейро-нечіткої мережі. Кількість входів цієї мережі залежить від кількості стовпців навчальної вибірки. Кількість нейронів 2-4 шари відповідає кількості правил.

На рис. 11.6 показаний приклад мережі ANFIS, в якій дві вхідні змінні (для опису кожної використано по три терми) та дев’ять правил. Кількість правил залежить від вибору параметрів у Generate FIS.

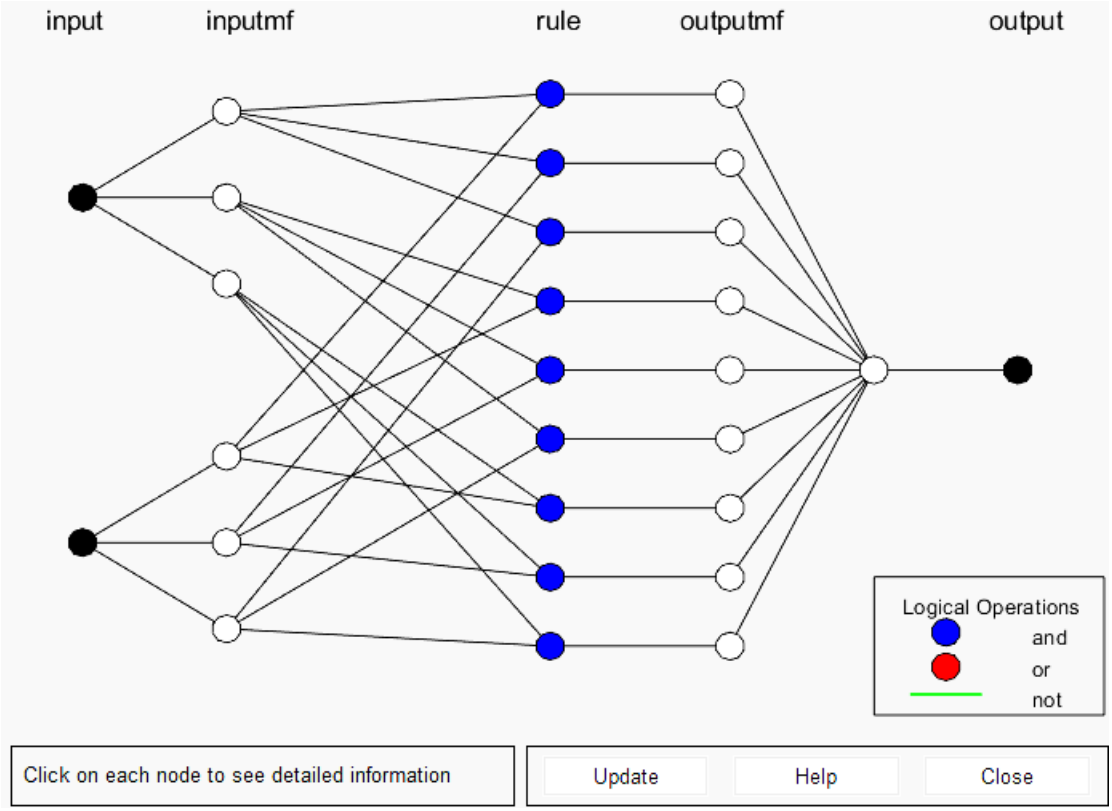


Рисунок 11.6 – Приклад нейронечіткої структури

Натискання кнопки *Clear Plot* дозволяє очистити область візуалізації.

В області завантаження даних *Load data* розташовані:

− *Type* – меню вибору типу даних: *Traning* – навчальна вибірка; *Testing* - тестування вибірка; *Checking* – контрольна вибірка; *Demo* – демонстраційний приклад;

− *From* – меню вибору джерела даних: *Disk* – диск; *Worksp*. - робоча область MatLab;

− *Load Data* – кнопка завантаження даних;

− *Clear Data* – кнопка очищення даних.

Протягом одного сеансу роботи ANFIS-редактора слід завантажувати дані одного формату, тобто кількість вхідних змінних у вибірках має бути однаковим.

Generate FIS – меню створення вихідної системи нечіткого логічного висновку, що містить такі альтернативи:

− *Load from disk* – завантаження існуючої системи з диска;

− *Load from worksp*. - Завантаження системи з робочої області MatLab;

− *Grid partition* – генерування опису вхідних змінних за методом решітки (без кластеризації);

− *Sub. Clustering* – генерування методом субкластеризації.

В області також розташована кнопка *Generate*, після натискання якої генерується вихідна система нечіткого логічного висновку.

При виборі *Grid partition* з'являється вікно введення параметрів методу решітки (див. рис. 11.7), в якому потрібно вказати кількість термів для кожної вхідної змінної та тип функцій приналежності для вхідних та вихідних змінних.

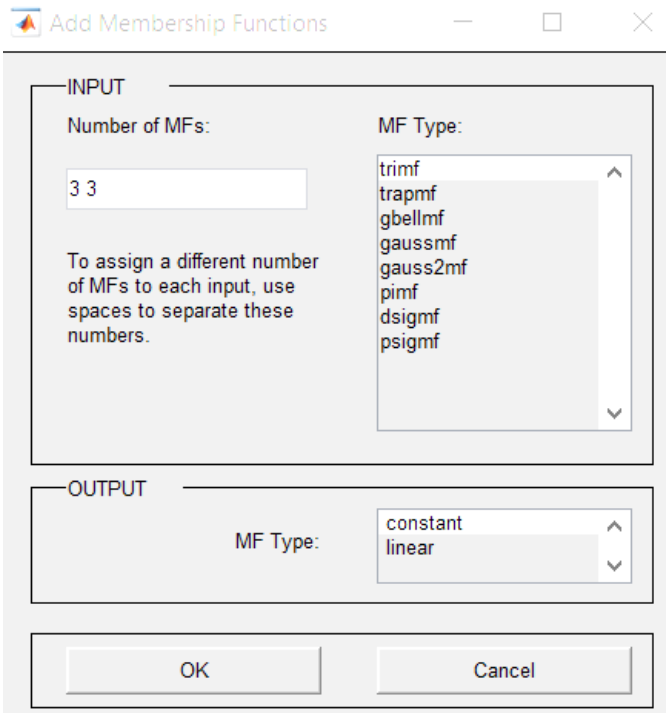


Рисунок 11.7 - Вікно введення параметрів для методу решітки

Таким чином, при виборі методу грат виконується нечітке розбиття базових шкал вхідних змінних. Налаштування параметрів термів не потрібно, а кількість правил дорівнює добутку потужностей терм-множин лінгвістичних змінних, що описують посилки (див. рис. 11.7).

Під час вибору Sub. clustering з'являється вікно введення наступних параметрів методу субкластеризації (див. рис. 11.8).

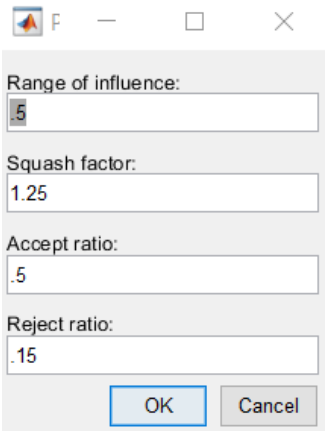


Рисунок 11.8 – Вікно введення параметрів для субтрактивної кластеризації

На рисунку 11.8 позначено:

- *Range of influence* – рівні впливу вхідних змінних;

- *Squash factor* – коефіцієнт подавлення;

- *Accept ratio* – коефіцієнт, що встановлює у скільки разів потенціал даної точки повинен бути вищим за потенціал центру першого кластера для того, щоб центром одного з кластерів була призначена розглянута точка;

- *Reject ratio* -коефіцієнт, що встановлює у скільки разів потенціал даної точки повинен бути нижче потенціалу центру першого кластера, щоб точка, що розглядається, була виключена з можливих центрів кластерів.

Область навчання (*Train FIS*) містить меню вибору методу оптимізації (*Optim. method*), поле завдання необхідної точності навчання (*Error tolerance*), поле завдання кількості ітерацій навчання (*Epochs*) і кнопку Train Now, натискання якої запускає режим навчання. Проміжні результати навчання виводяться в область візуалізації та робочу область MatLab. В ANFIS-редакторі реалізовано два методи навчання:

– *Backpropa* – метод зворотного поширення помилки, заснований на ідеях методу якнайшвидшого спуску;

- *Hybrid* - гібридний метод, що поєднує метод зворотного розповсюдження помилки з методом найменших квадратів.

В області тестування (*Test FIS*) розташовані меню вибору вибірки та кнопка *Test Now*, при натисканні якої відбувається тестування нечіткої системи з виведенням результатів у область візуалізації.

Розглянемо приклад розв'язання задач кластеризації засобами MatLab. Проблема полягає у вилучення керуючих правил з інформації про процес управління, яка являє собою вхід - вихідні дані пристрою управління. У англомовній літературі такий підхід називається *data driven*, тобто. синтез, керований даними або синтез за даними. В якості джерела даних може бути база даних, оператор процесу або існуючий лінійний регулятор.

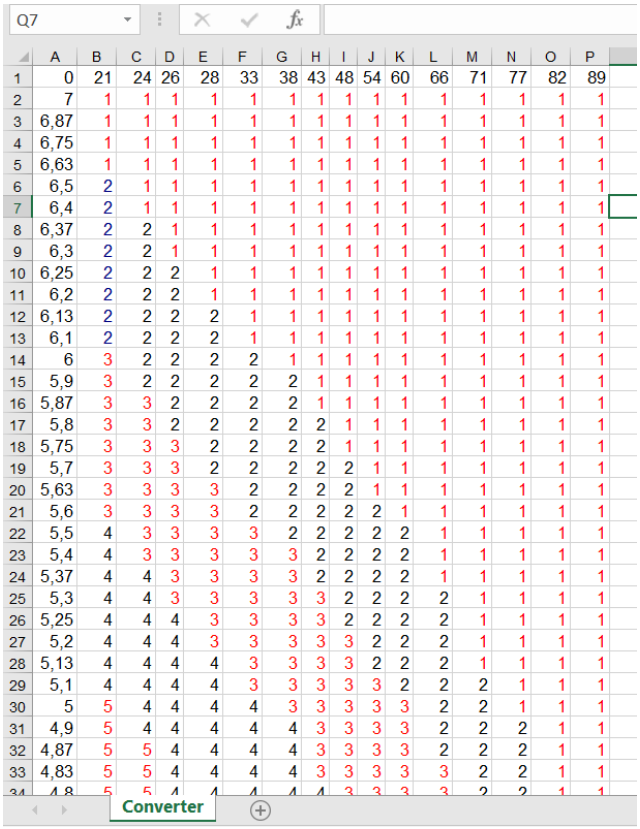
Мета синтезу нечіткого логічного регулятора з наступною заміною ним існуючого регулятора полягає у підвищенні швидкодії та точності (при заміні оператора), або у збільшенні перешкодостійкості та розширенні робочого діапазону (при заміні лінійного регулятора).

11.7 Вилучення керуючих правил з інформації про процес управління

Розглянемо на прикладі вилучення керуючих правил з інформації про процес управління, що описана базою даних.

За експериментальними даними будуємо у MS Excel матрицю залежності вихідного параметра *y* від двох вхідних параметрів x_1 та x_2, які розташуємо відповідно у стовбці «А» та строчці «1» відповідно (див.

рис. 11.9). Надаємо ім’я файлу data2.xls (ім’я може буди будь яким).

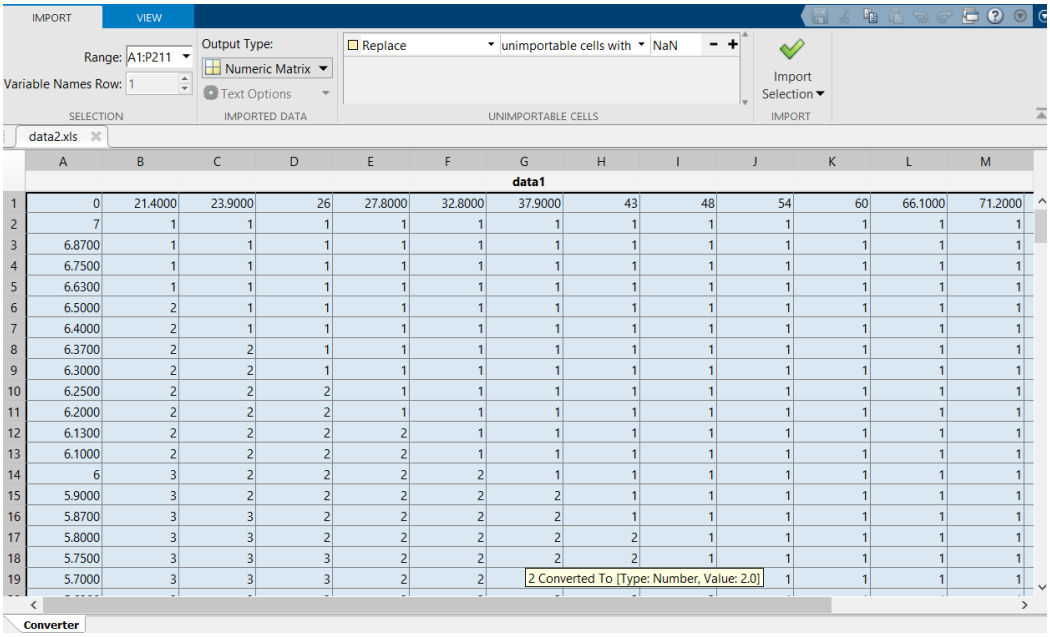


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	0	21	24	26	28	33	38	43	48	54	60	66	71	77	82	89
2	7	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
3	6,87	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
4	6,75	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
5	6,63	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	6,5	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7	6,4	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1
8	6,37	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1
9	6,3	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1
10	6,25	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1
11	6,2	2	2	2	1	1	1	1	1	1	1	1	1	1	1	1
12	6,13	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1
13	6,1	2	2	2	2	1	1	1	1	1	1	1	1	1	1	1
14	6	3	2	2	2	2	1	1	1	1	1	1	1	1	1	1
15	5,9	3	2	2	2	2	2	1	1	1	1	1	1	1	1	1
16	5,87	3	3	2	2	2	2	1	1	1	1	1	1	1	1	1
17	5,8	3	3	2	2	2	2	2	1	1	1	1	1	1	1	1
18	5,75	3	3	3	2	2	2	2	1	1	1	1	1	1	1	1
19	5,7	3	3	3	2	2	2	2	2	1	1	1	1	1	1	1
20	5,63	3	3	3	3	2	2	2	2	1	1	1	1	1	1	1
21	5,6	3	3	3	3	2	2	2	2	2	1	1	1	1	1	1
22	5,5	4	3	3	3	3	2	2	2	2	2	1	1	1	1	1
23	5,4	4	3	3	3	3	3	2	2	2	2	1	1	1	1	1
24	5,37	4	4	3	3	3	3	2	2	2	2	1	1	1	1	1
25	5,3	4	4	3	3	3	3	3	2	2	2	2	1	1	1	1
26	5,25	4	4	4	3	3	3	3	2	2	2	2	1	1	1	1
27	5,2	4	4	4	3	3	3	3	3	2	2	2	1	1	1	1
28	5,13	4	4	4	4	3	3	3	3	2	2	2	1	1	1	1
29	5,1	4	4	4	4	3	3	3	3	3	2	2	2	1	1	1
30	5	5	4	4	4	4	3	3	3	3	3	2	2	1	1	1
31	4,9	5	4	4	4	4	4	3	3	3	3	2	2	2	1	1
32	4,87	5	5	4	4	4	4	3	3	3	3	2	2	2	1	1
33	4,83	5	5	4	4	4	4	3	3	3	3	3	2	2	1	1

Converter

Рисунок 11.9 - Матриця залежності вихідного параметра y від двох вхідних параметрів x_1 та x_2

Імпортуємо базу даних (файл data2.xls) у MatLab як цифрову матрицю (Numerical Matrix) (див. рис. 11.10).



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	0	21.4000	23.9000	26	27.8000	32.8000	37.9000	43	48	54	60	66.1000	71.2000
2	7	1	1	1	1	1	1	1	1	1	1	1	1
3	6.8700	1	1	1	1	1	1	1	1	1	1	1	1
4	6.7500	1	1	1	1	1	1	1	1	1	1	1	1
5	6.6300	1	1	1	1	1	1	1	1	1	1	1	1
6	6.5000	2	1	1	1	1	1	1	1	1	1	1	1
7	6.4000	2	1	1	1	1	1	1	1	1	1	1	1
8	6.3700	2	2	1	1	1	1	1	1	1	1	1	1
9	6.3000	2	2	1	1	1	1	1	1	1	1	1	1
10	6.2500	2	2	2	1	1	1	1	1	1	1	1	1
11	6.2000	2	2	2	1	1	1	1	1	1	1	1	1
12	6.1300	2	2	2	2	1	1	1	1	1	1	1	1
13	6.1000	2	2	2	2	1	1	1	1	1	1	1	1
14	6	3	2	2	2	2	1	1	1	1	1	1	1
15	5.9000	3	2	2	2	2	2	1	1	1	1	1	1
16	5.8700	3	3	2	2	2	2	1	1	1	1	1	1
17	5.8000	3	3	2	2	2	2	2	1	1	1	1	1
18	5.7500	3	3	3	2	2	2	2	1	1	1	1	1
19	5.7000	3	3	3	2	2	2	2	1	1	1	1	1

Рисунок 11.10 – Вікно імпорту бази даних у цифрову матрицю

Перетворимо базу даних у 3 стовбця – масиву даних, де 1 стовбець відповідає вхідному параметру x_1, 2 стовбець вхідному параметру x_2, а 3 – вихідному значенню y. Програма перетворення:

```
%%
p=data2(2:end,1);
M=data2(1,2:end);
tabT=data2(2:end,2:end);
[max_i max_j]=size(tabT);
%%
xxT=zeros(max_j*max_i,3);
for i=1:max_i
   for j=1:max_j
      xxT(i+(j-1)*max_i,1)=p(i);
      xxT(i+(j-1)*max_i,2)=M(j);
      xxT(i+(j-1)*max_i,3)=tabT(i,j);

   end
end
%%
anfisedit
```

Результатом виконання програми є формування перетвореної бази даних з ім'ям *xxT* (див. рис.11.11).

xxT

3150x3 double

	1	2	3	4
1	7	21.4000	1	
2	6.8700	21.4000	1	
3	6.7500	21.4000	1	
4	6.6300	21.4000	1	
5	6.5000	21.4000	2	
6	6.4000	21.4000	2	
7	6.3700	21.4000	2	
8	6.3000	21.4000	2	
9	6.2500	21.4000	2	
10	6.2000	21.4000	2	
11	6.1300	21.4000	2	
12	6.1000	21.4000	2	
13	6	21.4000	3	
14	5.9000	21.4000	3	
15	5.8700	21.4000	3	
16	5.8000	21.4000	3	

Рисунок 11.11 – Вікно бази даних з ім'ям xxT

У модулі ANFIS завантажуємо базу даних *xxT* натисканням кнопки *Load Data*. Попередньо здійснив налаштування: Type – Traning Data; From – worksp. ,як зображено на рисунку 11.12.

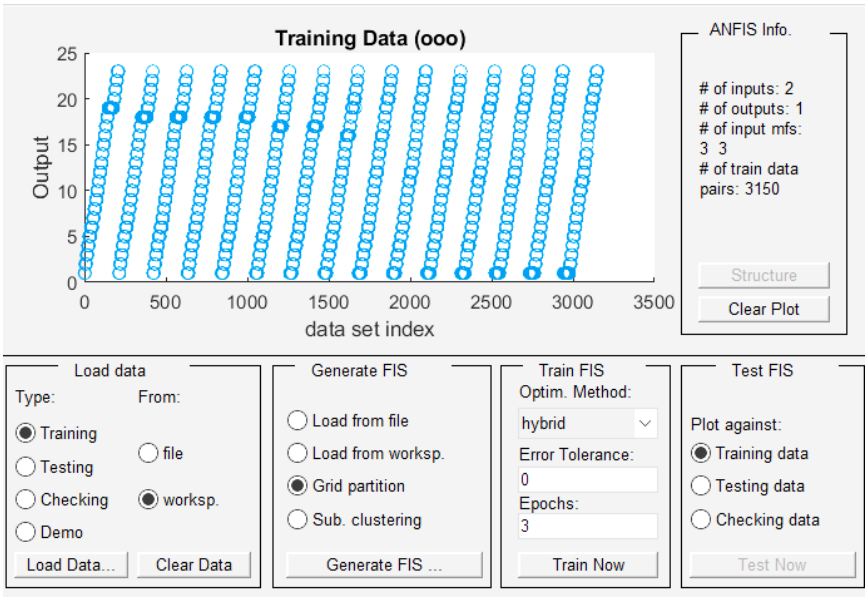


Рисунок 11.12 - Результат завантаження бази даних до модуля ANFIS

Після завантаження даних у меню побудови системи нечіткого логічного висновку *Generate FIS* позначаємо генерування опису вхідних змінних за методом решітки (без кластеризації) - *Grid partition* (див. рис. 11.12).

При виборі *Grid partition* з'являється вікно введення параметрів методу решітки (див. рис. 11.13), в якої вказуємо кількість термів рівної 5 для кожної вхідної змінної та тип функцій приналежності *gaussmf* для вхідних та вихідних змінних (тип функції приналежності може бути іншим в залежності від типу об’єкту управління)

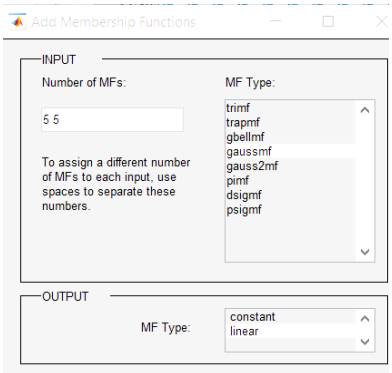


Рисунок 11.13 – Вікно введення параметрів методу решітки

У вікні модуля *ANFIS* натисканням кнопки *Structure* можливо переглянути згенеровану структуру система нечіткого логічного виводу представляє у вигляді нейро-нечіткої мережі (див. рис. 11.14). Для даного випадку кількість входів мережі дорівнює 2 - кількості стовпців навчальної вибірки. Кількість нейронів 2-4 шари відповідає кількості

правил яка дорівнює 25.

На наступному кроці проводиться навчання побудованої нейронечіткої моделі. В область навчання (*Train FIS*) обраємо гібридний метод оптимізації (*Optim. Method - Hybrid*), у полі завдання необхідної точності навчання (*Error tolerance*) встановлюємо значення 0, а у полі завдання кількості ітерацій навчання (*Epochs*) встановлюємо 10 епох (див. рис. 11.12).

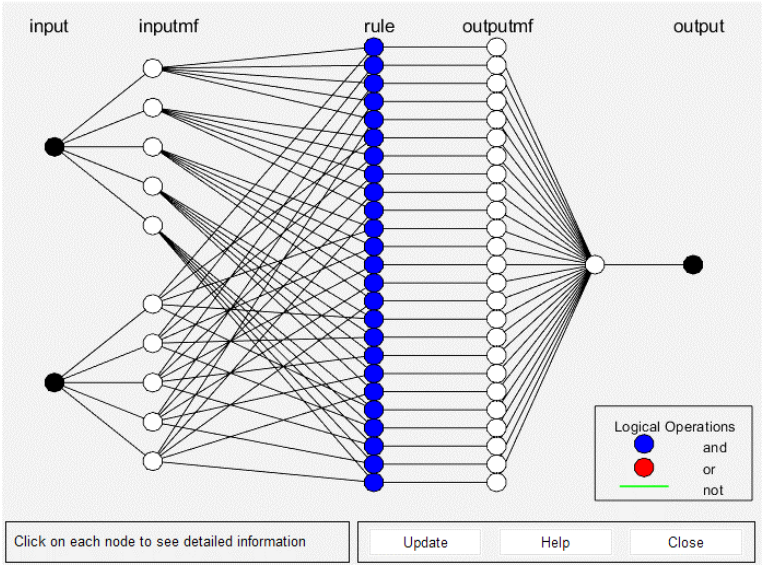


Рисунок 11.14 – Згенерована нейронечітка структура

Проводимо навчання натисканням кнопки *Train Now*. Проміжні результати навчання наведені на рисунку 11.15 виводяться в область візуалізації та робочу область MatLab.

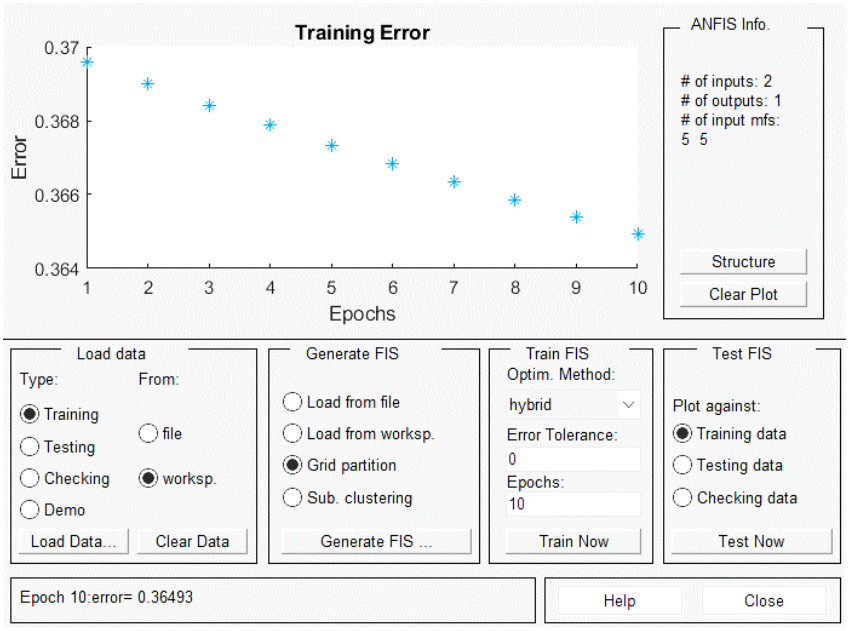


Рисунок 11.15 - Проміжні результати навчання

По завершенню навчання за даними навчання *Training Data* проводитися тестування нейронечіткої системи (*Test FIS*). Після натискання кнопки *Test Now*, відбувається тестування нечіткої системи з виведенням результатів у область візуалізації які наведені на рисунку 11.16.

Рисунок 11.16 – Результати тестування нечіткої системи

Після виконання вище наведених операції у вікні редактора модуля ANFIS (див. рис.12 .5) в меню Edit викликаємо редактор системи нечіткого висновку (Fuzzy Inference System – FIS) натисканням - FIS Properties (див. рис. 11.17).

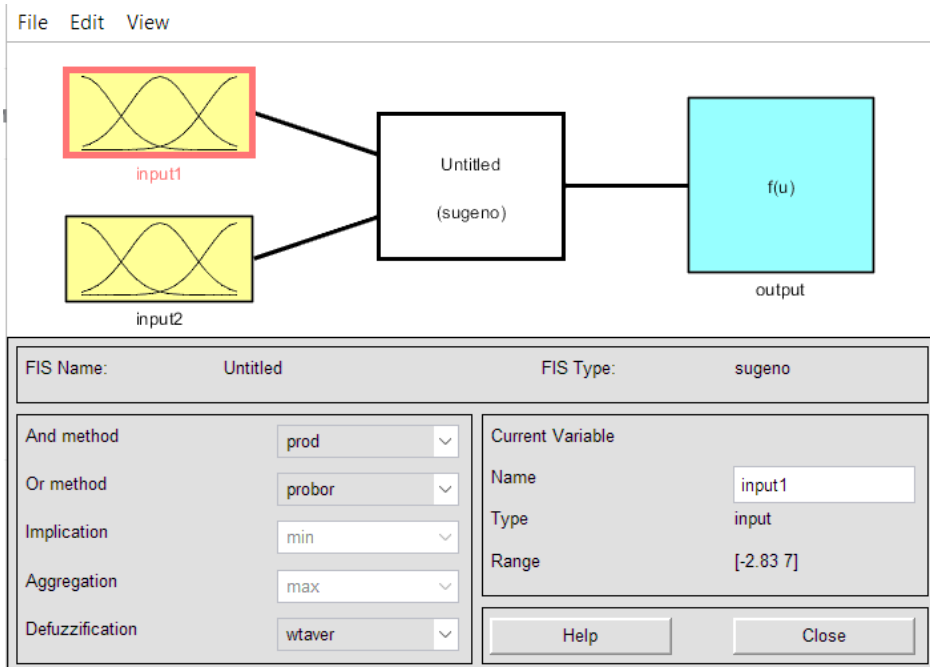


Рисунок 11.17 – Згенерований інтерфейс FIS editor

Опис у синтезовану *FIS editor* відповідає налаштуванням здійсненим *Generate FIS* (див. рис. 11.13).

Для опису вхідних та вихідних логічних змінних у *Generate FIS призначено* гаусовські функцій приналежності.

Результат налаштування функцій приналежності змінних наведено на рис. 11.18.

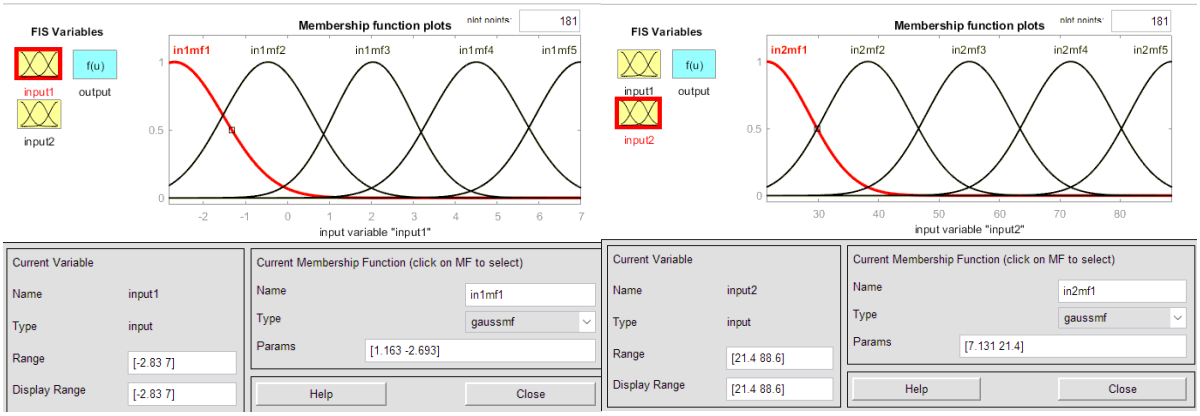


Рисунок 11.18 – Результат налаштування функцій приналежності змінних

Сформульовані управляючі правила наведено з рис. 11.19

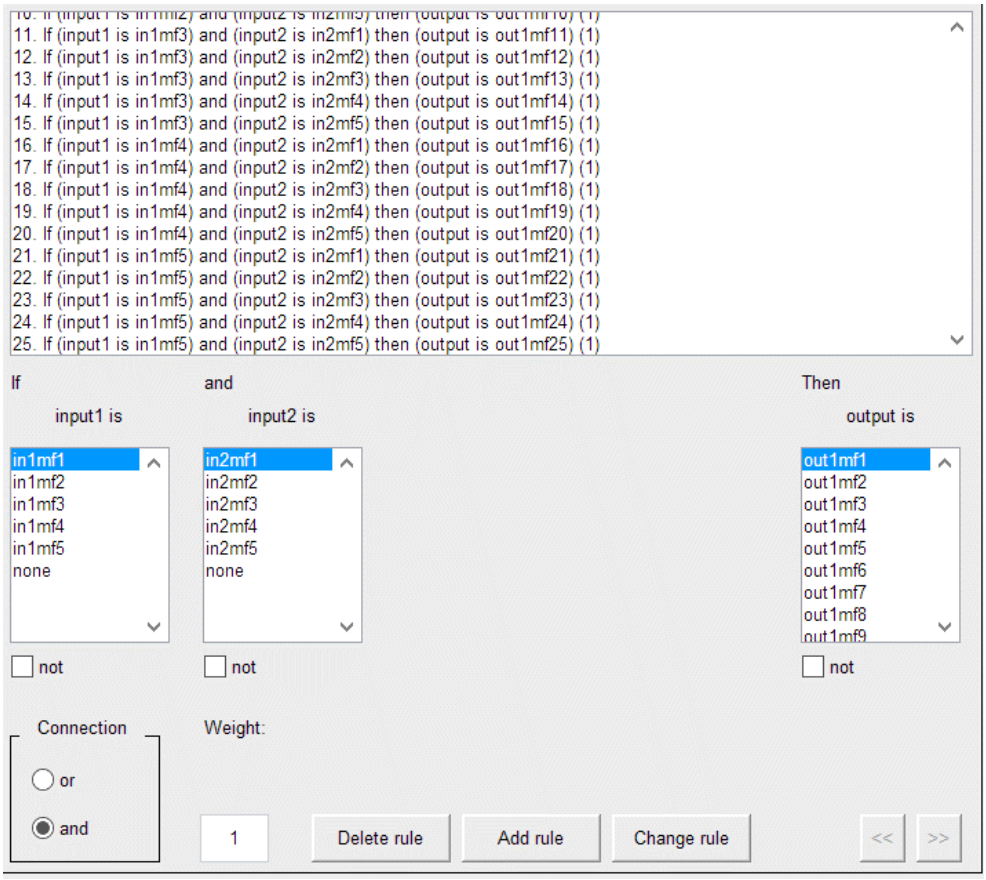


Рисунок 11.19 - Налаштовування опису правил функціонування нечіткого супервізору НЛР_ПІД

Після опису правил можливо за допомогою

- інтерфейсу *Rule Viewer* можливо переглянути роботу системи нечіткого висновку за різних вхідних даних (див. рис. 11.20).

− інтерфейсу Surface Viewer можливо переглянути графічне подання закону управління (рис. 11.21).

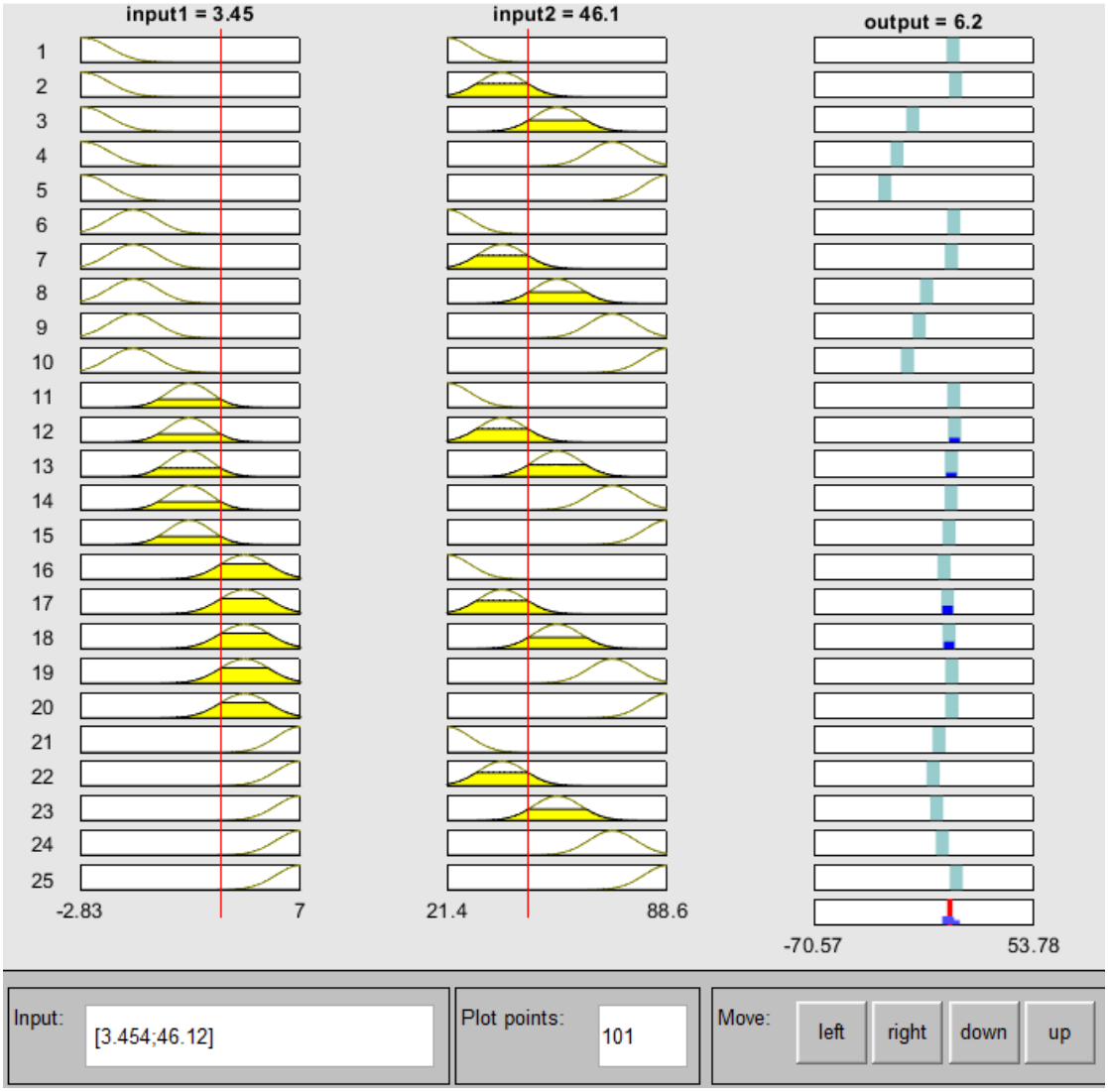


Рисунок 11.20 - Робота системи нечіткого висновку

Примітка. Можлива на першому етапі здійснити налаштування опису системі нечіткого логічного висновку у *FIS editor* після цього завантажити його у модуль ANFIS. Для цього у модулі ANFIS меню *Generate FIS* необхідно завантажити нечітку систему з робочої області MatLab − *Load from disk,* або з диска – *Load from disk*

Перевіримо функціонування згенерованої нечіткої системи управління за базою даних. Для цього побудуємо Simulink модель наведену на рисунку 11.22.

У редакторі системи нечіткого висновку здійснює запис результатів налаштування у Fuzzy Logic Controller. Для цього у вкладці

File здійснюємо Export в математичну модуль нечіткого контролера From Workspase вказавши ім’я Fuzzy Logic Controller.

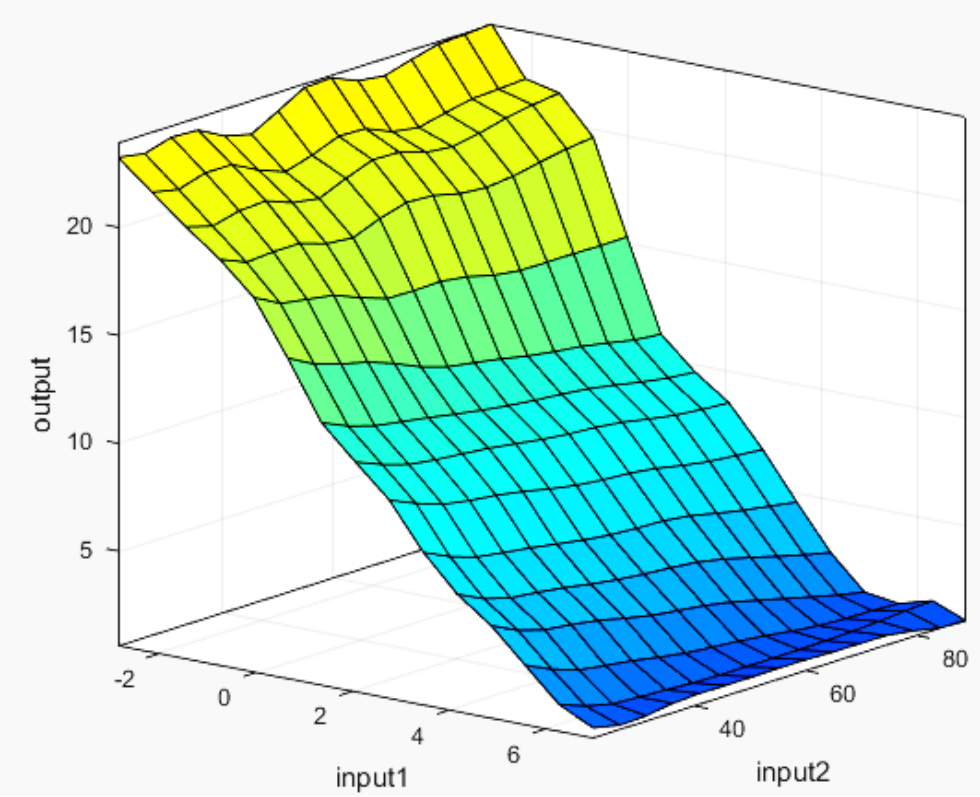


Рисунок 11.21 - Графічне подання закону управління

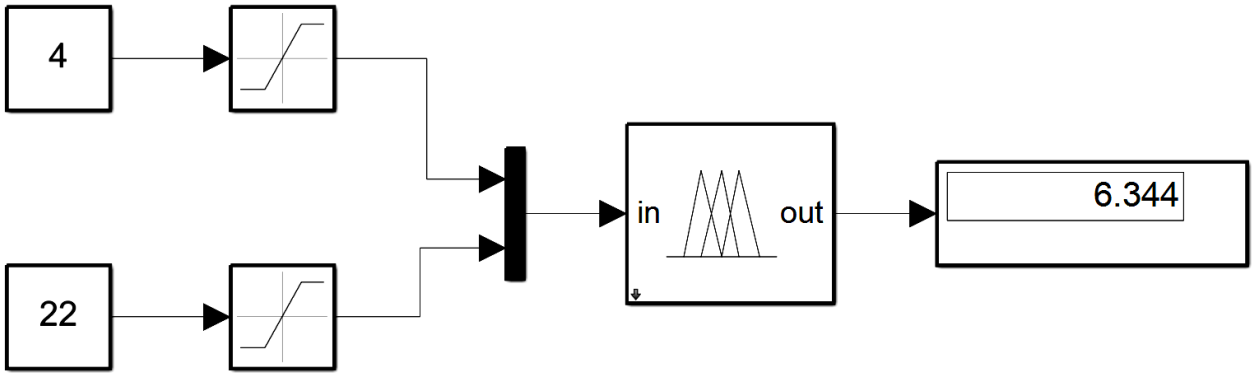


Рисунок 11.22 - Simulink модель функціонування згенерованої нечіткої системи управління за базою даних

Перевірка правильності функціонування моделі здійснюється шляхом завдання на вхід Fuzzy Logic Controller двох вхідних параметрів x_1 та x_2 з бази даних. При вірному налаштування нечіткої системи управління на виході Fuzzy Logic Controller формується відповідне значення y. (див. рис. 11.22).

11.8 Синтез нечіткого регулятора по даним

Розглянемо на прикладі синтез нечіткого регулятора по даним. Нехай є система управління з ПД-регулятором, коефіцієнти якого знайдено методом Зиглера-Нікольса.

Визначення коефіцієнтів передачі регулятора методом Зиглера-Нікольса

Задаємо на математичної моделі K_D =0, а коефіцієнт пропорційної складової регулятора збільшуємо до значення при якому у системі виникає коливальний перехідний процес з однаковим періодом коливань. У даному випадку такі коливання у системі управління настають при значенні K=K_P=92.

На графіку перехідного процесу визначається період коливань T=1,2.

Розраховуємо коефіцієнти передачі ПД-регулятора

$$K_P = 0.6K = 0{,}6 \cdot 92 = 55{,}2;$$

$$K_D = \frac{K_P T}{8} = \frac{55{,}2 \cdot 1{,}2}{8} = 8{,}3.$$

Математична модель системи управління з налаштованим ПД-регулятором наведена на рисунку 11.23.

На рисунку 11.24 наведено графік перехідного процесу системи управління з ПД-регулятором.

На рис. 11.25 та 11.26 показані графіки зміни помилки управління, її похідної та сигналу управління протягом перехідного процесу.

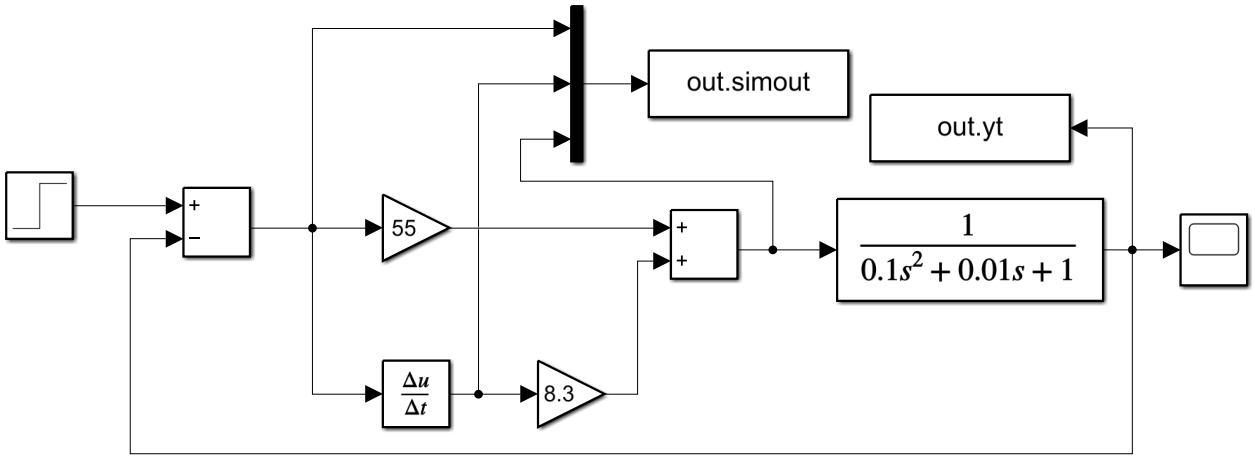


Рисунок 11.23 – Математична модель системи управління з ПД-регулятором

Потрібно синтезувати НЛР виходячи з інформації про процес управління.

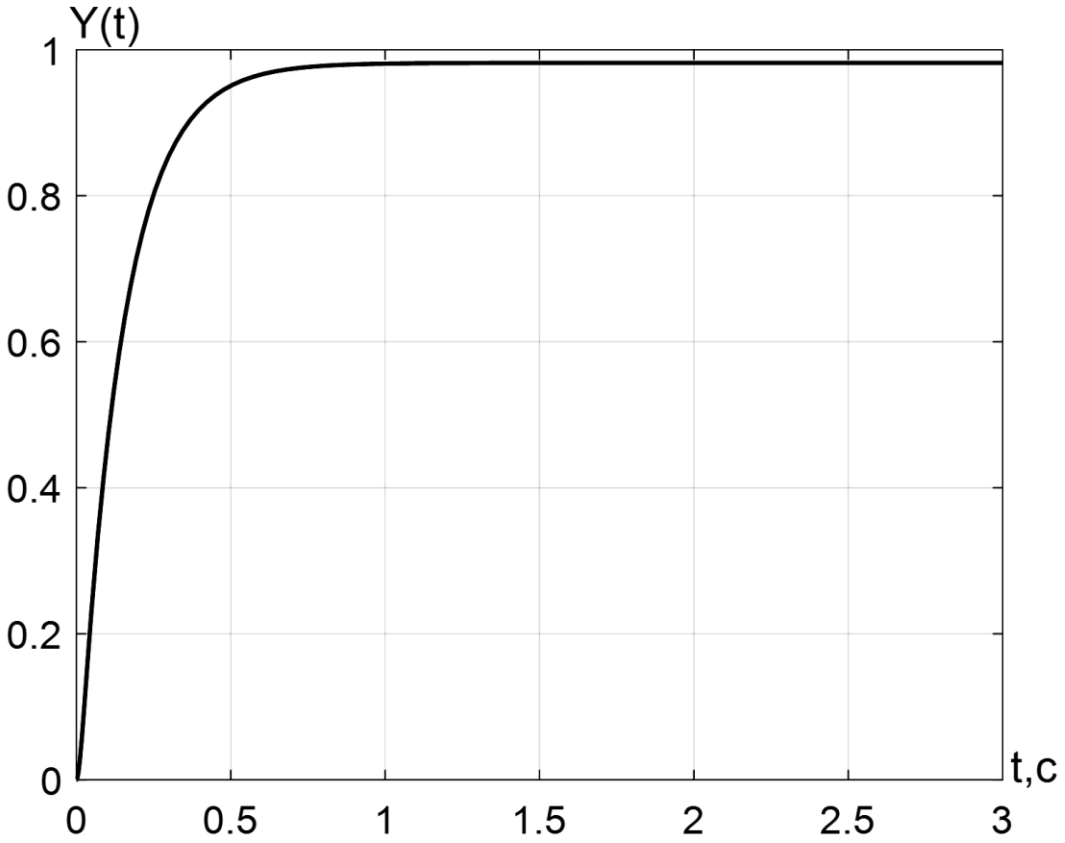


Рисунок 11.24 – Графік перехідного процесу системи управління з ПД-регулятором

Побудуємо поверхню відгуку ПД-регулятора за допомогою команд

```
x1=out.simout(:,1);
x2=out.simout(:,2);
x3=out.simout(:,3);

figure (1)
plot(out.tout,out.yt); grid;

figure(2)
plot(out.tout,x1,out.tout,x2);grid;

figure (3)
[X, Y] = meshgrid(x1, x2);
Z=55*X+8.3*Y;
plot3(X,Y,Z);
figure(4)
plot(out.tout,x3);grid;
```

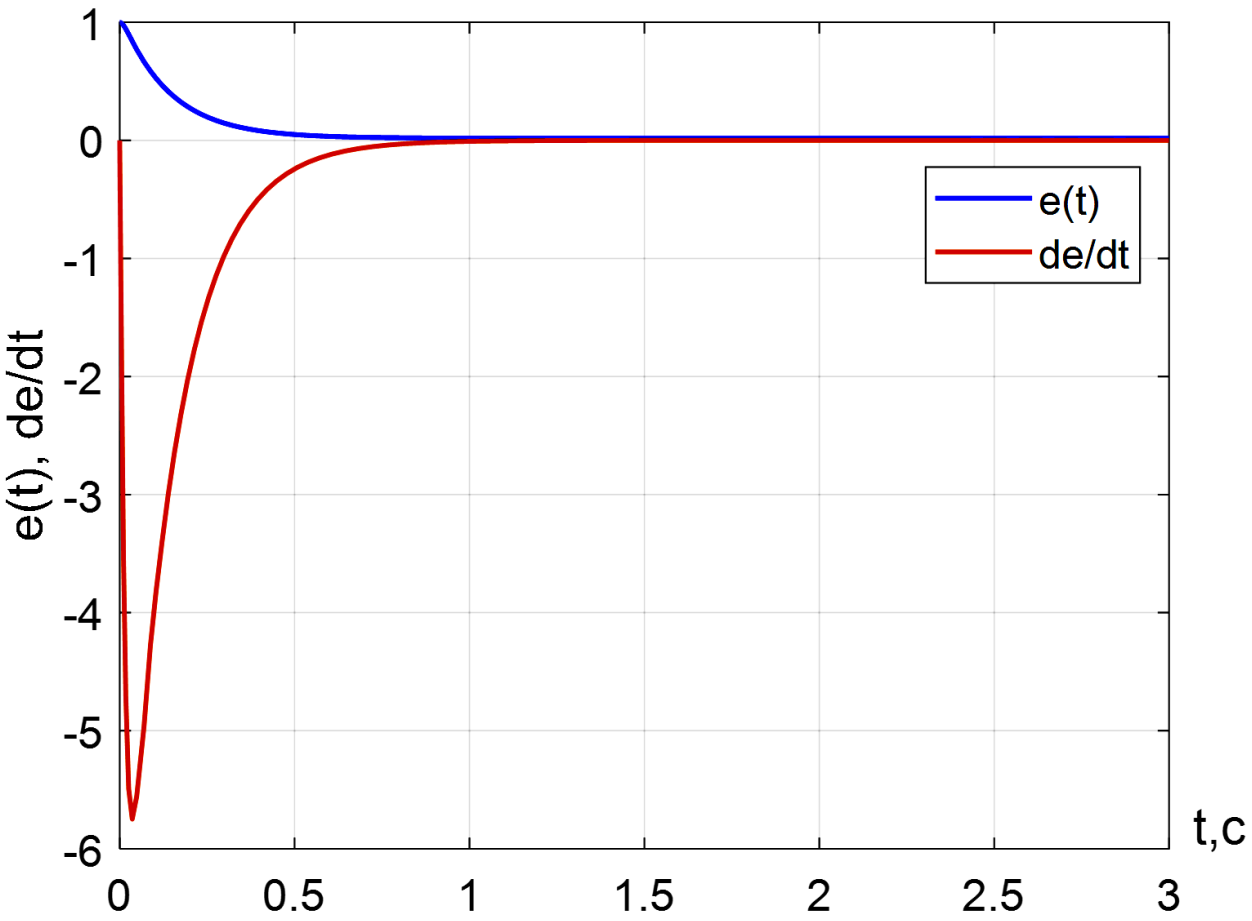


Рисунок 11.25 – Графік зміни помилки та її похідної

Результат виконання програми подано на рис. 11.27. Масштаб горизонтальних осей відповідає діапазону зміни змінних (див. рис. 11.25). Слід підкреслити, що поверхня відгуку є площиною, і будь-яка точка цієї площини завжди відповідає будь-якій вхідної комбінації $e(t)$ та $de(t)/dt$.

Використання блоку out.simout (див. рис. 11.22) дозволяє запам'ятати масив розміром [3×N], у якому перший стовпець відповідає помилці управління $e(t)$, другий – похідної помилки $de(t)/dt$ та третій – сигналу управління $u(t)$.

Таким чином, розглядаючи N моментів часу, отримуємо множину навчальних трійок чисел, що описують рух об'єкта еталонної траєкторії.

Синтез нечіткого регулятора полягає в побудові за цими навчальними даними нечітких правил, так що одне правило має відповідати цілій групі близьких в певному сенсі рядків матриці *simout*.

Виконаємо кластеризацію масиву *simout* за допомогою методу нечітких С-середніх:

```
>> [V, M, obj_fcn] = fcm(simout, 5)
>> V
```

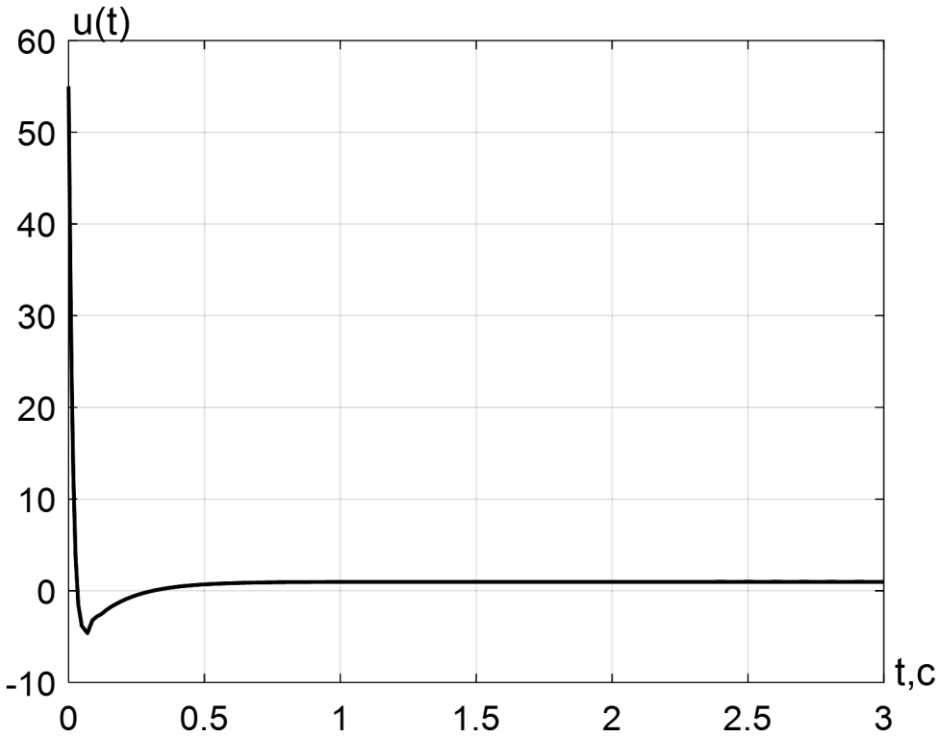


Рисунок 11.26 – Графік зміни сигналу управління за час перехідного процесу

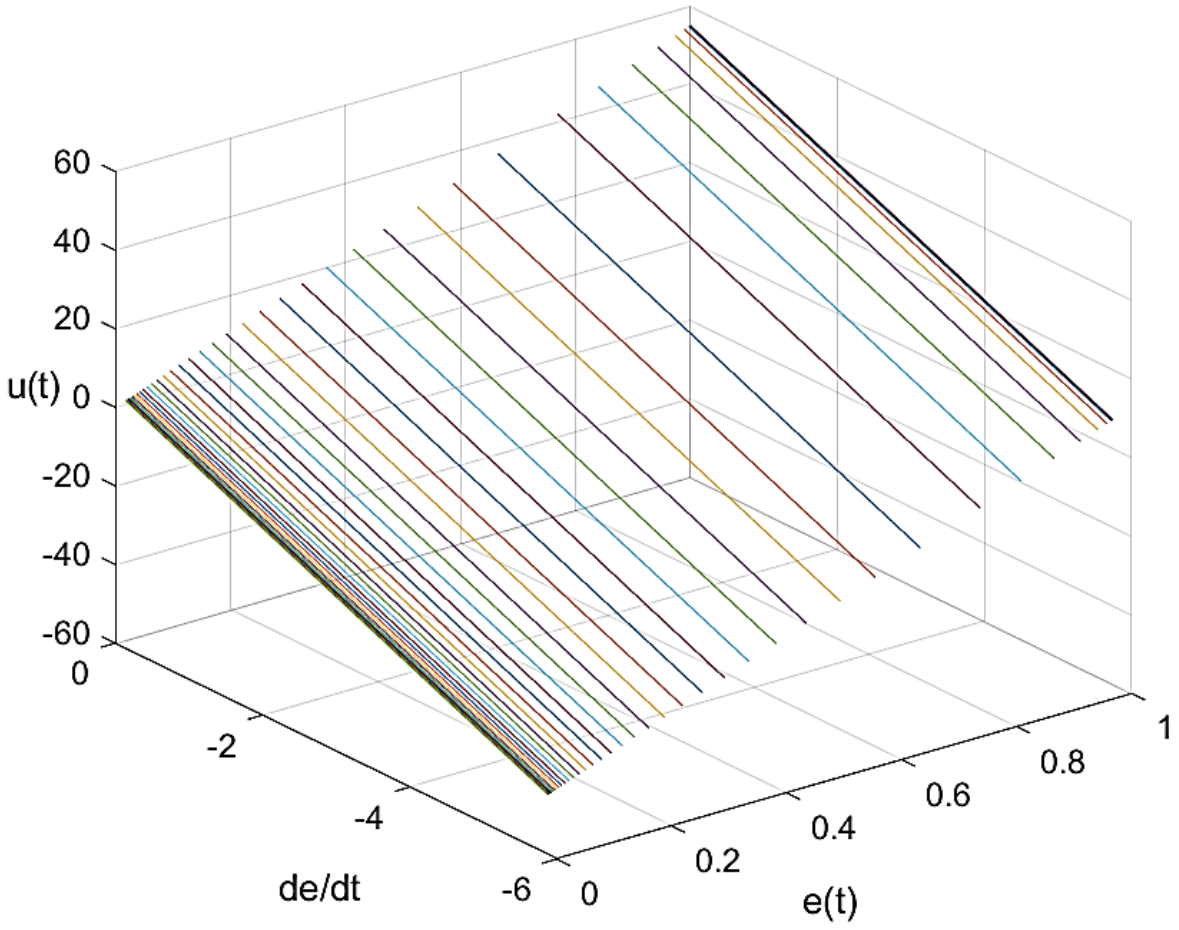


Рисунок 11.27 – Поверхня відгуку ПД-регулятора

V =

0.9999	-0.1131	54.0535
0.0418	-0.1756	0.8412
0.9665	-3.8177	21.4727
0.9909	-2.0456	37.5241
0.5348	-3.8221	-2.3102

Кожен стовпець матриці *V* визначає центри термів відповідної лінгвістичної змінної: 1-й стовпець – «помилка управління», 2-й стовпець – «похідна помилки управління», 3-й стовпець – «сигнал управління».

Налаштуємо редактор системи нечіткого висновку (Fuzzy Inference System – FIS) натисканням - FIS Properties (див. рис. 11.28).

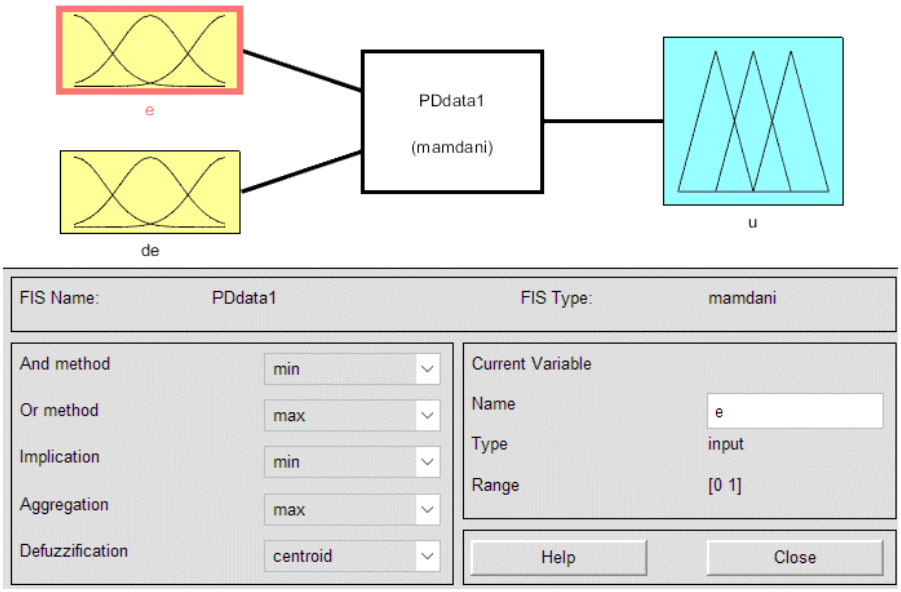


Рисунок 11.28 – Налаштуємо редактор системи нечіткого висновку *FIS editor*

У *FIS editor* задається опис систему нечіткого логічного висновку *Mamdani*. Для створюваної системи обирається вид логічного зв'язку (*And method – min*) та (*Or method – max*), вид імплікації (*Implication – min*), спосіб агрегування висновків правил (*Aggregation – max*) та метод дефазифікації (*Defuzzification – centroid*).

У меню *Edit* послідовно додаємо 2 вхідні змінні «помилка управління», «похідна помилки управління» та вихідну змінну.

Для опису вхідних логічних змінних у редакторі функцій приналежності (*Membership Function Editor*), задаємо для кожної змінної гаусовьку функцію приналежності.

Терми лінгвістичної змінної «Помилка управління» розміщуються відповідно до рис. 11.29 з розміром базової шкали (Range= [0 1]) з 5

параметрами гаусовької функції функцій приналежності, з центрами зазначеними 1-им стовбцем матриці *V*:

Name = "e1"; Type = "gaussmf"; Param = [0.1062 0.9943];
Name = "e2"; Type = "gaussmf"; Param = [0.1062 0.3631];
Name = "e3"; Type = "gaussmf"; Param = [0.1062 0.6634];
Name = "e4"; Type = "gaussmf"; Param = [0.1062 0.0332];
Name = "e5"; Type = "gaussmf"; Param = [0.1062 0.1578].

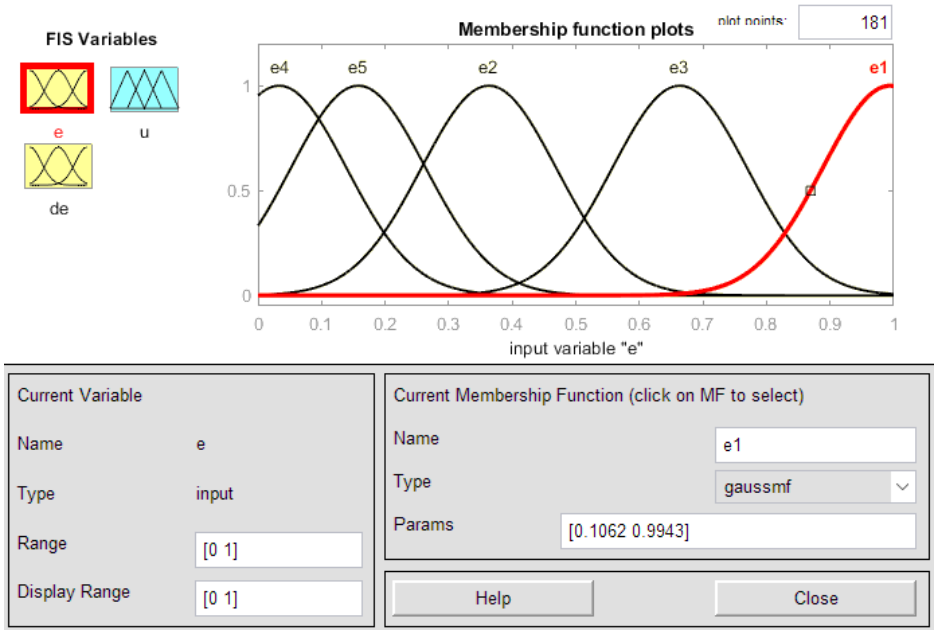


Рисунок 11.29 – Результат налаштування функцій приналежності вхідних змінних «Помилка управління»

Для опису вхідних логічних змінних «Похідна помилки управління» у редакторі функцій приналежності (*Membership Function Editor*), з розміром базової шкали (Range= [-4.8 -0.111]) задаємо для кожної вихідної змінної гаусовську функцію приналежності з центрами зазначеними 2 стовбцем матриці *V*. З параметрами:
Name = "de1"; Type = "gaussmf"; Param = [0.498 -0.569];
Name = "de2"; Type = "gaussmf"; Param = [0.498 -2.553];
Name = "de3"; Type = "gaussmf"; Param = [0.498 -4.756];
Name = "de4"; Type = "gaussmf"; Param = [0.498 -0.1108];
Name = "de5"; Type = "gaussmf"; Param = [0.498 -1.029].

Результат налаштування функцій приналежності вхідних змінних «Похідна помилки управління» на рис. 11.30.

Для опису вихідної логічної змінної у редакторі функцій приналежності (*Membership Function Editor*), з розміром базової шкали (*Range*= [-3 50]), задаємо для кожної вихідної змінної задається гаусовська функція приналежності з центрами зазначеними 3 стовбцем матриці *V*.

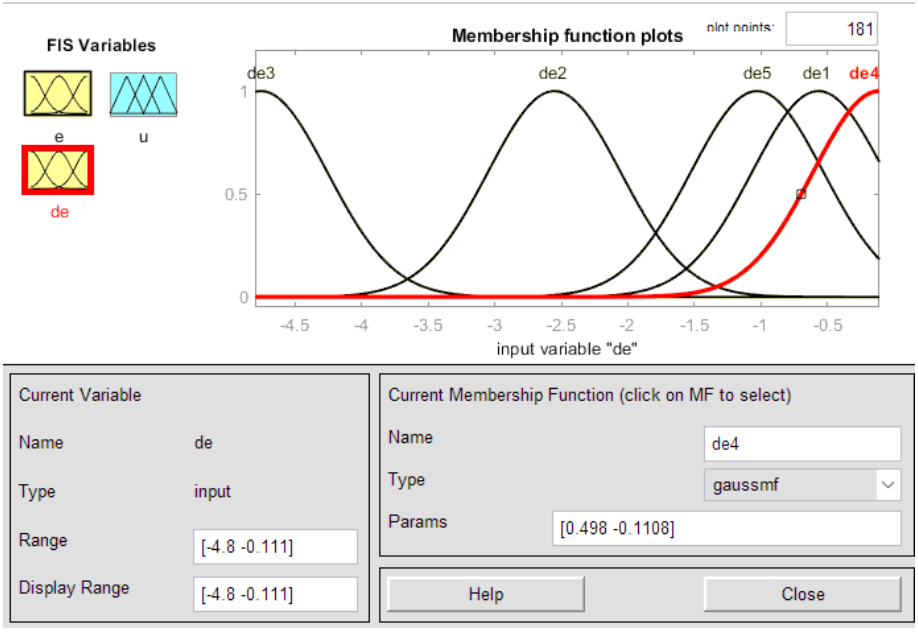


Рисунок 11.30 – Результат налаштування функцій приналежності вхідних змінних «Похідна помилки управління»

Name = "u1"; Type = "gaussmf"; Param = [5.629 49.96];
Name = "u2"; Type = "gaussmf"; Param = [5.629 -1.215];
Name = "u3"; Type = "gaussmf"; Param = [5.629 -2.985];
Name = "u4"; Type = "gaussmf"; Param = [5.629 0.9039];
Name = "u5"; Type = "gaussmf"; Param = [5.629 0.1374].

Результат налаштування функцій приналежності вхідних змінних наведено на рис. 11.31.

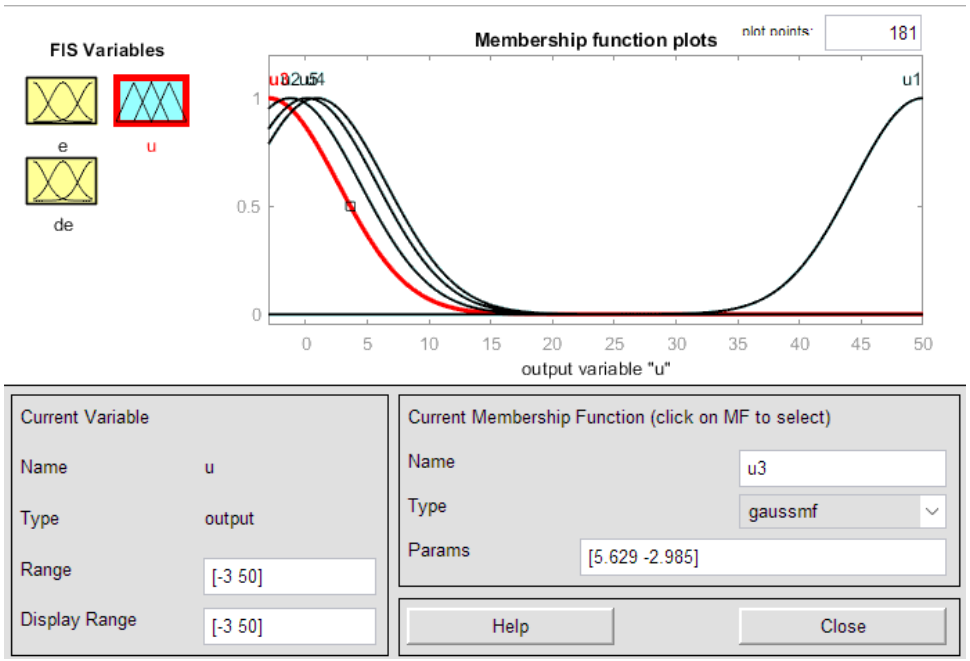


Рисунок 11.31 – Результат налаштування функцій приналежності вихідної змінної

Після опису вхідні та вихідні лінгвістичних змінних, здійснюється опис правил у вікні редактора Rule Editor, Кожен рядок матриці V визначає одне правило управління. Управляючі правила сформульовано згідно з рис. 11.32:

1. if (e is e1) and (de is de1) then (u is u1) (1)
2. if (e is e2) and (de is de2) then (u is u2) (1)
3. if (e is e3) and (de is de3) then (u is u3) (1)
4. if (e is e4) and (de is de4) then (u is u4) (1)
5. if (e is e5) and (de is de5) then (u is u5) (1)

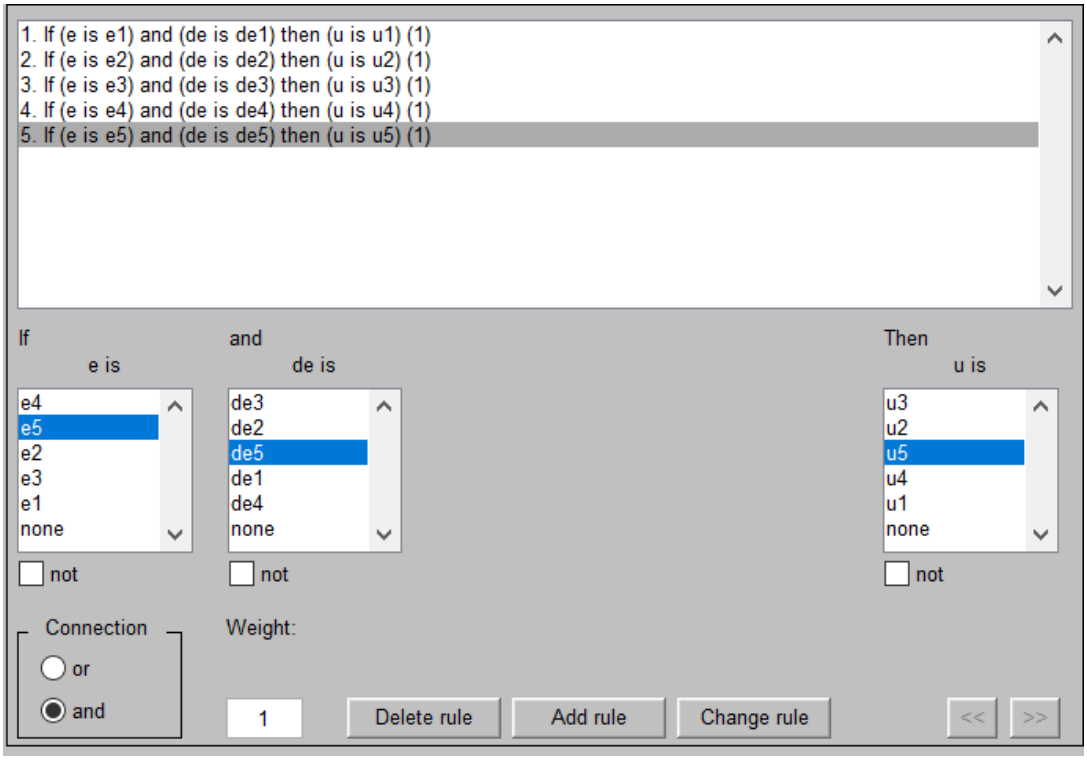


Рисунок 11.32 - Налаштовування опису правил функціонування нечіткого НЛР_ПД за експериментальними даними

Посилки у правилах пов'язані (connection) за допомогою операції *and*. Введене правило забезпечене ваговим коефіцієнтом (*Weight*=1).

Після опису правил можливо за допомогою

- інтерфейсу *Rule Viewer* можливо переглянути роботу системи нечіткого висновку за різних вхідних даних (див. рис. 11.33).

− інтерфейсу Surface Viewer можливо переглянути графічне подання закону управління (рис. 11.34).

Після процедур налаштування нечіткого НЛР_ПД у редакторі системи нечіткого висновку здійснює запис результатів налаштування у Fuzzy Logic Controller. Для цього у вкладці File здійснюємо Export в математичну модуль нечіткого контролера From Workspase вказавши ім’я Fuzzy Logic Controller.

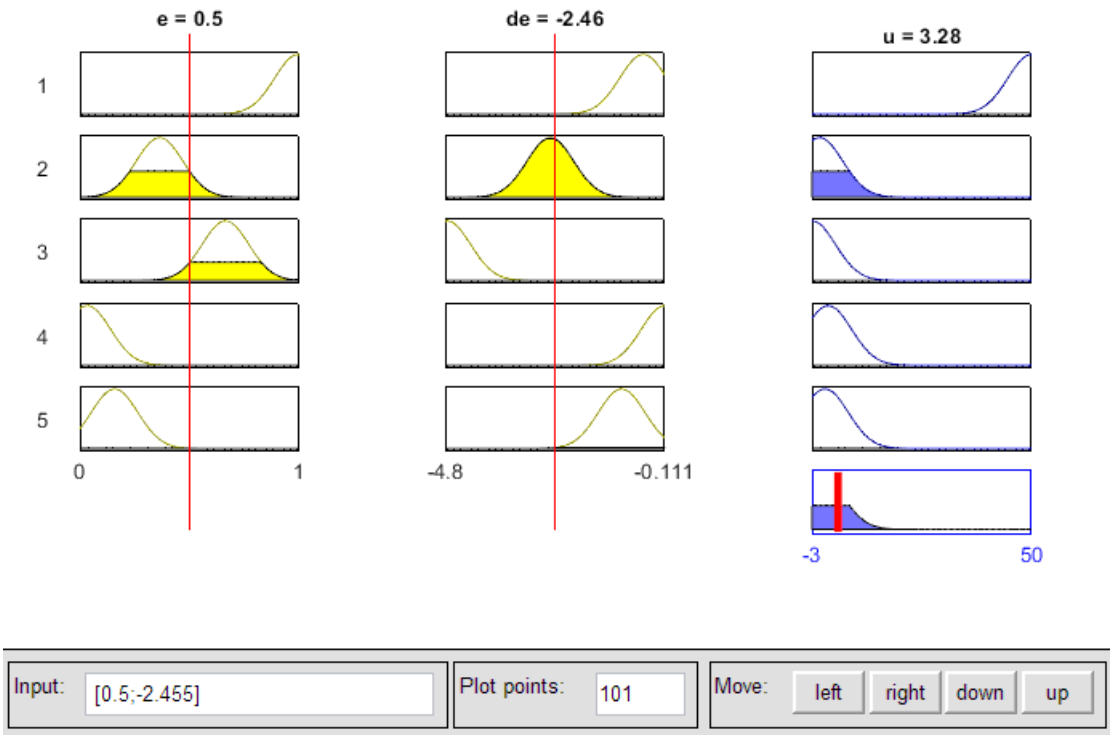


Рисунок 11.33 - Робота системи нечіткого супервізору НЛР_ПІД

На рис. 11.35 представлені графіки перехідного процесу у системі управління з нечітким супервізором НЛР_ПІД.

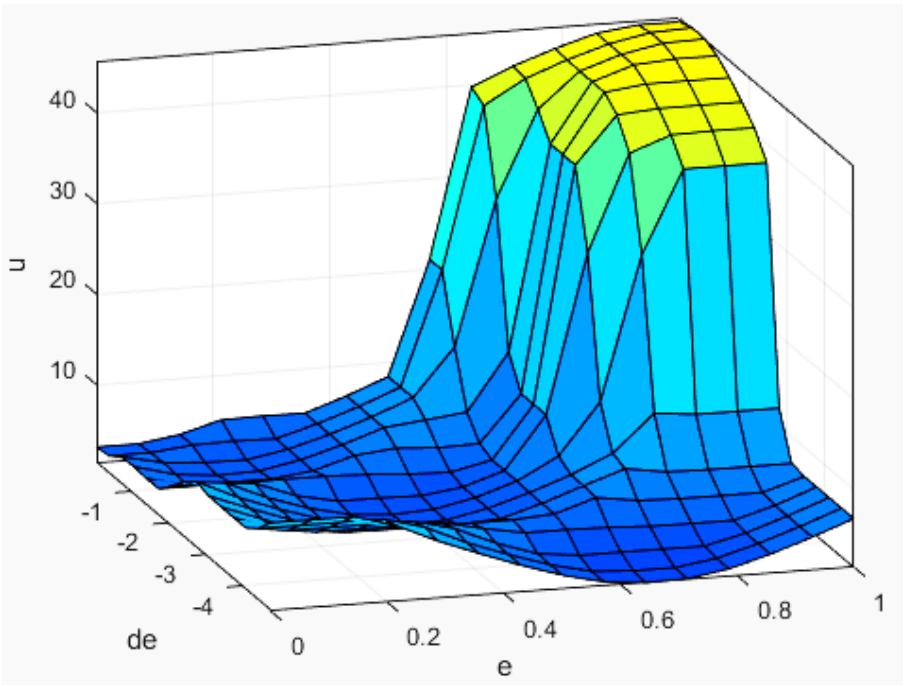


Рисунок 11.34 - Графічне подання закону управління

Очевидно, що опис термів на рис. 11.29 - 11.31 не повністю відповідають вимогам до нечіткої системи, щоб вона могла

апроксимувати будь-яку функцію:

1) функції належності нечітких правил мають бути опуклими та нормальними;

2) нечіткі множини, що описують терми посилок та висновків, повинні утворювати нечітке розбиття відповідних областей визначення;

3) база правил має бути повною.

Автоматично виявляється виконаним лише вимога 1. Вимога 3 також можна вважати виконаною, оскільки при кластеризації генеруються всі комбінації посилок правил, однак без виконання вимоги 2 це не має сенсу.

Синтезована нечітка система управління наведена на рисунку 11.35 за результатом отриманого графіку перехідного процесу виявляється нестійкою (див. рис. 11.36).

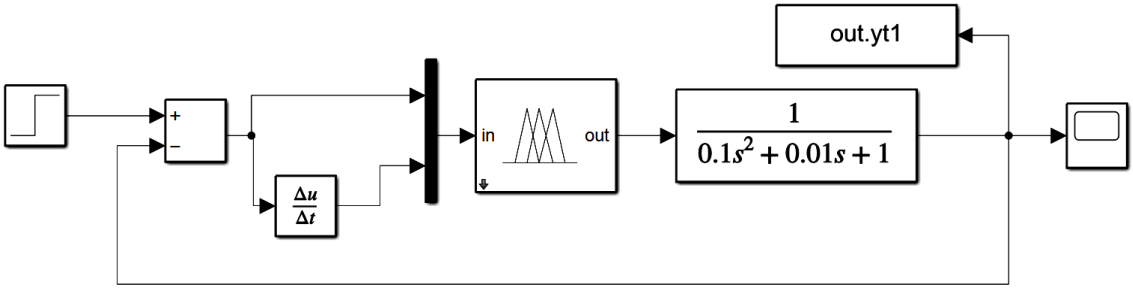


Рисунок 11.35 – Математична модель синтезованої нечіткої системи управління

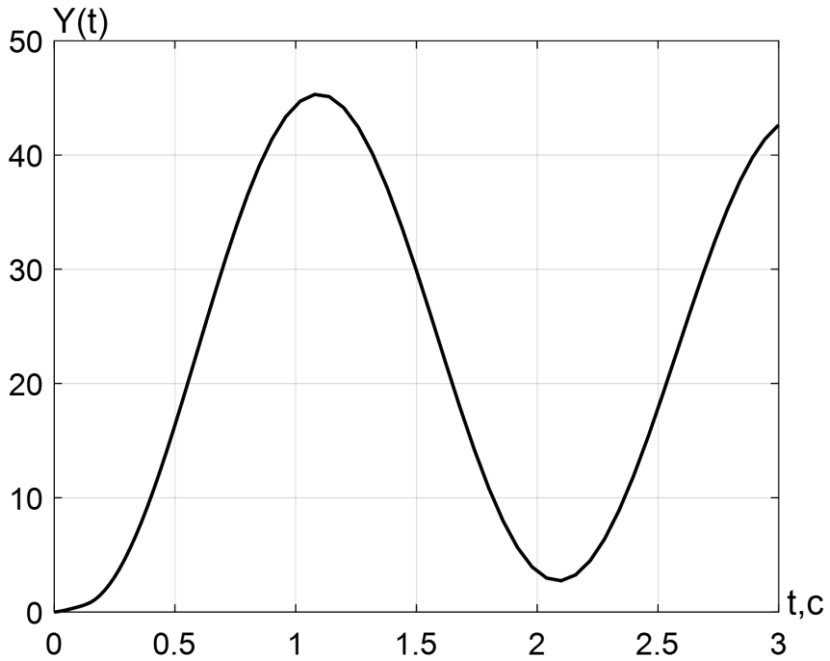


Рисунок 11.36 – Графік перехідного процесу

Наведений приклад показує, що навчальні дані задані графіками рис. 11.25 та 11.26, недостатні для синтезу НЛР шляхом кластеризації.

Для поліпшення якості навчальної вибірки потрібно, щоб у ній були присутні (з деяким кроком) всі варіанти комбінацій вхідних змінних у допустимому діапазоні значень. Такі дані можна використовувати з отриманих масивів експериментальних даних з блоку out.simout (див. рис. 11.22) який формує масив розміром [3×*N*], у якому перший стовпець відповідає помилці управління *e*(*t*), другий – похідної помилки *de*(*t*)/*dt* та третій – сигналу управління *u*(*t*).

Здійснимо синтез НЛР_ПД з використанням повного діапазону значень у модулі.

Викликаємо модуль *ANFIS* командою

```
>> anfisedit
```

У модулі ANFIS завантажуємо базу даних out.simout натисканням кнопки *Load Data*. Попередньо здійснив налаштування: Type – Traning Data; From – worksp. ,як зображено на рисунку 11.37.

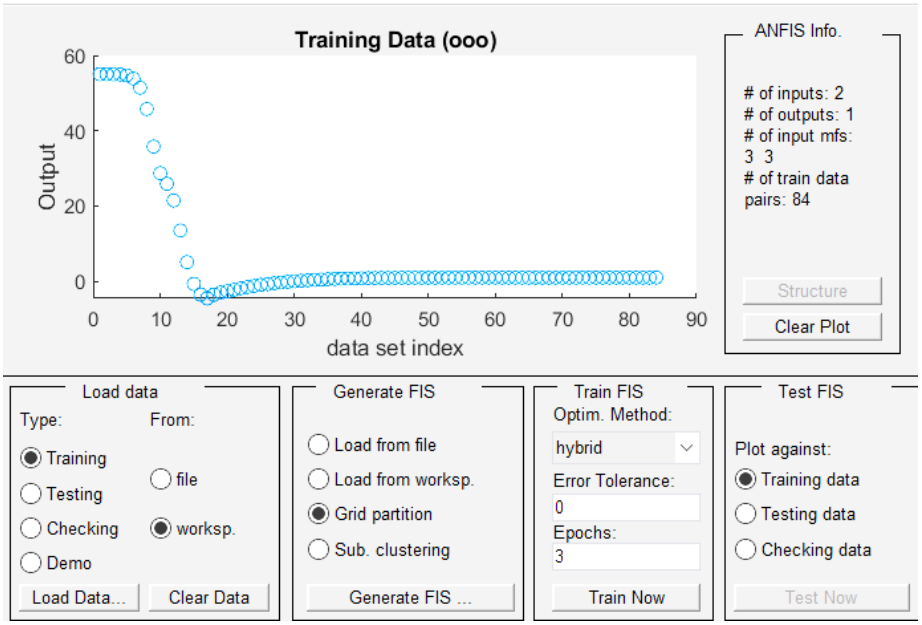


Рисунок 11.37 - Результат завантаження бази даних до модуля ANFIS

Після завантаження даних у меню побудови системи нечіткого логічного висновку *Generate FIS* позначаємо генерування опису вхідних змінних за методом решітки (без кластеризації) - *Grid partition* (див. рис. 11.37).

При виборі *Grid partition* з'являється вікно введення параметрів методу решітки (див. рис. 11.38), в якої вказуємо кількість термів рівної 7 для кожної вхідної змінної та тип функцій приналежності *gaussmf* для вхідних та вихідних змінних (тип функції приналежності може бути іншим в залежності від типу об’єкту управління)

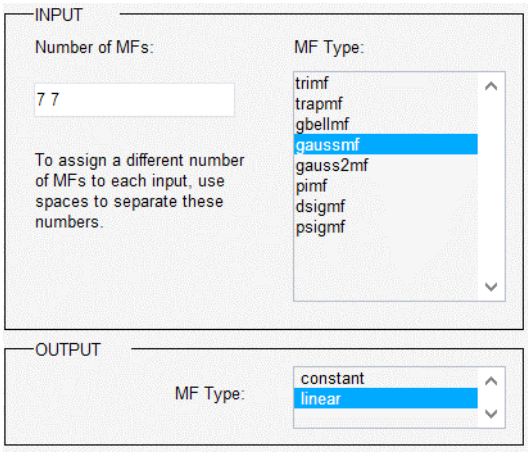


Рисунок 11.38 – Вікно введення параметрів методу решітки

У вікні модуля *ANFIS* натисканням кнопки *Structure* можливо переглянути згенеровану структуру система нечіткого логічного виводу представляє у вигляді нейро-нечіткої мережі (див. рис. 11.39). Для даного випадку кількість входів мережі дорівнює 2 - кількості стовпців навчальної вибірки. Кількість нейронів 2-4 шари відповідає кількості правил яка дорівнює 49.

На наступному кроці проводиться навчання побудованої нейронечіткої моделі. В область навчання (*Train FIS*) обраємо гібридний метод оптимізації (*Optim. Method - Hybrid*), у полі завдання необхідної точності навчання (*Error tolerance*) встановлюємо значення 0, а у полі завдання кількості ітерацій навчання (*Epochs*) встановлюємо 10 епох (див. рис. 11.12).

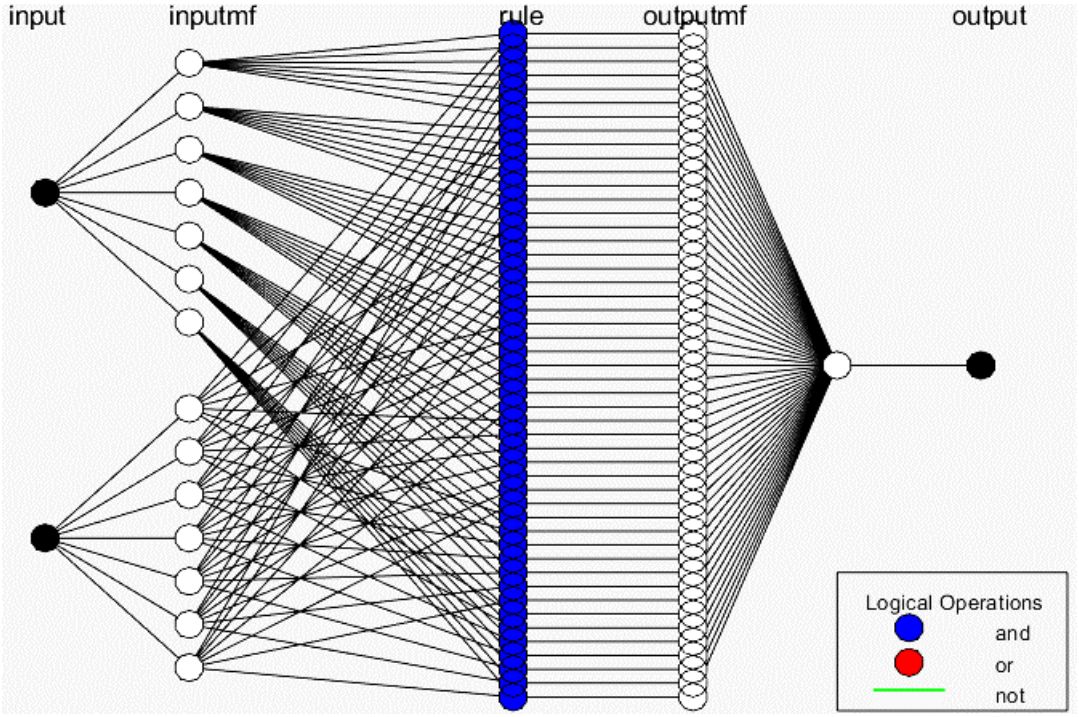


Рисунок 11.39 – Згенерована нейронечітка структура

Проводимо навчання натисканням кнопки *Train Now*. Проміжні результати навчання наведені на рисунку 11.40 виводяться в область візуалізації та робочу область MatLab.

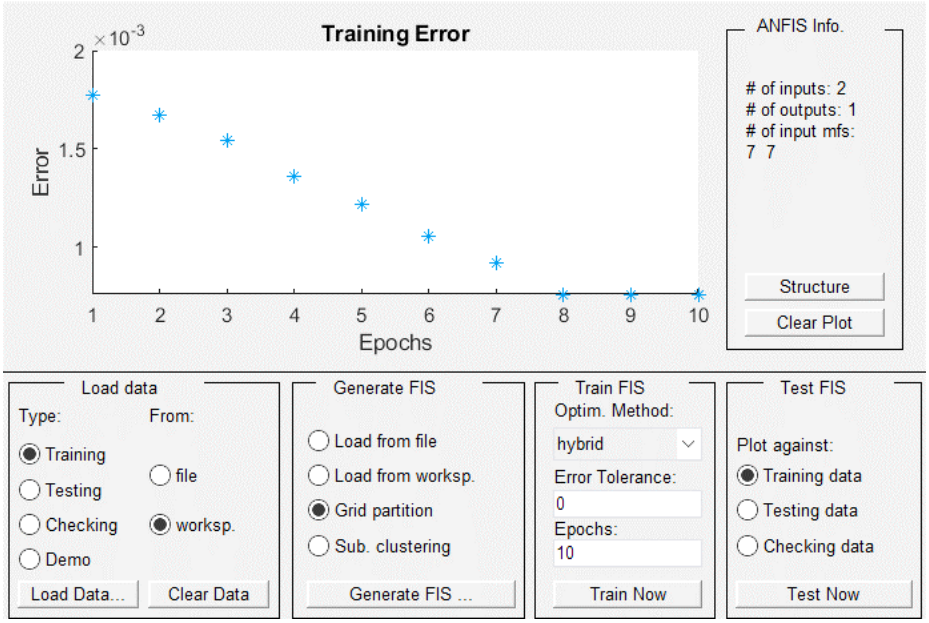


Рисунок 11.40 - Проміжні результати навчання

По завершенню навчання за даними навчання *Training Data* проводитися тестування нейронечіткої системи (*Test FIS*). Після натискання кнопки *Test Now*, відбувається тестування нечіткої системи з виведенням результатів у область візуалізації які наведені на рисунку 11.41.

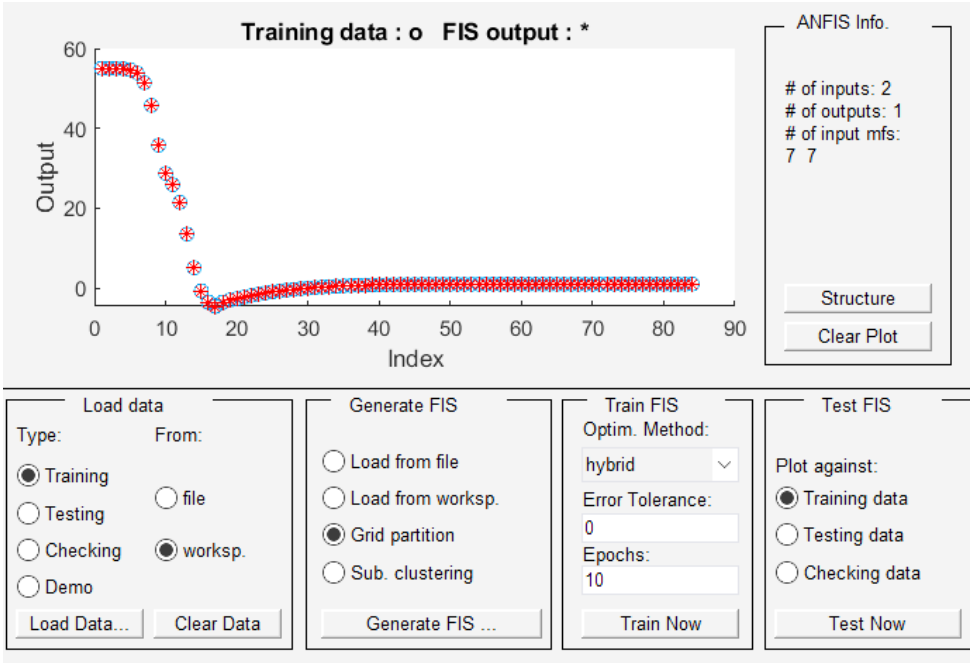


Рисунок 11.41 – Результати тестування нечіткої системи

Після виконання вище наведених операції у вікні редактора модуля ANFIS (див. рис.11.5) в меню Edit викликаємо редактор системи нечіткого висновку (Fuzzy Inference System – FIS) натисканням - FIS Properties (див. рис. 11.42).

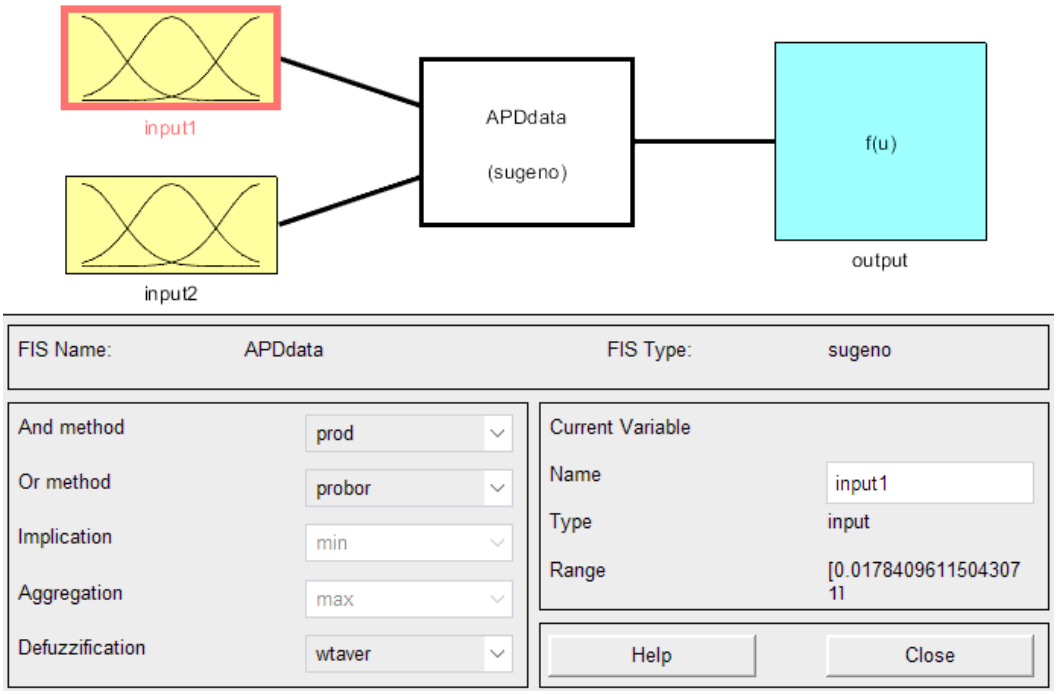


Рисунок 11.42 – Згенерований інтерфейс FIS editor

Опис у синтезовану *FIS editor* відповідає налаштуванням здійсненим *Generate FIS* (див. рис. 11.39).

Для опису вхідних та вихідних логічних змінних у *Generate FIS призначено* гаусовські функцій приналежності.

Результат налаштування функцій приналежності змінних наведено на рис. 11.43.

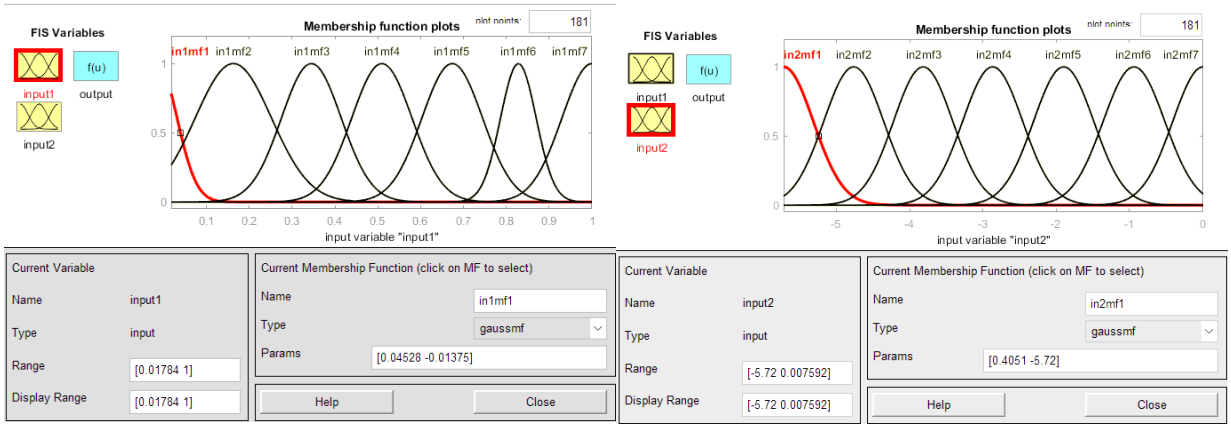


Рисунок 11.43 – Результат налаштування функцій приналежності вхідних змінних

Сформульовані управляючі правила наведено з рис. 11.44.

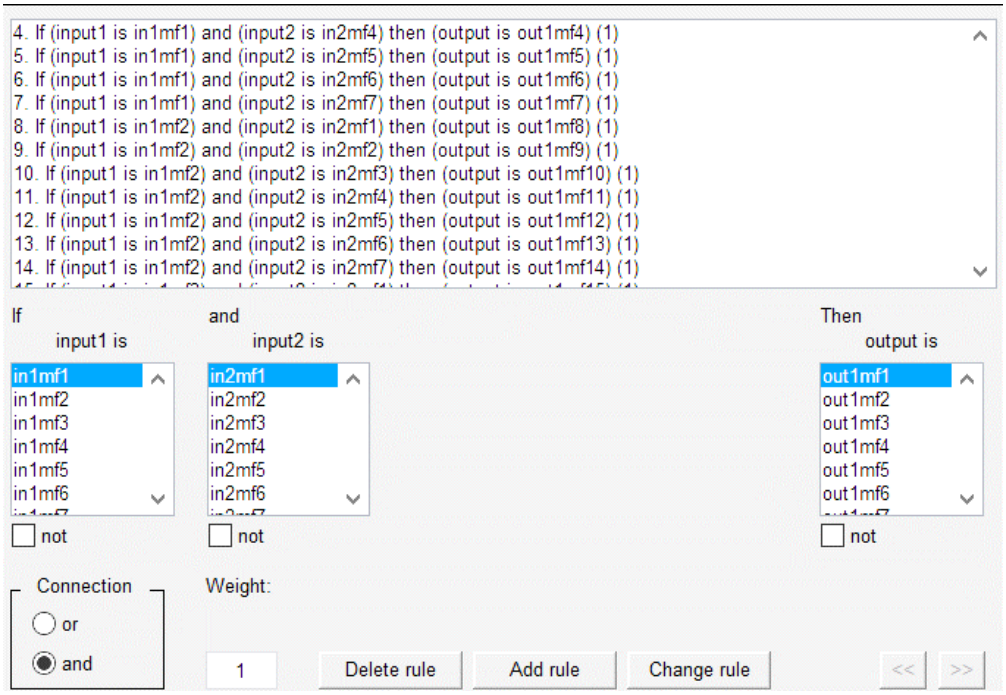


Рисунок 11.44 - Налаштовування опису правил функціонування нечіткого НЛР_ПД

Після опису правил можливо за допомогою

- інтерфейсу *Rule Viewer* можливо переглянути роботу системи нечіткого висновку за різних вхідних даних (див. рис. 11.45).

− інтерфейсу Surface Viewer можливо переглянути графічне подання закону управління (рис. 11.46).

Перевіримо функціонування згенерованої нечіткої системи управління за базою даних . Для цього використовуємо Simulink модель наведену на рисунку 11.35.

У редакторі системи нечіткого висновку здійснює запис результатів налаштування у Fuzzy Logic Controller. Для цього у вкладці File здійснюємо Export в математичну модуль нечіткого контролера From Workspase вказавши ім’я Fuzzy Logic Controller.

На рис. 11.47 показаний перехідний процес під управлінням синтезованого НЛР.

Принцип суперпозиції лінійних систем гарантує, що для лінійного регулятора будь-які дані, що знаходяться поза межами визначення рис. 11.27, викличуть правильну реакцію, тобто коефіцієнти ПД-регулятора описують всю площину, а не лише її окрему частину.

Однак НЛР є нелінійним регулятором, він вимагає розбиття вхідного простору на підобласті, яким відповідає певний сигнал керування. Якщо якась область не описана («біла пляма»), реакції регулятора не буде.

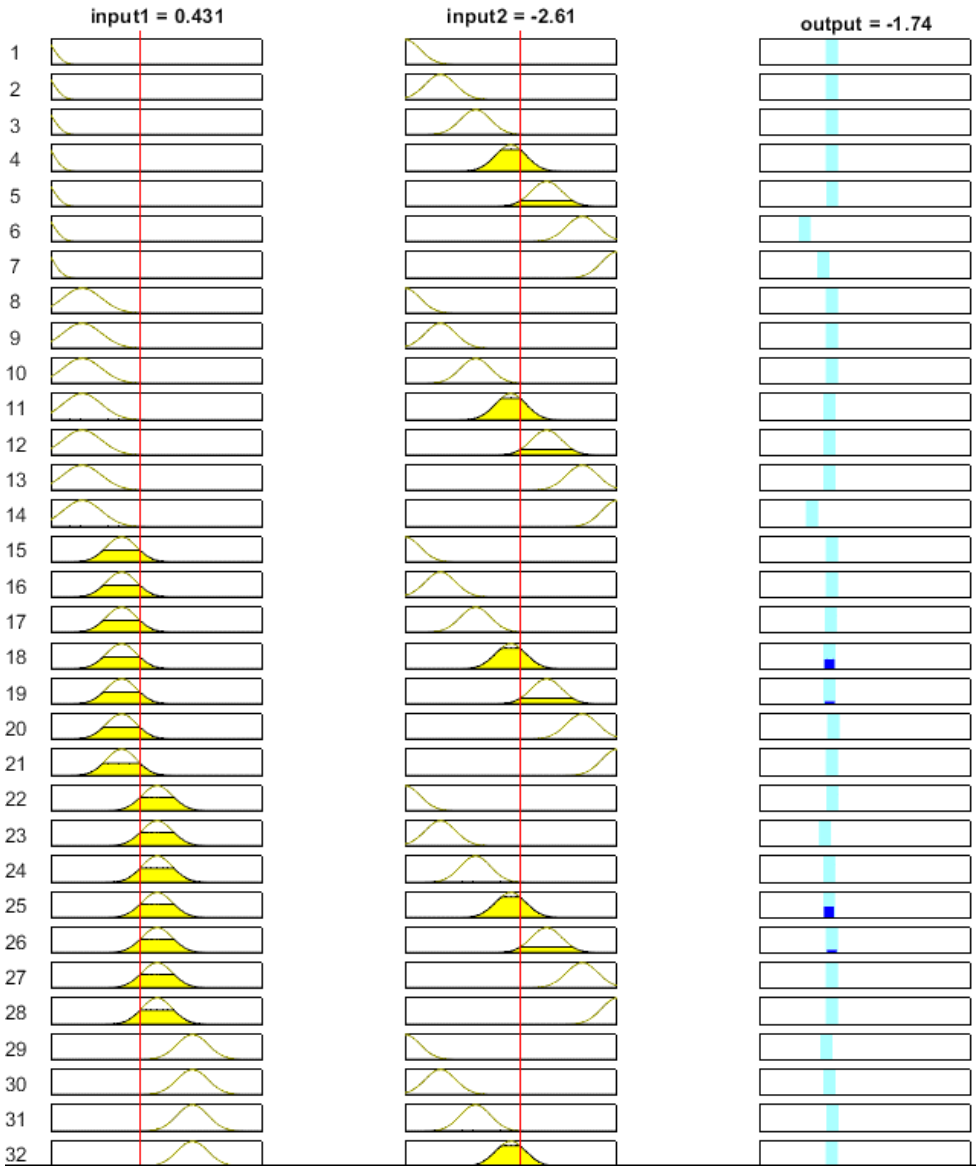


Рисунок 11.45 - Робота системи нечіткого висновку

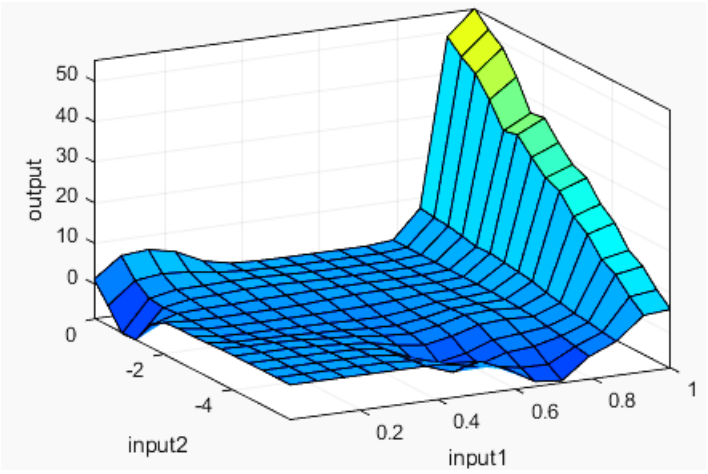


Рисунок 11.46 - Графічне подання закону управління

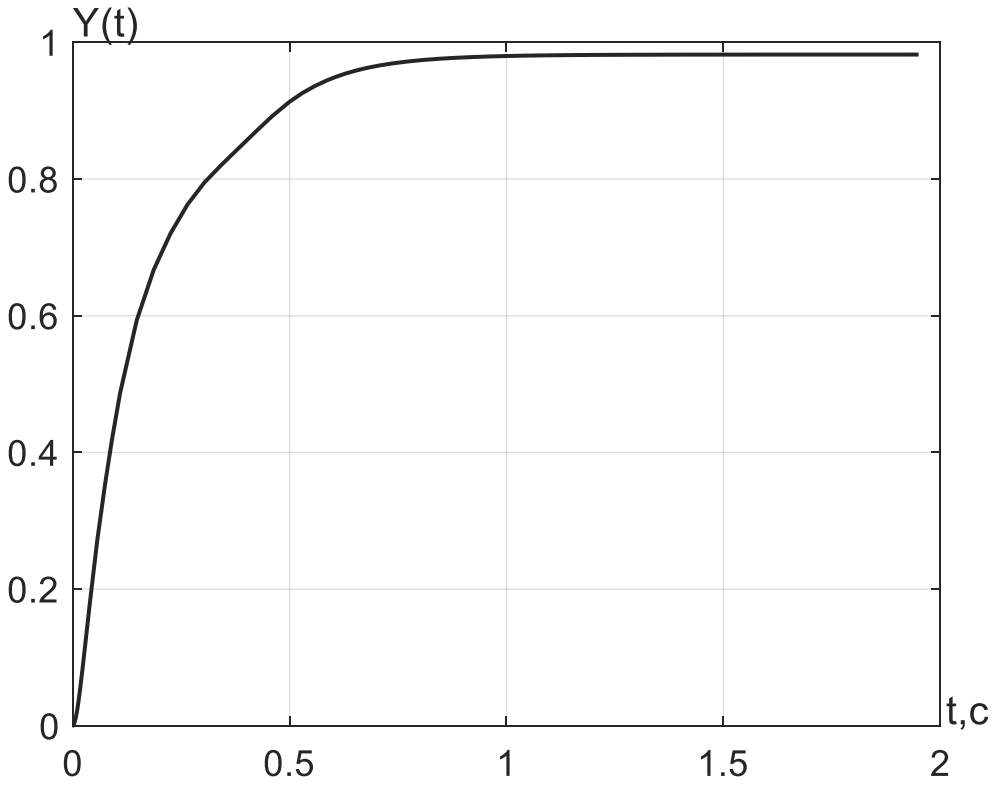


Рисунок 11.47 – Перехідний процес під управлінням синтезованого НЛР

Тому для навчання НЛР дані, отримані при розгляді реакції ПД-регулятора на стрибок, виявляються «поганими», оскільки під час перехідного процесу $e(t)$ та $de(t)/dt$ були уніполярними, і знаходилися у досить вузькому діапазоні значень.

Використана література

1 . Albus J. S., Meystel A. M. Intelligent Systems: Architecture, Design, and Control / Wiley, New York, 2002.

2 A. P. Engelbrecht. Computational Intelligence: An Introduction / Wiley, Chichester, U.K., 2002.

3 Badiru A. B., Cheung J. Y. Fuzzy Engineering Expert Systems with Neural Network Applications / John Wiley, New York, NY, 2002.

4 Апостолюк В. О. Інтелектуальні системи керування: конспект лекцій [Текст] / В. О. Апостолюк, О. С. Апостолюк. – К.: НТУУ «КПІ», 2008. – 88 с.

5 Антоненко В. М. Сучасні інформаційні системи і технології: управління знаннями : навчальний посібник / В. М. Антоненко, С. Д. Мамченко, Ю. В. Рогушина. – Ірпінь : Національний університет ДПС України, 2016. – 212 с. ISBN 978-966-337-418-5

6 Інтелектуальні системи управління: Експертні системи - основи проектування та застосування в системах автоматизації [Електронний ресурс] : навч. посіб. для студ. спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» / КПІ ім. Ігоря Сікорського; уклад.: Л. Д. Ярощук. – Електронні текстові дані (1 файл: 2,56 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2019. – 136с.

7 Giese H., Rumpe B. Science and Engineering of Cyber-Physical Systems (Dagstuhl Seminar 11441), Dagstuhl Reports, vol. 1, no. 11, pp. 1–22, 2012.

8 Conti M. Looking ahead in pervasive computing: challenges and opportunities in the era of cyber-physical convergence," Pervasive and Mobile Computing, 2011.

9 Sha L., Gopalakrishnan S. Cyber-physical systems: A new frontier, Machine Learning in Cyber Trust, pp. 3–13, 2009.

10 Horv´ath I., Gerritsen B. Cyber-physical systems: Concepts, technologies and implementation principles, in Tools and Methods of Competitive Engineering Symposium (TMCE), 2012, pp. 19–36.

11 Lee E., "Computing needs time," Communications of the ACM, vol. 52, no. 5, pp. 70–79, 2009.