РОЗДІЛ 1
СУЧАСНІ ІНТЕЛЕКТУАЛЬНІ ТЕХНОЛОГІЇ В СИСТЕМАХ УПРАВЛІННЯ

У різних сферах практичної діяльності, пов’язаних з вирішенням завдань управління та контролю, оптимізації та моделювання, пошуку та вибору, розпізнавання та класифікації [1], гостро постала необхідність інтелектуальної підтримки для подолання труднощів в складних ситуаціях і при обмежених ресурсах. Методи та засоби штучного інтелекту доходять до споживача у вигляді інтелектуальних технологій, які практично інваріантні до тієї чи іншої проблемної області [2]. Традиційно до ІТ відносять нечітку логіку (НЛ), генетичні алгоритми (ГА) і нейронні мережі (НМ). ІТ успішно використовуються для створенні складних систем управління. Вимоги до пристроїв управління – забезпечувати надійне управління об'єктом в різних режимах його роботи [3], бути стійким як до різких змін, так і до повільної деградації параметрів системи управління, враховувати можливу наявність шумів і зовнішніх передбачених і непередбачених впливів.

1.1 Експертні системи

До експертних систем (ЕС) відносять системи, що базуються на знаннях, тобто системи, обчислювальна можливість яких є наслідком їх нарощуваної бази знань і тільки в другу чергу визначається методами, які використовуються. Методи інженерії знань (методи ЄС) значною мірою, незалежно від того, в яких областях вони можуть застосовуватися: медицина, військові технології, обчислювальна техніка, електроніка, сільське господарство, математика, космос, менеджмент, бізнес, право, мають спільні принципи та підходи. Зараз ЄС використовуються при вирішення завдань наступних типів: прийняття рішень в умовах невизначеності (неповноти), трактування символів і сигналів, прогнози, діагностика, конструювання, планування, керування, контроль [4].

1.1.1. Структура експертної системи

Базовими компонентами будь-якої ЕС є (див. рис. 1.1): вирішувач (інтерпретатор), робоча пам'ять (РП), яка може мати назву база даних (БД), база знань (БЗ), елементи набуття знань, діалоговий і пояснювальний блоки.

База даних використовуються для зберігання даних (вихідних і проміжних) розв'язуваного завдання [5]. Термін цілковито співпадає за назвою, але не за сенсом з терміном, що використовуються в інформаційно-пошукових системах (ІПС) і системах керування базами даних (СКБД) [6] для маркування всіх наявних даних, що знаходиться в системі. База знань в ЕС використовується для тривалого зберігання даних, що характеризують досліджувану область (а не поточних даних) [6], і умов, які передбачають доцільні зміни даних цієї області.

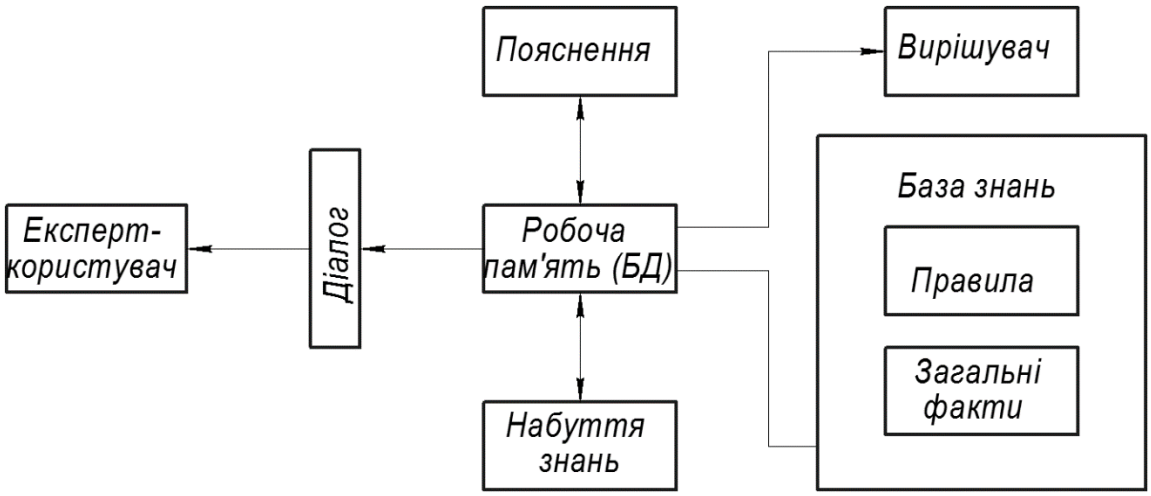

Рисунок 1.1 – Схема узагальненої експертної системи

Вирішувач, оперуючи вихідними даними з БД і знаннями з БЗ, створює необхідну послідовність умов, які, використовуються до вихідних даних [6], призводять до рішення задачі. Компонента придбання знань автоматизує процес наповнення ЕС знаннями, здійснюваний користувачем. Пояснювальна компонента пояснює, як система одержала вирішення задачі (або відсутність рішення) та які знання при цьому вона використала, що спрощує експерт-

тестування системи та збільшує довіру користувача до одержання результату. Діалогова компонента орієнтована на доброзичливе спілкування з усіма категоріями користувачів як під час вирішення завдань [6], так і під час набуття знань, пояснення результатів.

До створення ЕС долучаються представники широкого спектру спеціальностей: експерт проблемної галузі, проблеми якої буде вирішувати ЕС; інженер-когнітолог – фахівець з розробки ЕС; програміст – фахівець з розробки інструментальних засобів (ІТЗ) [7]. Потрібно відзначити, що не включення в перелік учасників розробки когнітолога найчастіше призводить до невдачі в процесі розробки ЕС або значно збільшує термін можливої реалізації.

Значна кількість існуючих ЕС базується на припущенні незмінності предметної галузі та вирішують лише статичні завдання [8], такі системи носять назву статичних. ЕС, які використовуються для роботи з динамічними предметними галузі та займаються вирішенням статичних або динамічних завдань, називаються динамічними. Вирішення найголовніших практичних неструктурованих завдань може бути тільки за використання динамічних, а не статичних ЕС. Статична ЕС співпадає з традиційною схемою (див. рис. 1.1) [9].

1.1.2. Системи нечіткого виводу

Головне місце в нечіткій логіці та відповідних системах керування посідає нечіткий висновок. Процес нечіткого виведення являє собою послідовність здобуття нечітких висновків з урахуванням нечітких умов або передумов за допомогою використання понять і принципів нечіткої логіки. Цей процес об'єднує в собі всі основні концепції теорії нечітких множин: функції приналежності, лінгвістичні змінні, нечіткі логічні операції, методи нечіткої імплікації та нечіткі композиції [10].

Системи нечіткого виводу використовуються для реалізації процесу нечіткого виведення і служать світоглядним базисом усієї існуючої нечіткої логіки. Досягнуті успіхи в застосуванні цих систем для вирішення значного

класу задач керування стали основою становлення нечіткої логіки як прикладної науки з широким спектром додатків. Системи нечіткого виводу дозволяють вирішувати задачі автоматичного управління, групування даних, розрізнення образів, прийняття рішень, роботизованого навчання та багато інших.

Враховуючи те що розробка та використання систем нечіткого виводу має міжгалузевий характер, даний напрямок досліджень тісно пов'язаний з широким колом інших науково-прикладних напрямків, таких як: нечітке моделювання, нечіткі експертні системи, нечітка асоціативна пам'ять, нечіткі логічні контролери, нечіткі регулятори та просто нечіткі системи [11].

1.1.3. Базова архітектура систем нечіткого виводу

Системи нечіткого виводу – це окремий випадок продукційних нечітких систем або систем нечітких правил продукції, в яких вимоги та висновок окремих правил формуються в вигляді нечітких висловлювань за значеннями, так як ті чи інші лінгвістичні висловлювання мають базисне значення в контексті існуючої нечіткої логіки.

Нечіткими лінгвістичними висловлюваннями називаються висловлювання наступних видів [12].

Вислів «$b \in a$», де b – назва лінгвістичної змінної, a – відповідне значення, якому відповідає окремий лінгвістичний терм з базової терм-множини T лінгвістичної змінної b.

Вираз «$b \in Na$», де N – це модифікатор, який відповідає виразам: «ДУЖЕ», «БІЛЬШ АБО МЕНШ», «БАГАТО БІЛЬШЕ» та іншим, які отримуються за допомогою процедур G і M даної лінгвістичної змінної.

Частини висловлювання, отримані з висловлювань попередніх видів і нечітких логічних операцій в формі зв'язок: «І», «АБО», «ЯКЩО-ТО», «НЕ».

З урахуванням, що в системах нечіткого виведення нечіткі лінгвістичні висловлювання займають головне місце, вигідно використовувати адаптоване

формулювання «нечіткі висловлювання».

Прикладом нечіткого висловлювання слугує «тиск мастила високий», в рамках якого лінгвістичної змінної «тиск мастила» присвоюється значення «високий». При цьому передбачається, що на універсальній множині X змінної «тиск мастила» визначено відповідний лінгвістичний терм «високий» модифікатором «ДУЖЕ», який змінює відповідний лінгвістичний терм «високий» на основі використання деякої розрахункової формули, наприклад для операції концентрації CON(A) нечіткого безлічі A для терма «високий».

Змінної «тиск мастила» присвоюється значення «високий», а іншій лінгвістичній змінній «відстань до пресової матриці» присвоюється значення «близьке». Ці нечіткі висловлювання з'єднані логічною операцією нечітка кон'юнкція (операції нечітке «І»).

Правила нечітких продукцій в системах нечіткого виведення.

Система нечіткого виведення виступає окремим прикладом продукційних нечітких систем або систем нечітких умов продукції. Основна особливість нечітких правил формуються у вигляді нечітких висловлювань щодо значень тих чи інших лінгвістичних змінних.

Спрощений варіант правила нечіткої продукції, який функціонує в системах нечіткого виведення:

$$RULE\ <\#>:\ IF\ "\beta_1\ \epsilon\ \alpha'",\ THEN\ "\beta_2\ \epsilon\ \alpha''". \qquad (1.1)$$

Нечітке висловлювання $"\beta_1\ \epsilon\ \alpha'"$ вимога даного правила нечіткої продукції, а нечітке висловлювання $"\beta_2\ \epsilon\ \alpha''"$ – нечітке закінчення вказаного правила. При цьому $\beta_1 \neq \beta_2$.

Основні процедури що виконуються в системах нечіткого виведення.

Фазифікація – процедура заміни чіткої вхідної змінної нечіткою формою за рахунок функцій належності [13].

Завданням фазифікації є визначення відповідності між визначеним (зазвичай – чисельним) значенням деякої вхідної змінної системи нечіткого

виведення та значенням функції приналежності відповідного їй терма вхідної лінгвістичної змінної. По завершенню цього етапу для всіх вхідних змінних повинні мати конкретні значення функцій приналежності за кожним з лінгвістичних термів, що задіяні в проміжних умовах бази правил системи нечіткого виведення.

Агрегування є процедурою визначення ступеню достовірності умов за кожним з правил пропонованої системи нечіткого виведення [13].

Процедура агрегування здійснюється наступним чином. На початку даного етапу відбувається встановлення значень достовірності всіх проміжних систем нечіткого виведення, тобто безліч значень $B = \{b_i'\}$. Наступним є перевірка кожної з умов правил системи нечіткого виведення. Якщо умова правила визначена як «нечітке висловлювання», то ступінь його достовірності дорівнює відповідному значенню b_i'.

Якщо умова складається з декількох проміжних умов, при цьому лінгвістичні змінні в проміжних умовах попарно не дорівнюють один одному, то визначається ступінь достовірності складного висловлювання на базі відомих значень достовірності проміжних умов. Для визначення результату нечіткої кон'юнкції або зв'язки "І" використовується формула перетину нечітких множин, а для визначення показника нечіткої диз'юнкції або зв'язки «АБО» може бути використана формула сумації нечітких множин. За цієї умови b_i' використовується як аргумент відповідних логічних операцій. За рахунок цього знаходяться кількісні значення достовірності всіх умов правил системи нечіткого виведення.

Закінчення етапу агрегування відбувається тоді, коли будуть визначені всі значення b_i'' для кожного з правил, які включені до даної бази правил системи нечіткого виведення.

Активізація – це процедура чи процес визначення ступеня достовірності всіх висновків правил нечітких продукцій [14]. Активізація в цілому суттєво

подібна до композиції нечітких відносин, але не еквівалентна їй. У системах нечіткого виведення застосовуються лінгвістичні змінні, котрі втрачають своє значення як результат формули для нечіткої композиції. При створенні баз правил системи нечіткого виведення визначаються вагові коефіцієнти F_i для кожного правила. Процедура активізації здійснюється за наступною послідовністю. До виконання цього етапу необхідним є визначення достовірності значення всіх правил системи нечіткого виведення, тобто значень $B'' = \{b_i''\}$ і значення вагових коефіцієнтів F_i для кожного правила. Наступним розглядається кожне з рішень правил системи нечіткого виведення.

У випадку, коли висновок складається з кількох проміжних висновків, причому лінгвістичні змінні в проміжних висновках попарно не рівні один одному, то ступінь достовірності кожного з проміжних висновків дорівнює алгебраїчному добутку відповідного значення b_i'' на ваговий коефіцієнт F_i. За цим методом визначаються всі значення ступенів достовірності проміжних висновків для кожного з правил, що включені в базу правил системи нечіткого виведення. Ці значення позначаються як $C = \{c_1, c_2, ..., c_q\}$, де q – сумарна кількість проміжних висновків в базі правил.

По завершенню етапу пошуку значень $C = \{c_1, c_2, ..., c_q\}$ визначаються функції належності кожного з проміжних висновків для досліджуваних вихідних лінгвістичних змінних. За для цього можливо використання одного з існуючих методів, які є модифікацією методів нечіткої композиції:

min-активізації $$\mu'(y) = \min\{c_i, \mu(y)\} \quad (1.2)$$

prod- активізації: $$\mu'(y) = c_i \cdot \mu(y) \quad (1.3)$$

average- активізації: $$\mu'(y) = 0{,}5 + (c_i + \mu(y)) \quad (1.4)$$

Етап активізації є завершеним, у випадку коли для всіх вихідних лінгвістичних змінних, що є складовими в кожному конкретному проміжному висновку правил нечітких продукцій, отримають значення функції приналежності нечітких множин їх еквівалентів, сукупність нечітких множин:

$c_1, c_2, ..., c_q$, де q загальна кількість проміжних висновків в базі правил системи нечіткого виведення.

Акумуляція є процедурою або процесом знаходження функції для кожної з вихідних лінгвістичних змінних множин [15] $W = \{w_1, w_2, ..., w_q\}$. Пріоритетним завданням акумуляції є поєднання або акумулювання всіх ступенів достовірності висновків (проміжних висновків) для отримання функції причетності кожної з вихідних змінних. Необхідність виконання цього етапу полягає в тому щоб проміжні висновки, що належать до відповідних вихідних лінгвістичних змінних, відповідали різним правилам системи нечіткого виведення.

Послідовність проведення процедури акумуляції наступна. На початку етапу передбачаються відомі значення достовірності всіх підзвітних для кожного з правил R_k, що знаходяться у розглянутій основі правил R системи нечіткого виводу, у формі сукупності нечітких множин: C_1, C_2, ..., C_q, де q – загальна кількість підключених у базі правила далеко наслідок аналізується кожна з вихідних лінгвістичних змінних w_j W і відноситься до неймовірних множин: C_{j1}, C_{j2}, ..., C_{jq}. Результат етапу для вихідної лінгвістичної змінної w_j визначає як з'єднання нечітких множин C_{j1}, C_{j2}, ..., C_{jq} за загальною формулою.

Акумуляція є закінченою у випадку, коли для всіх вихідних лінгвістичних змінних будуть визначені результуючі функції приналежності нечітких множин їх значень. Сукупність нечітких множин: C1 ', C2', ..., Cs ', де s – загальна кількість вихідних лінгвістичних змінних в базі правил системи нечіткого виведення.

1.2 Нейронні мережі

Нейронна мережа (НМ) – це розподілений паралельний процес, що складається з елементарних одиниць обробки інформації [16], що накопичують

експериментальні знання та надають їх для подальшої обробки. Нейронна мережа отримує інформацію з навколишнього середовища і використовує їх в процесі навчання. Для накопичення знань застосовуються зв'язки між нейронами, які носять назву синоптичних ваг.

1.2.1 Структура і функціонування одиночного нейрона

Базова одиниця нервової системи – нервова клітина, що носить назву нейрон [17]. На рис. 1.2 зображена спрощена модель одиничного елементу нервової системи.

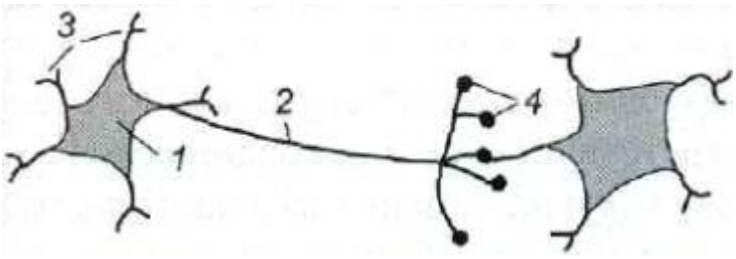

Рисунок 1.2 – Модель нейрона і його з'єднання з сусіднім нейроном: 1 – тіло клітини; 2 – аксон; 3 – дендрити; 4 – синапси

Складовими частинами нейрона є тіло клітини та два види відростків: вхідні (дендрити) і вихідні (аксони). Кількість вихідних відростків є фіксованою та складає один канал, за допомогою якого відбувається передача інформації до інших нейронів [18].

У загальному випадку нейрон сприймає збудження від значної кількості нейронів (в окремих випадках понад тисячі). Головний мозок людини складається з близько 10^{11} нейронів, які комунікують між собою через мільйони з'єднань. Кожен нейрон виконує функцію передачі збудження інших нейронів за допомогою нервових стиків, синапсів, в основі якої лежать складні електрохімічні процеси. Головним завданням синапсів є виконання ролі репитерів інформації, що призводить до посилення або послаблення сигналу,

який подорожує нервовою системою. В результаті до нейрона приходять одночасно сигнали, які мають збудливий та гальмівний вплив. Нейрон сумує збуджуючі та гальмуючі імпульси та у випадку, якщо результат перевищує пороговий показник, сигнал з виходу нейрона пересилається за допомогою аксона до інших нейронів.

Для опису моделі нейрона прийняті наступні позначення: n – кількість каналів отримання інформації нейроном; $x_1,...x_n$ – вхідні сигнали, $x=[x_1,...x_n]^T$; $\omega_0,...\omega_n$ – синапчині ваги, $\omega=[\omega_1,...\omega_n]^T$; y – вихідний сигнал нейрона; ω_0 – порогове значення; f – функція активації [19].

Формула, яка описує процес функціонування нейрона:

$$y=f(s), \tag{1.5}$$

де $s=\sum_{i=1}^{n} x_i w_i$.

Схема імітаційної моделі нейрона приведена на рис. 1.3.

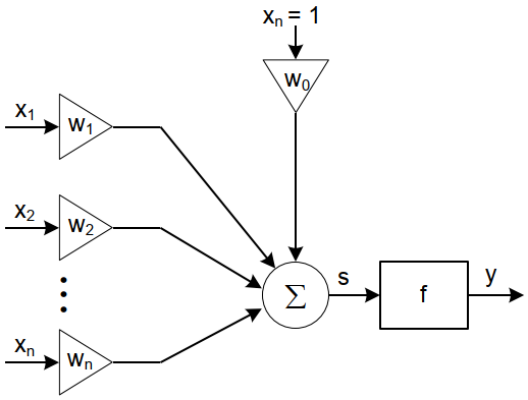

Рисунок 1.3 – Схема моделі нейрона

Функція активації нейрона f приймає різні форми залежно від пропонованої імітаційної моделі. Робота нейрона описується наступним чином. На першому етапі нейрон отримує вхідні сигнали x_0, $x_1,...x_n$, які множаться на відповідні їм ваги ω_0, $\omega_1,...\omega_n$. Отримані значення сумуються. Після чого

виникає сигнал s, який зображає функціонування лінійної частини нейрона. Далі цей сигнал поступає на етап активації, який найчастіше має нелінійний характер. Значення сигналу x_0 дорівнює 1, а вага ω_0 перевищує порогове значення. Найбільшим феноменом нейрону є легке навчання, що зводиться до підбору значень ваг [20].

Штучні нейронні мережі використовують конкретні модифікації приведеної моделі. Штучні нервові клітини поєднуються між собою в існуючих моделях за тими ж принципами, що й їх прототипи в людському мозку.

1.2.2 Модель персептрона

Функціонування персептрона описується виразом [21] (рис 1.4):

$$y = f\left(\sum_{i=1}^{n} x_i \omega_i + \theta\right). \tag{1.6}$$

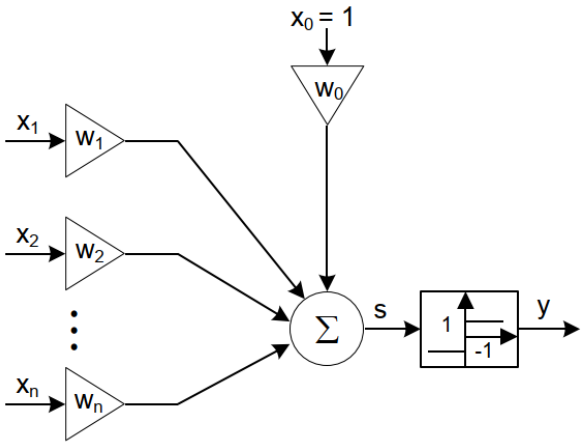

Рисунок 1.4 – Структура персептрона

Функція f є дискретною ступінчастою функцією – біполярною (можливі значення -1 або 1):

$$f(s) = \begin{cases} 1, & для \quad s > 0, \\ -1, & для \quad s \le 0. \end{cases} \tag{1.7}$$

На етапі активації персептрон може приймати тільки два діаметрально

протилежні вихідні значення, тому класифікація сигналу, що потрапляє на його вхід, здійснюється у вигляді векторів $x \sim [x_1 ..., x_n]^T$. Наприклад, нейрон персептрон, що має один вхідний канал, може розпізнавати сигнал або позитивним або негативним.

У навчанні мережі постійно відбувається процес модифікації ваги персептрона. Метод навчання персептрона отримав назву «навчання з учителем» або «навчання під наглядом» [22]. Завдання вчителя є подача на вхід персептрона сигналів $x(t) = [x_0(t),\ x_1(t),\ ...,\ x_n(t)]^T$, t = 1, 2, ..., для яких відомі істинні значення вихідних сигналів $d(t)$, t = 1,2, ..., званих еталонними сигналами [23].

Групи таких вхідних вибірок і відповідних їм значень еталонних сигналів називається навчальною послідовністю [23]. При використанні методів даної групи після введення вхідних значень розраховується вихідний сигнал нейрона. По завершенню відбувається модифікація ваг з метою мінімізації похибки між еталонним сигналом і вихідним сигналом персептрона. Актуальний алгоритм навчання персептрона виглядає наступним чином:

– задати початковим вагам персептрона довільні значення;

– на вхідні калани нейрона подати навчальний вектор $x = x(t) = [x_0(t),\ x_1(t),\ ...,\ x_n(t)]^T$, t = 1,2, ...;

– підрахувати вихідне значення персептрона за формулою (1.6);

– отримане вихідне значення $y(t)$ порівняти з еталонним значенням $d(x(t))$, що знаходиться в навчальній послідовності;

– модифікувати ваги наступним чином:

а) якщо y (x (t)) ≠ d (x (t), то wi (t +1) = wi (t) + d (x (t)) xi (t);

б) якщо y (x (t)) = d (x (t)), то wi (t +1) = wi (t), тобто значення ваг не змінюються;

– перейти до кроку 2.

1.2.3 Модель нейрона з сигмоїдою на виході

Структура нейрона з сигмоїдою на виході ідентична структурі персептрона. Зміна назви обумовлена функцією активації, яка має форму сигмоідальної з уніполярною або біполярною функцією. Це безперервні функції, що описуються виразами [24]:

уніполярна функція

$$f(x) = \frac{1}{1 + e^{-\beta x}} \tag{1.8}$$

біполярна функція

$$f(s) = \begin{cases} 1, & для \quad s > 0, \\ -1, & для \quad s \le 0. \end{cases} \tag{1.9}$$

На рис. 1.5 приведені графіки уніполярних функцій при різних значеннях параметра бета. Простий аналіз отриманих результатів дозволяє зробити висновок, що при малих значеннях параметра бета функція має пологий характер, але з ростом значення цього параметра графік стає більш вертикальним аж до здобуття порогового виду.

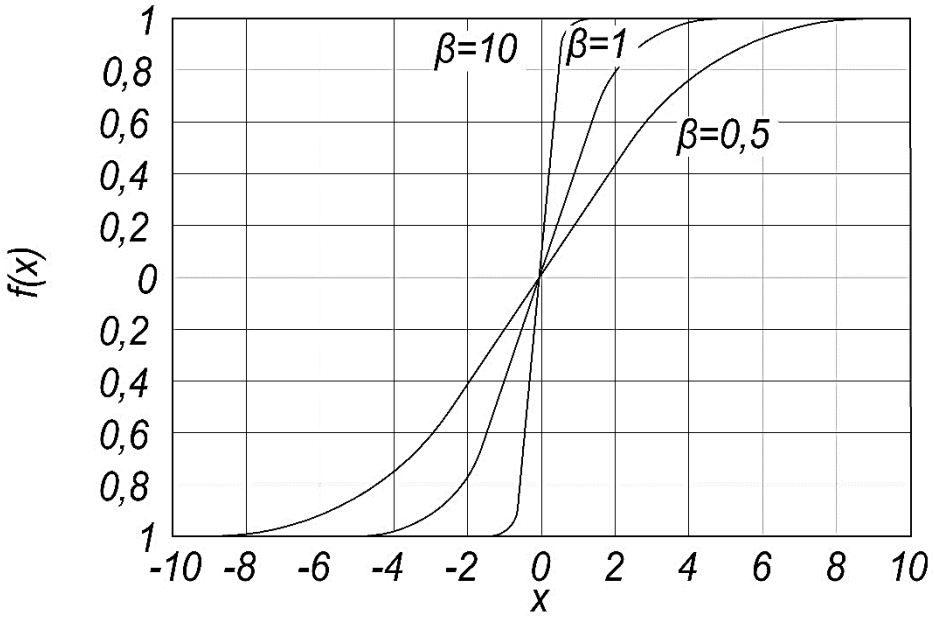

Рисунок 1.5 – Графік уніполярної функції активації при різних значеннях параметра β

Головною перевагою сигмоїдальних нейронів вважається диференційована функція активації [24]. Також похідні цих функцій легко знаходяться, оскільки вони набувають досить простих форм:

– для уніполярної функції:

$$\frac{df(x)}{dx} = \beta f(1)(1-(x)) \tag{1.10}$$

– для біполярної функції:

$$\frac{df(x)}{dx} = \beta(1-f^{2}(x)) \tag{1.11}$$

Структура нейрона з сигмоїдою на виході представлена на рисунку 1.6.

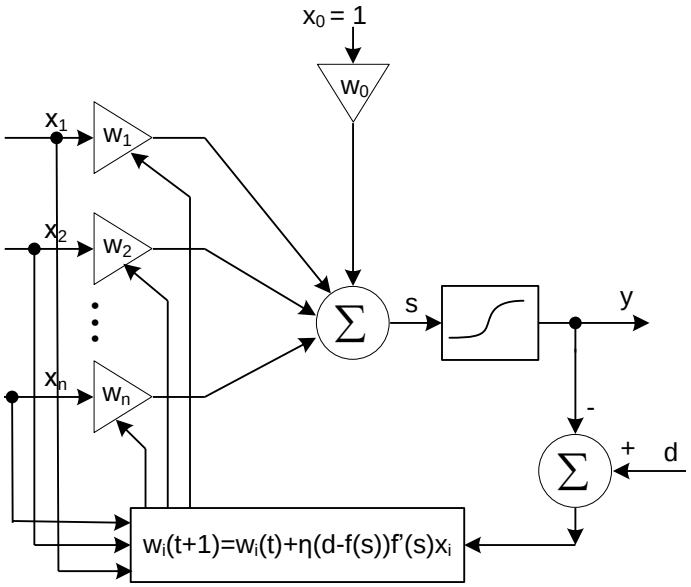

Рисунок 1.6 – Структура нейрона з сигмоїдою на виході

Вихідний сигнал описується виразом:

$$y(t) = f\left(\sum_{i=0}^{n} w_i(t)x_i(t)\right). \tag{1.12}$$

Величина похибки визначається:

$$Q(w) = \frac{1}{2}\left[d - f\left(\sum_{i=0}^{n} w_i x_i\right)\right]^2 \tag{1.13}$$

1.2.4 Модель нейрона Хебба

На рис. 1.7 зображена схема моделі нейрона Хебба. Вона ідентична структурам моделей типу нейрона з сигмоїд на виході, однак відрізняється специфічним методом навчання, так званим правилом Хебба. Розроблено два варіанта правила Хебба: «з учителем» і «без учителя». Хебб в процесі досліджень звернув увагу, що зв'язки між двома нейронами посилюються, якщо вони активізуються синхронно [25].

Хебб розробив алгоритм, в якому ваги модифікуються за наступною формулою:

$$w_i(t+1) = w_i(t) + \Delta w_i, \tag{1.14}$$

Звідси випливає:

$$\Delta w_i = \eta y x_i. \tag{1.15}$$

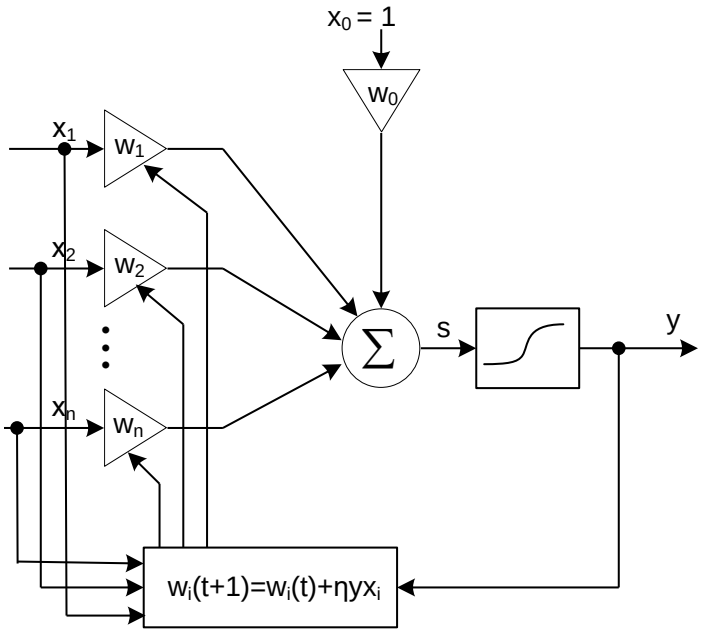

Рисунок 1.7 – Структура нейрона Хебба

Під час навчання одиночного нейрона відбувається модифікація значення ваги пропорційно значення сигналу, поданого на *i*-й вхід клітини, так і значення

вихідного сигналу, що враховує коефіцієнт навчання.

При такому навчанні нейрона не задіяні еталонні вихідні значення, даний метод називається метод «навчання без учителя». Невелика модифікація залежності (1.15) приводить до другого методу навчання нейрона Хебба – «навчання з учителем»:

$$\Delta w_i = \eta x_i d, \tag{1.16}$$

де d – еталонний сигнал.

Необхідно зазначити, що алгоритм Хебба має недолік – значення ваг не мають обмежень на збільшення. Для усунення цього недоліку в літературі пропонуються різні модифікації правила [26].

1.2.5 Структура і функціонування мережі

Багатошаровими нейронними мережами називаються структури що складаються не менше як з двох прошарків: вхідний і вихідний. До складу мереж можуть буди включені проміжні, приховані шари.

У випадку, коли мережа складається лише з двох шарів, то вхідний шар прирівнюється до прихованого шару. В багатошарових нейронних мережах обмін сигналами здійснюється тільки нейронами, які не розташовані в одному шарі.

У межах одного шару взаємодії нейронів не відбувається. Інформація передається від вхідного шару до вихідного, функція зворотного зв'язку відсутня. Структура односпрямованої тришарової мережі приведена на рис. 1.8 [27].

Другою назвою методу «навчання з учителем» є навчання під наглядом. Навчання відбувається в наступний спосіб: спочатку вхідні значення з навчальної послідовності подаються на вхід мережі, після чого послідовно розраховуються вихідні значення кожного нейрона від вхідного до вихідного шару.

Далі визначається реакція мережі на сигнал, отриманий на її вхід.

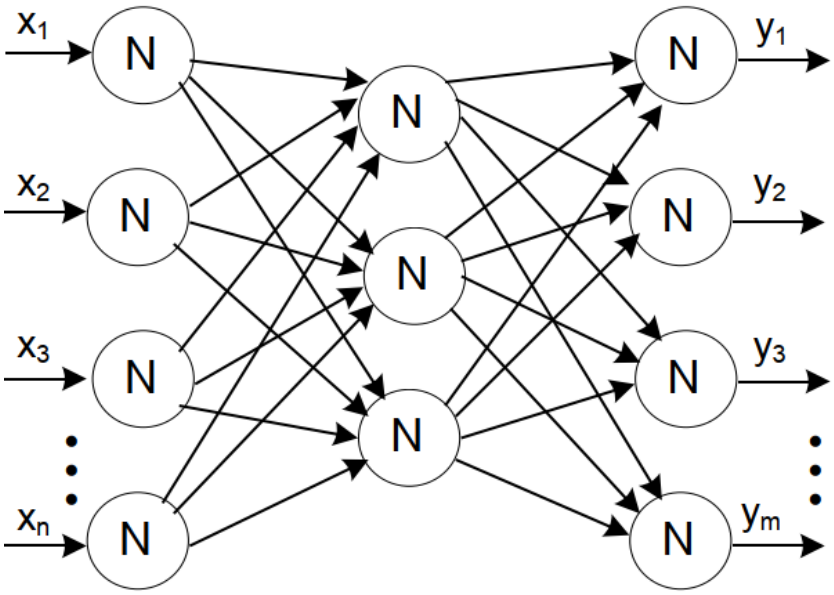

Рисунок 1.8 – Структура тришарової нейронної мережі

Необхідно так модифікувати ваги мережі, щоб вихідне значення максимально досягло базового значення. Це і є головна ідея алгоритмів розглянутого класу, оскільки «вчитель» зазначає, якою має бути реакція мережі.

1.2.6 Алгоритм зворотного поширення помилки

У більшості випадків навчання відбувається наступним чином: підраховується сума створених вхідних сигналів та відповідні їм ваги. Наступним етапом є передача на вхід обраної функції активації, по виконанню якої з'являвся вихідний сигнал нейрона. Для визначення похибки сигналу необхідно підрахувати різницю між фактичним вихідним значенням і еталонним значенням. Так само розраховуються похибки всіх інших шарів в багатошарових мережах. Складнощі виникають лише з визначенням похибок в прихованих шарах, так як вчителю не відомі еталонні значення на виходах нейронів, що знаходяться в цих елементах мережі.

Цю проблему вирішують за допомогою технології зворотного поширення

помилки [28-30], яка ефективно вирішує завдання навчання багатошарових нейронних мереж. Для складання цього алгоритму необхідно формально зазначити відповідну величину похибки. Вона являє собою функцію $Q(w)$ щодо вектору (вектор всіх ваг мережі), в якій в ролі змінних виступають всі ваги багатошарової нейронної мережі. Розкладання функції $Q(w)$ в ряд Тейлора в безпосередній близькості від відомого фактичного рішення в напрямку p має такий вигляд [28]:

$$Q(\mathrm{w}+\mathrm{p})=Q(\mathrm{w})+[\mathrm{g}(\mathrm{w})]^{\mathrm{T}}\mathrm{p}+0{,}5\mathrm{p}^{\mathrm{T}}\mathrm{H}(\mathrm{w})\mathrm{p}+\ldots, \tag{1.17}$$

де g(w) позначає вектор градієнта:

$$g(w)=\left[\frac{\partial Q}{\partial w_i},\frac{\partial Q}{\partial w_2},\frac{\partial Q}{\partial w_3},\ldots,\frac{\partial Q}{\partial w_n}\right]^T, \tag{1.18}$$

H(w) – гессіан, матриця других похідних:

$$H(w)=\begin{bmatrix}\frac{\partial^2 Q}{\partial w_1\partial w_1}\cdots\frac{\partial^2 Q}{\partial w_1\partial w_n}\\ \vdots\\ \frac{\partial^2 Q}{\partial w_n\partial w_1}\cdots\frac{\partial^2 Q}{\partial w_n\partial w_n}\end{bmatrix}. \tag{1.19}$$

Ваги модифікуються за формулою:

$$\mathrm{w}(t+1)=\mathrm{w}(t)+\eta(t)\mathrm{p}(t), \tag{1.20}$$

де η – коефіцієнт навчання.

Ваги можуть модифікуватися тривалий час, до моменту, поки функція не досягне мінімуму або її значення не стане меншим за значення порогового показника. Завдання сконцентроване на пошуку вектору, це забезпечує зменшення похибки на виході мережі та на чергових кроках алгоритму. Це означає, що на наступних етапах має виконуватися нерівність. Обмеження ряду Тейлора, апроксимуючого функцію похибки лінійним розкладом, має вигляд [31]:

$$Q(\mathrm{w}+\mathrm{p})=Q(\mathrm{w})+[\mathrm{g}(\mathrm{w})]^{\mathrm{T}}\mathrm{p}. \tag{1.21}$$

Залежність функції Q(w) від вагів, які розташовані на кроці t, а Q(w+p) –

від вагів, розташованих на кроці $(t+1)$, тоді для вирішення нерівності $Q(w(t+1))<Q(w(t)$ досить вибрати вектор $p(t)$, при якому $g(w(t)^T p(t)<0$, ця умова виконується при:

$$p(t)=-g(w(t)). \qquad (1.22)$$

При включенні формули (1.22) в залежність (1.20) буде отримано наступний вираз для визначення зміни вагів в багатошарових мережах:

$$w(t+1) = w(t) - \eta g(w(t)). \qquad (1.23)$$

Залежність (1.23) носить назву *«правило найшвидшого спуску».* Для використання виразу (1.23) з максимальною ефективністю з метою отримання алгоритму зворотного розповсюдження помилки необхідно ввести відповідне позначення. Структура мережі приведена на рис. 1.9. У всіх шарах знаходиться N_k елементів, k =1, ..., L, позначених як N^k_i, i = 1, ..., N_k. Елементи N^k_i – це окремі нейрони, кожен з них може мати сигмоїду на виході. Зображена нейронна мережа має N_0 входів, на які надходять сигнали $x_1(t)$, ...,$x_{N0}(t)$, що задаються в векторній формі:

$$x = [x_1(t),...,x_{N0}(t)]^T, \ t = 1,2, \qquad (1.24)$$

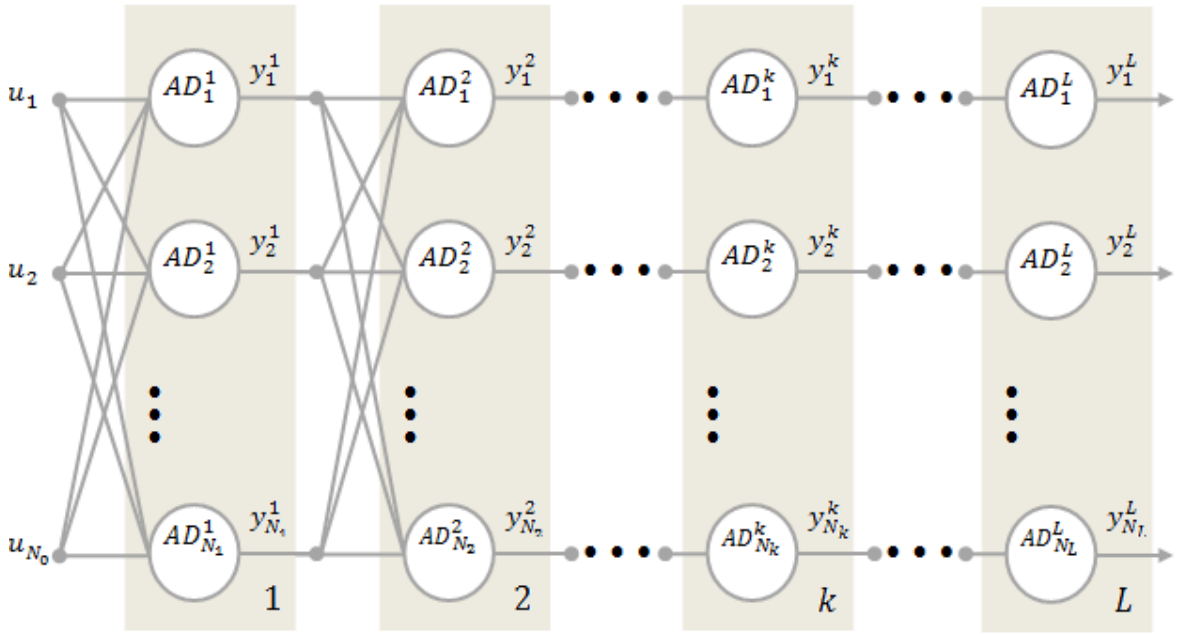

Рисунок 1.9 – Багатошарова нейронна мережа

Вихідний сигнал *i*-го нейрона в *k*-м шарі позначається $y_i^{(k)}(t)$, $i = 1,..,N_k$, $k = 1,..., L$. На рис. 1.10 зображена розгорнута структура *i*-го нейрона в *k*-ому шарі. Нейрон N^k_i має N_k входів, що утворюють вектор [32]:

$$x^{(k)}(t) = [x_{01(k)}(t),..., x^{(k)}_{N_{k-1}}(t)]^T, \tag{1.25}$$

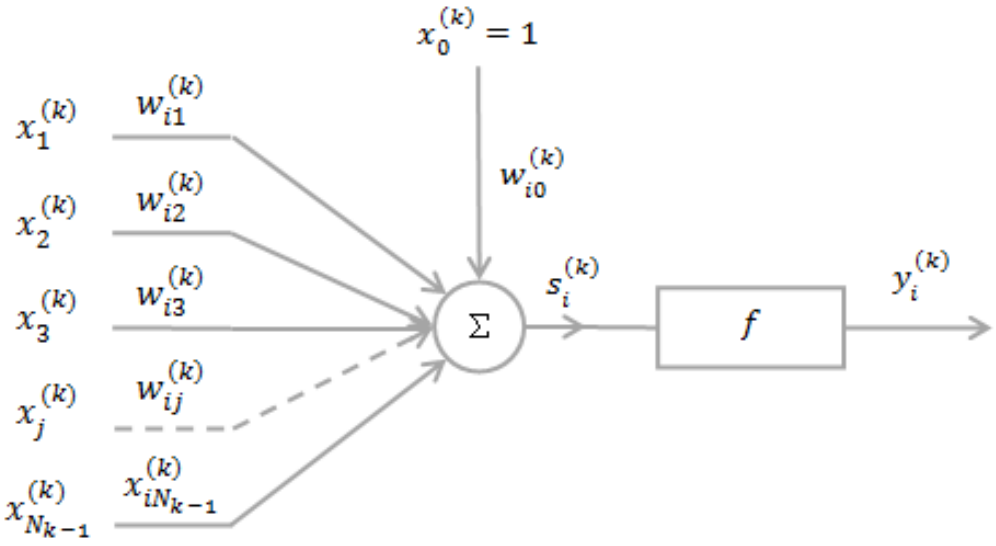

Рисунок 1.10 – Структура нейрона

Вхідний сигнал нейрона N_1 пов'язаний з вихідним сигналом (к-1)-го шару наступним чином:

$$x_i^{(k)}(t) = \begin{cases} x_i(t) & \text{для } k = 1, \\ y_i^{(k-1)}(t) & \text{для } k = 2,..., L, \\ +1 & \text{для } i = 0,\ k = 1,..., L. \end{cases} \tag{1.26}$$

На рис. 1.10 за допомогою символа (*i*) позначена вага входу до і-го нейрона, I = 1, ... $I_к$, розташованого в k-му шарі, який з'єднує цей нейрон з у-м вхідним сигналом. Вектор ваг нейрона буде визначатися:

$$w_i^{(k)}(t) = [w_{i,0}^{(k)}(t),..., w_{iN_{k-1}}^{(k)}(t)]^T, \; k=1,...,L, \quad i = 1..,N_k. \tag{1.27}$$

Вихідний сигнал нейрона в момент $N^k_i = 1,2, ...$ визначається як:

$$y_i^{(k)}(t) = f(si^{(k)}(t)) \tag{1.28}$$

причому

$$s_i^{(k)}(t) = \sum_{j=0}^{N_{k-1}} w_{ij}^{(k)}(t) x_j^{(k)}(t). \tag{1.29}$$

Вихідні сигнали нейронів L-го шару:

$$y_1^L(t), y_2^{(L)}(t), ..., y_{N_L}^L(t) \tag{1.30}$$

одночасно є вихідними сигналами мережі в цілому. Вони порівнюються з так званими еталонними сигналами мережі

$$d_1^L(t), d_2^{(L)}(t), ..., d_{N_L}^L(t). \tag{1.31}$$

У результаті алгоритм зворотного поширення помилки можна записати у вигляді [33]:

$$y_i^{(k)}(t) = f'(s_i^{(k)}(t)), \; s_i^{(k)}(t) = \sum_{j=0}^{N_{k+1}} w_{ij}^{(k)}(t) x_j^{(k)}(t); \tag{1.32}$$

$$Q_i^{(k)}(t) = \begin{cases} d_i^{(L)}(t) - y_i^{(L)}(t) & \text{для} \quad k = L, \\ \sum_{m=1}^{N_{k+1}} \delta_m^{(k+1)}(t) w_{mi}^{(k+1)}(t) & \text{для} \quad k = 1, ..., L-1; \end{cases} \tag{1.33}$$

$$\tag{1.34}$$

$$\delta_i^{(k)}(t) = \varepsilon_i^{(k)}(t) f'(s_i^{(k)}(t)); \tag{1.35}$$

$$w_{ij}^{(k)}(t+1) = w_{ij}^{(k)}(t) + 2\eta \delta_i^{(k)}(t) x_j^{(k)}(t). \tag{1.36}$$

При навчанні нейрону методом зворотного поширення помилки мережа модифікує його вагу кожного разу при подачі на її вхід навчального вектору. Такий підхід називається поступовою актуалізацією ваг, але розроблений інший спосіб: на вхід мережі послідовно подаються навчальні вектори, проводяться розрахунки відповідних сигналів на виході мережі, по завершенню відбувається порівняння з еталонними значеннями та підсумовуються похибки кожної ітерації. По закінченню кожної епохи відбувається пред'явлення навчальних вибірок з послідуючим коригуванням значень всіх ваг з урахуванням накопиченого значення похибки. Цей спосіб носить назву кумулятивна актуалізація ваг.

Важливим аспектом є процес ініціалізація ваг мережі. Максимальна близькість стартових значення ваг до оптимальних призводить до швидкого

завершення навчання. Навчання мережі – це шлях пошуку мінімуму функції похибки. Це функція кількох змінних, в ролі яких виступають показники ваг мережі. На рисунку 1.11 показано гіпотетичний графік функції похибки для однієї змінної (однієї ваги).

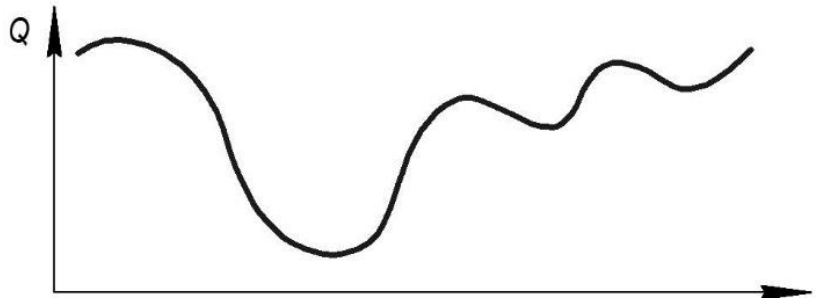

Рисунок 1.11 – Теоретична функція похибки однієї змінної

Недоліком методу найшвидшого спуску є наявність кількох локальних мінімумів, при цьому алгоритм пошуку рішення сходиться до одного з них. Тому більш популярний модифікований алгоритм зворотного поширення помилки [34].

Найпримітивнішим способом, що вирішує недолік локальних мінімумів, є використання різних стартових значень ваг для навчання нейронної мережі [35]. Величини ваг обираються довільним чином з встановленого інтервалу з використанням рівномірного розподілу. Потім нейронна мережа застосовує алгоритм зворотного поширення помилки для навчання.

У випадку, коли похибка навчання закінчує процес зменшення або починає ріст, повторно випадковим чином обираються ваги, але з нового інтервалу значень. Процес навчання мережі відбувається паралельно зі спостереженням за кінцевим значенням похибки з метою знаходження локального мінімуму. Запропонований метод збільшує період навчання мережі.

Після встановлення великого початкового значення показника ваг середньоквадратична похибка на виході мережі є практично незмінною. Це

обґрунтовано занадто високим вихідним сигналом нейрона, а саме лінійної його частини, як результат насичення функції активації. Корегування значень ваг в алгоритмі зворотного поширення помилки здійснюється пропорційно похідній функції. Графік похідної приведено на рисунку 1.12.

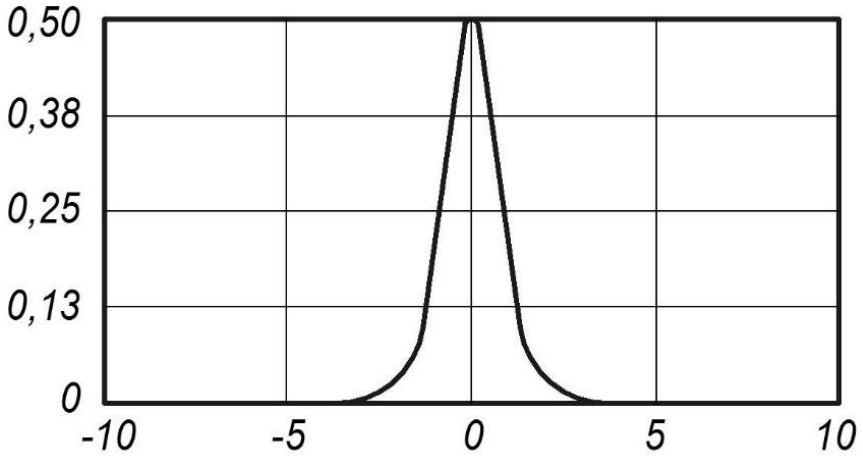

Рисунок 1.12 – Графік похідної функції активації

Тому існує обернено пропорційна залежність між абсолютним значенням сигналу на виході з лінійної частини нейрона та корекцією ваг, як результат швидкість навчання мережі буде незначною. Головним завданням підбору значення ваг є пошук таких довільних значень, які в результаті давали б максимально наближені до одиниці показники сигналу на виході лінійної частини. Коефіцієнт навчання має суттєвий вплив на збіжність алгоритму зворотного поширення помилки. На даний момент не існує універсального методу добору його значення. Найчастіше крок корекції обирається з інтервалу (0,1). У випадку, коли на графіку цільової функції полога, тоді значення градієнта невеликі, та при великих значеннях коефіцієнта швидкість навчання за допомогою алгоритму буде вищою. Коли ж цільова функція виявляється прямовисною, то більші величини коефіцієнта навчання призведуть до осциляції близького рішення та значно подовжують процес навчання мережі.

Головними факторами, що впливають на підбір коефіцієнта, є вирішуване завдання та досвід людини, яка навчає мережу [36].

1.3 Нейро-нечіткі мережі

Нечіткі нейронні мережі або гібридні мережі створенні з метою комбінації переваг нейронних мереж і систем нечіткого виводу [37]. По-перше, вони дають змогу розробляти та подавати моделі систем у вигляді правил нечітких продукцій, котрі характеризуються наочністю та простотою змістовної інтерпретації. По-друге, при побудові правил нечітких продукцій застосовуються методи нейронних мереж, що є більш зручним і менш трудомістким процесом для системних аналітиків. Сучасне використання нейро-нечітких мереж визнається фахівцями як один з найперспективніших методів для розв'язку слабо або погано структурованих завдань прикладного системного аналізу.

Нейро-нечітка мережа по факту є багатошаровою нейронною мережею спеціальної структури з відсутніми зворотними зв'язками, в якій застосовуються стандартні (не нечіткі) сигнали, функції активації та ваги, а виконання операції підсумовування ґрунтується на використанні фіксованої Т-норми, Т-конорми чи деякої іншої безперервної операції. Для гібридної нейронної мережі величини входів, виходів і ваг є матеріальні числа з відрізка [0, 1].

Головна ідея, на якій базується модель нейро-нечітких мереж – це використання існуючої вибірки даних для ухвалення параметрів функцій належності, які краще відповідають деякій системі нечіткого виведення. При цьому для визначення параметрів функцій застосовуються процедури навчання нейронних мереж.

У системі MATLAB (Fuzzy Logic Toolbox) нейро-нечіткі мережі реалізовані у формі адаптивної системи нейро-нечіткого виводу ANFIS.

Гібридна мережа ANFIS являє собою нейронну мережу, що має один вихід з декількома входами. При цьому терми вхідних лінгвістичних змінних зображуються стандартними функціями належності системи MATLAB. Але нейро-нечітка мережа ANFIS є системою нечіткого виводу FIS віднесеного до типу Сугено нульового чи першого порядку, в котрій кожне з правил має постійну вагу, що дорівнює 1. В середовищі MATLAB користувач має змогу коригувати та налаштовувати гібридні мережі ANFIS за аналогією з системами нечіткого виведення за допомогою інструментів усіх розглянутих раніше засобів пакету Fuzzy Logic Toolbox [38].

1.4 Інтелектуальні системи управління

Однак, системи керування суттєво відмінні від систем прогнозування, спостереження, діагностики, конструювання та планування. Відмінною особливістю інтелектуальних систем керування є їх класифікаційне віднесення до класу динамічних систем, які працюють в режимі реального часу і мають в своєму складі підсистеми взаємодії з зовнішнім світом (датчики, виконавчі пристрої).

1.4.1 Концепція і визначення поняття інтелектуальної системи

Розвиток технологій та створення високо продуктивних мікропроцесорів з великим об'ємом пам'яті, можливість організації мульти мереж для реалізації паралельних обчислень, з одного боку, та потреба обробки великих масивів інформації, використання баз знань для генерації направленої діяльності – з іншого, призвели до створення інтелектуальних систем.

Під інтелектуальною системою розуміють об'єднану інформаційним процесом структуру технічних засобів і програмного забезпечення, котрі працюють у взаємозв'язку з оператором або незалежно від нього; здатну на основі відомостей і знань при наявності мотивації синтезувати, генерувати

рішення про дію та знаходити ефективні шляхи вирішення завдань.

Узагальнена структура інтелектуальної системи приведена на рис. 1.13.

Базуючись на інформації про навколишнє середовище та аналізуючи свій стан, система за наявності пам'яті та мотивації синтезує мету, яка разом з іншими даними обробляється динамічною експертною системою. Остання з використанням бази знань генерує експертну оцінку, на основі якої приймається рішення про реакцію та прогнозуються її результати.

Відповідно до прийнятого рішення виробляється механізм управління, тобто синтезується той чи інший алгоритм або закон управління, який реалізується за допомогою різних виконавчих органів і впливає безпосередньо на об'єкт управління. Отримані результати керуючого впливу порівнюються з запланованими (засоби зворотного зв'язку) [39].

У разі невідповідності отриманих результатів на базі нової експертної оцінки виконуються дії системи, що усувають цю невідповідність. Якщо відповідність неможливо отримати, то корегується мета. Дана структура інваріантна до об'єкта керування та носить універсальний характер.

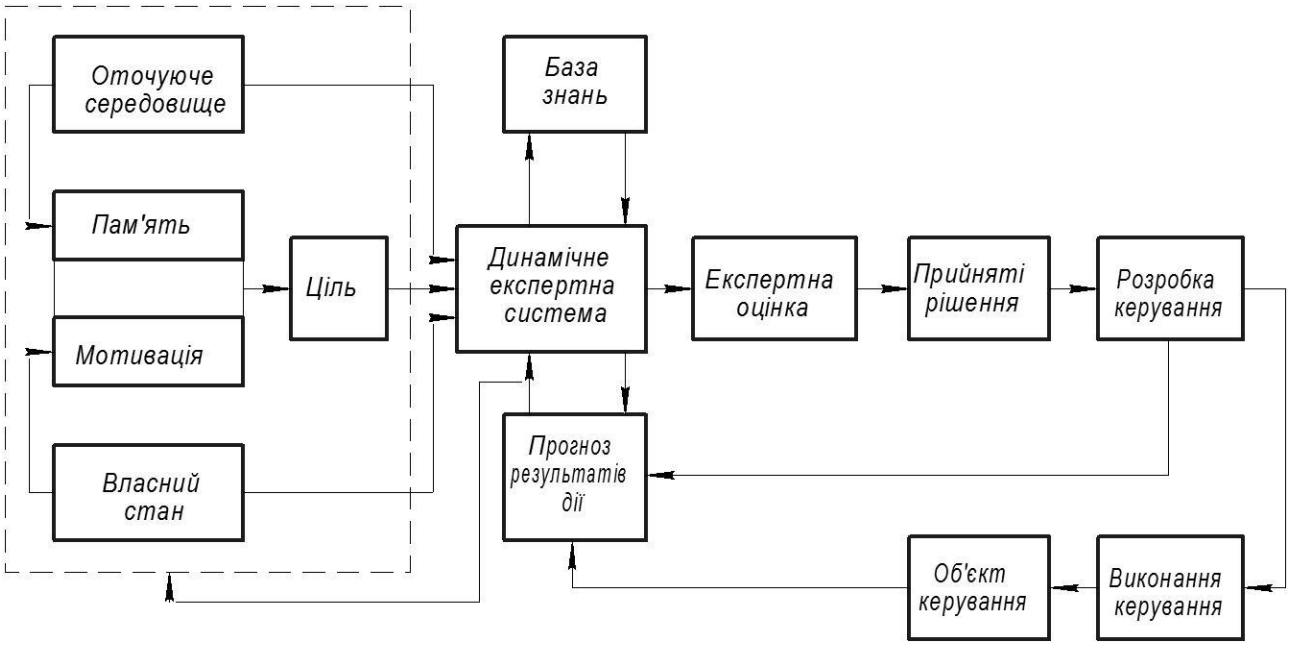

Рисунок 1.13 – Структура інтелектуальної системи

1.4.2 Структура інтелектуальної системи управління

Інтелектуальні системи здатні генерувати мету, приймати рішення до дії, забезпечувати дію для досягнення мети, прогнозувати значення параметрів результату дії і зіставляти їх з реальними, утворюючи зворотний зв'язок, коригувати мету або управління [40]. На рисунку 1.14 зображена структурна схема ІС, де представлені два блоки системи: генерація мети та її реалізація.

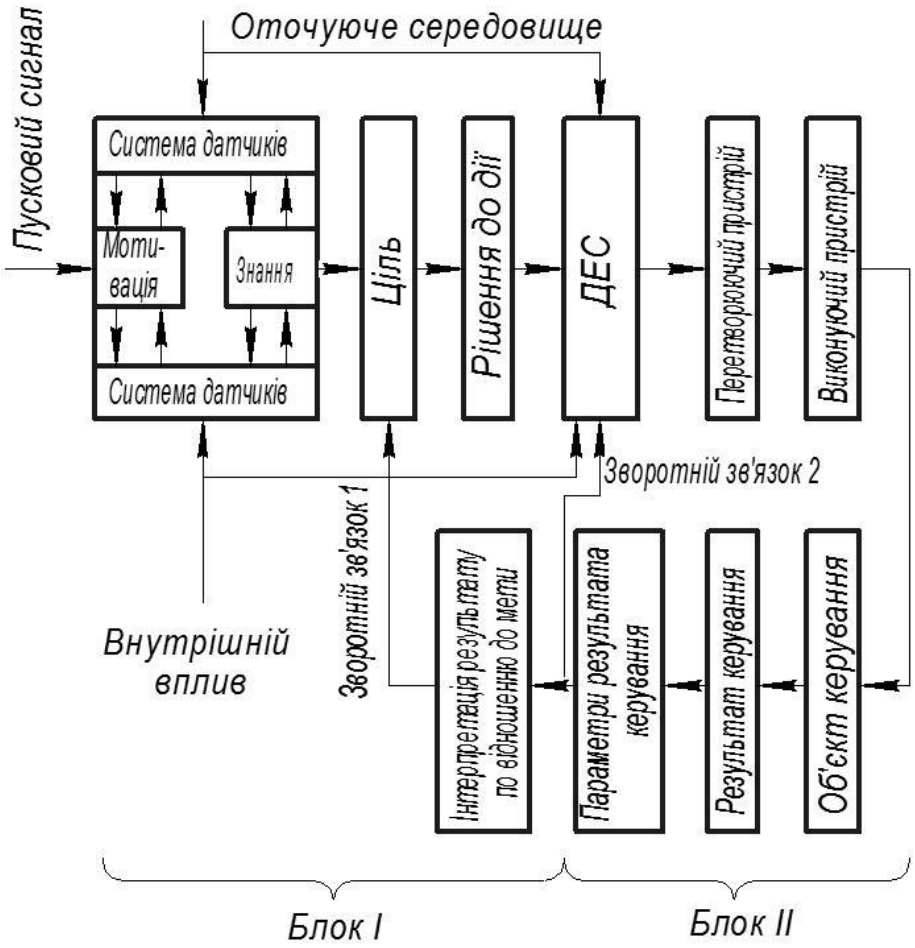

Рисунок 1.14 – Структурна схема ІС

У блоці генерації мети на основі оцінки інформації, яка надходить від системи датчиків, при наявності достатньої мотивації та знань кристалізується мета та перелік дії для її досягнення. Оцінка інформації проводиться під впливом пускових сигналів. Зміни в навколишньому середовищі та власному стані системи можуть призводити до потреби (мотивації), а при наявності знань

може бути згенеровано мету. Під метою розуміється [41] ідеалізований прогноз результату діяльності системи. Продовжується оцінка інформації про оточуюче середовище та стан системи, в тому числі об'єкта керування, при зіставленні варіантів шляху досягнення мети приймається рішення до дії.

У блоці реалізації мети динамічної експертної системи (ДЕС) на основі відомостей, отриманих у режимі реального часу [42] про оточуюче середовище та свій стан, за умови наявності мети «незнання» здійснюється експертна оцінка, приймається рішення про керування, прогнозує результати дії. Представлений у вигляді коду алгоритм керування перетворюється в фізичний сигнал і поступає на виконавчі пристрої. Об'єкт керування, отримуючи сигнал від виконавчих елементів, виконує відповідну дію, результати якої оформлюються в вигляді параметрів, по ланцюгу зворотного зв'язку 2 поступають до ДЕС, де порівнюються з прогнозованими. В той самий час параметри результату дії, розділені відповідно до властивостей цілі, оброблюються в блоці І та можуть використовуватися для якісної оцінки отриманого результату: наприклад, задача виконана, але результат не задовольняє додаткові (необов'язкові) умови. Якщо мета досягається за всіма параметрами, то керування підкріплюється, в іншому випадку відбувається корекція. У випадку, коли мета недосяжна, змінюються параметри мети. Слід враховувати, що при непередбачуваних змінах стану оточуючого середовища або об'єкта керування, або системи в цілому, може бути згенерована нова мета та організація її досягнення. Структура ІС поряд з новітніми містить класичні елементи та зв'язку, центральне місце в ній займає ДЕС.

1.4.3 Динамічні експертні системи і база знань

ІС можливо описати за рахунок наступних виразів:

$$T \times X \times S \xrightarrow{\alpha 1} M \times T;$$

$$T \times M \times S \xrightarrow{\alpha 2} З \times T;$$

$$C \times T \times S \xrightarrow{\alpha 3} R \times T;$$

$$T \times X = \{A \times T\} X \times T + \{B \times T\} U \times T;$$

$$T \times Y = \{D \times T\} X \times T;$$

$$T \times R \times Y \xrightarrow{\alpha 4} З \times T,$$

де T – множина моментів часу; X, S, M, C, R і Y – множини станів системи, навколишнього середовища, мотивації, мети, прогнозованого та реального результату; A, U і D – матриці параметрів; α_i – інтелектуальні оператори перетворення, що використовують знання.

У цьому описі сполучаються уявлення об'єктів системи у вигляді множини значень або множини висловлювань, або інших форм. Опис динамічних характеристик ІС може бути здійснений в просторі станів. Інтелектуальні оператори, які реалізують сприйняття, уявлення, розвиток поняття, думки та висновки в процесі пізнання, є формальним засобом обробки даних і знань, а також – ухвалення рішень. Усі вище зазначені аспекти мають бути покладені в основу побудови ДЕС, які функціонують в режимі реального часу та реальному світі.

ДЕС є певне комплексне утворення, здатне оцінювати стан системи та середовища, порівнювати параметри бажаного та отриманого результатів дії, ухвалювати рішення та виробляти керування, що сприяє досягненню цілі. Знання, які отримує експертна система, можна поділити на три категорії [43]:

1) концептуальне знання – це знання, втілене в словах людської мови або, точніше, в науково-технічних термінах, класах і властивостях об'єктів оточуючого середовища. До цієї категорії відносяться зв'язки, відносини та залежності між поняттями та їх властивостями, також виражені словами та термінами. Концептуальне знання – це галузь фундаментальних наук, якщо враховувати, що поняття є вищим продуктом матерії – мозку;

2) фактуальне знання – це сукупність даних про якісні та кількісні характеристики визначених об'єктів. З цією категорією знань зв'язуються визначення «інформація» та «дані». Будь-яке знання несе інформацію та може бути надано в вигляді даних; фактуальне знання – це те, з чим завжди мали

справу ЕОМ і з чим вони мають справу й досі. Базою даних називають сучасну форму накопичення даних. Для створення баз даних, для пошуку в них необхідної інформації потрібно спиратися на концептуальне знання;

3) алгоритмічне знання – це те, що заведено називати словами «вміння», «технологія» та ін. В обчислювальній справі алгоритмічне знання реалізується у вигляді алгоритмів, програм і підпрограм. Дана реалізація алгоритмічного знання зветься програмним продуктом. Найбільш розповсюджені форми програмного продукту – програмні системи, прикладне програмне забезпечення й інші, зорієнтовані на конкретну галузь застосування, ДЕС. Концептуальні знанні базується на організації та використання прикладного програмного забезпечення.

Концептуальне знання є вищою, визначальною категорією знання, з точки зору практики інші категорії можуть здаватися більш важливими. Виходячи з цього, вірогідно, концептуальне знання досить рідко втілюється у формі, доступній для опрацювання на ЕОМ [44]. А у випадку втілення, це відбувається неповно та однобічно. Людина, в більшості випадків, залишається носієм концептуального знання, як результат гальмування процесів автоматизації. Системи, що реалізує всі три категорії знання, але виносить концептуальне знання на передній план і працює з урахуванням його інтенсивного користування, називають базою знань. Розроблення та широке вживання баз знань в ІС – одне з найактуальніших завдань. Концептуальну частину бази знань називають моделлю предметної області, алгоритмічну частину – програмною системою, актуальну частину – базою даних.

Наступна функція ДЕС – це вирішення задач. Рішення завдання машиною можливо тільки в випадку наявності формальної специфікації. Остання мусить спиратися на певну базу знань. Модель предметної галузі описує загальну ситуацію, в якій з'явилася задача, а специфікація – зміст задачі. Разом вони дають змогу встановити, які теоретичні зв'язки та залежності, в яких сполученнях і в якій послідовності мають бути використані для розв'язку

задачі.

Прикладні програми становлять собою об'єктивні засоби, що стоять за цими залежностями, та містять алгоритми [45] для розв'язання виникаючих при цьому рівнянь. Врешті, база даних постачає всі початкові дані або їх частину для виконання визначених алгоритмів, дані, що не підходять, мають міститися в специфікації. Цим трьом складовим частинам баз знань відповідають три етапи розв'язку задачі [46]: 1) побудова абстрактної програми розв'язання (включаючи появу завдання, його постановку та специфікацію); 2) переклад задачі на придатну для машини мову; 3) трансляція та виконання програми.

Розробка абстрактної програми пов'язана з наданням та опрацюванням концептуального знання в ІС і за визначенням є набутком штучного інтелекту. Штучний інтелект зв'язують з опрацюванням текстів, усних дописів на природній мові, з аналізом та опрацюванням інформації.

1.4.4 Структура і функції динамічної експертної системи

До функцій ДЕС також належать оцінка результатів розв'язку задачі, формування властивостей майбутнього результату дії, ухвалення рішення про керування, вироблення алгоритму управління та звірення параметрів бажаного та фактичного результатів. Передбачається моделювання процесів для оцінювання можливих наслідків і правильності розв'язку задачі.

У реальних випадках є проблема опису досліджуваних предметів. Такий опис неправомірно рахувати частиною специфікації задачі, бо щодо одного об'єкта встановлюється багато задач, це необхідно враховувати під формування бази знань. Окрім того, може виявитися, що викладене завдання не вирішується до кінця автоматично, приміром, через неповноту специфікації або опису об'єкта. Отож в ІС доцільний на визначених стадіях інтерактивний режим праці з ДЕС. Необхідно пам'ятати, що модель предметної галузі описує загальну ситуацію (знання), а специфікація – зміст задачі. Головними проблемами є створення єдиного програмного середовища та синтез алгоритмів

безпосередньо по постановці задачі.

Уявлення алгоритму вирішення залежать від мети, яку досягає ІС, наявних база знань, завдань і прийнятих рішень. Відповідно до цього виділяють три типи ДЕС. Структура ДЕС першого типу зображена на рис. 1.15.

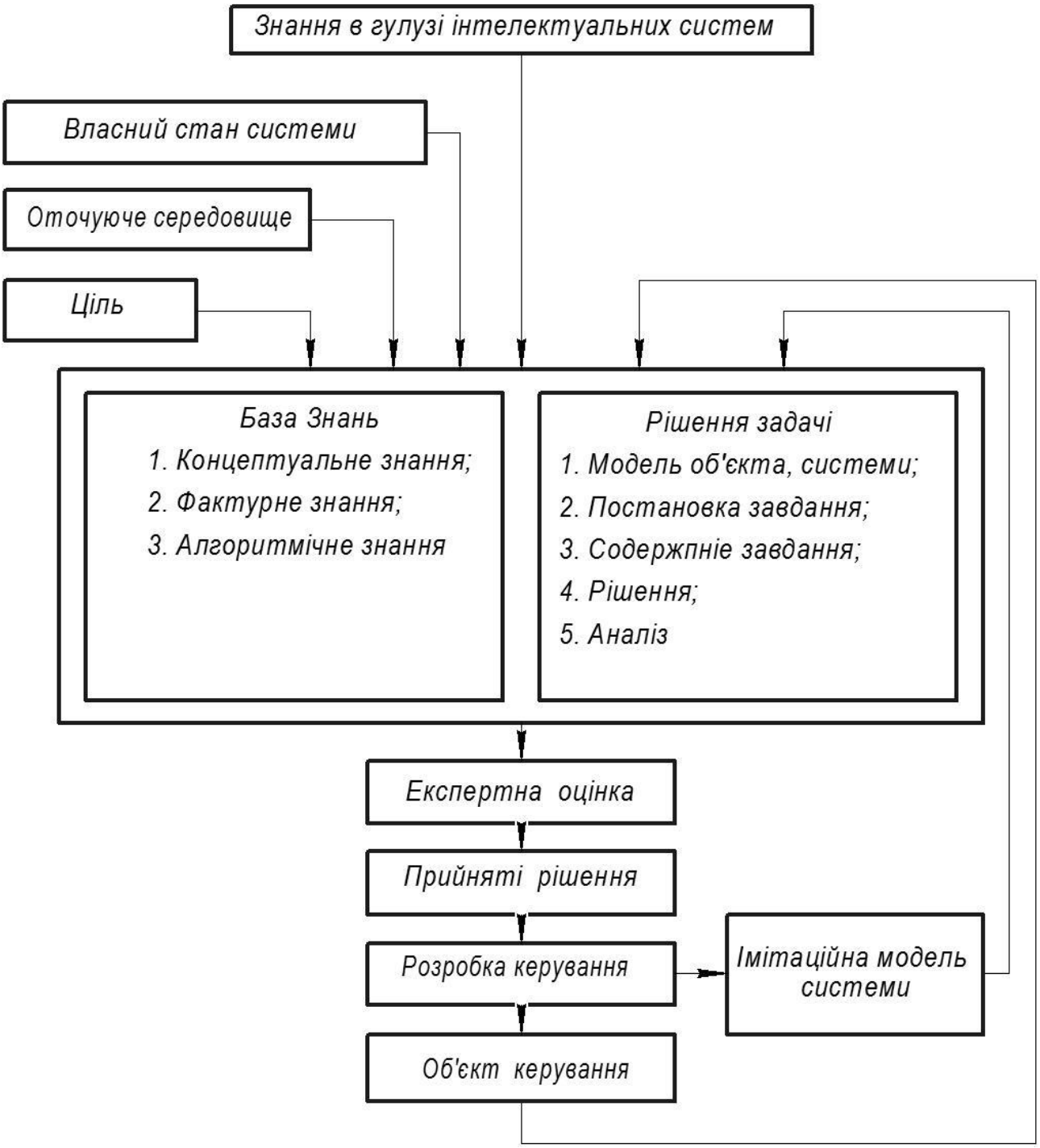

Рисунок 1.15 – Структура ДЕС першого типу

Мається на увазі, що концептуальне та фактуальне знання точно віддзеркалюють процеси та відомості, пов'язані з предметною галуззю.

Вирішення задачі, яка виникає в цій галузі, буде одержано на основі жорстких математичних методів згідно із постановкою та специфікацією. Підсумки дослідження рішення та прогноз застосовуються для одержання експертної оцінки й ухвалення рішення про потребу керування. Далі на основі підходящого алгоритму керування, існуючого в базі знань, формується управляючий вплив.

Ефективність й несуперечність цієї дії, перш ніж вона надійде на об'єкт керування, оцінюється на імітаційній математичній моделі. Оцінка мусить виконуватися швидше справжніх процесів в ІС. Але ДЕС, що реалізують ухвалені рішення, є складними програмними комплексами, що призначені для автоматичного ухвалення рішення або для допомоги особам, котрі ухвалюють рішення, та при оперативному керуванні складними системами й процесами, що працюють в умовах обмежень часу.

ДЕС другого типу, на відміну від ДЕС першого типу, призначені для вирішення важко формалізованих задач під час відсутності повної та достовірної інформації (рис. 1.16). У них застосовуються експертні моделі, збудовані з урахуванням знань експертів – спеціалістів у цій проблемної галузі, та евристичні методи пошуку розв'язку. Однією з головних проблем при проектуванні ДЕС другого типу є необхідність вибору формального апарату з метою опису процесів ухвалення рішення й побудова з його урахуванням моделі прийняття рішень, адекватної проблемній галузі. В ролі такого апарату, як правило, використовують продукційні системи. Однак, основні дослідження проводяться в контексті трактування виробничої системи алгоритмічним методом з притаманною їй послідовною схемою пошуку розв'язку.

Отримані в результаті моделі найчастіше є неадекватними дійсним проблемам галузі, характеризуються відсутністю диференціації процесу пошуку рішення (рис. 1.16). Вирішенням є паралелізм при пошуку [48].

Рисунок 1.16 – Структура ДЕС другого рівня

Варто орієнтуватися на поєднання ДЕС першого та другого типів у розрахунково-логічну ДЕС третього типу, де база знань сполучає як жорсткі математичні формули, так й інформацію експертів, відповідно – нежорсткі евристичні методи з жорсткими математичними методами пошуку рішення, причому вага будь-якого компонента визначається максимально реалістичним описом предметної галузі та способом пошуку рішення (рис. 1.17).

Рисунок 1.17 – Структура ДЕС третього рівня

Розробка ДЕС супроводжується наступними проблеми [49]:

- визначення складу та подальше формування бази знань;
- створення нових і використання вже відомих теорій і методів для математичного опису інформаційних процесів в ІС;
- створення способів подачі та організації використання знань;
- створення алгоритмів і програмного забезпечення для реалізації «гнучкої логіки»;
- пошук придатних обчислювальних середовищ для впровадження

паралельних алгоритмів при формуванні ДЕС.

1.4.5 Вимоги до ДЕС

У динамічних експертних системах робота відбувається в складі системи, укомплектованої зворотними зв'язками, тому існує необхідність забезпечення сталої роботи таких ІС. З класичних міркувань прийнято вважати, що тривалість реакції ДЕС на вхідний сигнал, що витрачається на обробку вхідної інформації і визначення керуючого впливу, є тільки величиною затримки. Частковий аналіз дозволяє оцінити зміну періодичних властивостей системи, це є методом визначення запасу стійкості. За потреби є функція корекції за рахунок використання фільтрів.

Однак, з точки зору класичної теорії управління ІС є системами, які мають складну структуру та зв'язки, аналіз стійкості яких традиційними методами здійснити дуже складно.

1.4.6 Нейромережеві технології інтелектуальних систем

Технології програмування, що застосовуються в системах керування, розділяються на традиційні, які базуються на звичайних обчислювальних процедурах, та «системи штучного інтелекту» або інтелектуальні системи [50]. Характерними прикладами останніх вважаються експертні системи та нейронні мережі. На вибір типу програмного забезпечення (ПЗ) впливає ряд факторів, головні з яких: вимоги галузі застосування та вартість розробки ПЗ.

Так, наприклад, система G-2, яка використовує весь спектр сучасних інформаційних технологій, дозволяє вирішувати досить широке коло завдань управління, але її застосування стає проблематичним при рішень задач прикладного характеру не стільки через високу вартість системи, скільки через високі вимоги до обчислювальних ресурсів.

Загальновідома система CIAM, що використовує класичні обчислювальні процедури, є обмеженою та може вирішувати тільки задачі моделювання в

досить вузькому спектрі, це стримує можливості її застосування в основному рамками навчального процесу.

Більш просунута система моделювання Matlab + Simulink має можливість працювати з широким класом об'єктів і підходів, у тому числі з використанням елементів штучного інтелекту (Neural Network Toolbox, Fuzzy Logic Toolbox). Складнощі, обумовлені поєднанням системи з реальними об'єктами та окремими програмними модулями, звужують сферу використання системи завданнями моделювання [51].

На практиці є ряд завдань управління, де інформація про об'єкт може бути нецілковитою, неконкретною або нечіткою, у цьому випадку застосування класичних обчислювальних алгоритмів стає малоефективним і не дає бажаного результату. Також зв'язок між вхідними та вихідними параметрами може бути складно модельованим, а часом просто неможливим. У таких випадках позитивний результат можна досягти за рахунок використання нейромережевих технологій.

1.5 Розробка методів і засобів створення систем управління

Як показав аналіз в області теорії і практики штучного інтелекту, в даний час створені досить ефективні технології штучного інтелекту, які застосовуються в різних практичних додатках, в тому числі й в управлінні [52]. Однак, більшість цих технологій використовуються для розробки, дослідження та впровадження локальних систем управління, призначених в основному для вирішення завдань стабілізації [53] деяких змінних технологічного процесу.

1.5.1 Концепція розвитку методів ІТ для створення гібридних і інтелектуальних систем управління

Проведені численні дослідження [54-56], показали, що ІТ можна використовувати при розробці безпосередньо моделі оптимального управління

процесом, а не моделі самого технологічного процесу. Тобто такі технології дозволяють розробляти відразу ж алгоритми управління, на відміну від традиційного ланцюжка: розробка структури моделі процесу – проведення експериментальних досліджень на об'єкті – ідентифікація моделі – формулювання оптимізаційної задачі – підбір методу оптимізації – розробка алгоритму оптимального управління. Традиційний підхід передбачає тривалий (часом кілька років), дорогий і не завжди успішний шлях створення системи оптимального управління.

Використання ІТ дозволяє вирішувати аналогічні завдання відразу ж, і як показав досвід досить успішно [57]. Справа в тому, що методи штучного інтелекту припускають використання знань, досвіду та інтуїції людей-експертів, добре знайомих з предметною областю. Тобто тут використовується так званий ефект «готових знань». На відміну від цього розробка математичної моделі (основного компонента системи) є процесом створення «нових знань», і тому вимагає досить тривалого часу на проведення теоретичних досліджень, а також великих матеріальних і трудових витрат для проведення експериментальних досліджень та ідентифікації моделі.

До того ж досвідчені оператори за час тривалої роботи навчилися вести процес в оптимальних режимах при різних вихідних ситуаціях. Передача «готових знань» від людей-експертів в базу знань інтелектуальної системи значно спрощує створення інтелектуальних систем, а їх експлуатація дозволяє виключити ефект «людського фактору» при управлінні процесом (це такі властивості людського організму як: втома, недостатньо швидка реакція, недостатня психологічна стійкість, сонливість при монотоній роботі, незначний досвід роботи молодих операторів та інші причини).

Використовуючи ідею розробки замість моделі самого процесу, моделі процесу управління ним в синергії з сучасними здобутками інформаційно-технічного прогресу, пропонується наступну трьохетапну процедуру створення систем оптимального управління технологічними процесами (див. рис. 1.18).

Рисунок 1.18 – Трьохетапна процедура створення гібридних і інтелектуальних систем управління

На першому етапі проводяться апріорні дослідження особливостей об'єкта управління з літературних джерел, публікацій у періодичних виданнях та документації. Як правило, діючі процеси повинні були пройти тривалий етап наукових досліджень, випробувань, перш ніж вони були впроваджені у виробництво. Напевно залишилися матеріали цих досліджень, а також спроби створення математичних моделей даного процесу. Необхідний ретельний аналіз всієї цієї інформації з тим, щоб використовувати її при розробці

інтелектуальних систем управління. Особливо це важливо при можливому створенні гібридних систем управління (ГСУ).

На цьому ж етапі необхідно проаналізувати досліджуваний процес як об'єкт управління з виявленням вхідних і вихідних, контрольованих і неконтрольованих, керованих і некерованих змінних. При цьому необхідно оцінити інерційність об'єкта по різних каналах, клас об'єкта (безперервний або дискретний), ступінь повноти інформації про змінні об'єкта, робочий діапазон зміни змінних об'єкта. Після ретельного аналізу наявної інформації необхідно скласти структуру майбутньої системи управління, що значно полегшить подальшу роботу.

На другому етапі розробляється модель процесу управління. За допомогою досвідчених експертів визначається основна мета управління (аналог цільової функції в оптимізаційних задачах), яка, як правило, відома та яку зазвичай прагнуть досягти досвідчені оператори. Потім методом ранжування зі спільного переліку всіх типів змінних визначаються ті, які, на думку експертів, є основними для даного об'єкта (процесу).

Основним завданням другого етапу є складання матриці планування повного факторного експерименту (ПФЕ). За допомогою матриці ПФЕ [58] створюється модель управління об'єктом (процесом). При цьому, наприклад, для трирівневих факторів повне число можливих поєднань числа факторів при двох вхідних змінних $N = 3^2 = 9$, для трьох змінних – $3^3 = 27$.

Наприклад, при двох вхідних змінних складається матриця планування ПФЕ, наведена в таблиці 1.1.

Таблиці виду 1.1 є основою для розробки інтелектуальних систем, так як в них зосереджений багаторічний досвід, знання та інтуїція людей-експертів у конкретній предметній області. Від якості матриці ПФЕ буде залежати ефективність роботи всієї системи управління.

Таблиця 1.1 – Матриця планування ПФЕ

№ експерименту	X_1	X_2	Y^e оцінка експерта
1	0,0	0,0	
2	0,0	0,5	
3	0,0	1,0	
4	0,5	0,0	
5	0,5	0,5	
6	0,5	1,0	
7	1,0	0,0	
8	1,0	0,5	
9	1,0	1,0	

Величини: 0,0; 0,5; 1,0 означають мінімальне, середнє та максимальне значення вхідних змінних X_1 і X_2. Експертові залишається лише з урахуванням свого досвіду, знань і інтуїції проставити значення вихідної змінної Y^e (керуючого впливу) в діапазоні від 0,0 до 1,0. Нормалізація в діапазоні від 0 до 1 вхідних і вихідних змінних здійснюється за формулою:

$$\bar{x} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \qquad (1.37)$$

де: $\bar{x}$ – нормалізоване (від 0 до 1) значення вхідної або вихідної змінної;

x – поточне значення змінної;

$x_{\min}$, $x_{\max}$ – мінімальне та максимальне значення змінної.

Складання матриці планування експериментів набагато зручніше для експертів, ніж рекомендоване в всіх підручниках і публікаціях складання правил нечітких продукцій виду. При цьому експерту немає необхідності вигадувати нескінченні терми: («дуже багато», «дуже-дуже мало», «цілком нормально») – він просто ставить значення вихідний (керуючої) змінної в діапазоні від 0,0 до 1,0. При цьому матриця планування ПФЕ [58] може бути використана для чотирьох різних методів створення моделі управління: планування експерименту, експертні системи, нейронні мережі, нейро-нечіткі алгоритми.

На відміну від добре відомого класичного методу планування експерименту складання матриці планування ПФЕ за допомогою експертів значно прискорює та здешевлює цю процедуру. Експерти проводять так звані «уявні експерименти» замість дорогих, реально проведених активних експериментів. Крім того, потрібно враховувати, що проведення активних експериментів в умовах діючого виробництва нереально через можливе виникнення аварійних ситуацій при зміні змінних процесу від мінімальних їх значень до максимальних значень, і назад. До того ж на багатьох підприємствах просто немає можливості змінювати змінні згідно матриці планування ПФЕ.

Необхідно підкреслити, що вихідні значення Y_i є насправді керуючими змінними, тому матриця планування відображає модель управління процесом для всіх запланованих експертами поєднань вхідних змінних. Для розрахунку значень у проміжних поєднаннях вхідних змінних (наприклад, для $X_1 = 0,21$ і $X_2 = 0,74$) необхідно синтезувати модель управління процесом, що є основним завданням другого етапу.

Важливим є і те, що матриця планування ПФЕ може бути використана для створення моделі управління чотирма різними методами [59]: планування експерименту, нечітких алгоритмів, нейронних мереж, нейро-нечітких мереж, гібридних моделей.

Необхідно відзначити, що найбільш ефективно спільно з інтелектуальними моделями використовувати відомі математичні залежності, виявлені на першому етапі досліджень. При цьому необхідно бути впевненим у тому, що такі залежності адекватно відображають ті чи інші фізико-хімічні закономірності конкретного процесу.

На третьому етапі проводиться дослідження створених моделей управління. При цьому здійснюються такі заходи.

Отримані моделі піддаються ретельному дослідженню та аналізу їх чутливості, стійкості, однозначності. Для чого проводиться моделювання процесу управління при різних змінах вхідних змінних, будуються криві зміни

вихідних змінних при зміні вхідних змінних, і проводиться їх аналіз спільно з експертами.

Після завершення дослідження моделей, отриманих різними методами, проводиться порівняльний аналіз на їх адекватність. Для чого за допомогою моделей розраховуються вихідні змінні при значеннях вхідних змінних, взятих з матриці планування ПФЕ і порівнюються з оцінками, даними експертом. Після чого формується матриця порівняння (див. таблицю 1.2).

Таблиця 1.2 – Матриця порівнянь розрахункових і експериментальних значень вихідної величини

№ експерименту	X_1	X_2	Y^P оцінка моделі	Y^e оцінка експерта
1	0,0	0,0		
2	0,0	0,5		
3	0,0	1,0		
4	0,5	0,0		
5	0,5	0,5		
6	0,5	1,0		
7	1,0	0,0		
8	1,0	0,5		
9	1,0	1,0		

Помилки моделювання різними способами. Наприклад, абсолютна помилка у відсотках розраховується за формулою:

$$\delta = 100 \frac{1}{N-1} \sum_{i=1}^{N} \left| Y^e - Y^p \right| \qquad (1.38)$$

де Y^e і Y^p – відповідно експериментальні і розрахункові значення вихідних змінних.

Абсолютна помилка розраховується для моделей, отриманих чотирма різними способами, а потім провадить виробництво їх порівняльний аналіз. Модель з найменшою абсолютною помилкою вважається найбільш адекватною.

Найбільш адекватна модель повинна пройти імітаційні випробування в

умовах діючого виробництва [60]. При цьому на вхід моделі подають дійсні вхідні змінні, зняті з вимірювальної апаратури промислового агрегату, а результати моделювання (вихідна керуюча змінна) порівнюється зі значенням управління, реально здійснюваним досвідченим оператором-технологом. У разі задовільного результату імітаційних випробувань проводиться інтеграція моделі в промисловий контролер. В іншому випадку, все починається спочатку – повернення до першого етапу, і уточнення всіх параметрів моделі.

1.5.2 Розробка програмно-технічних засобів для промислових інтелектуальних систем управління

Для реалізації нечітких правил, розроблених в середовищі Fuzzy Logic Toolbox пакету MatLab, в системі управління технологічним процесом, їх необхідно завантажити в програмований логічний контролер (PLC) [61]. При цьому потрібно зробити інтеграцію контролера із середовищем моделювання Simulink пакета MatLab, в яку включають систему нечітких правил, розроблену в вищеназваному блоці Fuzzy Logic ToolBox [62].

HC900 є вдосконаленим контролер, який реалізує контурне і логічне управління і має модульну конструкцію, що дозволяє задовольнити вимоги управління та збору даних для широкого діапазону технологічного обладнання. Контролер забезпечує чудову якість управління на базі замкнутого контуру ПІД – регулювання (пропорційно-інтегрально-диференціального) і більш стійку обробку аналогових сигналів, ніж більшість логічних контролерів, без погіршення ефективності виконання логічних операцій. Передбачено окремий цикл швидкого сканування для виконання широкого асортименту логічних і обчислювальних функціональних блоків [63]. Логічні блоки можуть також виконуватися одночасно з аналоговими функціональними блоками. Ці функціональні блоки можна повністю інтегрувати в комбіновану стратегію управління аналоговими і логічними величинами для забезпечення стійкої ефективності управління.

Для інтеграції використовується технологія OPC. – це набір повсюдно прийнятих специфікацій, що надають універсальний механізм обміну даними в системах контролю і управління. Абревіатура OPC розшифровується як OLE for Process Control. OLE – це Object Linking and Embedding – технологія зв'язування та впровадження об'єктів в інші документи і об'єкти, розроблені корпорацією Майкрософт.

OPC заснована на моделі розподілених компонентних об'єктів Microsoft COM / DCOM і встановлює вимоги до класів об'єктів доступу до даних і їх спеціалізованим (custom) інтерфейсів для використання розробниками клієнтських і серверних додатків.

Розробку стандартів OPC, їх опис, підтримку і пропаганду веде добровільна міжнародна організація OPC Foundation, розташована в містечку Boca Raton, штат Флорида США. Організація налічує понад 250 членів, в числі яких компанії, що займають лідируючі позиції в області автоматизації: Honeywell, Fisher-Rosemount, Siemens, Wonderware, Intellution і інші.

На сьогоднішній день технологія OPC є стандартом в області побудови систем автоматизації. OPC-сервер являє собою програмне середовище, що забезпечує одночасний уніфікований спосіб доступу до даних для різних програмних пакетів. Це можуть бути SCADA-пакети різних виробників або інше програмне забезпечення.

В даний час існує декілька специфікацій OPC, що знайшли своє застосування в різних областях автоматизації і передачі даних. В описуваному методі використовується специфікація OPC DataAccess (OPC DA). Це основний і найбільш затребуваний стандарт. Він описує набір функцій обміну даними в реальному часі з ПЛК і іншими пристроями.

Переваги використання OPC в АСУ [64]: пристосовність системи до різних мереж, масштабованість і модернізованих. Додатковою перевагою є підтримка стандарту OPC виробниками продуктів для автоматизованих систем. Ця підтримка дозволяє об'єднати різні компоненти в одній системі.

OPC-сервер являє собою структуру, що має наступну ієрархію, показану на рисунку 1.19.

З боку клієнта ця структура представляється у вигляді стандартного для OPC набору груп і тегів, що належать даному екземпляру сервера. Ім'я групи утворюється як Channel [n] .Device [m], ім'я тега – як відповідне ім'я, вказане при створенні тега при конфігуруванні сервера.

OPC сервер використовується як сполучна ланка між середовищем моделювання і контролером. Передача даних між середовищем моделювання і сервером підтримується на рівні Simulink, які мають в своєму складі бібліотеку OPC Toolbox, що дозволяє конфігурувати OPC клієнт. Передача даних між контролером і сервером підтримується на рівні драйверів, які в даному випадку входять до складу дистрибутива OPC сервера.

Для налаштування роботи сервера з контролером потрібно створити новий канал. На цьому етапі вибирається драйвер пристрою і вказуються параметри з'єднання. В даному випадку зв'язок з контролером здійснюється через COM-порт за допомогою конвертера COM-PPI.

Далі необхідно створити новий пристрій. На цьому етапі вибирається тип контролера зі списку підтримуваних обраним драйвером пристроїв. Далі необхідно додати теги, в які буде писати дані контролер і з яких він буде отримувати інформацію. Зв'язок тега з певної змінної контролера здійснюється шляхом зазначення адреси змінної в пам'яті контролера при створенні тега. Кількість тегів і їх прив'язка до змінних залежать від завантаженої в контролер програми.

Для забезпечення роботи Simulink з OPC в модель необхідно додати об'єкти OPC Configuration, OPC Read і OPC Write. Ці об'єкти знаходяться в розділі OPC Toolbox середовища Simulink.

Об'єкт OPC Configuration служить для конфігурації зв'язку з OPC-сервером. Підтримуються як локальні сервери, так і сервери, що знаходяться в мережі. Підтримується одночасна робота з декількома серверами.

Об'єкт OPC Read служить для читання значення зазначеного тега сервера. Даний об'єкт має три виходи – V, Q і T. На вихід V (value) надходить безпосередньо значення, на вихід Q (quality) – його показник якості, на вихід T (timestamp) – час останнього оновлення тега.

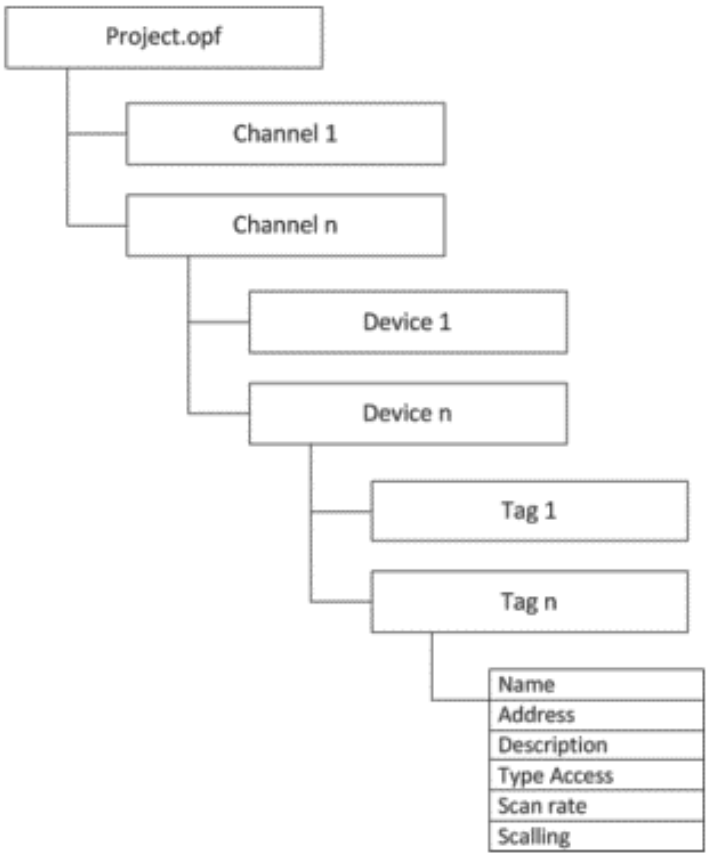

Рисунок 1.19 – Ієрархія OPC-сервера

Об'єкт OPC Write служить для запису інформації в зазначений тег сервера. Читання і запис відбуваються з періодом, рівним кроку моделювання. Для забезпечення коректної роботи крок моделювання необхідно вказувати рівним величині scan rate, що задається при конфігуруванні сервера. При роботі моделі з OPC-сервером моделювання відбувається в реальному часі.

Далі виробляємо запис даних з Simulink в контролер. Обробка імпортованих з Simulink нечітких правил виконується за допомогою стандартних функціональних модулів нечіткого управління контролера НС900.

Висновки до розділу

Штучний інтелект в своєму розвитку пройшов етапи становлення теоретичних розробок, практичних застосувань в області експертних і проблемно-орієнтованих інтелектуальних систем. В даний час відбувається його розширення в області штучних когнітивних систем і систем з креативними здібностями.

Сучасні інтелектуальні системи обробки інформації та управління досягли рівня, на якому виявляється можливим використання глибинних знань про пристрій і роботу нервової системи людини. Використання когнітивного підходу, що виник в психології, при розробці інтелектуальних систем дозволило наблизити їх за функціями до нервової системи людини. В результаті очікується створення все більш потужних інтелектуальних інформаційних і керуючих систем з навчанням і самонавчанням. Вони будуть мати посилені когнітивні можливості за рахунок симуляції когнітивних функцій і процесів, які в нервовій системі людини відповідають за сприйняття, навчання, мислення, свідомість. Розвиток цього напрямку дозволить створювати інтелектуальні системи з елементами мислення та усвідомлення поведінки. Додавання деяких креативних здібностей, пов'язаних, наприклад, з автоматичною побудовою гіпотез і моделей, а також самонавчанням для вирішення нових завдань, дозволить ще більше підвищити ефективність інтелектуальних систем. У зв'язку з цим з'явилися нові напрямки, пов'язані з розробкою штучного мозку і штучної нервової системи роботів, які стосуються не просто штучного інтелекту, а також його розвитку у вигляді штучного розуму.

Інтелектуальні технології, що дозволяють створювати практично корисні інтелектуальні системи, знаходяться в безперервному розвитку. В даний час для реалізації технології експертних систем, нечітко-логічних і нейромережевих систем, а також технології багатоагентних систем є досить потужні інструментальні засоби. Вони швидко удосконалюються шляхом додавання пакетів програм і засобів проектування апаратних реалізацій.

Розвиваються нові технології в області так званих нейроморфних систем, що моделюють деякі структури мозку, а також в області паралельних обчислень і квантових комп'ютерів. Ці технології повинні істотно підняти рівень інтелектуальності систем в майбутньому.

Потрібно відзначити, що розвиток інтелектуальних інформаційних і керуючих систем в останні роки дає все більш значущі практичні результати. Досить навести приклади створення «розумних» інформаційних систем, які перемагають людину в шаховій грі, а також систем управління гуманоїдними роботами, що змагаються між собою на всесвітніх чемпіонатах з футболу роботів.

Однак, до створення інтелектуальних систем, що володіють розумністю людини, поки ще далеко. Вивчення та розвиток нових підходів та інтелектуальних методів вирішення важкоформалізованих завдань обробки інформації та управління дозволить забезпечити розвиток у цьому напрямку. Перспективами реалізації такого вектору є створення розумних систем, здатних обробляти інформацію як людина, або штучних нервових систем гуманоїдних роботів, справжніх помічників і друзів людини в найближчому майбутньому.

Список використаних джерел

1. Юкаева В.С., Зубарева Е.В., Чувикова В.В. Принятие управленческих решений : учебник. М. : Издательскоторговая корпорация «Дашков и К°», 2012. 324 с.

2. Rinaldo Macedo de Morais, Samir Kazan, Silvia Inês Dallavalle de Pádua, André Lucirton Costa An analysis of BPM lifecycles: from a literature review to a framework proposal. *Business Process Management Journal*. 2014. Vol. 20 Issue: 3, P. 412–432, doi: 10.1108/BPMJ-03-2013-0035

3. Baker N., Alexander F., Bremer T., Hagberg A., Kevrekidis Y., Najm H. & Lee S. Workshop report on basic research needs for scientific machine learning: Core technologies for artificial intelligence. USDOE Office of Science (SC), Washington,

DC (United States), 2019 URL: https://www.osti.gov/servlets/purl/1478744?ref=https://githubhelp.com.

4. Карелин В.П. Интеллектуальные технологии и системы искусственного интеллекта для поддержки принятия решений. *Вестник Таганрогского института управления и экономики*. 2011. № 2. С. 79–84.

5. Pavlo A., Paulson E., Rasin A., Abadi D.J., DeWitt D.J., Madden S. & Stonebraker M. A comparison of approaches to large-scale data analysis. In Proceedings of the 2009 ACM SIGMOD. *International Conference on Management of data*, 2009, June, P. 165–178.

6. Безшанова А.Е., Миронова А.И., Моргунова О.В. Экспертные системы: теоретические аспекты. *Физика и медицина: создавая будущее* : матриалы III научно-практической конференции студентов и молодых ученых, 2017. С. 151–156.

7. Cavalcante R. C., Brasileiro R.C., Souza V.L., Nobrega J.P. & Oliveira A.L. Computational intelligence and financial markets: *A survey and future directions. Expert Systems with Applications*. 2016. № 55, P. 194–211.

8. Едил Б.К., Касимова Б.Р. Создание алгоритма работы экспертной системы для определения состава оборудования гибридной системы энергоснабжения на альтернативных источниках энергии. *Информационные технологии в экологии:* материалы Всероссийской научно-практической конференции, посвященной Году экологии в России. 2018. С. 121–127.

9. Dogac A., Yuruten Birol, & Spaccapietra S. A generalized expert system for database design. *IEEE Transactions on Software Engineering*. 1989. № 15(4). P. 479–491.

10. Корабльов М.М., Овчаренко І.В. Адаптація моделей нечіткого виводу з використанням штучних імунних систем. 2007. С. 73–76. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2017/dec/7246/15.pdf

11. Фомина А.В. и др. Управление развитием высокотехнологичных предприятий наукоемких отраслей промышленности. *Scientific magazine*

Kontsep, 2014. 400 с.

12. Van Rooij R. Vagueness and linguistics. *In Vagueness: A guide*. Springer, Dordrecht. 2011. P. 123–170.

13. Костикова А.В., Скитер Н.Н. Формирование динамической базы знаний систем нечеткого вывода для оценки объектов, изменяющихся во времени. *E-Management*. 2018. Т. 1. №. 1. С. 52–59.

14. Кравець П., Киркало Р. Системи прийняття рішень з нечіткою логікою. 2009. С. 115–123. URL: http://vlp.com.ua/files/special/17_0.pdf

15. Перепелиця О.О. Проектування інтелектуальної системи на основі нечітких знань. 2019. URL: http://eprints.library.odeku.edu.ua/id/eprint/5972/1/Perepelija_Proektuvannja_intelektualnoj_system.pdf

16. Dagli C.H. (ed.). Artificial neural networks for intelligent manufacturing. *Springer Science & Business Media*. 2012. 469 p.

17. Branco T., Häusser M. The single dendritic branch as a fundamental functional unit in the nervous system. *Current opinion in neurobiology*. 2010. Т. 20. №. 4. С. 494–502.

18. Турчин В.Ф. Феномен науки: кибернетический подход к эволюции М. : ЭТС. 2000. Т. 368. С. 7.

19. Колесниченко К.В., Колесниченко И.П. Математическая модель нейрона. *Лесной вестник. Forestry bulletin*. 2005. №. 4. С. 107–116.

20. Montague P.R., Dayan P. & Sejnowski T.J. A framework for mesencephalic dopamine systems based on predictive Hebbian learning. Journal of neuroscience. 1996. №16(5) P. 1936–1947.

21. Кузин Д.А., Запевалов А.В., Сырчин А.В. Реляционная модель представлениямногослойного персептрона. *Современные наукоемкие технологии*. 2013. №. 4. С. 45–48.

22. Субботін С.О. Нейронні мережі. Навчальний посібник. 2014. Запоріжжя : ЗНТУ, 2014.132 с.

23. Карлов Д.Н. Алгоритм скоростного обучения многослойного

персептрона. *Электронный сетевой политематический журнал «Научные труды КубГТУ»*. 2018. №. 3. С. 167–173.

24. Мусакулова Ж.А. Модель нейрона с входной сигмоидальной функцией активации. *Перспективы развития информационных технологий*. 2012. №. 7. С.90–96.

25. Васенков Д.В. Методы обучения искусственных нейронных сетей. *Компьютерные инструменты в образовании*. 2007. №. 1. С. 20–29.

26. Зимина С.В. Флуктуации весовых коэффициентов в искусственной нейронной сети с алгоритмом Хэбба. *Нейрокомпьютеры: разработка и применение*. 2013. Т. 4. С. 3–8.

27. Кононюк А.Ю. Нейроні мережі і генетичні алгоритми. К. : «Корнійчук», 2008. 446 с.

28. Whittington J.C. & Bogacz R. An approximation of the error backpropagation algorithm in a predictive coding network with local Hebbian synaptic plasticity. *Neural computation*. 2017. №29(5). P. 1229–1262.

29. Коцовський В. Навчання штучних комплексних нейронних мереж методом зворотного поширення помилки. *Інформаційні технології як інноваційний шлях розвитку України у XXI столітті* : матеріали І Всеукраїнської науково-практичної конференції молодих науковців. Ужгород: ЗакДУ, 2013. С. 63–65.

30. Татьянкин В.М. Модифицированный алгоритм обратного распространения ошибки. *Приоритетные направления развития науки и образования*. 2014. №. 3. С. 197–198.

31. Горелова А.В., Любимова Т.В. Алгоритм обратного распространения ошибки. *Наука и современность*. 2015. №. 38. С. 151–158.

32. Lazovskaya T. & Tarkhov D. Multilayer neural network models based on grid methods. *In IOP conference series: materials science and engineering*, 2016, November. Vol. 158, No. 1. P. 12–61.

33. Калініна І.О. Дослідження алгоритмів навчання нейронних мереж в

задачах прогнозування. *Наукові праці Чорноморського державного університету імені Петра Могили. Сер.: Комп'ютерні технології*. 2009. №. 17, Вип. 104. С. 160–171.

34. Ilonen J., Kamarainen J.K. & Lampinen J. Differential evolution training algorithm for feed-forward neural networks. *Neural Processing Letters*. 2003. № 17(1). P. 93–105.

35. Бессмертный И.А. Искусственный интеллект : учебное пособие. СПб : СПбГУ ИТМО. 2010. 132 с..

36. Panchal, G., Ganatra, A., Shah, P., & Panchal, D. Determination of over-learning and over-fitting problem in back propagation neural network. *International Journal on Soft Computing*. 2011. № 2(2). P. 40–51.

37. Зайко Т.А., Олійник А.О., Субботін С.О. Побудова нейро-нечітких моделей на основі неструктурованих даних. *Штучний інтелект*. 2012. № 4. С. 546–556.

38. Walia N., Singh H. & Sharma A. ANFIS: Adaptive neuro-fuzzy inference system-a survey. *International Journal of Computer Applications*. 2015. № 123 (13). P. 32–38.

39. Papanikolaou K.A., Grigoriadou M., Kornilakis H. & Magoulas G.D. INSPIRE: an intelligent system for personalized instruction in a remote environment. *In Workshop on Adaptive Hypermedia*. Springer, Berlin, Heidelberg. 2001, August. P. 215–225.

40. Пупков К.А. Современные методы, модели и алгоритмы интеллектуальных систем. М. : РУДН. 2008. 154 с.

41. Stefanuk, V.L. Dynamic expert systems. *Kybernetes*. 2000. Vol. 29 No. 5/6, P. 702–709.

42. Рыбина Г.В., Старостенко К.И. Современная инструментальная база для построения динамических экспертных систем. *Информационно-измерительные и управляющие системы*. 2009. Т. 7. №. 8. С. 73–78.

43. Козлов А.Н. Интеллектуальные информационные системы : учебник.

Мин-во с-х. РФ, ФГБОУ ВПО Пермская ГСХА. Пермь: Изд-во ФГБОУ ВПО Пермская ГСХА, 2013. 278 с.

44. Redreev G.V., Kiyko P.V. & Shchukina N.V. Optimizing the Use of Expert Systems by Varying Databases. *In 2019 International Multi-Conference on Industrial Engineering and Modern Technologies (FarEastCon.* 2019, October. P. 1–4.

45. Сергієнко Л.Г., Більмак Т.С. Аналіз проблеми інтеграції в динамічних інтелектуальних системах і особливості сучасних інтелектуальних систем управління. *Збірник наукових праць за результатами всеукраїнської науково-практичної конференції «Сучасні аспекти механізації та автоматизації енергоємних виробництв».* Індустріальний інститут ДВНЗ ДонНТУ, 11-12 квітня 2017 року. С. 379.

46. Рыбина Г.В. Современные архитектуры динамических интеллектуальных систем: проблемы интеграции и основные тенденции. *Приборы и системы. Управление, контроль, диагностика*. 2017. №. 2. С. 1–12.

47. Волков С.Л., Казакова Н.Ф., Асабашвілі С.Д. Модель експертного оцінювання якісного стану технічної системи. *Вісник Національного технічного університету «ХПІ» : Механіко-технологічні системи та комплекси*. Х. : НТУ «ХПІ». 2017. № 44. С. 157–161.

48. Vitkus D., Salter J., Goranin N. & Čeponis D. Method for Attack Tree Data Transformation and Import Into IT Risk Analysis Expert Systems. *Applied Sciences.* 2020. № 10 (23). P. 84–23.

49. Тулегулов А.Д. и др. Методы унификации компонентов робототехнических систем. *Труды международного симпозиума «Надежность и качество»*. 2020. Т. 2. С. 241–243.

50. Адамик О.В., Сисюк С.В. Інформаційні системи управління підприємством: вибір базових технологій та програмного забезпечення *Глобальні та національні проблеми економіки*. 2016. Випуск № 14 (Грудень). С. 891–895.

51. Benantar M. Access control systems: security, identity management and trust models. *Springer Science & Business Media*. 2005. P.260

52. Fleming P.J., Purshouse R.C. Evolutionary algorithms in control systems engineering: a survey. *Control engineering practice*. 2002. Т. 10. №. 11. С. 1223–1241.

53. Нестеренко О.В., Савенков О.І., Фаловський О.О. Інтелектуальні системи підтримки прийняття рішень: навч. посібн. / за ред. П.І. Бідюка. Київ: Національна академія управління. 2016. 188 с.

54. Ponce-Cruz P. & Ramírez-Figueroa F.D. Intelligent control systems with LabVIEWTM. Springer Science & Business Media. 2009. P. 215.

55. Карелин В.П. Интеллектуальные технологии и системы искусственного интеллекта для поддержки принятия решений. *Вестник Таганрогского института управления и экономики*. 2011. №. 2. С. 79–84

56. Wilson C., Marchetti F., Di Carlo M., Riccardi A., & Minisci E. Classifying intelligence in machines: a taxonomy of intelligent control. *Robotics*. 2020. № 9 (3). 64 p.

57. Чорноус Г.О. Розробка інтелектуальної агентно-орієнтованої системи підтримки прийняття рішень на підприємстві. *Вісник Київського національного університету ім. Тараса Шевченка. Серія: Економіка*. 2014. № 160. С. 101–109.

58. Полупан К.Л. и др. Создание гибридной интеллектуальной системы управления рисками в условиях цифровой экономики. *Развитие цифровой экономики в условиях деглобализации и рецессии*. 2019. С. 61–96.

59. Романов А.В. Алгоритм и программное обеспечение идентификации ячеек радиотехнических устройств с демпфирующими ребрами. *Труды Международного симпозиума «Надежность и качество»*. 2012. Т. 1. С 245–252.

60. Мальков М.В., Олейник А.Г., Федоров А.М. Моделирование технологических процессов: методы и опыт. *Труды Кольского научного центра РАН*. 2010. №. 3. С 124–133.

61. Johnstone K., Blair S.M., Syed M.H., Emhemed A., Burt G.M., & Strasser T.I. Co-simulation approach using PowerFactory and MATLAB/Simulink to enable validation of distributed control concepts within future power systems. *CIRED-Open Access Proceedings Journal*. 2017(1). P. 2192–2196.

62. Цідило І.М. Вивчення майбутніми інженерами-педагогами комп'ютерного профілю пакету Fuzzy Logic Toolbox. *Сучасні інформаційні технології та інноваційні методики навчання в підготовці фахівців: методологія, теорія, досвід, проблеми*. 2013. №. 34. С. 468–473.

63. Popescu M. Hierarchical Control of a Fractionation Column using HC900 Controller. Petroleum-Gas University of Ploiesti Bulletin, Technical Series. 2009. № 61(3). P. 229-234.

64. Крюков М.Ю. Использование технологии OPC для промышленных АСУ ТП. Научное сообщество студентов. 2016. С. 176–177.