Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____
(повна назва)

Кафедра _____програмної інженерії_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____другий (магістерський)_____

_____Дослідження методів ШІ для планування ІТ-проектів
(тема)

Виконав:
студент _2_ курсу, групи _ІПЗм-22-3____
Москаленко Михайло Володимирович
(прізвище, ім'я, по батькові)
Спеціальність ____121 Інженерія програмного забезпечення
(код і повна назва спеціальності)
Тип програми ____освітньо-наукова_____
(освітньо-професійна або освітньо-наукова)

Керівник проф. каф. Качко О.Г._____
(посада, власне ім'я, прізвище)

Допускається до захисту

Зав. кафедри _____ __З.В. Дудар
(підпис) (власне ім'я, прізвище)

2024 р.

Харківський національний університет радіоелектроніки

Факультет _____Комп'ютерних наук_____

Кафедра _____Програмної інженерії_____

Рівень вищої освіти _____другий (магістерський)_____

Спеціальність ____121 - Інженерія програмного забезпечення
(код і повна назва)

Тип програми _____освітньо-наукова_____
(освітньо-професійна або освітньо-наукова)

Освітня програма_121 Інженерія програмного забезпечення__
(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

«_01__»__квітня__ 20 24_ р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові___Москаленку Михайлу Володимировичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи __Дослідження методів ШІ для планування ІТ проектів затверджена наказом університету від _01 квітня_2024_ р. № _258Cm
2. Термін подання студентом роботи до екзаменаційної комісії 03.06.2024 р.
3. Вихідні дані до роботи _Інтернет джерела з тематики кваліфікаційної роботи та науково-технічні публікації _____
4. Перелік питань, що потрібно опрацювати в роботі _аналіз використання методів штучного інтелекту в плануванні ІТ-проєктів; дослідження модифікації генетичного алгоритму для розв'язання задачі планування ІТ-проєктів; реалізація гібридного алгоритму для розв'язання задачі планування ІТ-проєктів; апробація результатів кваліфікаційної роботи
_____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Оцінка сучасного стану об’єкта дослідження	01.05.2024 – 06.04.2024	виконано
2	Огляд існуючих методів вирішення проблеми планування ІТ-проєктів	06.05.2024 – 07.06.2024	виконано
3	Огляд існуючих варіантів вирішення проблеми планування ІТ-проєктів	08.05.2024 – 09.05.2024	виконано
4	Огляд і аналіз існуючих новітніх технологій та їх алгоритмів вирішення проблеми планування ІТ-проєктів	10.05.2024 – 15.05.2024	виконано
5	Постановка задачі дослідження	16.05.2024 – 17.05.2024	виконано
6	Аналіз використання методів штучного інтелекту в плануванні ІТ-проєктами	18.0452024 – 24.05.2024	виконано
7	Дослідження модифікації генетичного алгоритму для розв’язання задачі планування ІТ-проєктів	25.05.2024 – 30.05.2024	виконано
8	Реалізація гібридного алгоритму для розв’язання задачі планування	01.06.2024 – 08.06.2024	виконано
9	Апробація результатів кваліфікаційної роботи	09.06.2024 – 15.06.2024	виконано
10	Оформлення пояснювальної записки	16.06.2024 – 20.06.2024	виконано
11	Розробка презентації	21.06.2024	виконано
12	Захист роботи	25.06.2024	виконано

Дата видачі завдання __01__ ___квітня__2024_ р.

Студент _____
(підпис)

Керівник роботи _____ _проф. каф. ПІ Олена КАЧКО
(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи: 95 сторінок, 15 рисунків, 10 таблиць, 20 формул, 3 додатки, 34 джерел.

НЕЙРОННА МЕРЕЖА, МАШИННЕ НАВЧАННЯ, ШТУЧНИЙ ІНТЕЛЕКТ, ДОСЛІДЖЕННЯ, ПЛАНУВАННЯ ІТ-ПРОЄКТІВ.

Об'єктом дослідження цієї кваліфікаційної роботи є гібридний алгоритм для оптимізації процесу планування ІТ-проєктів з обмеженими ресурсами.

Мета роботи полягає у вивченні доцільності та ефективності застосування гібридного алгоритму для вирішення проблеми планування. У ході дослідження використовувалися методи спостереження, наукові факти, порівняння, експеримент та аналіз.

У результаті роботи було запропоновано новий підхід до вирішення проблеми планування ІТ-проєктів з обмеженими ресурсами, зокрема, гібридний алгоритм, що поєднує генетичний алгоритм та самоорганізаційні карти Кохонена.

Проведене експериментальне дослідження на тестових даних показало обмежене покращення результатів. Цей напрямок має перспективи для подальшого дослідження і потребує детальнішого аналізу з метою оптимізації вхідних параметрів для підвищення ефективності роботи гібридного алгоритму.

ABSTRACT

Explanatory note to the qualification work: 95 pages, 15 pictures, 10 tables, 20 formulas, 3 appendics, 34 sources.

NEURON NETWORK, MACHINE LEARNING, ARTIFICIAL INTELLIGENCE.

The object of research of this qualification work is the hybrid algorithm for optimizing the planning process of IT projects with limited resources.

The purpose of the work is to study the expediency and effectiveness of using the hybrid algorithm to solve the planning problem. The research used methods of observation, scientific facts, comparison, experiment and analysis.

As a result of the work, a new approach to solving the problem of planning IT projects with limited resources was proposed, in particular, a hybrid algorithm combining a genetic algorithm and Kohonen self-organizing maps.

An experimental study conducted on test data showed a limited improvement in results. This direction has prospects for further research and requires a more detailed analysis in order to optimize the input parameters to increase the efficiency of the hybrid algorithm.

Заява щодо самостійного виконання кваліфікаційної роботи та можливості її публікації в електронному архіві відкритого доступу ElArKhNURE.

Я, Москаленко Михайло Володимирович, студент(ка) гр. ІПЗм-22-3, здобувач вищої освіти на другому (магістерському) рівні кафедри «Програмна інженерія», заявляю: моя кваліфікаційна робота на тему «Дослідження методів ШІ для планування ІТ-проектів», що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в електронному архіві відкритого доступу ElArKhNURE. Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений(на) з діючим положенням «Про протидію академічному плагіату в ХНУРЕ», згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ГА – генетичний алгоритм

ІТ – інформаційні технології

МН – машинне навчання

НМ – нейронні мережі

ОПМ – обробка природної мови

ШІ – штучний інтелект

CCM – critical chain method (метод критичного ланцюга)

CPM – critical path method (метод критичного шляху)

LFT – latest finish time

PMBOK – project panagement body of knowledge

RCPSP – resource-constrained project scheduling problem

SOM – self-organizing map

ВСТУП

У сучасному динамічному бізнес-середовищі більшість проєктів характеризуються високою складністю та потребують залучення значної кількості ресурсів. Ефективна організація проєкту передбачає оптимальний розподіл трудових і матеріальних ресурсів, що є суттєвим викликом для проєктних менеджерів при формуванні плану проєкту. Особливо гостро ця проблема виникає у сфері інформаційних технологій, де проєкти мають специфічні вимоги й потребують спеціалізованих підходів до планування та управління. Проблема планування проєктів є однією з ключових задач у проєктному менеджменті.

Залежно від пріоритетів і цілей проєкту, задача оптимізації може включати мінімізацію загальної тривалості проєкту або оптимальний розподіл ресурсів між завданнями. Незважаючи на різноманітність критеріїв оптимізації, процес планування залишається складним, потребуючи застосування ефективних методів і не маючи універсального вирішення.

Для підвищення ефективності розв'язання задачі планування проєктів пропонується дослідити можливість інтеграції самоорганізаційних карт Кохонена з генетичним алгоритмом. Використання SOM у поєднанні з генетичним алгоритмом має потенціал покращити ефективність пошуку оптимальних рішень і запобігти передчасній збіжності алгоритму до локальних оптимумів.Кваліфікаційна робота виконується згідно з державними стандартами [1]-[2].

1 АНАЛІЗ ВИКОРИСТАННЯ МЕТОДІВ ШТУЧНОГО ІНТЕЛЕКТУ В ПЛАНУВАННІ ІТ-ПРОЄКТІВ

1.1 Огляд процесів планування ІТ-проєктів

У сфері управління проєктами інформаційних технологій (ІТ) ефективне планування є ключовим чинником успіху. Успішний проєкт починається з правильного планування, яке включає детальний аналіз та стратегічний підхід до визначення обсягу робіт, ресурсів, часових рамок та вимог до якості.

Планування проєктів стикається з низкою викликів, зокрема з проблемою встановлення плану робіт в умовах обмежених ресурсів. Існуючі стандарти планування, такі як Project Management Body Of Knowledge (PMBOK), ISO 15288 та ISO 12207, надають цінні рекомендації щодо ефективного управління проєктами, але вони пропонують лише загальні настанови і не вирішують конкретні проблеми, що виникають під час планування.

Розвиток штучного інтелекту відкриває нові можливості для подолання цих викликів і підвищення ефективності планування проєктів. Розглянемо, як процеси планування ІТ-проєктів організовані згідно із загальними рекомендаціями.

Планування ІТ-проєктів вимагає чіткого визначення мети, цілей, обсягу робіт та основних зацікавлених сторін на початковому етапі. Стандарти ISO 15288 та ISO 12207 встановлюють рамки для різних процесів життєвого циклу систем і програмного забезпечення. Процес планування проєкту, згідно з цими стандартами, включає розробку та координацію ефективних планів, визначення проєктних цілей та обмежень,

управління ресурсами, встановлення структури розподілу робіт на основі системної архітектури та планування технічного управління.

Інтеграція сучасних інструментів штучного інтелекту, таких як самоорганізаційні карти Кохонена і генетичні алгоритми, може значно покращити процеси планування, забезпечуючи більш точне і гнучке управління ресурсами та завданнями, що дозволяє уникнути передчасної збіжності до локальних оптимумів і підвищує загальну ефективність проєктів.

Успішне управління IT-проєктами також вимагає інтеграції планування ризиків на ранньому етапі. Ефективне управління ризиками дозволяє мінімізувати можливий негативний вплив на проєкт і його цілі. Регулярний перегляд прогресу проєкту та його відповідності встановленим цілям допомагає вчасно виявляти та коригувати відхилення, забезпечуючи адаптивність і гнучкість управління проєктом.

Усі ці елементи разом формують основу для створення стійкого та успішного IT-проєкту. Ефективне планування, управління ресурсами та ризиками, а також використання сучасних інструментів штучного інтелекту забезпечують цілісність і успіх проєкту, роблячи його більш стійким до викликів і непередбачуваних обставин.

На рисунках 1.1 та 1.2 наведено процеси планування проєкту та їх декомпозицію, відповідно до поставленої задачі дослідження у даній роботі.

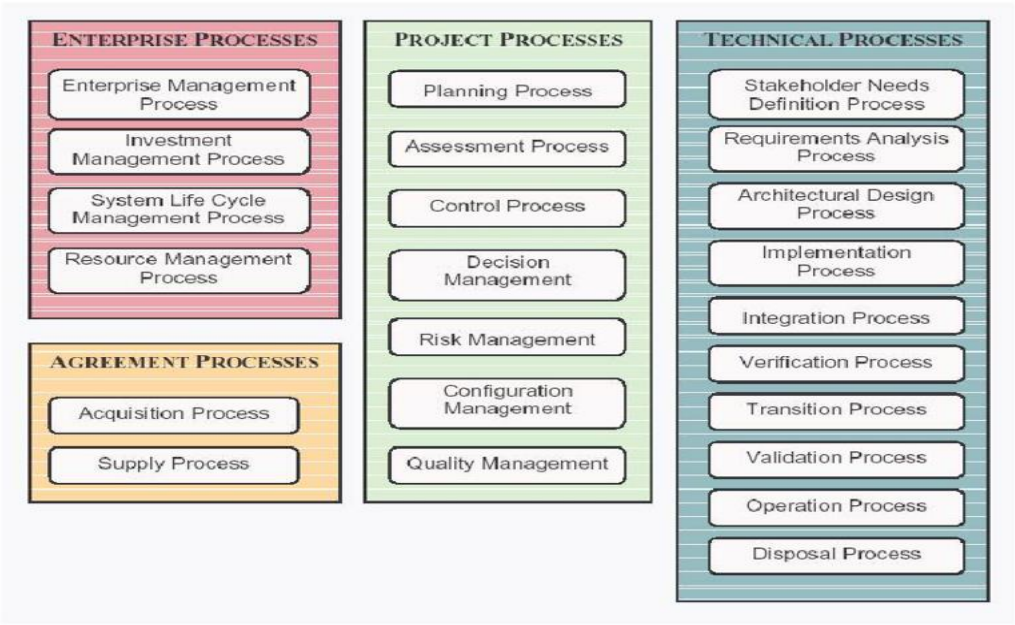

Рисунок 1.1 – Процеси планування проєкту відповідно до стандарту ISO 15288 (за даними [4])

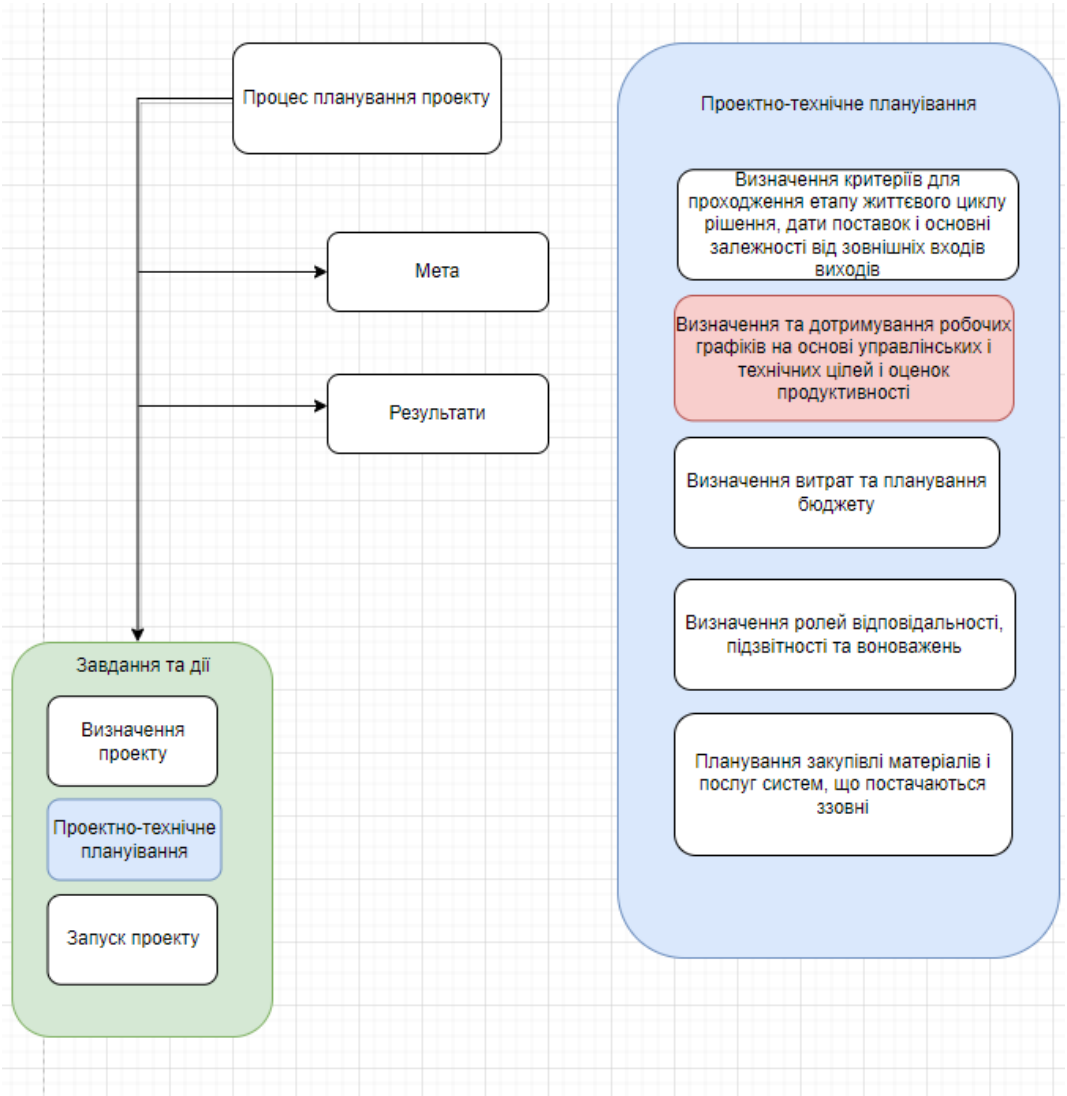

Рисунок 1.2 – Декомпозиція процесів планування проєкту відповідно до стандарту ISO 15288 (виконано самостійно)

У сфері планування проєктів інформаційних технологій методології, визначені стандартами ISO 15288 та ISO 12207, є типовими структурами, що пропонують комплексний підхід до управління життєвим циклом системи та програмного забезпечення відповідно. ISO 15288 зосереджується на проєктуванні систем, представляючи структурований процес, який охоплює всі фази життєвого циклу системи, від концепції до утилізації. Він підкреслює важливість інтеграції різних

дисциплін і процесів для забезпечення ефективної та результативної доставки системи.

ISO 12207, з іншого боку, орієнтований на життєвий цикл програмного забезпечення, забезпечуючи детальну структуру для планування, управління, розробки та підтримки програмних продуктів. Обидва стандарти встановлюють чіткі рамки для організації процесів, що допомагають у визначенні обсягу робіт, ресурсів, часових рамок та якості.

Ключовим аспектом цих методологій є інтеграція управління ризиками на ранньому етапі. Ідентифікація, аналіз та розробка стратегій реагування на потенційні ризики дозволяють мінімізувати їхній негативний вплив на проєкт. Регулярний перегляд прогресу проєкту та його відповідності встановленим цілям забезпечує вчасну ідентифікацію відхилень і їх корекцію, що підвищує адаптивність та гнучкість управління проєктом.

У сучасних ІТ-проєктах застосування штучного інтелекту, таких як самоорганізаційні карти Кохонена та генетичні алгоритми, може значно покращити процеси планування та управління. Ці інструменти сприяють точнішому та гнучкішому управлінню ресурсами і завданнями, допомагаючи уникнути передчасної збіжності до локальних оптимумів і підвищуючи загальну ефективність проєктів.

Загалом, комплексне планування, що включає управління ресурсами, ризиками та використання сучасних технологій, створює основу для успішного виконання ІТ-проєктів.

Обидва стандарти виступають за процесний підхід до управління проєктом, підкреслюючи важливість чітко визначених процесів, ролей і відповідальності протягом життєвого циклу проєкту. Їх об’єднує спільна мета —

забезпечення якості, надійності та ефективності реалізації IT-проєктів. У той час як ISO 15288 забезпечує широку структуру, застосовну до будь-якого системного проєкту, ISO 12207 спеціально адаптується до нюансів розробки програмного забезпечення, пропонуючи детальні вказівки щодо управління складнощами, властивими проєктам програмного забезпечення [3-4].

Отже, стандарт ISO 12207 є доповненням до ISO 15288 орієнтуючись на проєкти з розробки інформаційних технологій. Відповідність багатьох пунктів між цими стандартами очевидна та це поширюється на процес формування графіку робіт, який розглядається у рамках цієї роботи.

1.2 Аналіз методів планування проєктів по часу та ресурсам

У управлінні IT-проєктами, методи оптимізації графіків відіграють ключову роль, дозволяючи керівникам проєктів максимально ефективно використовувати доступний час і ресурси. У швидкоплинному технологічному середовищі, де гнучкість і швидкість є критичними, точне планування і оптимізація робочих процесів стають вирішальними факторами успіху. Ці методи допомагають забезпечувати вчасне виконання проєктів і відповідність бюджетним обмеженням, а також сприяють підвищенню якості кінцевого продукту через оптимальне використання ресурсів і ефективне управління ризиками.

1.2.1 Метод критичного шляху

Метод критичного шляху є потужним інструментом для управління проєктами, що дозволяє керівникам проєктів не лише планувати і координувати складні проєкти, але й забезпечувати їх своєчасне завершення. Використання сприяє підвищенню ефективності управління проєктами та допомагає уникати затримок і перевитрат, забезпечуючи високу якість кінцевого продукту. [**Error! Reference source not found.**].

При застосуванні методу критичного шляху (CPM) спочатку визначаються всі завдання, які потрібно виконати в рамках проєкту, і оцінюється їхня тривалість. На основі цієї інформації створюється мережевий графік, який показує взаємозв'язки між різними завданнями. Цей графік допомагає визначити критичний шлях — послідовність завдань, що безпосередньо впливають на кінцеву дату завершення проєкту. [**Error! Reference source not found.**].

На рисунку 1.3 показано критичний шлях, позначений червоним кольором, який проходить через завдання F і G. Ці завдання формують критичний шлях, оскільки будь-яке затримання у їх виконанні призведе до затримки усього проєкту. Завдання на критичному шляху не мають запасу часу, їх "slack" або "float" дорівнює нулю, як показано на рисунку. Завдання, які не входять до критичного шляху, мають певний запас часу, що дозволяє їм затримуватися без впливу на загальний графік проєкту..

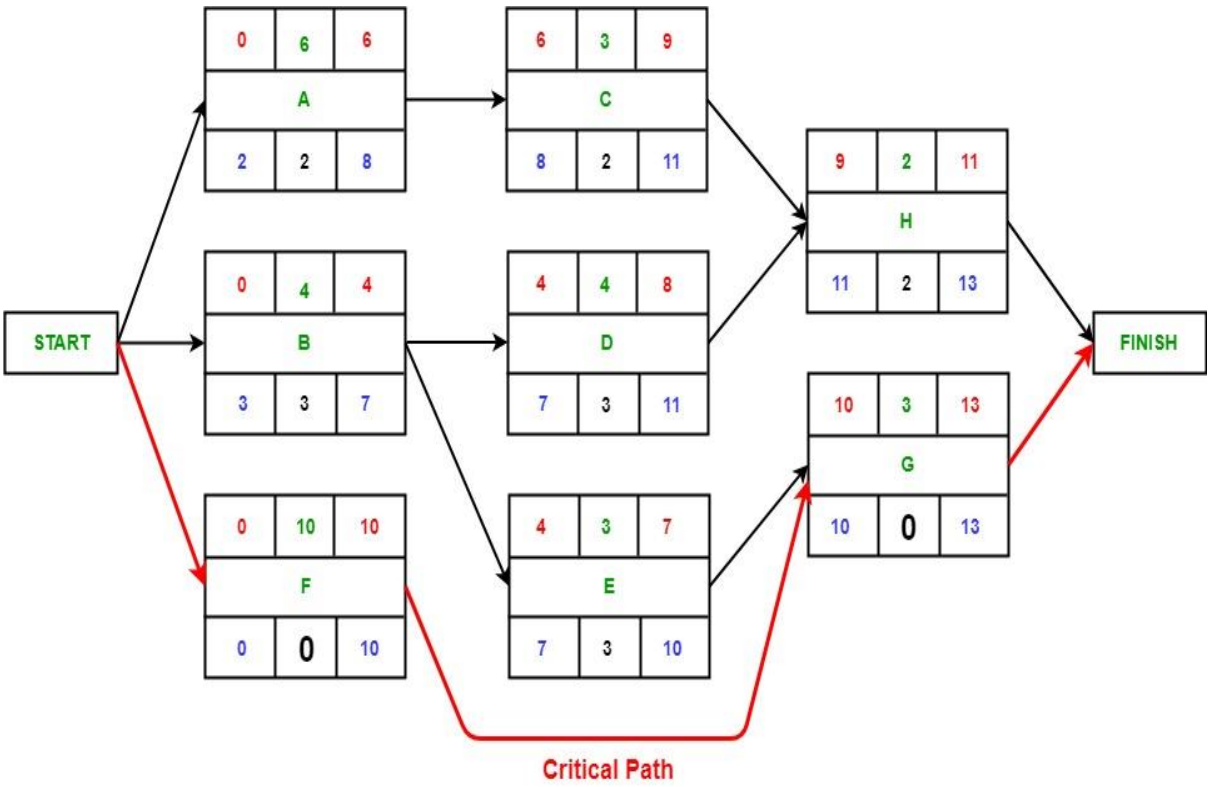

Рисунок 1.3 – Метод критичного шляху (за даними [31])

1.2.2 Метод критичних ланцюгів

Метод критичних ланцюгів (CCM) — це підхід до управління проєктами, що акцентує увагу на ресурсах, необхідних для виконання завдань проєкту. Він спрощує процес та допомагає менеджерам знижувати часові та витратні витрати [10].

CCM ідентифікує критичний ланцюг завдань, які впливають на тривалість проєкту, аналогічно до критичного шляху в CPM. Однак відрізняється від нього тим, що CCM враховує не лише послідовність завдань, а й доступність та використання ресурсів, необхідних для їх виконання. Цей підхід

дозволяє здійснювати більш реалістичні оцінки тривалості завдань, враховуючи можливі затримки через обмеження ресурсів.

Крім того, CCM вводить концепцію буферів часу, які додаються до не критичних завдань та кінця проєкту для забезпечення гнучкості у випадку змін або непередбачених затримок. Це створює "захисний шар" для графіка проєкту, що дозволяє керувати ризиками більш ефективно.

На рисунку 1.4 показано приклад мережевої діаграми з критичним ланцюгом. Він відображає критичний ланцюг, послідовність завдань, які йому відповідають, а також буфери часу. Завдання на критичному ланцюгу є тими, що безпосередньо впливають на загальний графік проєкту, і будь-яка затримка в цих завданнях може призвести до затримки всього проєкту.

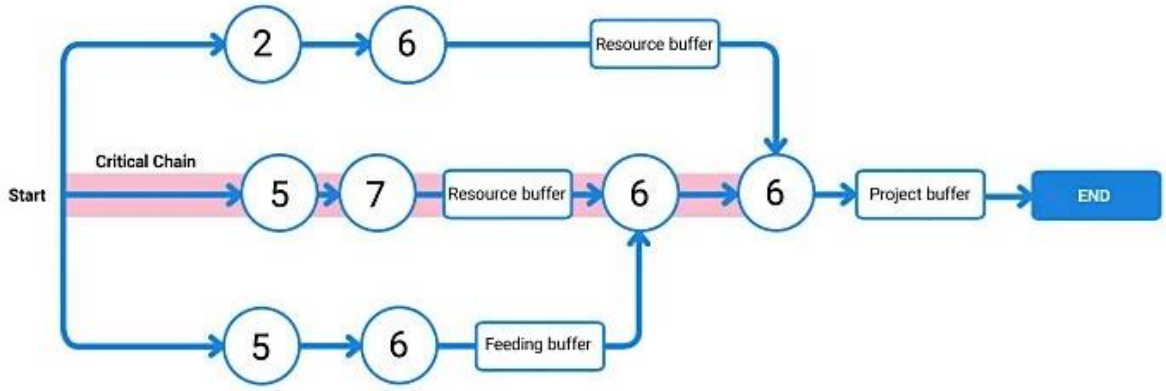

Рисунок 1.4 – Метод критичних ланцюгів (за даними [30])

1.2.3 Генетичні алгоритми

Зусилля керівництва щодо планування та контролю графіків часто затьмарюються неадекватністю координації та розподілу ресурсів у плануванні.[10] Задля запобігання таких обставин слід розглянути інші, сучасні методи, які можуть задовільнити потреби проєктного менеджменту у певних умовах. До таких методів відносяться генетичні алгоритми (ГА) — це методи пошуку та оптимізації, які наслідують принципи природного відбору та спадковості.

На рисунку 1.5 представлено умовну схему генетичного алгоритму.

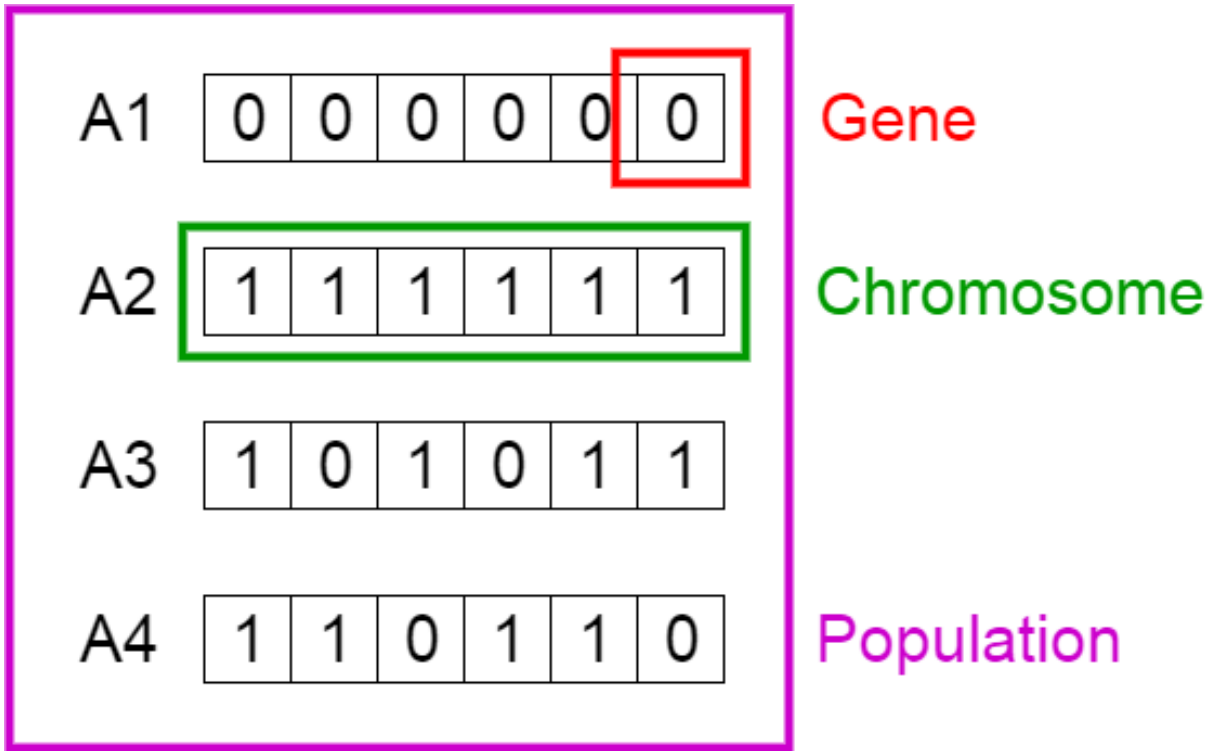

Рисунок 1.5 –Умовна схема генетичного алгоритму (виконано самостійно)

Генетичні алгоритми в контексті оптимізації графіків проєктів можуть бути використані для виявлення найбільш ефективного порядку завдань, розподілу ресурсів або створення

графіка, що забезпечить виконання проєкту за мінімальний час або з мінімальними витратами.

Кожен рядок у такому випадку може представляти «хромосому» або конкретне рішення, де «гени» (елементи хромосоми) визначають параметри рішення.

У «популяції», що складається з багатьох таких хромосом, відбувається процес еволюції, в результаті якого відбираються найкращі хромосоми, їх схрещування та мутація для створення нових поколінь рішень, які краще адаптовані до вирішення задачі.

Через багато таких поколінь алгоритм збільшує шанси знаходження оптимального або наближеного до оптимального рішення.

Таблиця 1.1 – Порівняння методів оптимізації графіків проєктів (виконано самостійно)

Критерій	CPM	CCM	Генетичні алгоритми
Базовий принцип	Планування за найбільш тривалою послідовністю завдань	Планування з урахуванням обмежень ресурсів та введенням часових буферів	Використання імітації природного відбору для знаходження оптимальних рішень

Кінець таблиці 1.1

Застосування	Прості та середньо складні проекти	Складні проекти зі значними обмеженням и на ресурси	Проекти з високою складністю та динамікою
Складність впровадженн я	Низька	Середня	Висока
Гнучкість	Низька	Середня	Висока
Оптимізація	Фокусується на оптимізації часу	Фокусується на використанні ресурсів.	Декілька параметрів
Управління ризиками	Має обмеження у врахуванні невизначеност і	Обмежено, у центрі уваги — ресурси	Висока адаптивніст ь

Критерій	CPM	CCM	Генетичні алгоритми

Застосування	Прості та середньо складні проекти	Складні проекти зі значними обмеженнями на ресурси	Проекти з високою складністю та динамікою
Базовий принцип	Планування за найбільш тривалою послідовністю завдань	Планування з урахуванням обмежень ресурсів та введенням часових буферів	Використання імітації природного відбору для знаходження оптимальних рішень
Складність впровадження	Низька	Середня	Висока
Гнучкість	Низька	Середня	Висока
Оптимізація	Фокусується на оптимізації часу	Фокусується на використанні ресурсів.	Декілька параметрів
Управління ризиками	Має обмеження у врахуванні невизначеності	Обмежено, у центрі уваги — ресурси	Висока адаптивність

Аналіз таблиці показує, що CPM найбільш підходить для проєктів, де важливо дотримуватися фіксованого графіку та коли зміни і ресурси передбачувані. CCM є більш відповідним для проєктів з великими ресурсними обмеженнями і потребою у вищій гнучкості у плануванні. Генетичні алгоритми є ефективними для складних проєктів, де необхідно швидко адаптуватися до змін і де потрібен глибокий аналіз для знаходження оптимальних рішень.

1.2.4 Аналіз і критика Project Management Body Of Knowledge

Project Management Body Of Knowledge (PMBOK) визнаний золотим стандартом в управлінні проєктами, пропонуючи кращі практики та рекомендації, які охоплюють усі аспекти проєкту. Втім, складність та бюрократичність PMBOK часто стають предметом критики, особливо в контексті швидкоплинних та гнучких проєктів. Цей стандарт може виявитися складним для адаптації у малих організаціях або в умовах, що швидко змінюються.

На рисунку 1.6 наведено життєвий цикл проєкту.

Рисунок 1.6 – Стандарт керування проєктами PMBOK (за даними [33])

У сучасних умовах існує потреба в інтеграції технологій, щоб подолати розрив між вимогами сучасного середовища управління проєктами, що є стабільним і конкурентоспроможним на бізнес-ринку, та можливостями, які пропонують традиційні інструменти планування управління проєктами.

Генетичні алгоритми становлять альтернативний підхід, що дозволяє адаптуватися до змін і оптимізувати процеси управління проєктами. Використовуючи принципи еволюції, такі алгоритми здатні об'єднувати різні задачі проєкту в єдиний оптимізаційний процес та будувати максимально ефективний план реалізації.

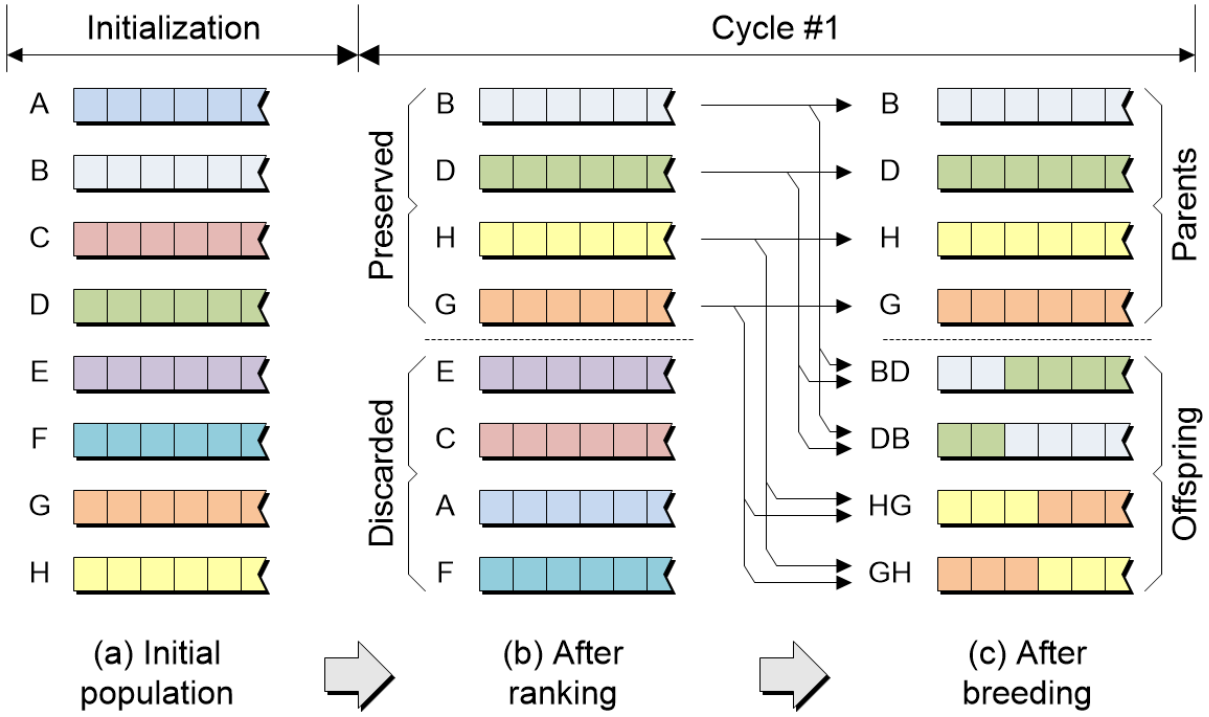

Рисунок 1.7 – Генетичний алгоритм (за даними [34])

Недоліки рекомендацій PMBOK проявляються в їх обмеженій гнучкості для динамічного управління проєктами. PMBOK акцентує увагу на строгій структурі та послідовності етапів проєкту, що може ускладнити здатність команди швидко реагувати на зміни в умовах проєкту. У той же час, методи, засновані на генетичних алгоритмах, дозволяють інтегрувати гнучкість і адаптивність у процеси планування, виходячи за межі традиційних підходів, що призводить до більш ефективного управління проєктами.

1.3 Огляд сучасних засобів для планування IT-проєктів

1.3.1 Визначення поняття «штучний інтелект» та його класифікація

Штучний інтелект — це галузь комп'ютерних наук, що зосереджена на створенні систем, які здатні виконувати завдання, що зазвичай потребують людського розуму. Такі завдання можуть включати в себе вивчення, розуміння мови, розпізнавання образів та прийняття рішень. Метою штучного інтелекту є не лише імітація людської поведінки, але й здатність до самонавчання та адаптації за допомогою алгоритмів машинного навчання [30].

Штучний інтелект поділяється на два основних типи: слабкий і сильний. Слабкий штучний інтелект, також відомий як вузький, спроможний автоматизувати конкретні завдання, зазвичай перевершуючи людей в цих областях, але діючи в обмежених рамках. Сильний штучний інтелект, або загальний, описує моделі, які здатні імітувати людське навчання та мислення на більш глибокому рівні.

Штучний інтелект використовує різноманітні техніки для вирішення завдань, наприклад:

– машинне навчання (ML) – алгоритми, які дозволяють комп'ютерам навчатися на основі даних;

– обробка природної мови (NLP) – технології, спрямовані на взаємодію між комп'ютером та людиною за допомогою природної мови;

– нейронні мережі – представляють собою набір алгоритмів, які оброблюють дані імітуючи структуру людського мозку.

– комп’ютерне бачення – машина на вході отримує зображення, відео чи візуальні медіа, зчитує інформацію з них, виокремлюючи лише те, що стосується поставленої задачі.

Отже, спектр задач, які ШІ здатен вирішувати дуже багатий. Такий інструмент може використовуватись у багатьох сферах людської діяльності Для вирішення задачі планування ІТ-проєктів слабкий ШІ буде якісним засобом, який допоможе покращити результати планування за коротші терміни із врахуванням потенційних ризиків.

1.3.2 Аналіз способів застосування засобів штучного інтелекту для вирішення задачі планування проєктів з обмеженими ресурсами

RCPSP (Resource-Constrained Project Scheduling Problem) є класичною проблемою у плануванні проєктів. Хоча вона досить добре досліджена, але все ж таки не має універсального рішення та потребує специфічних підходів в залежності від вимог конкретного проєкту.

Широкий спектр засобів ШІ дозволяє вирішувати поставлену задачу різними способами, враховуючи необхідні показники, які допоможуть закінчити проєкт у поставлені терміни з максимальною низкою виконаних задач.

Розглянемо деякі інструменти ШІ, які допомагають вирішити поставлену проблему.

МН (машинне навчання) - це набір методів, які часто використовуються для вирішення реальних проблем, де комп'ютерні системи можуть навчитися вирішувати завдання без явного програмування для цього [12]. Застосування машинного навчання для планування може бути ефективним для великих

проєктів з подібною динамікою та загальними даними. Цей підхід охоплює різні форми навчання, включаючи навчання під наглядом, під частковим наглядом, без вчителя та навчання з підкріпленням.

Кожен з цих напрямків має свої переваги та виклики. Наприклад, для перевірки правил можна використовувати методи машинного навчання, але вони потребують високоякісних навчальних даних для створення точних моделей, які можуть ефективно прогнозувати [13].

Навчання без вчителя дозволяє програмному забезпеченню самостійно виявляти патерни поведінки, що сприяє прогнозуванню результатів. Однак результати такого навчання зазвичай більш специфічні для конкретного проєкту.

Нейронні мережі можуть бути застосовані в різноманітних задачах, включаючи створення інструментів підтримки рішень для планування робочої сили та графіку. У таких моделях застосовуються методи підгонки для прогнозування попиту та виробництва, що особливо важливо для виробничих систем з непередбачуваним попитом [11].

Незважаючи на ефективність нейронних мереж, їх навчання супроводжується певними труднощами, такими як погана якість навчальних даних, складність у виборі гіперпараметрів, проблеми з градієнтами та загальна складність задачі [14].

Евристичні алгоритми належать до класу алгоритмів, які, хоча не завжди знаходять оптимальне рішення, але надають наближений варіант. Серед таких алгоритмів можна виділити генетичні алгоритми, гармонічний пошук, гравітаційний пошук та інші [15].

Основною перевагою цих алгоритмів є їх ефективність у вирішенні задач. Відмінні від точних алгоритмів, які вимагають значних обчислювальних ресурсів для знаходження оптимального рішення, евристичні алгоритми виявляються швидшими та більш гнучкими. Це особливо важливо у динамічних проєктах з великим обсягом даних, де потрібно ефективно оптимізувати процеси.

Евристичні алгоритми не гарантують знаходження оптимального рішення, що робить їх непридатними для деяких специфічних задач. Ефективність таких алгоритмів також сильно залежить від конкретного домену, в якому вони застосовуються, оскільки різні задачі можуть вимагати різних налаштувань, що може знижувати їх ефективність [29].

Для того щоб вирішити проблему невпевненості в оптимальності рішення, були розроблені гібридні алгоритми. Це алгоритми, які комбінують різні моделі для пошуку глобального оптимуму з метою досягнення кращих результатів. Вони можуть поєднувати переваги різних підходів, що дозволяє їм бути більш адаптивними та ефективними у рішенні складних завдань.

На сьогоднішній день існує багато видів гібридних алгоритмів, які успішно застосовуються у різних галузях діяльності для вирішення складних оптимізаційних задач.

Порівняльну характеристику методів ШІ наведено у таблиці 1.2.

Таблиця 1.2 – Порівняльна характеристика методів ШІ (за даними [12-15])

Підхід	Переваги	Недоліки	Придатність

МН	Точні прогнози тривалості діяльності; Керується даними	Залежить від якості даних; Непрямий підхід до вирішення	Обширні проекти з використанням історичних даних
НМ	Потужне розпізнавання патернів; Моделювання складних відносин	Вимагає великих навчальних даних; По природі «Чорна скринька»	Проєкти зіскладними характеристиками та специфічними завданнями
Евристичний	Адаптивний і гнучкий підхід; Ефективний для досягнення задовільних	Результати не завжди є оптимальними. Потрібне налагодження параметрів.	Проекти з потребою у швидких практичних вирішеннях.
Гібридний	Використовує найкращі аспекти різних методів;	Складнощі з інтеграцією та налаштуванням.	Проєкти великого масштабу та складності.

За результатами проведеного аналізу певного висновку щодо визначення найкращого методу дати не можна, оскільки кожен з них виступає краще серед інших при виконанні певних умов.

Задача планування проєкту з обмеженими ресурсами вимагає постійного коригування графіку завдань і призначення відповідних ресурсів. Методи МН (математичне програмування) та НМ (нейронні мережі) найбільш ефективні для великих та тривалих проєктів з великим обсягом роботи та історичними даними. Гібридні алгоритми, хоча і складніші, можуть бути краще пристосовані для специфічних потреб проєктів, але не

завжди забезпечують найкраще можливе рішення без додаткових конфігурацій.

Для стартапів та невеликих проєктів евристичні алгоритми є найбільш відповідним варіантом. Вони не вимагають значних обчислювальних ресурсів або історичних даних і зазвичай надають приблизно оптимальні результати..

Далі у роботі буде розглянуто роботу гібридного алгоритму та його застосування для вирішення RCPSP.

1.4 Постановка задачі дослідження

Метою кваліфікаційної роботи є розробка та аналіз методики використання генетичних алгоритмів для оптимізації процесу планування в IT-проєктах, зокрема для розв'язання задачі RCPSP. Як було зазначено раніше, загально рекомендовані методики формування плану робіт не завжди є ефективними для вирішення задач, тому дослідження цієї галузі буде актуальним напрямком для подальшого розвитку [28].

Основні завдання дослідження включають:

– Аналіз сучасного стану планування в IT-проєктах – Вивчення існуючих методів та інструментів планування, аналіз їх переваг та недоліків у контексті проєктів з обмеженими ресурсами.

– Дослідження ефективності застосування генетичних алгоритмів для розв’язання RCPSP – Розробка методики застосування генетичних алгоритмів, що включає алгоритмічні моделі, параметри оптимізації, процедуру

навчання та оптимізації, а також аналіз результативності генетичних алгоритмів.

– Розробка практичного рішення з використанням генетичного алгоритму для рішення RCPSP.

– Проведення апробації генетичного алгоритму для рішення RCPSP на основі тестових даних.

Це дослідження націлене на розробку модифікації генетичного алгоритму для підвищення ефективності планування ІТ-проєктів. Очікувані результати включають збільшення ефективності планування в ІТ-проєктах завдяки застосуванню генетичних алгоритмів, скорочення загального часу виконання проєктів та оптимізацію використання ресурсів. Також передбачається розширення можливостей управління проєктами шляхом адаптації до змінних умов і вимог, що забезпечить більшу гнучкість і динамічність у плануванні та виконанні проєктних робіт [27].

Це дослідження не тільки спрямоване на покращення управління ІТ-проєктами за допомогою інноваційних технологій штучного інтелекту, але й має на меті надати проєктним менеджерам потужний інструментарій для оптимізації робочих процесів.

Генетичні алгоритми, завдяки своїй універсальності та адаптивності, відкривають нові перспективи для ефективного розв'язання складних задач планування, що є особливо актуальним у сучасних динамічних умовах ІТ-галузі.

2 ДОСЛІДЖЕННЯ МОДИФІКАЦІЇ ГЕНЕТИЧНОГО АЛГОРИТМУ ДЛЯ РОЗВ’ЯЗАННЯ ЗАДАЧІ ПЛАНУВАННЯ ІТ-ПРОЄКТІВ З ОБМЕЖЕНИМИ РЕСУРСАМИ

2.1 Опис задачі планування проектів з обмеженими ресурсами та математичне представлення

RCPSP є класичною задачею комбінаторної оптимізації, яка має широкі практичні застосування в управлінні проектами та виробництвом. В контексті RCPSP проект представляється як набір робіт, кожна з яких має визначену тривалість виконання та вимагає певних ресурсів. Ресурси обмежені, і для виконання робіт вони повинні бути розподілені ефективно. Крім того, існує порядок виконання робіт, який визначається відношеннями передування між ними.

RCPSP належить до класу NP-складних задач, що свідчить про те, що для неї відсутні точні алгоритми розв'язання за поліноміальний час, особливо для великих розмірностей. Це означає, що зі збільшенням кількості робіт та ресурсів складність обчислень значно зростає.

Основною метою вирішення задачі RCPSP є побудова прийнятного розкладу виконання робіт проекту, який враховує обмеження на порядок виконання робіт та доступність ресурсів, а також оптимізує певний критерій ефективності. Найбільш поширеним критерієм є мінімізація часу завершення всього проекту.

Ця задача важлива для практичного застосування, оскільки її розв'язання допомагає ефективно управляти

складними проектами з обмеженими ресурсами і забезпечує оптимальне використання часу та ресурсів.

В загальному вигляді RCPSP можна сформулювати наступним чином:

– набір активностей $N = \{1,2,\dots,n\}$, який включає фіктивні активності для початкової та кінцевої подій;

– набір тривалостей для кожної діяльності: $D = d_1, d_2, \dots, d_n$;

– набір доступних ресурсів $R = R_1, R_2, \dots, R_m$, де m – номер кожного ресурсу;

– R_{ik} – це кількість ресурсів k необхідних для активності i. R_k – це сумарна кількість доступного ресурсу k;

– P – набір послідовностей та зв'язків між активностями. Активність i та активність j, визначають, що перша з них має спершу закінчитись, ніж почнеться наступна.

Для даної роботи буде розглянуто одну цільову функцію, яка ставить метою мінімізацію загального часу завершення проєкту, який є часом завершення останньої діяльності.

$$f(x) \rightarrow \min C_{max}, \tag{2.1}$$

де C – тривалість проєкту.

Оскільки врахування обмежень є критично важливим, то такі обмеження необхідно визначити. До таких обмежень відносяться відповідність до порядку виконання та обмеження за ресурсами.

Формула обмеження за порядком виконання записується наступним чином:

$$x_j \geq x_i + d_i, \quad \forall (i,j) \in P, \tag{2.2}$$

де P – набір послідовностей та зв'язків між активностями;

x_i – активність;

d_i – тривалість активності.

Формула обмеження за ресурсами:

$$\sum_{i \in N} r_{ik} \cdot u_i(t) \leq R_k, \quad \forall k \in \{1, \dots, m\}, \tag{2.3}$$

де $u_i(t)$ – використання активності і в часовий інтервал t;

R_k – це сумарна кількість доступного ресурсу k.

Час початку кожної діяльності має бути не меншим за 0:

$$x_i \geq 0, \quad \forall i \in N, \tag{2.4}$$

де N – множина натуральних чисел.

У кваліфікаційній роботі розглядається модифікація генетичного алгоритму з поєднанням SOM для забезпечення різноманітності поколінь та способу пошуку глобального оптимуму при дотриманні вищевказаних обмежень на рішення задачі оптимізації.

2.2 Архітектурний опис генетичних алгоритмів

Генетичні алгоритми (ГА) представляють собою потужний метаевристичний підхід до вирішення складних оптимізаційних задач, що базується на принципах еволюції та природного відбору. Цей клас алгоритмів, який належить до групи еволюційних алгоритмів, знайшов широке застосування у таких галузях, як штучний інтелект, машинне навчання, операційні дослідження та інженерія [17-18].

Основна ідея генетичних алгоритмів полягає в моделюванні процесу еволюції, в якому працює популяція потенційних розв'язків. Кожен потенційний розв'язок представляється у вигляді хромосоми, яка складається з послідовності генів. Гени можуть бути представлені бінарними значеннями, дійсними числами або іншими типами даних, в залежності від конкретної задачі.

Функція придатності у генетичних алгоритмах є вирішальним елементом, оскільки вона визначає якість кожного потенційного розв'язку в популяції. Це значення використовується для управління процесом еволюції шляхом відбору більш придатних розв'язків для створення нащадків.

Оператор селекції в генетичних алгоритмах відповідає за вибір батьківських особин з популяції на основі їхньої придатності. Існують різні методи селекції, такі як рулеткова селекція, турнірна селекція та ранжувальна селекція. Ці методи забезпечують баланс між вивченням простору пошуку та використанням потенційно перспективних розв'язків.

Схрещування є ключовим оператором у генетичних алгоритмах, який забезпечує обмін генетичною інформацією між батьківськими хромосомами для створення нащадків. Існують різні типи операторів схрещування, такі як одноточкове, двоточкове та рівномірне схрещування, кожен з яких має свої особливості та застосування в залежності від конкретної задачі.

Оператор мутації в генетичних алгоритмах виконує важливу роль у забезпеченні різноманітності та експлорації простору пошуку. Цей оператор вносить випадкові зміни в гени хромосом з певною ймовірністю. Основна ціль мутації полягає у тому, щоб запобігти передчасній збіжності алгоритму до локальних оптимумів і допомогти знаходити більш глобально оптимальні рішення.

Ймовірність мутації зазвичай встановлюється на низькому рівні, щоб зберегти баланс між дослідженням нових областей простору рішень і використанням вже знайдених потенційно корисних розв'язків. Занадто висока ймовірність мутації може призвести до втрати корисних властивостей популяції, тоді як занадто низька може спричинити затримку в розвитку нових розв'язків.

Типичні методи мутації включають зміну значень генів на випадкові, обмін значеннями між генами, інверсію послідовностей генів тощо. Конкретний вибір методу мутації залежить від природи задачі і типу розв'язків, які ми сподіваємося отримати.

Популяція – це колекція потенційних рішень задачі, представлених у формі хромосом. Популяція записується як

$$P(t) = \{x_1, x_2, \dots, x_N\}, \tag{2.5}$$

де t – номер покоління;

x_N – індивіди.

Функція придатності використовується для оцінки якості рішення, а саме відповідності індивіда до поставленої умови. У даній роботі функція придатності має на меті мінімізацію тривалості проєкту:

$$f: X \to R, \tag{2.6}$$

де R – множина значень для функції f;

X – область визначення функції f.

Вибір особин з популяції на основі їх пристосованості для участі в репродукції (стають батьками). Це може бути представлено як функція відбору *S*, застосована до популяції:

$$S\big(P(t)\big) = \{y_1, y_2, \dots, y_M\}, \tag{2.7}$$

де $M \leq N$ – кількість обраних особин;

y_M – відібрані особини.

Кросовер застосовується до пар батьківських особин для створення потомства наступного покоління. Функція кросоверу *C* може бути представлена як:

$$C\big(y_i, y_j\big) = \big(z_i, z_j\big), \tag{2.8}$$

де y_i, y_j – батьківські особини;

z_i, z_j – результати потомства.

Мутація вносить варіації у потомство та записується як:

$$M(z_i) = z_i', \tag{2.9}$$

де z_i – особина потомства перед мутацією;

z_i' – особина потомства після мутації.

Оновлення популяції – формується нове покоління особин, можливо, шляхом поєднання батьківських особин і потомства з подальшим відбором для підтримки розміру популяції:

$$P(t+1) = U(P(t), \{z_1', z_2', \dots, z_K'\}), \tag{2.10}$$

де U — функція оновлення, що формує наступне покоління;

K — кількість потомства;

z_K' – особина потомства після мутації.

Ітераційний процес повторюється протягом T поколінь або до досягнення деякого критерію зупинки з метою еволюції особин до кращих рішень [19].

2.3 Опис самоорганізаційних карт Кохонена

Самоорганізаційна карта Кохонена (SOM) - це штучна нейронна мережа для навчання без учителя, що перетворює багатовимірні дані в низькорозмірну дискретну карту, зберігаючи топологічні властивості. Використовується для кластеризації, візуалізації даних та зменшення розмірності.

Архітектура SOM складається з одношарової нейронної мережі, в якій нейрони організовані у вигляді двовимірної

решітки. Кожен нейрон пов'язаний з усіма вхідними елементами через вагові коефіцієнти, які адаптуються у процесі навчання. Ключовою особливістю SOM є збереження топологічних властивостей вхідних даних, тобто близькі вхідні вектори будуть відображатися на близькі нейрони на карті.

Застосування самоорганізаційних карт Кохонена (SOM) для вирішення задач планування проєктів, зокрема таких як RCPSP (Resource-Constrained Project Scheduling Problem), представляє собою цікавий напрямок досліджень. RCPSP є NP-складною задачею, що означає велику складність її оптимізації у великому масштабі [25].

Основна ідея застосування SOM полягає в тому, щоб знайти такий розклад виконання робіт проєкту, який би задовольняв обмеженням на ресурси і послідовність виконання робіт, і при цьому мінімізував загальний час виконання. SOM використовується для створення низькорозмірної дискретної карти, на якій кожен нейрон представляє можливе рішення або варіант розкладу проєкту.

1. Головні переваги використання SOM у цьому контексті включають: Топологічне збереження: SOM зберігає топологічні властивості вхідних даних, що означає, що близькі вхідні вектори відображаються на близьких нейронах на карті. Це дозволяє знаходити рішення, що є близькими за якістю до оптимальних, що може бути корисним у вирішенні RCPSP.

2. Обробка багатовимірних даних: SOM може ефективно обробляти великі обсяги багатовимірних даних, що стосується планування проєктів з великою кількістю завдань і ресурсів.

3. Адаптація до змін: SOM можуть досліджувати і адаптуватися до змін у вхідних даних або умовах проєкту, що

забезпечує більшу гнучкість у плануванні і управлінні проєктами.

Однак, важливо враховувати, що успішність застосування SOM залежить від декількох факторів, таких як правильний вибір параметрів навчання, розмір карти, архітектура мережі і кількість ітерацій. Поєднання SOM з еволюційними методами, наприклад генетичними алгоритмами, може підвищити ефективність пошуку оптимальних розв'язків у складних проектних умовах.

Гібридний алгоритм SOM-GA (самоорганізаційна карта Кохонена - генетичний алгоритм) є перспективним підходом до вирішення складної задачі планування проєктів з обмеженнями ресурсів (RCPSP). В даній роботі пропонується поєднати переваги обох методів з метою покращення ефективності пошуку оптимальних або майже оптимальних рішень. Ось деякі ключові переваги гібридного підходу:

4. Кластеризація рішень за допомогою SOM: SOM дозволяють кластеризувати хромосоми ГА (розв'язки проєкту) на основі їхніх характеристик, що включають значення придатності та генетичний склад. Кожен вузол на карті SOM представляє кластер схожих рішень. Це зберігає різноманітність у популяції, запобігаючи передчасній конвергенції до локальних оптимумів.

5. Покращений пошуковий простір: Гібридний підхід дозволяє ефективно досліджувати різні ніші в просторі рішень. SOM допомагає зосереджувати ГА на вже відомих перспективних ділянках простору рішень, що сприяє зниженню часу пошуку оптимальних рішень.

6. Збереження топологічних властивостей: SOM зберігають топологічні властивості вхідних даних, що дозволяє ефективно використовувати їх для картації складних просторів рішень, таких як RCPSP.

7. Паралельне виконання: Якщо врахувати можливість паралельного виконання, гібридний підхід може бути обчислювально ефективним для великих проєктів, де необхідно обробляти великі обсяги даних. Використання гібридного алгоритму SOM-GA може значно підвищити точність і швидкість знаходження оптимальних рішень в складних умовах планування проєктів, таких як RCPSP.

2.4 Опис гібридного алгоритму самоорганізуючих карт Кохонена та генетичного алгоритму

Гібридний алгоритм SOM-GA для вирішення задачі планування проєктів з обмеженнями ресурсів (RCPSP) має значні переваги. Він поєднує в собі здатність самоорганізаційної карти Кохонена (SOM) кластеризувати схожі рішення в просторі з роботою генетичного алгоритму (GA), що дозволяє ефективно досліджувати простір розв'язків і зменшувати ризик передчасної конвергенції. Цей підхід сприяє покращенню якості знайдених рішень і забезпечує збереження різноманітності в популяції, що є ключовим у вирішенні складних комбінаторних задач, таких як RCPSP.

Динамічне формування кластерів в SOM дозволяє адаптувати структуру кластерів під час розвитку популяції, що є

важливим для ефективності алгоритму в динамічному середовищі. SOM також забезпечує механізм зворотного зв'язку для ГА, ідентифікуючи недосліджені області у просторі рішень і спрямовуючи пошук ГА на менш досліджені регіони через кросовер і мутацію. Використання гібридного алгоритму SOM-GA має переваги для складних комбінаторних задач, таких як RCPSP, зокрема для великих проєктів з великою кількістю можливих рішень [24].

2.5 Огляд існуючих робіт

Запропонований підхід поєднує властивості SOM для кластеризації хромосом, що генеруються ГА під час оптимізації RCPSP. Після створення популяції хромосом ГА використовує SOM для їхнього відображення, де схожі хромосоми розподіляються по кластерам. Якщо ГА зазнає стагнації та застрягає у локальному оптимумі, застосовується метод збільшення різноманіття шляхом вибору хромосом із різних кластерів для створення нових нащадків. Це дозволяє уникнути локальних оптимумів і продовжити пошук для знаходження глобального оптимуму в задачі RCPSP.

У інших наукових роботах було досліджено інтеграцію SOM та ГА для різних типів задач. Наприклад, у роботі [21] розроблено гібридний алгоритм GASOM, який використовує SOM для отримання інформації з процесу еволюції та керування стратегією пошуку. Вони показали, що GASOM ефективно

запобігає передчасній конвергенції та перевершує інші варіанти ГА в складних проблемах.

У роботі [22] запропоновано гібридний підхід, де використовуються кілька SOM та ГА для навчання. SOM групують індивідууми в кожному поколінні, а ГА виконує операції в межах субпопуляцій, визначених цими кластерами. Цей підхід покращує ефективність локального пошуку ГА та дозволяє знаходити кращі рішення швидше, ніж стандартний ГА.

У контексті задачі RCPSP використання SOM може надати корисну інформацію про структуру простору рішень. Об'єднуючи схожі рішення, SOM дозволяють визначити перспективні регіони для подальшого використання ГА. Зберігаючи різноманітність за допомогою відбору хромосом з різних кластерів, цей підхід дозволяє досліджувати ширший діапазон простору пошуку та уникати потрапляння в локальні оптимуми.

Робота [22] щодо проблеми комівояжера також підтверджує ідею використання кластеризації разом з ГА для вирішення складних планувальних задач. Хоча сама задача комівояжера відрізняється від RCPSP, обидві вони передбачають пошук оптимальних послідовностей або розкладів з урахуванням обмежень.

Висновок з аналізу робіт підтверджує, що комбінація самоорганізуючихся карт (SOM) і генетичних алгоритмів (ГА) є перспективним напрямом у дослідницьких роботах, особливо для вирішення складних задач оптимізації, таких як RCPSP або проблеми комівояжера. Використання SOM для кластеризації індивідуальних рішень у популяції ГА дозволяє ефективно

керувати пошуком і уникнути застрягання в локальних оптимумах.

Одним із ключових переваг такого підходу є здатність SOM ідентифікувати та візуалізувати різні кластери рішень, що дозволяє наочно спостерігати розподіл індивідів у просторі рішень. Це може бути корисним для користувачів, які аналізують або визначають оптимальні рішення.

Варто відзначити, що таке поєднання може потребувати значних обчислювальних ресурсів через необхідність оптимізації конфігурацій при навчанні SOM та використанні ГА. Деякі спеціалізовані алгоритми пошуку можуть бути ефективнішими в певних умовах, але гібридний підхід з SOM та ГА залишається потужним інструментом для дослідження просторів великих розмірів і складних структур даних.

У цілому, інтеграція SOM та ГА відкриває широкі можливості для оптимізації і візуалізації складних проблем, і дозволяє досягати кращих результатів у вирішенні мультимодальних і оманливих задач.

2.6 Опис схеми реалізації гібридного алгоритму

Для реалізації гібридного алгоритму необхідно сформувати послідовність дій, що будуть виконані в процесі виконання. Візуалізація такої послідовності може бути виконана завдяки діаграмі активностей, що відображатиме взаємодію між різними компонентами системи.

Формування наочного алгоритму роботи алгоритму дозволить встановити як саме працює гібридний алгоритм та виділити нові кроки, що додаються для модифікації звичайного алгоритму.

Для цього було побудовано діаграму активностей, яку наведено на рисунку 2.1.

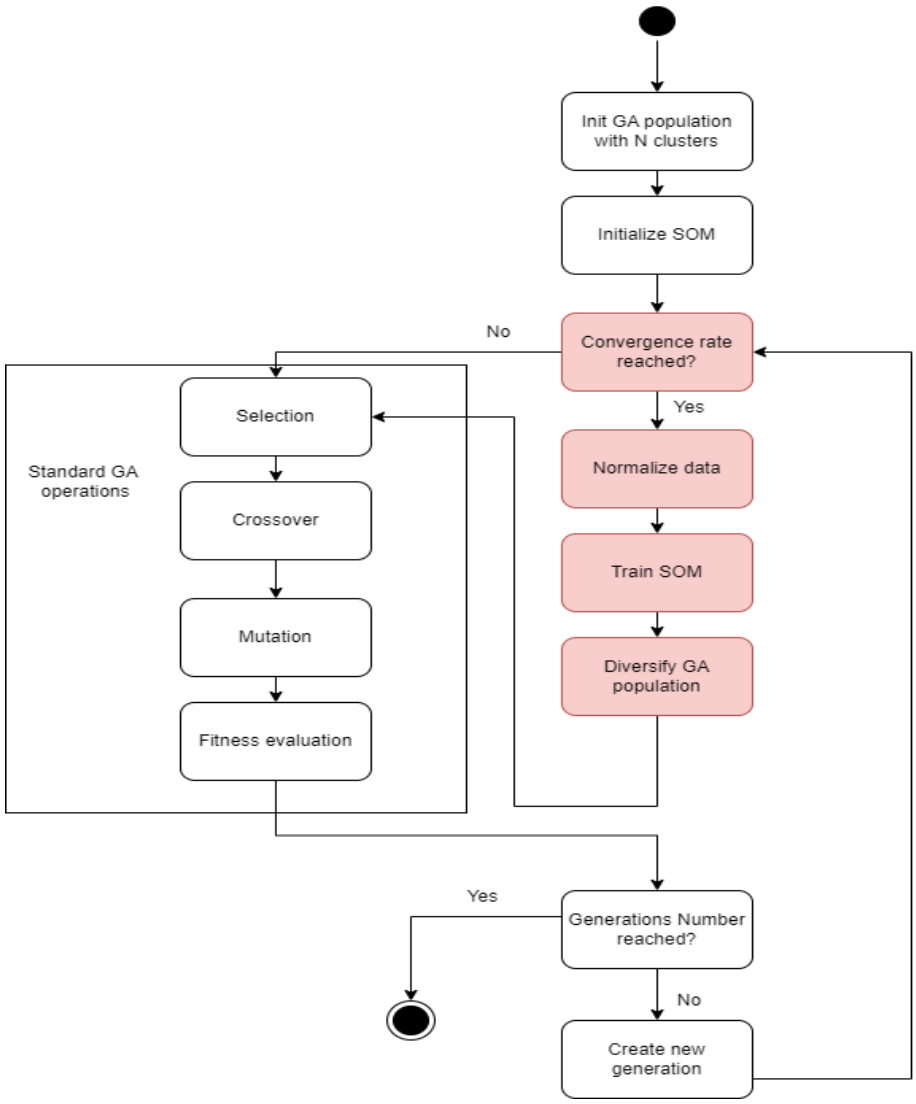

Рисунок 2.1 – Діаграма активностей роботи гібридного алгоритму (виконано самостійно)

З наведеної діаграми можна побачити, що більша частина логіки генетичного алгоритму залишається як і раніше, але у

гібридній реалізації додаються кроки для SOM. Оскільки навчання SOM є трудомістким та значно впливає на ефективність роботи програми в залежності від вхідних параметрів, то його навчання слід проводити тільки у випадках, коли це насправді необхідно [23].

2.7 Вибір методів для реалізації інтеграції генетичного алгоритму та самоорганізуючих карт Кохонена

2.7.1 Ініціалізація генетичного алгоритму

Процес ініціалізації в генетичному алгоритмі для RCPSP є важливою фазою, оскільки він встановлює початкову популяцію хромосом, які будуть еволюціонувати для пошуку оптимального розкладу проекту. Використання механізму вибору колеса рулетки на основі мінімального часу останнього завершення (LFT) дозволяє визначити пріоритет дій, що сприяє створенню хромосом з потенціалом для скорочення тривалості проекту. Такий підхід дозволяє підтримувати ефективність процесу еволюції популяції і спрямовує її на пошук оптимального графіку проєкту, уникаючи перешкод у вигляді передчасної конвергенції до локальних оптимумів.

Розрахунок мінімального LFT для кожної дії в мережі проекту є ключовим кроком у плануванні робіт. Він визначає останній можливий час завершення кожної дії таким чином, щоб проект завершився вчасно. Цей процес включає два основних кроки: прохід вперед і прохід назад.

Під час проходу вперед для кожної дії обчислюється найраніший можливий час початку і завершення, враховуючи

час завершення всіх її попередників у мережі. Це дозволяє визначити, коли кожна дія може початися, якщо виконувати її в послідовності з урахуванням залежностей.

Прохід назад визначає останній можливий час завершення для кожної дії, враховуючи час початку її наступників. Це дозволяє з'ясувати, до якого максимального терміну дії можуть бути виконані без затримок в завершенні проекту.

Мінімальний LFT для кожної дії є останнім можливим часом завершення, необхідним для вчасного завершення проекту. Використання цієї інформації дозволяє визначити критичні шляхи в мережі проекту і ефективно планувати виконання робіт.

Обчислення максимального LFT серед можливих видів діяльності:

$$LFT_{max} = \max_{i \in P} L\,FT(i), \tag{2.11}$$

де P – множина всіх можливих активностей;

$LFT(i)$ – це найпізніший час завершення активності i.

Обчислення ваги вибору для кожної можливої діяльності:

$$w(i) = LFT_{max} - LFT(i) + 1, \tag{2.12}$$

де LFT_{max} – максимальний LFT серед можливих видів діяльності;

$LFT(i)$ – найпізніший час завершення активності i.

Обчислення загальної ваги вибору:

$$W = \sum_{i \in P} w(i), \tag{2.13}$$

де P - множина всіх можливих активностей;

$w(i)$ – вага вибору діяльності і.

Призначення ймовірності вибору для кожної можливої діяльності. Наступна формула гарантує, що дії з нижчими LFT мають вищу ймовірність бути обраними.

$$P(i) = \frac{w(i)}{W}, \tag{2.14}$$

де $w(i)$ – вага вибору діяльності і;

W – загальна вага вибору.

Запропонований метод використання механізму колеса рулетки для вибору активностей у генетичному алгоритмі для задачі планування проектів (RCPSP) є ефективним способом формування початкової популяції. Він дозволяє створювати хромосоми, які відображають потенційні рішення з мінімальними часами завершення (LFT), що сприяє швидшому знаходженню оптимальних розкладів проектів або їх близьких варіантів. Процес вибору активностей на основі кумулятивних ймовірностей дозволяє контролювати розподіл вибраних активностей і максимізувати ймовірність утворення хромосом, які забезпечують оптимальність або наближеність до неї в подальших етапах оптимізації.

2.7.2 Метод селекції для генетичного алгоритму

Ранговий відбір в генетичному алгоритмі використовується для вибору батьківських хромосом на основі їх рангів, що визначаються за значеннями придатності. Цей метод спрямований на збереження різноманітності в популяції та уникнення передчасної конвергенції до локальних оптимумів.

Для проведення рангового відбору спочатку хромосоми сортуються в порядку спадання за їхніми значеннями придатності, починаючи від найбільш придатних до найменш придатних. Потім кожній хромосомі присвоюється ранг на основі її позиції в відсортованому списку: найбільш придатна хромосома отримує ранг N N, де N N - це загальна кількість хромосом у популяції, а найменш придатна хромосома отримує ранг 1.

Цей підхід дозволяє кращим хромосомам (тобто тим з вищими значеннями придатності) мати більшу ймовірність вибору для відтворення, зберігаючи при цьому можливість для менш придатних хромосом брати участь у процесі еволюції. Таким чином, ранговий відбір сприяє балансу між експлорацією та експлуатацією простору пошуку в генетичному алгоритмі. Імовірність вибору для кожної хромосоми розраховується за формулою:

$$P(i) = \frac{r_i}{\sum_{j=1}^{N} j} = \frac{2r_i}{N(N+1)}, \quad (2.15)$$

де N — розмір популяції;

r_i – її ранг.

Ранговий вибір є ефективним методом в генетичних алгоритмах для відбору батьківських хромосом на основі їх рангів, які визначаються за значеннями придатності. Цей підхід забезпечує баланс між використанням найкращих рішень і дослідженням простору пошуку, що є важливим для досягнення глобальної оптимізації в еволюційних алгоритмах.

Основні переваги рангового вибору полягають в тому, що він забезпечує сталу інтенсивність відбору протягом усього процесу еволюції. Це допомагає уникнути ситуації, коли сильні особини домінують в популяції на ранніх етапах, що може призвести до застрягання в локальних оптимумах. Ранговий підхід також менш чутливий до масштабування значень придатності, оскільки враховує тільки відносні рейтинги хромосом.

Дозволяючи навіть менш придатним особинам мати шанс бути обраними в якості батьків, ранговий вибір сприяє збереженню різноманітності в популяції і зменшує ризик передчасної конвергенції, коли алгоритм застрягає у локальних оптимумах.

Цей метод є популярним у багатьох застосуваннях генетичних алгоритмів через його здатність ефективно підтримувати різноманітність та сприяти пошуку глобального оптимума [2].

2.7.3 Вибір методу кросинговеру для генетичного алгоритма

Багатоточковий кросинговер у генетичних алгоритмах є методом, що використовується для створення нових комбінацій генетичного матеріалу шляхом обміну частинами між батьківськими хромосомами в декількох випадково вибраних точках. Цей підхід сприяє більш глибокому дослідженню простору пошуку та збереженню корисних генетичних структур в нових комбінаціях, що може підвищити ефективність пошукового процесу в ГА.

Багатоточковий кросинговер у спрощеному варіанті зображено на рисунку 2.3 [19].

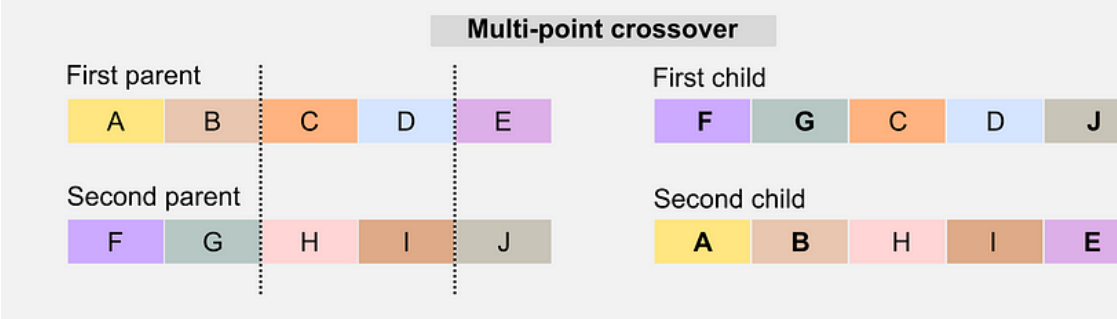

Рисунок 2.2 – Приклад багатоточкового кросинговеру (виконано самостійно)

Механізм ремонту, що використовується у вашій роботі, полягає в видаленні повторюваних дій і вставці відсутніх дій з метою збереження порядку дій у батьківських хромосомах. Цей підхід допомагає після кросинговеру забезпечити придатність новоутворених хромосом для подальшої еволюції в генетичному алгоритмі.

Кількість точок кросинговеру, k, є важливим гіперпараметром, який можна налаштовувати залежно від

конкретної проблеми та вимог до пошукової якості. Більше значення k призводить до більш значного обміну генетичним матеріалом між батьками, що може сприяти більш широкому дослідженню простору пошуку, але може також призвести до втрати корисних комбінацій генів. З іншого боку, нижче значення k забезпечує більш консервативний обмін генетичним матеріалом, що зберігає більшість оригінальних генетичних структур, але може обмежити рівень розвідки.

Оптимальне значення k варіюється в залежності від конкретної задачі і вимагає збалансованого підходу між інтенсивністю пошуку нових рішень і збереженням корисних генетичних структур в популяції.

До переваг багатоточкового кросинговеру відносяться дослідження ширшого пошукового простору, легкість реалізації та адаптивність до поставлених задач.

Тим не менш, у випадках, коли необхідно задіяти механізм відновлення, то він може бути досить коштовним для великих проєктів. Також оптимальна кількість точок перетину може змінюватися залежно від проблеми та поточного стану процесу пошуку, що ускладнює визначення найкращого значення для k та потребує додаткових експериментів.

2.7.4 Вибір методу мутації для генетичного алгоритму

Рівномірна мутація є простим і ефективним оператором для генетичних алгоритмів, який застосовується для внесення випадкових змін в гени хромосоми з певною ймовірністю. Цей

метод відрізняється від мутації Гауса, яка застосовується для неперервних просторів пошуку і вводить невеликий стохастичний шум у гени.

Операція рівномірної мутації полягає у виборі кожного гена хромосоми та його заміні новим значенням, яке вибирається з рівномірного розподілу в межах дозволеного діапазону для цього гена. Це означає, що кожен ген має однакову ймовірність бути зміненим на будь-яке інше значення з його діапазону значень.

Рівномірна мутація особливо ефективна для задач з дискретними або категоріальними просторами пошуку, де значення генів представляють собою дискретні категорії або варіанти. В таких випадках рівномірна мутація дозволяє ефективно досліджувати різні комбінації цих категорій і забезпечує збереження допустимих значень.

Цей оператор мутації є популярним вибором у багатьох застосунках генетичних алгоритмів через свою простоту і можливість адаптації до різних видів просторів пошуку.

Процес рівномірної мутації описується наступним чином.

Обирається хромосома з популяції для мутації на основі попередньо визначеної ймовірності мутації p_m.

Рівномірна мутація є простим і ефективним оператором генетичного алгоритму, що використовується для введення випадкових змін у гени хромосоми з певною ймовірністю $p\ m$ p m . Оператор вносить дискретні зміни, замінюючи поточні значення генів новими значеннями, обраними з відповідного дозволеного діапазону. Ймовірність $p\ m$ p m визначає частоту виконання мутацій і зазвичай обирається в межах від 0.01 до 0.1. Цей метод особливо підходить для задач з дискретними або

категоріальними змінними, де оптимальні рішення можна знайти шляхом комбінації різних варіантів значень.

2.7.5 Вибір алгоритму ініціалізації для самоорганізаючих карт Кохонена

Ініціалізація SOM з лінійним розпадом є методом, що використовується для встановлення початкових вагових коефіцієнтів у мережі. Цей підхід включає поступове зменшення значень ваг залежно від їх позиції на сітці нейронів. Така ініціалізація сприяє рівномірному розподілу початкових ваг серед нейронів і допомагає уникнути нерівномірного впливу від початкових умов на процес навчання. Це особливо важливо для задач, де необхідна точна моделювання простору пошуку і де важливо забезпечити стабільність та консистентність у результаті навчання мережі.

Для кожного нейрона (i, j) у SOM ініціалізується його ваговий вектор w_{ij} за допомогою функції лінійного розпаду:

$$w_{ij} = w_{max} - (w_{max} - w_{min}) \cdot \frac{d_{ij}}{d_{max}}, \quad (2.16)$$

де w_{max} і w_{min} — максимальне та мінімальне значення бажаного діапазону ваги відповідно;

d_{ij} — це відстань між нейроном (i, j) і вибраною опорною точкою, тобто, центром карти;

d_{max} — максимальна відстань між будь-яким нейроном і контрольною точкою.

Ініціалізація SOM з використанням лінійного розпаду вагових коефіцієнтів передбачає повторення процесу для кожного нейрона у мережі. Цей метод дозволяє рівномірно розподілити початкові ваги серед нейронів і забезпечити стабільність процесу навчання. Вибір розміру карти нейронів є також важливим аспектом ініціалізації. Рекомендації вказують, що розмірність карти повинна відповідати кількості вхідних елементів, щоб забезпечити належне розташування нейронів і логічну відстань між ними. Це важливо для ефективного кластеризування та аналізу вхідних даних у контексті задачі, яка вирішується за допомогою SOM.

Далі наведено формулу обчислення розмірності карти:

$$5 * \sqrt{N}, \tag{2.17}$$

де N – розмір популяції, на основі якої буде проведено навчання.

Завдяки функції лінійного розпаду ваги нейронів лінійно зменшуються зі збільшенням відстані від контрольної точки. Ця схема ініціалізації створює плавний градієнт ваг по всій карті з вищими значеннями біля центру та меншими значеннями до країв.

Ініціалізація лінійного розпаду забезпечує структурований і детермінований спосіб ініціалізації ваг SOM, зменшуючи випадковість і мінливість у початковому стані карти. Плавний градієнт вагових коефіцієнтів допомагає зберегти топологічні зв’язки між вхідними характеристиками,

що полегшує процес навчання та формування значущих кластерів.

Ефективність ініціалізації залежить від вибору вагового діапазону та контрольної точки. Якщо діапазон надто вузький або контрольна точка не підходить для вхідних даних, початкова карта може не охопити всю мінливість вхідного простору. Лінійна функція розпаду передбачає симетричний і рівномірний розподіл вхідних характеристик, що не завжди відповідає реальним даним. У деяких задачах більш прийнятною може бути нелінійна або асиметрична схема ініціалізації.

Незважаючи на обмеження, ініціалізація лінійного розпаду є широко використовуваною технікою в SOM через її простоту, детермінізм і можливість створення структурованої початкової карти. У поєднанні з відповідними алгоритмами та параметрами навчання, така ініціалізація допомагає SOM ефективно охопити структуру вхідного простору та підтримувати процес оптимізації в RCPSP та інших проблемах.

2.7.6 Вибір функції навчання для самоорганізаючих карт Кохонена

Швидкість навчання є важливим параметром у SOM, який визначає швидкість і величину оновлення ваг під час процесу навчання. Він контролює ступінь коригування ваги нейронів на основі вхідних шаблонів і впливу сусідів. Лінійне затухання – це часто використовуваний розклад швидкості навчання, який поступово зменшує швидкість навчання з часом. Ідея лінійного

розпаду полягає в тому, щоб почати з відносно високої швидкості навчання, щоб забезпечити швидку початкову адаптацію, а потім поступово зменшувати швидкість навчання, щоб точно налаштувати вагові коефіцієнти та стабілізувати SOM. Це сприяє швидкому початковому пристосуванню до загальної структури даних і поступовому вдосконаленню для детальнішої кластеризації.

Лінійний розпад швидкості навчання зазвичай задається у вигляді функції, яка починається з початкового значення та зменшується до кінцевого значення протягом заданої кількості ітерацій. Також використання лінійного розпаду дозволяє уникнути різких змін у навчанні, що може призвести до стабільніших результатів.

Незважаючи на те, що лінійне затухання є найпростішою формою розкладу швидкості навчання, воно має свої обмеження. Іноді може бути корисним використовувати більш складні схеми, такі як експоненціальне затухання або адаптивні методи, які коригують швидкість навчання на основі поточної продуктивності алгоритму. Вибір оптимальної функції навчання залежить від конкретної задачі, структури даних і вимог до точності та швидкості алгоритму.

Таким чином, швидкість навчання та її розклад є критично важливими елементами в алгоритмі SOM. Вони визначають, наскільки ефективно та швидко карта буде пристосовуватися до вхідних даних, і впливають на остаточну якість кластеризації. Лінійний графік розпаду визначається як:

$$\alpha(t) = \alpha_0 \left(1 - \frac{t}{T}\right), \qquad (2.18)$$

де $\alpha(t)$ – швидкість навчання на кроці часу t;

α_0 – початкова швидкість навчання;

t – поточний крок часу;

T – загальна кількість кроків часу або ітерації.

Для розв'язання RCPSP лінійний графік розпаду застосовується до швидкості навчання SOM під час фази навчання. SOM використовується для кластеризації рішень, створених ГА, забезпечуючи компактне представлення простору пошуку. Швидкість навчання визначає швидкість і величину оновлень ваг нейронів на основі вхідних рішень і впливу сусідства. Використовуючи лінійний розклад розпаду, SOM може спочатку швидко адаптуватися до рішень, створених ГА, а потім поступово стабілізуватися та зблизитися до репрезентативної карти простору пошуку.

При застосуванні інтеграції SOM і ГА для вирішення RCPSP лінійний розклад розпаду може допомогти SOM ефективно кластеризувати та представляти рішення, створені ГА, підтримуючи процес оптимізації. Однак важливо враховувати специфічні характеристики RCPSP і потенційно адаптувати графік швидкості навчання, щоб краще відповідати проблемній області. Цей підхід дозволяє SOM швидко налаштовуватися на загальну структуру даних на початкових етапах і поступово вдосконалювати свої ваги, що призводить до більш точної та стабільної кластеризації, яка краще підтримує процес еволюції ГА.

2.7. Вибір топології карти для самоорганізаючих карт Кохонена

Топологія карти визначає розташування та зв’язок нейронів у просторі. Вибір топології може суттєво вплинути на процес навчання, представлення вхідного простору та можливість інтерпретації отриманої карти. У контексті інтеграції SOM з ГА топологія гексагональної сітки особливо ефективна.

Топологія гексагональної сітки розташовує нейрони за правильною шестикутною схемою, де кожен нейрон з’єднаний із шістьма безпосередніми сусідами. Це розташування відрізняється від більш поширеної топології прямокутної сітки, де кожен нейрон з’єднаний із чотирма безпосередніми сусідами (вгорі, внизу, ліворуч і праворуч). Гексагональна топологія сітки пропонує кілька переваг перед прямокутною топологією сітки:

Покращений зв’язок: у шестикутній сітці кожен нейрон рівновіддалений від своїх шести сусідів, що призводить до більш рівномірного та симетричного шаблону зв’язку. Ця властивість забезпечує більш плавне та природне представлення вхідного простору, оскільки відстані між нейронами є більш послідовними та ізотропними.

Зменшення викривлення: гексагональна топологія сприяє зменшенню викривлень у просторі даних, оскільки шестикутники мають більш регулярну та симетричну форму, що дозволяє краще зберігати відстані та структуру вхідних даних у порівнянні з прямокутниками.

Більш ефективне представлення даних: завдяки рівномірному зв’язку та покращеній симетрії, гексагональні сітки можуть ефективніше представляти складні структури даних та виявляти приховані шаблони, що сприяє більш якісній кластеризації та візуалізації результатів.

Щоб реалізувати топологію гексагональної сітки в SOM, нейрони зазвичай розташовуються у двовимірній решітці, де кожен нейрон ідентифікується своїми координатами (i, j). Відстань між двома нейронами в гексагональній сітці можна обчислити за допомогою гексагональної метрики відстані, яка враховує унікальний шаблон зв’язності гексагонального розташування. Розглянемо для цього наступну формулу:

$$d_{hex}\big((i_1, j_1), (i_2, j_2)\big) = max(|i_1 - i_2|, |j_1 - j_2|, |i_1 + j_1 - i_2 - j_2|), \quad (2.19)$$

де i_1, j_1 – координати першого нейрона;

i_2, j_2 – координати другого нейрона.

Топологія гексагональної сітки враховує максимум абсолютних різниць координат з урахуванням діагональної зв’язності, що забезпечує більш рівномірний зв’язок між нейронами. При застосуванні цієї топології до розв'язання RCPSP з використанням SOM і ГА, карта служить для кластеризації та представлення рішень, створених ГА. Кожен нейрон на карті відповідає певній області в просторі рішень, а ваги нейронів представляють характеристики рішень у цій області. Гексагональне розташування дозволяє більш природно та плавно відобразити простір рішень, фіксуючи схожість і зв’язки між різними рішеннями [20].

Під час навчання SOM адаптує свої ваги на основі вхідних рішень і впливу сусідства, визначеного гексагональною відстанню. Гексагональна топологія гарантує, що оновлення вагових коефіцієнтів нейрона враховує рішення в його найближчому сусідстві, зберігаючи локальну структуру простору рішень. Це особливо корисно в RCPSP, де подібні рішення можуть мати загальні характеристики або подібні шаблони розподілу ресурсів [7].

Топологія гексагональної сітки також полегшує візуалізацію та інтерпретацію отриманої карти. Розташувавши нейрони у формі шестикутника, карта може ефективно виділяти кластери схожих рішень, визначати області високоякісних рішень і виявляти закономірності чи зв’язки між рішеннями. Це візуальне представлення може надати цінну інформацію про структуру простору рішень і керувати дослідженням та експлуатацією перспективних регіонів ГА.

Таким чином, топологія гексагональної сітки є ефективним вибором для розташування нейронів у самоорганізуючих картах. Порівняно з прямокутною топологією сітки, вона пропонує покращене підключення, покращену локальну взаємодію, зменшене спотворення та кращу візуальну інтерпретацію [18]. У застосуванні до інтеграції SOM і ГА для вирішення RCPSP, топологія гексагональної сітки забезпечує більш природне та гладке представлення простору рішень, полегшуючи кластеризацію та дослідження високоякісних рішень [16].

2.8 Висновки до розділу

У результаті цього розділу було проведено глибокий аналіз структури ГА та SOM, їх архітектурних особливостей та впливу на вирішення задачі RCPSP. Розглядалися як позитивні, так і негативні аспекти використання цих методів.

Завдяки проведеному аналізу було визначено техніки, які будуть використані в гібридному алгоритмі. До них належать методи селекції, мутації, вибору топології SOM, спосіб навчання та інші параметри. Крім того, було сформовано послідовність кроків для виконання гібридного алгоритму.

Цей систематичний підхід дозволяє ефективно інтегрувати ГА та SOM для покращення якості рішень у задачах RCPSP, забезпечуючи баланс між дослідженням та експлуатацією простору рішень.

3 РЕАЛІЗАЦІЯ ГІБРИДНОГО АЛГОРИТМУ ДЛЯ РОЗВ'ЯЗАННЯ ЗАДАЧІ ПЛАНУВАННЯ ІТ-ПРОЄКТІВ З ОБМЕЖЕНИМИ РЕСУРСАМИ

3.1 Опис інструментів для реалізації гібридного алгоритму

Реалізація гібридного алгоритму, який поєднує генетичний алгоритм (ГА) та самоорганізуючу карту (SOM), вимагає сучасних інструментів для забезпечення різноманіття засобів й задоволення поставлених вимог до якості алгоритму. У якості мови програмування було обрано Python v.3.9.0.

Вибір мови програмування Python для реалізації гібридного алгоритму SOM-GA для вирішення задачі розподілу обмежених ресурсів і планування (RCPSP) обумовлений рядом факторів та переваг, які надає ця мова для розробки складних алгоритмічних рішень.

Python є високорівневою мовою програмування, яка надає широкий спектр вбудованих функцій та бібліотек, що значно полегшує розробку складних алгоритмів. Завдяки простоті синтаксису та зручності в роботі з даними, Python дозволяє швидко прототипувати та тестувати різні підходи. Це особливо важливо при розробці гібридних алгоритмів, які потребують інтеграції кількох методів та оптимізації їх взаємодії.

Крім того, Python має велику кількість спеціалізованих бібліотек, які можуть бути корисні при реалізації ГА та SOM. Зокрема, бібліотеки, такі як NumPy і SciPy, надають потужні інструменти для роботи з числовими даними і матрицями, що є основою багатьох алгоритмів машинного навчання. Бібліотека Scikit-learn пропонує реалізацію багатьох алгоритмів

машинного навчання, включаючи SOM, а DEAP (Distributed Evolutionary Algorithms in Python) є популярним інструментом для створення генетичних алгоритмів.

Вибір Python також забезпечує легкість інтеграції з іншими інструментами та платформами, що важливо для створення комплексних систем. Крім того, велика спільнота розробників і доступність великої кількості навчальних матеріалів роблять Python чудовим вибором для розробки та підтримки складних алгоритмічних рішень.

Перелік усіх використаних бібліотек:

- math;
- random;
- os;
- timeit;
- pylab;
- minisom;
- numpy;
- matplotlib;
- sklearn.

Для реалізації гібридного алгоритму SOM-GA важливою є бібліотека NumPy, яка містить оптимізовані функції для роботи з багатовимірними масивами та матрицями, а також надає засоби для ефективного виконання математичних операцій. Ця бібліотека дозволяє швидко та зручно маніпулювати великими обсягами даних, що є критичним при розв'язанні задач комбінаторної оптимізації, таких як RCPSP.

Python також має потужні засоби для роботи з візуальними об’єктами, що є ключовими структурами даних при розв'язанні RCPSP. Оскільки алгоритм SOM часто використовується для

візуалізації багатовимірних даних на карті, яка візуально зрозуміла для людини, у роботі було виконано візуалізацію популяцій індивідів. Для цього використовувалися бібліотека Matplotlib та її графічні компоненти, які дозволяють будувати складні об’єкти відповідно до обраної архітектури SOM.

Бібліотека Math надає набір математичних функцій та констант, які використовуються для різноманітних обчислень у процесі роботи алгоритму. Зокрема, функції з цієї бібліотеки застосовуються для розрахунку розміру карти SOM, що є важливим параметром для ефективного навчання та кластеризації індивідів.

Таким чином, використання цих бібліотек забезпечує ефективну реалізацію гібридного алгоритму, об'єднуючи потужні інструменти для роботи з даними, візуалізацію та математичні обчислення.

Бібліотека Random є важливою для генерації псевдовипадкових чисел, що відіграє ключову роль у реалізації стохастичних операцій генетичного алгоритму. Вона використовується для випадкового вибору індивідів під час ініціалізації початкової популяції, а також для внесення випадкових змін у хромосоми під час мутації. Це забезпечує різноманітність популяції та дозволяє дослідити різні області простору пошуку.

Бібліотека Os надає функції для взаємодії з операційною системою, зокрема для роботи з файловою системою. Вона використовується для зчитування вхідних даних проекту з файлів, що дозволяє гнучко налаштовувати алгоритм для різних задач RCPSP без необхідності модифікації вихідного коду.

Timeit — це бібліотека для вимірювання часу виконання фрагментів коду. Вона застосовується для оцінки швидкодії різних компонентів алгоритму та вимірювання загального часу роботи програми. Це дає змогу аналізувати ефективність запропонованого підходу та порівнювати його з іншими методами.

Scikit-learn — це бібліотека машинного навчання, яка надає різноманітні інструменти для обробки та аналізу даних. У даному випадку використовується клас MinMaxScaler для нормалізації вхідних даних перед навчанням SOM, що покращує якість кластеризації та запобігає зміщенню ваг.

Окрім наведених аргументів, слід зазначити, що на даній мові програмування існує велика кількість бібліотек та репозиторіїв, написаних спільнотою програмістів, що дозволяє використовувати код з відкритим доступом та пришвидшувати написання коду. Таким чином, було обрано бібліотеку minisom, де вже існує стандартний функціонал для SOM, що дозволило використати існуючі методи.

За основу реалізації було взято відкритий репозиторій з кодом ГА. Даний репозиторій використовує методи, описані у розділі 2. Логіку алгоритму побудовано на основі.

3.2 Опис реалізації гібридного алгоритму

Реалізація гібридного алгоритму вимагає чіткої структури майбутнього коду [6]. Оскільки мова програмування Python підтримує об’єктно-орієнтовану парадигму, то було використано

класи для структуризації даних та методів. Таким чином стає зрозуміло якими характеристиками володіють певні об’єкти.

На рисунку 3.1 наведено діаграму класів гібридного алгоритму.

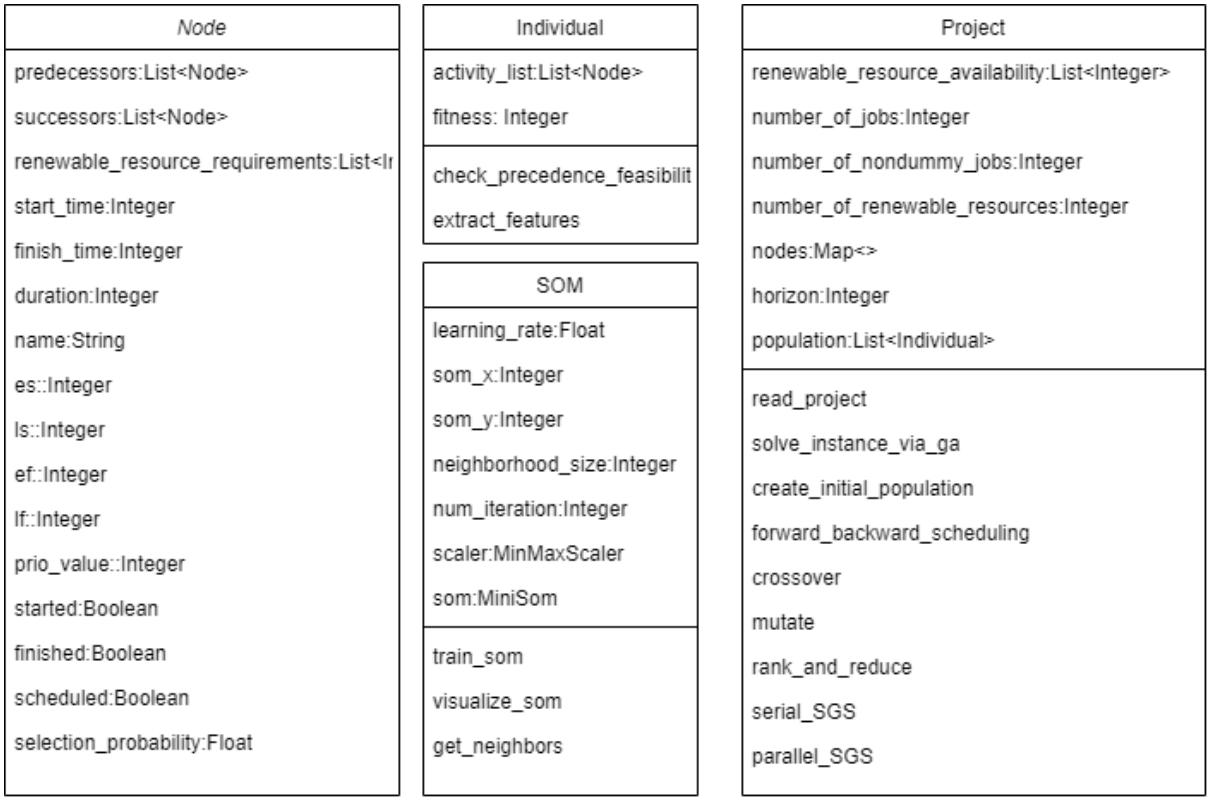

Рисунок 3.1 – Діаграма класів (виконано самостійно)

З наведеного рисунку можна побачити розподіл методів та полів для кожного класу, які логічно віднесено до необхідних структур.

Клас Node відповідає за репрезентацію кожної активності проєкту. Оскільки кожна активність представляє собою складний об’єкт, то вона має доволі велику кількість пов’язаних полів для подальшої обробки.

Клас Individual представляє собою, власне, графік проєкту та характеризується переліком активностей і методами, необхідними для перевірки їх послідовності. Такий графік має відповідати поставленим вимогам та дотримуватися

послідовності подій. Ця логіка є критично важливою, оскільки без неї не було б відповідності до ситуацій з реальних проєктів [5].

Клас Som реалізує обгортку над MiniSom з бібліотеки minisom. Це необхідно для пристосування основного класу до особливостей RCPSP та ідеї, яка пропонується в даній роботі, а саме пошуку сусідів найбільш відповідного індивіда для різноманіття популяції та додаткової візуалізації потенційних результатів на двовимірній карті.

Останній клас Project — містить основну логіку ГА та вилучення даних з тестового набору. Наведені методи використовують механізми, які детально описано в минулому розділі. Зчитування тестових даних відбувається з набору, поширеного у відкритому доступі в мережі інтернет, який зазвичай використовується для тестування робіт.

Ця структура класів забезпечує модульність і дозволяє легко управляти складними об’єктами, які пов'язані з процесом планування і оптимізації розкладів. Взаємодія між класами відбувається через чітко визначені інтерфейси, що сприяє гнучкості та розширюваності системи, полегшуючи майбутні модифікації та покращення [4].

Розглянемо елементи кожного класу, які наведено в таблицях 3.1-3.4.

Таблиця 3.1 – Опис елементів класу Node (виконано самостійно)

Назва	Призначення	Тип даних	Модифікатор доступу
predecessors	Попередні активності у вигляді списку	list	public
successors	Наступні активності у вигляді списку	list	public
renewableResourceRequirements	Вимоги до поновлюваних ресурсів у вигляді списку	list	public
startTime	Час початку активності	int	public
finishTime	Час завершення активності	int	public
duration	Тривалість активності	int	public
name	Назва активності	str	public
es	Ранній час початку активності	int	public
ls	Пізній час початку активності	int	public
ef	Ранній час завершення активності	int	public
lf	Пізній час завершення активності	int	public

Кінець таблиці 3.1

1	2	3	4
prioValue	Пріоритетне значення активності	int	public
started	Прапорець, що вказує, чи розпочата	bool	public
finished	Прапорець, що вказує, чи завершена	bool	public
scheduled	Прапорець, що вказує, чи запланована активність	bool	public
selectionProbability	Ймовірність вибору активності	float	public

Таблиця 3.2 - Опис елементів класу Project (виконано самостійно)

Назва	Призначення	Тип даних	Модифікатор доступу
renewableResourceAvailability	Список доступності поновлюваних ресурсів	list	public
numberOfJobs	Кількість робіт у проекті	int	public
numberOfNndummyJobs	Кількість робіт у проекті, виключаючи фіктивні	int	public

Кінець таблиці 3.2

1	2	3	4
numberOfRenewableResources	Кількість типів поновлюваних ресурсів	int	public
nodes	Словник активностей проекту	dict	public
horizon	Горизонт планування (максимальна тривалість проекту)	int	public
population	Список індивідів у популяції	list	public

Таблиця 3.3 - Опис елементів класу Individual (виконано самостійно)

Назва	Призначення	Тип даних	Модифікатор доступу
somX	Розмір карти SOM по осі X	int	public
somY	Розмір карти SOM по осі Y	int	public
learningRate	Швидкість навчання SOM	float	public
neighborhoodSize	Розмір околу в SOM	int	public
numIteration	Кількість ітерацій навчання SOM	int	public
scaler	Скейлер для нормалізації даних	MinMaxScal	public
som	Об'єкт мапи SOM	MiniS	public

Діаграму активностей варіантів реалізації пошуку сусідів для диверсифікації популяції наведено на рисунку 3.2.

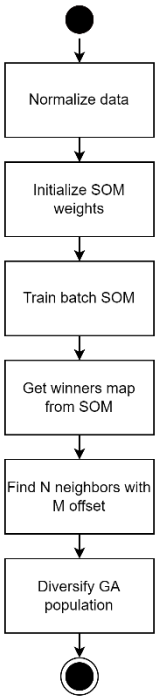

Рисунок 3.2 – Діаграма активностей варіантів реалізації диверсифікації популяції генетичного алгоритму (виконано самостійно)

На основі рисунку 3.2 можна спостерігати, що першим кроком є нормалізація даних, за якою слідує ініціалізація вагів та тренування самоорганізуючих карт (SOM). В результаті навчання отримується карта переможців, що складається з координат у двовимірному просторі. Однак через велику схожість графіків може виявитися неправомірним пошук найближчих сусідів, тому була введена змінна offset. Ця змінна визначає кількість найближчих сусідів, які слід пропустити для вибору переможців з інших кластерів, що дозволяє регулювати різноманіття індивідів у популяції.

Після вибору сусідів і пошуку відповідних індивідів у ненормалізованих даних вони заміщують встановлену кількість індивідів у популяції, що призводить до отримання бажаного результату.

Операція навчання SOM може бути трудомісткою, залежно від встановлених параметрів. Для запобігання можливим затримкам в обчисленнях було вирішено проводити навчання лише в залежності від параметру конвергенції, який визначає, скільки поколінь ГА найкращий результат придатності не змінюється, і при його досягненні відбувається навчання.

3.3 Висновки до розділу

Отримана реалізація гібридного алгоритму використовує сучасні засоби та підходи до програмування. Розроблено програмний код, який об'єднує два вказані алгоритми і включає проміжню візуалізацію результатів навчання SOM.

Для покращення розуміння роботи гібридного алгоритму було створено діаграму класів, що використовуються в процесі реалізації, з детальним описом кожного класу. Також була розроблена діаграма активностей, яка демонструє, яким чином проводиться диверсифікація популяції з метою уникнення локального оптимуму.

Повний текст програмного коду наведено в додатку А.

4 АПРОБАЦІЯ РЕЗУЛЬТАТІВ КВАЛІФІКАЦІЙНОЇ РОБОТИ

4.1 Загальний опис апробації

Апробація розробленого гібридного алгоритму SOM-GA для задачі RCPSP має на меті порівняння його ефективності зі стандартним генетичним алгоритмом. Для цього використовується широко відомий у науковій спільноті набір тестових даних, який використовується для оцінки алгоритмів планування проєктів з обмеженими ресурсами.

Процес апробації включає проведення серії запусків стандартного ГА і гібридного SOM-GA з різними параметрами, такими як розмір популяції, ймовірності схрещування та мутації, розмір карти SOM та кількість ітерацій. Результати кожного запуску фіксуються та усереднюються для отримання достовірних оцінок продуктивності алгоритмів. Ключовими показниками ефективності, які враховуються під час апробації, є час виконання проєкту та швидкість збіжності алгоритмів. За допомогою статистичного аналізу результатів визначається значущість відмінностей між стандартним ГА та гібридним SOM-GA.

Окрім кількісного аналізу, апробація включає якісну оцінку роботи гібридного алгоритму, зокрема візуалізацію процесу навчання SOM та кластеризації рішень. Це допомагає краще розуміти динаміку пошуку оптимальних рішень та роль SOM у підтримці різноманітності популяції та запобіганні передчасній збіжності.

Таким чином, апробація демонструє не лише чисельні переваги гібридного алгоритму SOM-GA, але й візуальні аспекти, що ілюструють процес адаптації та ефективності у вирішенні задачі RCPSP.

4.2 Опис вхідних даних

Для тестування гібридного алгоритму SOM-GA було використано набір тестових даних J30, який є одним зі стандартних наборів для оцінки алгоритмів розв'язання задачі RCPSP. У цьому наборі міститься 450 задач, кожна з яких складається з 30 активностей проєкту.

Задачі в наборі J30 мають різноманітні характеристики, що дозволяє оцінити ефективність алгоритмів в різних умовах, таких як:

– кількість ресурсів: задачі містять різну кількість ресурсів, які необхідні для виконання активностей проєкту;

– складність мережі: задачі відрізняються за структурою мережі активностей, що визначає залежності та обмеження між ними;

– коефіцієнт використання ресурсів: задачі мають різний ступінь завантаженості ресурсів, що впливає на складність планування;

– тривалість активностей: задачі містять активності з різною тривалістю виконання.

Приклад вхідних даних з наведеного набору тестових даних наведено у таблиці 4.1.

Таблиця 4.1 – Вхідні дані для перевірки гібридного алгоритму (виконано самостійно)

Характеристика		Опис	Дані
projects		Кількість проєктів	1
jobs (incl. supersource/sink)		Кількість робіт, в тому числі фіктивні	32
horizon		Максимальний термін проєкту	166
resources	renewable	Кількість поновлюваних ресурсів	4
	nonrenewable	Кількість непоновлюваних ресурсів	0
	doubly constrained	Кількість подвійно обмежених ресурсів	0

Таблиця 4.3 – Вхідні дані з інформацією про послідовність активностей проєкту (виконано самостійно)

Характеристика	Опис	Дані
jobnr.	Ідентифікатор активності	1
#modes	Кількість режимів	1
#successors	Кількість пов’язаних наступних активностей	3
successors	Активності пов’язаних наступників	2 3 4

Таблиця 4.4 – Вхідні дані з інформацією про тривалість активностей проєкту та їх запити щодо ресурсів (виконано самостійно)

Характеристика	Опис	Дані
jobnr.	Ідентифікатор активності	1
mode	Режим	1
duration	Тривалість активності	10
R1	Ресурс 1	6
R2	Ресурс 2	9
R3	Ресурс 3	7
R4	Ресурс 4	10

Для апробації гібридного алгоритму SOM-GA у задачі планування проєктів використовується стандартний набір даних J30. Цей набір містить інформацію про активності проєкту, їх тривалості, потреби у ресурсах і відношення передування між ними. Особливо важливим є визначення доступності ресурсів, яка відбувається за допомогою вказаних значень часу для кожного з ресурсів. Кожен ресурс має власну показник доступності та у тестовому наборі це записується як R1 = 17, R2 = 15, R3 = 17, R4 = 17.

Кожна задача в наборі J30 представлена у вигляді файлу з описом активностей, їх тривалості, потреб у ресурсах та відношень передування. Ці файли є вхідними даними для алгоритмів планування проєктів, які мають на меті знайти оптимальний розклад виконання активностей з урахуванням обмежень на ресурси та відношень передування.

Використання стандартного набору даних J30 для апробації гібридного алгоритму SOM-GA дозволяє отримати

порівнювані результати з іншими дослідженнями та оцінити ефективність запропонованого підходу в контексті відомих задач RCPSP.

4.3 Хід апробації гібридного алгоритму

Апробація гібридного алгоритму передбачає процес виконання програми двома способами: з використанням існуючого підходу, тобто стандартного ГА, та з використанням запропонованого гібридного підходу, який поєднує ГА з самоорганізаційними картами Кохонена (SOM-GA). Метою такого порівняння є оцінка ефективності та доцільності застосування гібридного алгоритму для розв'язання задачі RCPSP.

Перш ніж перейти до безпосереднього розгляду відмінностей у результатах роботи обох підходів, доцільно проаналізувати вихідні дані, отримані в процесі роботи гібридного алгоритму SOM-GA. Ці дані можуть надати цінну інформацію про характеристики алгоритму та особливості його роботи.

Одним з ключових аспектів вихідних даних гібридного алгоритму є візуалізація процесу навчання SOM та кластеризації індивідів. SOM дозволяє відобразити багатовимірний простір пошуку на двовимірну сітку нейронів, зберігаючи топологічні властивості даних. Візуалізація карти SOM може показати розподіл індивідів у просторі пошуку, виявити кластери схожих розв'язків та допомогти зрозуміти структуру простору пошуку.

Аналіз візуалізації SOM може надати інформацію про те, наскільки ефективно алгоритм досліджує простір пошуку, чи формуються чіткі кластери розв'язків, та чи відбувається конвергенція до оптимальних або близьких до оптимальних розв'язків. Якщо на карті SOM спостерігаються чіткі кластери з низькою внутрішньокластерною відстанню та високою міжкластерною відстанню, це може свідчити про ефективність алгоритму у знаходженні різноманітних та якісних розв'язків.

Крім того, вихідні дані гібридного алгоритму містять інформацію про найкращі знайдені розв'язки, а саме їх фітнес-значення. Аналіз цих даних дозволяє оцінити якість отриманих розв'язків, перевірити дотримання обмежень на ресурси та пріоритетні відношення між активностями, а також порівняти результати з відомими оптимальними розв'язками або розв'язками, отриманими іншими методами.

Такий підхід допомагає зрозуміти, чи досягає алгоритм стабільних результатів, чи здатен уникати локальних оптимумів та чи забезпечує він хорошу різноманітність розв'язків. Це особливо важливо для комплексних задач, де потрібно балансувати між якістю розв'язків та часом, необхідним для їх знаходження.

Розглянемо візуалізацію SOM на рисунку 4.1.

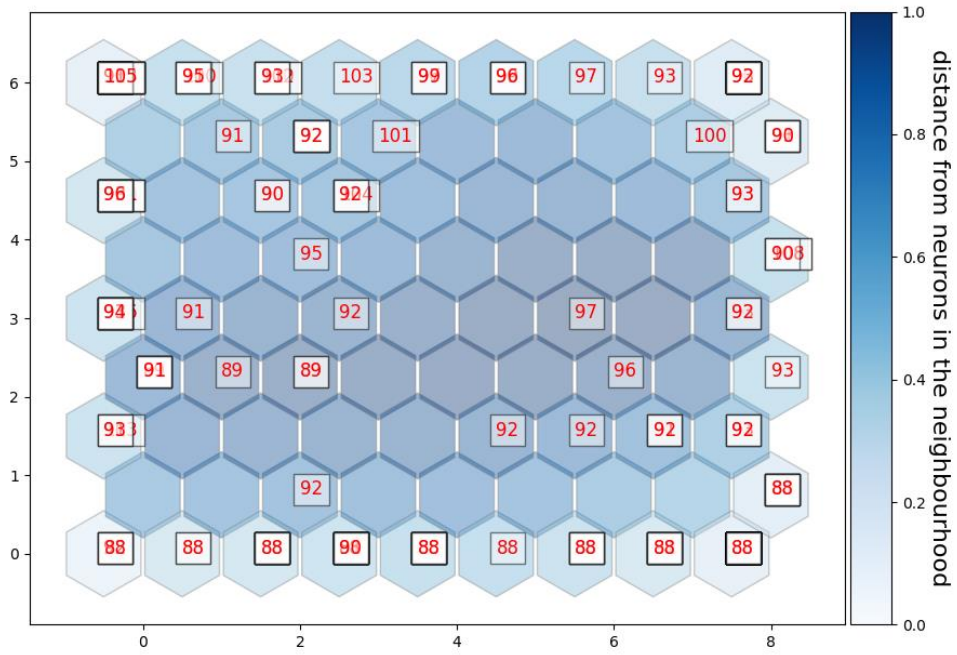

Рисунок 4.1 – Візуалізація кінцевого виклику навчання самоорганізаційних карт Кохонена (виконано самостійно)

На даному рисунку візуально помітний розподіл між різними індивідами відповідно до їх тривалостей проєкту. Індивіди поділяються на кластери, кожен з яких можна виділити візуальним шляхом. Кластеризація відбувається на основі схожості послідовностей активностей, за що відповідає процес навчання у самоорганізаційних картах Кохонена (SOM). З кожною ітерацією кількість вхідних даних збільшується, оскільки з періодичністю в 5 кроків додаються нові індивіди для розширення популяції для навчання. Таким чином, карта з кожним навчанням змінюється та формує нові кластери відповідно до наявних даних.

У результаті кожного запуску програму у консоль виводиться найкраща придатність з усієї популяції. Приклад наведено на рисунку 4.2.

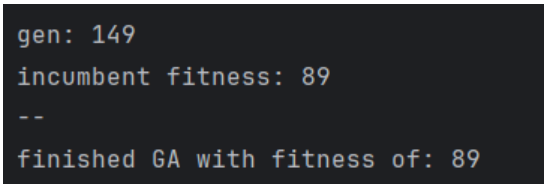

Рисунок 4.2 – Приклад консольного виводу після виконання програми (виконано самостійно)

Іншим аспектом вихідних даних гібридного алгоритму є статистичні показники, такі як похибка квантування та топографічна похибка. Ці показники допомагають оцінити стабільність і ефективність роботи алгоритму, а також зробити висновки про вплив різних параметрів на його продуктивність.

Похибка квантування (quantization error) є мірою відмінності між вхідними векторами та їх найближчими нейронами-переможцями на карті SOM після навчання. Вона показує, наскільки добре карта SOM апроксимує розподіл вхідних даних. Для кожного вхідного вектора знаходиться нейрон-переможець — нейрон, ваговий вектор якого має найменшу евклідову відстань до вхідного вектора. Похибка квантування обчислюється як середнє значення квадратів відстаней між кожним вхідним вектором та його нейроном-переможцем.

Топографічна похибка (topographic error) є мірою збереження топології вхідних даних на карті самоорганізаційної карти Кохонена (SOM) після навчання. Вона показує, наскільки добре карта SOM зберігає просторові відношення між вхідними векторами. Для кожного вхідного вектора знаходяться два нейрони-переможці: перший та другий за величиною активації. Якщо ці два нейрони не є сусідами на карті SOM, вважається, що для даного вхідного вектора топологія не була збережена.

Топографічна похибка обчислюється як частка вхідних векторів, для яких перший та другий нейрони-переможці не є сусідами.

На рисунку 4.3 зображено графік з результатами наведених похибок протягом 10000 ітерацій.

```
Quantization error: 0.8987231844731277
Topographic error: 0.12903225806451613
```

Рисунок 4.3 – Похибка квантизації та топографічна (виконано самостійно)

В результаті проведення навчання похибка мала значні коливання, але згодом зменшувалась.

Для виконання апробації гібридного алгоритму було взято один із наборів даних, описаних раніше. Його найкраща тривалість, знайдена евристичним методом, становить 85 одиниць часу. Для проведення досліду було використано два методи, а саме ГА без модифікації інтеграції іншого алгоритму та гібридний алгоритм. Щоб виконати дослід та перевірити поведінку, було проведено 50 ітерацій для кожного методу.

Було використано наступні параметри:

- Кількість індивідів у популяції 50
- Кількість поколінь 150
- Ймовірність мутації 0.25
- Рівень досягання конвергенції 30
- Рівень навчання для SOM 0.7
- Відстань між сусідами 10
- Кількість ітерацій SOM 10000
- Кількість індивідів для заміщення після навчання SOM 20

У результаті отримано наступні результати:

– Для стандартного ГА: середня тривалість 87.56; час виконання 280 секунд.
– Для гібридного алгоритму SOM-GA: середня тривалість 87.28; час виконання 440 секунд.

Відповідно до отриманих результатів апробації гібридного алгоритму SOM-GA, можна зробити висновки, що покращення у роботі алгоритму в порівнянні зі стандартним генетичним алгоритмом відбулось незначне. Хоча інтеграція самоорганізаційних карт Кохонена мала на меті підвищити ефективність пошуку оптимальних розв'язків та запобігти передчасній збіжності, реальний вплив на якість розв'язків виявився обмеженим.

Це може бути пов'язано з кількома факторами:

– Параметри алгоритму: Можливо, параметри гібридного алгоритму не були оптимальними для даної задачі або тестових наборів даних. Налаштування таких параметрів, як кількість ітерацій SOM, рівень навчання, ймовірності схрещування та мутації могли б вплинути на ефективність роботи алгоритму.

– Обчислювальна складність: Додавання SOM збільшує обчислювальну складність алгоритму, що призводить до збільшення часу виконання без значного покращення якості результатів. Використання самоорганізуючих карт Кохонена може бути надмірно ресурсомістким для задач такого типу, особливо якщо простір рішень не є надто складним.

– Вплив кластеризації: Хоча SOM забезпечує кластеризацію рішень, можливо, що простір рішень для RCPSP не настільки складний, щоб виграти від додаткової

кластеризації. Можливо, RCPSP не має значних переваг від кластеризації, яку забезпечує SOM, що обмежує потенційний виграш від інтеграції цих технологій.

Незважаючи на це, гібридний алгоритм SOM-GA показав деяке покращення у середній тривалості виконання проєктів, що свідчить про потенціал цього підходу. Подальші дослідження можуть зосередитись на налаштуванні параметрів алгоритму та оптимізації його обчислювальної ефективності. Крім того, варто дослідити можливість застосування інших підходів до покращення генетичних алгоритмів, які можуть краще відповідати специфіці RCPSP.

Незважаючи на це, гібридний алгоритм SOM-GA показав деяке покращення у середній тривалості виконання проєктів, що свідчить про потенціал цього підходу. Подальші дослідження можуть зосередитись на налаштуванні параметрів алгоритму та оптимізації його обчислювальної ефективності.

Одним із факторів, який необхідно враховувати, є збільшення часу виконання гібридного алгоритму порівняно зі стандартним генетичним алгоритмом. Додаткові обчислення, пов'язані з навчанням та використанням SOM, призводять до зростання обчислювальної складності та, відповідно, до збільшення часу роботи алгоритму.

Це може бути особливо відчутним для великих проектів з великою кількістю активностей та обмежень.

Варто зазначити, що наведені показники ефективності відрізняються в залежності від набору тестових даних, на яких проводилась апробація. Це свідчить про необхідність

подальшого дослідження та аналізу факторів, які впливають на ефективність SOM-GA в різних умовах.

Одним із напрямків для потенційного покращення роботи гібридного алгоритму є зміна стратегії заміщення індивідів на основі результатів кластеризації SOM. У поточній реалізації застосовується підхід, за якого індивіди заміщуються найближчими сусідами відносно найкращого індивіда. Проте, можливо, інші стратегії заміщення, такі як вибір представників з різних кластерів або врахування щільності розподілу індивідів, могли б привести до кращих результатів. Це потребує додаткових досліджень та експериментів.

Для подальшого розвитку та вдосконалення гібридного алгоритму SOM-GA можуть бути запропоновані напрямки, такі як оптимізація параметрів алгоритму, вдосконалення стратегії заміщення індивідів, аналіз ефективності на різних наборах даних, зменшення обчислювальної складності та розширення можливостей візуалізації результатів. Загалом, подальші дослідження та експерименти з налаштуваннями та стратегіями роботи гібридного алгоритму можуть допомогти краще зрозуміти його потенціал та покращити його ефективність для вирішення задач планування проектів з обмеженими ресурсами.

Крім того, оптимізація вхідних параметрів алгоритму, таких як розмір популяції, ймовірності мутації та схрещування, розмір карти SOM і кількість ітерацій навчання, може суттєво вплинути на ефективність розв'язання задачі RCPSP. Проте, пошук оптимальних значень цих параметрів потребує значних обчислювальних ресурсів та часу, оскільки передбачає багаторазовий запуск алгоритму на різних наборах даних з різними комбінаціями параметрів.

Для підвищення ефективності цього процесу можна використовувати методи автоматичної оптимізації параметрів, такі як грід-пошук, пошук Байєса, генетичні алгоритми, метод випадкового пошуку та адаптивні методи. Застосування цих методів може допомогти знайти оптимальні параметри швидше та ефективніше, що сприятиме підвищенню продуктивності гібридного алгоритму SOM-GA.

Подальші дослідження в цьому напрямку можуть також зосередитись на аналізі впливу різних параметрів на продуктивність алгоритму в різних умовах та на різних наборах даних, що допоможе краще зрозуміти та налаштувати алгоритм для конкретних застосувань у задачах планування проектів з обмеженими ресурсами.

4.4 Висновки до розділу

У результаті проведеної апробації було встановлено, що існуюча реалізація гібридного алгоритму має обмежене покращення ефективності, але гібридний алгоритм SOM-GA залишається перспективним напрямком для подальшого дослідження та вдосконалення. Майбутні дослідження можуть зосередитись на пошуку більш ефективних стратегій інтеграції SOM та ГА, оптимізації параметрів алгоритму, а також на аналізі факторів, які впливають на його ефективність в різних умовах.

Додаткові дослідження в цій області можуть надати більш оптимальні результати та принести додаткову користь у сфері планування ІТ-проєктів. Використання сучасних методів оптимізації параметрів, таких як грід-пошук, пошук Байєса, генетичні алгоритми та адаптивні методи, може значно підвищити продуктивність гібридного алгоритму. Аналіз впливу різних параметрів на продуктивність алгоритму в різних умовах допоможе краще зрозуміти його роботу та налаштувати для конкретних застосувань у задачах планування проектів з обмеженими ресурсами.

ВИСНОВКИ

У процесі кваліфікаційної роботи було вивчено методи вирішення задачі планування IT-проєктів з обмеженими ресурсами за допомогою засобів штучного інтелекту.

У першому розділі проведено огляд та аналіз існуючих методів планування проєктів, таких як метод критичного шляху, метод критичних ланцюгів та генетичні алгоритми. Визначено перспективність використання генетичних алгоритмів у поєднанні з самоорганізаційними картами Кохонена для вирішення задачі планування проєктів з обмеженими ресурсами.

Другий розділ присвячено дослідженню модифікації генетичного алгоритму для розв'язання задачі RCPSP шляхом інтеграції з самоорганізаційними картами Кохонена. Проаналізовано структуру та архітектурні особливості генетичних алгоритмів і карт Кохонена, визначено методики для застосування у гібридному алгоритмі SOM-GA. Сформовано послідовність кроків для виконання гібридного алгоритму.

Третій розділ охоплює програмну реалізацію гібридного алгоритму SOM-GA із застосуванням сучасних інструментів та підходів. Створено діаграму класів та діаграму активностей, що відображають структуру та логіку роботи алгоритму. Реалізовано механізм диверсифікації популяції генетичного алгоритму за допомогою карт Кохонена.

Четвертий розділ присвячено апробації розробленого гібридного алгоритму SOM-GA на тестових даних. Проведено порівняльний аналіз ефективності гібридного алгоритму та стандартного генетичного алгоритму. Отримані результати

свідчать про обмежене покращення ефективності гібридного підходу, але вказують на перспективність подальших досліджень у цьому напрямку.За результатами дослідження можна зробити висновок, що використання генетичних алгоритмів у поєднанні з самоорганізаційними картами Кохонена є перспективним підходом для вирішення задачі планування IT-проєктів з обмеженими ресурсами. Запропонований гібридний алгоритм SOM-GA демонструє потенціал для покращення ефективності пошуку оптимальних розкладів проєктів, однак потребує подальшого вдосконалення та дослідження.

Напрямками майбутніх досліджень можуть бути оптимізація параметрів алгоритму, пошук ефективніших стратегій інтеграції карт Кохонена з генетичним алгоритмом, а також аналіз факторів, що впливають на ефективність гібридного підходу в різних умовах. Практична цінність розробленого програмного засобу полягає в можливості його використання для оптимізації планування IT-проєктів.

За тематикою кваліфікаційної роботи було опубліковано тези доповіді на тему «Дослідження методів ШІ для планування IT-проектів».

Кваліфікаційну роботу виконано відповідно до методичних вказівок.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008:2015. Інформація та документація. Звіти у сфері науки і техніки. Структура та правила оформлювання, Чинний від 22.06.2015. Київ: ДП «УкрНДНЦ», 2016, 26 с.2.

2. ДСТУ 8302:2015 «Інформація та документація. Бібліографічне посилання. Загальні положення та правила складання».

3. Iso 12207. Process Improvement with CMMI® v1.2 and ISO Standards. 2008. С. 321–357. URL: https://doi.org/10.1201/9781420052848.axf (дата звернення: 01.05.2024).

4. Iso 15288. Process Improvement with CMMI® v1.2 and ISO Standards. 2008. С. 295–319. URL: https://doi.org/10.1201/9781420052848.axe (дата звернення: 01.05.2024).

5. Artificial intelligence and machine learning / N. Kühl та ін. Electronic Markets. 2022. URL: https://doi.org/10.1007/s12525-022-00598-0 (дата звернення: 11.05.2024).

6. Florez-Perez L., Song Z., Cortissoz J. C. Using machine learning to analyze and predict construction task productivity. Computer-Aided Civil and Infrastructure Engineering. 2022. URL: https://doi.org/10.1111/mice.12806 (дата звернення: 13.05.2024).

7. Shelar V., Saranathan G., Rezaei S. What challenges can you expect when training HM models?. URL: https://www.linkedin.com/advice/3/what-challenges-can-you-expect-when-training-ann-txsre (дата звернення: 15.05.2024).

8. Patnaik M., Melani Adrian A. Cognitive Big Data Intelligence with a Metaheuristic Approach. 2022. 356 c.

9. Artigues C. The Resource-Constrained Project Scheduling Problem. Resource-Constrained Project Scheduling. London, UK. С. 19–35. URL: https://doi.org/10.1002/9780470611227.ch1 (дата звернення: 20.05.2024).

10. Zhang H., Xu H., Peng W. A Genetic Algorithm for Solving RCPSP. 2008 International Symposium on Computer Science and Computational Technology, м. Shanghai, China, 20–22 груд. 2008 р. 2008. URL: https://doi.org/10.1109/iscsct.2008.255 (дата звернення: 20.05.2024).

11. Eiben A. E., Smith J. E. Introduction to Evolutionary Computing. Berlin, Heidelberg : Springer Berlin Heidelberg, 2015. URL: https://doi.org/10.1007/978-3-662-44874-8 (дата звернення: 20.05.2024).

12. Cetin U., Gundogmus Y. E. Feature Selection with Evolving, Fast and Slow Using Two Parallel Genetic Algorithms. 2019 4th International Conference on Computer Science and Engineering (UBMK), м. Samsun, Turkey, 11–15 верес. 2019 р. 2019. URL: https://doi.org/10.1109/ubmk.2019.8907165 (дата звернення: 23.05.2024).

13. Kohonen T. Self-Organizing Maps. Berlin, Heidelberg : Springer Berlin Heidelberg, 2001. URL: https://doi.org/10.1007/978-3-642-56927-2 (дата звернення: 20.05.2024).

14. Amor H. B., Rettinger A. Intelligent exploration for genetic algorithms. the 2005 conference, м. Washington DC, USA,

25–29 черв. 2005 р. New York, New York, USA, 2005. URL: https://doi.org/10.1145/1068009.1068250 (дата звернення: 14.05.2024).

15. Kan S., Fei Z., Kita E. Application of self-organizing maps to genetic algorithms. OPTI 2009, м. Algarve, Portugal, 8–10 черв. 2009 р. Southampton, UK, 2009. URL: https://doi.org/10.2495/op090011 (дата звернення: 14.05.2024).

16. Liu D., Wang X., Du J. A Clustering-Based Evolutionary Algorithm for Traveling Salesman Problem. 2009 International Conference on Computational Intelligence and Security, м. Beijing, China, 11–14 груд. 2009 р. 2009. URL: https://doi.org/10.1109/cis.2009.80 (дата звернення: 14.05.2024).

17. Cheng M.-Y., Huang K.-Y. GENETIC ALGORITHM-BASED CHAOS CLUSTERING APPROACH FOR NONLINEAR OPTIMIZATION. Journal of Marine Science and Technology. 2010. Т. 18, № 3. URL: https://doi.org/10.51400/2709-6998.1891 (дата звернення: 14.04.2024).

18. Cheng M.-Y., Tran D.-H., Wu Y.-W. Using a fuzzy clustering chaotic-based differential evolution with serial method to solve resource-constrained project scheduling problems. Automation in Construction. 2014. Т. 37. С. 88–97. URL: https://doi.org/10.1016/j.autcon.2013.10.002 (дата звернення: 14.06.2024).

19. Rajput M. Building generations by Genetic algorithm. URL: https://ai.plainenglish.io/building-generations-by-genetic-algorithm-1c02ebf6fd8b (дата звернення: 22.05.2024).

20. Advances in Computational Intelligence / ред.: I. Rojas, G. Joya, A. Catala. Cham : Springer International

Publishing, 2015. URL: https://doi.org/10.1007/978-3-319-19222-2 (дата звернення: 18.06.2024).

21. NumPy documentation – NumPy v1.26 Manual. NumPy. URL: https://numpy.org/doc/stable/ (дата звернення: 02.06.2024).

22. API Reference – Matplotlib 3.9.0 documentation. Matplotlib – Visualization with Python. URL: https://matplotlib.org/stable/api/index.html (дата звернення: 19.06.2024).

23. math - Mathematical functions. Python documentation. URL: https://docs.python.org/3/library/math.html (дата звернення: 19.06.2024).

24. random - Generate pseudo-random numbers. Python documentation.
URL: https://docs.python.org/3/library/random.html (дата звернення: 19.06.2024).

25. os - Miscellaneous operating system interfaces. Python documentation.
URL: https://docs.python.org/3/library/os.html (дата звернення: 19.06.2024).

26. timeit - Measure execution time of small code snippets. Python documentation.
URL: https://docs.python.org/3/library/timeit.html (дата звернення: 19.06.2024).

27. MiniSom is a minimalistic implementation of the Self Organizing Maps. GitHub.
URL: https://github.com/JustGlowing/minisom (дата звернення: 15.06.2024).

28. Python implementation of a Genetic Algorithm for the Resource-Constrained Project Scheduling Problem. GitHub. URL: https://github.com/derhendrik/RCPSP_GA/ (дата звернення: 10.06.2024).

29. Hartmann S. A competitive genetic algorithm for resource-constrained project scheduling. Naval Research Logistics. 1998. Т. 45, № 7. С. 733–750. URL: https://doi.org/10.1002/(sici)1520-6750(199810)45:7%3C733::aid-nav5%3E3.0.co;2-c (дата звернення: 15.05.2024).

30. Successful Critical Chain Project Management | Lucidchart URL: https://www.lucidchart.com/blog/critical-chain-project-management (дата звертання: 20.06.2024).06.2024)

31. Zosym M. Метод критичного шляху (Critical path method).
URL: https://www.maxzosim.com/metod-kritichnogo-shliahu/ (дата звернення: 20.06.2024)

32. Москаленко М.В. Сучасні напрямки програмної інженерії та інноваційні системи навчання. Машинне навчання та штучний інтелект. Міжнародний молодіжний форум «Радіоелектроніка і молодь в XXI столітті», який відбувся 16-18 квітня 2024р.

33. The five PMBOK process groups URL: https://www.projectengineer.net/the-five-pmbok-process-groups/ (дата звернення: 20.06.2024)

34. Maxfield M. When genetic algorithms meet artificial intelligence. URL: https://www.eejournal.com/article/when-genetic-algorithms-meet-artificial-intelligence/ (дата звернення: 20.06.2024)