НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

«На правах рукопису»

УДК 004.032.26(043.3)

«До захисту допущено»
В.о. завідувача кафедри
_____ М.В.Грайворонський
“___”_____2019 р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності: 113 Прикладна математика

на тему: «Модель розвитку простих організмів з використанням генетичних алгоритмів та глибинного навчання»

Виконав (-ла): студент (-ка) _6__ курсу, групи __ФІ-71мн_____
(шифр групи)

Миколович Юрій Васильович_____
(підпис)

Науковий керівник: Олександр Арсенійович Орєхов, кандидат фізико-математичних наук_____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант _____ _____ _____
(назва розділу) (науковий ступінь, вчене звання, , прізвище, ініціали) (підпис)

Рецензент_____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ
Кафедра інформаційної безпеки

Рівень вищої освіти – другий (магістерський) за освітньо-науковою програмою
Спеціальність (спеціалізація) – 125 Кібербезпека («Системи, технології та математичні методи кібербезпеки»)

ЗАТВЕРДЖУЮ
В.о. завідувача кафедри
_____ М.В.Грайворонський
(підпис)
«___»_____2019 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Миколовичу Юрію Васильовичу

1. Тема дисертації: «Модель розвитку простих організмів з використанням генетичних алгоритмів та глибинного навчання»,
науковий керівник дисертації: Олександр Арсенійович Орєхов, кандидат фізико-математичних наук,
затверджені наказом по університету від «02» квітня 2019 р. № 1023-с

2. Термін подання студентом дисертації 06.05.2019 р.

3. Об’єкт дослідження _____
_____

4. Предмет дослідження _____
_____
_____
_____

5. Перелік завдань, які потрібно розробити _____
_____
_____
_____
_____
_____

6. Орієнтовний перелік ілюстративного матеріалу _____
_____
_____

7. Орієнтовний перелік публікацій _____
_____
_____

8. Консультанти розділів дисертації*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка

Студент _____ (підпис) _____ (ініціали, прізвище)

Науковий керівник дисертації _____ (підпис) _____ (ініціали, прізвище)

* Консультантом не може бути зазначено наукового керівника магістерської дисертації.

РЕФЕРАТ

Обсяг роботи: 71 сторінок, 18 ілюстрацій, 3 таблиці, 15 джерел літератури.

Об’єктом дослідження кваліфікаційної роботи є модель поведінки простих організмів .

Предметом дослідження є модифікації поведінки організмів при внесенні змін в базовий алгоритм та зміні набору параметрів .

Метою магістерської дисертації є побудова та дослідження моделі мікроеволюційної поведінки групи простих організмів, що описуються нейронною мережею.

У роботі було виконано теоретичний огляд предметної області, розглянуто існуючі методи реалізації генетичних алгоритмів. Розглянуті основні положення та визначення, що відносять до сфери нейронних мереж. На основі проведеного аналізу була побудована модель мікроеволюційної поведінки групи простих організмів. Та модифікації поведінки організмів при внесенні змін в базовий алгоритм та зміні набору параметрів.

Генетичні алгоритми, еволюційні алгоритми, нейронні мережі, нейрон, прості організми, елітизм, мутація, схрещування.

ABSTRACT

The term paper includes 71 pages, 18 illustrations, 3 table, 15 sources of literature.

The objects of study: behavioral model of simple organisms.

The subject of study: modifications of the behavior of organisms when making changes to the basic algorithm and changing the set of parameters.

The goal of the work: is construction and research of the model of microevolutionary behavior of a group of simple organisms described by a neural network.

The paper contains the theoretical review of the subject area, existing methods for the implementation of genetic algorithms are considered. The main provisions and definitions concerning the sphere of neural networks are considered. On the basis of the analysis, a model of microevolutionary behavior of a group of simple organisms was constructed. And modifications of the behavior of organisms when making changes to the basic algorithm and change the set of parameters.

Genetic algorithms, evolutional algorithms, neural networks, neuron, simpl organisms, elitizm, mutation, crossover.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ГА – Генетичні алгоритми;

ЕА – Еволюційні алгоритми;

AVG – середнє значення пристосованості;

ВСТУП

Еволюція - це основний механізм, за допомогою якого життя адаптується для того, щоб виживати та розвиватися. Це біологічне явище надихає такі успішні оптимізаційні методи, як генетичні алгоритми. Як відомо еволюція спирається на поняття випадкової мутації та випадкової генетичної рекомбінації. Ці поняття стають основою для генетичного алгоритму.

Еволюція і навчання - це дві структури оптимізації, які в живих системах працюють в різних масштабах та з різними перевагами. Відповідні комбінації двох забезпечують взаємодоповнюваність і є важливою частиною успіху живих систем.

Генетичні алгоритми та нейронні мережі стануть основую для реалізації моделі мікроеволюційної поведінки групи простих організмів. Поведінка та розвиток організмів повинні відповідати реальним прикладам. В даній моделі будуть реалізовані такі біологічні поняття, як розвиток, схрещування, селекція.

Також на основі реалізованої моделі буде розглянути залежність поведінки членів популяції від набору параметрів. А саме розмір популяції та еліт в самій популяції та складності організмів.

Актуальність роботи.

Нейронні мережі на даний момент використовують у широкому спектрі задач. Використання генетичних алгоритмів для оптимізації нейронних мереж дозволяє отримати позитивний виграш в часових ресурсах.

Дослідження на тлі широкого спектру існуючих моделей дозволить виявити найкращі рішення, та отримати нові підходи до розв’язку задачі.

Об’єкт дослідження

Об’єктом дослідження кваліфікаційної роботи є модель поведінки простих організмів .

Предмет дослідження

Дослідження модифікації поведінки організмів при внесенні змін в базовий алгоритм та зміні набору параметрів **.**

Мета і завдання дослідження

Метою магістерської дисертації є побудова та дослідження моделі мікроеволюційної поведінки групи простих організмів, що описуються нейронною мережею.

Методи дослідження

Методи дослідження, які було використано у роботі – спостереження, порівняння, аналіз та моделювання. Було опрацьовано теоретичний матеріал та публікації за тематикою дослідження, проаналізовано предметну область дослідження, проведено аналіз роботи генетичних алгоритмів , огляд існуючих та використовуваних в роботі нейронних мереж, шляхом моделювання мікроеволюційної поведінки групи простих організмів та порівняння отриманих результатів були виявлені оптимальні значення параметрів моделі.

Завдання дослідження

Для досягнення поставленої мети було поставлено наступні завдання дослідження:

1) розглянуті основні положення та визначення, що відносять до сфери нейронних мереж;

2) проаналізувати предметну область, розглянути існуючі методи реалізації генетичних алгоритмів;

3) реалізувати модель мікроеволюційної поведінки групи простих організмів;

4) проаналізувати модифікації поведінки організмів при внесенні змін в базовий алгоритм та зміні набору параметрів.

Наукова новизна одержаних результатів

Наукова новизна магістерської дисертації полягає комбінації методів нейронних мереж та генетичних алгоритмів для їх оптимізації.

Запропонована модель підходить для моделювання простих живих організмів та може бути застосована у реальних проектах.

1 ЕВОЛЮЦІЙНІ ТА ГЕНЕТИЧНІ АЛГОРИТМИ

Еволюційні алгоритми — напрям в штучному інтелекті (розділ еволюційного моделювання), що використовує і моделює біологічну еволюцію [1].

Еволюційні алгоритми найчастіше використовують як пошукові алгоритми, які змінюють набір розв'язків через повторювані трансформації. Факт того, що множину розв'язків одночасно змінюється якісно відрізняє еволюційні алгоритми від традиційних алгоритмів. Іншою важливою різницею є те, що згадані трансформації використовують оператори трансформації взяті з теорії біологічної еволюції (мутації, природній відбір, самовідтворення).

Існує три базових варіанти еволюційних алгоритмів: Генетичні алгоритми, еволюційні стратегії, та еволюційне програмування. Хоча ці методи мають спільну основу, всі вони були створені та реалізовані абсолютно незалежно один від одної.

В генетичних алгоритмах, у всіх розв'язків є їх оцінка, що визначає їх пристосованість. Наприклад в Генетичних алгоритмах, множина розв'язків представлена множиною хромосом.

В цілому, всі види еволюційних алгоритмів працюють по стандартному набору кроків. Для початку генерується випадкова популяція певних індивідів. Далі розраховується пристосованість кожного індивіда.

1.1 Еволюційні алгоритми

Еволюційне обчислення є загальним терміном для методів оптимізації, натхненних еволюційною теорією Дарвіна. У природній еволюції люди прагнуть до виживання і відтворення в конкурентному середовищі. Ті, у кого

більше сприятливі риси, отримані внаслідок успадкування або мутації, мають більше шансів на успіх.

Еволюційні алгоритми є специфічними реалізаціями концепції еволюційного обчислення. Еволюційні алгоритми вирішують обчислювальні проблеми, керуючи набором (населення) осіб. Кожен індивідуум кодує рішення кандидата на обчислювальну задачу в своєму геномі. Для вивчення простору рішення породжує потомство батьківської популяції через операторів рекомбінації поєднувати властивості батьків.

Додатково оператори мутації застосовуються для введення випадкових варіацій з метою розширення досліджень і запобігання передчасній конвергенції. Нова популяція створюється шляхом вибору з безліч осіб з батьківської популяції і та їх потомства. Це процес рекомбінації, мутації і працює в одному поколінні і повторюється протягом всього виконання алгоритму.

Для керівництва пошуком найкращого розвязку, використовується еволюційний тиск у вигляді функції пристосованості. Пристсовані індивіди мають більший шанс на продовження роду та виживання.

Завдяки своїй паралелізованій конструкції та пристосованості для складних задач, еволюційне алгоритми є цінним інструментом у створенні моделей там де класичні методи не є ефективними.

Ключовим завданням у застосуванні та реалізації еволюційних алгоритмів є вибір еволюційних параметрів. Навіть найпростіші реалізації мають значну кількість параметрів.

Вибір значень параметрів має значний вплив на ефективність еволюційного алгоритму для різних проблем та різних видів однакових задач. Крім того, використання фіксованих параметрів протягом всі поколінь може бути неоптимальним.

Для врахування цього бажано, щоб еволюційні алгоритми були адаптивними. У цьому контексті йдеться про адаптацію та динамічний

контроль параметрів еволюції (не слід плутати з біологічною терміновою адаптацією).

Існують різні види адаптації, які можна використовувати в еволюційних алгоритмах :

1) Адаптація рівня навколишнього середовища змінює спосіб, у який окремі особи оцінюються навколишнім середовищем, наприклад, змінюючи функцію фітнесу.

2) Адаптація на рівні населення змінює параметри, які впливають на всю або деяку частину населення, наприклад, шляхом зміни чисельності населення.

3) Адаптація на індивідуальному рівні дозволяє вибирати параметри специфічних особин у популяції, наприклад підвищення ймовірності мутацій у осіб з низьким значення функції фітнесу.

4) Адаптація на рівні компонентів змінює параметри, які є специфічними для певної частини генома, наприклад, шляхом управління ймовірністю мутації одного гену.

1.2 Генетичні алгоритми

Генетичні алгоритми - це алгоритм оптимізації, який використовує принципи генетики та природнього відбору. Його часто використовують для пошуку оптимальних рішень для складних задач. Він часто використовується для розв'язання задач оптимізації, досліджень і машинного навчання. Генетичні алгоритми мають можливість знайти «достатньо швидко» «досить точний» розв’язок. Це робить генетичні алгоритми привабливими для використання при розв'язанні задач оптимізації.

Генетичні алгоритми, вперше були запропоновані Джоном Холандом в 1975. Вони були створені на основі ідеї Дарвіна про природній відбір. Центральна ідея, якого це селекція та виживання сильнішого. Важливими в

даній теоріє є наступні поняття, які варто, перед початком обговорення генетичних алгоритмів зазначити і, які будуть використовуватися надалі. Покоління - це множина можливих розв’язків даної проблеми. Хромосома є

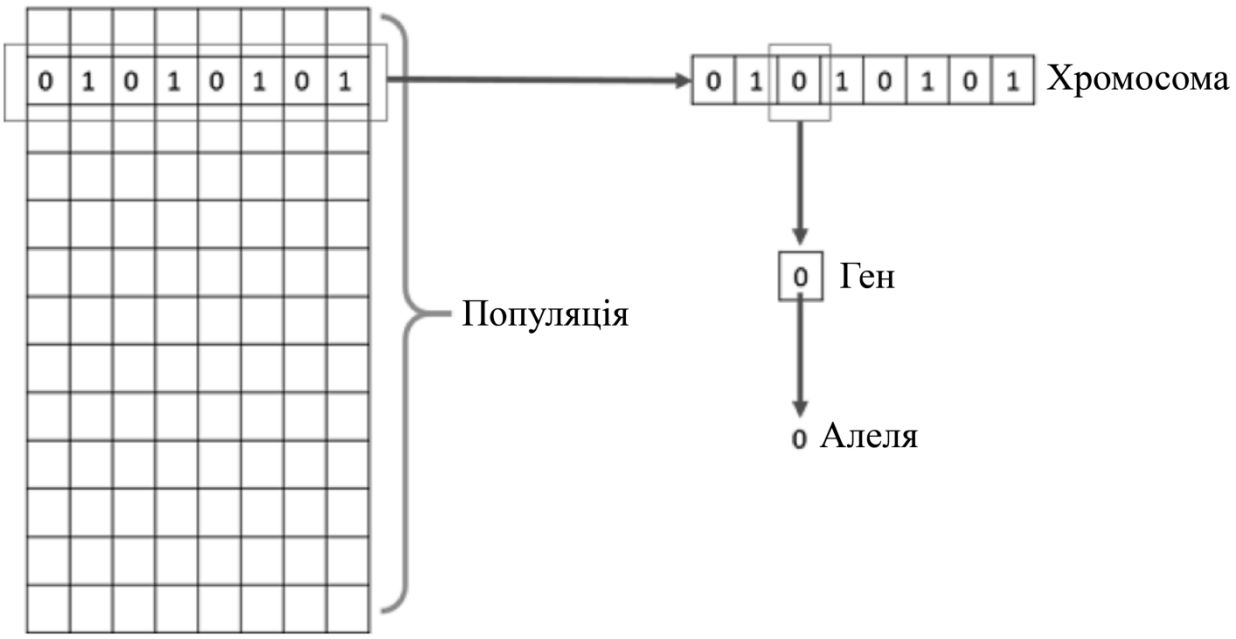


Рисунок 1.1- Загальне представлення моделі

одним з таких рішень даної проблеми. Ген - один з елементів хромосоми. Аллелі - це значення певної хромосоми. Функція пристосованості визначається як функція, яка на вході приймає розв’язок, і на виході видає його відповідність даній проблемі. Генетичні оператори змінюють генетичний склад потомства. До них відносяться перехід, мутація, відбір і т.п.

Через процес природньої селекції організми адаптуються та оптимізують свої шанси на виживання в даному їм природньому середовищі. Також відбуваються випадкові мутації генів організмів, які передаються їх дітям. Якщо мутація виявилась вдалою, ці діти мають більший шанс для виживання, і в свою чергу для продовження роду. Якщо мутація виявилась невдалою, то діти індивіда мають менший шанс на виживання та продовження роду, тому невдалі гени помруть з ними.

Аналогічно, ГА підтримують “популяцію” кандидатів розв’язків для даної задачі. Елементи випадково обираються з даної популяції та продовжую свій рід комбінуючи частини двох батьківських розв’язків. Ключова особливість в тому, що ймовірність елемента бути обраним для репродукції основана на їх пристосованості, по суті функції рішення. З часом непридатні розв’язки зникають з популяції та замінюються на дітей кращих розв’язків. В генетичних алгоритмах розв’язки представляються бінарно. Значення пристосованості оцінює ці розв’язки

Теоретичні дослідження були викликані інтересом до розуміння та підвищення ефективності роботи ГА. Кінцева ціль полягала у проектуванні ефективного та надійного ГА. Для досягнення даної мети важливо розуміти, як ГА працюють та для якого типу задач їх варто використовувати. Майже всі теоретичні роботи в ГА випливають з цих двох фундаментальних питань[1].

1.3 Структура генетичних алгоритмів

Багато різних ГА були створені використовуючи одну і ту ж ідею природнього відбору. Вони можуть мати різні оператори, але більшість ГА основані на простому прототипі стандартної ГА, який має лише базові оператори. Модель складається з наступних елементів: ініціалізація населення, а потім ітераційний, що включає у себе оцінку та розмноження(селекція, кросове, мутація).

Перший крок у будь-якій ГА полягає в тому, щоб розробити відповідну схему представлення хромосом, які є кандидати на розв’язок для розглянутих задач. Шифрування - процес презентації потенціального розв’язку, який містить набір змінних рішень, у вигляді рядка коду.

Потенціальний розв’язок можна представити у різних видах форматів: бінарний код, букви, дійсні числа. Залежно від характеру проблеми може бути обрана відповідна схема подання. Класичним вважається бінарне

представлення. Зашифровані змінні називаються генами, а значення цих генів (0,1) називаються алелями.

Рисунок 1.2 - Модель ГА

Представивши розв’язок у зручному для нас вигляді, ми маємо знати на скільки хорошим цей розв’язок є. Для цього ми маємо визначити цільову функцію. Так як ГА працює лише з зашифрованими даними, тому йому не важливий фізичний аспект завдання, то одним з його найбільших плюсів є можливість оптимізації розв’язків, де зв'язок між ними та функції пристосованості невідомі чи занадто складні для формулювання. Значення цільової функції індивідуального рішення можна отримати, через відомі значення чи за допомогою комп'ютерної симуляції. Цей крок встановлює відповідність між змінними та функцією в чорному ящику. Завданням може бути, як максимізація, так і мінімізація цільової функції.

Після того, як розв’язки зашифровані, можна перейти до процесу еволюції. Спершу, популяція зашифрованих розв’язків, іншими словами

хромосом, створюється випадково. Далі, популяція проходить ітераційний процес розвитку та репродукції. Кожна хромосома отримує своє значення функції пристосованості відповідно до цільової функції. Далі, хромосоми нового покоління створюються використовуючи оператори репродукції на старому поколінні. Це три базових оператори: селекція, кросове, мутація. Новостворені хромосоми знову оцінюються. Ітерація продовжиться поки популяція розв'язків не дійде до оптимального розв'язку[2].

1.3.1 Популяція

Популяція - це підмножина рішень у поточному поколінні. Вона також може бути визначена як набір хромосом. Є кілька речей, які необхідно мати на увазі при роботі з населенням ГА. Перше це те, що різноманітність населення завжди потрібно підтримувати, інакше ми можемо прийти до завчасного сходження до розв'язку. Друге це те, що розмір популяції має бути не занадто великим, так як це може привести до сповільнення роботи генетичного алгоритму, але занадто мала популяція може бути недостатньою для продовження роду. Таким чином, оптимальний розмір популяції обирається методом спроб та помилок. Популяція зазвичай визначається як двовимірний масив - розмір популяції, розмір хромосом.

Існує два методи для ініціалізації популяції. Перший - випадкова ініціалізація. В цьому способі популяції заповнюється абсолютно випадковими розв'язками. Другий – евристична ініціалізація. В ньому популяції заповнюється на основі, якихось евристичних даних.

Було доведено, що популяції не варто повністю заповнювати за допомогою евристичного методу, так як результат в популяції, що всі розв'язки в популяції мають маленьку різноманітність. Було доведено, що саме випадкові розв'язки, приводять популяцію до оптимального значення. Тому, використовуючи евристичну ініціалізацію, ми просто встановлюємо

декілька хороших розв’язків, заповнюючи решту популяції випадковими індивідами. Також спостерігалося, що евристична ініціалізація в деяких випадках лише впливає на початкову придатність населення, але в кінцевому підсумку саме різноманіття рішень призводить до оптимального значення.

Існують дві широко використовувані моделі населення. У моделі стаціонарного стану ГА, у кожній ітерації генерується один або два нащадки і вони замінюють одну чи дві особи з популяції. Стаціонарний стан ГА також відомий як інкрементний ГА. У моделі покоління ми генеруємо n нащадків, де n - розмір популяції, і в кінці ітерації все населення замінюється новим[3].

1.3.2 Функція пристосованості

Функція пристосованості є функцією, яка приймає кандидата на розв’язок проблеми в якості вхідних даних і видає, на скільки «підходить» наше «хороше» рішення стосовно розглянутої проблеми.

Розрахунок значення функції пристосованості проводиться багаторазово в ГА, а тому він повинен бути досить швидким. Повільне обчислення значення функція пристосованості може негативно вплинути на ГА і зробити даний метод надзвичайно повільним.

У більшості випадків функція пристосованості та цільова функція однакові, оскільки метою є або максимізація, або мінімізація даної цільової функції. Однак, для більш складних проблем з кількома цілями та обмеженнями, конструктор алгоритму може вибрати іншу функцію придатності.

Функція фітнесу повинна бути достатньо швидкою для обчислення. Також вона повинна кількісно виміряти, наскільки підходить дане рішення або як придатні особи можуть бути отримані з даного рішення.

У деяких випадках розрахунок функціональної придатності безпосередньо може бути неможливим через властиві складності проблеми.

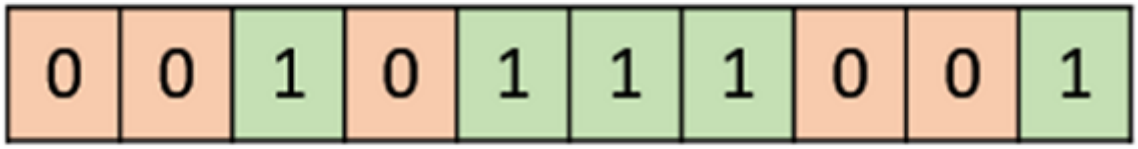
У таких випадках ми апроксимуємо функцію пристосованості до відповідних потреб.

1.3.3 Представлення генотипу

Одним з найважливіших рішень, які необхідно прийняти під час реалізації генетичного алгоритму, є прийняття представлення, яке буде використано для представлення наших рішень. Було відмічено, що неправильне представлення може призвести до некоректної роботи ГА. Тому, вибираючи правильне представлення, маємо знайти належне визначення відображення між фенотипом і просторами генотипу, що є суттєвим для успіху ГА. У цьому розділі я представлю деякі з найбільш часто використовуваних представлень для генетичних алгоритмів.

Двійкове представлення. Це одне з найпростіших і найбільш широко використовуваних представлень у ГА. У цьому типі подання генотип складається з бітових рядків.

Для деяких завдань, коли простір рішення складається з нульових змінних рішення - двійкове представлення є природним. Візьмемо, наприклад, проблему 0/1 Кнапсака. Якщо є n елементів, то ми можемо представити рішення двійковим рядком з n елементів, де x-елемент вказує, чи вибрано елемент x (1) чи ні (0).



Для інших проблем, зокрема тих, що стосуються чисел, ми можемо представляти числа у вигляді двійковим поданням. Проблема з цим типом кодування полягає в тому, що різні біти мають різне значення і тому оператори мутації і кросовера можуть мати небажані наслідки. Це може бути

вирішено певною мірою за допомогою програми Grey Coding, оскільки зміна одного біта не має масового впливу на рішення.

Реальне представлення. Для задач, де ми хочемо визначити гени за допомогою неперервних, а не дискретних змінних, реальне ціннісне представлення є найбільш природним. Точність цих справжніх чисел із плаваючою точкою обмежена лише комп'ютером.

0.5	0.2	0.6	0.8	0.7	0.4	0.3	0.2	0.1	0.9

Цілочислене представлення. Для дискретно представлення генів ми не завжди можемо обмежити простір рішення двійковим "так" або "ні". Наприклад, якщо ми хочемо кодувати чотири відстані - Північ, Південь, Схід і Захід, ми можемо кодувати їх як {0,1,2,3}. У таких випадках краще ціле уявлення.

1	2	3	4	3	2	4	1	2	1

Представлення перестановок. У багатьох задачах рішення представляється порядком елементів. У таких випадках найбільш підходящим є перестановочне представлення.

Класичним прикладом такого подання є проблема торгового агента (TSP). При цьому продавець повинен провести екскурсію по всіх містах, відвідати кожне місто рівно один раз і повернутися до стартового міста. Загальна довжина туру повинна бути мінімізована. Рішення цього TSP, природно, є впорядкованістю або перестановкою всіх міст і тому використання представлення перестановок має сенс для цієї проблеми.

1	5	9	8	7	4	2	3	6	0

1.3.4 Селекція

Селекція батьків – це процес вибору батьків, які продовжують свій рід тобто рекомбінуються, щоб створити нащадків для наступного покоління. Вибір батьків є дуже важливим для швидкості конвергенції ГА, оскільки хороші батьки приводять нащадків до кращого та точнішого рішення.

Важливо, уважно розглядати процес селекції, щоб запобігти одній частій проблемі, коли один правильний розв'язок, забирає на себе всю популяцію, протягом декількох поколінь, що приводить до того, що усі рішення будуть близькими один до одного в просторі розв'язків, що в свою чергу приведе до втрати різноманітності. Підтримка великого розмаїття населення надзвичайно важливо для успіху ГА. Захоплення всієї популяції одним надзвичайно придатним рішенням відоме як передчасне зближення і є небажаним станом в ГА.

Оператор селекції створює число копій кожної хромосоми для подальших операцій. Процедура селекції працює по правилу виживання сильнішого, яке виділяє більше квот для копій кращих розв'язків. Існують різні способи селекції. Найчастіше використовується пропорційна селекція, яка виділяє стільки копій кожної хромосоми пропорційно значенню їх функції придатності. Далі новостворене покоління замінить собою оригілну популяцію для подальших операцій. Таким чином, ймовірність виникнення хромосом, які нам підходять росте.

На етапі відбору потрібно з усієї популяції вибрати певну її частку, яка залишиться «в живих» на цьому етапі еволюції. Є різні способи проводити відбір. Пропорційна селекція є одним з найпопулярніших способів вибору батьків. При цьому кожен може стати батьком з вірогідністю, яка пропорційна його придатності. Імовірність виживання особини повинна залежати від значення функції пристосованості. Сама частка тих, хто вижив s зазвичай є параметром генетичного алгоритму, і її просто задають заздалегідь. За підсумками відбору з N особин популяції S повинні

залишитися особини, які увійдуть до підсумкової популяцію S . Таким чином, кращі індивіди мають більше шансів на спаровування і поширення своїх особливостей на наступне покоління. Таким чином, така стратегія відбору дає більше можливостей продовження роду та розвитку тим

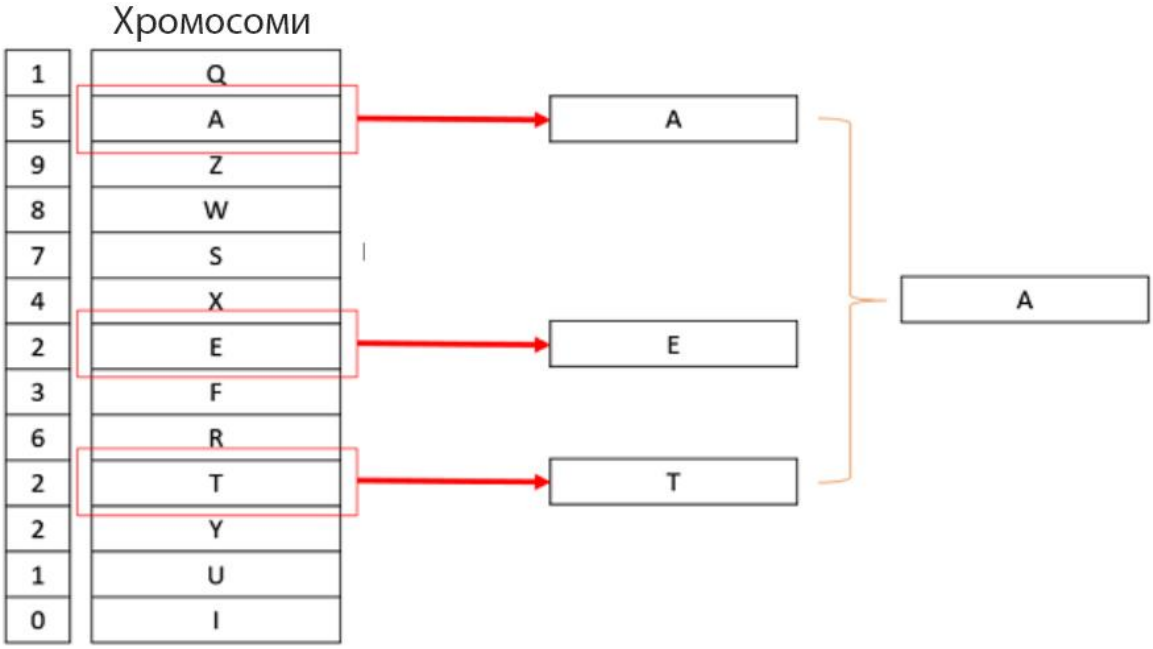


Рисунок 1.3- Турнірний метод

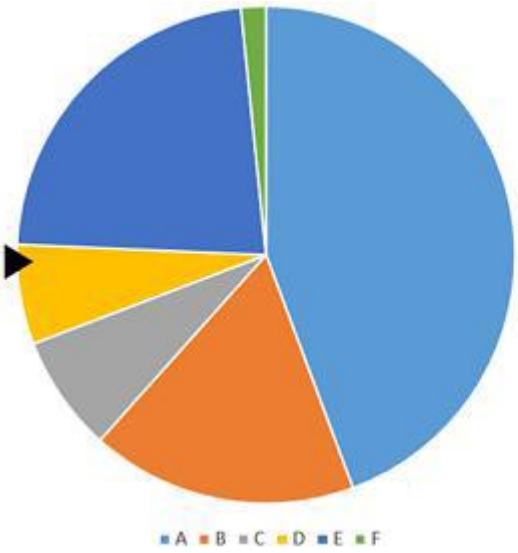
індивідам, хто краще підходить. Якщо розглянути кругову діаграму, в якій кільце ділиться на n-індивідів. Кожен індивід отримує ту частину, яка відповідає його функції пристосованості.

Турнірна селекція - спочатку випадково вибирається встановлену кількість особин (зазвичай дві), а потім з них вибирається особина з кращим значенням функції пристосованості. Зазвичай, прийнято розмір групи особин приймати за двійку. Такий турнір називають двійковим. Головна перевага турнірного відбору у тому, що він не втрачає вибірковості у випадку, коли у ході еволюції всі елементи популяції стають приблизно рівними[4].

Метод рулетки - ймовірність вибору особини тим імовірніше, чим краще її значення функції пристосованості. У методі рулетки аналогічно використовується колесо. На окружності кола обирається фіксована точка, та

колесо обертається. Область колеса, що приходить перед фіксованою точкою, вибирається у якості батьків. Для другого батька повторюється той самий процес.

Зрозуміло, що індивіди з більшим значенням функції пристосованості мають більший шанс на продовження роду. Тому ймовірність вибору особи безпосередньо залежить від його придатності. Хоча ймовірність вибору кандидата з більшим значення функцію пристосованості більша, все ще існує ймовірність того, що вони зникнуть бо ймовірність їх вибору не дорівнює 1.



Хромосоми	Фітнес
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

Рисунок 1.4- Метод рулетки

Порівняйте це з іншими способами відбору, що виключають відсоток найслабших кандидатів. В даному методі існує ймовірність того, що деякі слабкі рішення переживуть процес відбору. Зрозуміло, що індивіди з більшим значенням функції пристосованості мають більший шанс на продовження роду. Тому ймовірність вибору особи безпосередньо залежить від його придатності. Хоча ймовірність вибору кандидата з більшим значенням функції пристосованості більша, але все ще існує ймовірність того, що вони зникнуть, бо ймовірність їх вибору не дорівнює 1. Порівняйте це з іншими способами відбору, що виключають відсоток найслабших кандидатів. В даному методі існує ймовірність того, що деякі слабкі рішення переживуть процес відбору. Це відбувається тому, що навіть якщо

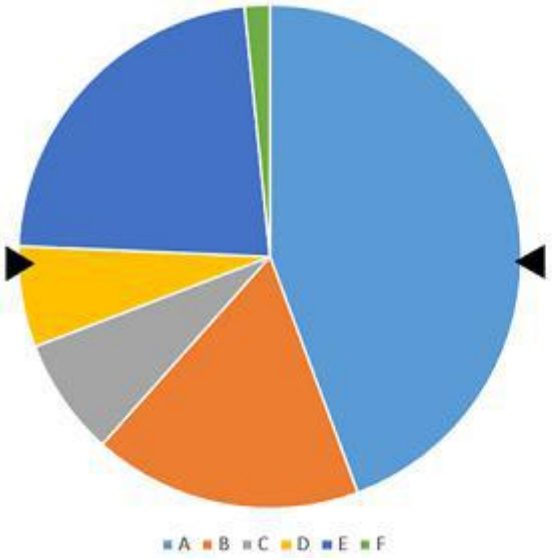
ймовірність того, що слабкі рішення виживуть, є низькою, вона не є нульовою, а це означає, що вони все ще можуть вижити.

Це є перевагою, оскільки існує ймовірність того, що навіть слабкі розв'язки можуть мати деякі особливості або характеристики, які можуть виявитися корисними після процесу рекомбінації.

Інші методи відбору, такі як стохастична універсальна вибірка або вибір турніру, часто використовуються на практиці. Це пояснюється тим, що вони мають менше стохастичних шумів або є швидшими, легкими для реалізації.

Для реалізації даного методу потрібно зробити наступне. Обчислити *S* фітнес. Створити випадкове число між (0,1). Починаючи з вершини популяції, додавати фітнес поки не отримаємо *P*, *P*<*S*.

Стохастична вибірка (SUS). Стохастична вибірка досить схожа на метод рулетки, але замість того, щоб мати лише одну фіксовану точку, ми маємо кілька фіксованих точок, як показано на зображенні 1.4. Тому обох батьків вибирають всього за одне обертання колеса. Також, така установка гарантує індивідам з високим значенням фітнесу, бути обраними принаймні один раз.



Хромосоми	Фітнес
A	8.2
B	3.2
C	1.4
D	1.2
E	4.2
F	0.3

Рисунок 1.5- Стохастичний метод

Слід зазначити, що стохастичний метод не працює у випадках, коли функція фітнесу може приймати від'ємне значення.

Відбір по рейтингу працює з від’ємними значеннями фітнесу та в основному використовується, коли індивідууми в популяції мають дуже близькі значення фітнесу (це відбувається зазвичай наприкінці алгоритму). Це приводить до того, що кожний індивід має майже рівну ймовірність (як у пропорційному методі), як показано на 3.5, і отже кожний індивід, не залежно від їх співпадіння значень функції фітнесу, мають приблизно однакову ймовірність стати батьками.

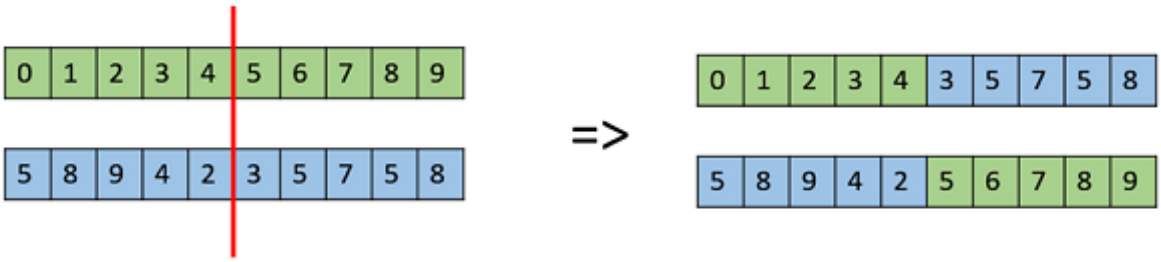
При цьому ми вилучаємо поняття пристосованості під час вибору батька. Проте кожна людина в популяції класифікується відповідно до їхньої придатності. Відбір батьків залежить від рангу кожної особи, а не від його придатності. Особи, що займають більш високі позиції, є кращими, ніж більш низькі.

Випадкова селекція. У цій стратегії ми випадковим чином вибираємо батьків з існуючого населення. Немає жодного фільтру на те хто виживе та продовжить рід, тому ця стратегія, як правило, уникається[5].

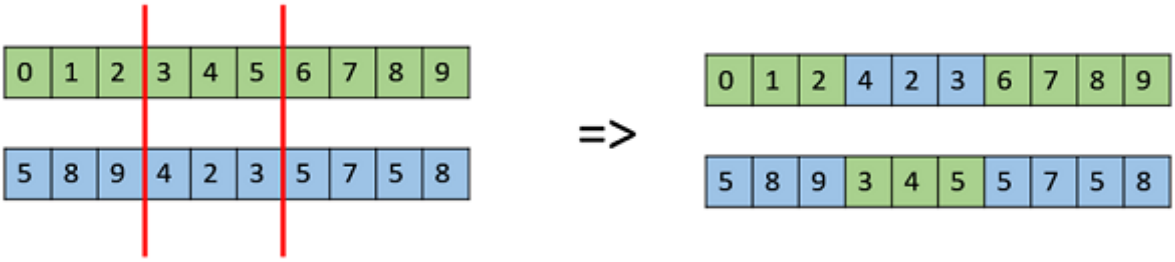
1.3.5 Схрещування

Оператор схрещування обмінюється шматочками генів між хромосомами. Спершу, популяція після селекції ділиться на групи, кожна з яких містить дві хромосоми. Далі дві хромосоми (батьки) в кожній групі міняються сегментами їй генетичного коду один з одним, для того щоб створити дві нові хромосоми (дітей). Далі діти стають на місце їх батьків у популяції. Через процес схрещування, ми отримуємо нові хромосоми у популяції, таким чином маємо шанс отримати кращі хромосоми. Це неможливо отримати у процесі селекції. Є багато схем схрещування, такі як, моно кросове, мульті-схрещування.

Одноточкове схрещування. У цьому схрещуванні вибирається випадкова точка схрещування, а хвости двох батьків обмінюються, щоб отримати нові хромосоми.



Багатоточкове схрещування - це узагальнення одноточкового схрещування, в якому чергуються сегменти для заміни, та отримання нових хромосом.

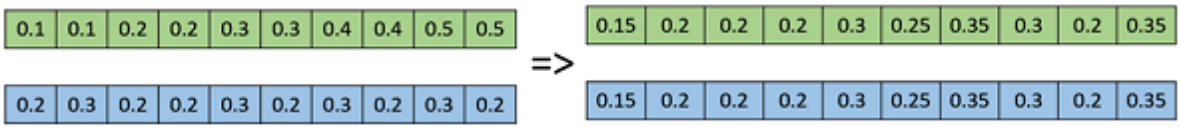


Повна арифметична рекомбінація. Зазвичай використовується для цілочисленних представлень хромосом, та працює розраховуючи середнє значення обох батьків, використовуючи наступні формули :

$$offspring_{with} = parent_{with1}(a) + parent_{with2}(1 - a) \quad (2)$$

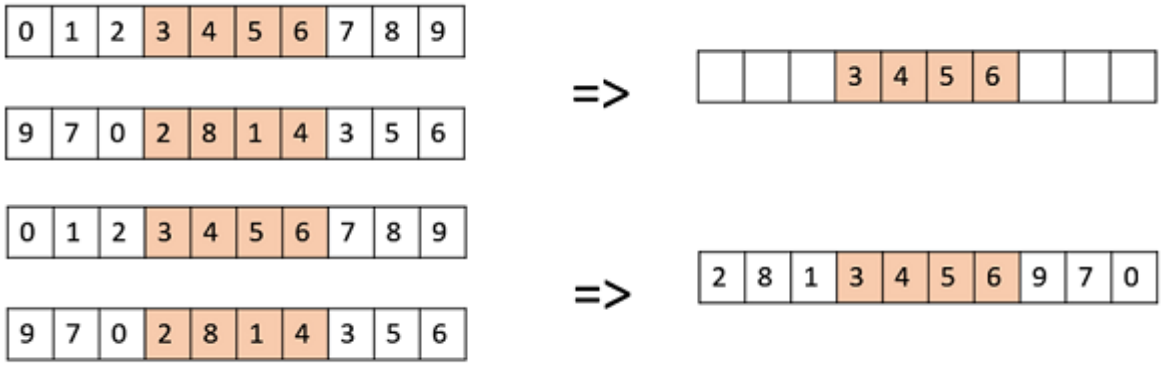
$$offspring_{who} = parent_{who1}(a) + parent_{who2}(1 - a) \quad (3)$$

У випадку коли $a=0,5$ обоє дітей можна розглянути так як на наступній картинці:



Схрещування Девіса (*OX1*). *OX1* використовується для передачі інформації, щодо відносного упорядкування нащадків. Він працює наступним чином. Спершу обираються, дві випадкові точки схрещування у батьків, та копіює сегменти між першим з батьків та першим з нащадків. Далі, починаючи з другої точки схрещування у другому з батьків, копіюємо

всі невикористані числа у першого нащадка, закінчуючи хромосому. Далі повторюємо, цей процес для другого нащадка.



Існує багато інших кросове рів, таких, як часткове відображення *(PMX)*, кросове на основі порядку *(OX2)*,кросове кільця, тощо.

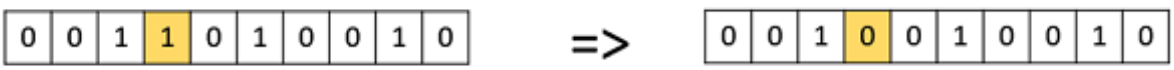
1.3.6 Мутація

Мутація – операція, яка змінює індивідуальні алелі у випадкових частинах випадкової хромосоми з ймовірність (1,0). Мутація це досить випадкова операція. Вона може створити, як хорошу так і погану хромосому, які надалі розвинуться або зникнуть у наступній селекції. Так як, ціль - це пошук оптимальних рішень, то одне погане рішення знаходиться в популяції. Однак, корисно, коли хороше рішення з’являється у процесі мутації.

Мутація - це частина ГА, яка пов'язана з «дослідженням» простору пошуку. Було відмічено, що мутація є суттєвою для роботи ГА, тоді як схрещування не є суттєвим.

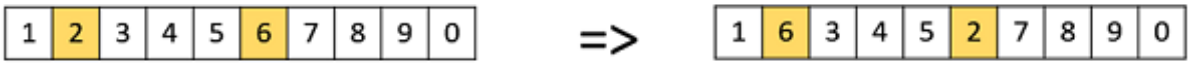
Існує декілька видів мутації. Деякі з них будуть приведені далі.

Біт-фліп-мутація. У цій бітовій мутації відбираємо один або більше випадкових бітів і перевертаємо їх. Це використовується для двійкових зашифрованих ГА.

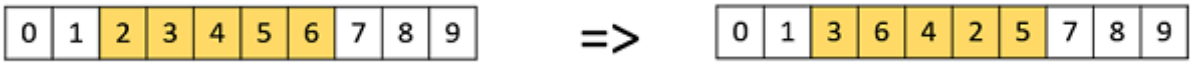


Випадкове скидання - це розширення фліп-біта для цілочисельного представлення. У ньому довільно вибраному гену присвоюється випадкове значення з набору допустимих значень.

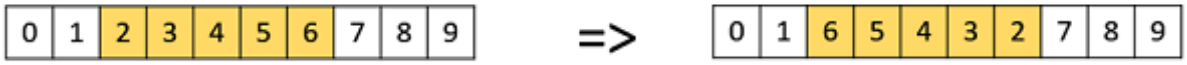
Мутація обміну. У мутації обміну ми обираємо дві алелі у хромосомі випадковим чином і змінюємо значення. Це поширене метод в шифруванні на основі перестановок.



Скрамбл мутація. В ній, з усієї хромосоми, обирається підмножина генів та їх значення перемішуються випадковим чино.



Інверсивна мутація. У інверсійній мутації ми вибираємо підмножину генів, подібно до скрамбл мутації, але замість того, щоб перетасувати підмножину, ми просто інвертуємо всю рядок у підмножині.



1.3.7 Політика відбору

Політика відбору визначає, які особини вимруть, і які мають бути збережені в наступному поколінні. Вона має вирішальне значення, оскільки вона повинна гарантувати, що успішні індивіди не будуть зникати з множини наступного покоління, і в той же час необхідно зберегти різноманітність популяції.

Деякі ГА використовують елітизм. Простіше кажучи, це означає, що поточні найбільш пристосовані члени популяції завжди переходять в

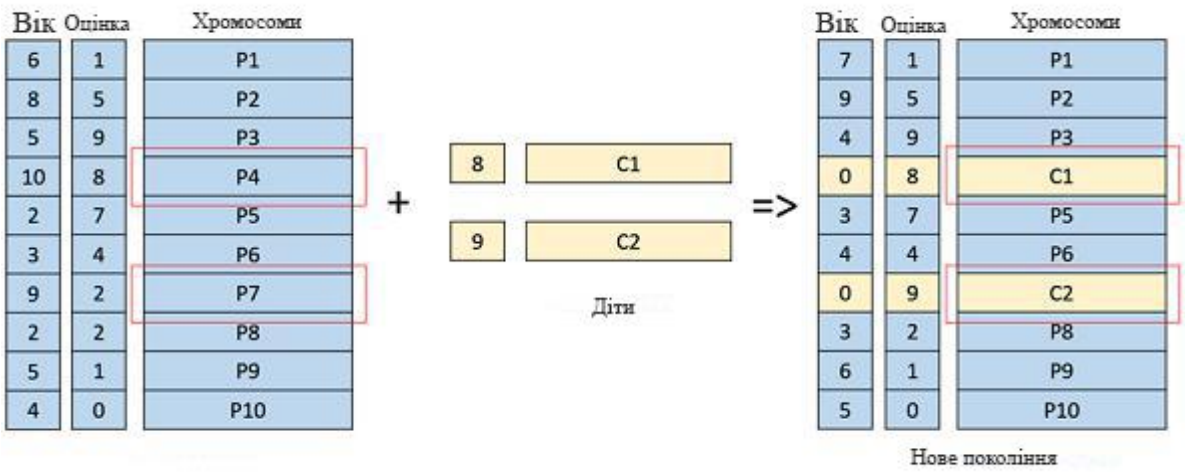


Рисунок 1.6- Відбір за віком

наступне покоління. Тому ні в якому разі не можна втратити найбільш пристосованого члена поточного населення.

Найпростіша політика - це вигнати випадкових членів з населення, але такий підхід часто має проблеми конвергенції, тому широко використовуються такі стратегії.

Відбір за віком. У віковій селекції ми не маємо поняття придатності. Вона ґрунтується на припущенні, що кожному індивіду дозволено знаходитися в популяції кінцеву кількість поколінь, де їм дозволено розмножуватися, після чого їх виганяють з населення незалежно від того, наскільки високе значення функції фітнесу у даного індивіда.

Наприклад, на рисунку 1.6 вік - це кількість поколінь, для яких індивід перебуває в популяції. Найстаріші члени популяції, тобто P4 і P7, викидаються з популяції і вік решти членів збільшується на один.

Відбір на основі фітнесу. У цьому виборі на основі фітнесу нащадки, як правило, замінюють найменш пристосованих осіб у популяції. Вибір найменш

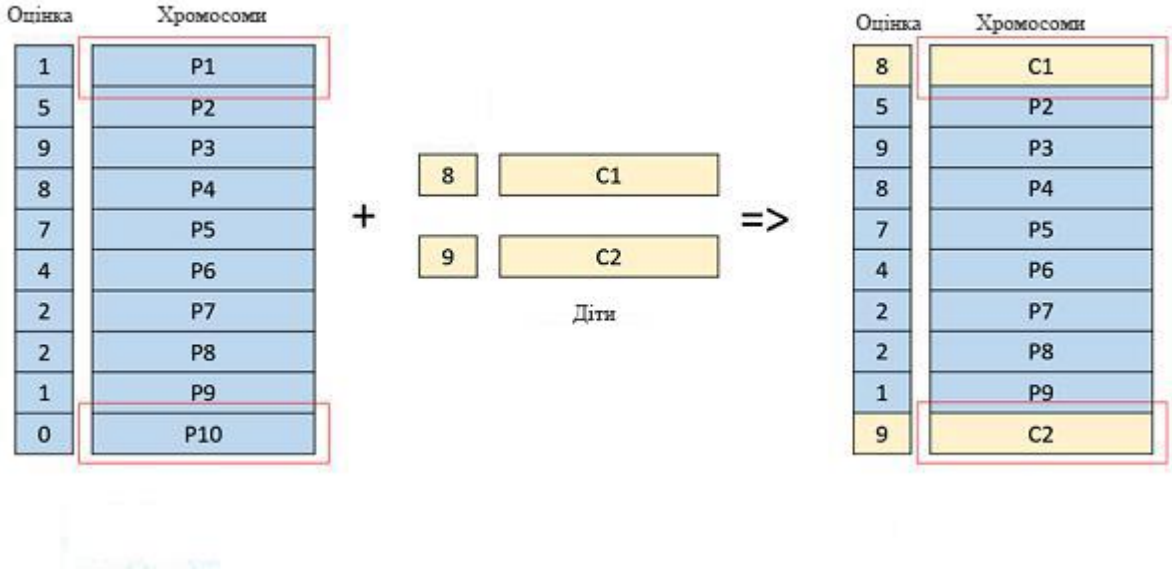


Рисунок 1.7- Відбір на основі пристосованості

придатних індивідів може бути зроблений за допомогою варіації будь-якої політики вибору, описаної раніше - вибір турніру, пропорційний вибір фітнесу тощо.

Наприклад, на рисунку 1.7 діти замінюють найменш придатних особин P1 і P10 популяції. Слід зазначити, що оскільки P1 і P9 мають однакове значення фітнесу, рішення про видалення індивідуума з населення є довільним[6].

1.4 Теорема схем та інші

Перша відносно строга модель була створена Джоном Холандом. В його теоремі схем він описує, як певні шматки коду процвітають або зникають залежно від їх придатності відносно середнього значення. Ця теорема пояснює, чому для певних задач певний клас генетичних алгоритмів є ефективним. Дана теорема описує за допомогою математики, як оператори селекції, схрещування та мутації працюють на покращення правильних схем. “Схема” розумітимемо підмножину бінарних розв’язків. Наприклад: 1**0.

Елементами підмножини, яку представляє цей шаблон тоді будуть 1000, 1010, 1100 та 1110.

Теорема схем говорить про ймовірність для певної схеми вижити та потрапити у наступне покоління, враховуючи, що ця схема з теперішнього покоління. Класичне визначення теореми схем наступне :

$$E[N(S,t+1)] \geq \left\{1 - \chi\frac{l(S)}{l-1} - \mu k(s)\right\} r(S,t)N(S,t), \quad (1)$$

$N(S,t+1)$ - кількість елементів схеми S в поколінні t;

$E[N(S,t+1)]$ - очікувана кількість S в t+1 поколінні;

l – довжина хромосом;

$l(S)$ – довжина довжина схеми S, що визначається, як відстань кількості розрядів між першим і останньою вказаною цифрою в схемі S., наприклад: l [(*, 1, *, 0, *)] = 2;

μ – швидкість мутацій;

X – швидкість схрещування;

Праву частину нерівності (1), можна розділити на три частини. Перша відповідає селекції, $r(S,t)N(S,t)$ дає число прикладів схеми S після операції вибору, пропорційної фітнесу. $\chi\frac{l(S)}{l-1}$ та $\mu k(s)$ показують ймовірність руйнування схеми S через кросове або мутацію.

Також варто розглянути можливість того, що схема S може бути відкинута через кросове або мутацію, але така ж схема може бути створена через на зразок тієї ж мутацією та схрещуванням. Розглянемо приклад схрещування, де схема (*,*,*,1,0,*) може бути зруйнована при точці схрещування між третім та четвертим числом. Але хромосоми (1,1,1,1,0) та (1,1,0,0,0) існують в популяції, та можуть стати частиною схрещування, в якому точка схрещування буде між третім та четвертим числом, тоді нова хромосома (1,1,1,0,0) буде створена, як частина (*,*,*,1,0,*) схеми. Таким чином права частина рівняння (1) показує меншу ймовірність продовження життя схеми ніж насправді.

Таким чином будівельні блоки- це короткі частини схем зі значенням пристосованості більшим середнього. Відповідно до Теореми схем, число цих будівельних блоків в популяції буде експоненціально рости через усі покоління. Теорема будівельних блоків говорить, що через селекцію, кросове та мутацію, хороші хромосоми з часом з'являються, у процесі з'єднання безлічі фрагментів будівельних блоків. Інакше кажучи, гіпотеза говорить про відношення між пристосованістю будівельних блоків та хромосом. Хоча і немає на даний момент чітких доказів правильності гіпотези будівельних блоків, вона правильна у більшості випадків, як було показано у безлічі практичних робіт.

Теорема, вона як і раніше служить основою багатьох теоретичних досліджень ГА. Деякі дослідження посилаються на дану теореми, чи просто модифікують її. Іншим підходом до моделювання ГА є модель ГА як марковського процесу. Марковські моделі спрямовані на опису поведінки збіжності ГА. Ці моделі точніші, але, на жаль, як правило, вони набагато складніші та пропонують мало практичних методів по розробці компонентів ГА. Також основуючись на вивчення механізмів ГА, Гольдберг запропонував гіпотезу будівельних блоків. За допомогою даної гіпотези було створено серію ГА з покращеною ефективністю та надійністю.

Генетичні алгоритми не працюють на будь-яких видах задач. Тому важливо розуміти, які типи задач, ГА здатний розв'язати, або, що робить деякі задачі складними для них. Центральна відповідь на це запитання знаходиться у понятті епістазу. Простіше кажучи, епістаз- це взаємозалежність між параметрами рішення, яка може бути нелінійною і, отже, робити проблему важкою для ГА. Давідор запропонував метод вимірюють епістаз [29]. Восс і Лієпін показали, що в епістаз у будь-якій задачі можна звести до простішого типу задачі. Однак для складних задач, такий перехід сам може виявитися величезним завданням[7].

1.5 Застосування Генетичного алгоритму

Генетичні алгоритми з моменту їх розробки були використані в безлічі різних сфер. У сфері інженерії, Яо (1992) використав даний алгоритм для розрахунку параметрів для нелінійних систем [1], Джонс(1996) використав ГА для створення дизайну клітин [2], Голд (1998) ввів ГА в розрахунки для кінематичного дизайну лез турбіни [3]. В галузі планування, Тімоті (1993) оптимізував послідовність задач за допомогою ГА [3], Дарвен (1994) створив архітектуру для роботи магазину планування. У сфері комп'ютерних наук, Ро (1995) використовує ГА для дизайну розподіленої бази даних [4]. В сфері обробки зображень, Такідонада (1993) використав ГА для реалізації автоматизованої сегментації та інтерпретації зображень[5], Хуанг (1998) реалізував виявляючи стратегії для розпізнавання облич [6].

Через те що ГА не має чіткої сфери використання, його використання не обмежене, а отже може бути застосовано до багатьох комбінаторних проблем оптимізації в різних дисциплінах. Генетичні алгоритми використовуються в основному в задачах оптимізації різних видів, але вони часто використовуються і в інших областях застосування. Далі перерахуємо деякі з областей, в яких часто використовуються генетичні алгоритми.

Оптимізація - генетичні алгоритми найчастіше використовуються в задачах оптимізації, в яких ми повинні максимізувати або мінімізувати задану цільову функцію за заданим набором обмежень.

Економіка - ГА також використовуються для характеристики різних економічних моделей, таких як павутинна модель, теорія рівноваги, теорія ігор, ціноутворення активів тощо.

Нейронні мережі - ГА також використовуються для навчання нейронних мереж, особливо рекурентних нейронних мереж.

Паралелізація - ГА також мають дуже хороші можливості у сфері паралелізації, і виявляються дуже ефективним засобом у вирішенні певних проблем, а також забезпечують хорошу область для досліджень.

Обробка зображень - ГА використовуються для різних завдань обробки цифрових зображень (DIP).

Планування - ГА використовуються також для вирішення різноманітних проблем планування, зокрема, проблеми з розкладу.

Генерація траєкторії роботів - для планування шляху, який приймає робот-рука, переміщення з однієї точки в іншу.

Параметричне проектування літальних апаратів - ГА були використані для проектування літаків шляхом зміни параметрів і вдосконалення оптимальних рішень.

Аналіз ДНК - GA використовували для визначення структури ДНК з використанням спектрометричних даних зразок[8].

Висновки до розділу 1

Генетичні та еволюційні алгоритми, є загальними термінами для методів оптимізації, натхненних еволюційною теорією Дарвіна. У природній еволюції люди прагнуть до виживання і відтворення в конкурентному середовищі. Ті, у кого більше сприятливі риси, отримані внаслідок успадкування або мутації, мають більше шансів на успіх.

Процес природного відбору починається з відбору найбільш пристосованих осіб з популяції. Вони виробляють потомство, яке успадковує характеристики батьків і буде додано до наступного покоління. Якщо батьки мають краще значення функції пристосованості, їхні діти будуть кращими, ніж батьки і матимуть більше шансів на виживання. Цей процес продовжує повторюватися і в кінці кінців буде знайдено покоління з найбільш пристосованими особами.

Ця теорія проста: якщо населення хоче процвітати, вона повинна постійно самовдосконалюватися, це виживання найбільш пристосованих. Кращий елемент населення повинен надихати нащадків, але і не забувати про решту індивідів , щоб зберегти деяку різноманітність і вміти адаптуватися у випадку зміни природного середовища.

2 НЕЙРОННІ МЕРЕЖІ

Нейронна мережа - це тип машинного навчання, який працює по принципу роботи людського мозку. Створюється штучна нейронна мережа, яка за допомогою алгоритму дозволяє комп'ютеру навчатися шляхом включення нових даних.

У двох словах, людський мозок - це не більше ніж мільярди мільярдів крихітних нейронів, пов'язаних синапсами, які спільно працюють над прийняттям рішень. Нейрон перегляне електричну інформацію від наших органів почуттів і передасть дані іншому нейрону, який потім перегляне і перейде до іншого, а потім іншого, і до іншого, поки зрештою ланцюг активованих нейронів не зможе знайти рішення. Кожен нейрон обробляє дані незалежно, але все ще залежить від решти мережі, щоб отримати будь-які нові дані або зробити що-небудь корисне.

Комп'ютер з нейронною мережею навчається виконувати завдання, аналізуючи навчальні приклади, які попередньо були обрані. Загальним прикладом завдання для нейронної мережі з використанням глибокого навчання є завдання розпізнавання об'єктів, де нейронна мережа представлена великою кількістю об'єктів певного типу, таких як кішка або знак вулиці, і комп'ютер, аналізуючи повторювані закономірності в представлених зображеннях, навчається класифікувати нові зображення.

2.1 Структура нейронних мереж

Штучна нейронна мережа намагається імітувати наш мозок. Всередині штучної нейронної мережі (ANN) маленькі процесори (які спочатку називалися персептронами Френка Розенблатта, ще в 1950-х роках), які беруть дані, виконують прості обчислення, а потім виробляють один або більше виходів. Ці виходи направляються до наступного шару процесорів, званих прихованим шаром, разом з входами інших процесорів для того, щоб зробити остаточне рішення.

Описуючи алгоритм нейронної мережі, зазвичай важливо уточнити три речі.

Архітектура. Архітектура визначає, які змінні беруть участь у мережа та їх топологічні відносини - наприклад, змінні задіяні в нейронній мережі можуть бути вагами зв'язків між ними нейронів, разом з функціями активації нейронів.

Правило активності. Більшість моделей нейронних мереж мають коротку динаміку: місцеві правила визначають, як діяльність нейронів змінюється у відповідь один одному. Зазвичай правило активності залежить від ваги (параметрів) в мережі.

Правило навчання. Правило навчання вказує, яким чином ваги нейронної мережі змінюються з часом. Зазвичай правило навчання залежить від активації нейронів. Це також може залежати від значень значення цілі, заданого вчителем, і поточне значення ваг.

Часто правила діяльності та правила навчання можуть бути отримані з ретельно вибраної цільової функції[9].

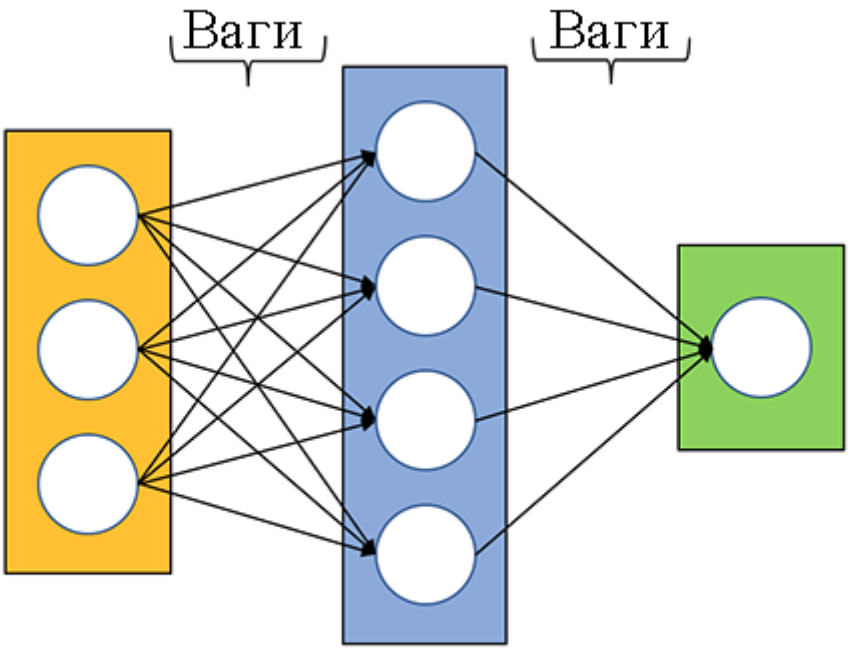
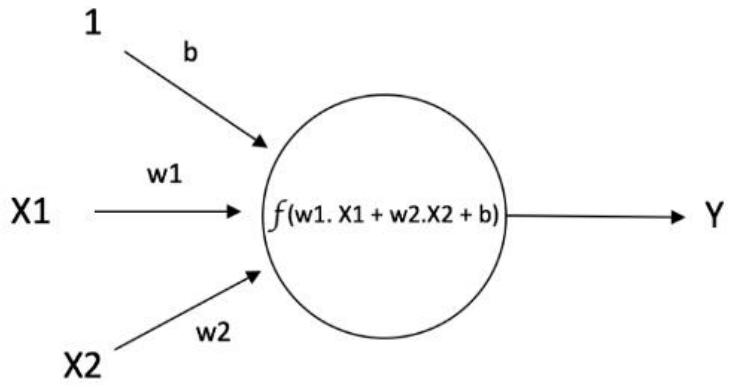


Рисунок 2.1- Нейронна мережа

В основному для опису нейронних мереж приводять аналогію з роботою мозку. Опишемо нейронну мережу, як математичну функцію, яка відображає заданий вхід на бажаний результат.

Нейронні мережі складаються з наступних компонентів :

1) Вхідний шар, x ;
2) Довільна кількість прихованих шарів ;
3) Вихідний шар ;
4) Набір ваг між шарами нейронів, W і b ;
5) Вибір функції активації для кожного прихованого шару, σ ;



$$Y = f(w1.\ X1 + w2.X2 + b)$$

Рисунок 2.2- Нейрон

Наведена на малюнку 2.1 нейронна мережа показує архітектуру двошарової нейронної мережі (вхідний шар, як правило, виключається при підрахунку кількості шарів у нейронній мережі).

Нейронні мережі є сімейством методів машинного навчання, побудованих на основі людського мозку. Можливість знаходження прихованих звязків у межах даних є ключовою здатністю для будь-якого підходу Data Scientist і Neural Network, що може бути особливо корисним для вилучення звязків із зображень, відео чи мови.

У базовій нейронній мережі ви тренуєте систему, проганяючи через модель окремі випадки за один раз і оновлюючи ваги. Мета полягає в тому, щоб з часом мережі повинні бути налаштовані на ваші дані, мінімізуючи помилки. Це оновлення ваг у базовій нейронній мережі є результатом двостороннього процесу з використанням методів подачі даних і зворотного поширення:

Перенаправлення передбачає обробку спостережень один за одним через мережу. Враховуючи ваги на місці, модель повинна виробляти прогноз, а з цього прогнозу і фактичного результату можна обчислити похибку вашої моделі для одного спостереження[10].

Зворотне розповсюдження передбачає прийняття цієї помилки через мережу для коригування індивідуальних ваг, щоб краще відобразити фактичний результат. Ці нові ваги потім використовуються для наступного спостереження. На жаль, зворотне поширення вважається багатьма недосвідченими ентузіастами глибокого навчання як алгоритм, який залякує і важко зрозуміти. Часто люди плутають зворотне поширення з градієнтним спуском, але насправді це дві окремі питання. Мета першого полягає в тому, щоб ефективно обчислити градієнт, тоді як другий - використовувати обчислений градієнт для оптимізації.

Існують деякі ключові фактори, які необхідно враховувати під час підбору моделі для ваших даних:

Швидкість навчання: Ви не хочете, щоб ваші ваги були занадто чутливими, оскільки ваша модель не хоче сильно змінюватися через одне конкретне навчання. Так само ви не хочете, щоб ваша модель була неактивною до нових даних. Тому важливо встановити відповідну швидкість навчання в моделі оптимізації.

Дизайн мережі (кількість прихованих шарів і кількість нейронів в кожному шарі) може змінюватись залежно від конкретної проблеми, яку ви хочете вирішити, і від того, яка гілка сімейства нейросіток ви використовуєте. Загалом, ви можете збільшити кількість нейронів через модель, якщо ви хочете мати більше можливостей або зменшити, якщо ви хочете отримати корисні абстрактні функції. Це стає ще більш важливим, якщо ви використовуєте більш складні методи, такі як Deep Learning.

Відомо, що нейронні мережі можуть бути повільними для навчання (хоча вони є масштабованими і доступні алгоритми навчання зі зменшенням карти). Деякі нейрони і всі їхні з'єднання випадають з мережі з заданою ймовірністю.

2.1.1 Функція втрат

Функція втрати. Для того, щоб контролювати наш прогрес і переконатися, що ми рухаємося в правильному напрямку, ми повинні регулярно обчислювати значення функції втрати. "Взагалі кажучи, функція втрат покликана показати, наскільки ми далекі від" ідеального "рішення".

Всі алгоритми в машинному навчанні покладаються на мінімізацію або максимізацію функції, яку ми називаємо «цільовою функцією». Група зведених до мінімуму функцій називається «функціями втрат». Функція втрати є показником того, наскільки хороша модель у прогнозування очікуваного результату. Найбільш часто використовуваним методом знаходження мінімальної точки функції є «градієнтний спуск».

Не існує жодної функції втрати, яка працює для всіх типів даних. Це залежить від ряду факторів, вибору алгоритму машинного навчання, часової ефективності градієнтного спуску, простоти пошуку похідних і впевненості в прогнозах.

Коли використову.nm нейронні мережі ? Як і будь-яка інша техніка, пов'язана з наукою даних, нейронні мережі є однією сім'єю багатьох підходів, які ви могли б прийняти для вирішення бізнес-проблеми, використовуючи великі обсяги даних. Техніка дуже інтенсивно поширюється на процесори і дає результати, які важко інтерпретувати. Якщо ви готові пожертвувати інтерпретативністю, нейронні мережі можуть бути корисною технікою.

Особливо це стосується вирішення проблем, з якими людський мозок дуже добре розуміє, наприклад, текст, зображення або розпізнавання голосу.

Оскільки ми маємо випадковий набір ваг, ми повинні змінити їх таким чином, щоб зробити наші вхідні дані рівними відповідним виходам з нашого набору даних. Це робиться за допомогою методу, який називається backpropagation[11].

Функція зворотного поширення працює за допомогою функції втрат для обчислення відстані від цільового виходу мережі. Нагадаємо, що для будь-якої задачі машинного навчання ми (прагнемо) вибирати вагові коефіцієнти, що забезпечують оптимальну оцінку функції, яка моделює наші дані навчання. Іншими словами, ми хочемо знайти набір ваг *W*, який мінімізує на виході *J (W)*.

2.2 Нейрони та функція активації

Як випливає з назви, нейронні мережі були натхненні нейронною архітектурою людського мозку, і, як у людському мозку, основний будівельний блок називається нейроном. Його функціональність схожа на людський нейрон, тобто приймає деякі входи і виводить вихід. У чисто математичних термінах нейрон у світі машинного навчання є заповнювачем для математичної функції, і його єдина робота полягає в тому, щоб забезпечити вихід, застосовуючи функцію на вхідних даних.

Один нейрон має *I* входів *xi* і один вихід яку ми тут будемо називати y. (Див. Малюнок 2.2) Пов'язана з кожним входом - вага *wi (i = 1, ..., I)*. Також, можливий додатковий параметр w0 нейрона, який називається зміщенням, яке ми можемо розглядати як вагу пов'язаний з входом x0, який постійно встановлюється рівним одиниці[12].

Функцію, що використовується в нейроні, зазвичай називають функцією активації. Існує багато різних функцій активації, таких, як , Sigmoid, tanh і ReLU наприклад.

Сигмоіду або логістичну функцію визначають математично як

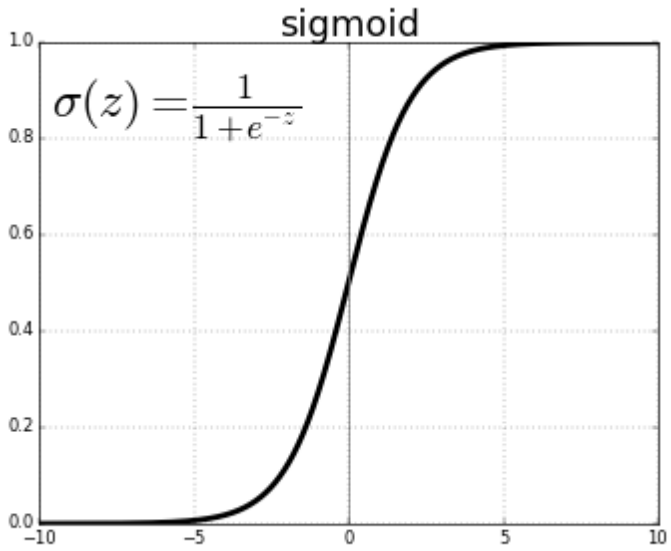


Рисунок 2.3 - Сігмоїда

Значення функції прямує до нуля, коли *z* прямує до мінус нескінченності і прямує до 1, коли *z* прямує до нескінченності. Маємо на увазі, що ця функція є наближенням поведінки залежної змінної і є припущенням. Тепер постає питання, чому ми використовуємо сигмоподібну функцію як одну з апроксимаційних функцій. Є декілька причин використовувати саме цю функцію[13].

Вона фіксує нелінійність даних. Хоча і в наближеному вигляді, але поняття нелінійності є істотним для точного моделювання.

Сигмоподібна функція дифференційована і тому може бути використана з градієнтним спуском і зворотним поширенням для розрахунку ваг різних шарів.

Однак сигмоподібна функція також страждає від проблеми зникнення градієнтів.

Як видно з малюнка, сигмоподібна функція стискає вхід у дуже малий діапазон [0,1] і має дуже круті градієнти. Таким чином, залишаються великі області вхідного простору, де навіть великі зміни призводять до дуже невеликих змін у виході. Це називається проблемою зникнення градієнта. Ця проблема зростає зі збільшенням кількості шарів і, таким чином, відбувається стагнація вивчення нейронної мережі на певному рівні[14].

Функція tanh (z) є масштабованою версією сигмоїди, а її вихідний діапазон - [- 1,1] замість [0,1].

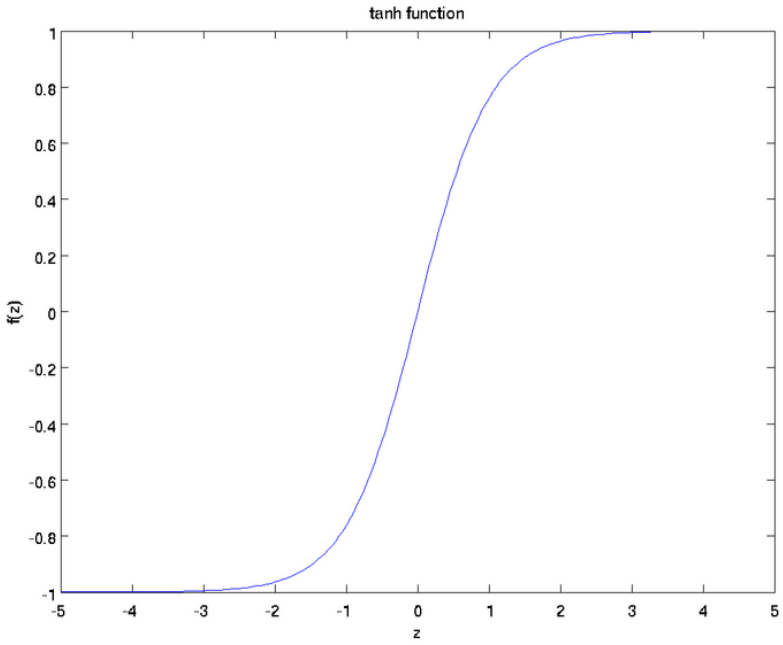


Рисунок 2.3 – Графік tahn

Загальною причиною використання функції tanh в деяких задачах замість сигмоподібної функції є те, що оскільки дані центровані навколо 0, то похідні вище. Більш високий градієнт допомагає покращити швидкість навчання. Перевагою є те, що негативні входи будуть відображені негативними значеннями, а нульові входи будуть відображені поблизу нуля

на графіку tanh. Функція також диференційована та монотонна , а її похідна - не монотонна. Функція tanh в основному використовується класифікацією між двома класами[12].

2.3 Застосування нейронних мереж

З урахуванням всього сказаного вище нейронні мережі і те, як вони працюють, для яких реальних додатків вони підходять? Нейронні мережі мають широку придатність до реальних світових бізнес-завдань. Фактично, вони вже успішно застосовуються в багатьох галузях.

Штучні нейронні мережі (ANN) є в даний час "гарячою" науковою областю в медицині, і вважається, що вони будуть отримувати широке застосування в біомедичних системах у найближчі кілька років. В даний час дослідження здійснюється в основному на моделюванні частин людського тіла і розпізнаванні захворювань від різних сканувань (наприклад, кардіограм, сканувань CAT, ультразвукових сканувань тощо).

Бізнес - це сфера, яку можна описати, з кількома загальними напрямками спеціалізації, такими як бухгалтерський облік або фінансовий аналіз. Майже будь-яке застосування нейронної мережі вписується в одну з галузей бізнесу чи фінансового аналізу.

Існує певний потенціал для використання нейронних мереж у комерційних цілях, включаючи розподіл ресурсів та планування. Існує також великий потенціал для використання нейронних мереж для видобування баз даних, тобто пошук моделей, які є неявними в явно зберігається інформації в базах даних. Більшість фінансованих робіт у цій сфері класифікуються як приватні. Таким чином, неможливо повідомити про повний обсяг

виконуваної роботи. Більшість робіт - застосування нейронних мереж, таких як мережа Hopfield-Tank для оптимізації та планування.

Але щоб дати вам більш конкретні приклади; ANN використовуються також у наступних специфічних задачах: визнання голосу у комунікаціях; діагностика гепатиту; відновлення телекомунікацій з несправного програмного забезпечення; інтерпретація багатозначних китайських слів; виявлення підводних шахт; аналіз текстури; розпізнавання тривимірних об'єктів; розпізнавання рукописного слова; і розпізнавання обличчя[15].

Висновки до розділу 2

Нейронні мережі є набором алгоритмів, змодельованих по прикладу мозку людини, призначених для розпізнавання закономірностей. Вони інтерпретують дані за допомогою свого роду машинного сприйняття, маркування або кластеризації вихідних даних. Моделі, які вони розпізнають, є числовими, містяться у векторах, в які повинні бути перекладені всі дані реального світу, будь то зображення, звук, текст або часові ряди.

В обчисленях великої кількості інформації, нейронні мережі показують хороші результати. Їхня здатність вчитися на прикладі робить їх дуже гнучкими і потужними. Крім того, немає необхідності розробляти алгоритм для виконання певного завдання; тобто немає необхідності розуміти внутрішні механізми цього завдання. Вони також дуже добре підходять для систем реального часу через їх швидку реакцію та час обчислень, що зумовлено їх паралельною архітектурою.

3 АНАЛІЗ ТА РЕАЛІЗАЦІЯ АЛГОРИТМУ

Моделювання життя та розвитку простих організмів, вимагає особливого підходу. Для нашої моделі, ми використовуємо нейронні мережі, як базу для простих живих організмів. Для відтворення процесу розвитку, скористаємося генетичними алгоритмами, які нададуть нашій моделі гнучкість, та можливість пристосовуватися до різних умов життя.

Організми в даній моделі будуть жити в умовах, коли їм, для розвитку необхідно харчуватися, якомога більшою кількістю їжі для успішного життя та продовження роду. Успішні індивіди отримають можливість продовжити свій рід та передати нащадкам, значення свої ваг в нейронній мережі. Таким чином успішні значення ваг зможуть продовжити існування.

Також для даної моделі розглянемо поведінку живих організмів, при зміні алгоритмів їх існування. Для цього розглянемо різні умови виживання, та розглянемо різну складність у даних організмів.

3.1 Реалізація моделі

Розглянемо, основний організм, який буде фундаментом та буде контролюватися повністю пов’язаною нейронною мережою. На вхід для нейронної сітки буде подаватися значення кута повороту до найближчої частинки їжі, яке буде нормалізоване значення в діапазоні від [-1, +1].

Оскільки на вхід у нашу нейронну мережу подаються значення нормалізовані в діапазоні [-1, +1], та на виході ми будемо отримувати

аналогічні значення в границі [-1, +1], то для функцій активації використаємо функції tanh або sigmoida.

Оскільки наш єдиний вхід варіюється від [-1, +1] і наші бажані виходи також варіюються від [-1, +1], функції tanh або sigmoida будуть нашими функціями активації. На рисунку 3.1 приведена вигляд нейронної мережі, кожного організму, на якій зображено її входи та виходи, а також з вигляд прихованого шару.

Клас організму містить нейронну мережу, а також функції, які оновлють значення швидкості та положення організму у просторі. Під час першої ініціалізації організмів , їх положення, напрямок, швидкість, прискорення і ваги нейронної мережі будуть генеруються випадковим чином, і надалі розвиватися за допомогою генетичних алгоритмів.

Клас їжі - це простий об'єкт, який містить розташування x / y та енергетичну цінність їжі. Енергетична цінність безпосередньо впливає на придатність організмів, які її поглинули. В даній моделі енергетична цінність залишається постійною і встановлюється рівною одиниці, але вона може бути випадковою або можливо спробувати створити місцевості в, яких енергетична цінність більша, таким чином змоделювавши нерівномірність розповсюдження їжі. Крім того, введемо функцію поновлення їжі, яка буде регенерувати харчові частки після її вживання організмом. Завдяки цьому загальна середня кількість частинок їжі на кожному кроці симуляції буде залишатися постійною.

Організми будуть розвиватися та мутувати за допомогою генетичного алгоритму (Га). Генетичні алгоритми, які працюють, імітуючи природний біологічний процес еволюції, будуть починати роботу з початкової популяції, і через відбір, кросовер і мутацію протягом багатьох поколінь виникнуть оптимальні індивіди.

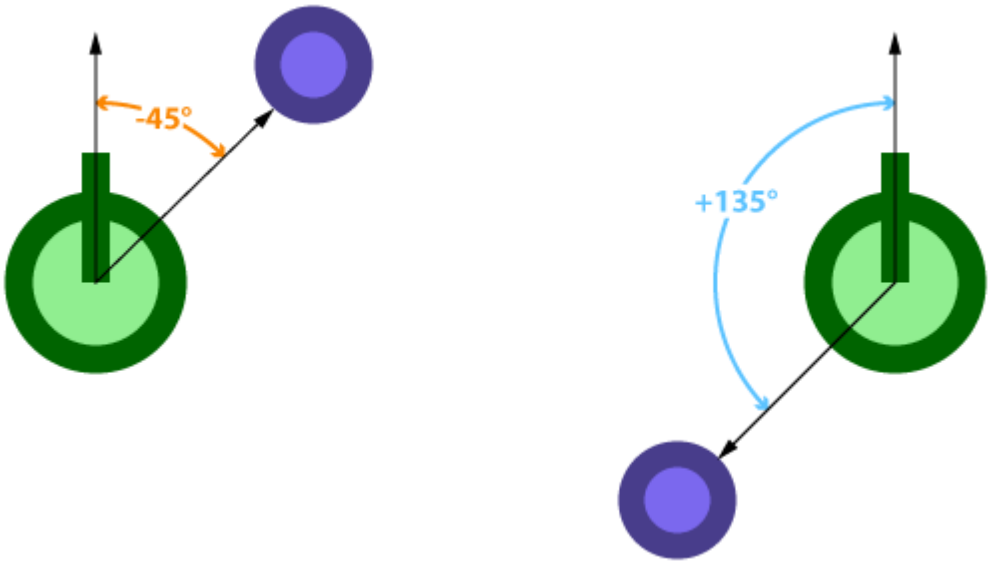


Рисунок 3.1- Рух організмів у просторі

Генетичний алгоритм не гарантовано знайде оптимальну поведінку індивіда в даній моделі, але при правильно підібраних початкових умовах, він знайде ті значення ваг при, яких поведінка організму буде оптимальною.

В реалізації нашої моделі використаємо наступні поняття.

Селекція - найпростіша форма селекції називається елітизмом, він відбирає і переносять до наступного покоління процент найбільш пристосованих індивідів індивідів з теперішнього покоління.

Кросовер використовує ту ж саму групу індивідів, вибраних під час етапу елітизму, пари індивідуумів будуть випадковим чином обрані, як батьки, і за допомогою них буде сформовано нове потомство. Оскільки ми маємо справу з вагами нейронної мережі, наступне рівняння буде використано для перехрещення ознак між двома батьками і отриманим потомством. Де a є випадковим чином згенерованим числом між нулем та одиницею. Значення *parent1*, і *parent2* представляють ваги нейронної мережі.

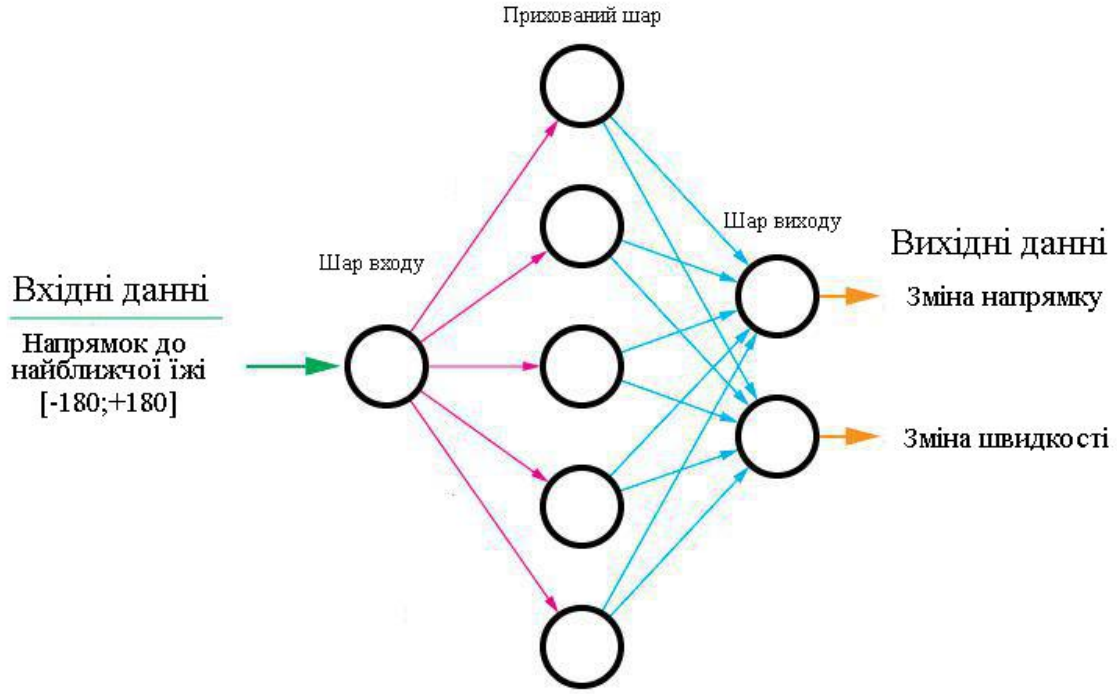


Рисунок 3.2 - Нейронна мережа організмів

Оскільки у нас дві матриці, що об'єднуються (wih і who), реалізація буде наступною:

$$offspring\_with = parent\_with_1\ (a) + parent\_with_2\ Đ$$

$$offspring\_who = parent\_who_1\ (a) + parent\_who_2\ í \quad a)$$

Існують два потенційних підходи при виконанні операцій схрещування в генетичному алгоритмі, його можна виконувати на наступних елементах популяції:

1) Еліти в популяції, ймовірно, ті, які також будуть безпосередньо переходити в наступне покоління
2) Все населення у попляції .

Якщо проблема має велику популяцію, то застосування описаної вище стратегії еліт, швидше за все, призведе до того, що ваш алгоритм ніколи не досягне оптимального рішення або не досягне його після занадто великої кількості ітерацій. Причина цього полягає в тому, що якщо ваша популяція велика, то, оскільки ініціалізація вашої початкової популяції є випадковою, існує ймовірність, що "найбільш придатні" особи (і) в перші кілька ітерацій не обов'язково приведуть вас до оптимального рішення. Тому застосування стратегії еліт занадто рано може змусити вас застрягти на місцевому оптимальному розв’язку назавжди або занадто довго. Якщо популяція відносно невелика, стратегія еліт чудово працює.

Після створення нового потомства, для всіх індивідів генерується випадкове число між нулем та одиницею. Якщо це значення нижче порогу мутації індивіду, відбудеться мутація. У випадку, якщо відбувається мутація, випадкова вага однієї з двох матриць ваги нейронної мережі буде замінена випадковою величиною, яка знаходиться в межах *+/- 10%* від початкового значення. Це приведе до незначних змін в організмів, які потенційно можуть привести до створення нового більш пристосованого організма. Ефект мутації обмежується такими рамками, оскільки ми хочемо уникнути катастрофічної невдачі в нейронній мережі, що може потенційно паралізувати організм.

Останнім важливим елементом моделювання, є створення симуляції. Тривалість симуляції буде визначається діленням загального часу моделювання gen_time (в секундах) на тривалість часу одного етапа; t. Наприклад, якщо довжина моделювання встановлена 100 секунд, а t дорівнює *1/25* секунди, то буде всього 2500 кроків, які потрібно моделювати. Під час кожного кроку часу буде оновлюватися кілька ключових значень.

Виявлення зіткнень . Тут ми перевіряємо зіткнення між організмами та частинками їжі. Коли виявляється зіткнення, цей організм оновлюватиме

свою фітнес-функцію, а частинка їжі відновлюється у новому випадковому місці.

Рисунок 3.3- Точка переходу до другої фази

Визначення найбільш близької частинки їжі. Для кожного організму необхідно визначити найближчу частинку їжі.

Визначення напрямку до найближчої частинки їжі. Як тільки для кожного організму буде визначена найближча частинка їжі, необхідно розрахувати напрямок до цієї частинки.

Використовуючи оновлене (і нормалізоване) значення напрямку, подаємо запит в нейронну мережу в кожному організмі. Виходячи з відповіді нейронної мережі,напрямок, швидкість і позиція оновлюються.

Також розмір популяції буде встановлений у розмірі 50 індивідів. Для популяцій меншого розміру, процес еволюції буде проходити повільніше. Для популяцій більшого розміру час роботи симуляції збільшиться.

Будемо розглядати середнє значення функції пристосованості організмів (AVG), як параметр успішності цілої популяції.

Особливість поведінки генетичних моделей з часом схожа на поведінку макро- та мікроекономічних моделей. У них є дві фази: швидкий розвиток та стабілізація з часом.

Як видно з малюнка 3.3. перші в перших трьох поколіннях популяція різко розвивається. Тобто популяція знаходиться у фазі швидкого розвитку. Після третього покоління популяції переходить у фазу відносної стабілізації.

3.2 Підбір та аналіз параметру елітизму

Розглянемо коефіцієнт елітизму. Як уже було сказано, коефіцієнт елітизму показує, який процент популяції з попереднього покоління продовжить існування в наступному. Візьмемо межі для даного параметру в (0.07,0.45), для нашої моделі.

Найкращий середній результат модель показала для значень елітизму близьких 0.2. Для значень менших даного модель дає гірший середній результат, та найгірший для більших значень.

Інтерпретувати дані результати можна наступним чином. Для великих значень елітизму. Велика частина популяції продовжує існувати в наступних поколіннях. Таким чином ймовірність того, що з’явиться новий успішний індивід падає, і вся популяція стагнує, і втрачає свою гнучкість, що помітно на графіках. А для маленьких значень елітизму , у популяції майже немає успішних індивідів, які б могли продовжити ділитися успішними хромосомами зі своїми нащадками.

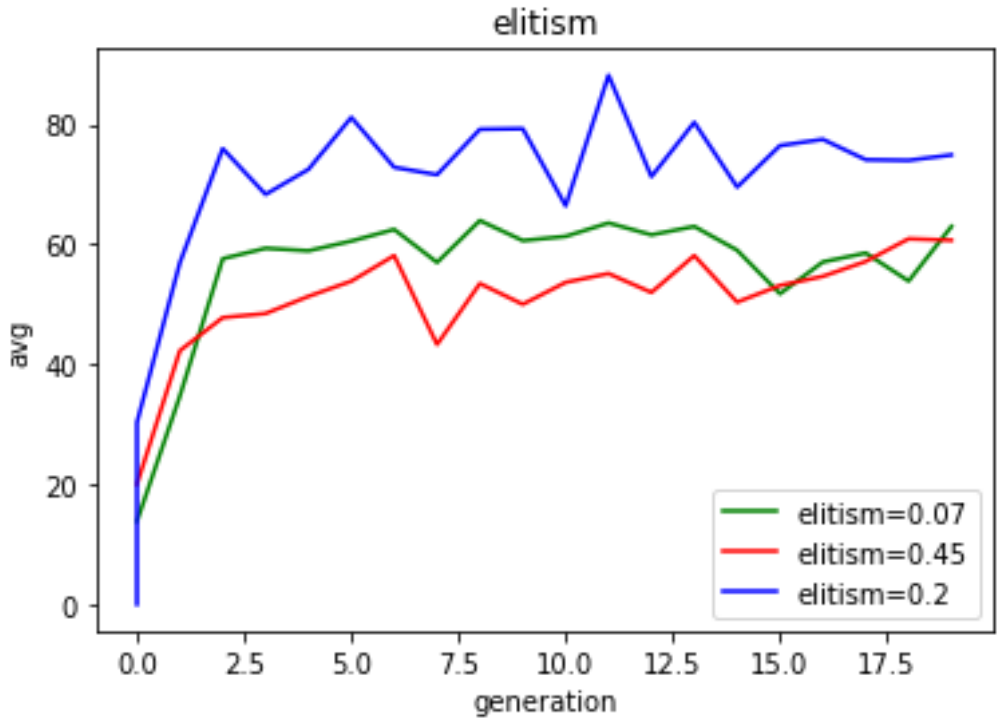


Рисунок 3.4- Середнє значення пристосованості

З ростом значення елітизму момент фазового переходу різниться. Для великих значень елітизму розвиток припиняється набагато швидше ніж для маленьких. Уже на другому поколінні популяції переходить із фази різкого еволюційного розвитку у фазу стабільності. Як уже було зазначено, це викликано тим, що у популяції виділяється великий процент еліти, яка швидко починає домінувати над іншими представниками виду. Така різниця викликана тим, що чим більший коефіцієнт елітизму, тим менша частина популяції підпадає під роботу функцій мутації та схрещування.

Для складніших організмів дана закономірність зберігається рис 3.5. В таблиці 3.1. порівняння середніх значень функцій пристосованості для різних значень функцій елітизму.

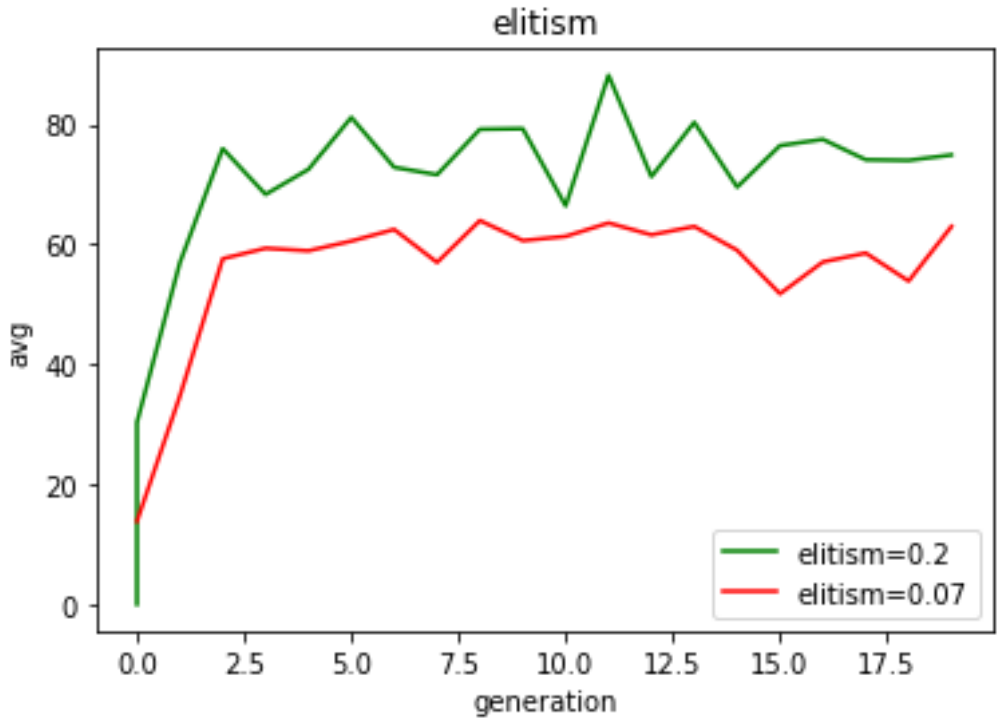


Рисунок 3.5 - Середнє значення пристосованості

Проблема елітизму в тому, що вона змушує ГА сходитися на місцевих максимумах замість глобального максимуму, тому чистий елітизм - це лише гонка до найближчого локального максимуму, і ви отримуюте невелике поліпшення. Це можна виправити кількома способами, наприклад мутувати еліту випадковим чином, тобто ввести ентропію. Це стандартна проблема в алгоритмах пошуку. Локальні максимуми є реальною проблемою для рекурсивної оптимізації в цілому.

Так як елітизм змушує генетичний алгоритм сходитися до локальних максимумів замість глобальних максимумів. В основному елітизм - це лише гонка до найближчих локальних максимумів і, як тільки ви потрапляєте на місцеві максимуми, популяціє перестає розвиватися.

Таблиця 3.1 – Матриця взаємної кореляції змінних.

Gen	Ell=0.45	Ell=0.07	Ell=0.2
1	30.02	23.98	28.76
2	62.86	49.74	62.32
3	62.88	65.8	63.6
4	61.86	67.58	71.6
5	77.6	59.74	81.56
6	69.62	72.02	76.18
7	57.68	73.68	78.96
8	62.08	66.64	75.0
9	65.68	67.18	78.62

3.3 Кількість нейронів у прихованому шарі нейронної мережі

Прихований шар нейронної мережі є проміжним шаром між входним та вихідним шаром. Прихований шар складається з прихованого нейронів. Приховані нейрони - це нейрони, які не є ні у вхідному, ні в вихідному шарі.

Перенасичення:

Якщо в прихованому шарі занадто багато нейронів, це може призвести до перенавчання. Перенавчання виникає, коли в мережі присутні зайві нейрони.

Недостатність:

Якщо число нейронів менше в порівнянні зі складністю проблеми, то великі об’єми даних не можуть бути повністю необробленими. Це

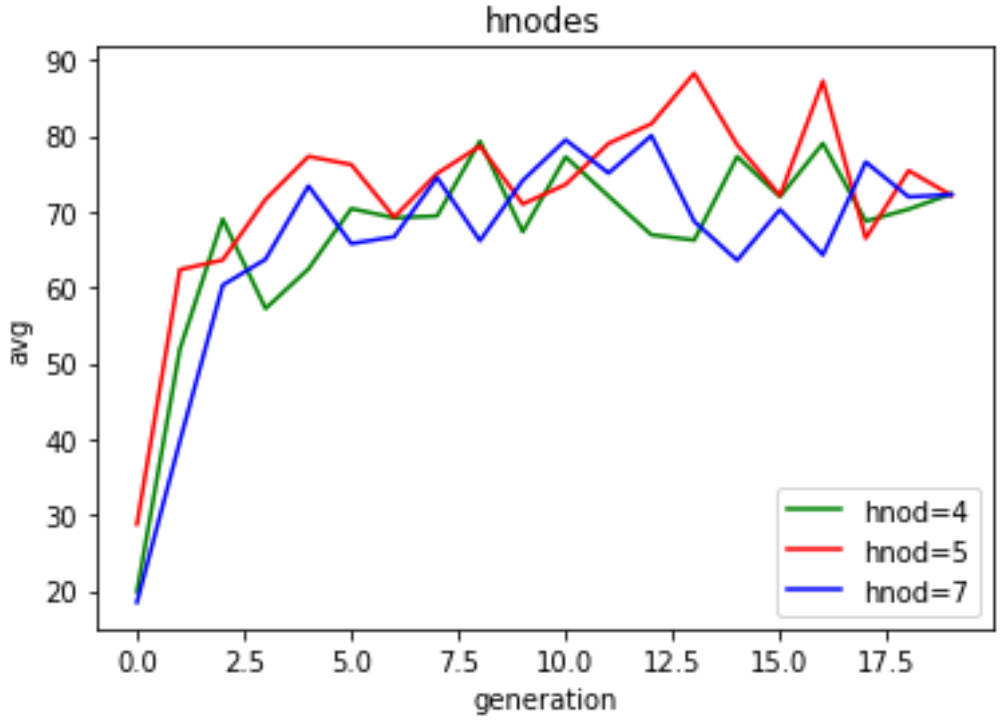


Рисунок 3.6- Середнє значення пристосованості

відбувається тоді, коли в прихованому шарі занадто мало нейронів для аналізу складного набору даних.

Розглянемо параметр кількості нейронів у прихованому шарі нейронної мережі. Зі збільшенням кількості нейронів у прихованому шарі, організми ускладнюються.

Для визначення оптимальної кількості нейронів у прихованому шарі почнемо з найменших кількості нейронів, і продовжимо збільшуючи їх кількість, і розглянемо середня значення функції фітнесу для всіх поколінь з різною кількістю нод. Тобто спробуємо багато мереж з різною кількістю прихованих нод, та оцінимо їх.

Для організмів з малою кількістю нейронів у прихованому шарі. Процес еволюції проходить найменшу кількість кроків. Але нажаль зворотній бік у тому, що середнє значення пристосованості (AVG), буде приймати найменші значення значення. У організмів з кількість прихованих нод рівною чотирьом, кількість ваг нейронної мережі є найменшою. Тому то процес розвитку і проходить так швидко, але це приводить до малого різноманіття у популяції, та малої кількості можливостей для розвитку.

Для найскладніших організмів ситуації протилежна, процес швидкого розвитку проходить найдовше. І при тому, що він проходить протягом 5 поколінь, результати залишаються на середньому рівні. Складніші організми ледь переганяють в середньому по своїй пристосованості найпростіші. Це все викликано збільшенням кількості ваг з'єднань у нейронній мережі.

Іншими словами, кількість нейронів у шарі така, як ви хочете. Чим більше їх, тим складнішими стають організми. Чим складніше організми, тим більше необхідних вхідних даних, і вони більш схильні до перенавчання.

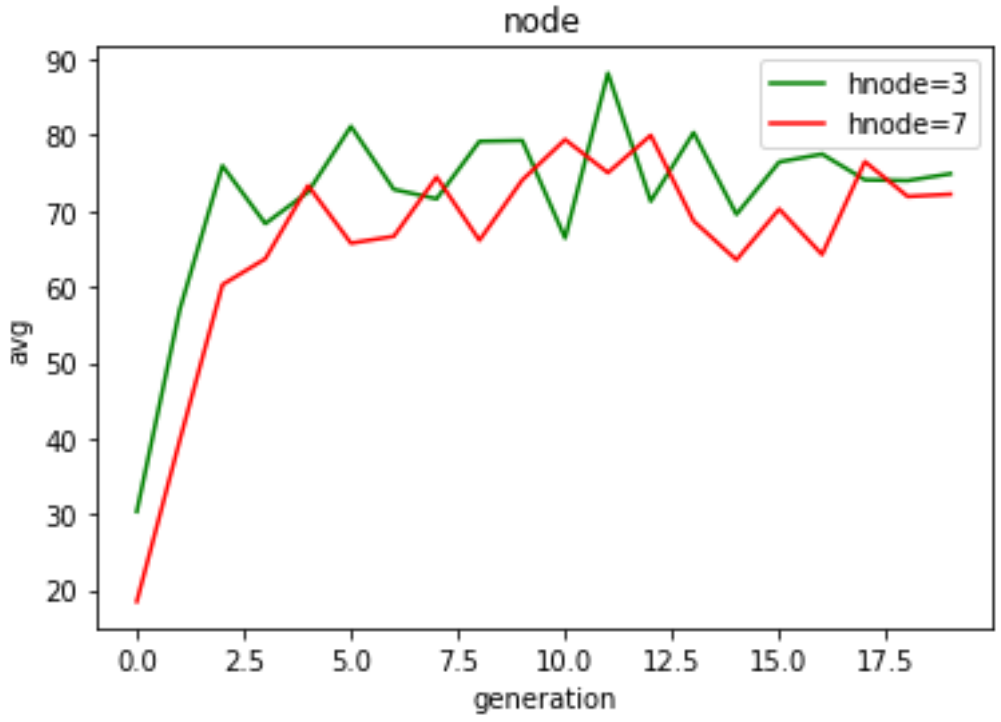


Рис. 3.7 - Середнє значення пристосованості

Таким чином, у виборі складності структури організмів варто дотримуватися балансу, таким чином, щоб організм, достатньо довго розвивався не застряваючи в локальному максимумі і при тому мав найкраще середнє значення функції пристосованості для покоління.

Для нашої моделі п’ять нейронів у прихованому шарі, буде найоптимальнішим варіантом, для швидкого і головне ефективного розвитку організмів, як це видно з рисунка 3.5 та з таблиці 3.2.

Таблиця 3.1 – Матриця взаємної кореляції змінних.

Gen	Node=4	Node=5	Node=7
1	19.84	28.76	18.46
2	51.82	62.32	39.68
3	69.0	63.6	60.28
4	57.2	67.58	63.7
5	62.42	71.6	73.36
6	79.26	72.02	65.76
7	67.32	78.62	66.66
8	62.08	81.56	74.48
9	72.02	75.38	66.14

Для нашої моделі п’ять нейронів у прихованому шарі, та значення елітизму рівним 0.2 буде найоптимальнішим варіантом, для успішного розвитку популяції, та отримання найвищих середніх значень функції пристосованості таблиця 3.3.

Таблиця 3.3 – Матриця взаємної кореляції змінних.

Gen	Best	Worst	AVG	SUM	MORE THEN AVG	LESS THEN AVG	MORE THEN AVG(%)	LESS THEN AVG (%)
1	135	2	28.76	1329	15	35	30.0%	70.0%
2	130	8	62.32	3226	23	27	46.0%	54.0%
3	147	3	63.6	3241	21	29	46.0%	54.0%
4	206	4	71.6	3162	22	28	42.0%	58.0%
5	193	10	81.56	3162	25	25	50.0%	50.0%
6	167	3	76.18	3403	28	22	60.0%	40.0%
7	186	6	78.96	3985	27	23	56.0%	44.0%
8	189	9	75.0	3777	26	24	52.0%	48.0%
9	193	7	78.62	3660	25	25	46.0%	54.0%

3.3. Розмір популяції

Одним факторів для нашої моделі було визначення оптимального розміру популяції. Дослідники зазвичай стверджують, що «маленька» чисельність населення може привести алгоритм до поганих рішень, а велика

чисельність популяції приведе до значно більших витрат часу на роботу алгоритму. Отже у нас компромісом буде, подача достатньої кількості індивідів, для оптимальної роботи алгоритму та отримання хороших результатів.

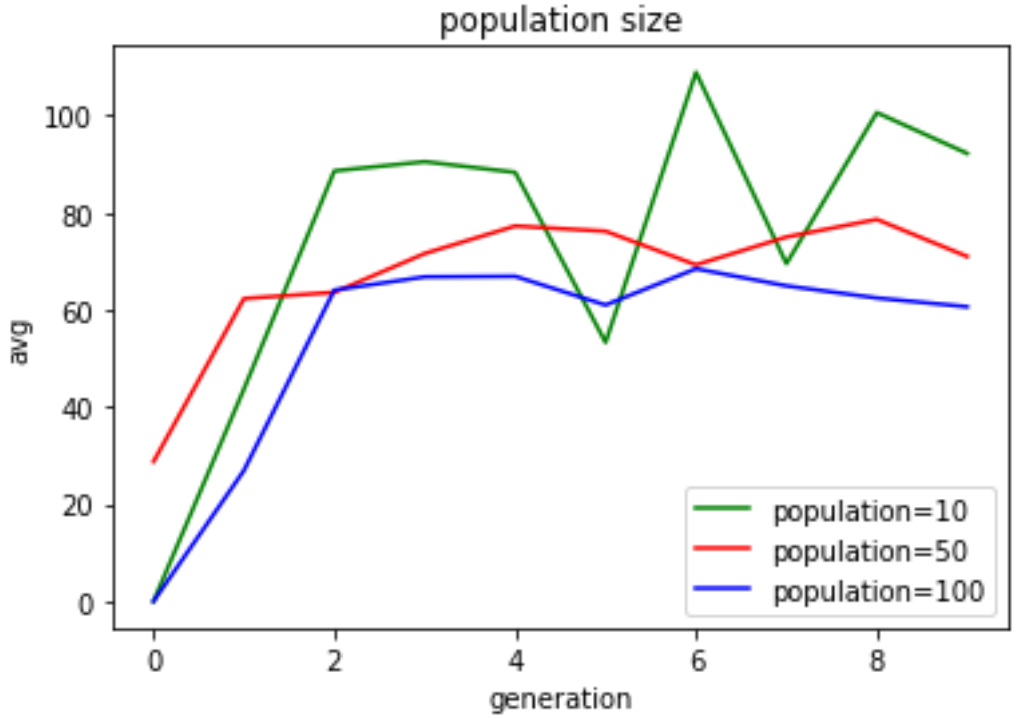


Рис. 3.8- Середнє значення пристосованості

Як правило, на практиці варто задати діапазон численності населення, та обрати ті значення, які будуть найоптимальнішими. Для нашої задачі найоптимальніший розмір популяції рівний п’ятдесяти.

Для популяцій меншого розміру процес еволюції проходить набагато довше, тому що у алгоритму занадто мало інформації, для оптимальної роботи. Також функція пристосованості в середньому дає кращий результат. Але значення графік функції пристосованості в стадії стабільного розвитку показує великі флуктуації викликані малим розміром популяції. Через що ймовірність повністю невдалих поколінь росте.

Для значень більше все навпаки. Функція пристосованості в середньому дає набагато менший результат. Але при тому сам розвиток проходить плавно.

Найоптимальнішим для даної моделі буде значення популяції в п’ятдесят особин. Це значення буде давати непоганий середній результат, та невеликі відхилення від середнього значення функції пристосованісті у плавній фазі розвитку.

Таким чином ми визначили оптимальні значення параметрів та модифікацію поведінки індивідів. В порівнянні з іншими моделями фазовий перехід відбувається набагато швидше.

Висновки до розділу 2

В даному розділі ми реалізували модель мікроеволюційної поведінки групи простих організмів, що описуються нейронною мережею. Поведінка організмів була реалізована за допомогою повністю пов’язаної нейронної мережі. Організми розвиваються та мутують за допомогою генетичного алгоритму (Га). Ваги зв’язків в нейронній мережі були прийняті за хромосоми. І так чином була отримана модель.

Для реалізованої моделі було розглянуто, як модифікується поведінка організмів при внесенні змін в базовий алгоритм та зміні набору параметрів. А саме параметрів елітизму, кількості нейронів в прихованому шарі та розміру популяції.

ВИСНОВКИ

В магістерській дисертації було проаналізовано предметну область, розглянуто існуючі моделі, застосовні для реалізації моделі мікроеволюційної поведінки групи простих організмів. Для опису даних організмів були використані нейронні мережі та Генетичний алгоритм (ГА).

Застосовані алгоритми дозволили отримати модель в , якій кожен організм складається з нейронної мережі, в якій ваги штучних нейронів та з'єднань грають роль генетичного коду. Який в свою чергу оптимізується за допомогою генетичних алгоритмів (ГА).

У першому розділі було основні положення та визначення, що відносять до сфери Еволюційних (ЕА) та Генетичних алгоритмів (ГА). Крім того докладно поданий математичний апарат для побудови ГА. Проаналізовано можливі підходи для реалізації мутації, схрещування та селекції для кожного поняття були отримані результати та висновки щодо найкращого варіанту їх реалізації., з урахуванням особливостей предметної області.

У другому розділі були розглянуті основні положення та визначення, що відносять до сфери нейронних мереж. Докладно розкриті парадигми роботи нейронних мереж. Було розглянуто новітні рішення та підходи до використання нейронних мереж. Виходячи з проведеного огляду

В практичній частині після аналізу предметної області було побудовано модель поведінки простих організмів в середовищі. Поведінка організмів була реалізована за допомогою повністю пов'язаної нейронної мережі. Організми розвиваються та мутують за допомогою генетичного алгоритму.

Аналіз моделі дозволив виявити оптимальні параметри елітизму та кількості нейронів в прихованому шарі. А саме *ell=0,2* та *node=5*. Було досліджено, що оптимальність даного значення елітизму є наслідком

особливостей роботи ГА. А також було запропоновано новий підхід для реалізації елітизму. Даний підхід вимагає мутувати частину еліт для підтримання різноманіття в популяції та попереджує виникнення локальних максимумів. Щодо параметру *node* , було показано, що найоптимальніший та найстабільніший розвиток організмів відбувається при значенні *node=5*.

Робота відзначається використанням комбінації методів нейронних мереж та генетичних алгоритмів для їх оптимізації. Запропонована модель підходить для моделювання простих живих організмів та може бути застосована у реальних проектах.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1 Голдберг Д. Е. Генетичні алгоритми оптимізації та машинне навчання / Д. Е. Голдберг. – 84 с.

2 Мічалевіц З. Генетичні алгоритми+структура данних=еволюційна програма / З. Мічалевіц. – 94 с.

3 Гаскаров Д. В. Мала вибірка / Д. В. Гаскаров, В. І. Шаповало., 2000.

4 Хаупт Р. Л. Практичне використання генетичних алгоритмів / Р. Л. Хаупт, С. Е. Хаупт.. – 67 с.

5 Калоній Д. Мультиоб'єктивна оптимізація з використанням еволюційних алгоритмів / Д. Калонмій.. – 31 с.

6 Sivanandam S. N. Introduction to Genetic Algorithms / S. N. Sivanandam.. – 318 с.

7 Altenberg L. The Schema Theorem and Price’s Theorem. Foundations of genetic algorithms / L. Altenberg.. – 73 с.

8 Mallawaarachchi V. Introduction to Genetic Algorithms—Including Example Code [Електронний ресурс] / Vijini Mallawaarachchi – Режим доступу до ресурсу: https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3.

9 Карпатний А. Рецепт для навчання нейронних мереж / Карпатний Андрій. 2018. – 364с.

10 Александер І. Вступ до нейронних обчислень / І. Александер, Х. Мортон.. – 31 с. – (2-ге видання).

11 Двало Е. Нейронні мережі / Е. Двало. 1987. – 239 с.

12 Алкон Д. Л. Нейронні системи та память / Д. Л. Алкон., 1989. – 45 с.

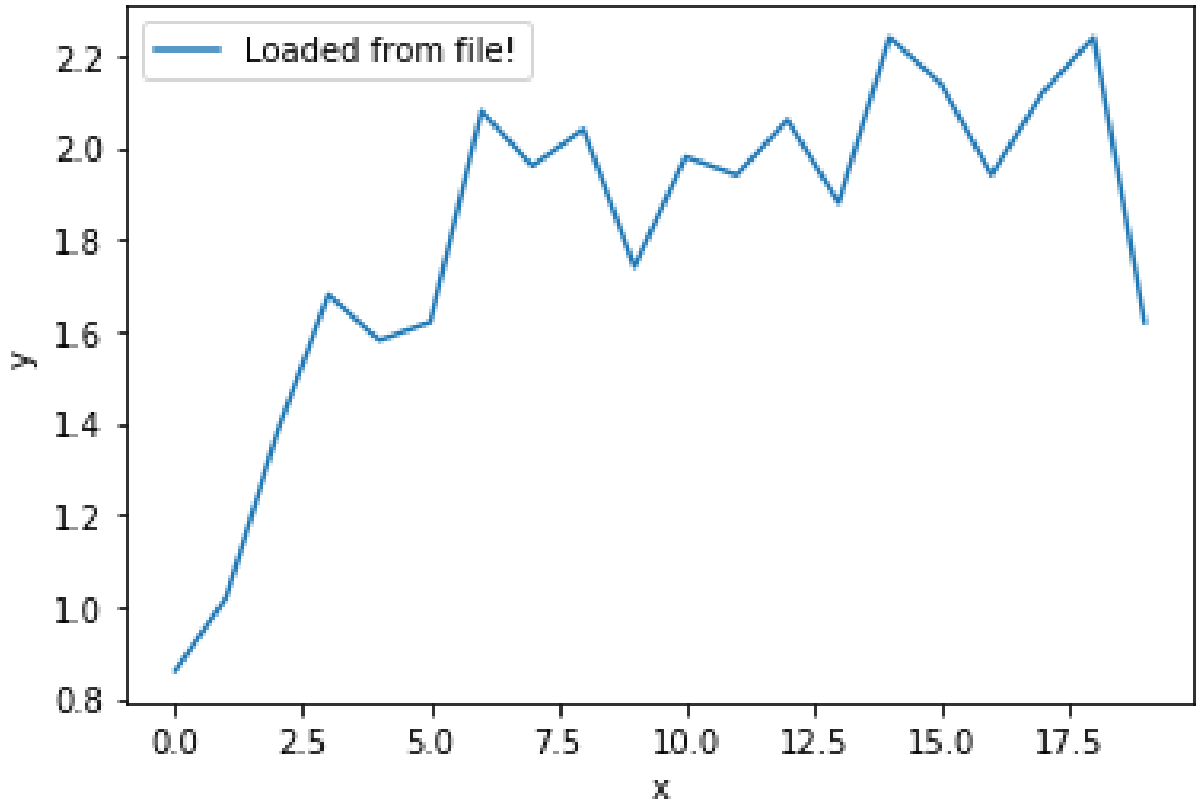
13 Шанар Ш. Функція активації в нейронних мереж [Електронний ресурс] / Ш. Шанар – Режим доступу до ресурсу: https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6.

14 Донас Н. Нейронні мережі [Електронний ресурс] / Н. Донас – Режим доступу до ресурсу: https://towardsdatascience.com/recurrent-neural-networks-and-lstm-4b601dd822a5.

15 Рой Н. Еволюція простих організмів [Електронний ресурс] / Н. Рой – Режим доступу до ресурсу: https://nathanrooy.github.io/posts/2017-11-30/evolving-simple-organisms-using-a-genetic-algorithm-and-deep learning/#%D0%9A%D0%9F%D0%86%D0%92%D0%95%D0%96%D0%90.

ДОДАТКИ

ДОДАТОК А Графік роботи алгоритму для функції входу sigm



ДОДАТОК Б Графік роботи алгоритму для різних параметрів