

Університет економіки та права “КРОК”

В.В. Троцько

МЕТОДИ ШТУЧНОГО ІНТЕЛЕКТУ

Навчально-методичний посібник

Київ-2019

ЗМІСТ

ВСТУП	2
РОЗДІЛ 1. ГЕНЕТИЧНІ АЛГОРИТМИ	3
1.1. Сутність генетичних алгоритмів	3
1.2. Приклад виконання класичного генетичного алгоритму	6
1.3. Мутації в генетичних алгоритмах	10
1.4. Різновиди генетичних алгоритмів	14
1.5. Використання генетичних алгоритмів	16
РОЗДІЛ 2. НЕЙРОННІ МЕРЕЖІ	17
2.1. Штучний нейрон.....	17
2.2. Моделювання логічних функцій за допомогою штучних нейронів.....	21
2.3. Навчання штучного нейрона	23
2.4. Штучні нейронні мережі.....	27
2.5. Мережа Гопфілда.....	27
2.6. Багатошарові нейронні мережі.....	31
2.7. Метод зворотного поширення помилки.....	32
2.8. Використання методу зворотного поширення помилки для прогнозування часових рядів	39
2.9. Мережі Кохонена.....	41
2.10. Карти Кохонена	49
2.11. Інші різновиди нейронних мереж	50
2.12. Практичне використання штучних нейронних мереж	51
РОЗДІЛ 3. ІНТЕЛЕКТУАЛЬНІ АГЕНТИ.....	53
3.1. Поняття інтелектуального агента	53
3.2. Алгоритм Q-навчання інтелектуальних агентів.....	53
3.3. Використання інтелектуальних агентів	59
РОЗДІЛ 4. КОЛЕКТИВНИЙ ІНТЕЛЕКТ.....	61
4.1. Визначення колективного інтелекту	61
4.2. Використання мурашиного алгоритму для вирішення задачі комівояжера	61
ПРАКТИЧНІ ЗАВДАННЯ	68
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....	93

ВСТУП

Посібник призначений для використання студентами, які вивчають курс «Штучний інтелект» за спеціальністю «Комп'ютерні науки». В ньому розглянуті теми, які є найскладнішими з теоретичної точки зору. До цих тем належать: «Генетичні алгоритми», «Штучні нейронні мережі», «Інтелектуальні агенти» та «Колективний інтелект». Поряд з теоретичними питаннями особлива увага в посібнику приділена питанням практичної реалізації розглядуваних тем. В кінці кожного розділу міститься набір питань, відповідаючи на які студенти зможуть закріпити навчальний матеріал.

Для набуття більш досконалих практичних навичок в роботі з окремими методами штучного інтелекту та полегшення можливості програмної реалізації в додатку наведений перелік лабораторних робіт, які виконуються під час вивчення курсу із варіантами завдань щодо їх виконання. Для більш глибокого вивчення окремих методів штучного інтелекту і розширення своїх знань в цій сфері пропонується відповідний список літератури.

Зазначений посібник може також бути корисним для студентам інших напрямів навчання та тим, хто цікавиться з методами штучного інтелекту.

РОЗДІЛ 1. ГЕНЕТИЧНІ АЛГОРИТМИ

1.1. Сутність генетичних алгоритмів

Генетичний алгоритм – це алгоритм пошуку, що використовується для вирішення окремих задач через здійснення підбору, комбінування і варіації окремих параметрів з використанням методів та прийомів, що нагадують процеси біологічної еволюції. Враховуючи це, в генетичних алгоритмах використовуються терміни, подібні до тих, які використовується в біології (популяція, мутація, покоління, схрещування, потомство, хромосоми, гени тощо). Генетичні алгоритми є частиною так званого еволюційного моделювання, яке використовує положення теорії Дарвіна про еволюцію видів для побудови систем подібних до інтелектуальних. Особливістю генетичних алгоритмів є те, що вони мають в основному описовий характер. Розрахунки в них вкрай спрощені і це дозволяє досить швидко зрозуміти сутність їх роботи і успішно застосовувати на практиці.

Досягнення генетики дозволили виявити, що під час біологічного відбору в групі живих істот одного виду (популяції) виживають ті особини які мають набір хромосом, що дозволяють пристосовуватися до умов середовища проживання з максимальною ефективністю. Такий природний відбір характеризує весь механізм еволюції живих істот, відповідно до якого частка істот з максимальною пристосованістю до умов навколишнього середовища повинна зростати. Цей процес особливо помітний на тих біологічних видах, розмноження яких відбувається досить швидко і процес зміни поколінь займає нетривалий період часу (наприклад, комахи або щурі). Після зміни десятків поколінь пристосованість всієї популяції помітно зростає (наприклад, змінюється колір особин відповідно до кольору місцевості).

Механізм відтворення особин в популяції являє собою процес злиття пар хромосом, які успадковуються від батька та матері їх нащадками. Цей процес називається кросовером (crossover), під час якого ДНК матері і батька розпаровуються і зливаються в одну ДНК нащадка випадковим чином. Тобто дитина отримує ознаки обох батьків та якісь власні ознаки, спричинені поєднанням пар ДНК.

Іншим важливим фактором, який впливає на спадковість особин є мутація. Мутація являє собою випадкову зміну ДНК під впливом певних чинників. Одним із таких відомих чинників є радіація. Вважається, що мутація спричиняє появу нових біологічних видів. Вона також може призводити до негативних наслідків для особин, спричиняючи хвороби і послаблюючи життєздатність організму.

Отже, для реалізації генетичного алгоритму необхідне моделювання відтворення описаного процесу біологічної еволюції для певної кількості особин або популяції. При цьому так як і в реальних умовах під час реалізації генетичних алгоритмів можливе штучне втручання в процес природного відбору з метою отримання властивостей певних особин. Таке втручання здавна відомо людям, які використовували і використовують його для виведення нових порід тварин. При такому втручанні відбираються окремі особини з певними властивостями (наприклад кольором очей або забарвленням шерсті) і в подальшому створюються умови для схрещування цих особин між собою з метою збереження необхідних властивостей. Такий підхід називається елітизмом.

Для реалізації генетичного алгоритму необхідно здійснити:

- моделювання створення початкової популяції;
- забезпечити врахування впливу навколишнього середовища на особин популяції (здійснити врахування пристосованості особин в популяції- відбір);
- моделювання поведінки хромосом під час схрещування(розмноження);
- врахувати вплив мутацій на особин популяції;
- змодельовати відтворення нового покоління в популяції.

До етапів «класичного» генетичного алгоритму належать [1]:

- ініціалізація, або вибір вихідної популяції хромосом;
- оцінка пристосованості хромосом в популяції;
- перевірка умови зупинки алгоритму;
- селекція хромосом;
- застосування генетичних операторів;
- формування нової популяції;
- вибір найбільш ефективної хромосоми.

До генетичних операторів відносяться оператори схрещування та оператори мутації. Оператор схрещування забезпечує обмін між хромосомами частинами ДНК, а оператор мутації виконує зміну хромосоми випадковим

чином. Узагальнена блок-схема генетичного алгоритму наведена на рис. 1.1. Таку блок-схему вважають «класичною». Однак, існує декілька варіацій генетичного алгоритму, які відрізняються одна від одної умовами виконання зазначених етапів. До таких алгоритмів відносяться – гібридний алгоритм (алгоритм Девіса), острівна модель (Island model), модель СНС та деякі інші[2-4].

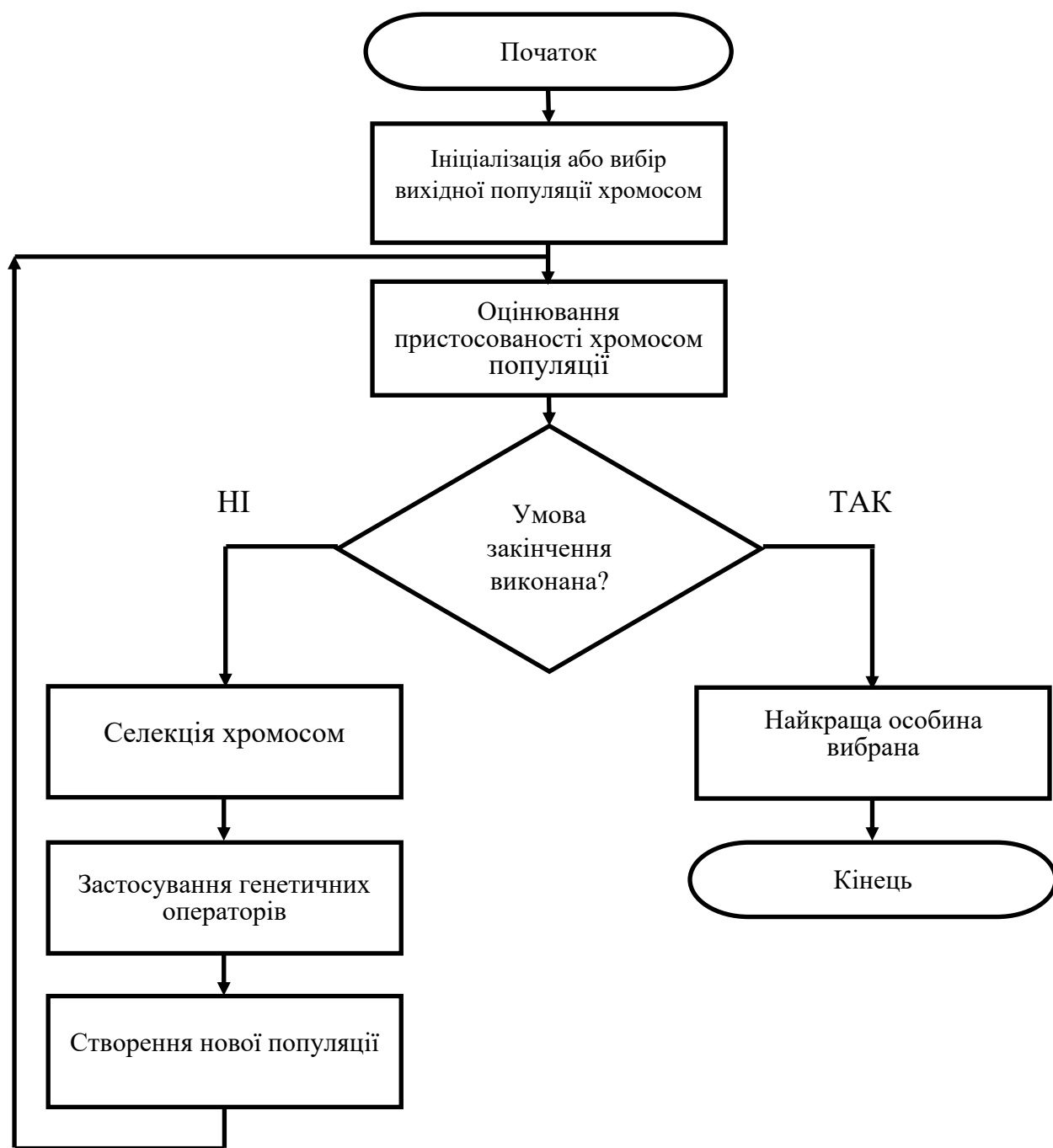


Рис. 1.1. Узагальнена блок-схема генетичного алгоритму

1.2. Приклад виконання класичного генетичного алгоритму

Цей приклад є суттєво спрощеним, що дозволяє зрозуміти послідовність виконання генетичного алгоритму і отримання певних результатів за допомогою нього.

Нехай хромосоми складаються з 13 генів, а популяція складається з 10 наборів хромосом. Згенерувати таку популяцію, тобто виконати етап алгоритму «Ініціалізація, або вибір вихідної популяції хромосом» можна за допомогою стандартних засобів генерування випадкових чисел програми Microsoft Excel або подібного табличного процесора (табл. 1.1).

Таблиця 1.1

Вихідна популяція прикладу

Набори популяції	Хромосоми												
Набір 1	0	0	1	0	0	1	0	1	0	1	1	1	1
Набір 2	0	1	1	0	1	0	0	1	1	0	1	1	0
Набір 3	1	0	1	1	0	1	1	0	1	1	1	0	1
Набір 4	1	0	1	0	0	0	0	0	1	1	0	0	0
Набір 5	1	0	1	1	1	1	0	1	1	1	0	1	0
Набір 6	0	1	1	1	1	1	1	1	1	0	1	0	1
Набір 7	0	0	0	1	0	1	1	1	1	1	0	0	1
Набір 8	1	1	1	1	1	1	0	1	0	1	0	1	1
Набір 9	0	1	0	0	1	1	0	0	0	1	1	0	1
Набір 10	0	1	0	1	0	1	1	0	0	1	0	1	0

Завдання полягає в тому, щоб вивести «ідеальний набір». Такий набір буде найбільш пристосованим до умов навколишнього середовища і складатиметься з 13 одиниць.

Етап алгоритму «Оцінювання пристосованості хромосом в популяції» виконаємо через підрахунок кількості одиниць для кожного набору, табл. 1.2.

Таблиця 1.2

Пристосованість наборів хромосом в популяції прикладу

Набори популяції	Функція пристосованост	Частка від загальної кількості, %
Набір 1	7	9,33%
Набір 2	7	9,33%
Набір 3	9	12,00%
Набір 4	4	5,33%
Набір 5	9	12,00%
Набір 6	10	13,33%
Набір 7	7	9,33%
Набір 8	10	13,33%
Набір 9	6	8,00%
Набір 10	6	8,00%

Із другої колонки табл. 1.2 видно, що найбільш пристосованими є набори 6 та 8 (по 10 хромосом мають бажане значення 1). Однак, «ідеального набору» серед представників популяції яка б мала 13 одиниць в хромосомі немає (умова закінчення алгоритму не виконана). Необхідно переходити до етапу селекції хромосом. Для цього розраховується частка яку вносить кожен набір в загальну пристосованість популяції, третя колонка табл. 1.2. Після цього створюється так зване “колесо рулетки” [5] – коло, розділене на сектори, що являють собою зазначену частку.

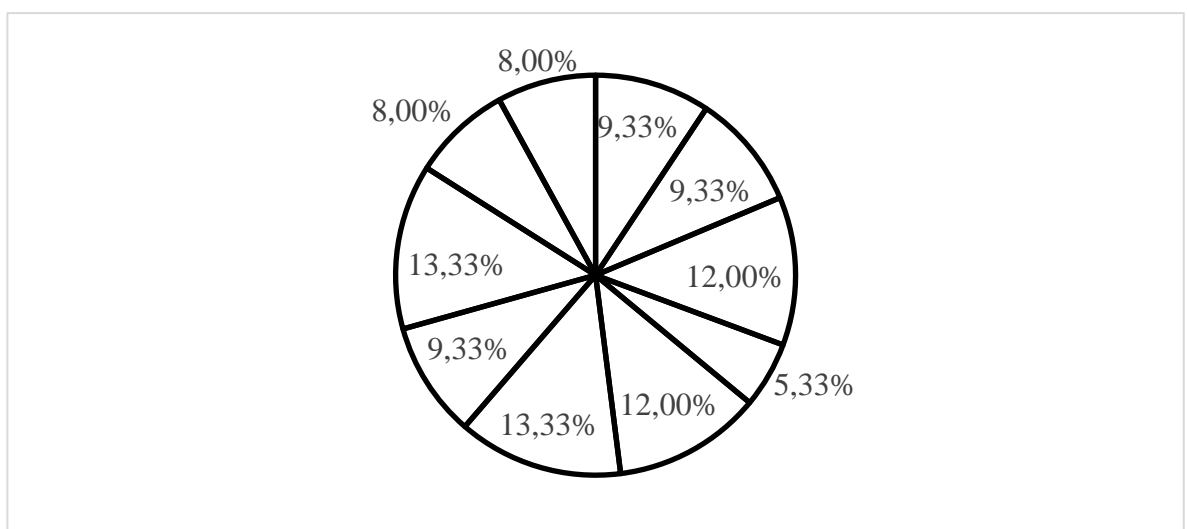


Рис. 1.2. Зовнішній вигляд колеса рулетки для розглядуваного прикладу

Селекція відбувається шляхом генерації випадкових чисел і попадання їх до того чи іншого сектору колеса рулетки. До якого сектору попало число, той набір хромосом виявився життєздатним і є обраним для подальшого розмноження. Генерація випадкових чисел та їх попадання в певні сектори колеса рулетки також може бути виконана стандартними засобами табличних процесорів. Результат такої генерації і подальшого аналізу хромосомних наборів наведений в табл. 1.3. За наявності певних навичок використання колеса рулетки не обов'язкове. Воно лише забезпечує більшу наглядність.

Таблиця 1.3

Результат селекції з поясненням

Набори популяції	Функція пристосов.	Частка від загальної кількості, %	Наростання відсотків	Випадкове число	Набори хромосом які селектовані
Набір 1	7	9,33%	9,33%	24	Набір 3 (між 18,67% та 30,67%)
Набір 2	7	9,33%	18,67%	34	Набір 4
Набір 3	9	12,00%	30,67%	23	Набір 3
Набір 4	4	5,33%	36,00%	86	Набір 9
Набір 5	9	12,00%	48,00%	3	Набір 1
Набір 6	10	13,33%	61,33%	82	Набір 8
Набір 7	7	9,33%	70,67%	22	Набір 3
Набір 8	10	13,33%	84,00%	27	Набір 3
Набір 9	6	8,00%	92,00%	92	Набір 10
Набір 10	6	8,00%	100,00%	13	Набір 2

Отже в результаті селекції «вижили» особини з наборами хромосом під номерами 1,2,3,4,8,9,10. Саме ці набори стануть створювати нову популяцію. Випадково особливо стійкими виявилися набори хромосом (не особини) під номером 3.

Оскільки приклад є спрощеним, можливістю мутації хромосом можна знехтувати. При цьому вважатимемо, що всі набори повинні схрещуватися, тобто ймовірність мутації дорівнює нулю, а ймовірність схрещування одиниці.

Для здійснення схрещування випадково оберемо окремі точки схрещування. Оскільки наборів 10 то точок схрещування повинно бути 5 (батько і мати утворюють пару). Обрані точки схрещування 4, 7, 10, 2, 8.

Утворимо із селектованих наборів пари, із табл. 1.2 та останньої колонки табл. 1.3. Для цих пар виставимо точки схрещування і здійснимо схрещування між ними. Схрещування передбачає обмін генами між наборами хромосом. До точки схрещування ніякого обміну не відбувається, але після точки схрещування здійснюється обмін – якщо в клітинках різні значення 0-1 або 1-0 вони міняються місцями, а якщо однакові все залишається як є і ніякого обміну не відбувається. Як видно на рис. 1.3 в четвертій парі два набори хромосом абсолютно однакові і між ними ніякого схрещування не відбудеться. Зате в третій парі (набір 1 та набір 8) після схрещування в новоутвореному наборі (набір н6) кількість одиниць збільшилася до 11. Отже для цього набору значення функції пристосованості зросло.

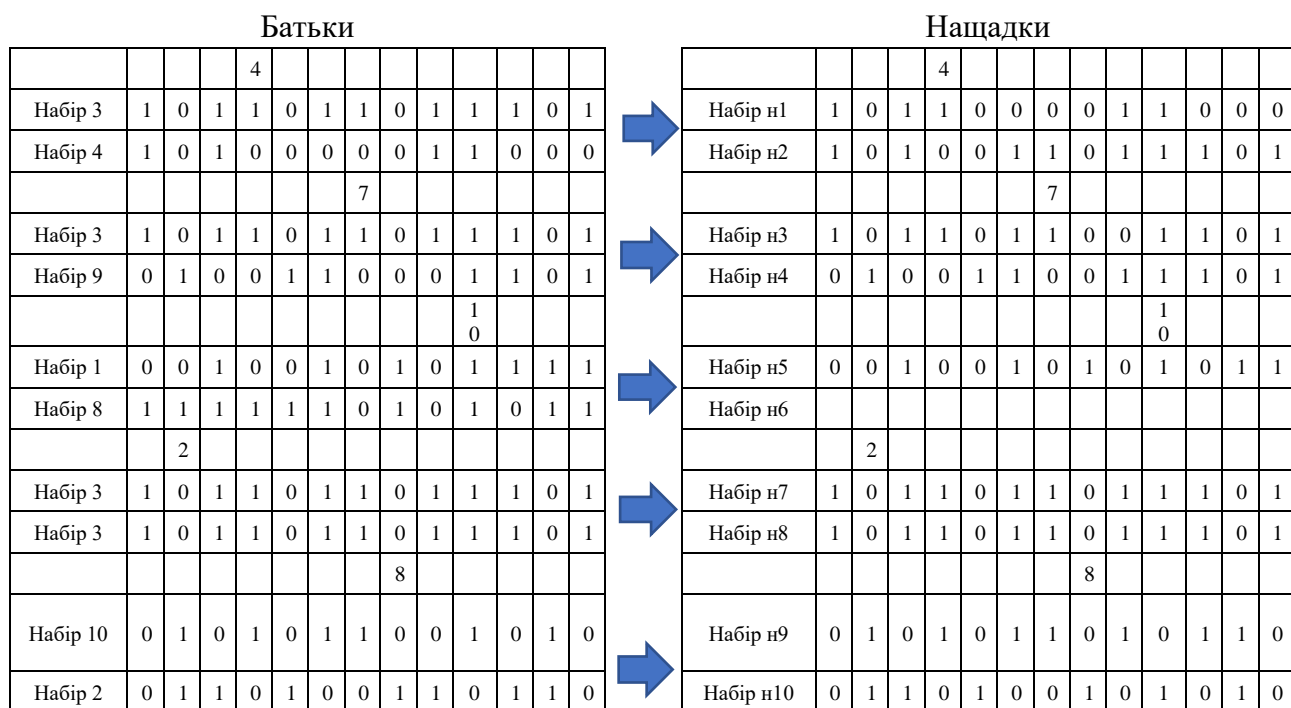


Рис. 1.3. Результат схрещування з використанням точок

Нова популяція матиме такі набори хромосом, табл. 1.4.

Таблиця 1.4

Набори хромосом нової популяції

Набори	Хромосоми													Функція пристосов.
Набір н1	1	0	1	1	0	0	0	0	1	1	0	0	0	5
Набір н2	1	0	1	0	0	1	1	0	1	1	1	0	1	8
Набір н3	1	0	1	1	0	1	1	0	0	1	1	0	1	8
Набір н4	0	1	0	0	1	1	0	0	1	1	1	0	1	7
Набір н5	0	0	1	0	0	1	0	1	0	1	0	1	1	6
Набір н6	1	1	1	1	1	1	0	1	0	1	1	1	1	11
Набір н7	1	0	1	1	0	1	1	0	1	1	1	0	1	9
Набір н8	1	0	1	1	0	1	1	0	1	1	1	0	1	9
Набір н9	0	1	0	1	0	1	1	0	1	0	1	1	0	7
Набір н10	0	1	1	0	1	0	0	1	0	1	0	1	0	6

Повторюючи декілька разів операції селекції, схрещування(застосування генетичних операторів) та створення нової популяції відповідно до алгоритму на рис. 1.1 можна створити «ідеальний набір» відповідно до початкової умови прикладу. Однак, оскільки в цьому прикладі виключені мутації та кількість наборів хромосом і генів незначна спроби створення такого «ідеального набору» подібним примітивним способом буде часто призводити до виродження або ситуації коли утвориться декілька однакових наборів (як у четвертій парі рис. 1.3) і процес просто зациклиться. Більш ймовірно досягти «ідеального набору» використовуючи підхід ґрунтований на елітизмі (штучному призначенні «вигідних» для отримання результату точок схрещування) або застосовувати мутації.

1.3. Мутації в генетичних алгоритмах

В попередньому прикладі були розглянуті основні етапи генетичного алгоритму де не використовувалися методи моделювання мутації. Для моделювання цього явища в генетичних алгоритмах можна використовувати різноманітні способи. Часто використовують генератори випадкових чисел. Один із способів моделювання мутації наведений в прикладі з наступною постановкою задачі.

Існує граф неевклідового типу (з невизначеною довжиною ребер) в якому порядок $n=5$, розмір $m=5$, рис. 1.4. Необхідно знайти лінійне розташування вершин з мінімальною довжиною ребер. Умовно розміри вершин графа ігноруються.

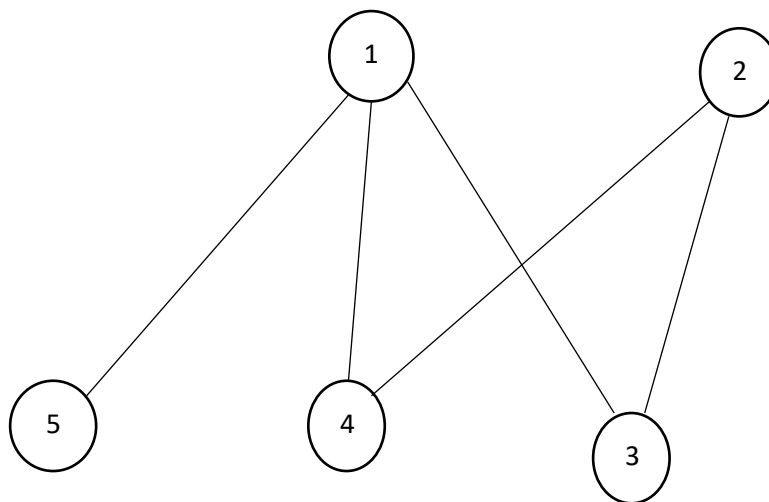


Рис. 1.4. Вихідний граф задачі

Початкове лінійне розташування вершин графа можна здавати довільно. У прикладі воно задано у відповідності з номерами вершин, рис. 1.5. На рис. 1.5 видно, що сума довжин (без врахування реальної геометричної довжини) ребер графа становить 12. Вершини графі будемо вважати хромосомою з п'ятьма генами.

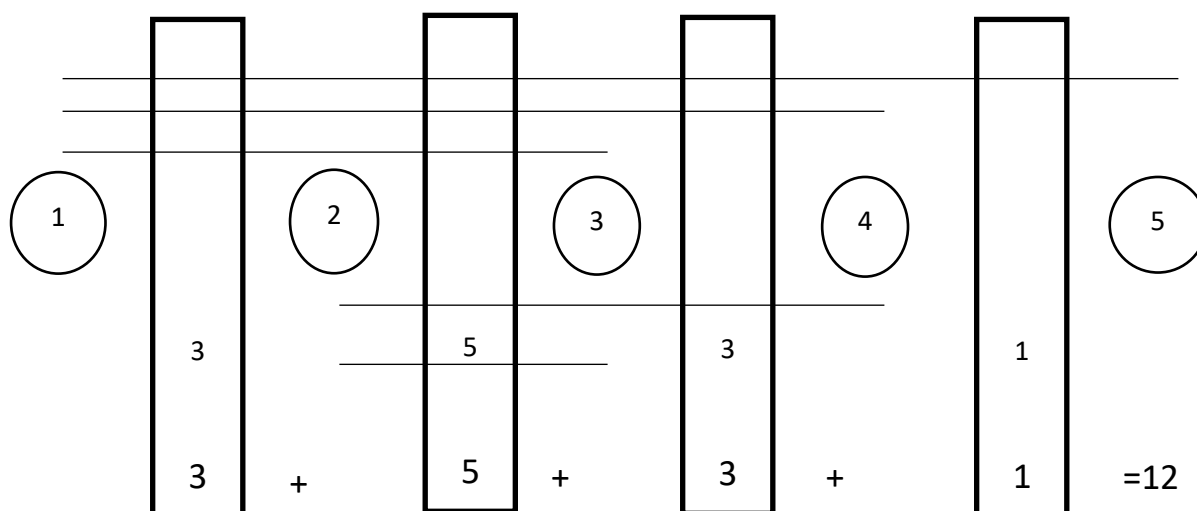


Рис. 1.5. Лінійне розташування вершин графа

Вирішення задачі.

На першому етапі задається набір хромосом певної кількості. Для демонстрації обрано 3 набори хромосом з різним розташуванням генів(вершин графа), рис. 1.6.

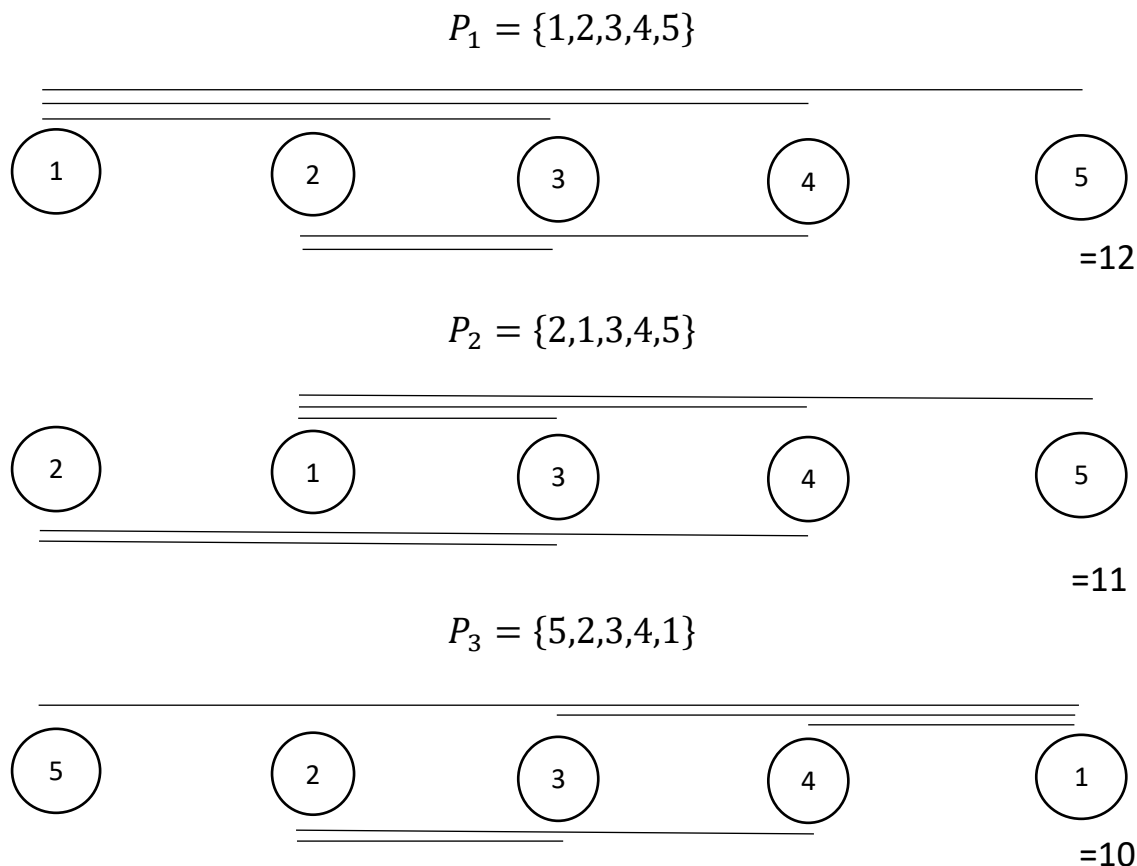


Рис. 1.6. Початковий набір хромосом з різним розташуванням генів(перше покоління)

На другому етапі проводиться вибір хромосом з найліпшим набором генів (мінімальною кількістю довжин ребер). У прикладі це хромосома з набором $P_3 = \{5,2,3,4,1\}$. Саме ця хромосома буде піддаватися мутації. Перша хромосома з сумою довжин ребер 12 видаляється з набору. На її місці буде знаходитися нова мутована хромосома. У випадках однакової кількості довжин обирається перша хромосома набору.

На третьому етапі здійснюється мутація обраної на другому етапі хромосоми $P_3 = \{5,2,3,4,1\}$. У цьому прикладі обраний спосіб мутації, який полягає у послідовній заміні певної кількості генів. Перший ген обраної

хромосоми залишається незмінним, решта генів міняють свої позиції у зворотному напрямку. Результат буде таким $P_{1M} = \{5, \underline{1,4,3,2}\}$. Мутацію наступного покоління будемо здійснювати починаючи зміни другого гену, далі мутацію починатимемо з третього гену і т.д. Набір хромосом після мутації обраної хромосоми матиме наступний вигляд, рис. 1.7.

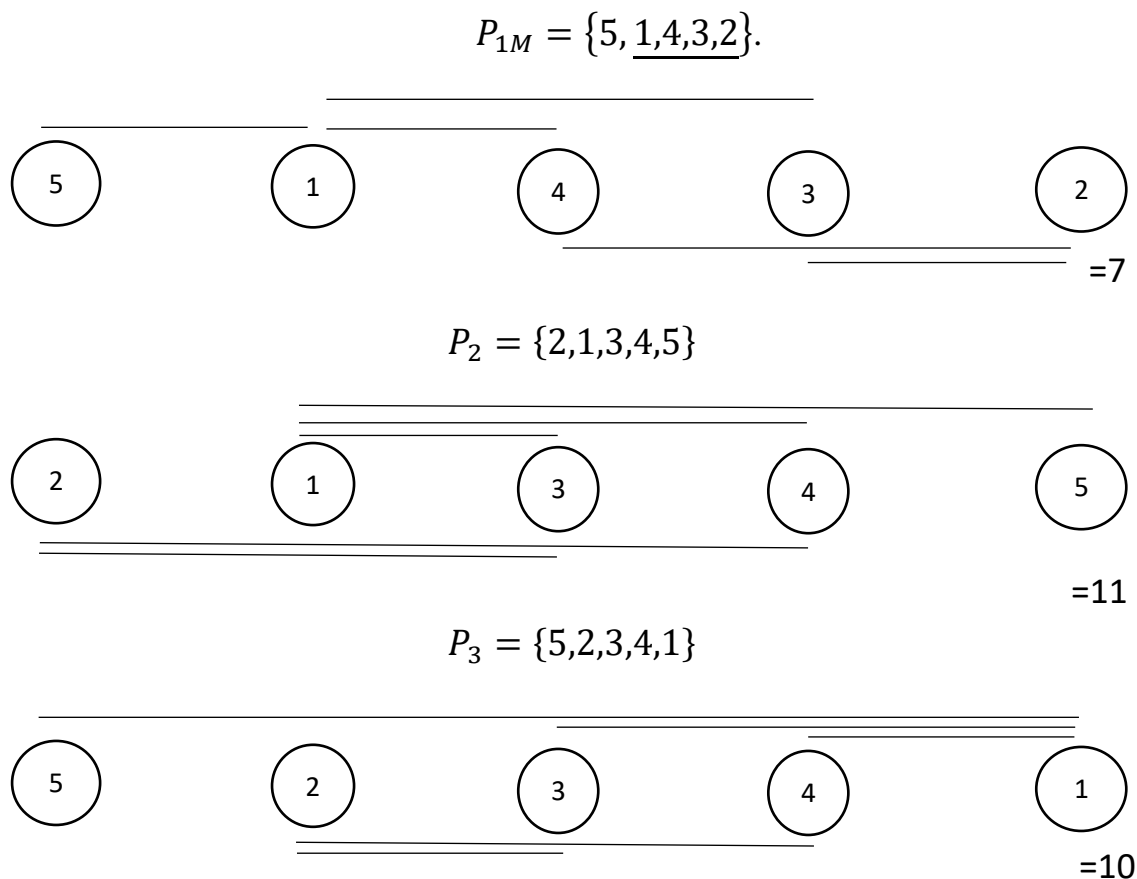


Рис. 1.7. Набір хромосом після першої мутації (друге покоління)

На четвертому етапі здійснюється повтор трьох попередніх етапів до досягнення мінімального значення суми довжин ребер.

На рис. 1.7 видно, що сума довжин ребер графа для мутованої хромосоми суттєво менша ніж в інших – 7. Ця хромосома для набору, який існує після першої мутації (нового покоління хромосом) також є переможцем, оскільки у неї сума довжини ребер мінімальна. Вона обирається для подальшої мутації. Однак, перед мутацією необхідно із набору рис 1.7 видалити хромосому з найгіршим набором. Це хромосома $P_2 = \{2,1,3,4,5\}$. У неї сума довжин ребер найбільша –

11. Замість неї буде стояти хромосома з найкращим набором попереднього покоління. Набір хромосом після другої мутації показаний на рис. 1.8.

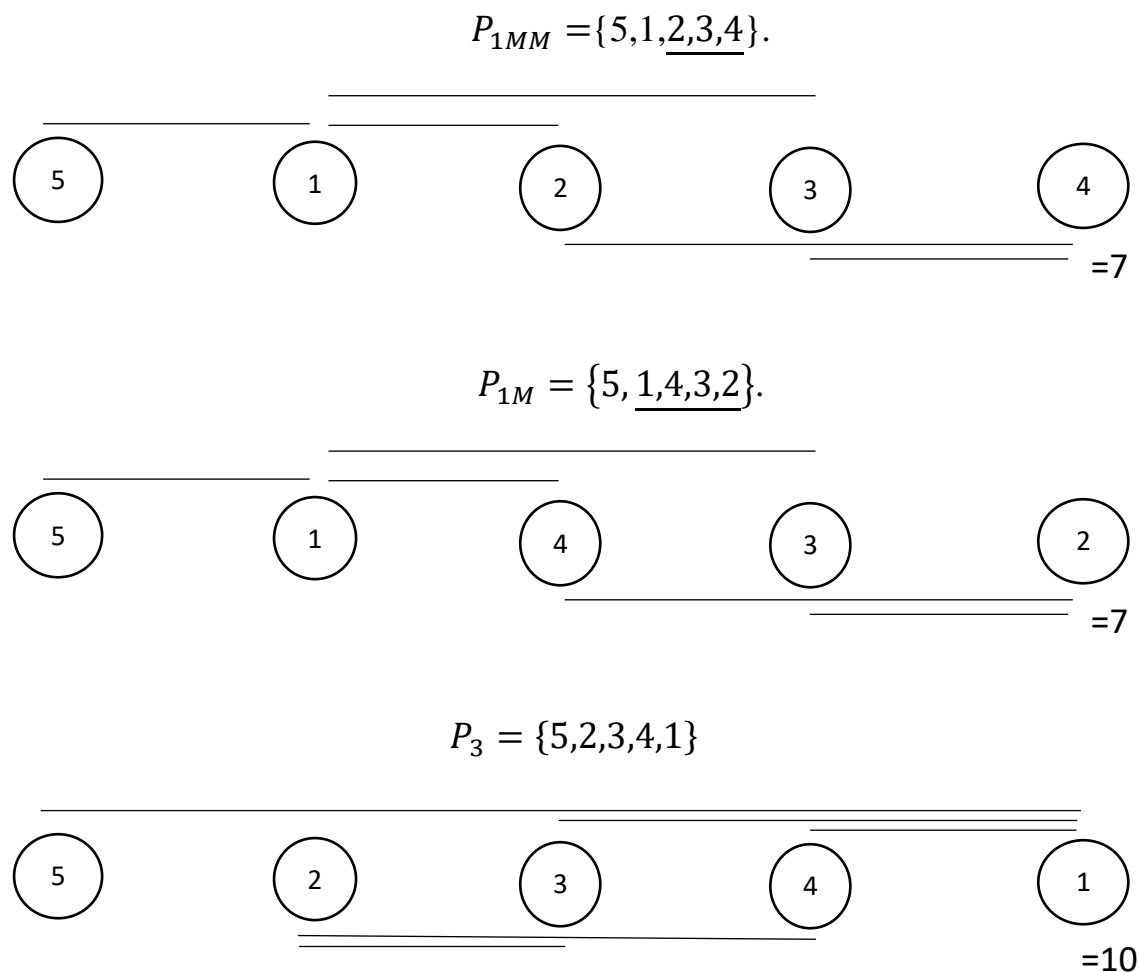


Рис. 1.8. Набір хромосом після другої мутації

Очевидно, що подальша мутація найліпшої хромосоми не приведе до покращення результату. В цьому можна переконатись самостійно, змінивши місцями вершини 3 і 4 у хромосомі $P_{1MM} = \{5, 1, \underline{2, 3, 4}\}$ і побудувавши ребра для нових наборів.

1.4. Різновиди генетичних алгоритмів

Генетичні алгоритми розділяють на два типи: алгоритми, які містять одну популяцію та алгоритми, в яких існують кілька популяцій. В свою чергу, алгоритми з однією популяцією розділяють на глобальні та просто алгоритми з

однією популяцією. Типовим представником глобального алгоритму з однією популяцією є алгоритм під назвою «господар-раб» (master-slave). В цьому алгоритмі існує одна популяція, але оцінка цільової функції здійснюється декількома процесорами. Один процесор (господар) здійснює збереження популяції, виконує операції за генетичним алгоритмом і розподіляє особин між підлеглими процесорами. Цей алгоритм називають також глобально-паралельним.

Алгоритм з однією популяцією був розглянутий в пунктах 1.2 та 1.3 і може використовуватися на масових паралельних комп'ютерах.

Алгоритми з декількома популяціями особливо популярні під назвою “розподілені генетичні алгоритми” [6]. Такі алгоритми призначені для зниження передчасної збіжності до локального оптимуму, стимуляції різноманітності і пошуку альтернативних рішень тієї ж проблеми. Вони ґрунтуються на розподілі популяції на кілька окремих підпопуляцій, кожна з яких буде оброблятися генетичним алгоритмом, незалежно від інших. Крім того, різноманітні міграції індивідів породжують обмін генетичним матеріалом серед популяцій, які зазвичай покращують точність і ефективність алгоритму. Потужність еволюційних алгоритмів посилюється із застосуванням розподілених обчислень. Тут намагаються промодельовати природну модель взаємодії окремих популяцій, для вирішення наступних завдань:

- зменшення ймовірності досягнення передчасної збіжності;
- збільшення різноманітності популяції;
- збільшення швидкості досягнення глобального оптимуму або максимально оптимальному вирішенню завдання.

До класу розподілених генетичних алгоритмів належить так звана острівна модель в якій окремі популяції розвиваються на імітованих островах, незалежно одна від одної, а генетичне змішування популяцій задається користувачем з використанням операторів міграції. Це дозволяє здійснювати відбір заданих рис окремих популяцій.

1.5. Використання генетичних алгоритмів

Використання генетичних алгоритмів сьогодні надзвичайно різноманітне. Ось тільки невеликий перелік галузей та задач в яких застосовуються генетичні алгоритми – вирішення оптимізаційних задач, криптоаналіз, вирішення фінансово-економічних задач, розробка ігрових стратегій, формування оптимальних запитів в базах даних, моделювання “штучного життя”, навчання штучних нейронних мереж, вирішення ряду технічно-конструкторських завдань та завдань біоінформатики тощо. Окремі приклади застосування генетичних алгоритмів можна знайти у відповідних джерелах [7-9].

Контрольні питання

1. Що таке генетичні алгоритми?
2. Якими є основні етапи генетичного алгоритму?
3. Як працює механізм кросоверу?
4. Що відбувається з популяцією, в якій відсутнє моделювання мутації?
5. Де застосовуються генетичні алгоритми?
6. Які існують різновиди генетичних алгоритмів?

РОЗДІЛ 2. НЕЙРОННІ МЕРЕЖІ

2.1. Штучний нейрон

Штучним нейроном називають математичну модель, будова якої ґрунтується на уявленні роботи клітини мозку живих істот – нейрона. Математична модель такого нейрона була запропонована В. Маккалохом та В. Піттсом в 1943 році разом з моделлю мережі, що складається з цих нейронів. Практичну реалізацію штучного нейрона у вигляді електричного пристрою втілював у життя вчений із США Ф. Розенблатт в 1958 році [10], назвавши її персептроном. Різниця між цими двома моделями показана в [11]. Оскільки до сьогодні робота клітин мозку досі до кінця не вивчена за основу її математичної моделі були взяті основні елементи, які вона має (рис. 2.1) – ядро клітини, відростки по яких передаються електричні нервові імпульси до ядра - дендрити, аксон – відросток, що проводить нервові імпульси до інших нервових клітин.

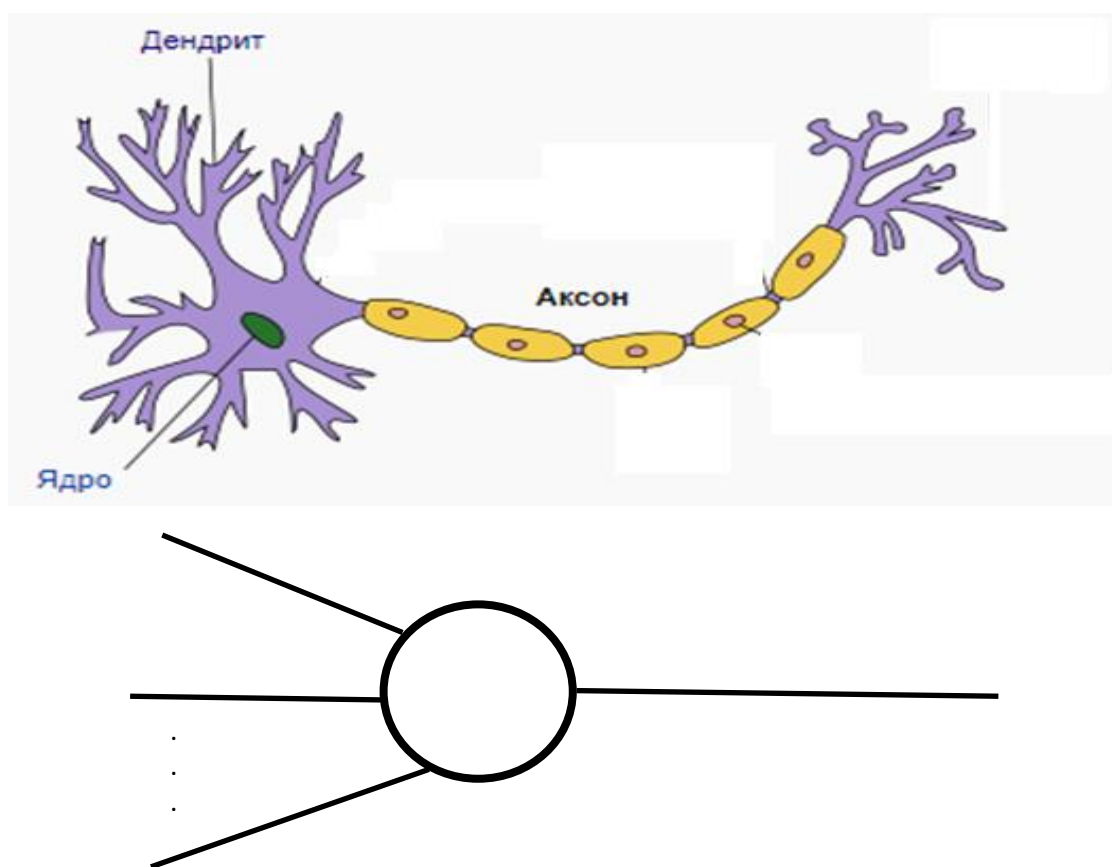


Рис. 2.1. Порівняння будови нейрона та зображення моделі штучного нейрона

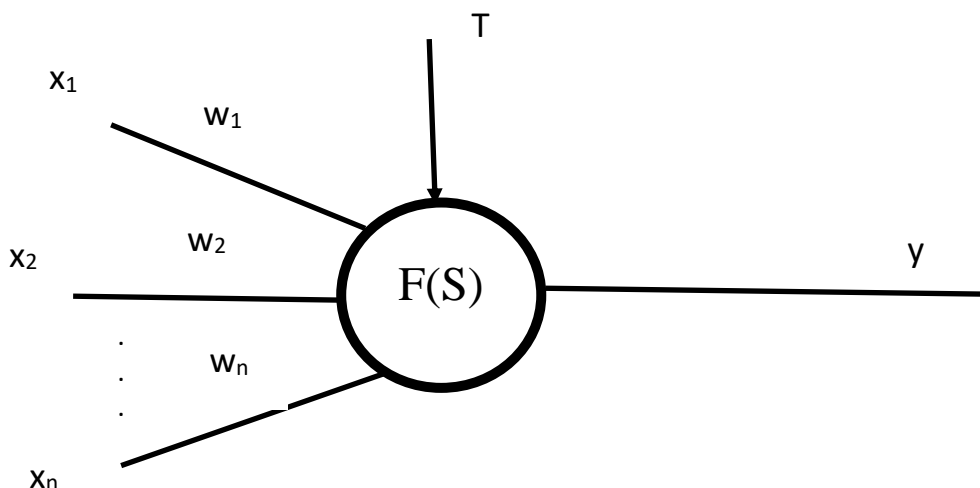


Рис. 2.2. Математичні величини які використовуються для опису штучного нейрона

На рис. 2.2 x_1, x_2, \dots, x_n – вхідні сигнали (синапси), w_1, w_2, \dots, w_n – вагові коефіцієнти, що характеризують вхідні сигнали (симпатичні ваги), $F(S)$ – математична функція активації (її ще називають передавальною функцією), T – певний заданий поріг за якого нейрон “збуджується” (є не у всіх типів), y – вихід штучного нейрона.

Зважена сума S обчислюється за наступною формулою:

$$S = x_1 \cdot w_1 + x_2 \cdot w_2 + \dots + x_n \cdot w_n \quad (2.1)$$

Функція активації $F(S)$ – визначає залежність сигналу на виході штучного нейрона від зваженої суми сигналів на його входах. Використання різних функцій активації дозволяє вносити нелінійність в роботу нейрона і в цілому нейронної мережі.

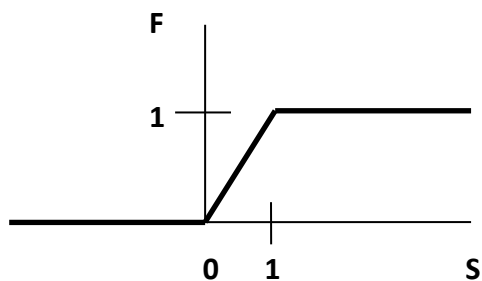
Типи функцій активації нейронів.

Лінійна передавальна функція (значення на виході нейрона лінійно пов'язане із ваговою сумою вхідних сигналів)

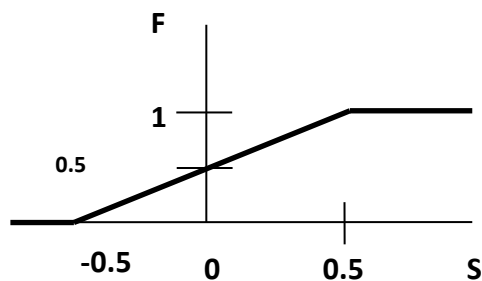
$$F(S) = tS, \quad (2.2)$$

де t – параметр функції.

$$F(S) = \begin{cases} 0 & \text{якщо } S \leq 0 \\ 1 & \text{якщо } S \geq 1 \\ S & \text{інші випадки} \end{cases} \quad (2.3)$$



a



b

Рис. 2.3. Графіки лінійної передавальної функції (b - зі зсувом по осях)

Порогова передавальна функція (функція Гевісайда).

$$F(S) = \begin{cases} 1 & \text{якщо } S \geq T \\ 0 & \text{інші випадки} \end{cases} \quad (2.4)$$

T – деяка постійна порогова величина, що моделює нелінійну передавальну характеристику

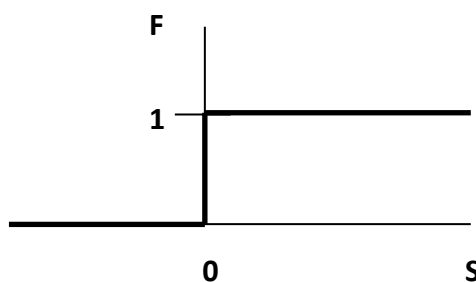


Рис. 2.4 – Графік функції Гевісайда

Сигмоїдна передавальна функція.

$$F(S) = \frac{1}{1 + \exp(-S)} \quad (2.5)$$

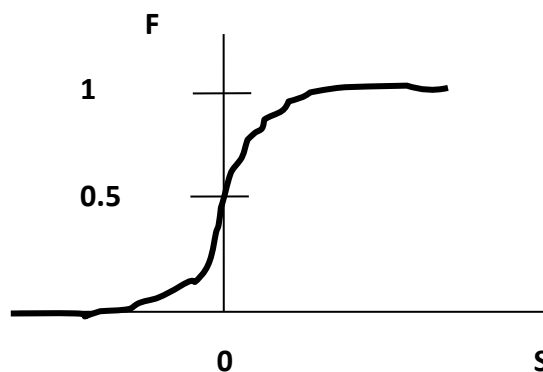


Рис. 2.5. Графік сигмоїдної функції

Сигмоїдна функція диференційована на всій осі абсцис та має властивість посилювати слабкі сигнали краще, ніж великі, і запобігає насиченню від великих сигналів, оскільки вони відповідають областям аргументів, де сигмоїд має пологий нахил. Ці властивості роблять сигмоїдну функцію широкоживаною в нейронних мережах.

Крім перерахованих функцій існує багато інших які використовуються для побудови нейронних мереж. Наприклад, логістична, експоненціальна, гіперболічний тангенс та інші. Ці передавальні функції широко описані в ряді джерел. Більш детально про типи передавальних функцій активації штучного нейрона можна дізнатися в [12]. Інформацію про них можна отримати також з Інтернету.

Оскільки штучні нейрони подібні між собою за будовою їх класифікація здійснюється, виходячи із розташування у мережі. Вхідні нейрони – не здійснюють обчислень і передають вхідні сигнали, проміжні нейрони – виконують обчислювальні операції, вихідні нейрони – знаходяться на виході мережі і також виконують операції обчислення. Вихідний сигнал штучного нейрона у може подаватися на декілька нейронів мережі. Структурна схема штучного нейрона показана на рис. 2.6.

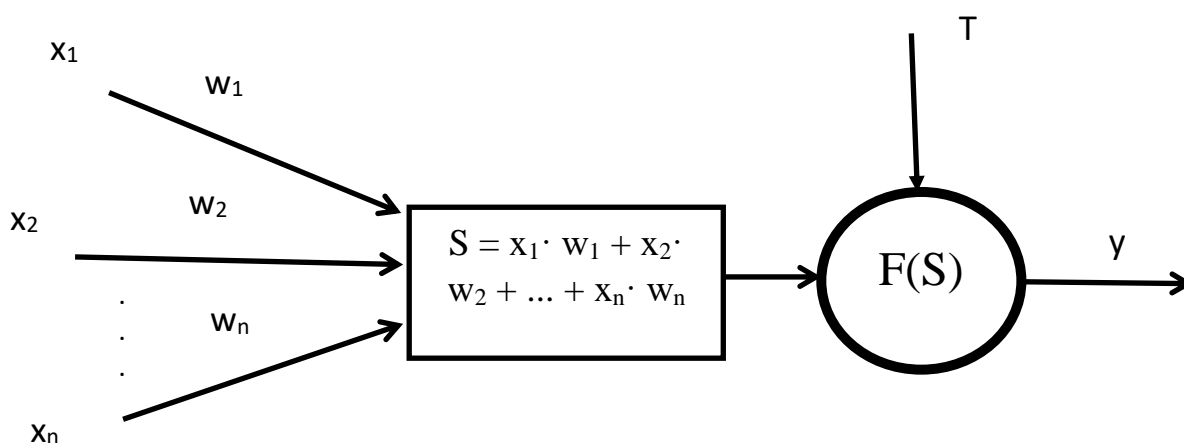


Рис. 2.6. Структурна схема штучного нейрона

2.2. Моделювання логічних функцій за допомогою штучних нейронів

Штучний нейрон може бути основою для моделювання логічних функцій – І, АБО, НІ, Виключне АБО (XOR) та інших. Таке моделювання дозволяє пояснити сутність роботи штучних нейронів або мереж. Для моделювання логічних функцій за допомогою штучних нейронів необхідно:

- мати таблицю істинності для певної функції;
- знати кількість синапсів функції;
- знати значення ваг функції;
- знати будову нейрона або мережі або нейрона за допомогою якого реалізується та чи інша функція.

В якості прикладу моделювання наведена модель логічної функції АБО-НЕ (стрілка Пірса). Таблиця істинності для цієї функції має вигляд, табл. 2.1.

Таблиця 2.1

Таблиця істинності для функції АБО-НЕ (стрілка Пірса)

x_1	x_2	$F(S)$
0	0	1
0	1	0
1	0	0
1	1	0

Штучний нейрон для цієї функції матиме вигляд, рис. 2.7.

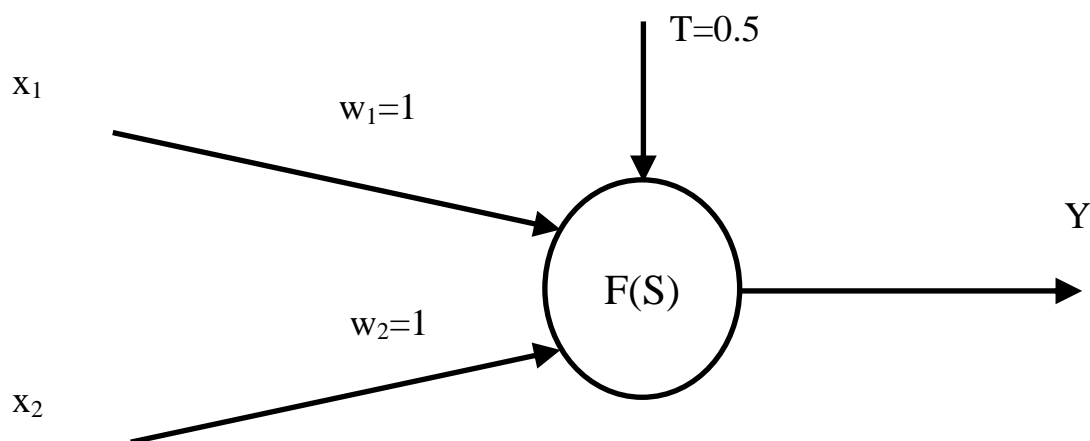


Рис. 2.7. Штучний нейрон для моделювання функції АБО-НЕ

$$F(S) = \begin{cases} 0 & \text{якщо } S \geq 0,5 \\ 1 & \text{якщо } S < 0,5 \end{cases} \quad (2.6)$$

Для $x_1=0$ та $x_2=0$

$$S = w_1 * x_1 + w_2 * x_2 = 1 * 0 + 1 * 0 = 0.$$

$Y = F(S) = 1$ (бо відповідно до (2.6) $S < 0,5$).

Для $x_1=0$ та $x_2=1$

$$S = w_1 * x_1 + w_2 * x_2 = 1 * 0 + 1 * 1 = 1.$$

$Y = F(S) = 0$ (бо відповідно до (2.6) $S \geq 0,5$).

Для $x_1=1$ та $x_2=0$

$$S = w_1 * x_1 + w_2 * x_2 = 1 * 1 + 1 * 0 = 1.$$

$Y = F(S) = 0$ (бо відповідно до (2.6) $S \geq 0,5$).

Для $x_1=1$ та $x_2=1$

$$S = w_1 * x_1 + w_2 * x_2 = 1 * 1 + 1 * 1 = 2.$$

$Y = F(S) = 0$ (бо відповідно до (2.6) $S \geq 0,5$).

Моделювання більш складних логічних функцій здійснюється з використанням комп'ютерної мережі, наприклад мережа для моделювання функції XOR виглядатиме наступним чином, рис. 2.8 [13].

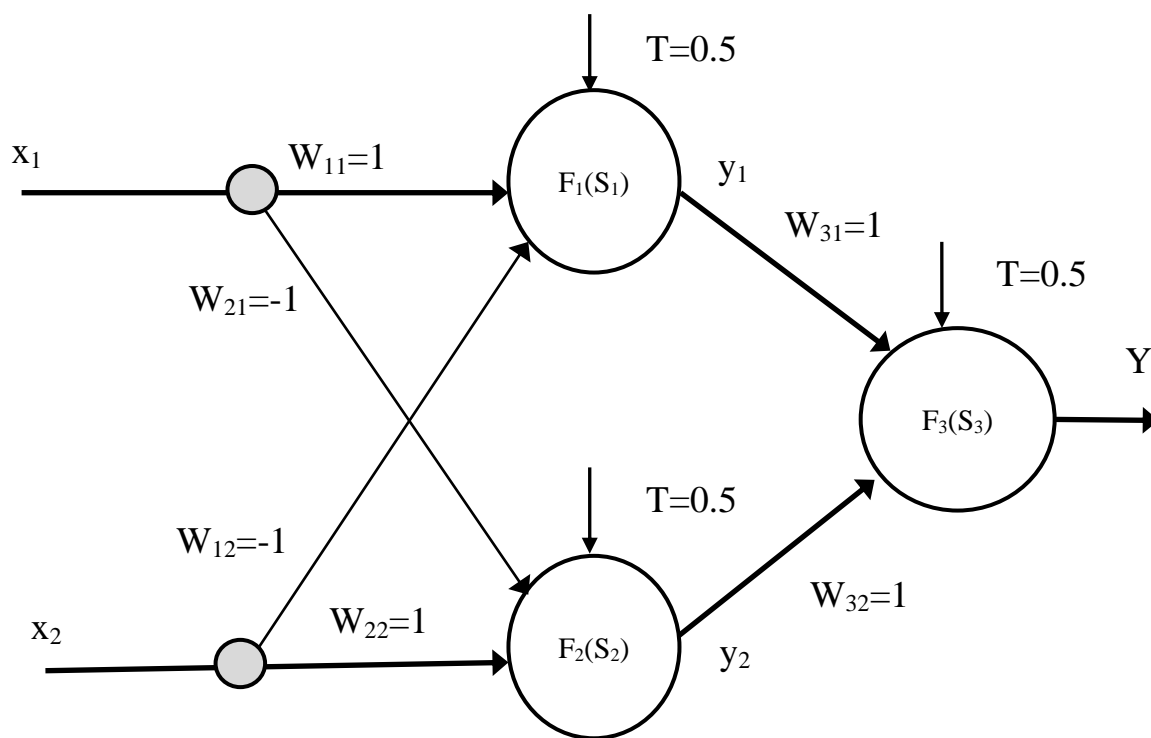


Рис. 2.8. Штучний нейрон для моделювання функції XOR

2.3. Навчання штучного нейрона

Під навчанням штучного нейрона розуміють налаштування його ваг з метою отримання бажаного вихідного сигналу при певній комбінації вхідних сигналів. Навчання штучного нейрона (а також і штучної нейронної мережі) може здійснюватися шляхом впливу людини на процес навчання («навчання з учителем») та без такого впливу («навчання без учителя»).

За допомогою навчання можна пояснити принцип на якому ґрунтується розпізнавання образів. Метод навчання штучного нейрона був запропонований Ф. Розенблаттом під назвою метод корекції помилки. Цей метод є методом «навчання з учителем» і передбачає незмінність ваги зв'язку до тих пір, поки поточна реакція штучного нейрона залишається правильною. При появі неправильної реакції вага змінюється на одиницю, а знак (+/-) визначається протилежним від знаку помилки. Існує декілька модифікацій цього методу. Для детального пояснення його роботи використаємо наступний приклад:

Існує штучний нейрон з дев'ятьма входами який необхідно навчити здійснювати розпізнавання двох символів, рис. 2.9.

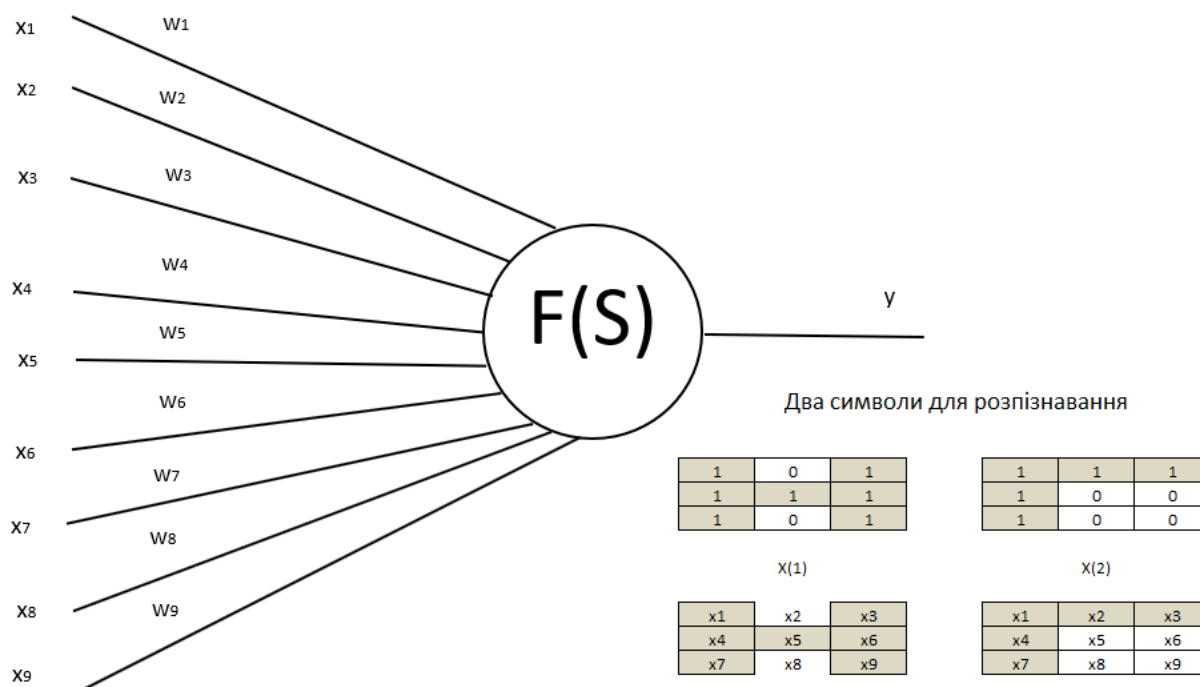


Рис. 2.9. Зображення штучного нейрона та символів для розпізнавання

Необхідно здійснити налаштування вагових коефіцієнтів $\{w_1, w_2 \dots w_9\}$ заданого нейрона таким чином, щоб на виході y утворювалися 1 для першого символу та 0 для другого.

Коригування ваг здійснюється за формулами[14]:

$$w_{ij}^{(l)} = w_{ij}^{(l-1)} + \Delta_j \cdot x_i, \text{ де } \Delta_j = \begin{cases} 0, & \text{якщо } y_j = t_j \\ 1, & \text{якщо } y_j = 0, t_j = 1. \\ -1, & \text{якщо } y_j = 1, t_j = 0., \end{cases} \quad (2.7)$$

де

$w_{ij}^{(l)}$ – значення ваги на l -му кроці навчання(коригування);

$w_{ij}^{(l-1)}$ – значення ваги попередньому $(l-1)$ кроці навчання (коригування);

Δ_j – значення коригування для j -го нейрона (в прикладі $j=1$, оскільки нейрон один);

x_i – елемент множини для навчання нейрона;

y_j – вихідний сигнал нейрона (0 або 1).

У якості функції $F(S)$ використовується бінарна функція

$$F(S) = y = \begin{cases} 1, & \text{якщо } s \geq 0 \\ 0, & \text{якщо } s < 0 \end{cases} \quad (2.8)$$

Послідовність здійснення навчання.

1 крок

Задається множина для навчання

$$M = \{X(1) = (1,0,1,1,1,1,0,1), \quad t(1) = 1, \quad X(2) = (1,1,1,1,0,0,1,0,0), \quad t(2) = 0\}.$$

Всі вагові коефіцієнти дорівнюють 0 ($w_1, w_2, \dots, x_9 = 0$)

2 крок

Здійснюється коригування вагових коефіцієнтів. На вхід нейрона послідовно подаються вектори $X^{(1)}$ та $X^{(2)}$, обчислюється вихідний сигнал у який порівнюється з заданим значенням $t^{(1)}$ або $t^{(2)}$ і у випадку неспівпадіння здійснюється коригування за формулою (2.7).

Відповідно до встановлених значень при подаванні першого вектора $X(1)$ на вхід нейрона значення всіх ваг дорівнюють 0, а $y_j = t_j = 1$. Отже, відповідно до (2.7) $\Delta_j = 0$. Подаючи другий вектор $X(2)$ на вхід нейрона значення $y_j = 1$, оскільки попередній вектор при обчисленні дав значення $s = x_1 \cdot 0 + x_2 \cdot 0 + \dots + x_9 \cdot 0$, а при $s=0, y=1$ відповідно до (2.8). В такому разі $\Delta_j = -1$ бо $y_j = 1, t_j = 0$. Використовуючи значення $\Delta_j = -1$ обчислимо ваги другого вектора за формулою (2.7). Наприклад, $w_1 = w_1 + \Delta_j \cdot x_1 = 0 + (-1) \cdot 1 = -1$. Значення обчислених ваг разом із значеннями y, t та s наведені в табл. 2.2.

Таблиця 2.2

Значення обчислених величин

delta=0		delta=-1		delta=1		delta=-1		delta=1		delta=0	
t=1		t=0		t=1		t=0		t=1		t=0	
s=0		s=-5		s=3		s=-2		s=-2		s=3	
y=1		y=1(бо s=0)		y=0(бо s=-5)		y=1(бо s=3)		y=0(бо s=-2)		y=0(бо s=-2)	
X(1)	W	X(2)	W	X(1)	W	X(2)	W	X(1)	W	X(2)	W
1	0	1	-1	1	0	1	-1	1	0	1	0
0	0	1	-1	0	-1	1	-1	0	-1	1	-1
1	0	1	-1	1	0	1	0	1	0	1	0
1	0	1	-1	1	0	1	0	1	0	1	0
1	0	0	0	1	1	0	1	1	1	0	1
1	0	0	0	1	1	0	1	1	1	0	1
1	0	1	-1	1	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	1	0	1	1	1	0	1

3 крок

В таблиці дві останні виділені колонки показують, що ваги перестали змінюватися. Отже навчання нейрона закінчилося. Отриманий результат дозволяє розпізнавати два образи, використовуючи 1 для першого та 0 для другого.

Отже, за три кроки здійснено навчання штучного нейрона з методом корекції помилки. Для перевірки отриманого результату «зіпсуємо» вхідні символи і подамо їх на вхід нейрона:

$$X(1d) = \{1,0,1,1,1,1,1,0,0\},$$

1	0	1
1	1	1
1	0	0

$$X(2d) = \{0,1,1,1,0,0,1,0,0\}$$

0	1	1
1	0	0
1	0	0

За результатами обчислень маємо:

$$s(1d) = X(1d) \cdot W = 1 \cdot 0 + 0 \cdot (-1) + 1 \cdot 0 +$$

$$1 \cdot 0 + 1 \cdot 1 + 1 \cdot 1 + 1 \cdot 0 + 0 \cdot 0 + 0 \cdot 0) = 2.$$

$$s(2d) = X(2d) \cdot W = 0 \cdot 0 + 1 \cdot (-1) + 1 \cdot 0 +$$

$$1 \cdot 0 + 0 \cdot 1 + 0 \cdot 1 + 1 \cdot 0 + 0 \cdot 0 + 0 \cdot 1) = -1.$$

Відповідно до (2.8) $s(1d) \geq 0 \rightarrow y = 1$. Оскільки $y = 1$, перший символ розпізнаний як Н. $s(2d) < 0 \rightarrow y = 0$. Оскільки $y = 0$, другий символ розпізнаний як Г. Не дивлячись на задане викривлення нейрон коректно «розпізнав» символи.

Розглянутий метод корекції помилки має ряд недоліків. По-перше, використовуючи один штучний нейрон можна розпізнати лише два символи. По-друге, точність розпізнавання є низькою, оскільки нейрон може розпізнавати символи або образи, які незначним чином відрізняються від еталонних. Цю тезу легко перевірити, ще більше спотворивши вхідні символи.

2.4. Штучні нейронні мережі

Штучні нейронні мережі – це математичні програмна моделі, які складаються зі штучних нейронів, пов'язаних між собою та мають здатність до навчання. Під навчанням нейронної мережі розуміють здатність до підвищення продуктивності шляхом зміни внутрішніх параметрів за певним алгоритмом. Найпростіша штучна нейронна мережа складається з одного штучного нейрона. Штучні нейронні мережі бувають різні за будовою, і відповідно називаються по-різному. Досить простий варіант штучної нейронної мережі називається мережею Гопфілда (Хопфілда).

2.5. Мережа Гопфілда

Будова цієї мережі показана на 2.10. На цьому рисунку видно, що нейронна мережа Гопфілда складається тільки з одного шару нейронів і має специфічну форму зворотних зв'язків де виходи кожного нейрона з'єднані з входами решти нейронів (окрім входу власного нейрона). Тобто, $w_{11} = w_{22} = w_{33}$. Ваги зв'язків в такій мережі можна подати у вигляді матриці:

$$W = \begin{bmatrix} 0 & w_{12} & \dots & w_{1n} \\ w_{21} & 0 & \dots & w_{2n} \\ \vdots & \vdots & 0 & \vdots \\ w_{n1} & w_{n2} & \dots & 0 \end{bmatrix} \quad (2.9)$$

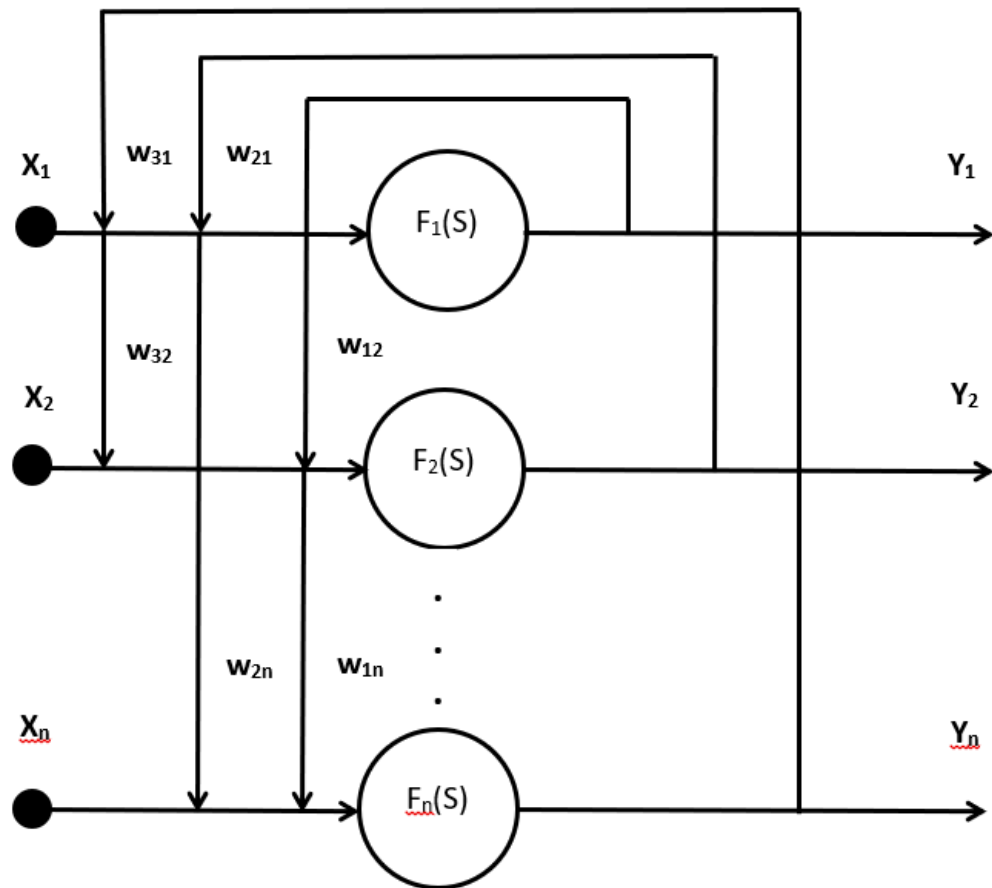


Рис. 2.10. Умовне зображення мережі Гопфілда

Кожен нейрон мережі Гопфілда може перебувати у двох станах, тобто

$$Y_i = \begin{cases} 1 \\ -1 \end{cases} \quad (2.10)$$

Через це нейрони мережі Гопфілда часто називають спінами. Оскільки мережа містить n штучних нейронів за її допомогою можна розпізнати 2^n образів (символів або зображень). Взаємодія нейронів у цій мережі описується виразом:

$$E = \frac{1}{2} \sum_{i,j=1}^n w_{ij} x_i x_j \quad (2.11)$$

Для демонстрації розпізнавання за допомогою мережі Гопфілда наведено наступний приклад який демонструє метод «навчання без учителя».

Існує три образи(символи – Н,Г,О), які подаються на вхід мережі та один «зіпсований» образ. Необхідно дізнатися до якого із трьох символів належить «зіпсований». Три символи та «зіпсований» символ показані на рис. 2.12.

1	-1	1
1	1	1
1	-1	1

1	1	1
1	-1	-1
1	-1	-1

1	1	1
1	-1	1
1	1	1

1	1	1
1	1	1
1	1	1

Рис. 2.12. Вихідні символи та «зіпсований» символ

На рис. 2.12 в незафарбованих клітинках замість нулів стоять -1. Це зроблено виключно для зручності і не має принципового значення, просто якщо стани нейронів відповідно до (2.11) знаходяться в межах (1;-1) обчислення проводити зручніше. Також, виходячи із рис 2.12 для розпізнавання «зіпсованого» символу необхідно використати мережу із 9 штучних нейронів. Вхідні вектори для всіх символів записані наступним чином – $X(1) = \{1, -1, 1, 1, 1, 1, 1, -1, 1\}$, $X(2) = \{1, 1, 1, 1, -1, -1, 1, -1, -1\}$, $X(3) = \{1, 1, 1, 1, -1, 1, 1, 1, 1\}$. Вихідний вектор або вектор «зіпсованого» нейрону, відповідно – $Y = \{1, 1, 1, 1, 1, 1, 1, 1, 1\}$.

Для того, щоб сформувати матрицю ваг для зв'язків, вигляду (2.9) перемножимо вхідні вектори самі на себе (транспонований вектор на вектор) і отримані три матриці додамо. Ці дії нескладні і їх можна виконати за допомогою комп'ютерних програм, наприклад за допомогою MS Excel. Приклад множення першого вхідного вектора (результат множення двох інших векторів не наведений, його можна виконати самостійно) і сума трьох матриць наведені нижче:

$$\begin{bmatrix} 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} \times [1, -1, 1, 1, 1, 1, 1, -1, 1] = \begin{bmatrix} 1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 \\ -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\ 1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 \\ -1 & 1 & -1 & -1 & -1 & -1 & -1 & 1 & -1 \\ 1 & -1 & 1 & 1 & 1 & 1 & 1 & -1 & 1 \end{bmatrix} \quad (2.12)$$

В результуючій матриці для того, щоб привести її до виду (2.9) діагональ заповнена нулями, хоча ця дія не є обов'язковою.

$$W = \begin{bmatrix} 0 & 1 & 3 & 3 & -1 & 1 & 3 & -1 & 1 \\ 1 & 0 & 1 & 1 & -3 & -1 & 1 & 1 & -1 \\ 3 & 1 & 0 & 3 & -1 & 1 & 3 & -1 & 1 \\ 3 & 1 & 3 & 0 & -1 & 1 & 3 & -1 & 1 \\ -1 & -3 & -1 & -1 & 0 & 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & 1 & 1 & 0 & 1 & 1 & 3 \\ 3 & 1 & 3 & 3 & -1 & 1 & 0 & -1 & 1 \\ -1 & 1 & -1 & -1 & -1 & 1 & -1 & 0 & 1 \\ 1 & -1 & 1 & 1 & 1 & 3 & 1 & 1 & 0 \end{bmatrix} \quad (2.13)$$

Множення цієї матриці на транспонований вектор Y дає вектор, елементи якого обробляються бінарною функцією (2.10) кожного з дев'яти нейронів мережі в якій замість 0 ставимо -1.

$$W \times Y = \begin{bmatrix} 10 \\ 0 \\ 10 \\ 10 \\ -6 \\ 8 \\ 10 \\ -2 \\ 8 \end{bmatrix} \begin{matrix} F_1(S) \\ \rightarrow F_2(S) \rightarrow \\ \rightarrow F_3(S) \rightarrow \\ \rightarrow F_4(S) \rightarrow \\ \rightarrow F_5(S) \rightarrow \\ \rightarrow F_6(S) \rightarrow \\ \rightarrow F_7(S) \rightarrow \\ \rightarrow F_8(S) \rightarrow \\ F_9(S) \end{matrix} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ 1 \\ 1 \\ -1 \\ 1 \end{bmatrix} \quad (2.14)$$

Отриманий результат не схожий на жоден із вхідних векторів ($X(1)$, $X(2)$, $X(3)$). Тому операція множення $W \times Y$ повторюється ще раз, але тепер в ній в

якості вектора Y використовується отриманий результат, а матриця W залишається незмінною.

$$W \times Y = \begin{array}{c} F_1(S) \\ \left[\begin{array}{c} 14 \\ 4 \\ 14 \\ 14 \\ -4 \\ 4 \\ 14 \\ 0 \\ 4 \end{array} \right] \rightarrow \begin{array}{c} F_2(S) \\ F_3(S) \\ F_4(S) \\ F_5(S) \\ F_6(S) \\ F_7(S) \\ F_8(S) \end{array} \rightarrow \left[\begin{array}{c} 1 \\ 1 \\ 1 \\ 1 \\ -1 \\ 1 \\ 1 \\ 1 \\ 1 \end{array} \right] \\ F_9(S) \end{array} \quad (2.15)$$

Отриманий результат повністю співпадає з $X(3)$. Якщо і надалі повторювати обчислення він не зміниться. Отже, «зіпсований» символ розпізнаний мережею як 0, що є коректним результатом.

Наведений приклад демонструє так званий синхронний режим роботи мережі Гопфілда, коли розпізнається цілий образ. При асинхронному режимі роботи образ розпізнається фрагментарно, частинами. Такий режим роботи вважається більш придатним для реалізації на практиці.

Перевагою мережі Гопфілда є швидкість її роботи. Для розпізнавання складних образів можна використовувати тисячі нейронів. При цьому швидкість обчислення буде незначною. Це дає змогу використовувати цю мережу при вирішенні ряду оптимізаційних задач. Розширенням мережі Гопфілда є мережа Хемінга[15].

2.6. Багатошарові нейронні мережі

В багатошарових нейронних мережах окрім вхідного та вихідного шарів існує ще декілька (один або більше) проміжних шарів(їх ще називають прихованими). Нескладний варіант багатошарової штучної нейронної мережі показаний на рис. 2.13 [16].

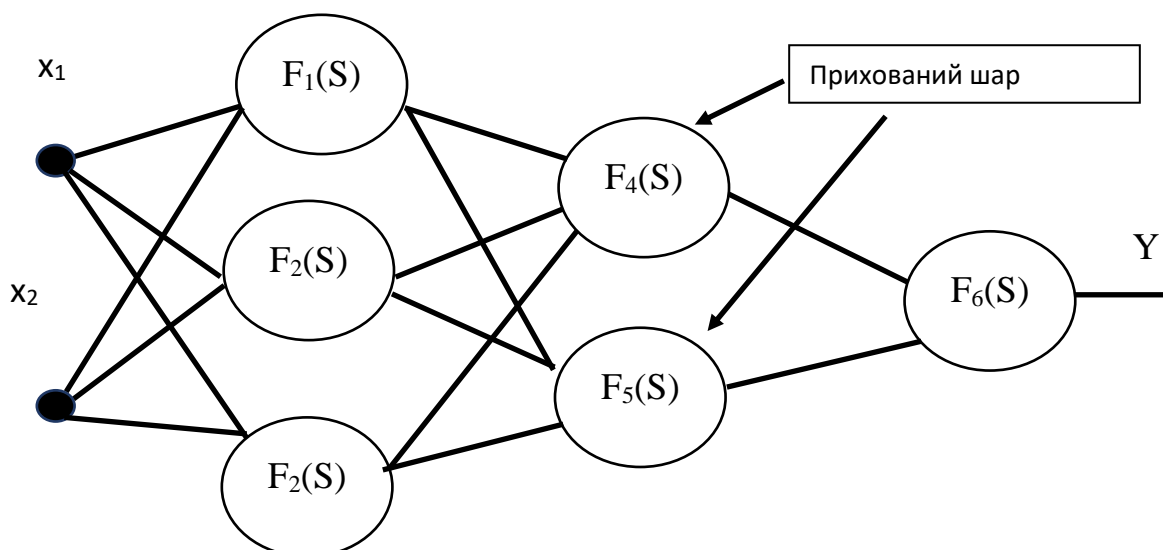


Рис. 2.13. Варіант тришарової штучної нейронної мережі

На рис. 2.13 x_1 та x_2 є синапсами зразу для трьох нейронів. Ця мережа складається з трьох шарів. У першому шарі три нейрони, у другому – два, і у третьому – один. Кількість шарів та кількість нейронів у кожному шарі може бути різною. Перший шар називають вхідним, останній – вихідним, а середній шар або шари – прихованими. Також виходи нейронів вхідного шару є синаптичними вагами для нейронів наступного шару. Не обов'язково кожен вихід нейрона певного шару зв'язаний з усіма нейронами наступного шару. Існує багато різних варіантів зв'язків між нейронами штучної мережі (у тому числі і зворотних). На прикладі мережі рис. 2.13 досить зручно пояснювати навчання з використанням методу зворотного поширення помилки.

2.7. Метод зворотного поширення помилки

Метод зворотного поширення помилки (backpropagation) є одним із методів навчання багатошарових нейронних мереж. Теоретичною основою методу зворотного поширення помилки є оптимізація в результаті якої визначається настільки добре мережа вирішує поставлені перед нею завдання. Для обчислень використовують градієнтний спуск. При цьому функцію оцінки роботи мережі звичайно записують наступним чином:

$$E(\{w_{i,j}\}) = \frac{1}{2} \sum_{k \in Outputs} (t_k - o_k)^2, \quad (2.16)$$

де

i, j – індекси вузлів, з'єднаних між собою штучних нейронів мережі;

$w_{i,j}$ – відповідні ваги зв'язків між штучними нейронами;

o_k – вихід k -го вузла (нейрона або нейронів вихідного шару);

t_k – правильна відповідь мережі (при $k \in Outputs$).

На практиці метод зворотного поширення помилки полягає у корекції значень вектору $Y = \{y_1, y_2, \dots, y_n\}$ на виході нейронної мережі відносно заданих величин $Z = \{z_1, z_2, \dots, z_n\}$, де n – кількість штучних нейронів вихідного шару нейронної мережі. На рис. 2.14 та рис. 2.15 показані окремі етапи методу. Для спрощення показана мережа, що складається з трьох нейронів, на вхід якої подаються два значення S_1 та S_2 . У якості передавальних функцій всіх нейронів мережі обрані сигмоїдні(сигмоїдальні) функції $F(S) = \frac{1}{1+e^{-s}}$.

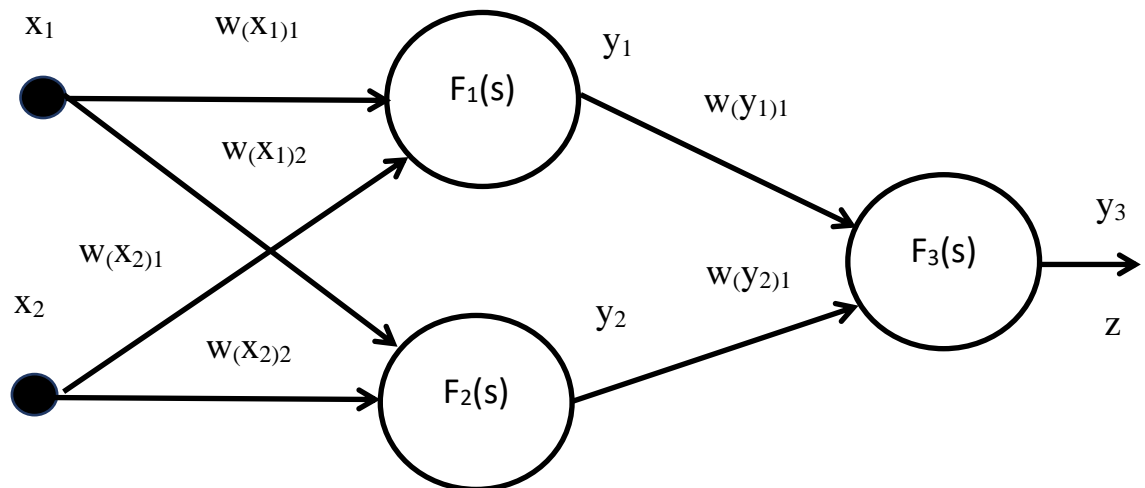


Рис. 2.14. Штучна нейронна мережа з позначеннями вхідних сигналів, ваг зв'язків, значеннями виходів нейронів та заданою величиною z

Послідовність обчислень в методі здійснюється наступним чином.

На першому етапі (рис. 2.15) обчислюється значення y_3 . Формули для обчислень наступні:

$$\begin{aligned} y_1 &= x_1 \cdot w_{(x_1)1} + x_2 \cdot w_{(x_2)1}; \\ y_2 &= x_1 \cdot w_{(x_1)2} + x_2 \cdot w_{(x_2)2}; \\ y_3 &= y_1 \cdot w_{(y_1)1} + y_2 \cdot w_{(y_2)1} \end{aligned} \quad (2.17)$$

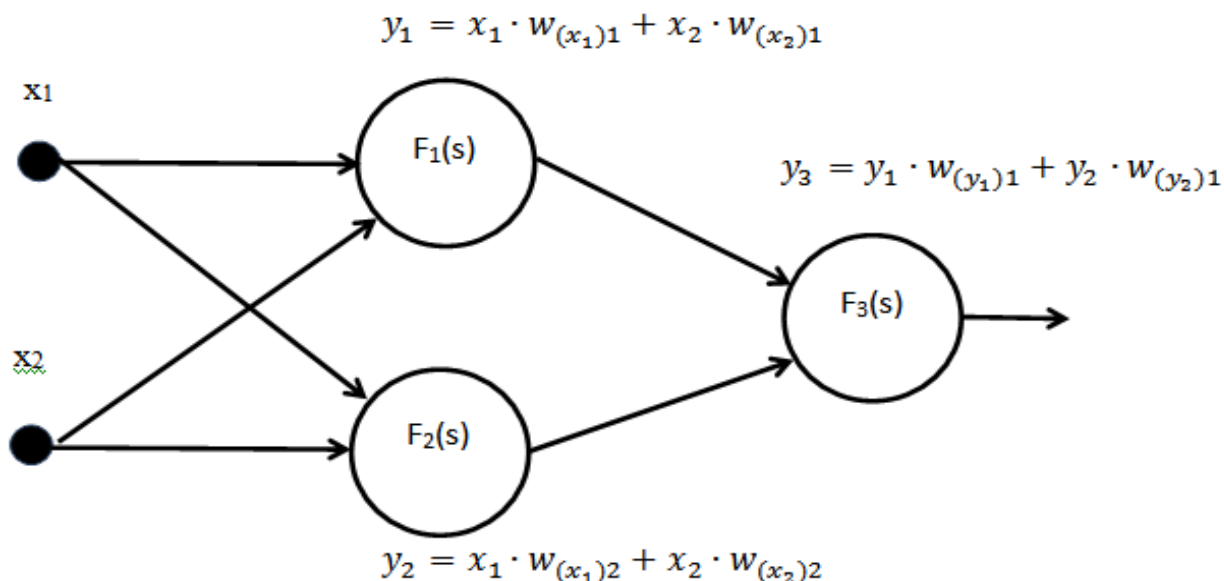


Рис. 2.15. Перший етап методу зворотного поширення помилки

На другому етапі (рис. 2.16) обчислюється значення різниці δ між заданим значенням z та розрахованим значенням y_3 та обчислюються значення різниць для всіх нейронів штучної мережі:

$$\begin{aligned} \delta &= y_3 - z; \\ \delta_1 &= w_{(y_1)1} \cdot \delta; \\ \delta_2 &= w_{(y_2)1} \cdot \delta. \end{aligned} \quad (2.18)$$

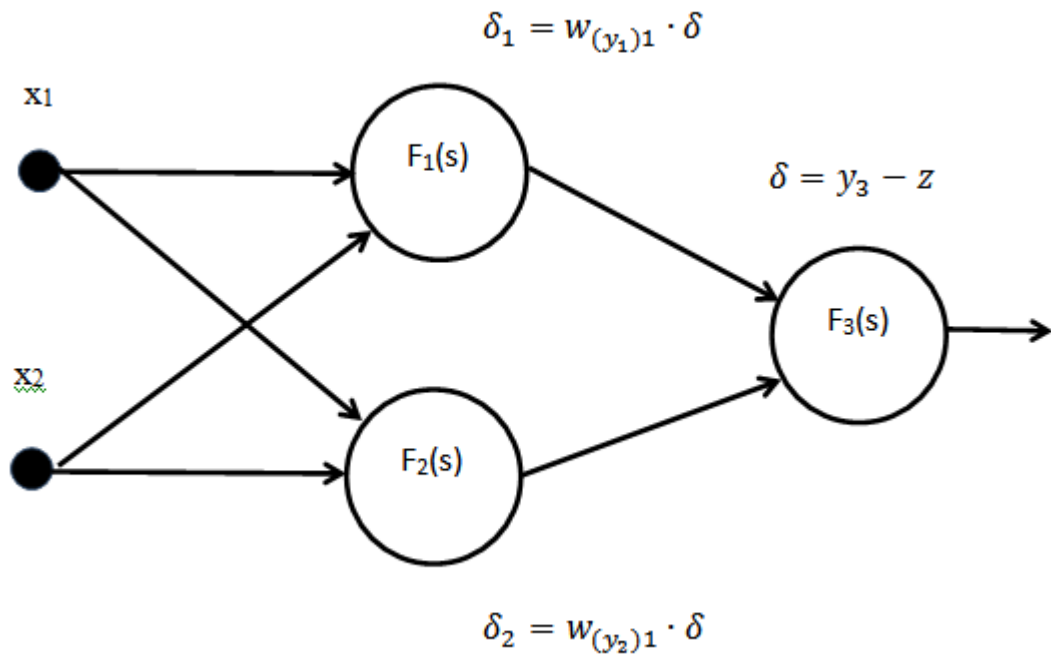


Рис. 2.16. Другий етап методу зворотного поширення помилки

На третьому етапі (рис. 2.17) розраховуються значення скоригованих ваг зв'язків штучної мережі з використанням похідних функцій активації нейронів. Похідною від сигмоїдної функції, яка використовується в розглядуваній мережі є $F'(S) = F(S) \cdot (F(S) - 1)$. Наприклад, для $F'_1(S)$ рис. 2.16 похідна обчислюється за формулою $F'_1(S) = \left(\frac{1}{1+e^{-y_1}}\right) \cdot \left(1 - \frac{1}{1+e^{-y_1}}\right)$. Формули для обчислень значень наступних ваг такі:

$$\begin{aligned}
 w'_{(x_1)1} &= w_{(x_1)1} + \eta \cdot F'_1(S) \cdot x_1 \cdot \delta_1; \\
 w'_{(x_2)1} &= w_{(x_2)1} + \eta \cdot F'_1(S) \cdot x_2 \cdot \delta_1; \\
 w'_{(x_1)2} &= w_{(x_1)2} + \eta \cdot F'_2(S) \cdot x_1 \cdot \delta_2; \\
 w'_{(x_2)2} &= w_{(x_2)2} + \eta \cdot F'_2(S) \cdot x_2 \cdot \delta_2; \\
 w'_{(y_1)1} &= w_{(y_1)1} + \eta \cdot F'_3(S) \cdot y_1 \cdot \delta; \\
 w'_{(y_2)1} &= w_{(y_2)1} + \eta \cdot F'_3(S) \cdot y_2 \cdot \delta;
 \end{aligned}
 \tag{2.19}$$

У наведених формулах величина η являє собою коефіцієнт швидкості навчання або крок, який коливається в межах 0..1 і може задаватися користувачем довільно.

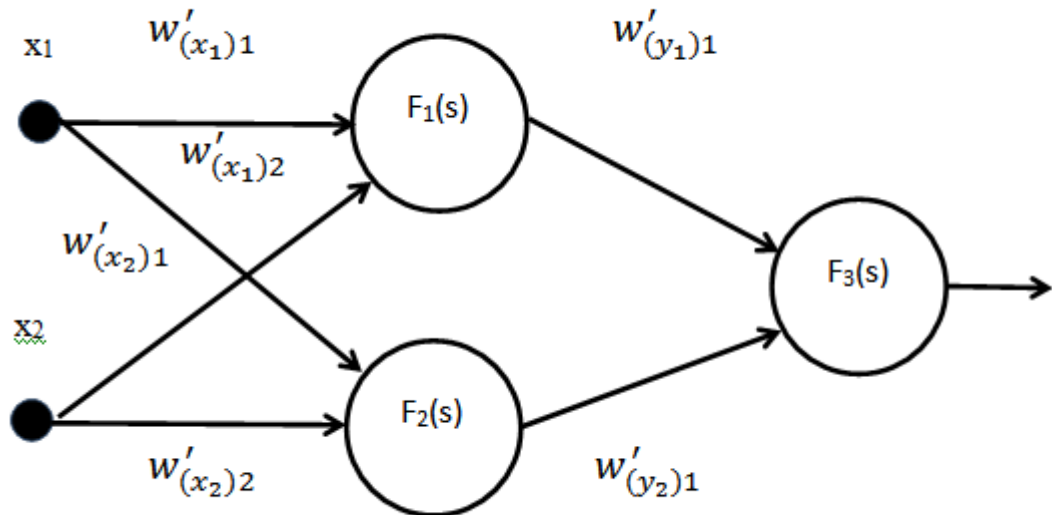


Рис. 2.17. Третій етап методу зворотного поширення помилки зі значеннями отриманих ваг зв'язків

Наведені етапи (їх ще називають епохами) повторюються до тих пір поки значення δ не стане рівним нулю, або поки подальші обчислення не призводять до зміни значення цієї величини. У цьому випадку штучна мережа вважається навченою. Графік зміни значення δ у відповідності до кількості повторів обчислень показний на рис. 2.18.

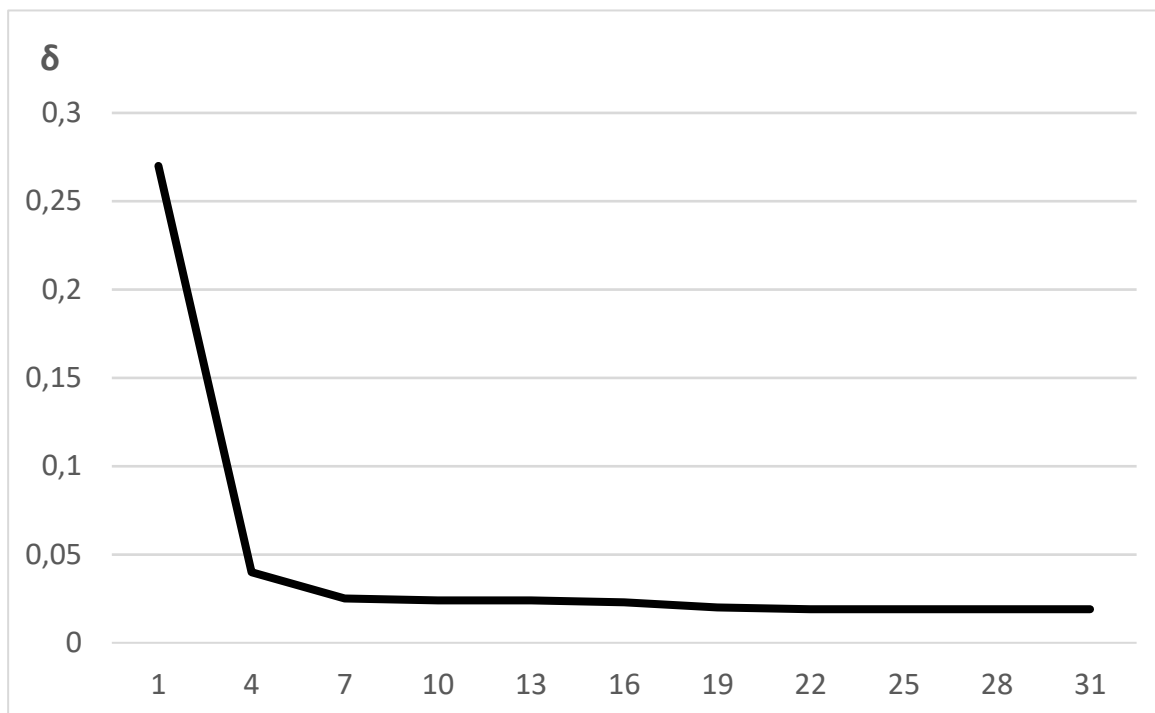


Рис. 2.18. Графік зміни значення δ (приклад)

Метод зворотного поширення помилки може бути використаний також для навчання мережі, які складається з одного штучного нейрона. Сьогодні в мережі Інтернет існує велика кількість програм в яких реалізований алгоритм методу зворотного поширення помилки.

Переваги методу.

Однією з переваг методу є простота реалізації та розуміння його роботи.

Недоліки методу.

До недоліків методу відносять перш за все невизначеність терміну навчання мережі, який може бути досить тривалим. Вказують навіть на те, що нейронна мережа взагалі може не навчитися. Причини цього можуть бути такі [17-19]:

Параліч мережі.

У процесі навчання мережі значення ваг можуть в результаті корекції стати дуже великими величинами. Це може призвести до того, що всі або більшість нейронів будуть функціонувати при дуже великих вихідних значеннях, в області, де похідна стискаючої функції дуже мала. Так як помилка, що посиляється назад у процесі навчання, пропорційна цій похідній, то процес навчання може

практично завмерти. У теоретичному відношенні ця проблема погано вивчена. Зазвичай цього уникають зменшенням розміру кроку η , але це збільшує час навчання. Різні евристики використовувалися для запобігання від паралічу або для відновлення після нього, але поки що вони можуть розглядатися лише як експериментальні.

Локальні мінімуми.

Зворотне поширення використовує різновид градієнтного спуску, тобто здійснює спуск вниз по поверхні помилки, безперервно підлаштовуючи ваги в напрямку до мінімуму. В складній мережі поверхня помилки значним чином деформовані і складається з великої кількості нерівностей (пагорбів, складок і ярів), розташованих в просторі значної розмірності. Результат обчислень може потрапити в локальний мінімум (неглибоку долину), при наявності набагато більш глибоких мінімумів. В точці локального мінімуму всі напрямки ведуть вгору, і результат буде локально оптимальним. Статистичні методи навчання можуть допомогти уникнути такої ситуації, але вони досить повільні.

Розмір кроку.

Уважний розбір доведення збіжності показує, що корекції ваг передбачаються нескінченно малими. Це нездійсненно на практиці, тому що веде до безкінечного часу навчання. Розмір кроку повинен братися скінченним. Якщо розмір кроку фіксований і дуже малий, то збіжність надто повільна, якщо ж він фіксований і занадто великий, то може виникнути параліч або постійна нестійкість. Ефективно збільшувати крок до тих пір, поки не припиниться поліпшення оцінки в даному напрямку антиградієнта і зменшувати, якщо такого покращення не відбувається. Відмічають також відмітити можливість перенавчання мережі, що є скоріше результатом помилкового проектування її топології. При дуже великій кількості нейронів втрачається властивість мережі узагальнювати інформацію. Весь набір образів, наданих до навчання, буде вивчений мережею, але будь-які інші образи, навіть дуже схожі, можуть бути класифіковані невірно.

2.8. Використання методу зворотного поширення помилки для прогнозування часових рядів

Здійснення прогнозування часових рядів за допомогою нейронних мереж передбачає початкове їх навчання, отримання необхідних ваг з подальшим використанням їх для прогнозування. Для такого навчання використовуються тренувальні ряди, табл. 2.3.

Таблиця 2.3

Приклад тренувального ряду для тренування нейрона

x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15
1,59	5,73	0,48	5,28	1,35	5,91	0,77	5,25	1,37	4,42	0,26	4,21	1,9	4,08	1,4

Приклад та дані тренувального ряду отримані з [13]. В цьому джерелі для прогнозування був використаний штучний нейрон наступного виду, рис. 2.19. Такий вибір очевидно здійснений з метою наочності пояснення. На практиці для прогнозування використовують більш складні нейронні мережі, які містять декілька шарів штучних нейронів.

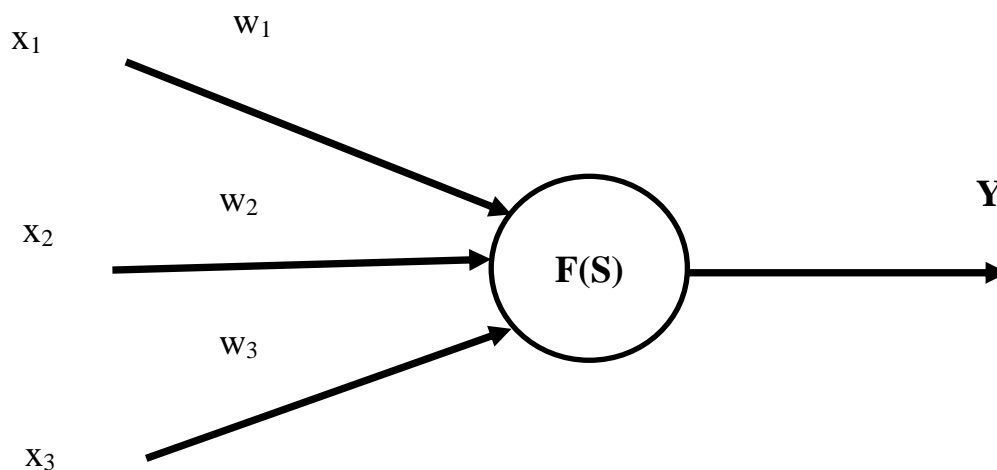


Рис. 2.19. Штучний нейрон для прогнозування часового ряду

Під час навчання на вхід нейрону подаються трійки значень із табл. 2.3, четверте значення цієї таблиці є очікуваним виходом нейрона Y . Кожна наступна трійка просувається вперед змінюючи вхідні та вихідні значення у послідовності показані в табл. 2.4.

Таблиця 2.4

Послідовність зміни тренувальних значень часового ряду під час навчання штучного нейрона

i-й набір даних	Вхід нейрона	Вихід (Y)
1	1,59; 5,73; 0,48	5,28
2	5,73; 0,48; 5,28	1,35
3	0,48; 5,28; 1,35	5,91
4	5,28; 1,35; 5,91	0,77
...
10	4,42; 0,26; 4,21	1,9

Для навчання використовуються 13 значень табл. 2.3, останні два значення цієї таблиці використовуються для перевірки результатів навчання. Навчання здійснюється з використання алгоритму зворотного поширення помилки, розглянутого вище.

Наведений приклад досить легко моделюється за допомогою засобів Microsoft Excel VBA шляхом створення макросу, який забезпечує використання заданого числа циклів при розрахунках. Така модель наведена в практичних завданнях.

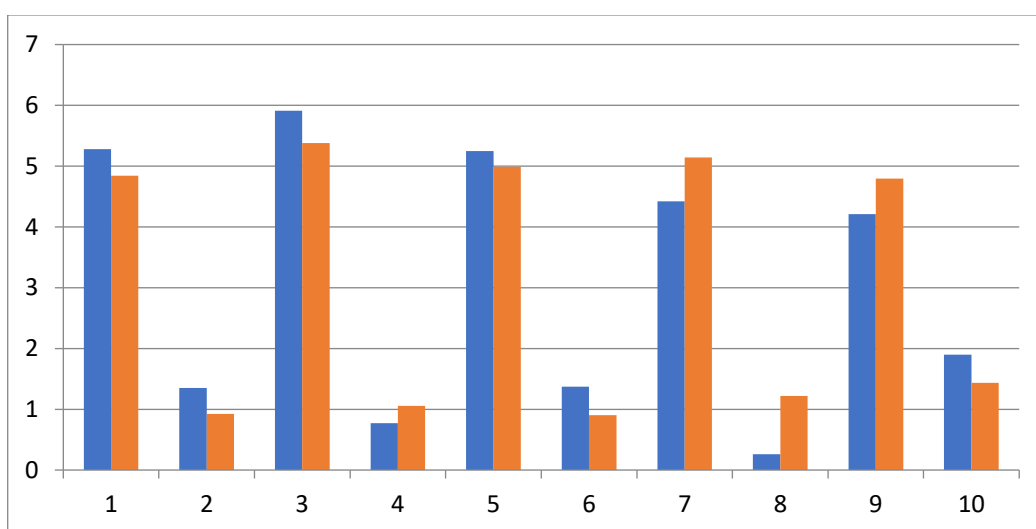


Рис. 2.20. Результат прогнозування. Задані значення – перший ряд стовпчиків, прогнозовані – другий ряд

Перевірка результатів навчання для двох останніх значень табл. 2.3 дає наступний результат:

для x_{14} – задане значення – 4,08, прогнозоване – 5,18;

для x_{15} – задане значення – 1,4, прогнозоване – 1,72.

2.9. Мережі Кохонена

До мереж Кохонена відносять групу штучних нейронних мереж з одним шаром. В більшості своїй їх представляють у вигляді одновимірних – 1D, двовимірних – 2D, (рис. 2.21) та тривимірних – 3D, (рис. 2.22). Існує також ряд модифікацій цього типу мереж[20].

В мережах Кохонена використовується метод навчання який називають конкурентним. Цей метод не передбачає використання вихідних сигналів (y_1, y_2, \dots, y_n). Сутність методу полягає у тому, щоб визначити нейрони з найбільшими вагами (так званих переможців). Стосовно цих нейронів відбувається зміна ваг, для решти нейронів ваги залишаються незмінними. Такий принцип називають «переможець забирає все». Використання конкурентного методу добре підходить для здійснення кластерного аналізу.

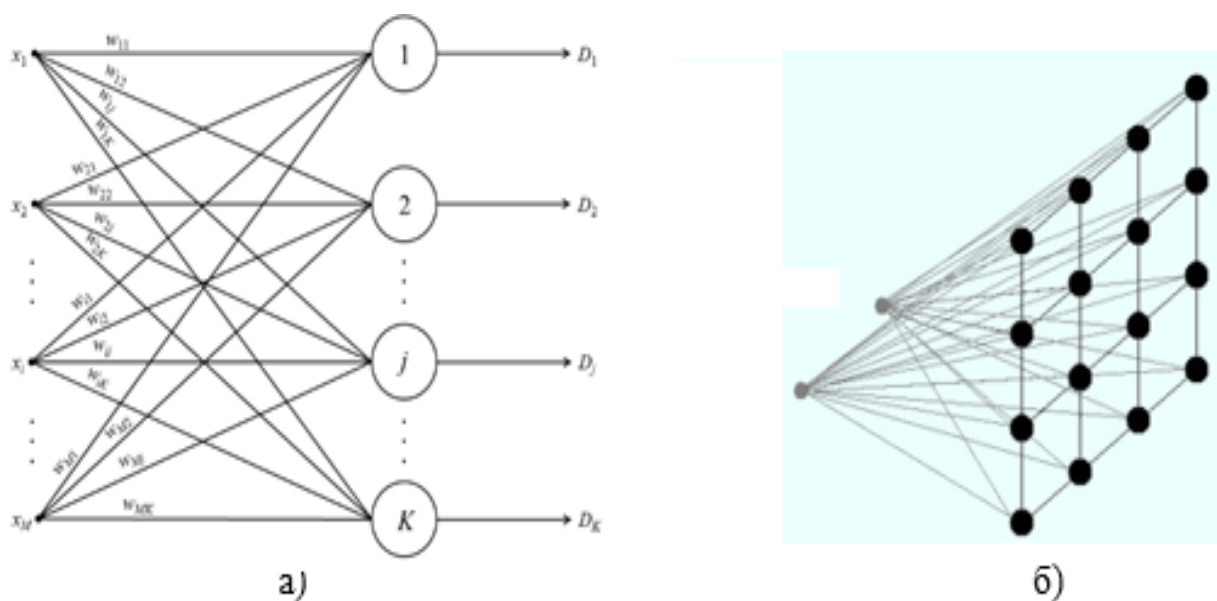


Рис. 2.21. Приклади одновимірної а) та двовимірної б) мереж Кохонена

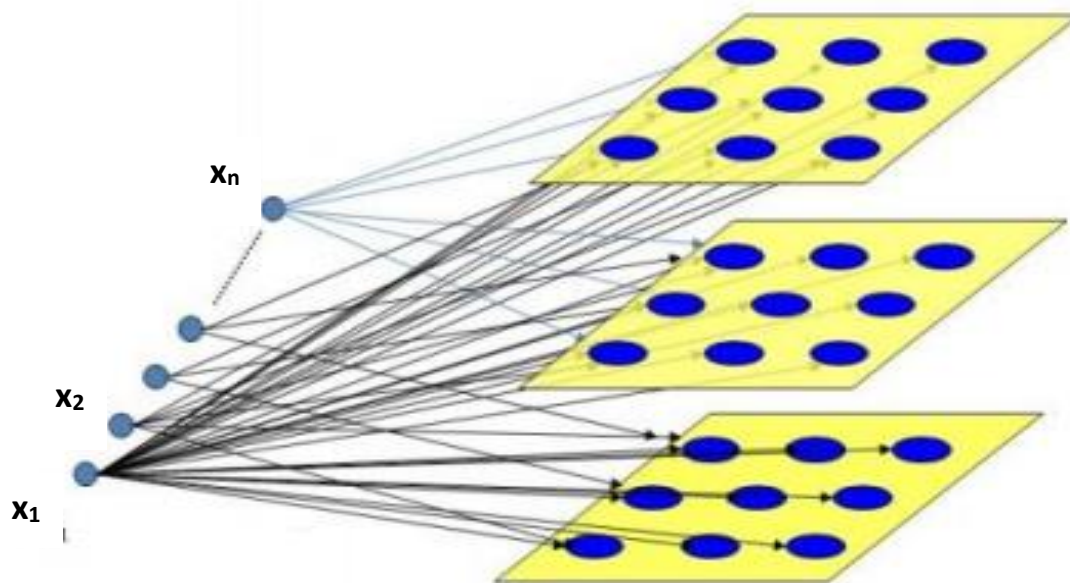


Рис. 2.22. Тривимірна мережа Кохонена

Пояснення принципу роботи мережі Кохонена та здійснення кластерного аналізу на основі обробки її результатів виконаємо, використовуючи наступний приклад. Необхідно згрупувати студентів відповідно до їх успішності, використовуючи дані таблиці 2.5. Задано початковий коефіцієнт швидкості навчання який дорівнює 0,30. З кожною епохою він зменшується на 0,05.

Таблиця 2.5

Вихідні дані для вирішення задачі кластеризації

№	Прізвища	Стать	Заліки	Предмети					Коефіцієнт стипендії
				Теорія прийняття рішень	Системний аналіз	Штучний інтелект	Операційні системи	Бази даних	
		x1	x2	x3	x4	x5	x6	x7	x8
1	Прізвище 1	Ч	Так	60	79	60	72	63	1
2	Прізвище 2	Ч	Ні	60	61	30	5	17	0
3	Прізвище 3	Ж	Ні	60	61	30	66	58	0
4	Прізвище 4	Ч	Так	85	78	72	70	85	1,25
5	Прізвище 5	Ж	Так	65	78	60	67	65	1
6	Прізвище 6	Ж	Так	60	78	77	81	60	1,25
7	Прізвище 7	Ж	Так	55	79	56	69	72	0
8	Прізвище 8	Ч	Ні	55	56	50	56	60	0

Продовження табл. 2.5

9	Прізвище 9	Ч	Ні	55	60	21	64	50	0
10	Прізвище 10	Ч	Ні	60	56	30	16	17	0
11	Прізвище 11	Ж	Так	85	89	85	92	85	1,75
12	Прізвище 12	Ж	Так	60	88	76	66	60	1,25
13	Прізвище 13	Ч	Ні	55	64	0	9	50	0
14	Прізвище 14	Ч	Так	80	56	50	69	50	0
15	Прізвище 15	Ч	Ні	55	60	30	8	60	0

Нейронна мережа Кохонена для вирішення цієї задачі показана на рис. 2.23. Вона складається з чотирьох нейронів на вхід яких подають вектор з 7 сигналів – $\{x_1, x_2, \dots, x_7\}$, які відповідають нормалізованим даним таблиці 2.5 (змінна x_8 не буде використовуватися для обчислень в мережі, але враховуватиметься під час кластерного аналізу. Нормалізація здійснюється за формулою

$$\tilde{x}_{ij} = \frac{x_{ij} - x_{jmin}}{x_{jmax}}, \quad (2.20)$$

де

\tilde{x}_{ij} – i -те нормалізоване значення j -ї колонки таблиці 2.5;

x_{ij} – i -те значення j -ї колонки таблиці 2.5;

x_{jmin} – мінімальне значення j -ї колонки таблиці 2.5;

x_{jmax} – максимальне значення j -ї колонки таблиці 2.5.

Результат нормалізації представлений в табл. 2.6. В цій таблиці колонки з позначенням статі і зданих заліків нормовані наступним чином – чоловіча стаття – 1, жіноча – 0, всі заліки дані – 1, не всі заліки здані – 0. Решта колонок нормовані за формулою (2.20).

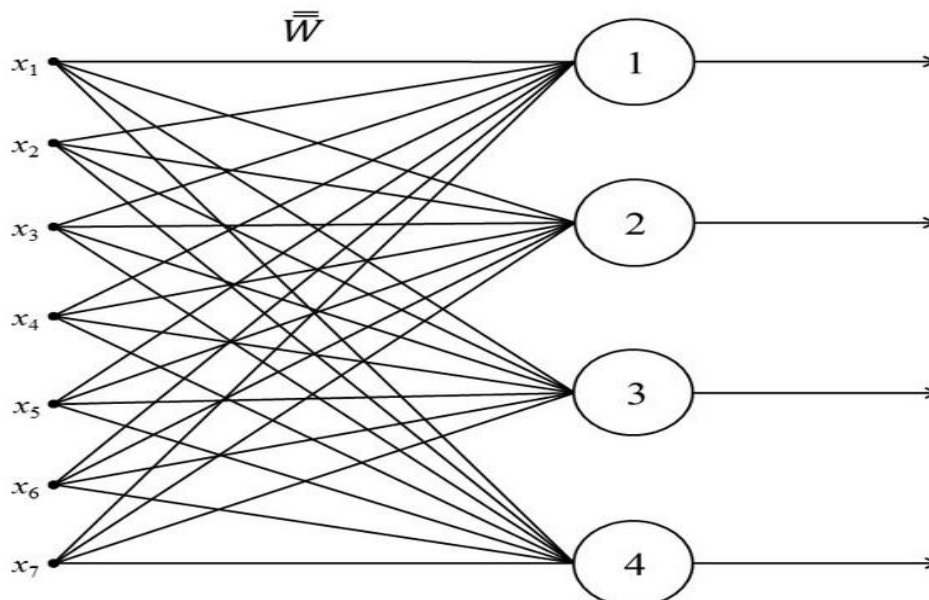


Рис. 2.23 Нейронна мережа Кохонена для розглядуваного прикладу

Таблиця 2.6

Результат нормалізації таблиці 2.5

№	Прізвища	Стать	Заліки	Предмети					Коефіцієнт стипендії
				Теорія прийняття рішень	Системний аналіз	Штучний інтелект	Операційні системи	Бази даних	
		x1	x2	x3	x4	x5	x6	x7	x8
1	Прізвище 1	1	1	0,059	0,258	0,706	0,728	0,541	0,571
2	Прізвище 2	1	0	0,059	0,056	0,353	0,000	0,000	0,000
3	Прізвище 3	0	0	0,059	0,056	0,353	0,663	0,482	0,000
4	Прізвище 4	1	1	0,353	0,247	0,847	0,707	0,800	0,714
5	Прізвище 5	0	1	0,118	0,247	0,706	0,674	0,565	0,571
6	Прізвище 6	0	1	0,059	0,247	0,906	0,826	0,506	0,714
7	Прізвище 7	0	1	0,000	0,258	0,659	0,696	0,647	0,000
8	Прізвище 8	1	0	0,000	0,000	0,588	0,554	0,506	0,000
9	Прізвище 9	1	0	0,000	0,045	0,247	0,641	0,388	0,000
10	Прізвище 10	1	0	0,059	0,000	0,353	0,120	0,000	0,000
11	Прізвище 11	0	1	0,353	0,371	1,000	0,946	0,800	1,000
12	Прізвище 12	0	1	0,059	0,360	0,894	0,663	0,506	0,714
13	Прізвище 13	1	0	0,000	0,090	0,000	0,043	0,388	0,000
14	Прізвище 14	1	1	0,294	0,000	0,588	0,696	0,388	0,000
15	Прізвище 15	1	0	0,000	0,045	0,353	0,033	0,506	0,000

Таблиця ваг мережі рис. 2.23 має наступний вигляд

Таблиця ваг мережі

Нейрони	Ваги						
	W_{1j}	W_{2j}	W_{3j}	W_{4j}	W_{5j}	W_{6j}	W_{7j}
1	0,2	0,2	0,3	0,4	0,4	0,2	0,5
2	0,2	0,8	0,7	0,8	0,7	0,7	0,8
3	0,8	0,2	0,5	0,5	0,4	0,4	0,4
4	0,8	0,8	0,6	0,7	0,7	0,6	0,7

Оскільки мережа має чотири нейрони, кількість кластерів також буде дорівнювати чотирьом.

Створення і аналіз кластерів передбачає такі етапи.

Перший етап. Розрахунок відстаней до нейронів з використанням значень ваг. Для цього відбираються рядки таблиці 2.6 і розраховуються відстані до всіх нейронів за формулою:

$$R_j = \sqrt{\sum_{i=1}^M (\tilde{x}_i - w_{ij})^2}, \quad (2.21)$$

де

R_j – відстань для j -го нейрона;

\tilde{x}_i – нормоване значення i -го рядка(із таблиці 2.6);

w_{ij} – вага i -го рядка для j -го нейрона(із таблиці 2.7);

M – кількість значень у рядку.

Наприклад результат такого обчислення для рядка 10 дає значення – для першого нейрона 1,08, для другого – 1,85, для третього – 0,87, для четвертого – 1,52. Мінімальне значення належить третьому нейрону. Отже це «нейрон-переможець». Саме для цього нейрона проводиться коригування ваг, для інших – ні (за принципом «переможець забирає все»). Коригування ваг здійснюється за формулою:

$$w_{ij}^{(q+1)} = w_{ij}^{(q)} + v \cdot (\tilde{x}_i - w_{ij}^{(q)}), \quad (2.22)$$

де

$w_{ij}^{(q+1)}$ – нове значення ваги;

$w_{ij}^{(q)}$ – попереднє значення ваги;

ν – коефіцієнт швидкості навчання.

Відповідно для третього нейрона переможця нові значення ваг третього рядка табл. 2.7 будуть дорівнювати:

$$0,8+0,3(1-0,8)=0,86;$$

$$0,2+0,3(0-0,2)=0,14;$$

$$0,5+0,3(0,059-0,5)=0,37;$$

$$0,5+0,3(0-0,5)=0,35;$$

$$0,4+0,3(0,353-0,4)=0,385;$$

$$0,4+0,3(0,12-0,4)=0,316;$$

$$0,4+0,3(0-0,4)=0,28.$$

На першому етапі такі обчислення необхідно зробити для всіх рядків табл. 2.7. Обчислення цих рядків завершує одну епоху.

На другому етапі коефіцієнт швидкості навчання необхідно зменшити на 0,05 і повторити обчислення в послідовності яка була задана на першому етапі. Зменшуючи з кожною епохою коефіцієнт швидкості навчання на 0,05 і обчислюючи ваги нейронів, отримуємо результуючу таблицю ваг, табл. 2.8.

Таблиця 2.8

Результуюча таблиця ваг

Нейрони	Ваги						
	W1j	W2j	W3j	W4j	W5j	W6j	W7j
1	0,06	0,06	0,13	0,16	0,37	0,52	0,49
2	0,00	1,00	0,12	0,31	0,85	0,76	0,61
3	1,00	0,00	0,02	0,04	0,31	0,23	0,32
4	0,99	0,99	0,26	0,17	0,71	0,71	0,57

На третьому етапі виконується виділення кластерів і їх аналіз. Для виділення кластерів використовується формула:

$$N_K = \min\{\sum_{i=1}^M (\tilde{x}_i - w_{iK})^2\}, \quad (2.23)$$

де

K – кількість нейронів(кластерів);

M – кількість сигналів на вході (у наведеному прикладі – 7);

N_K – K -й кластер;

w_{iK} – i -та вага K -го нейрона.

Наприклад, викривуючи результуючу таблицю 2.8 та перший рядок табл. 2.6 отримаємо табл. 2.9.

Таблиця 2.9

Приклад отримання кластера

0,8816	0,8816	0,0054	0,0095	0,1146	0,0427	0,0029	0,3265	2,265
0,9989	0,0000	0,0042	0,0024	0,0194	0,0011	0,0046	0,3265	1,357
0,0000	0,9996	0,0017	0,0483	0,1536	0,2526	0,0482	0,3265	1,831
0,0000	0,0000	0,0396	0,0080	0,0000	0,0005	0,0009	0,3265	0,375(мінімальне значення)

Цей приклад отримання кластерів є суто демонстративним, тому що під час проміжних обчислень кластери вже розподілені і їх можна легко визначити за мінімальними відстанями ваг. Резульгуюча таблиця розподілу кластерів наведена нижче.

Таблиця 2.10

Таблиця розподілу кластерів за наведеним прикладом разом із результатами аналізу

№	Розмір кластера	Стать	Отримані заліки	Середній бал	Коефіцієнт стипендії	Висновок
1	3	Ж	Ні	55	0	Студенти жіночої статі, які погано вчаться і не отримують стипендію
2	5-7,11,12	Ж	Так	72,3	1,05	Студенти жіночої статі які гарно вчаться і отримують стипендію
3	2,8-10,13,15	Ч	Ні	42,3	0	Студенти чоловічої статі, які погано вчаться і не отримують стипендію
4	1,4,14	Ч	Так	68,6	0,75	Студенти чоловічої статі які в більшості своїй гарно вчаться і отримують стипендію

В табл. 2.10 в останній колонці дано пояснення всім кластерам відповідно до вихідних значень. У підсумку маємо чотири кластери, по яких нерівномірно розподілилися дані вихідної таблиці 2.5. У першому кластері, опинилося тільки значення третього рядка, в четвертому – першого, четвертого та чотирнадцятого. Найбільш насиченими виявилися другий та третій кластери. Загалом групу студентів можна характеризувати як середню (приблизно половина навчається – погано і дещо більше половини – добре). При цьому гірші показники у навчанні демонструють представники чоловічої статі. Геометрична інтерпретація розподілу прізвищ за кластерами(нейронами) показана на рис. 2.24.

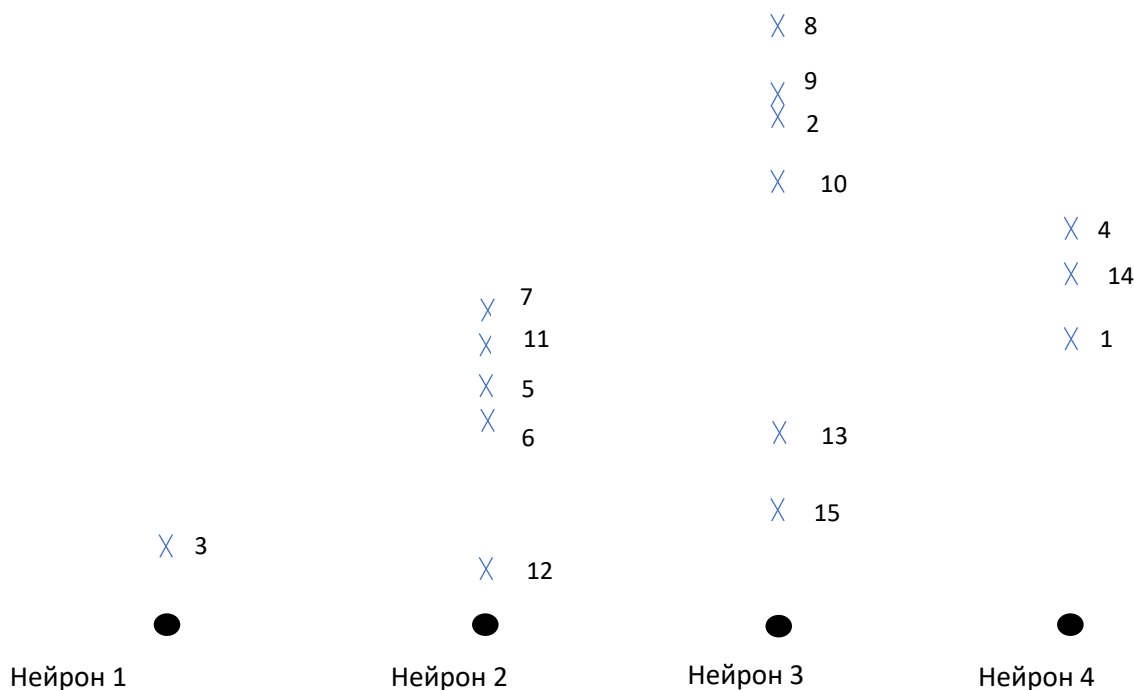


Рис. 2.24. Геометрична інтерпретація розподілу прізвищ за кластерами (номери рядків відповідають прізвищам табл. 2.5)

2.10. Карти Кохонена

Подібне до розглянутого конкурентне навчання використовують в мережах Кохонена великої розмірності – тривимірних і вище. Для отримання прийнятних для аналізу даних в таких мережах їх перетворюють з багатовимірних на двовимірні. Перетворені двовимірні мережі називають картами Кохонена. Ці карти застосовуються при вирішенні ряду практичних задач, таких, як стиснення цифрової інформації, задачах класифікації, здійснення аналізу, наукових дослідженнях тощо. На рис. 2.25 показаний результат обробки багатовимірної мережі за допомогою карти Кохонена, виконаний в програмі Deductor Studio [21].

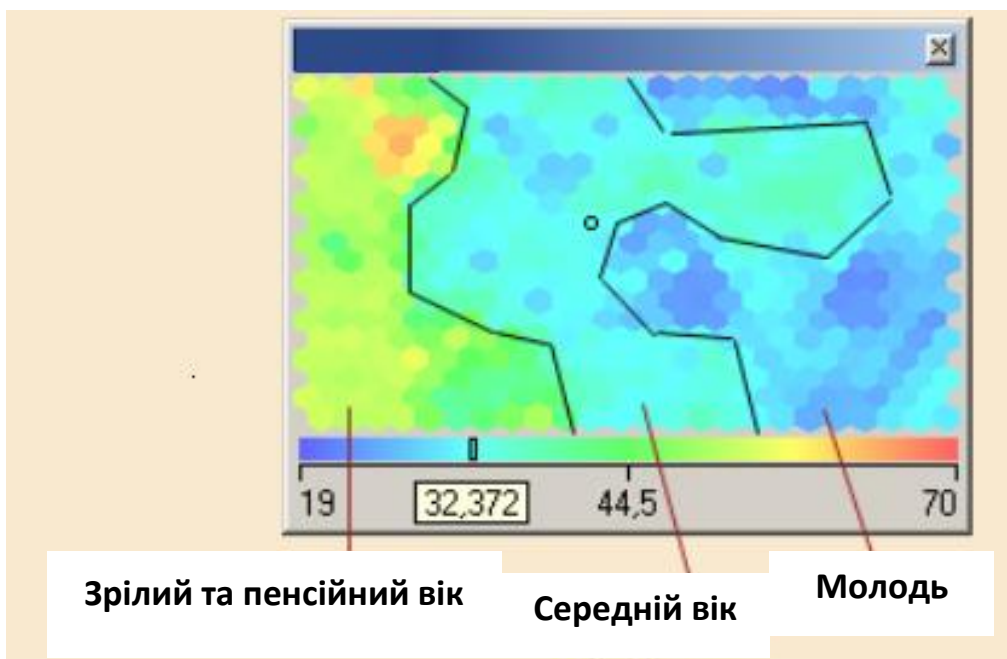


Рис. 2.25. Результат обробки багатовимірної мережі для аналізу абонентів стільникового зв'язку за дев'яти показниками за допомогою програми Deductor Studio

2.11. Інші різновиди нейронних мереж

В цьому посібнику розглянуті лише окремі різновиди із усього широкого спектру існуючих штучних нейронних мереж. Крім розглянутих існує багато видів штучних нейронних мереж, які відрізняються за будовою і використовуються для вирішення багатьох практичних задач. Наприклад, глибинні мережі, згорткові (розгорткові) мережі, мережі Джордана, мережі Елмана, мережі Ворда та інші. Топологічно вони мало відрізняються від розглянутих мереж. Всі вони мають вхідний шар, прихований шар та вихідний шар нейронів. Відмінності складають лише типи зв'язків між нейронами

Детально ознайомитися з різними видами штучних нейронних мереж та їх застосування можна у відповідних вітчизняних або іноземних джерелах [12,22,23].

2.12. Практичне використання штучних нейронних мереж

Під час розгляду функціонування окремих мереж вище було зазначено, що одним із застосувань нейронних мереж є розпізнавання образів, вирішення завдань оптимізації та кластерного аналізу. Крім цього нейронні мережі застосовуються в багатьох галузях людської діяльності, таких як медицина, економіка та фінанси, обробка інформації, техніка зв'язку тощо.

Зокрема, відомим прикладом застосування нейронних мереж в економіці є прогнозування ситуації на фондовому ринку коли в якості вхідної інформації в нейронних мережах використовуються зміни значень котирувань. Оскільки природа фондових ринків суттєво утруднює виділення єдиного показника оцінювання існує необхідність у використанні багатокритеріальних моделей, які можуть бути побудовані з використанням нейронних мереж. Зокрема в лондонському відділенні банку Нью-Йоркського банку (Citibank N.A. London) розроблені програми з прогнозування курсів валют, що ґрунтуються на використанні нейронних мереж [24].

Також частим явищем в банківській сфері є використання нейронних мереж для управління фінансовими ризиками. В якості показників оцінювання в цих мережах використовуються показники динаміки розвитку компанії, кредитна історія, стабільність фінансових показників тощо.

В медицині нейронні мережі активно використовуються в діагностиці різноманітних захворювань, таких як онкологічні та серцево-судинні.

Також нейронні мережі застосовуються під час проектування мереж зв'язку та здійснення їх оптимізації. Вони використовуються для знаходження оптимального трафіку між комунікаційними вузлами. Окрім управління маршрутизацією потоків, нейронні мережі також використовуються для отримання ефективних рішень в сфері проектування нових телекомунікаційних мереж, а також для швидкого кодування та декодування даних, стиснення відеоінформації та вирішення ряду інших задач.

У галузі безпеки і охоронних системах нейронні мережі необхідні для ідентифікації особи, розпізнавання голосу, осіб в натовпі, розпізнавання

автомобільних номерів, аналіз аерокосмічних знімків, моніторингу інформаційних потоків, виявлення підробок тощо.

Наведений перелік використання нейромереж є неповним. Існує цілий ряд перспективних напрямків їх використання, серед яких системи технічного діагностування та лінгвістичний переклад, системи відеоаналізу тощо.

Такий широкий спектр застосування штучних нейронних мереж свідчить про наявність умов для їх подальшого розвитку в науці і техніці.

Контрольні питання

1. Який склад штучного нейрона?
2. Які бувають види штучних нейронних мереж?
3. В чому суть методу зворотного поширення помилки?
4. Як здійснюється розпізнавання образів за допомогою нейронних мереж?
5. В чому полягає призначення і особливість нейронної мережі Гопфілда?
6. Яким чином уникають паралічу нейронної мережі?
7. Для чого застосовують карти Кохонена?
8. Як здійснюється прогнозування за допомогою штучних нейронних мереж?

РОЗДІЛ 3. ІНТЕЛЕКТУАЛЬНІ АГЕНТИ

3.1. Поняття інтелектуального агента

Під терміном «інтелектуальні агенти» в штучному інтелекті розуміють сутність (наприклад, прилади, роботи, виконавчі пристрої тощо) що спостерігають за навколишнім середовищем і діють у ньому, при цьому їхня поведінка раціональна в тому розумінні, що вони здатні до розуміння і їхні дії завжди спрямовані на досягнення якої-небудь мети[25]. В цьому контексті інтелектуальні агенти діють з використанням зародків мислення, подібних до мислення людини, або інтелектуально розвинених істот. Вони можуть:

- здійснювати генерацію відповіді за відсутності готового рішення;
- виконувати пізнавальне виділення об'єктивних умов, суттєвих для дії;
- здійснювати узагальнене, опосередковане відображення дійсності;
- відшукувати і відкривати суттєво нове;
- виявляти і досягати проміжних цілей[26].

Дуже часто на практиці інтелектуальні агенти є роботами, які демонструють набір дій які виконує людина – грають в ігри, виконують певні роботи тощо.

У штучному інтелекті існує декілька груп агентів. Агенти з простою поведінкою, цілеспрямовані агенти, агенти що навчаються та інші.

3.2. Алгоритм Q-навчання інтелектуальних агентів

Стимулююче навчання є одним із видів навчання з підкріпленням. Під час цього навчання інтелектуальний агент діючи в певному середовищі отримує за свої дії певну «винагороду» у вигляді накопичених балів. Кінцевим завданням інтелектуального агента є вироблення моделі поведінки, яка дозволяє оптимально діяти в заданому середовищі. Основою для Q-навчання є так звана функція корисності

$$Q[s, a] = R[s, a] + \textit{Gamma} \cdot \textit{Max}(Q[s', a'], \quad (3.1)$$

де

s – елемент множини станів $S(s_1, s_2 \dots s_n)$ в яких може перебувати агент;

a – елемент множини дій агента $A(a_1, a_2 \dots a_n)$;

s' – елемент попереднього стану з множини станів $S(s_1, s_2 \dots s_n)$ в яких може перебувати агент;

a' – елемент попередньої дії агента $A(a_1, a_2 \dots a_n)$;

Γ – швидкість навчання від 0 до 1 (рекомендоване значення 0,8).

Алгоритм Q-навчання можна пояснити на досить розповсюдженішому прикладі, який описаний в багатьох джерелах, (рис. 3.1).

Інтелектуальному агенту (нехай це буде рухомий робот) необхідно навчитися самостійно знаходити виходи із помешкання, показаного на рис. 3.1. В цьому помешканні для зручності формалізації кожна кімната пронумеровані. Вихід назовні із кімнат номер 1 та номер 4 також має номер –5.

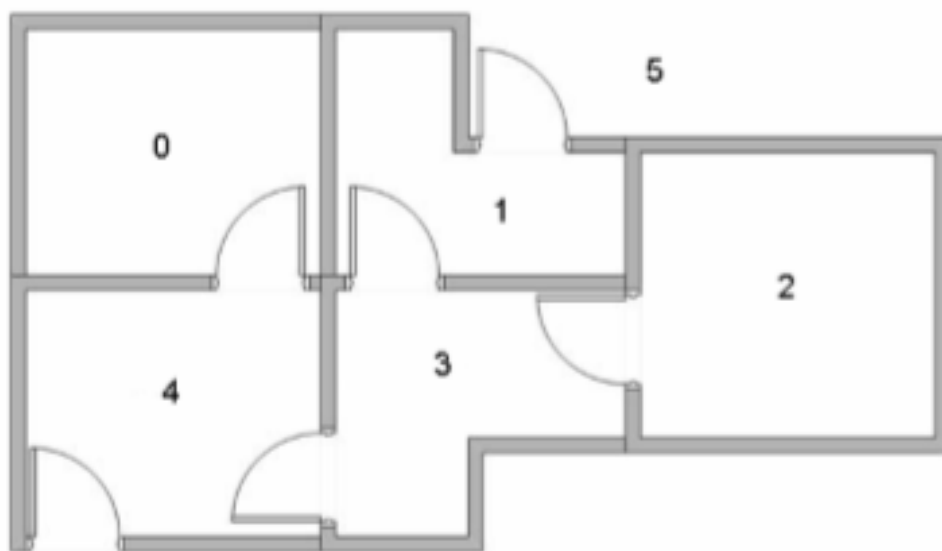


Рис. 3.1. Схематичний план помешкання для Q-навчання інтелектуального агента

Пронумеровані кімнати та вихід назовні дають можливість описати стани в яких може перебувати агент:

0 стан - перебування у кімнаті №0;

1 стан - перебування у кімнаті №1;

2 стан - перебування у кімнаті №2;

3 стан - перебування у кімнаті №3;

4 стан - перебування у кімнаті №4;

5 стан - вихід.

Стан 5 є цільовим. Оскільки мета агента – вихід з помешкання.

Рух агента під час переходу із одного стану в інший (під час переміщення із кімнати в кімнату та виходу назовні) можна подати у вигляді графа в якому також позначити винагороду за вихід балами. Такий граф показаний на рис. 3.2. Перебування назовні також винагороджується в 100 балів.

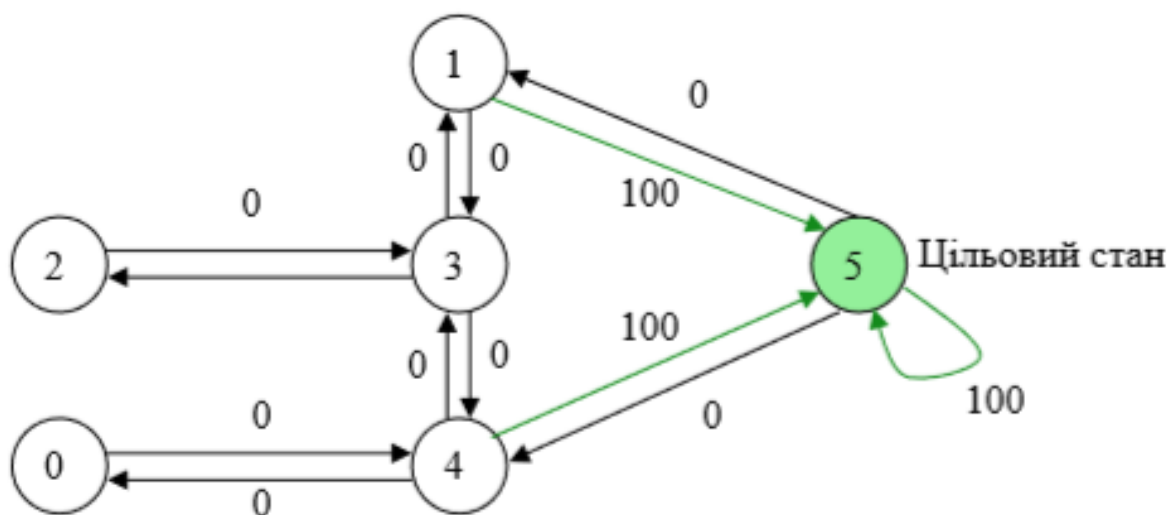


Рис. 3.2. Граф можливих переходів інтелектуального агента та винагорода за вихід та перебування назовні

Масив станів R в прикладі подається у вигляді матриці, рис. 3.3. На цьому рисунку позначка ‘-1’ вказує на неможливість потрапляння у кімнату, позначка ‘0’ – про наявність такої можливості, а цифри 100 – винагороду за потрапляння на вихід.

		Стан					
		0	1	2	3	4	5
Дія	0	-1	-1	-1	-1	0	-1
	1	-1	-1	-1	0	-1	100
	2	-1	-1	-1	0	-1	-1
	3	-1	0	0	-1	0	-1
	4	0	-1	-1	0	-1	100
	5	-1	0	-1	-1	0	100

Рис. 3.3. Масив R можливих стані агента

Масив Q (пам'ять інтелектуального агента) подана з припущенням про наявність початкової інформації про цільовий стан, рис. 3.4.

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	0	0	0	100
2	0	0	0	0	0	0
3	0	0	0	0	0	0
4	0	0	0	0	0	100
5	0	0	0	0	0	100

Рис. 3.4. Масив Q – пам'ять інтелектуального агента

Методика навчання агента реалізована наступним чином:

1 крок. Агент випадковим чином опинився в кімнаті 3. З матриці R він може дізнатися як йому діяти. Можливі переходи з цієї кімнати в інші можуть бути до кімнат 1,2 або 4. Агент випадковим чином обирає перехід до кімнати 1.

2 крок. Агент знаходиться в кімнаті 1. Звідси він може потрапити до кімнати 3 або вийти назовні (стан 5). Оскільки переходячи до стан 5 агент отримає винагороду в 100 балів він природно обере тільки цей шлях. Тоді винагорода за попередню дію, яку агент здійснив, перейшовши з кімнати 3 в кімнату 1 (зі стану 3 в стан1) буде складати (з урахування того, що $\Gamma=0,8$):

$$Q[3, 1] = R[3, 1] + 0,8 \cdot \text{Max}(Q[1, 3], Q[1, 5]) = 0 + 0,8 \cdot 100 = 80 \quad (3.2)$$

В масив Q можна записати значення, рис. 3.5.

	0	1	2	3	4	5
0	0	0	0	0	0	0
1	0	0	0	0	0	100
2	0	0	0	0	0	0
3	0	80	0	0	0	0
4	0	0	0	0	0	100
5	0	0	0	0	0	100

Рис. 3.5. Стан масиву Q після кроку 2

Після цього кроку граф можливих переходів інтелектуального агента буде мати вигляд, рис. 3.6.

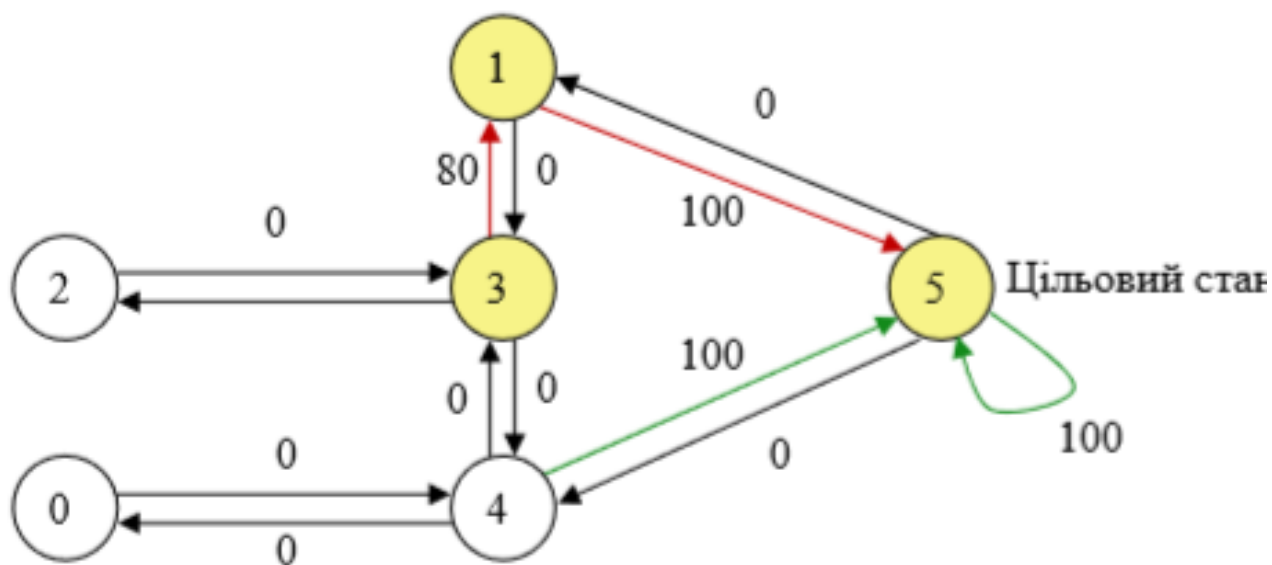


Рис. 3.6. Вигляд графа можливих переходів агента після 2 кроку

3 крок. Виконати подібні обчислення для всіх переходів інтелектуального агента. Наприклад, перехід із кімнати 1 до кімнати 3 (повернення назад) дасть наступний результат.

$$Q[1, 3] = R[1, 3] + 0,8 \cdot \text{Max}(Q[3, 1], Q[1, 3]) = 0 + 0,8 \cdot 80 = 64.$$

Результуюче обчислення масиву Q матиме вигляд, рис 3.7 що відповідає графу рис. 3.8.

	0	1	2	3	4	5
0	0	0	0	0	80	0
1	0	0	0	64	0	100
2	0	0	0	64	0	0
3	0	80	51	0	80	0
4	64	0	0	64	0	100
5	0	80	0	0	80	100

Рис. 3.7 Стан масиву пам'яті Q інтелектуального агента

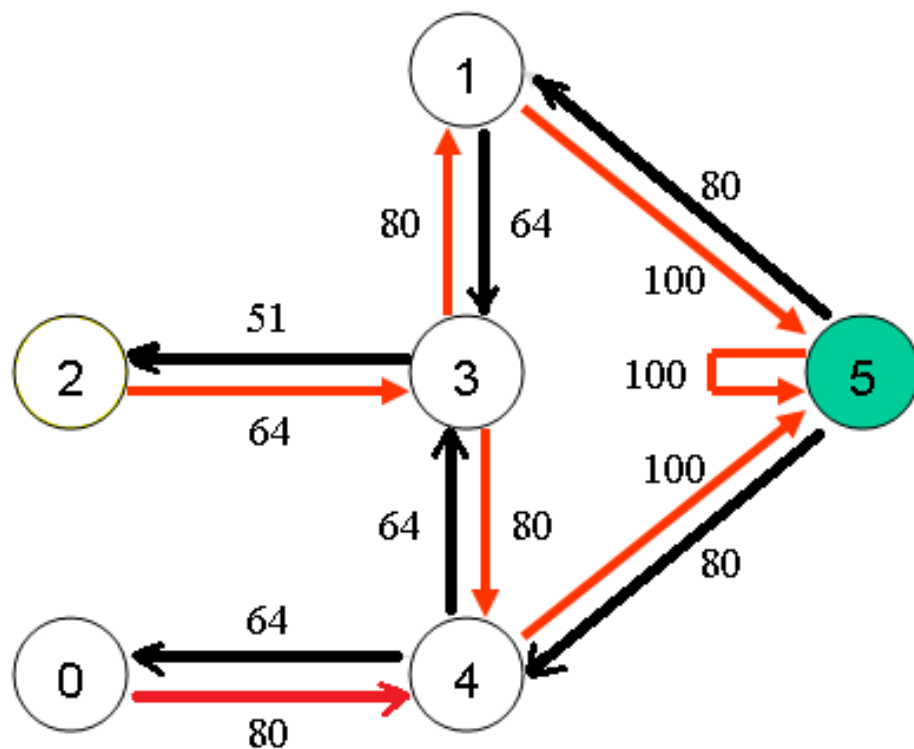


Рис. 3.8. Результат графа можливих переходів інтелектуального агента

Тепер, використовуючи свою пам'ять Q агент може без проблем самостійно вибратися з будь-якої кімнати помешкання рухаючись у напрямку з максимальною нагородою. Однак, на практиці навчання такого інтелектуального

агента здійснюється дещо іншим чином. Він рухається самостійно, переїжджаючи з кімнати в кімнату і виїжджаючи назовні, потім знову повертається, доки не «запам'ятає» всі можливі варіанти. При цьому в масив пам'яті Q постійно записуються бали винагороди. Зовнішній вигляд такого «реального» масиву пам'яті показаний на рис. 3.9. Він нічим не відрізняється від масиву на рис. 3,7 остільки містить ті ж самі цифри помножені на 5.

	0	1	2	3	4	5
0	0	0	0	0	400	0
1	0	0	0	320	0	500
2	0	0	0	320	0	0
3	0	400	256	0	400	0
4	320	0	0	320	0	500
5	0	400	0	0	80	500

Рис. 3.9. Стан масиву пам'яті Q інтелектуального агента при самостійному пошуку виходу

3.3. Використання інтелектуальних агентів

Інтелектуальні агенти знайшли широке застосування для вирішення цілого ряду різноманітних завдань прикладного характеру. Окремі задач оптимізації інтелектуальні агенти активно використовуються для інформаційного пошуку, створення експертних систем, інтенсифікації процесу навчання, вирішення завдань комп'ютерної безпеки, моніторингу догожнього руху, створенні систем електронної комерції та в багатьох інших застосуваннях. Однак, слід розрізняти поняття інтелектуальних агентів як програм, що самостійно виконують завдання, указане користувачем комп'ютера, протягом тривалих проміжків часу та інтелектуальних агентів які здатні до розуміння і виконання самостійних завдань. Перше значення використовується у комп'ютерній науці, а друге в штучному інтелекті. Тому, коли мова йде про інтелектуальних агентів слід чітко розрізняти ці два поняття. Головною ознакою інтелектуальних агентів, що використовуються в штучному інтелекті є автономність. Використання

інтелектуальних агентів у штучному інтелекті є більш обмеженим і тут мова може йти про вбудовану програмну систему або робота. Решта застосувань до штучного інтелекту не відносяться, хоча і мають таку саму назву.

Контрольні питання

1. Що таке інтелектуальні агенти? В чому різниця цього терміну в комп'ютерній науці та в штучному інтелекті?
2. Що є основою Q-навчання інтелектуальних агентів?
3. В чому відмінність масиву R від масиву Q ?
4. Як відбувається Q-навчання інтелектуальних агентів? Що для цього використовується?

РОЗДІЛ 4. КОЛЕКТИВНИЙ ІНТЕЛЕКТ

4.1. Визначення колективного інтелекту

Колективний інтелект або ройовий інтелект досить нове положення в теорії штучного інтелекту [27]. Природна аналогія колективного інтелекту ґрунтується на спостереженні за поведінкою великих популяцій живих істот дії яких є інтелектуально обумовленими при вирішенні певних завдань виживання. До таких істот відносяться мурахи, бджоли, птахи, риби та інші істоти, які утворюють колонії або зграї що діють за певним алгоритмом, використовуючи переваги чисельності для пошуку їжі, уникнення загроз з боку хижаків тощо. Головними ознаками колективного інтелекту є – наявність групи особин (в більшості своїй досить чисельної); можливість універсального способу інформаційного обміну в групі; наявність спільної мети; чітка підпорядкованість особин спільній меті; можливість керування діями групи.

Алгоритми які розроблені в галузі колективного інтелекту застосовуються, перш за все, в задачах комбінаторної оптимізації та для розв'язування задачі комівояжера.

4.2. Використання мурашиного алгоритму для вирішення задачі комівояжера

Задача комівояжера полягає у знаходженні найвигіднішого (в окремому випадку найкоротшого) маршруту, що проходить через вказані міста хоча б по одному разу. В дуже багатьох випадках ця задача використовується для демонстрації роботи того чи іншого методу [28]. На прикладі цієї задачі також можна пояснити яким чином працює мурашиний алгоритм. Якщо пристосувати умови задачі до цього алгоритму, то мураха-розвідник у пошуках за їжею має обійти всі місця де ця їжа знаходиться не заходячи повторно в одне й те саме місце. Під час обходу місць мураха використовує пахучі речовини (феромони), які були залишені попередниками на кожному з маршрутів. Інтенсивність запаху феромонів, які залишили попередники – задається. Схема можливих маршрутів мурахи показана на рис. 4.1.

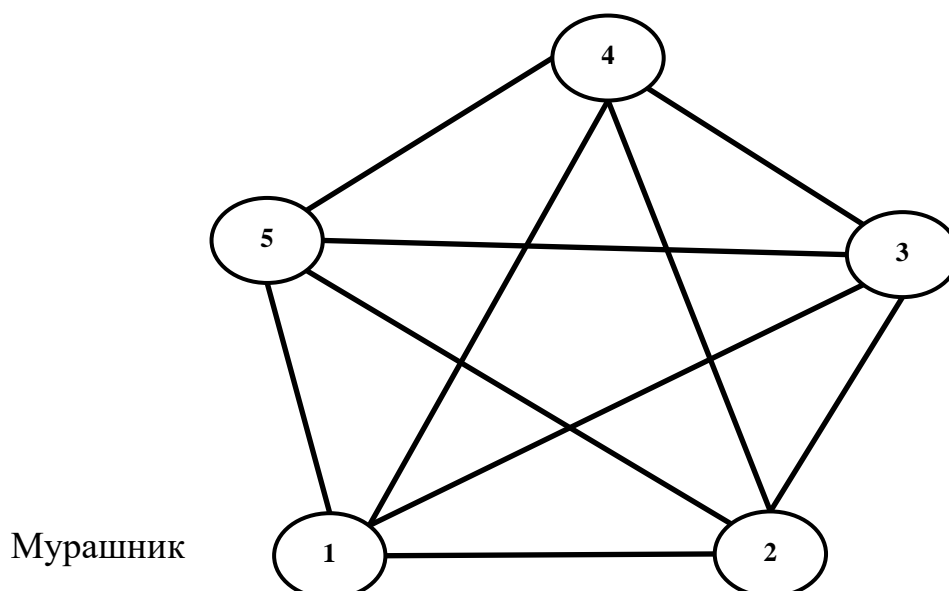


Рис. 4.1. Приклад можливих маршрутів мурахи в задачі комівояжера

Ймовірність переходу по i -му маршруту обчислюється за формулою:

$$P_i = \frac{\left(\frac{1}{l}\right)_k^q \cdot f_i^p}{\sum_{k=0}^N \left(\frac{1}{l}\right)_k^q \cdot f_k^p}, \quad (4.1)$$

де

P_i – ймовірність переходу шляхом i ;

l_i – довжина i -го переходу;

f_i – кількість феромонів на i -му переході;

q – величина, що визначає «жадібність» алгоритму;

p – величина, яка визначає «стадність» алгоритму;

$q+p=1$.

Слід зазначити, що випадки коли $q=0$ (відповідно $p=1$) або коли $p=0$ ($q=1$) є крайніми випадками при яких вплив однієї із зазначених складових $\left(\frac{1}{l}\right)^q$ або f_i^p анулюється. Якщо $q=1$, то феромони не випаровуються зовсім і відповідно мураха жадібно прямує по маршруту з найбільш інтенсивним запахом, а якщо $p=1$, то феромони після кожного проходу мурахи не залишаються і кожна нова

мураха повинна йти за маршрутом не орієнтуючись на запах попередніх мурах(відсутність колективного впливу стадності). В наведеному прикладі ці два значення q і p ігноруються для спрощення, що ніяк не впливає на сутність самого алгоритму.

Для прикладу використання мурашиного алгоритму використана таблиця значень в якій вказані маршрути пересування із одного місця з їжею до іншого, довжина кожного з маршрутів (в умовних одиницях) та інтенсивність запаху феромонів на кожному з маршрутів (3 – максимальна інтенсивність, 1 – мінімальна інтенсивність).

Таблиця 4.1

Таблиця вихідних даних для прикладу мурашиного алгоритму

№	Маршрути	Довжина	Інтенсивність запаху феромонів
	m	l	τ
1	1-2	56	2
2	1-3	39	3
3	1-4	43	2
4	1-5	39	1
5	2-3	50	3
6	2-4	61	2
7	2-5	41	1
8	3-4	54	3
9	3-5	62	1
10	4-5	37	2

Послідовність роботи мурашиного алгоритму під час вирішення задачі комівояжера наступна.

На **першому етапі** мурахою розвідником здійснюється вибір початкового маршруту. Для цього за формулою (4.1) розраховуються ймовірності переходів мурахи з мурашника за кожним з можливих маршрутів. Наприклад, перехід з пункту 1 (мурашника) до їжі в пункті 2 (за маршрутом 1-2 розраховується наступним чином:

$$P_{1-2} = \frac{\frac{1}{56} \cdot 2}{\frac{1}{56} + \frac{1}{39} + \frac{1}{43} + \frac{1}{39}} = 0,19 \quad (4.2)$$

Значення ймовірностей для першого переходу наступні – $P_{1-2} = 0,19$; $P_{1-3} = 0,42$; $P_{1-4} = 0,25$; $P_{1-5} = 0,14$.

На **другому етапі** за допомогою ймовірнісного вибору визначається за яким маршрутом буде здійснено перехід. Таке визначення робиться шляхом використання колеса рулетки, подібно до того як це робилося у генетичному алгоритмі. Замість колеса також можна використовувати інші побудови, наприклад відрізок на такому відкладаються ймовірності. На рис. 4.2 показаний такий відрізок.

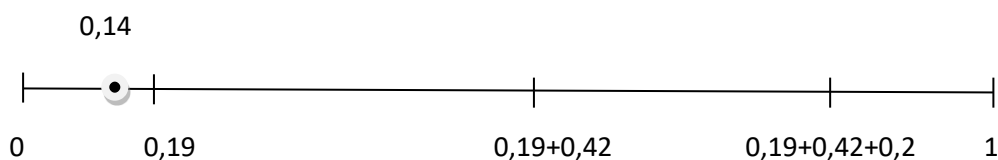


Рис. 4.2. Відрізок ймовірностей для прикладу

Використання ймовірнісної функції RANDBETWEEN із програми MS Excel дала значення 0,14 (рис. 4.2). Отже, мураха повинна переміститися за маршрутом 1-2, рис. 4.3.

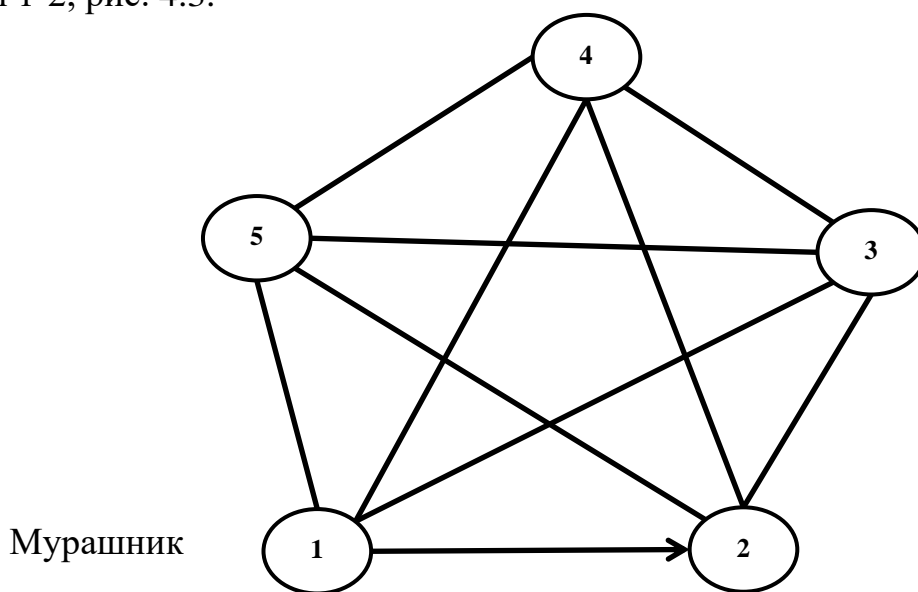


Рис. 4.3 – Переміщення мурахи після другого етапу

На **третьому етапі** виконуються обчислення наступних маршрутів переміщення мурахи за принципом, викладеним на першому та другому етапах. При цьому кількість обчислень зменшується оскільки повертатися в уже пройдені місця не можна. Результати обчислень та вибору маршрутів показані на рис. 4.4.

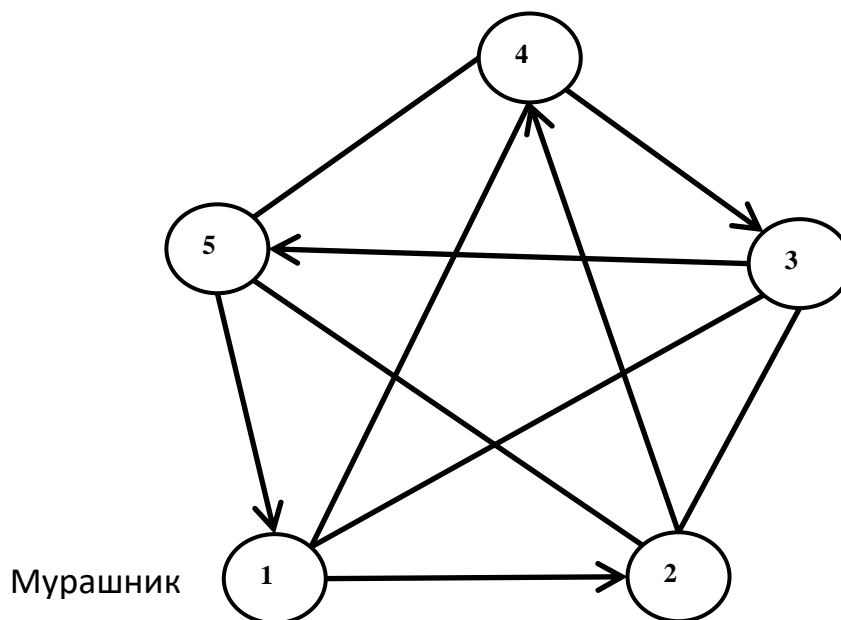


Рис. 4.4. Результат вибору маршрутів за даними прикладу

При цьому сумарна довжина визначених маршрутів складає 272 одиниці. Очевидно, що ця довжина не є мінімальною, отже слід повторювати пошук шляху іншими мурахами-розвідниками, поки мінімальне значення не буде знайдено. При цьому слід враховувати два фактори. По-перше, кожна мураха-розвідник залишатиме на маршрутах свої власні феромони, і по-друге, перед кожним походом нової мурахи феромони відкладені попередниками випаровуються і втрачають запах. Ці фактори враховуються за допомогою відповідної формули:

$$\tau^* = \tau \cdot (1 - d) + \Delta\tau, \quad (4.3)$$

де

d – задана інтенсивність випаровування феромонів від 0 до 1;

$\Delta\tau$ – додана кількість феромонів мурахою-розвідником, яка розраховується за формулою $\Delta\tau = \frac{Q}{L_0}$. (Q – задана величина, L_0 – довжина шляху, пройденого однією мурахою).

В табл. 4.2 наведені значення для стану феромонів після висихання і проходу однієї мурахи-розвідника.

Таблиця 4.2

Значення для стану феромонів після висихання і проходу однієї мурахи-розвідника та значення проведених обчислень (обрані маршрути – виділені)

№	Маршрут и	Довжин а	Інтенсивніс ть запаху феромонів	Нові феромо ни	Висиханн я	Стан феромонів після висихання і проходу однієї мурахи
	m	l	τ	$\Delta\tau=Q/L$ 0	$\tau(1-d)$	$\tau(1-d)+\Delta\tau$
1	1-2	56	2	0,11	1,9	2,01
2	1-3	39	3	0	2,85	2,85
3	1-4	43	2	0	1,9	1,9
4	1-5	39	1	0	0,95	0,95
5	2-3	50	3	0	2,85	2,85
6	2-4	61	2	0,11	1,9	2,01
7	2-5	41	1	0	0,95	0,95
8	3-4	54	3	0,11	2,85	2,96
9	3-5	62	1	0,11	0,95	1,06
10	4-5	37	2	0	1,9	1,9

Слід зазначити, що мурашиний алгоритм є досить ефективним для вирішення задачі комівояжера або подібних до неї задач пошуку оптимальних рішень. Перевагою цього алгоритму є очевидна простота реалізації. Існує декілька модифікацій алгоритму, які дозволяють розширити його застосування для вирішення різноманітних задач[29-31]. Зокрема, одним із підходів до модернізації алгоритму є використання елітарного, а не ймовірнісного підходу, коли для переходу використовується найбільш ймовірний маршрут, а не той,

який заданий випадковим чином. Слід зазначити, що існує досить багато версій мурашиного алгоритму та задач які вирішуються з його допомогою. Детальніше з темою мурашиних алгоритмів можна ознайомитися в першоджерелі [32].

Науковий напрямок штучного інтелекту включає в себе багато застосунків і додатків, для вивчення окремих із яких необхідне ґрунтовне опрацювання значного навчального матеріалу. Зокрема в посібник не включений матеріал, який стосується онтології, експертних систем та інших, що пояснюється обмеженнями обсягу навчального плану. Багато з напрямів штучного інтелекту ще не оформилися у вигляді як класичних теоретичних напрацювань, тому дослідники ведуть активний науковий пошук за ними.

В посібнику надано допоміжний матеріал для засвоєння студентами предмету «Штучний інтелект», а також перелік лабораторних робіт які можуть використовуватися і вдосконалюватися викладачами цього предмету. Окремі розділи посібника можуть бути розширені і поглиблені викладачами при написанні власних робочих програм або навчальних матеріалів, з метою зорієнтувати студентів на вивчення тих положень, які відповідають специфіці того чи іншого навчального закладу або предмету.

Посібник також може бути використаний як початкове джерело для тих, кого цікавлять питання штучного інтелекту і розуміння того, яким чином здійснюється технічна реалізація його напрацювань.

Контрольні питання

1. Що таке колективний інтелект? Які його головні ознаки?
2. Для вирішення яких задач використовується колективний інтелект?
3. Які основні етапи мурашиного алгоритму?
4. Яким чином визначається ймовірність переходу в мурашиному алгоритмі?
5. Яким чином досягається оптимальний маршрут в мурашиному алгоритмі?

ПРАКТИЧНІ ЗАВДАННЯ

Лабораторна робота №1

Моделювання генетичного алгоритму

Мета роботи: Виробити практичні навички роботи з генетичними алгоритмами

Зміст завдання. Знайти максимуми функцій з використанням генетичного алгоритму відповідно до варіанту, використовуючи наведений приклад.

Таблиця А.1

Перелік функцій

Номер варіанту	Вид функції
1	$z = \frac{xy}{x^2 + 3y^2 + 2}$
2	$z = \frac{2}{x + y^3}$
3	$z = 2x^3 + y^2 - 5$
4	$z = \frac{y}{x^2 + 2y^2 + 1}$
5	$z = \frac{x}{x + y^2 - 2}$
6	$z = \frac{x^2 + 3y^2 + 2}{xy}$
7	$z = 5x^2 - y^2 + 12$

Приклад.

Необхідно знайти максимум функції, використовуючи засоби MS Excel.

$$z = \frac{x}{x^2 + 2y^2 + 1} \quad (\text{А. 1})$$

Рішення. Перші значення x та y обрані випадковим чином. Мутація не застосовується.

x	-2	-1	0	2	Перше покоління
y	0	-2	-1	1	
	Хромосома 1	Хромосома 2	Хромосома 3	Хромосома 4	
	z1	z2	z3	z4	Життєздатність покоління
Значення функції	-0,4	-0,1	0	0,2857	-0,214286
	Ця хромосома має мінімальне значення функції і виявилася нежиттєздатною	Для схрещування залишаються тільки ці хромосоми			
Схрещування(crossovering)					
	Хромосома 4(лідер)	Хромосома 3	Хромосома 2		
Хромосома з найбільш стійкими генами	2	0	-1		
	1	-1	-2		
x	0	-1	2		
y	1	1	-1	-2	Друге покоління
	Хромосома 1	Хромосома 2	Хромосома 3	Хромосома 4	
	z1	z2	z3	z4	Життєздатність покоління
Значення функції	0	-0,25	0,2857	0,1538	0,189560
	Для схрещування залишаються тільки ці хромосоми	Ця хромосома має мінімальне значення функції і виявилася нежиттєздатною	Для схрещування залишаються тільки ці хромосоми		

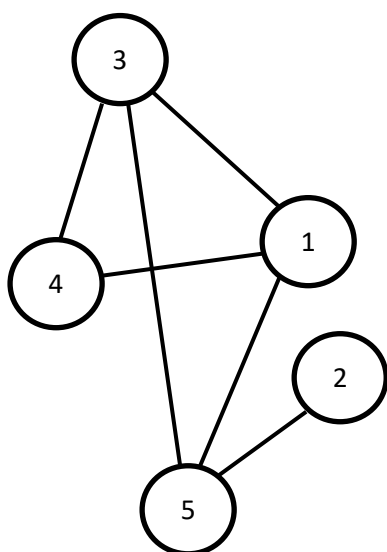
Схрещування(crossovering)				
	Хромосома 3 (лідер)	Хромосома 4	Хромосома 1	
Хромосома з найбільш стійкими генами	2 -1	2 -2	0 1	
x	2	0	2	
y	-1	-1	-2	2 1
	Хромосома 1	Хромосома 2	Хромосома 3	Хромосома 4
	z1	z2	z3	z4
Значення функції	0,2857	0	0,1538	0,2857
	Для схрещування залишаються тільки ці хромосоми	Ця хромосома має мінімальне значення функції і виявилася нежиттєздатною	Для схрещування залишаються тільки ці хромосоми	
Схрещування(crossovering)				
	Хромосома 1(лідер)	Хромосома 4	Хромосома 3	
Хромосома з найбільш стійкими генами(таких дві)	2 -1	2 -1	2 -2	
Очевидно, що за відсутності мутації ці значення x,y обертають функцію в максимум	2 -1	2 -1	2 1	2 -2
	z1	z2	z3	z4
Значення функції	0,2857	0,2857	0,2857	0,153846154
	Життєздатність покоління			
	1,011			

Лабораторна робота №2

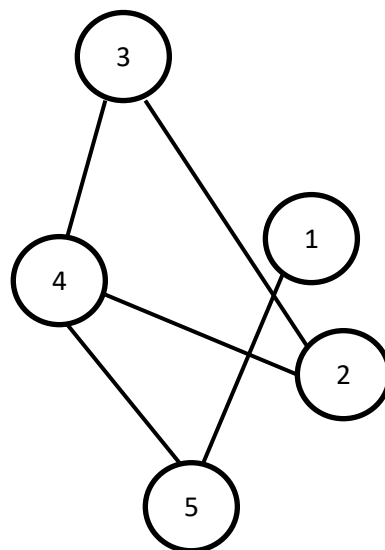
Моделювання мутації в генетичних алгоритмах

Мета роботи: Виробити навички застосування мутації при роботі з генетичними алгоритмами

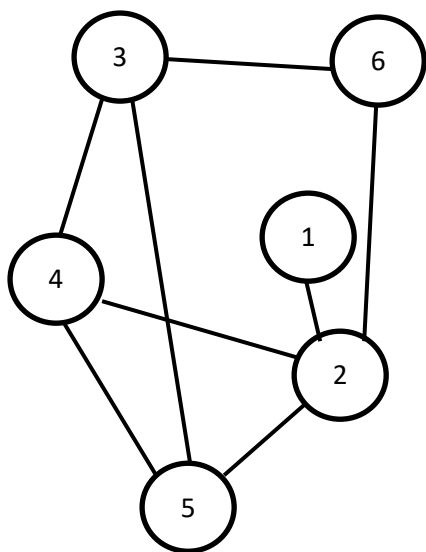
Зміст завдання. Використовуючи, метод моделювання мутації наведений в прикладі, розділу 1 знайти лінійне розташування вершин з мінімальною довжиною ребер для одного з наведених варіантів.



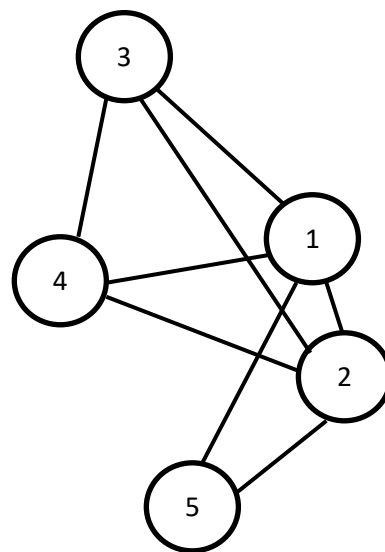
Варіант 1



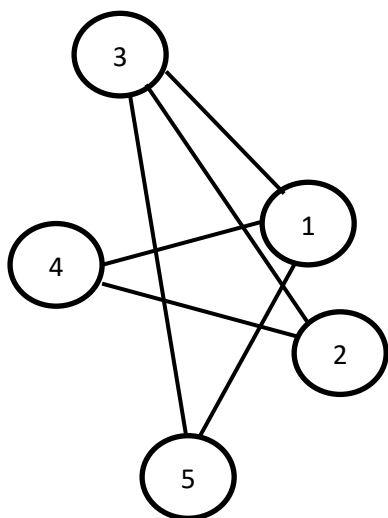
Варіант 2



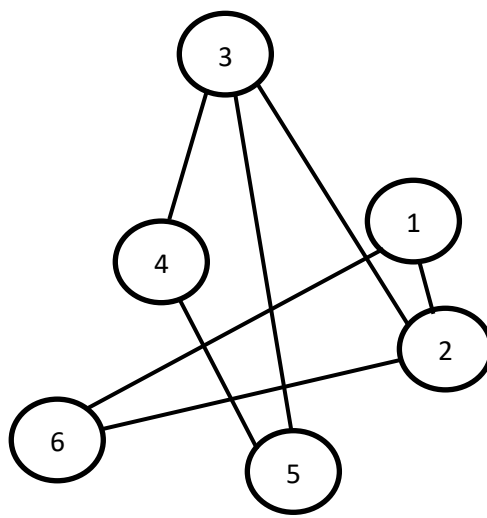
Варіант 3



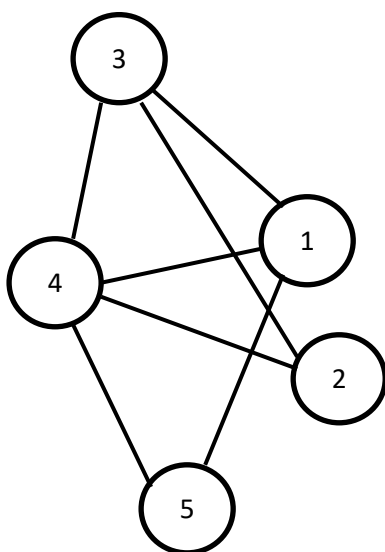
Варіант 4



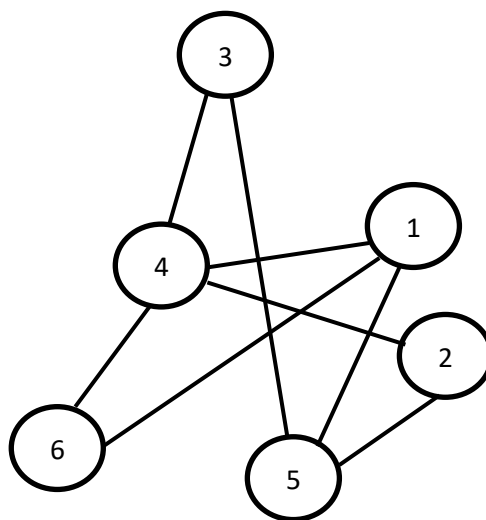
Варіант 5



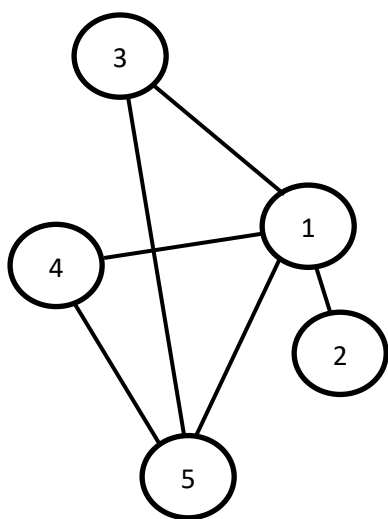
Варіант 6



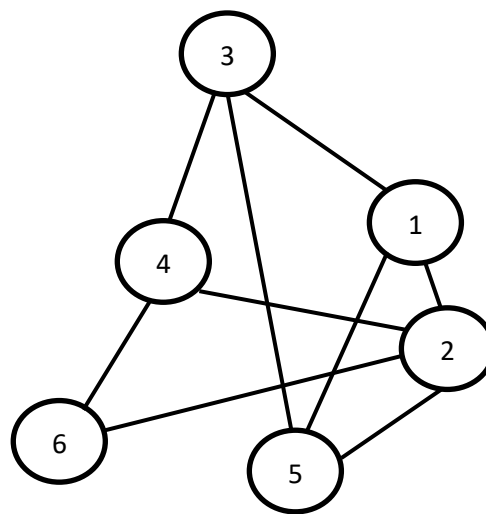
Варіант 7



Варіант 8



Варіант 9



Варіант 10

Лабораторна робота №3

Моделювання логічних функцій за допомогою штучних нейронів

Мета роботи: Виробити навички моделювання роботи штучних нейронів

Зміст завдання. Використовуючи штучні нейрони змоделювати роботу простого RS-тригера на базі MS Excel у відповідності зі схемою, рис. А.1.

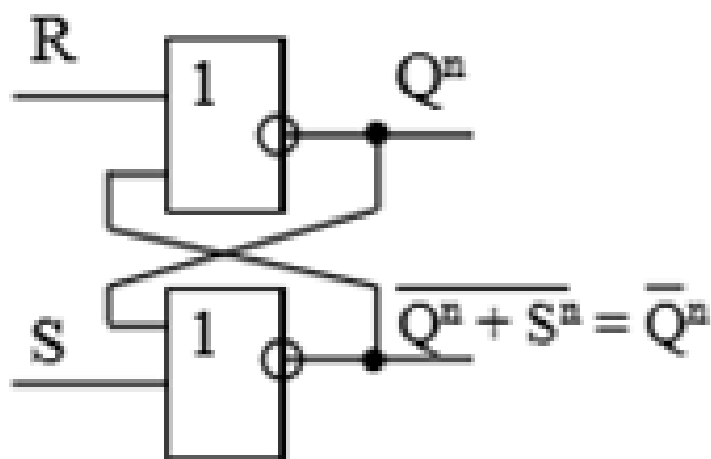


Рис. А.1. Схема RS-тригера на елементах АБО-НІ

Використовуючи штучні нейрони змоделювати роботу підсумовуючого дворозрядного лічильника на базі MS Excel у відповідності зі схемою, рис. А.2.

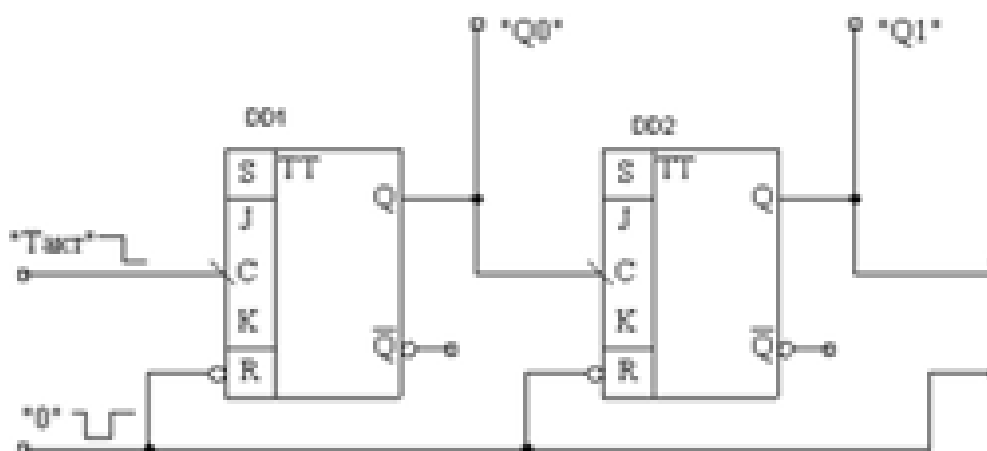


Рис. А.2. Схема дворозрядного лічильника

Використовуючи штучні нейрони змоделювати роботу повного лінійного шифратора на базі MS Excel у відповідності зі схемою, рис. А.3.

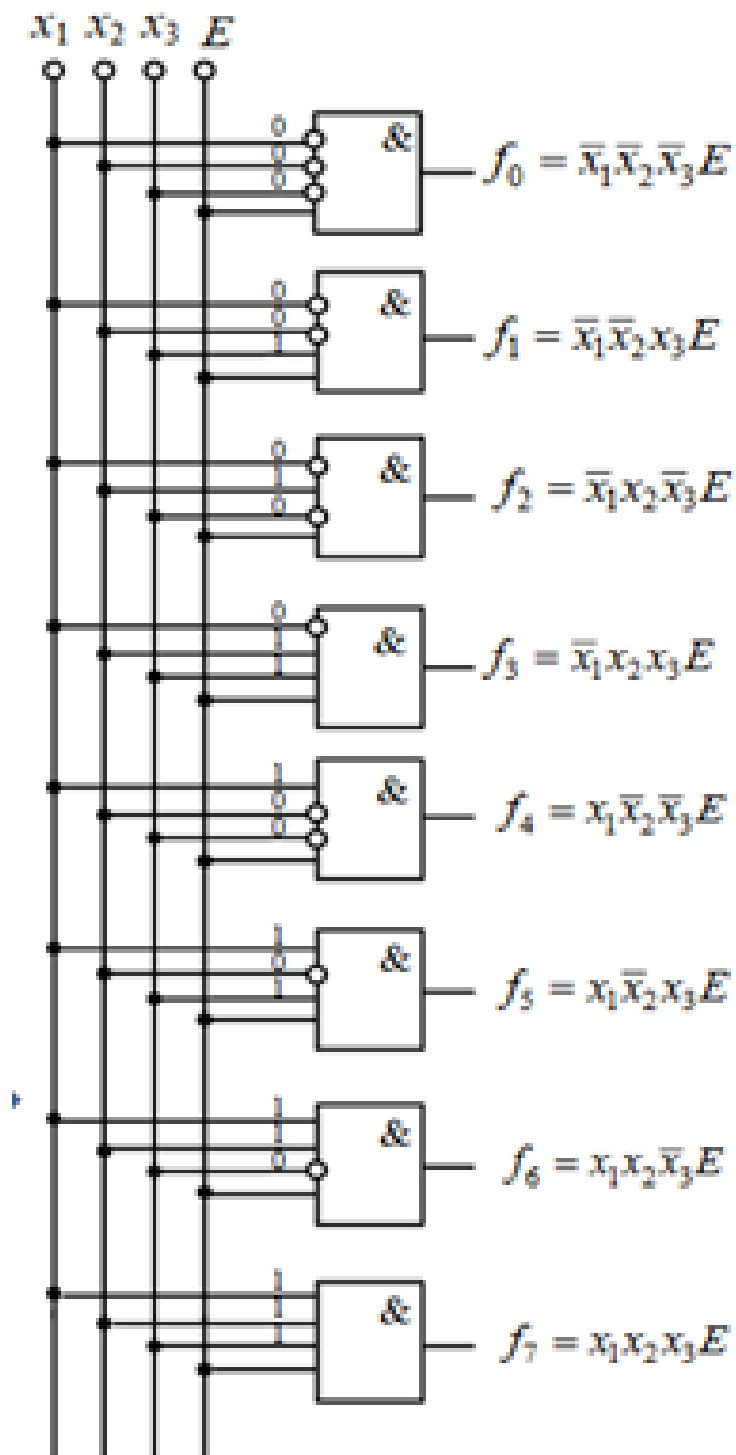


Рис. А.3. Схема повного лінійного шифратора

Лабораторна робота №4

Прогнозування часових рядів за допомогою штучного нейрона

Мета роботи: Вивчити можливості прогнозування часового ряду за допомогою штучного нейрона

Зміст завдання.

1. Відкрити файл Lab4.xlsx. Вибрати з таблиці Klj рядок відповідно до варіанту часового ряду заданого викладачем для здійснення прогнозування.
2. Для обраного рядка створити таблицю для прогнозування часового ряду, використовуючи 12 перших значень відповідно до формул.

$$E = \sum_{i=1}^N (Y_i - y_i)^2, \quad (\text{A. 2})$$

де

Y_i – обчислене значення виходу нейрона;

y_i – правильне значення наступного члену часового ряду;

N – кількість наборів вхідних даних.

$$E'_i = (Y_i - y_i) \times \frac{\exp(-s_i)}{(1 + \exp(-s_i))^2} \times x_i, \quad (\text{A. 3})$$

де

E'_i – i -те значення похідної функції помилки(одне з трьох);

x_i – значення часового ряду;

$s_i = x_{i-3} \times w_1 + x_{i-2} \times w_2 + x_{i-1} \times w_3$ – зважена сума активації штучного нейрона;

w_1, w_2, w_3 – синаптичні ваги штучного нейрона.

$$\Delta w_i = -v \times E'_i \quad (\text{A. 4})$$

де

v – коефіцієнт швидкості навчання.

$$\Delta w_{\text{середнє}} = \frac{1}{N} \sum_{i=1}^N \Delta w_i, \quad (\text{A. 5})$$

де

$\Delta w_{\text{середнє}}$ – середнє значення синоптичної ваги для i -го набору.

$$w_{\text{відкориговане}} = w + \Delta w_{\text{середнє}}, \quad (\text{A. 6})$$

де

$w_{\text{відкориговане}}$ – відкориговане значення синоптичної ваги, яке перезаписується з кожним циклом навчання.

Таблиця для прогнозування часового ряду, реалізована на MS Excel показана на рис. А.4.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
41																				
42	N	w1	w2	w3	x1	x2	x3	S	Yi	y	(Y-y)2	E	E1	E2	E3	0<v<1	Δw1	Δw2	Δw3	
43	1	0	0	0	1,59	5,73	0,48	0	5	5,28	0,0784	78,3994	-0,1113	-0,4011	-0,0336	0,1	0,01113	0,04011	0,00336	
44	2	0	0	0	5,73	0,48	5,28	0	5	1,35	13,3225	78,3994	5,228625	0,438	4,818	0,1	-0,5228625	-0,0438	-0,4818	
45	3	0	0	0	0,48	5,28	1,35	0	5	5,91	0,8281	78,3994	-0,1092	-1,2012	-0,30713	0,1	0,01092	0,12012	0,0307125	
46	4	0	0	0	5,28	1,35	5,91	0	5	0,77	17,8929	78,3994	5,5836	1,427625	6,249825	0,1	-0,55836	-0,1427625	-0,6249825	
47	5	0	0	0	1,35	5,91	0,77	0	5	5,25	0,0625	78,3994	-0,08438	-0,369375	-0,04813	0,1	0,0084375	0,0369375	0,0048125	
48	6	0	0	0	5,91	0,77	5,25	0	5	1,37	13,1769	78,3994	5,363325	0,698775	4,764375	0,1	-0,5363325	-0,0698775	-0,4764375	
49	7	0	0	0	0,77	5,25	1,37	0	5	4,42	0,3364	78,3994	0,11165	0,76125	0,19865	0,1	-0,011165	-0,076125	-0,019865	
50	8	0	0	0	5,25	1,37	4,42	0	5	0,26	22,4676	78,3994	6,22125	1,62345	5,2377	0,1	-0,622125	-0,162345	-0,52377	
51	9	0	0	0	1,37	4,42	0,26	0	5	4,21	0,6241	78,3994	0,270575	0,87295	0,05135	0,1	-0,0270575	-0,087295	-0,005135	
52	10	0	0	0	4,42	0,26	4,21	0	5	1,9	9,61	78,3994	3,4255	0,2015	3,26275	0,1	-0,34255	-0,02015	-0,326275	
53																	-0,2589965	-0,04051875	-0,241938	Середнє

Рис. А.4. Таблиця для прогнозування часового ряду

3. Створити копію таблиці рис. А.4 і розмістити її нижче. В цій таблиці в клітинках де записані синоптичні ваги $w1$, $w2$, $w3$ записати суми значень $w1+\Delta w1$, $w2+\Delta w2$, $w3+\Delta w3$, рис. А.5.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
41																					
42	N	w1	w2	w3	x1	x2	x3	S	Yi	y	(Y-y)2	E	E1	E2	E3	0<v<1	Δw1	Δw2	Δw3		
43	1	0	0	0	1,59	5,73	0,48	0	5	5,28	0,0784	78,3994	-0,1113	-0,4011	-0,0336	0,1	0,01113	0,04011	0,00336		
44	2	0	0	0	5,73	0,48	5,28	0	5	1,35	13,3225	78,3994	-0,28625	0,438	4,818	0,1	-0,5228625	-0,0438	-0,4818		
45	3	0	0	0	0,48	5,28	1,35	0	5	5,91	0,8281	78,3994	-0,1081	-1,2012	-0,30713	0,1	0,01092	0,12012	0,0307125		
46	4	0	0	0	5,28	1,35	5,91	0	5	0,77	17,8929	78,3994	5,5836	-1,427625	6,249825	0,1	-0,55836	-0,1427625	-0,6249825		
47	5	0	0	0	1,35	5,91	0,77	0	5	5,25	0,0625	78,3994	-0,08438	-0,369375	-0,04813	0,1	0,0084375	0,0369375	0,0048125		
48	6	0	0	0	5,91	0,77	5,25	0	5	1,37	13,1769	78,3994	5,363325	0,698775	4,764375	0,1	-0,5363325	-0,0698775	-0,4764375		
49	7	0	0	0	0,77	5,25	1,37	0	5	4,42	0,3364	78,3994	0,11165	0,76125	0,19805	0,1	-0,011165	-0,076125	-0,019865		
50	8	0	0	0	5,25	1,37	4,42	0	5	0,26	22,4676	78,3994	6,22125	1,62345	5,2377	0,1	-0,622125	-0,162345	-0,52377		
51	9	0	0	0	1,37	4,42	0,26	0	5	4,21	0,6241	78,3994	0,270575	0,87295	0,05135	0,1	-0,0270575	-0,087295	-0,005135		
52	10	0	0	0	4,42	0,26	4,21	0	5	1,9	9,61	78,3994	3,4255	0,2015	3,26275	0,1	-0,34255	-0,02015	-0,326275		
53																	-0,2589965	0,04051875	-0,241938	Середнє	
54																					
55	1	-0,259	-0,04	-0,2	1,59	5,73	0,48	-0,76	3,186	5,28	4,383872	19,29988	-0,72275	-2,6046428	-0,21819	0,1	0,07227543	0,26046428	0,021818997		59,09952
56	2	-0,259	-0,04	-0,2	5,73	0,48	5,28	-2,78	0,584	1,35	0,587318	19,29988	-0,24133	-0,8902163	-0,22239	0,1	0,02413319	0,002021628	0,022237911		
57	3	-0,259	-0,04	-0,2	0,48	5,28	1,35	-0,66	3,396	5,91	6,3179	19,29988	-0,2706	-2,9766201	-0,76107	0,1	0,02706018	0,297662006	0,076106763		
58	4	-0,259	-0,04	-0,2	5,28	1,35	5,91	-2,85	0,546	0,77	0,050287	19,29988	-0,06109	-0,0156201	-0,06838	0,1	0,00610921	0,001562014	0,006838152		
59	5	-0,259	-0,04	-0,2	1,35	5,91	0,77	-0,78	3,153	5,25	4,39693	19,29988	-0,61114	-2,675439	-0,34858	0,1	0,06111409	0,267543899	0,034857665		
60	6	-0,259	-0,04	-0,2	5,91	0,77	5,25	-2,83	0,556	1,37	0,66232	19,29988	-0,25263	-0,032914	-0,22441	0,1	0,02526253	0,003291396	0,022441335		
61	7	-0,259	-0,04	-0,2	0,77	5,25	1,37	-0,74	3,222	4,42	1,434817	19,29988	-0,20143	-1,3733969	-0,35839	0,1	0,02014315	0,137339692	0,03583912		
62	8	-0,259	-0,04	-0,2	5,25	1,37	4,42	-2,48	0,769	0,26	0,259532	19,29988	0,189959	0,04957013	0,159927	0,1	-0,0189959	-0,004957013	-0,0159927		
63	9	-0,259	-0,04	-0,2	1,37	4,42	0,26	-0,6	3,551	4,21	0,434662	19,29988	-0,20683	-0,6673062	-0,03925	0,1	0,02068347	0,066730622	0,003925331		
64	10	-0,259	-0,04	-0,2	4,42	0,26	4,21	-2,17	1,021	1,9	0,772242	19,29988	-0,35615	-0,0209503	-0,33923	0,1	0,03561544	0,002095026	0,033923299		
65																	0,27340084	1,033735349	0,241995873		

w1+Δw1

Рис. А.5. Таблиці для прогнозування часового ряду. У другій таблиці скориговані синаптичні ваги

4. Створити макрос для розрахунку значень синоптичних ваг, скориставшись інструментарієм MS Excel (Вкладка «Вид», кнопка «Макроси», підпункт меню «Создать макрос»). При створенні макросу слід перекопіювати значення (не формули) другої таблиці в першу.

5. Відредагувати макрос в середовищі MS Excel VBA таким чином, щоб навчання повторювалося 200 разів(200 епох навчання) і зберегти його, рис. А. 6.

```

Нейронні мережі_лабораторні роботи — копия.xlsx - Module2 (Code)
(General) Macro1
Sub Macro1 ()
'
' Macro1 Macro
'
' Keyboard Shortcut: Ctrl+y
'
Dim i As Integer
For i = 0 To 200
Range("B55:D64").Select
Selection.Copy
Range("B43").Select
Selection.PasteSpecial Paste:=xlPasteValues, Operation:=xlNone, SkipBlanks _
:=False, Transpose:=False
Next i
End Sub

```

Рис. А.6. Зразок коду з відредагованим макросом

Джерело: створено автором

6. Провести обчислення активізувавши макрос. Побудувати відповідну стовпчасту гістограму заданих та прогнозованих значень, рис. А.7.

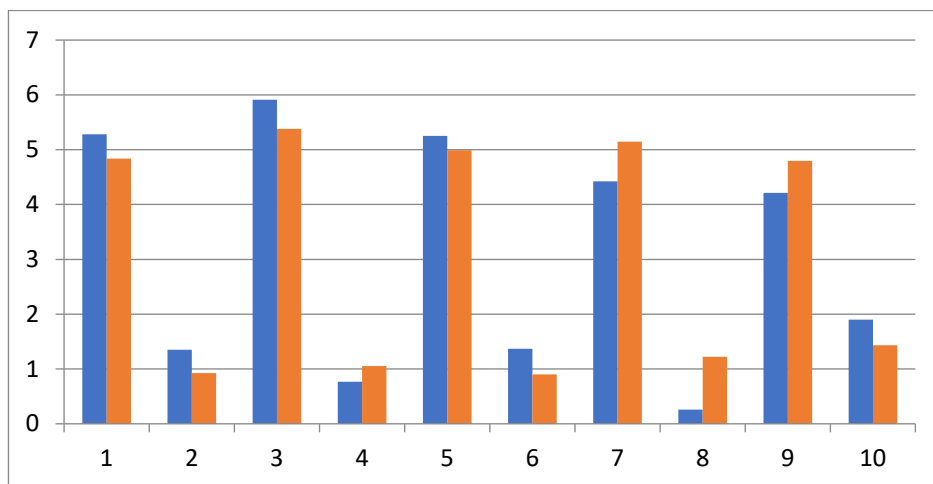


Рис. А.7. Приклад гістограми заданих та прогнозованих значень u_i .
(перші стовпчики пари – задані значення, другі – прогнозовані)

7. Перевірити результати прогнозування розрахувавши три останні значення часового ряду для отриманих ваг і порівнявши їх із заданими.

В таблиці А.2 наведені варіанти значень часових рядів для виконання роботи.

Таблиця А.2

Варіанти значень часових рядів для виконання роботи

Варіант	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15
1	0,17	1,00	1,00	0,42	1,67	5,00	0,42	1,33	5,67	0,33	1,17	1,67	0,08	0,50	1,33
2	1,00	1,17	0,25	2,67	0,83	0,17	5,33	0,83	0,33	4,00	1,67	0,25	3,00	1,67	0,08
3	1,00	0,67	0,67	0,50	1,00	1,17	0,67	1,50	1,17	0,67	0,17	0,67	1,50	1,50	0,83
4	0,08	0,83	2,00	0,08	0,33	2,67	0,42	1,33	5,00	0,42	0,17	4,00	0,33	0,17	5,33
5	0,25	0,83	1,33	0,33	0,83	5,33	0,33	0,50	1,00	0,08	0,33	0,67	0,08	0,67	0,33
6	0,17	1,50	0,50	0,17	1,67	4,50	0,42	1,17	4,00	0,08	0,17	0,50	0,25	0,33	5,00
7	1,25	1,17	3,33	0,75	1,50	3,67	0,42	0,83	2,33	1,25	0,67	2,67	0,92	1,33	0,67
8	0,08	1,17	2,33	0,17	0,83	1,67	0,33	1,50	2,67	0,25	0,33	3,67	0,25	1,67	2,33
9	1,00	1,00	0,17	3,67	1,50	0,42	1,67	0,83	0,17	2,67	1,67	0,42	2,33	0,33	0,17
10	0,25	0,17	2,33	0,08	0,33	3,00	0,17	0,33	4,33	0,25	0,17	2,67	0,42	1,67	0,67
11	1,33	0,67	0,33	4,67	0,17	0,42	1,67	0,83	0,25	1,67	1,17	0,08	4,33	0,83	0,42
12	1,67	1,00	1,50	1,00	0,67	0,67	1,50	1,17	1,67	1,67	1,17	1,17	0,67	1,00	0,83
13	0,42	1,67	1,00	0,33	0,83	2,33	0,25	1,00	0,33	0,42	0,83	1,00	0,42	0,17	1,33
14	0,33	0,33	3,00	0,33	0,50	3,00	0,08	1,50	2,00	0,33	1,67	0,67	0,25	0,17	1,67
15	0,33	0,33	2,00	0,08	0,33	5,00	0,08	1,17	4,00	0,17	0,17	1,50	0,42	0,50	2,50
16	0,83	0,50	1,33	0,17	0,17	1,67	0,17	1,33	3,67	0,33	0,33	4,67	1,00	0,67	1,67
17	0,25	0,67	4,00	0,33	0,67	1,00	0,42	0,17	3,00	0,17	1,50	3,00	0,08	0,17	0,33

18	6,00	1,50	0,17	2,00	0,33	0,33	5,67	1,00	0,25	3,67	0,33	0,42	2,00	0,67	0,25
19	0,17	1,33	5,33	0,25	1,67	0,67	0,33	0,83	5,67	0,33	0,67	5,00	0,08	0,50	2,00
20	1,33	0,33	0,17	5,33	0,33	0,33	3,67	0,17	0,17	2,00	1,17	0,33	3,67	0,17	0,08
21	1,17	0,83	0,33	1,67	1,50	0,83	0,67	1,67	0,33	1,17	0,17	1,00	0,17	1,17	0,17
22	0,25	1,17	4,67	0,08	0,17	2,00	0,17	1,17	1,67	0,33	0,83	5,33	0,42	1,67	5,33
23	0,08	1,33	0,67	0,08	0,83	1,33	0,17	0,67	1,67	0,17	1,67	4,33	0,17	1,67	4,67
24	0,42	0,17	2,50	0,08	1,33	5,00	0,33	0,67	2,50	0,17	0,50	2,00	0,42	0,83	2,00
25	0,50	0,33	5,67	0,25	0,33	3,00	1,17	0,67	4,00	1,00	0,17	3,67	0,58	0,50	3,00
26	0,33	0,67	1,33	0,17	1,00	3,00	0,33	0,50	3,67	0,42	0,33	2,33	0,25	0,83	3,33
27	2,67	1,33	0,42	1,67	1,50	0,42	2,67	1,33	0,33	3,00	1,17	0,33	2,67	0,17	0,42
28	0,08	0,50	1,00	0,08	1,17	2,33	0,08	1,33	4,33	0,25	0,17	1,00	0,33	1,00	5,00
29	3,67	0,50	0,42	4,00	1,17	0,17	1,33	1,50	0,17	2,67	1,50	0,42	4,00	0,33	0,17
30	0,67	0,33	0,50	0,67	0,17	1,17	0,50	1,50	1,33	1,17	1,33	0,50	0,17	0,83	1,50
31	0,25	1,33	3,67	0,42	1,50	0,67	0,08	1,67	4,67	0,08	1,00	0,67	0,25	0,83	0,67

Лабораторна робота №5

Дослідження нейронної мережі Хопфілда

Мета роботи: Вивчити особливості роботи нейронної мережі Хопфілда з використанням комп'ютерних засобів

Зміст завдання. Вивчити роботу нейронної мережі Хопфілда, здійснюючи розпізнавання символів комп'ютерною програмою. Для розпізнавання символів завантажити комп'ютерну програму Норф.exe, рис. А.8.

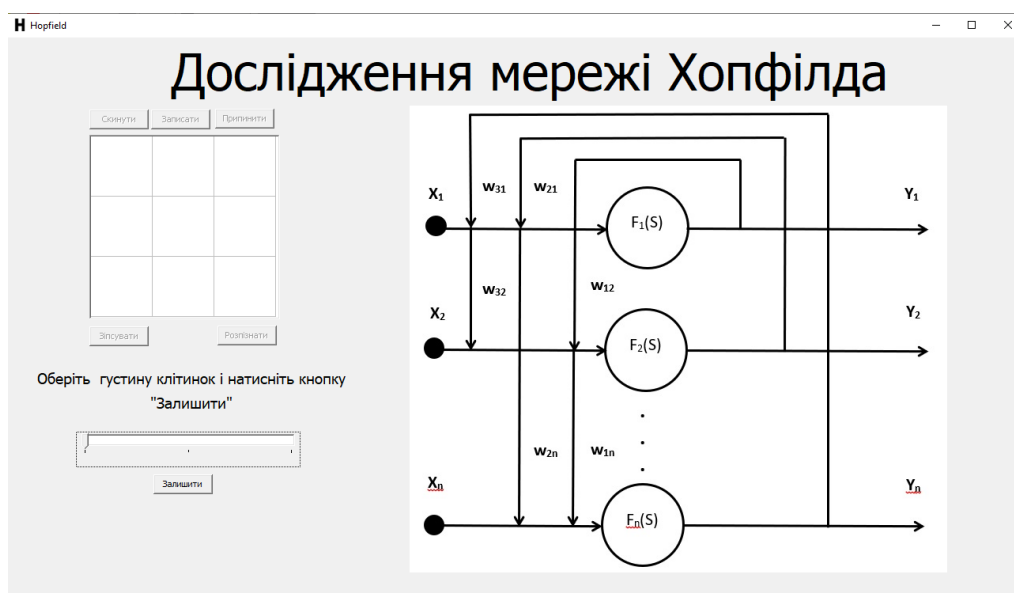


Рис. А.8. Зовнішній вигляд комп'ютерної програми Норф.exe

За допомогою мишки обрати густину сітки у вікні для введення символів (початков гусина 3X3 елементи) і натиснути кнопку «Залишити». Після такого натискання активізуються кнопки «Записати» і «Припинити». Використовуючи ліву кнопку мишки ввести символ у вікно для введення символів і натиснути кнопку «Записати». Виникне повідомлення про запис символу, рис. А.9.

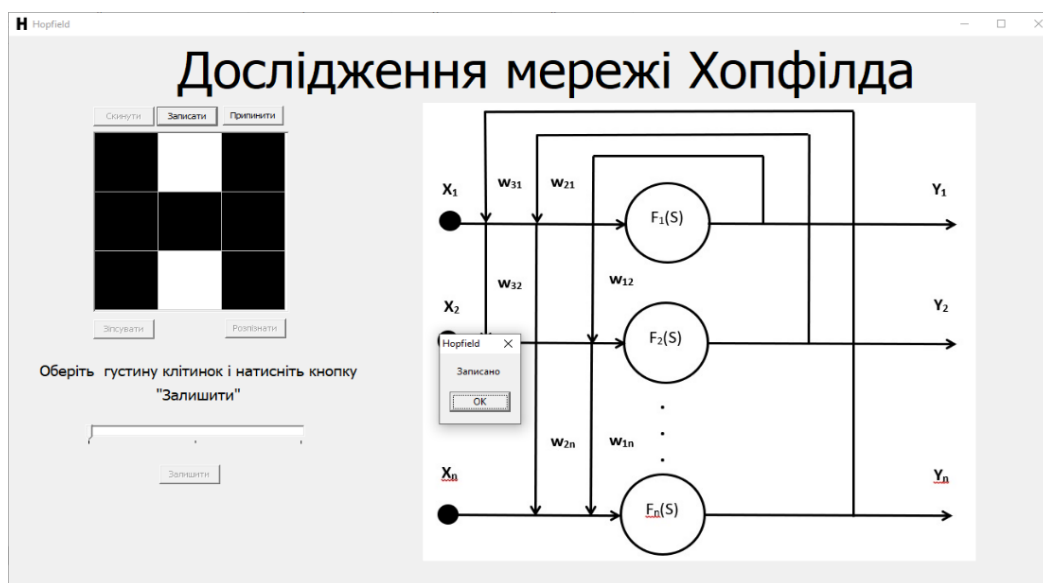


Рис. А.9. Зовнішній вигляд програми після запису символу

Використовуючи мишку записати ще декілька символів. Ліва кнопка мишки використовується для заповнення кожної літинки чорним кольором, права для заповнення білим кольором. Для припинення запису символів, натиснути кнопку «Припинити». Таке натинення блокує кнопку «Записати» і активізує кнопки «Скинути» «Зіпсувати» і «Розпізнати». Зіпсувати один із введених символів у вікні для введення символів, використовуючи мишку і натиснути кнопку «Зіпсувати», рис. GHB.

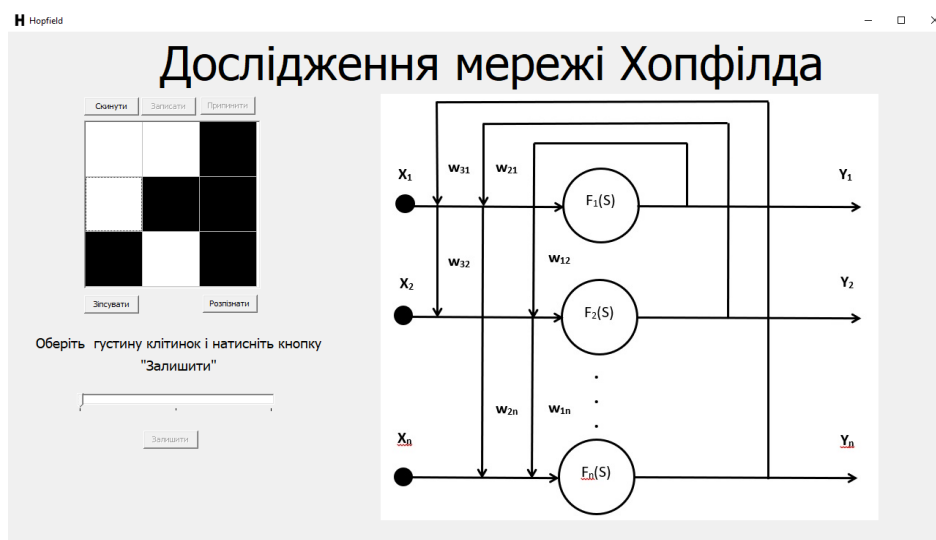


Рис. А.10. Зовнішній вигляд комп'ютерної програми Норф.ехе із зіпсованим СИМВОЛОМ

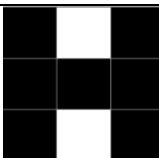
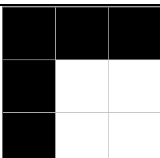
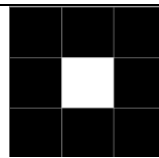
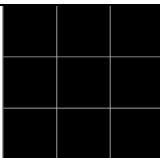
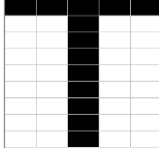
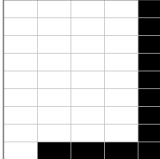
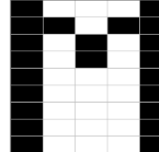
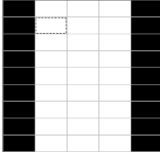
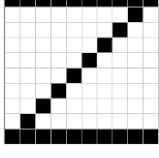
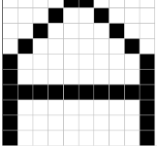
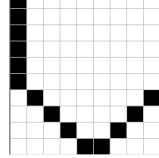
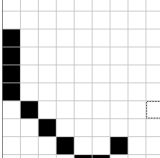
Натиснути кнопку «Розпізнати». Після цього натиснення у вікні для введення символів виникне один із введених розпізнаний незіпсований символ. Якщо котрийсь із символів був введений некоректно можна використати кнопку «Скинути». У такому випадку доведеться вводити всі символи спочатку. Програмою передбачено введення до 20 символів. В табл. А.3 наведені рекомендовані символи для розпізнавання для різної густини сітки.

Під час виконання роботи необхідно.

1. Провести тестування програми і навчитися користуватися нею.
2. Здійснити розпізнавання символів, використовуючи різну густину сітки вікна для введення символів 3X3, 5X9, 10X10.
3. Визначити умови за яких розпізнавання символів за допомогою мережі Хопфілда є неможливим.

Таблиця А.3

Рекомендовані символи для розпізнавання для різної густини сітки

№	Густина сітки	Номери символів			Зіпсований символ
		№1	№2	№3	
1	3X3				
2	5X9				
3	10X10				

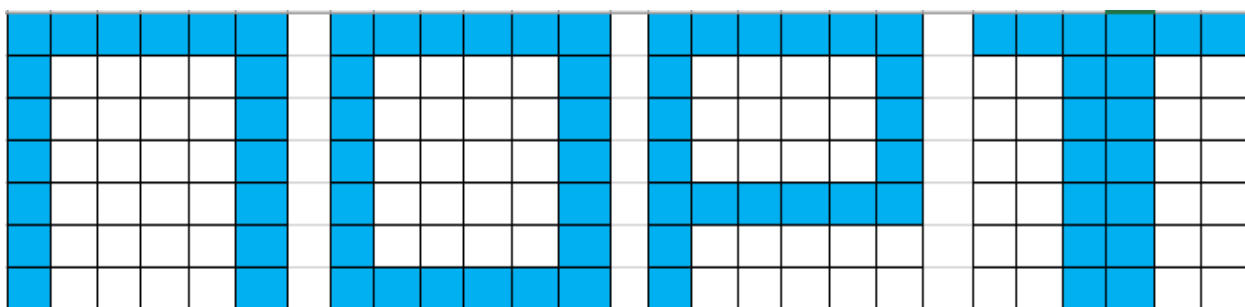
Джерело: створено автором

Лабораторна робота №6

Розпізнавання символів за допомогою комп'ютерної мережі зі зворотним розповсюдженням помилки

Мета роботи: Розпізнавання символів за допомогою нейронної мережі зі зворотним розповсюдженням помилки

Зміст завдання. Розпізнати літери за допомогою нейронної мережі



Кожній літері привласнимо певний код, який нейронна мережа повинна видати як вихідний вектор, при розпізнаванні відповідної літери. Коди літер:

П - 0 0

О - 0 1

Р - 1 0

Т - 1 1

Для роботи з fannExplorer створимо файли з даними для навчання та тестування. Файл для навчання test1.train буде містити наступні дані: 4 - чотири образи для навчання 36 - елементів у вхідному векторі (пікселі літери) 2 - елементи у вихідному векторі (дворозрядний код літери) Зміст файлу test1.train:

4	36	2			
1	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
0	0				
1	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	1	1	1	1	1
0	1				
1	1	1	1	1	1
1	0	0	0	0	1
1	0	0	0	0	1
1	0	0	0	0	1
1	1	1	1	1	1
1	0	0	0	0	0
1	0	0	0	0	0
1	0				
1	1	1	1	1	1
0	0	1	1	0	0
0	0	1	1	0	0
0	0	1	1	0	0
0	0	1	1	0	0
0	0	1	1	0	0
0	0	1	1	0	0
1	1				

Файл для тестування test1.test буде містити ті ж самі данні, але дещо викривлені та без вихідних векторів, щоб перевірити здатність мережі до розпізнавання.

```

4 36 2
0 1 1 1 1 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
0 0

```

```

0 1 1 1 1 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 1 1 1 1 0
0 1

```

```

1 1 1 1 1 1
1 0 0 0 0 1
1 0 0 0 0 1
1 0 0 0 0 1
1 1 1 1 1 1
1 0 0 0 0 0
1 0 0 0 0 0
1 0

```

```

1 1 1 1 1 1
0 0 1 1 0 0
0 0 1 1 0 0
0 0 1 1 0 0
0 0 1 1 0 0
0 0 1 1 0 0
0 0 1 1 0 0
1 1

```

Як видно з наведених тестових даних, в перших двох літерах інвертовано декілька пікселів (один в літері П, та два в - О). Так як формат файлу не дозволяє повністю прибрати вихідні вектори (програма видасть помилку) усі вихідні дані для всіх образів вказані рівними нулю. Файли test1.train та test1.test необхідно

розмістити у наступній папці: ...
 fann\fannExplorer21\fann\fannSoap\fannKernel\Release\net\

Тепер у програмі fannExplorer необхідно створити нейронну мережу, що буде складатися з трьох шарів: 1 шар - вхідний, 36 нейронів, по одному на піксель зображення. 2 шар - прихований, 36 нейронів. 3 шар - вихідний, два нейрони, для отримання дворозрядного коду літери. Створити мережу можна через пункт меню File->New Neural Network..., після його натиснення з'явиться діалогове вікно, у якому слід вказати необхідні параметри мережі. Потім у вкладці Algorithm виберемо параметри нейронів, такі як вид функції активації, алгоритму навчання, тощо. В наведеному прикладі обрана лінійна функція активації та змінено функцію визначення похибки навчання з Tanh на Linear.

На наступному кроці слід запустити процес навчання. Після процесу навчання можна перейти до тестування нейронної мережі. На рисунку А.11 зображене вікно програми fannExplorer із зображенням нейронної мережі після перерахованих вище дій.

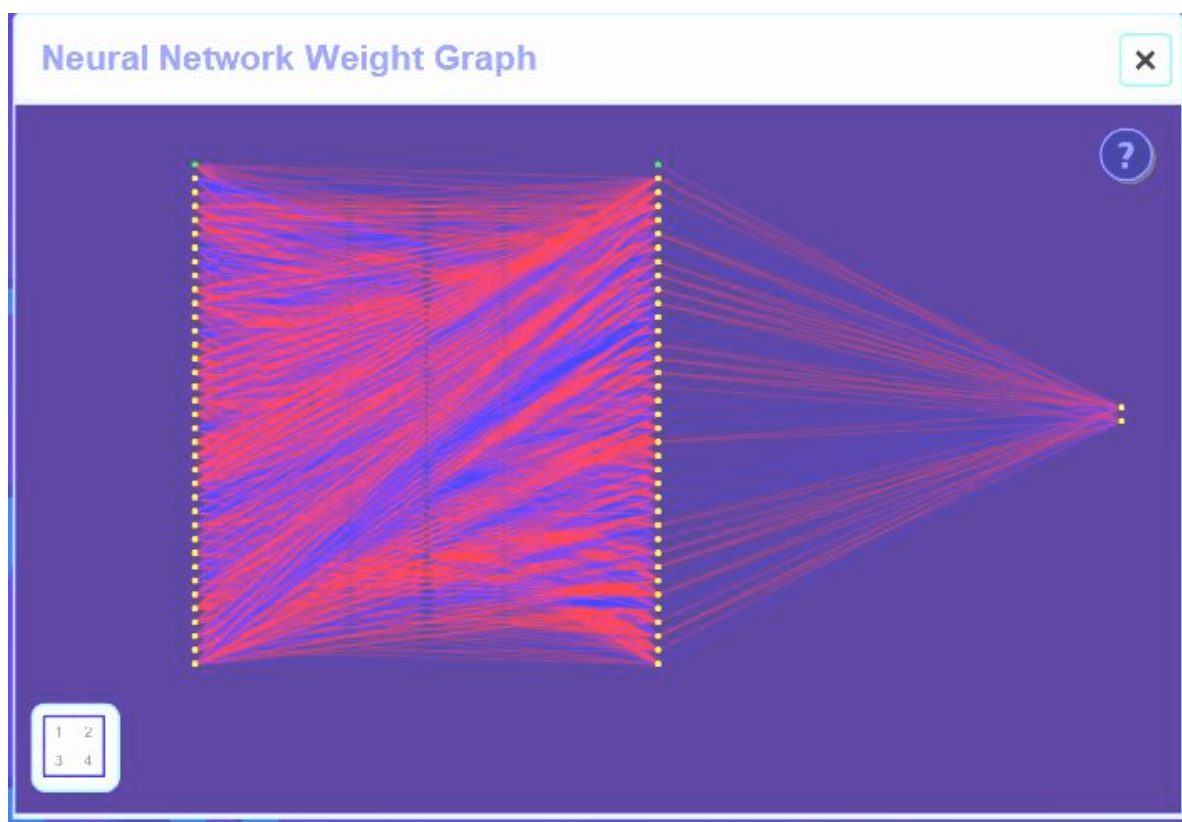


Рис. А.11. Зовнішній вигляд вікна програми fannExplorer із зображенням мережі

На вкладці testing можна побачити результати розпізнавання. У полі Select output neuron for plot необхідно вибрати номер вихідного нейрону. Спочатку вибираємо номер 1. На графіку вертикальна вісь - значення виходу нейрону, горизонтальна вісь - номер образу з тестової вибірки.

Зеленим відмічені дані вказані у файлі, вони скрізь рівні нулю так як і було вказано. Червоним вказані дані одержані в процесі розпізнавання нейронною мережею. Значення 1-го нейрону (див. рис 3.44) – 1 образ - вихід першого нейрона 0, 2 образ - вихід першого нейрона 0, 3 образ - вихід першого нейрона 1, 4 образ - вихід першого нейрона 1.

Вибираємо другий нейрон у полі Select output neuron for plot. Значення 2-го нейрону (див. рис. 3.55) – 1 образ - вихід першого нейрона 0, 2 образ - вихід першого нейрона 1, 3 образ - вихід першого нейрона 0, 4 образ - вихід першого нейрона 1.

Очевидно, що розпізнавання виконано коректно, незважаючи на додані помилки.

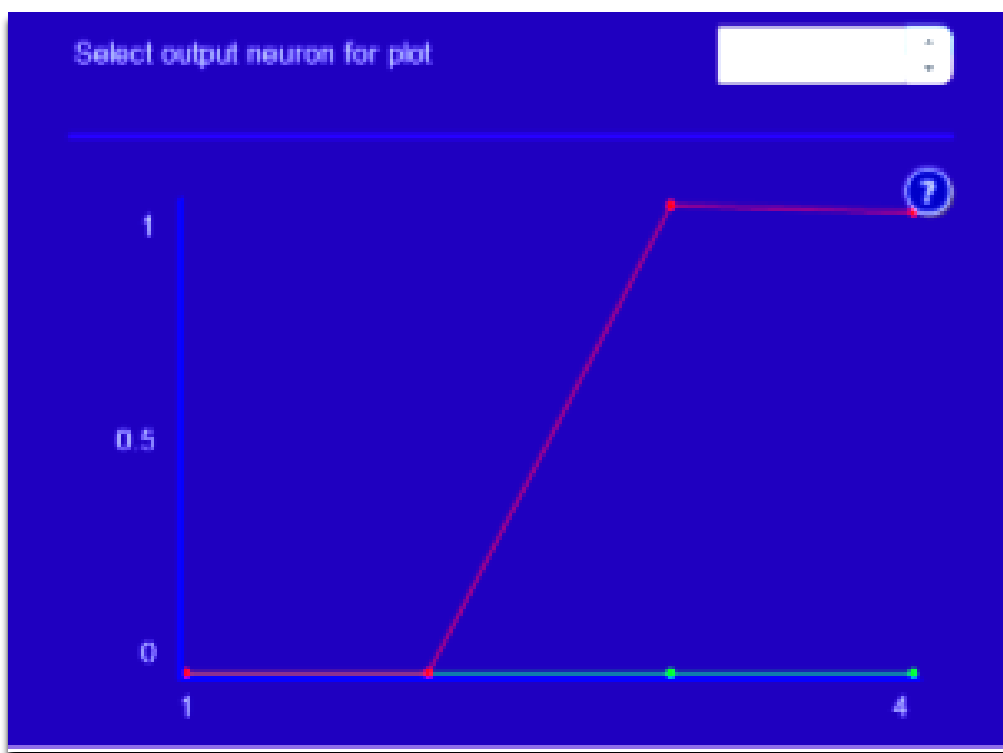


Рис. А.12. Значення першого нейрону

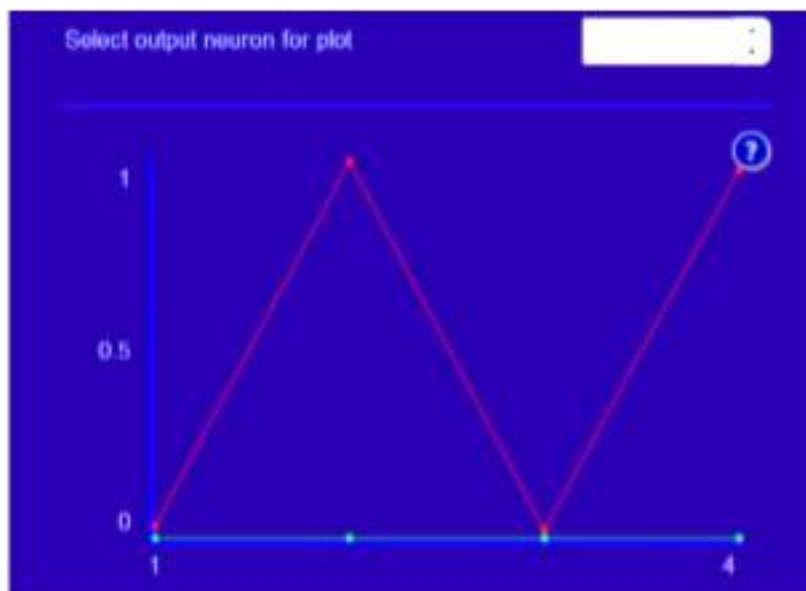


Рис. А.13. Значення другого нейрону

За аналогією виконати розпізнавання літер в словах КРОК, СМОГ, КОРТ, БОРТ, ЛІФТ, додаючи при цьому 3,4,5,6,7 помилок. Визначити за якої кількості помилок розпізнавання виконується некоректно. Зробити висновки.

Лабораторна робота №7

Використання мережі Кохонена для здійснення кластерного аналізу

Мета роботи: Вироблення практичних навичок в роботі з нейронною мережею Кохонена

Зміст завдання: Оцінити інвестиційну привабливість країн, використовуючи одновимірну мережу Кохонена, за даними, наведеними в таблиці. Для обчислень використати MS Excel.

Таблиця А.4

Дані для розрахунків

№	назви країн	Показники оцінювання			
		фінансова система	динаміка економічного розвитку	стабільність політичної системи	наявність необхідних кадрів
1	країна 1	3	3	2	3
2	країна 2	3	2	4	4
3	країна 3	5	1	2	2
4	країна 4	2	1	2	1
5	країна 5	2	3	1	1
6	країна 6	2	3	1	1
7	країна 7	10	10	10	10

8	країна 8	10	10	10	2
9	країна 9	1	1	1	1
10	країна 10	3	3	2	3
11	країна 11	3	3	2	3
12	країна 12	2	1	2	3
13	країна 13	2	1	3	3
14	країна 14	2	1	1	3
15	країна 15	1	1	1	1
16	країна 16	3	1	1	3
17	країна 17	1	1	1	2
18	країна 18	1	1	1	1
19	країна 19	4	3	3	4
20	країна 20	1	1	1	1
21	країна 21	2	3	3	3
22	країна 22	4	4	3	4
23	країна 23	4	3	2	2
24	країна 24	4	3	2	2
25	країна 25	1	1	3	1
26	країна 26	1	1	1	1
27	країна 27	1	1	1	1
28	країна 28	1	1	1	1
29	країна 29	2	4	1	2
30	країна 30	1	1	1	1

Лабораторна робота №8

Навчання інтелектуальних агентів за допомогою алгоритму Q-навчання

Мета роботи: Навчитися використовувати алгоритм Q-навчання для навчання інтелектуальних агентів

Зміст завдання. Створити Q-матрицю для заданих варіантів та реалізувати програмне забезпечення для ілюстрації роботи алгоритму Q-навчання

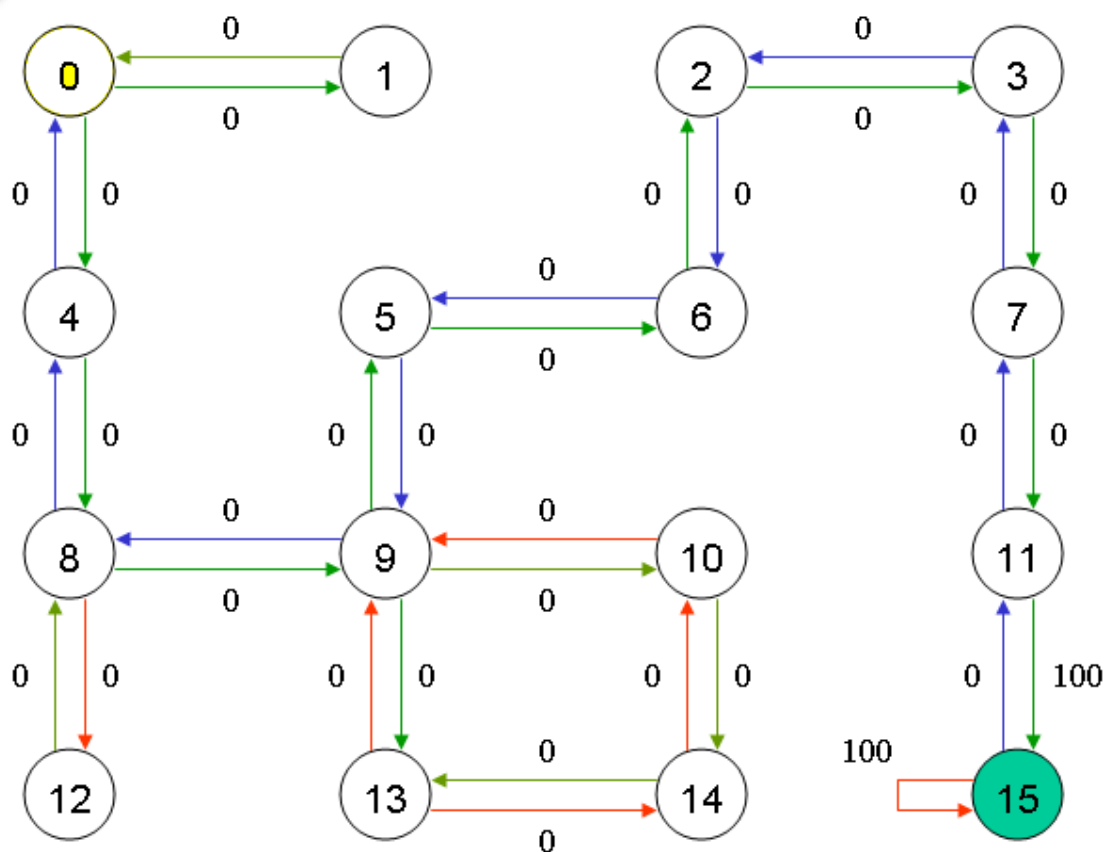


Рис. А.14 – Варіант 1

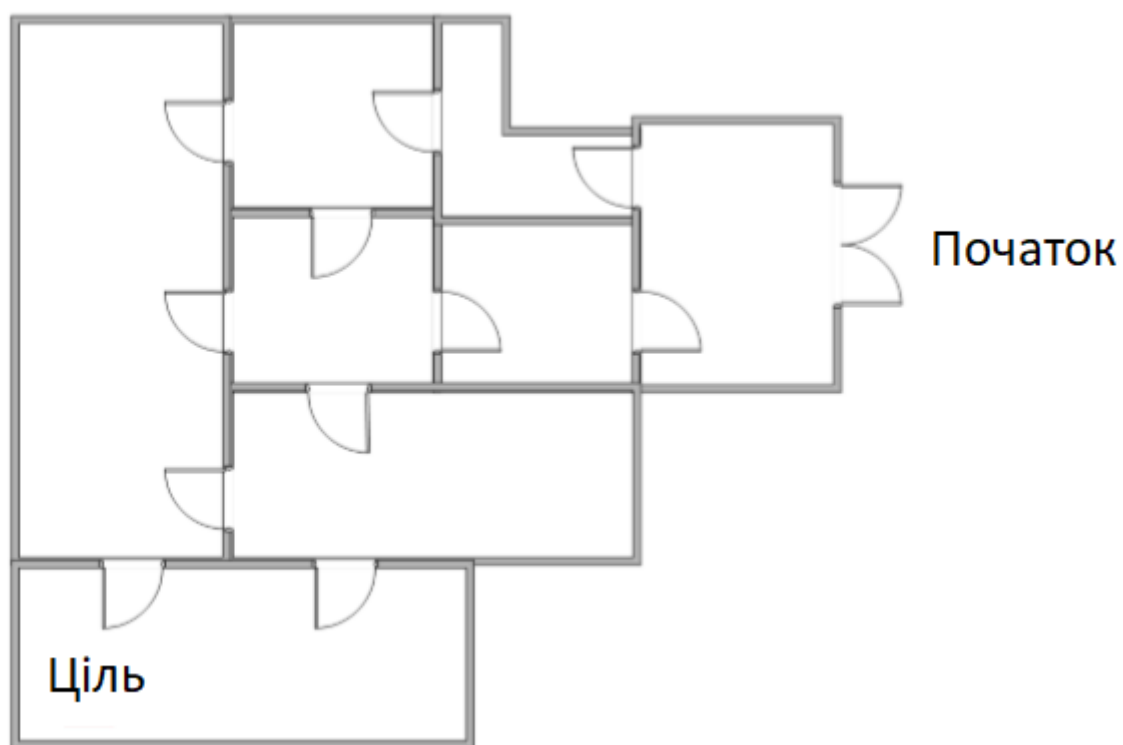


Рис. А.15. Варіант 2

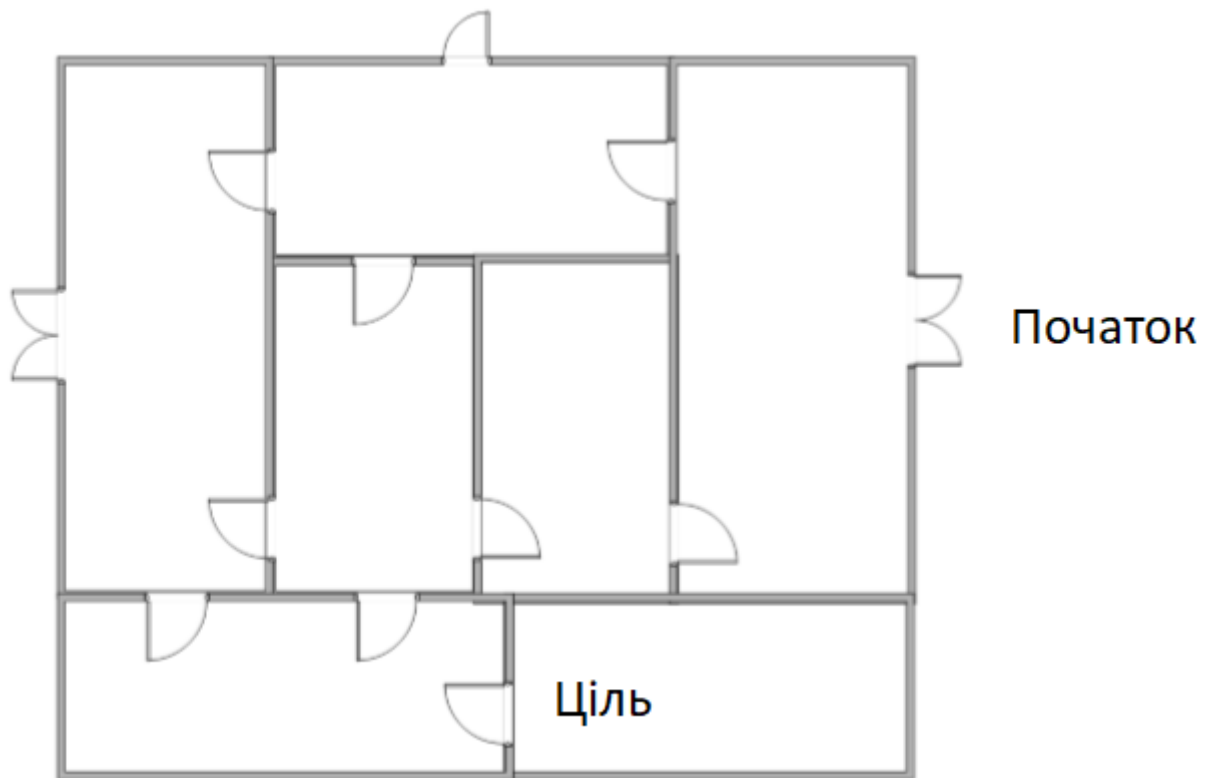


Рис. А.16. Варіант 3

Лабораторна робота №9

Вирішення задачі комівояжера з використанням мурашиного алгоритму

Мета роботи: Виробити навички роботи з ройовими алгоритмами

Зміст завдання. Використовуючи MS Excel розрахувати значення відстаней заданої схеми маршрутів (рис. А.17) у відповідності з варіантом, заданим у табл. А.5 для п'яти мурах-розвідників.

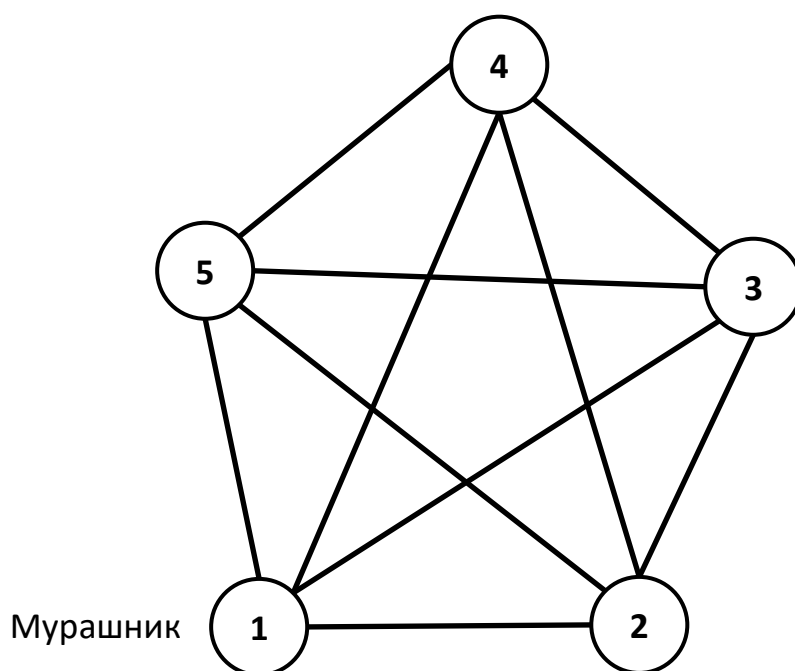


Рис. А.17. Схема маршрутів для виконання роботи

Таблиця А.5

Варіанти для розрахунків

№	Маршрути	Довжина	Інтенсивність запаху феромонів
	m	l	τ
1	1-2	22	3
2	1-3	77	2
3	1-4	81	3
4	1-5	51	2
5	2-3	32	2
6	2-4	30	2
7	2-5	87	3
8	3-4	34	1
9	3-5	24	1
10	4-5	13	2

№ варіанта	p	q	d
Варіант 1	0,78	0,22	45
Варіант 2	0,21	0,79	56
Варіант 3	0,26	0,74	48
Варіант 4	0,53	0,47	27
Варіант 5	0,35	0,65	42
Варіант 6	0,95	0,05	47
Варіант 7	0,59	0,41	40
Варіант 8	0,59	0,41	42
Варіант 9	0,17	0,83	36
Варіант 10	0,81	0,19	39

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Кононюк А.Ю. Нейроні мережі і генетичні алгоритми: Науково-практичне видання. – Київ: Корнійчук, 2008. – 446 с.
2. Mitchell M. An Introduction to Genetic Algorithms. Cambridge, MA: The MIT Press, 1996.
3. Букатова И.Л. Эволюционное моделирование и его приложения. — Москва: Наука, 1979. — 231 с.
4. Ситник В.Ф. Система підтримки прийняття рішень: Навч. посібник. — Київ: КНЕУ, 2004. — 614 с.
5. Глибовець М.М., Гулаєва Н.М. Еволюційні алгоритми: підручник. — Київ.: НаУКМА, 2013. — 828 с.
6. Глибовець, М.М. Гібридний генетичний алгоритм вирішення задачі оптимізації структури інтегральної схеми [Текст] / М.М. Глибовець, С.С. Гороховський, О.В. Краткова // Інженерія програмного забезпечення. – 2011. – № 1. – С. 68-74.
7. А.Д. Кожуховський, О.О. Намофілова Застосування генетичних алгоритмів у задачі про укладання ранця // Автоматизовані системи управління і прилади автоматики: електрон. наук. фахове вид. 2015. URL: <https://cyberleninka.ru/article/n/zastosuvannya-genetichnih-algoritmiv-u-zadachi-pro-ukladannya-rantsya> (дата звернення: 06.12.2019).
8. І.В. Калініна, О.І. Лісовиченко Використання генетичних алгоритмів в задачах оптимізації // Міжвідомчий науково-технічний збірник. 2015. – № 1(26). URL: <https://ela.kpi.ua/handle/123456789/16458> (дата звернення: 06.12.2019).
9. О.Р. Овчіннікова Використання генетичних алгоритмів в моделюванні міграційних процесів // Соц.-ек.проблеми сучас.періоду України, 2013, Вип. 3(101) URL: [http://ird.gov.ua/sep/sep20133\(101\)/sep20133\(101\)_458](http://ird.gov.ua/sep/sep20133(101)/sep20133(101)_458) OvchynnikovaOR.pdf (дата звернення: 06.12.2019).
10. Штучний нейрон // Вікіпедія – Вільна енциклопедія. URL: https://uk.wikipedia.org/wiki/Штучний_нейрон. (дата звернення: 06.12.2019).

11. Akshay Chandra Lagandula Perceptron: The Artificial Neuron (An Essential Upgrade To The McCulloch-Pitts Neuron) // URL: <https://towardsdatascience.com/perceptron-the-artificial-neuron-4d8c70d5cc8d>

12. Хайкин Саймон Нейронные сети: полный курс, 2 е издание.: Пер с англ. – Москва: Издательский дом «Вильямс», 2006. – 1104 с.

13. Комп'ютерні системи штучного інтелекту. Методичні вказівки до виконання лабораторних робіт студентами денної та заочної форми навчання спеціальностей 123 "Комп'ютерна інженерія", 122 "Комп'ютерні науки та інформаційні технології" / Укл.: Є.В. Мелешко – Кіровоград: КНТУ, 2016. – С.8-13.

14. Соловьев Н.В. Распознавание образов в системах искусственного интеллекта методические указания по выполнению лабораторных работ // Санкт-Петербургский Гос. университет аэрокосмического приборостроения 2011. – С. 19 // URL: <https://studfiles.net/preview/2820578/page:3/> (дата звернення: 06.12.2019).

15. Надригайло Т.Ж., Молчанова К.А. Аналіз нейронних алгоритмів // Математичне моделювання: електрон. наук. фахове вид. 2015. URL: <http://www.dstu.dp.ua/Portal/Data/74/68/13-st13.pdf> (дата звернення: 06.12.2019).

16. Principles of training multi-layer neural network using backpropagation // URL: http://galaxy.agh.edu.pl/~vlasi/AI/backp_t_en/backprop.html (дата звернення: 06.12.2019).

17. Адаменко В.О., Мірських Г.О. Штучні нейронні мережі в задачах реалізації матеріальних об'єктів частина 2. Особливості проектування та застосування // Вісник Національного технічного університету України "КПІ" 213 Серія –Радіотехніка. Радіоапаратобудування. – 2012. – No48. – С. 213-221.

18. Wasserman P. D. Experiments in translating Chinese characters using backpropagation. Proceedings of the Thirty-Third IEEE Computer Society International Conference.. — Washington: D. C.: Computer Society Press of the IEEE, 1988.

19. Нейронные сети Кохонена // URL: <https://neuronus.com/theory/nn/955-nejronnye-seti-kokhonena.html> (дата звернення: 06.12.2019).

20. Т.В. Киприч, В.И. Дубровин Анализ самоорганизующихся карт Кохонена по критериям регулярности и точности аппроксимации // Радіоелектроніка, інформатика, управління: електрон. наук. фахове вид. 2007. URL: <https://cyberleninka.ru/article/v/analiz-modifikatsiy-samoorganizuyuschih-sya-kart-kohonena-po-kriteriyam-regulyarnosti-i-tochnosti-aproksimatsii> (дата звернення: 06.12.2019).
21. ПРАКТИКУМ [Р.120] Карты Кохонена в Deductor Studio // URL: <https://docplayer.ru/27573133-Praktikum-p-120-karty-kohonena-v-deductor-studio.html> (дата звернення: 06.12.2019).
22. Субботін С.О. Нейронні мережі: навч. посібник / С.О. Субботін, А.О. Олійник; за ред. С.О. Субботіна. – Запоріжжя : ЗНТУ, 2014. – 132 с.
23. Тимошук П.В. Штучні нейронні мережі Навчальний посібник / Львів: Видавництво Львівської Політехніки, 2011. – 444 с.
24. Ясенев В.Н. Автоматизированные информационные системы в экономике: Учебно-методическое пособие. – Н. Новгород, 2007. – С. 56.
25. Інтелектуальний агент // Вікіпедія – Вільна енциклопедія. URL: https://uk.wikipedia.org/wiki/Інтелектуальний_агент (дата звернення: 06.12.2019).
26. Властивості інтелектуальних агентів //Студопедія. URL: https://studopedia.com.ua/1_7219_vlastivosti-intelektualnih-agentiv.html (дата звернення: 06.12.2019).
27. Колективний інтелект // Вікіпедія – Вільна енциклопедія. URL: https://uk.wikipedia.org/wiki/Колективний_інтелект (дата звернення: 06.12.2019).
28. Джонс М.Т. Программирование искусственного интеллекта в приложениях / Тим Джонс; Пер. С англ. Осипов А.И. – Москва: ДМК Пресс, 2004. – 312 с:
29. В. Юзевич, Н. Крап Моделювання туристичних потоків з використанням мурашиних алгоритмів // Комп'ютерні науки та інформаційні технології. – 2011. – №710: електрон. наук. фахове вид. URL: <http://ena.lp.edu.ua:8080/handle/ntb/12116> (дата звернення: 06.12.2019).
30. С. Д. Штовба, О. М. Рудий Мурашині алгоритми оптимізації // Вісник Вінницького політехнічного інституту, № 4, с. 62-69: електрон. наук. фахове вид.

URL: <https://visnyk.vntu.edu.ua/index.php/visnyk/article/view/78> (дата звернення: 06.12.2019).

31. Г.В. Худов, І.А. Таран Використання мультиагентного (мурашиного) алгоритму для розпізнавання елементів замислу повітряного противника // Системи озброєння і військова техніка. — 2015. — № 3(43). — С. 179-185. URL: <http://www.hups.mil.gov.ua/periodic-app/article/15070> (дата звернення: 06.12.2019).

32. Marco Dorigo and Gianni Di Caro Ant Algorithms for Discrete Optimization IRIDIA, Université Libre de Bruxelles // URL: http://people.idsia.ch/~luca/ij_23-alife99.pdf.