Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет Автоматики і комп'ютеризованих технологій
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

# КВАЛІФІКАЦІЙНА РОБОТА
# Пояснювальна записка

Другий (магістерський)
(півень вищої освіти)

Розроблення підсистеми підтримки прийняття рішень для оптимізації маршрутів внутрішньозаводської логістики
(тема)

Виконав:
студент II курсу, групи АУТПм-22-2

Чернишенко О. В.
(прізвище, ініціали)

Спеціальності 151 Автоматизація та комп'ютерно-інтегровані технологій
(код і повна назва спеціальності)

Тип програми Освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Автоматизоване управління технологічними процесами
(повна назва освітньої програми)

Керівник проф. Безкоровайний В. В.
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри КІТАР _____ Невлюдов І. Ш.
(підпис) (прізвище, ініціали)

2024 р.

*Я, як студент ХНУРЕ (Чернишенко Олександр Володимирович), розумію і підтримую політику закладу із академічної доброчесності. Я не надав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.*

| Дата | Чернишенко О. В. |
| --- | --- |
| *04.01.2024 р.* |  |

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет *АКТ*
Кафедра *КІТАР*
Рівень вищої освіти *другий (магістерський)*
Спеціальність *151 Автоматизація та комп'ютерно-інтегровані технології*
Тип програми *Освітньо-професійна*
Освітня програма *Автоматизоване управління технологічними процесами*
(шифр і назва)

ЗАТВЕРДЖУЮ
Зав. кафедри КІТАР _____
(підпис)
«__» _____20__ р.

**ЗАВДАННЯ**
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові *Чернишенко Олександру Володимировичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Розроблення підсистеми підтримки прийняття рішень для оптимізації маршрутів внутрішньозаводської логістики*
Затверджена наказом по університету від *03.11.2023 р. № 1286 Ст*
2. Термін подання студентом роботи до екзаменаційної комісії *25.01.2024 р.*
3. Вихідні дані до роботи *Об'єкт дослідження – внутрішньозаводська логістика на промисловому об'єкті. Предмет дослідження – підтримка прийняття рішень при оптимізації логістичних маршрутів. Функція системи – побудова оптимальних логістичних маршрутів. Розмірність задач: кількість пунктів призначень – до 100; кількість транспортних засобів – до 20. Технічне забезпечення: IBM-сумісний персональний комп'ютер. Програмний засіб з віконно-графічним інтерфейсом.*

4. Перелік питань, що потрібно опрацювати в роботі *Вступ. Визначення мети, об'єкту та предмету дослідження. Аналіз сучасного стану керування запасами виробництва. Задачі керування на виробництві. Місце внутрішньозаводської логістики в процесі керування запасами. Задача комівояжера у контексті внутрішньозаводської логістики. Методи розв'язання задачі комівояжера. Постановка задачі дослідження. Формалізація задачі комівояжера. Застосування генетичного алгоритму для вирішення ЗК. Розробка вимог до програмного додатку. Опис основного функціоналу розробленого програмного додатку. Експерименти та аналіз результатів. Висновки.*

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій *Графічний матеріал у вигляді презентації – 10-15 аркушів форматуА4.*

6. Консультанти розділів роботи

| Найменування розділу | Консультант (посада, прізвище, ім'я, по батькові) | Позначка консультанта про виконання розділу | |
|---|---|---|---|
| | | підпис | дата |
| | | | |
| | | | |

**КАЛЕНДАРНИЙ ПЛАН**

| № | Назва етапів роботи | Термін виконання етапів роботи | Примітка |
|---|---|---|---|
| 1 | *Вступ. Визначення мети, об'єкту і предмету дослідження* | *12.11-20.11 2023* | *викон.* |
| 2 | *Огляд сучасного стану керування запасами виробництва* | *20.11-25.11 2023* | *викон.* |
| 3 | *Постановка задачі дослідження* | *25.11-27.11 2023* | *викон.* |
| 4 | *Вибір методів розв'язання* | *27.11-05.12 2023* | *викон.* |
| 5 | *Розробка вимог до програмного засобу* | *05.12-10.12 2023* | *викон.* |
| 6 | *Розробка програмного засобу* | *10.12-15.12 2023* | *викон.* |
| 7 | *Експерименти та аналіз результатів* | *15.12-29.12 2023* | *викон.* |
| 8 | *Розглядання питання охорони праці* | *29.12-01.01 2024* | *викон.* |
| 9 | *Оформлення пояснювальної записки* | *01.01-20.01 2024* | *викон.* |
| 10 | *Подання роботи на перевірку Інтернет-сервісом Unicheck* | *20.01-21.01 2024* | *викон.* |
| 11 | *Подання роботи на рецензію* | *21.01-22.01 2024* | *викон.* |
| 12 | *Подання роботи на підпис зав. кафедри* | *22.01-23.01 2024* | *викон.* |
| 13 | *Подання кваліфікаційної роботи в ЕК* | *23.01-24.01 2024* | *викон.* |

Дата видачі завдання *12.11.*2023 р.

Студент _____ Чернишенко О. В.
(підпис) (прізвище, ініціали)

Керівник роботи _____ проф. Безкоровайний В. В.
(підпис) (посада, прізвище, ініціали)

# РЕФЕРАТ

Пояснювальна записка: 107 с., 10 табл., 27 рис., 3 дод., 33 джерел.

АВТОМАТИЗОВАНА СИСТЕМА УПРАВЛІННЯ, ВНУТРІШНЬОЗАВОДСЬКА ЛОГІСТИСКА, ЛОГІСТИЧНА СИСТЕМА, ПІДТРИМКА ПРИЙНЯТТЯ РІШЕНЬ, ОПТИМІЗАЦІЯ.

Об’єкт дослідження – внутрішньозаводська логістика на промисловому об’єкті.

Предмет дослідження – підтримка прийняття рішень при оптимізації логістичних маршрутів.

Мета кваліфікаційної роботи – підвищення ефективності внутрішньозаводських логістичних систем за рахунок розробки підсистеми підтримки прийняття рішень в процесах керування ними.

Методи дослідження – теорія систем, системний аналіз, теорія прийняття рішень, методи комбінаторної оптимізації.

У кваліфікаційній роботі досліджено одну із складових проблеми керування запасами на виробництві – задачу оптимізації логістичних маршрутів. Для формалізації цієї задачі запропоновано використовувати задачу декількох комівояжерів з обмеженнями (MTSPC). Розроблено підсистему підтримки прийняття рішень у вигляді веб-додатку для знаходження оптимальних логістичних маршрутів, методом гілок та меж та генетичним алгоритмом, опираючись на координати пунктів призначень, кількості транспортних засобів та їх параметри. За результатами проведених експериментів надані рекомендації щодо оптимальних налаштувань параметрів генетичного алгоритму для вирішення задачі комівояжера.

Результати кваліфікаційної роботи апробовані у фаховій статті та двох міжнародних конференціях.

# ABSTRACT

Work contains: 107 pages, 10 tables, 27 figures, 33 sources, 3 applications.

AUTOMATED MANAGEMENT SYSTEM, LOGISTICS SYSTEM, DECISION SUPPORT, OPTIMIZATION.

Object of research – in-plant logistics at an industrial facility.

The subject of research is decision-making support in the optimization of logistics routes.

The purpose of the qualification work is to increase the efficiency of intra-plant logistics systems by developing a decision support subsystem in their management processes.

Research methods: systems theory, systems analysis, decision-making theory, combinatorial optimization methods.

The qualification work investigates one of the components of the problem of inventory management in production – the problem of optimizing logistics routes. To formalize this problem, it is proposed to use the multiply traveling salesman's problem with constraints (MTSPC). A decision support subsystem in the form of a web application has been developed to find optimal logistics routes using the method of branches and boundaries and a genetic algorithm, based on the coordinates of destinations, the number of vehicles and their parameters. Based on the results of the experiments, recommendations were made on the optimal settings of the genetic algorithm parameters for solving the traveling salesman's problem.

The results of the qualification work were tested in a professional article and two international conferences.

**ЗМІСТ**

# ПЕРЕЛІК СКОРОЧЕНЬ

ВЛ – виробнича логістика.

ГА – генетичний алгоритм.

ЗК – задача комівояжера.

ООП – об'єктно-орієнтований підхід.

ППРП – Підсистема підтримки прийняття рішень.

BnB – Branch and Bound Method.

CSS – Cascading Style Sheets.

CX – Cycle Crossover.

DP – Dynamic Programming method.

HTML – HyperText Markup Language.

JS – JavaScript.

MES – Manufacturing Execution Systems.

MFCS – Material Flow Control Systems.

MTSP – Multiple Traveling Salesman Problem.

TSP – Traveling Salesman Problem.

UML – Unified Modeling Language.

# ВСТУП

Внутрішньозаводська логістика істотно впливає на ефективність виробництва, охоплюючи всі аспекти руху матеріалів та продукції всередині підприємства. Оптимізація логістичних маршрутів в цьому контексті виступає як необхідність для мінімізації витрат, підвищення продуктивності, покращення обслуговування, зменшення впливу на довкілля та забезпечення гнучкості та адаптивності у виробничих процесах.

Вітчизняні та зарубіжні науковці значно просунулись у розробці методів оптимізації логістичних мереж [1, 2, 3], однак постійний розвиток технологій і зміни на ринку вимагають нових методів і рішень у сфері внутрішньозаводської логістики.

Одним із сучасних підходів до оптимізації логістичних мереж є використання генетичних алгоритмів, який має наступні переваги: адаптивність до різноманітних завдань, здатність до паралельного пошуку багатьох рішень одночасно, та гнучкість у обробці різних типів обмежень. Тому, є доцільним його реалізація у ППРР, проте необхідно дослідити його роботу в залежності від параметрів, для забезпечення максимальної адаптації до конкретних умов задачі.

Об’єкт дослідження – внутрішньозаводська логістика на промисловому об’єкті.

Предмет дослідження – підтримка прийняття рішень при оптимізації логістичних маршрутів.

Метою кваліфікаційної роботи є підвищення ефективності внутрішньозаводських логістичних систем за рахунок розробки підсистеми підтримки прийняття рішень у процесах їх управління. Для досягнення цієї мети було використано сучасні методи теорії систем, системного аналізу, теорії прийняття рішень та методи комбінаторної оптимізації.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– виконати огляд сучасного стану проблеми керування запасами на виробництві;

– провести аналіз задачі планування маршрутів доставки ресурсів на виробництві;

– дослідити особливості задачі оптимізації логістичних мереж;

– провести аналіз, формалізацію та обрати алгоритми для вирішення задачі оптимізації логістичних мереж;

– розробити програмний додаток у вигляді веб-сайту, у якому буде реалізовано обрані алгоритми;

– провести серію експериментів для надання рекомендацій щодо використання того чи іншого методу оптимізації логістичних мереж;

– оформити пояснювальну записку згідно з рекомендаціями [4], та вимогами ДСТУ 3008:2015 [5].

За результатами дослідження опубліковано статтю у збірнику студентських наукових статей «Автоматизація та приладобудування» ADED-2023(2) [6] (Харківський національний університет радіоелектроніки), підготовлено тези доповіді на двадцять сьомий Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті», конференція «Автоматизовані системи та комп’ютеризовані технології радіоелектронного приладобудування» [7] (Харківський національний університет радіоелектроніки) та Всеукраїнську науково-практичну конференцію «Комп’ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві» [8] (Харківський національний автомобільно-дорожній університет, 22 листопада 2023 р.).

# 1 ОГЛЯД СУЧАСНОГО СТАНУ КЕРУВАННЯ ЗАПАСАМИ ВИРОБНИЦТВА

Забезпечення підприємства – це комплексна система дій, спрямована на надання організації необхідних ресурсів для її нормальної роботи та розвитку. Ці ресурси можуть включати матеріальні активи, фінансові ресурси, інформаційні технології, людські ресурси та інші елементи, необхідні для виконання основних та допоміжних функцій підприємства.

Основна мета забезпечення підприємства полягає в тому, щоб забезпечити стабільне та ефективне функціонування організації. Це включає в себе забезпечення надійності поставок, оптимізацію витрат, якість придбаних ресурсів та їх своєчасність.

Забезпечення підприємства також передбачає стратегічний підхід до управління ресурсами, що охоплює всі аспекти діяльності компанії. Це включає в себе планування, аналіз і оптимізацію використання фінансових, людських, матеріальних та інших ресурсів для досягнення стратегічних цілей та забезпечення стійкого розвитку організації. Такий підхід допомагає ефективно використовувати ресурси, знижувати витрати, підвищувати конкурентоспроможність та досягати успіху на ринку.

## 1.1 Задачі керування на виробництві

На виробництві виникає ряд різноманітних задач керування, що покликані забезпечити ефективність, продуктивність та якість виробничого процесу. Основні аспекти керування, які виникають на виробництві описано нижче [9].

Керування запасами. У виробництві важливо забезпечити своєчасний та ефективний доступ до сировини та матеріалів. Оптимальне керування запасами забезпечує баланс між мінімізацією витрат на складування та

ризиком затримки виробництва через відсутність необхідних компонентів. Це також стосується контролю за готовою продукцією, щоб відповідати попиту без надлишкових витрат на складування.

Керування якістю. Якість – центральний елемент конкурентоспроможності продукту. Управління якістю передбачає систематичний моніторинг та аналіз виробничого процесу, щоб виявляти та виправляти відхилення. Це також включає в себе розробку стандартів, навчання персоналу та взаємодію з постачальниками для забезпечення відповідності сировини стандартам.

Керування персоналом. Ефективність виробництва великою мірою залежить від людського фактору. Управління персоналом включає в себе підбір, навчання, мотивацію та розвиток працівників. Організація робочого процесу, корпоративна культура, умови праці – все це впливає на продуктивність, задоволеність роботою та лояльність персоналу.

Керування технологічним процесом. Технологічний процес має бути не лише ефективним, але й гнучким для адаптації до змінних умов ринку. Управління технологічним процесом включає вибір оптимальних технологій, їх модернізацію та постійне удосконалення, а також контроль за дотриманням технологічних регламентів.

Кожен з цих аспектів має свої особливості та виклики. Для успішного керування виробництвом менеджмент має системно підходити до аналізу, планування та впровадження рішень, зосереджуючись на досягненні стратегічних цілей підприємства.

Проблему керування запасами можна поділити на наступні складові:

– визначення потреб у матеріалах. Визначення які та скільки ресурсів потрібно для виробничого процесу;

– планування закупівель. визначення, коли та від кого купувати ресурси;

– планування маршрутів доставки ресурсів. Оптимізація шляхів доставки матеріалів від постачальників до виробничих ліній.

– контроль запасів. Моніторинг рівнів запасів, щоб забезпечити їх достатність без надмірного накопичення;

– оцінка постачальників. Вибір та оцінка постачальників за якістю, ціною, надійністю та швидкістю доставки;

– прогнозування попиту. Аналіз даних продажів та ринкових тенденцій для прогнозування майбутнього попиту на продукцію.

Розглянемо проблему планування маршрутів доставки більш детально. Оптимізовані маршрути можуть значно зменшити транспортні витрати, що безпосередньо впливає на загальну рентабельність операцій. Точне та своєчасне доставляння ресурсів забезпечує безперебійне виробництво, що, у свою чергу, підвищує здатність компанії задовольняти потреби своїх клієнтів. Оптимізація маршрутів може значно скоротити час, необхідний для доставки ресурсів. Швидка доставка означає, що матеріали швидше потрапляють на виробництво, що зменшує простої та збільшує загальну продуктивність.

Отже, вирішуючи проблему планування маршрутів доставки ресурсів, компанії можуть значно підвищити ефективність своїх операцій, зменшити витрати та підвищити загальну конкурентоспроможність.

## 1.2 Місце внутрішньозаводської логістики в процесі керування запасами

Внутрішньозаводська логістика є ключовим елементом в управлінні запасами на будь-якому виробничому підприємстві. Вона відіграє центральну роль у забезпеченні ефективного руху матеріалів та товарів у межах виробництва, від початкової точки прийому до моменту відправки готової продукції.

Управління запасами має на меті забезпечення доступності необхідних матеріалів для виробничого процесу, при цьому мінімізуючи вартість їх зберігання. Внутрішньозаводська логістика, у свою чергу, спрямована на те, щоб ці матеріали доставлялися у відповідне місце в самий потрібний момент [10, 11]. Таке своєчасне та точне розміщення запасів може значно

зменшити витрати на їх зберігання та допомогти уникнути затримок у виробництві.

Також, правильно організована внутрішньозаводська логістика може сприяти оптимізації використання складських приміщень. Замість того, щоб матеріали надовго залишалися на складі, вони швидко та ефективно переміщаються через виробничий процес. Це не тільки допомагає звільнити цінний простір, але й знижує ризик пошкодження запасів.

Основна мета внутрішньозаводської логістики у контексті управління запасами – це забезпечення плавності виробничого процесу. Якщо матеріали надходять туди, де і коли їх потрібно, виробнича лінія функціонує безперебійно, що призводить до підвищення продуктивності та зниження загальних витрат.

В цілому, внутрішньозаводська логістика та управління запасами є взаємопов’язаними елементами виробничої системи. Вони працюють разом, щоб забезпечити, що ресурси використовуються ефективно, виробничий процес просувається без затримок, а підприємство може відгукнутися на змінювані вимоги ринку.

## 1.3 Задача комівояжера у контексті внутрішньозаводської логістики

Внутрішньозаводська логістика вивчає та оптимізує процеси переміщення товарів, матеріалів та інформації всередині підприємства або заводу. Ці процеси часто можуть бути складними та вимагають ефективного планування, щоб забезпечити мінімізацію витрат часу, людських ресурсів і матеріалів. Проте, попри це, внутрішньозаводська логістика може стикатися з рядом проблем.

Однією з ключових проблем є неефективність маршрутів доставки матеріалів чи готової продукції в межах заводу. Це може призвести до зайвих витрат на перевезення, збільшення часу доставки і непотрібних затримок в виробничому процесі. Саме тут і виникає задача комівояжера.

Задача комівояжера – це класична оптимізаційна проблема в теорії графів та комбінаторної оптимізації [12, 13]. Суть задачі полягає в знаходженні найкоротшого можливого маршруту, який б відвідав кожне місто (або точку) лише один раз і повертався до початкової точки. Існують різні варіації цієї задачі (рисунок 1.1).

У контексті внутрішньозаводської логістики це може бути застосовано для визначення найефективнішого маршруту для перевезення матеріалів або готової продукції всередині підприємства.

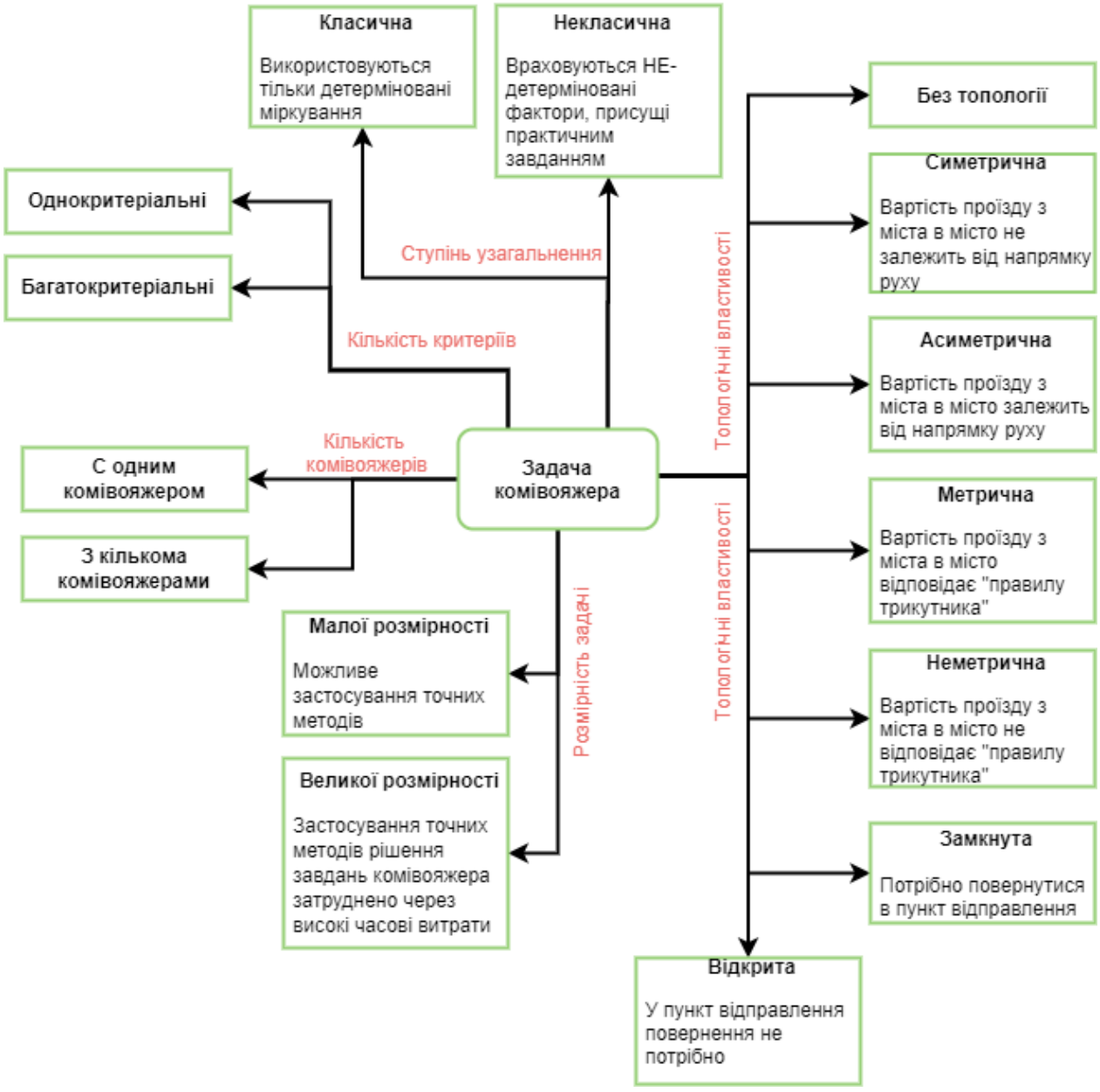


Рисунок 1.1 – Класифікація задач комівояжера

Класична задача комівояжера зосереджена на знаходженні найкоротшого маршруту, який б проходив через всі вказані точки лише один раз і повертався до початкової позиції. Однак, в умовах внутрішньозаводської логістики ця модель може бути недостатньою, оскільки виробничі умови набагато складніші та різноманітніші, ніж просте об’єднання точок маршруту.

Внутрішньозаводська логістика часто має справу з різними обмеженнями, такими як час роботи певних відділень, доступність ресурсів або специфічні вимоги до перевезення деяких матеріалів. Тому задача комівояжера з обмеженнями більше відповідає реальності заводської логістики.

Додатково, на великих заводах або комплексних виробничих об’єктах може бути декілька вантажних машин, кожна з яких виконує свої завдання. Тому задача про декілька комівояжерів стає доречною, так як вона моделює ситуації, коли декілька транспортних засобів одночасно працює над доставкою товарів або матеріалів до різних частин підприємства.

Використання класичної моделі задачі комівояжера може призвести до неефективних маршрутів у реальних виробничих умовах. Наприклад, маршрут може бути розрахований так, що він проходить через відділення, яке працює лише у певний час, або може ігнорувати вимоги зберігання певних матеріалів.

Таким чином, для ефективного управління логістикою на виробничому підприємстві часто потрібно використовувати модифікації задачі комівояжера, які відображають специфіку і реальні обмеження даного підприємства.

## 1.4 Методи розв’язання задачі комівояжера

Методи розв’язання задачі комівояжера [14] можна поділити на дві основні категорії: точні та наближені (рисунок 1.2).

Точні методи розв’язання задачі комівояжера знаходять маршрут з мінімальною вартістю. Їх основна перевага полягає в тому, що вони

гарантують оптимальний результат. Однак вони можуть бути обчислювально складними, особливо для великих наборів даних, що робить їх неефективними для використання в реальному часі або при великих розмірностях задачі.

З іншого боку, наближені методи [15] використовуються для швидкого знаходження наближеного рішення. Вони зазвичай набагато швидші, ніж точні методи, і є більш практичними для великих наборів даних. Наближені методи часто використовуються в реальному часі, коли швидкість є критичною. Проте вони не гарантують оптимального рішення.

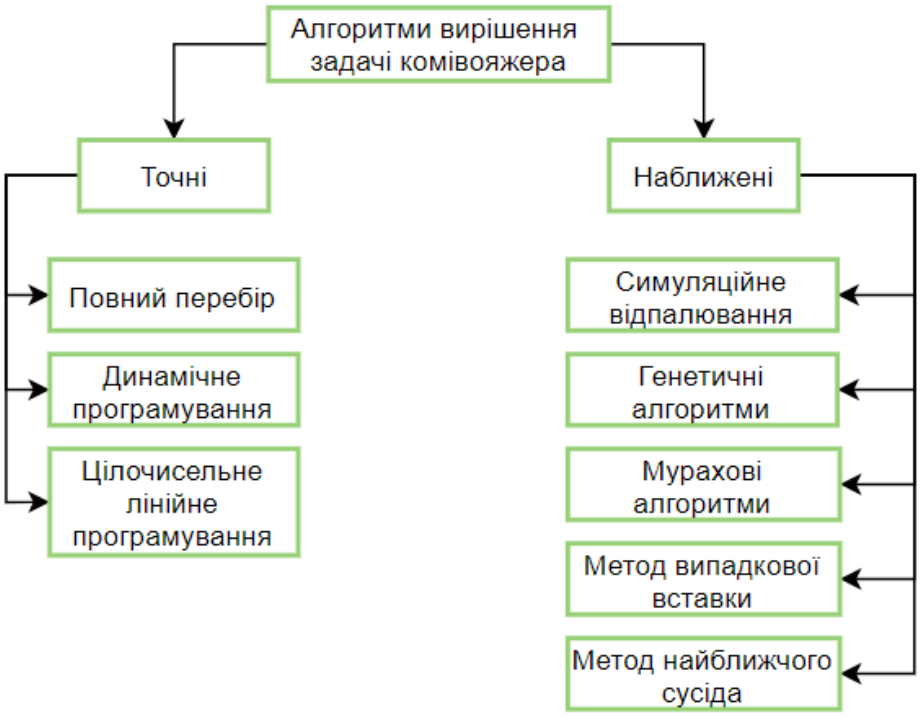


Рисунок 1.2 – Методи розв’язання задачі комівояжера

## 1.5 Постанова мети та задач роботи

Проаналізувавши сучасний стан проблеми керування запасами на виробництві, можна зробити висновок, що проблема оптимізації логістичних маршрутів залишається актуальної, а методи її вирішення досі потребують оптимізації, через постійне зростання вимог до точності та швидкості розв’язання.

Задача оптимізації логістичної мережі полягає у визначенні найефективнішого способу розподілу ресурсів, який дозволить мінімізувати загальні витрати та час доставки. Основна ціль полягає в тому, щоб знайти оптимальне рішення, яке б враховувало вартість транспортування, зберігання, обробку, а також забезпечувало б відповідність всім необхідним термінам і вимогам.

У процесі оптимізації враховуються різні змінні рішення, такі як розташування та кількість складів, маршрути доставки, розмір та частота вантажних партій, а також розподіл товарів між різними ланками ланцюга постачання. Ці змінні підлягають обмеженням, що включають ємність складів, транспортні засоби, часові рамки, та інші.

Не оптимальність логістичних маршрутів на виробництві може спричинити надлишкові витрати на транспортування, подовження термінів доставки та зайві затримки у процесі виробництва. Отже, є необхідність в розробці підсистеми підтримки прийняття рішень для оптимізації маршрутів внутрішньозаводської логістики.

Метою даної роботи є підвищення ефективності внутрішньозаводських логістичних систем за рахунок розробки підсистеми підтримки прийняття рішень в процесах керування ними, а саме для побудови оптимальних логістичних маршрутів доставки ресурсів в середині виробництва.

Для досягнення мети кваліфікаційної роботи необхідно розв’язати такі задачі:

– виконати огляд сучасного стану проблеми керування запасами на виробництві;

– провести аналіз задачі планування маршрутів доставки ресурсів на виробництві;

– дослідити особливості задачі комівояжера у контексті внутрішньозаводської логістики;

– провести аналіз, та обрати алгоритми для вирішення задачі комівояжера;

– розробити програмний додаток у вигляді веб-сайту, у якому буде реалізовано обрані алгоритми;

– провести серію експериментів для надання рекомендацій щодо використання того чи іншого методу вирішення ЗК.

## 1.6 Висновки до першого розділу

У цьому розділі було розглянуто проблему керування запасами на виробництві, було виділено ключові аспекти такі як керування запасами, якістю, персоналом та технологічним процесом, кожен з яких має важливе значення для ефективності та продуктивності виробничого процесу.

Значна увага була приділена внутрішньозаводській логістиці як ключовій складовій управління запасами, яка включає ефективний рух матеріалів та товарів у межах виробництва. Також було розглянуто задачу комівояжера в контексті оптимізації маршрутів доставки в межах виробництва. Розглянуто дві основні категорії методів розв’язання задачі комівояжера: точні та наближені.

Аналіз сучасного стану керування запасами на виробництві вказує на актуальність проблеми оптимізації логістичних маршрутів, з високими вимогами до точності та швидкості вирішення цих завдань. Разом з тим, розробка ефективних підходів до розподілу ресурсів є критичною для мінімізації витрат та часу доставки, враховуючи вартість транспортування, зберігання, обробки, а також необхідність дотримання всіх термінів та вимог.

Неоптимізовані логістичні маршрути ведуть до збільшення витрат, подовження часу доставки та затримок у виробництві, тому існує суттєва потреба у вдосконаленні систем підтримки прийняття рішень для внутрішньозаводської логістики.

Отже, можна зробити висновок, що існує потреба у засобах підтримки прийняття рішень для оптимізації маршрутів внутрішньозаводської логістики, з використанням сучасних методів та алгоритмів.

# 2 ПОСТАНОВКА ЗАДАЧІ ТА ВИБІР МЕТОДІВ РОЗВ'ЯЗАННЯ

Основною задачею внутрішньозаводської логістики є оптимізація маршрутів та розподіл ресурсів для ефективного переміщення матеріалів та продукції в межах промислового об'єкта. Задача декількох комівояжерів з обмеженнями може бути використана для моделювання та вирішення задач цього типу.

В контексті внутрішньозаводської логістики, комівояжери можуть представляти транспортні засоби або робочий персонал, відповідальний за перевезення матеріалів між різними локаціями на заводі. Обмеження можуть включати вантажопідйомність транспортних засобів, часові вікна для доставки або вивантаження, обмеження на робочий час персоналу, а також доступність доріжок та обладнання для переміщення матеріалів.

Рішення MTSP може бути інтегроване з системами управління виробництвом (Manufacturing Execution Systems, MES [16]) або системами управління матеріальними потоками (Material Flow Control Systems, MFCS [17]) для автоматизації планування та моніторингу внутрішньозаводської логістики в реальному часі.

## 2.1 Постановка задачі дослідження

Метою даної роботи є підвищення ефективності внутрішньозаводських логістичних систем за рахунок розробки підсистеми підтримки прийняття рішень в процесах керування ними. Конкретні цілі дослідження включають в себе:

– розробку математичної моделі задачі декількох комівояжерів з урахуванням обмежень, які характерні для внутрішньозаводської логістики, такі як вантажопідйомність транспортних засобів, часові вікна для доставки або вивантаження, обмеження на робочий час персоналу, доступність доріжок

та обладнання для переміщення матеріалів, тощо;

– вибір та реалізацію оптимізаційних алгоритмів для знаходження найкращих маршрутів та метою мінімізації часу та витрат при переміщенні матеріалів та продукції на підприємстві;

– розробку програмного засобу, який буде реалізувати алгоритму пошуку оптимальних маршрутів;

– проведення експериментальних досліджень та валідація розроблених алгоритмів на задача різної розмірності.

Розроблювальна підсистема підтримки прийняття рішень для оптимізації маршрутів внутрішньозаводської логістики (далі – програмний додаток) призначена для автоматизації процесу планування та оптимізації маршрутів переміщення товарів та матеріалів всередині підприємства. Ця система допомагає мінімізувати витрати часу та ресурсів, а також забезпечує точну інформацію для керівництва підприємства щодо логістичних операцій.

Очікувана тривалість прийняття рішення в залежить від конкретної задачі. Прокладання оптимального маршруту може бути виконано в реальному часі або попередньо розраховано з певним часовим запасом.

Зв’язок програмного додатку із загальною системою внутрішньозаводської логістики полягає в обміні даними та інформацією. Додаток отримує дані про замовлення, розміщення кінцевих пунктів, характеристики вантажів та кількість транспортних засобів від інших систем управління логістикою. Після обробки цих даних, вона генерує оптимальні маршрути та надсилає їх для подальшого виконання.

Вхідна інформація для розроблювального додатку включає такі дані:

– замовлення та їх характеристики (вага, об’єм, часові обмеження, тощо);

– інформація про кінцеві точки доставки, включаючи їх географічне розташування та доступність;

– дані про транспортні засоби, включаючи їх характеристики та кількість.

Вихідна інформація, яку надає програмний додаток, включає в себе оптимальні маршрути для виконання завдань та час розв’язання.

Розроблювальна підсистема підтримки прийняття рішень допомагає підприємствам ефективно управляти внутрішньозаводською логістикою, знижуючи витрати та підвищуючи продуктивність логістичних операцій.

## 2.2 Формалізація задачі комівояжера

Класичну задачу комівояжера (TSP, Traveling Salesman Problem) у контексті внутрішньозаводської логістики можна формалізувати наступним чином [18]: маємо $N$ пунктів призначень, між кожним пунктом відома відстань $C \in R^{N \times N}$. Треба побудувати такий маршрут $X^* \in R^{N \times N}$, який проходить кожний пункт 1 раз, крім початкового міста, та має мінімальну довжину. Тобто потрібно знайти замкнений гамільтонів цикл мінімальної довжини. Математична постановка класичної задачі комівояжера полягає у знаходженні такого маршруту:

$$X^* = \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} x_{ij}\,, \tag{2.1}$$

який буде задовольняти обмеження:

$$\sum_{i=1}^{N} x_{ij} = 1,\ j = \overline{1, N}\,, \quad \sum_{j=1}^{N} x_{ij} = 1, i = \overline{1, N} \tag{2.2}$$

$$u_i - u_j + N x_{ij} \le N - 1,\ i, j = \overline{1, N},\ i \ne j\,, \tag{2.3}$$

де $N$ – кількість пунктів призначення;

$X^*$ – маршрут обходу пунктів призначення;

$d_{ij}$ – вага (відстань, вартість) переходу з $i$-го пункту в $j$-й;

$x_{ij}$ – наявність переходу з $i$-го пункту в $j$-й, $x \in \{0,1\}$;

$u_j$ – номер кроку, на якому було відвідано $j$-й пункт;

$u_i$ – номер кроку, на якому було відвідано $i$-й пункт.

Формула (2.1) визначає цільову функцію – пошук маршруту з мінімальною вартістю. Обмеження (2.2) задають умову відвідування кожного пункту тільки один раз, а обмеження (2.3) забезпечує умову цілісності маршруту.

Задача декількох комівояжерів (MTSP, Multiple Traveling Salesman Problem) є загальним випадком задачі комівояжера [19]. Мета полягає в тому, щоб знайти найкоротший маршрут для кожного з $m$ комівояжерів, так, щоб кожний пункт призначення був відвіданий один раз одним комівояжером. Маршрути обходу кожного комівояжера починаються і закінчуються в одному і тому же пункті.

Таким чином, (2.1) набуває вигляду:

$$X^* = \sum_{k=1}^{m}\sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij} x_{ijk} \,. \tag{2.4}$$

з обмеженнями:

$$\sum_{k=1}^{m}\sum_{i=1}^{N} x_{ij} = 1,\ j = \overline{1,N}\,,\ \sum_{k=1}^{m}\sum_{j=1}^{N} x_{ij} = 1, i = \overline{1,N}\,; \tag{2.5}$$

$$u_i - u_j + N x_{ij} \le N - 1,\ \ i, j = \overline{1,N},\ \ i \ne j\,; \tag{2.6}$$

$$\sum_{k=1}^{m} x_{ijk} \ge 1, i = \overline{1,N},\ j = \overline{1,N}\,, \tag{2.7}$$

де $m$ – кількість комівояжерів;

$x_{ijk}$ – наявність переходу з $i$-го пункту в $j$-й для $k$-го комівояжера, $x \in \{0,1\}$.

Обмеження (2.7) задає умову того, щоб кожний комівояжер відвідав хоча б однин пункт.

Задача декількох комівояжерів з обмеженнями MTSPC (Multiple Traveling Salesman Problem with Constraint), наприклад на вантажопідйомність комівояжера є розширеним варіантом MTSP задачі. У цій задачі кілька комівояжерів повинні відвідати всі пункти призначення, при цьому кожний пункт відвідується рівно один раз, і повернутися назад до початкового пункту, враховуючи обмеження на вантажопідйомність кожного комівояжера. В цьому випадку цільова функція (2.4) залишається незмінною, але додаються нові обмеження:

$$0 \le l_k \le Q; \tag{2.8}$$

$$l_k = \sum_{i=1}^{N} q_i x_{ijk},\ j = \overline{1,N},\ k = \overline{1,m}, \tag{2.9}$$

де $Q$ – вантажопідйомність комівояжера;

$l_k$ – накопичена вага товару, яку несе $k$-й комівояжер;

$q_i$ – запит на товар для $i$-го пункту.

Таким чином, (2.8) обмежує максимальну довжину маршруту за накопиченою вагою (об’ємом) товару.

## 2.3 Вхідні данні до задачі комівояжера

Вхідні дані встановлюють основні параметри задачі та визначають середовище, у якому розв’язується задача. До них належать:

– кількість пунктів призначень $N$;

– кількість комівояжерів $m$;

– початковий та кінцевий пункт призначення, будемо вважати що маршрути починаються і закінчуються у пункті з номером $N$;

– певні обмеження на маршрут (вантажопідйомність, час);

– вага (відстань, вартість) переходу з $i$-го пункту в $j$-й, $d_{ij}$.

Множину пунктів призначень можна представити у вигляді неорієнтованого повнозв'язного графа. Розглянемо детальніше як задається вага на ребрах графу. Відстань між пунктами задається у вигляді матриці відстаней.

Матриця відстаней у задачі комівояжера – це квадратна матриця розміром $N \times N$, що представляє відстані між парами міст. Елемент матриці $d_{ij}$, $i = \overline{1,N}, j = \overline{1,N}$ представляє собою відстань між $i$-тим та $j$-тим пунктом. Матриця може бути симетричною, тобто відстань між пунктами призначення однакова у обох напрямках ($d_{ij} = d_{ji}$) або ні ($d_{ij} \neq d_{ji}$). Відстані мають бути позитивними, $d_{ij} \geq 0$. Головна діагональ, тобто відстань від пункту призначення до самого себе вважається нульовою, $d_{ii} = 0$.

Якщо вершини графа є точками на площині з координатами $x_i$ та $y_i$, $i = \overline{1,N}$, то відстань визначається за допомогою евклідової (2.10), манхетенської (2.11) або інших метрик відстані:

$$d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}; \quad (2.10)$$

$$d_{ij} = |x_i - x_j| + |y_i - y_j|. \quad (2.11)$$

Існує прийом, який дозволяє звести задачу MTSP до TSP, а саме задача $m$ комівояжерів з центральною базою. База (початковий пункт) має номер $N+1$. Нехай $C'$ позначає $(N+1) \times (N+1)$ матрицю відстаней задачі $m$ комівояжерів. Введемо $m-1$ фіктивну вершину з номерами $N+2, ..., N+m$ та

приймемо значення $d_{ij}^{'}$ рівними:

$$d_{ij}^{'} = d_{ij},\ i,j = \overline{1,N+m},\quad d'_{i,N+j} = d_{i,N+1},\ i = \overline{1,N+m},\ j = \overline{2,m};\qquad (2.12)$$

$$d'_{N+i,j} = d_{N+1,j},\ j = \overline{1,N+m},\ i = \overline{2,m},\quad d'_{N+i,N+j} = \infty,\ i,j = \overline{2,m}.$$

Таким чином, вирішивши задачу TSP для матриці $C^{'}$, вирішується задача MTSP для матриці $C$ з $m$ комівояжерами. В кінці необхідно злити всі пункти маршруту з номерами, які більше $N$, в одну з номером $N$. Недоліком цього методу є збільшення розмірності матриці, тим чином збільшується розрахункова складність задачі.

## 2.4 Метод гілок та меж для вирішення ЗК

Метод гілок та меж (Branch and Bound, B&B) – це загальна назва алгоритмів для знаходження оптимальних рішень у різних задачах оптимізації. Він систематично перевіряє всі можливі рішення, обмежуючи при цьому область пошуку за допомогою оцінки нижньої та верхньої меж.

Команда дослідників, до складу якої входили Дж. Літтл, К. Мурті, Д. Суїні та К. Керолл, представила варіант методу гілок і меж [20], який був адаптований конкретно для вирішення проблеми комівояжера. Пізніше ця версія алгоритму отримала назву «метод Літтла» на честь Дж. Літтла, першого з авторів цього дослідження.

Алгоритм гілок та меж для вирішення задачі комівояжера [21] можна сформулювати у вигляді наступних правил:

Крок 1. Знаходимо в кожному рядку матриці відстаней $C$ мінімальний елемент і віднімаємо його від усіх елементів відповідного рядка. Отримаємо матрицю, приведену по рядках. Якщо в матриці, яка приведена по рядках, існують стовпці, що не містять нуля, то приводимо її по стовпцях.

Крок 2. Знаходимо суму значень мінімальних елементів, тобто визначаємо константу приведення (2.13), яка буде нижньою межею множини всіх допустимих гамільтонових циклів.

$$Lb = \sum_{i=1}^{N} \min(row_i) + \sum_{j=1}^{N} \min(column_j), \tag{2.13}$$

де $Lb$ – нижня межа для вузла;

$\min(row_i)$ – мінімальний елемент $i$-го рядка після редукції;

$\min(column_j)$ – мінімальний елемент $j$-го рядка після редукції.

Крок 3. Для кожного нульового елемента приведеної матриці розрахуємо степінь нульового елемента. Для цього нуль в матриці замінюємо на знак «*» і знаходимо суму мінімальних елементів рядка і стовпця, які відповідають цьому нулю.

Крок 4. Вибираємо ребро ($i_0$, $j_0$), для якого степінь нульового елемента досягає максимального значення.

Крок 5. Розбиваємо множину всіх гамільтонових циклів $\Omega^0$ на дві підмножини $\Omega^1_{i_0 j_0}$ та $\Omega^1_{\overline{i_0 j_0}}$. Підмножина $\Omega^1_{i_0 j_0}$ включає в себе ребро ($i_0$, $j_0$), а $\Omega^1_{\overline{i_0 j_0}}$ його не містить. Для отримання матриці вартості для підмножини $\Omega^1_{i_0 j_0}$ викреслюємо рядок $i_0$ та стовпець $j_0$. Щоб не допустити виникнення негамільтонового циклу, потрібно знайти рядок $i$ і стовпець $j$ без знаку $\infty$, та на їх перетині поставити $\infty$.

Крок 6. Приводимо матрицю гамільтонових циклів $\Omega^1_{i_0 j_0}$ та розраховуємо нижню межу даної множини $\varphi(\Omega^1_{i_0 j_0})$. Аналогічні дії виконуємо для множини гамільтонових циклів $\Omega^1_{\overline{i_0 j_0}}$.

Крок 7. Порівнюємо нижні межі підмножин гамільтонових циклів $\Omega^1_{i_0 j_0}$ і $\Omega^1_{\overline{i_0 j_0}}$. Якщо $\varphi(\Omega^1_{i_0 j_0}) < \varphi(\Omega^1_{\overline{i_0 j_0}})$, то подальшому галуженню підлягає множина $\Omega^1_{i_0 j_0}$, інакше – $\Omega^1_{\overline{i_0 j_0}}$. Процес розбиття множин на підмножини супроводжується побудовою дерева розгалужень.

Крок 8. Якщо в результаті розгалужень отримуємо матрицю $2\times 2$, то визначаємо гамільтоновий цикл, який отримано розгалуженням, та розраховуємо його довжину $f(x)$.

Крок 9. Порівнюємо довжину гамільтонового циклу $f(x)$ з нижніми межами обірваних гілок. Якщо довжина $f(x)$ не перевищує їхніх нижніх меж, то задача вирішена. В іншому випадку розбиваємо гілки підмножин з нижньою межею, меншою отриманого шляху, до тих пір, поки не отримаємо маршрут з меншою довжиною або не переконаємося, що такого не існує

Перевага методу Літтла полягає в тому, що він дозволяє знаходити точне рішення для задачі комівояжера та враховує специфіку цієї задачі, забезпечуючи ефективність. Недоліком цього методу є значна обчислювальна складність при роботі з великими наборами даних.

Метод гілок та меж — це потужний інструмент для розв’язання задач оптимізації, але він може бути неефективним для дуже великих даних. У таких випадках можуть бути корисними евристичні методи або методи, які комбінують евристику з методами оптимізації.

## 2.5 Метод динамічного програмування для вирішення ЗК

Метод динамічного програмування (DP) є важливим інструментом для розв’язання оптимізаційних задач. У контексті задачі комівояжера, метод динамічного програмування може бути використаний для знаходження оптимального маршруту з найменшою загальною відстанню [22, 23]. Однією з найвідоміших реалізацій DP для цієї задачі є алгоритм Хелда-Карпа.

Алгоритм Хелда-Карпа використовує концепцію динамічного програмування, щоб розбити оригінальну задачу на менші підзадачі. Це досягається шляхом збереження результатів підзадач у таблиці для подальшого використання, що допомагає уникнути повторного вирішення одних і тих самих підзадач.

Рекурсивні відношення в алгоритмі Хелда-Карпа відіграють центральну роль у розв'язанні задачі комівояжера методом динамічного програмування, оскільки вони дозволяють ефективно обчислити найкоротший шлях, який відвідає всі міста та завершиться в початковому місті. Ці відношення визначаються для підзадачі задачі комівояжера, де розглядається підмножину міст і визначається найкоротший шлях, що включає всі міста з цієї підмножини та закінчується в певному місті. Розглянемо ці відношення детальніше.

Рекурсивна формула визначає мінімальну вартість маршруту до міста $j$ через всі міста в підмножині $S$, враховуючи вартість маршрутів до кожного іншого міста $k$ в $S$ та відстань від міста $k$ до міста $j$:

$$C(S, j) = \min_{k \in S, k \neq j} \{C(S - \{j\}, k) + d_{kj}\}, \tag{2.14}$$

де $d_{kj}$ – відстань від міста $k$ до міста $j$;

$C(S, j)$ – мінімальна вартість маршруту, який проходить через всі міста в підмножині $S$ і закінчується у місті $j$;

$S$ – підмножина міст, які потрібно відвідати;

$j$ – кінцеве місто маршруту в підмножині $S$.

Базовий випадок рекурсивних відношень встановлює вартість маршруту для початкового міста: $C(\{1\},1) = 0$. Рекурсивні відношення використовують попередні результати для обчислення нових значень. Результати зберігаються в таблиці, щоб уникнути повторного обчислення тих самих підзадач.

Ці рекурсивні відношення допомагають розбити первинну задачу на менші підзадачі, які можна розв’язати ефективніше, а потім об’єднати, щоб отримати рішення оригінальної задачі.

Алгоритм Хелда-Карпа має часову складність $O(n^2 \cdot 2^n)$, де $n$ - кількість пунктів призначень в задачі. Алгоритм також вимагає значного обсягу пам’яті, оскільки він зберігає результати всіх підзадач в таблиці для подальшого використання. Вимоги до пам’яті складають $O(n \cdot 2^n)$, оскільки для кожної з $2^n$ можливих підмножин пунктів призначень і кожного з $n$ пункту у кожній підмножині потрібно зберегти одне значення вартості.

Ці характеристики роблять алгоритм Хелда-Карпа значно ефективнішим, ніж простий повний перебір (який має часову складність $O(n!)$), але все ще недостатньо ефективним для великих задач. Через високі вимоги до часу та пам’яті, алгоритм Хелда-Карпа найкраще підходить для задач комівояжера з відносно малим числом міст.

## 2.6 Застосування генетичного алгоритму для вирішення ЗК

Генетичні алгоритми (GA) – це метод оптимізації, який наслідує принципи еволюції, такі як селекція, схрещування та мутація, для знаходження наближених рішень складних проблем. Вони працюють з популяцією потенційних рішень і еволюціонують її через покоління, поступово покращуючи якість рішень. GA широко використовується у складних оптимізаційних та пошукових задачах, де інші методи можуть виявитися неефективними. Застосування генетичних алгоритмів для вирішення як і TSP так і MTSP з обмеженнями детально описано в [24-27].

Генетичний алгоритм для задачі комівояжера базується на ідеї еволюції і природного відбору. У цьому контексті, маршрути комівояжера представлені як хромосоми (або індивіди) в популяції. Кожна хромосома є послідовністю міст, які комівояжер повинен відвідати. Щоб визначити якість кожного

маршруту (хромосоми), використовується функція пристосованості. У випадку задачі комівояжера, зазвичай ця функція пристосованості заснована на загальній довжині маршруту:

$$f(x) = d(x_j, x_i) + \sum_{i=1}^{N-1} d(x_i + x_{i+1}) + d(x_N, x_j), \qquad (2.15)$$

де $f(x)$ – функція пристосованості для маршруту $x$;

$j$ – номер початкового (і кінцевого) міста;

$d(x_i + x_{i+1})$ – відстань між містами $x_i$ та $x_{i+1}$;

$N$ – кількість пунктів призначень.

В процесі еволюції, хромосоми схрещуються і мутують, щоб створити нові маршрути. Схрещування зазвичай включає в себе обмін підпослідовностями між двома батьківськими маршрутами, тоді як мутація може включати в себе перестановку двох пунктів призначень у маршруті. Після генерації нових маршрутів використовується відбір для визначення, які хромосоми перейдуть до наступного покоління. Хромосоми з кращою пристосованістю мають більшу ймовірність бути вибраними. Генетичний алгоритм продовжується до тих пір, поки не буде досягнуто певної умови зупинки, наприклад, певна кількість поколінь чи стабілізація результатів.

Розглянемо особливості генетичного алгоритму, який буде використано для вирішення задачі MTSP з обмеженнями в підсистемі підтримки прийняття рішень для системи керування внутрішньозаводською логістикою.

Хромосоми представлені у вигляді одномірного масиву міст. Кожне місто є числом, яке характеризує його номер, і хромосома (або індивід) це масив цілих чисел, що представляють порядок відвідування міст. Приклад кодування маршруту для приведеної задачі MTSP до TSP, згідно методу описаного в пункті 2.2, наведений на рисунку 2.1. На ньому наведений маршрут обходу десяти міст для трьох комівояжерів.

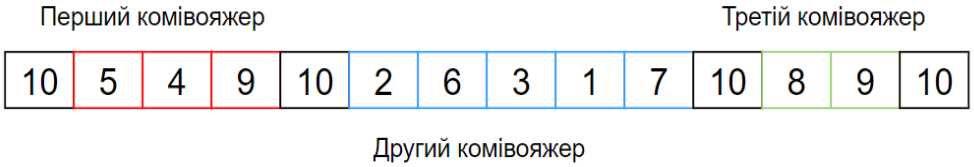


Рисунок 2.1 – Кодування маршруту

Початкове покоління, тобто початкові маршрути обходу міст, генерується випадковим чином. При цьому враховується початкове місто, яке завжди є першим та останнім у списку, що представляє маршрут.

Відбір для схрещування виконується за допомогою турнірного відбору. Турнірна селекція – це метод відбору у генетичних алгоритмах, де декілька індивідів випадково вибираються з популяції і проводиться «турнір» між ними. Індивід з найкращою пристосованістю вибирається як переможець. Алгоритм турнірної селекції наведено нижче.

1. Випадково виберіть $k$ індивідів з популяції.

2. Оцінюється пристосованість кожного індивіда у вибраному наборі. Індивід з найвищим значенням пристосованості є переможцем турніру.

3. Переможець турніру додається до нової популяції.

4. Кроки 1-3 повторюються потрібну кількість разів, щоб заповнити нову популяцію.

Турнірна селекція є досить ефективним і популярним методом відбору в генетичних алгоритмах, особливо коли потрібно зберегти генетичне різноманіття в популяції.

Схрещування. Використовується циклічне схрещування, де дочірні хромосоми формуються шляхом обміну генів між батьками в залежності від деякого випадково обраного початкового пункту.

Циклічний кросовер (CX) – це особлива техніка схрещування для генетичних алгоритмів, що розроблена для задач, в яких хромосоми представлені як послідовності, і де кожен елемент послідовності може з’являтися лише один раз. Основна ідея циклічного кросовера [28] полягає в обміні підциклами між двома батьківськими хромосомами, щоб створити двох

нащадків. Алгоритм CX наведений нижче.

1. Ініціалізація. Задаються дві пусті дочірні хромосоми $C_1$ та $C_2$. Обирається випадковий стартовий індекс $i$.

2. Формування циклу.

2.1. Збереження значення гена з позиції $i$ в першій батьківській хромосомі $P_1[i]$.

2.2. Знаходиться індекс $j$ такий, що $P_2[j]$ дорівнює $P_1[i]$.

2.3. Перехід до позиції $j$ в хромосомі $P_1$ і повторюються крок 2 доки не відбудиться повернення до початкового індексу $i$.

3. Копіюються гени з циклу від $P_1$ до $C_1$, а з відповідних позицій $P_2$ до $C_2$.

4. Заповнення решти генів. Для кожного індексу, який не був включений в цикл, копіюється ген з $P_2$ до с, а з $P_1$ до $C_2$.

Після завершення цього процесу, $C_1$ та $C_2$ будуть містити гени від обох батьків, але таким чином, що вони дотримуються порядку та позиції генів в батьківських хромосомах. Цей метод добре підходить для задач перестановок, таких як задача комівояжера, де важливо зберегти відносний порядок елементів.

Мутація виконується шляхом реверсування підпослідовності генів в хромосомі. Два гени для мутації вибираються випадково. Алгоритм мутації реверсом наведено нижче.

1. Вхідні дані: хромосома $X$ довжиною $n$.

2. Випадковим чином виберіть два різних індекси $i$ та $j$ такі, що $0 \le i$, $j < n$, $i < j$. Ці індекси визначають підпослідовність генів в хромосомі $X$, які будуть реверсовані.

3. Реверсування підпослідовності хромосоми $X$ між індексами $i$ та $j$, шляхом обміну місцями генів на позиціях $i$ та $j$, потім $i+1$ та $j-1$, і так далі, до середини підпослідовності.

Після завершення цього процесу, хромосома $X`$ буде містити мутовану версію оригінальної хромосоми $X$. Мутація шляхом реверсування підпослідовності генів може допомогти у введенні нових генетичних варіацій в популяцію, сприяючи пошуку оптимального рішення задачі.

Якщо протягом певної кількості поколінь найкраще рішення не покращується, алгоритм зупиняється. Також важливо зауважити, що при генерації початкового покоління, схрещуванні та мутації перевіряється відповідність маршруту обмеженням на вантажопідйомність комівояжера.

Розроблюваний генетичний алгоритм має наступні параметри:

– розмір популяції (int populationSize) – визначає кількість хромосом або рішень у популяції на кожному етапі алгоритму. Більший розмір популяції може забезпечити більший різноманітність рішень, але може збільшити час обчислення;

– максимальна кількість поколінь (int numGenerations) – встановлює ліміт на кількість поколінь, через які пройде алгоритм. Це захищає алгоритм від нескінченного виконання, якщо він не досягне потрібної умови зупинки;

– кількість елітних індивідів (int elitismCount) – визначає кількість найкращих рішень з поточної популяції, які будуть автоматично передані до наступного покоління без змін;

– максимальна кількість поколінь без покращення (int maxStagnation) – встановлює ліміт на кількість послідовних поколінь без жодного покращення рішення. Якщо цей ліміт досягнуто, алгоритм зупиняється, вважаючи, що оптимальне рішення досягнуто;

– номер початкового міста (int startCity) – визначає з якого міста починається маршрут.

– кількість мандрівників (int numTraveler) – вказує на кількість комівояжерів;

– ймовірність мутації (double mutationRate) – вказує ймовірність того, що конкретна хромосома буде піддана мутації. Цей параметр контролює рівень генетичної різноманітності в популяції;

– розмір турніру (int tournamentSize) – це число індивідів, які будуть випадково вибрані з популяції для участі в турнірі;

– вантажопідйомність комівояжера (int loadCapacity) –вказує на максимальний вантаж, який може бути перевезений одним комівояжером;

– обмеження по вазі для кожного міста (int[] restriction) – це масив, який вказує на кількість вантажу, який потрібно забрати або доставити до кожного конкретного міста. Коли маршрут включає певне місто, його обмеження (або вага) додається до загального вантажу підмаршруту.

Отже, за допомогою генетичних алгоритмів можна вирішувати будь-яку версію TSP, незалежно від її особливостей. ГА досліджують багато рішень одночасно завдяки популяційному підходу, що може допомогти у виявленні глобального оптимуму. Однак генетичні алгоритми не гарантують знаходження оптимального рішення. Вони вимагають налаштування ряду параметрів, що може значно вплинути на їхню ефективність.

## 2.7 Висновки до другого розділу

Розглядаючи ключові аспекти внутрішньозаводської логістики, зокрема, проблему оптимізації маршрутів, можна зробити висновок, що доцільним є її формалізація у вигляді задачі декількох комівояжерів з обмеженнями (MTSPC).

Проведено огляд найбільш перспективних методів вирішення ЗК, включаючи метод гілок та меж, метод динамічного програмування, і генетичний алгоритм, мають свої переваги та недоліки, а саме:

– метод динамічного програмування (алгоритм Хелда-Карпа) гарантує знаходження оптимального рішення задачі розбиваючи велику проблему на менші підзадачі, систематично розв’язуючи їх. До недоліків можна віднести експоненціальну складність, що робить його непрактичним для великих задач. Також він вимагає значної кількості пам’яті для зберігання проміжних результатів та його важко адаптувати для задач з обмеженнями;

– до переваг методу гілок та меж (метод Літтла) можна віднести знаходження точного оптимального рішення задачі та придатність для широкого спектру задач, включаючи різні види обмежень та специфікацій. Недоліком цього методу є різке падіння ефективності зі зростанням розмірності здачі;

– генетичний алгоритм може одночасно досліджувати багато рішень, що підвищує шанси на знаходження оптимального рішення. Він здатен обробляти різні типи обмежень і специфікацій. Проте, ГА не гарантує знаходження абсолютно найкращого рішення, та потребує ретельного налаштування параметрів.

Отже, значно знизити витрати та збільшити продуктивність внутрішньозаводської логістики, а також забезпечити більш точне планування та керування розподілу ресурсів на підприємствах можна за рахунок розробки підсистеми підтримки прийняття рішень, у якій буде реалізовано описані вище методи. Також, необхідно провести експерименти для виявлення закономірностей у роботі алгоритмів пошуку оптимальних маршрутів.

# 3 РОЗРОБКА ПРОГРАМНОГО ЗАСОБУ ТА ЕКСПЕРИМЕНТИ

Розробка програмного засобу є складним та багатоетапним процесом у сфері інформаційних технологій. Вона передбачає створення програмного продукту з визначеними функціональними можливостями та відповідно до конкретних вимог і завдань. Аналіз результатів розробки програмного засобу – це важлива частина процесу, яка допомагає визначити якість та ефективність створеного продукту.

Перший етап розробки – це аналіз вимог, коли визначаються потреби користувачів та функціональні вимоги до програмного засобу.

На другому етапі відбувається розробка самого програмного засобу. Програмісти пишуть код, тестують його та виправляють помилки.

Після завершення розробки проводиться тестування програмного засобу. Тестування допомагає виявити помилки та дефекти, які потребують виправлення.

Аналіз результатів розробки включає в себе оцінку якості програмного продукту, його продуктивності та відповідності вимогам.

## 3.1 Розробка вимог до програмного додатку

В якості підходу до розробки був обраний об’єктно-орієнтований підхід. Об’єктно-орієнтований підхід (ООП) є важливою парадигмою в програмуванні та розробці програмного забезпечення. Він базується на концепціях об’єктів та класів, що дозволяє структурувати програмний код у більш логічний та організований спосіб. ООП розглядає програму як сукупність взаємодіючих об’єктів, кожен з яких має визначені характеристики і може виконувати певні дії [29].

Основні принципи об’єктно-орієнтованого підходу включають:

– інкапсуляція. Цей принцип передбачає об’єднання даних та методів, що їх обробляють, в один об’єкт. Інкапсуляція дозволяє приховувати

деталі реалізації внутрішнього стану об’єкта та надавати доступ до нього лише через визначені інтерфейси;

– спадкування. Спадкування дозволяє створювати нові класи на основі існуючих, успадковуючи їх властивості та методи. Це спрощує повторне використання коду і дозволяє створювати ієрархії класів;

– поліморфізм. Поліморфізм дозволяє використовувати об’єкти різних класів з однаковим інтерфейсом способу виклику методів. Це спрощує роботу з об’єктами, оскільки ви можете взаємодіяти з ними уніфікованим способом;

– абстракція. Абстракція дозволяє визначати загальні характеристики об’єктів та ігнорувати незначні деталі. Вона допомагає зосередитися на важливих аспектах розробки програми і спрощує її розуміння та моделювання.

ООП сприяє реорганізації та розширенню коду, полегшує співпрацю великих команд, підвищує стабільність та підтримку програмних продуктів. Цей підхід добре підходить для моделювання складних систем, включаючи програмне забезпечення, що взаємодіє з обладнанням, базами даних та іншими компонентами.

Більшість сучасних методів об’єктно-орієнтованого аналізу та проектування ґрунтується на використанні UML. Unified Modeling Language – це стандартизована візуальна мова моделювання, яка використовується у сфері програмної інженерії та інформаційних технологій для аналізу, проектування та документування систем [30].

UML надає зручний і уніфікований спосіб відображення складних системних структур та їхньої поведінки за допомогою графічних діаграм. Мова включає в себе різні види діаграм, такі як діаграми класів, діаграми послідовностей, діаграми діяльності, діаграми взаємодії та інші. Кожен тип діаграми дозволяє відобразити конкретний аспект системи та взаємодію між її елементами.

Основні цілі UML включають в себе полегшення розуміння та

спілкування між розробниками, підвищення якості та продуктивності проектування програмних систем, а також створення чіткої документації для проектів. Використання UML дозволяє знизити ризики та покращити управління проектами у сфері програмної інженерії, роблячи її важливим інструментом у процесі розробки програмного забезпечення.

### 3.1.1 Розробка діаграми прецедентів

Діаграми прецедентів є важливим інструментом в аналізі та проектуванні систем, особливо в області об’єктно-орієнтованого програмування та інженерії програмного забезпечення. Цей вид діаграм дозволяє моделювати функціональність системи з точки зору її взаємодії з різними користувачами або зовнішніми системами.

Основною ціллю діаграм прецедентів є опис функціональних вимог до системи та визначення взаємодій між акторами (користувачами або зовнішніми сутностями) і прецедентами (функціональними одиницями системи). Актори представляють ролі, які виконують користувачі або зовнішні системи, а прецеденти описують конкретні дії або функції, які система виконує для задоволення потреб акторів.

Діаграми прецедентів складаються з наступних основних елементів:

– актори: це зовнішні сутності, які взаємодіють з системою. Вони можуть бути користувачами, іншими програмами або обладнанням;

– прецеденти: це конкретні функції або дії, які виконує система для задоволення потреб акторів. Кожен прецедент описується назвою та коротким описом;

– взаємодії: стрілки, які показують взаємодію між акторами і прецедентами. Вони вказують, як актори викликають прецеденти та які дані або послуги передаються.

Діаграми прецедентів допомагають зрозуміти, як система буде використовуватися в реальному середовищі і які функції вона повинна

надавати. Вони є важливою частиною аналізу вимог та допомагають уточнити функціональні вимоги до системи перед її розробкою. Крім того, діаграми прецедентів можуть використовуватися для визначення структури та організації системи під час проектування.

На рисунку 3.1 наведена діаграма прецедентів для розроблювального додатку. Система має лише одного актора – користувача.


Рисунок 3.1 – Діаграма прецедентів розроблювального додатку

### 3.1.2 Розробка діаграми класів

Діаграми класів є одним з ключових видів діаграм у мові моделювання UML, яка використовується у сфері програмної інженерії для аналізу, проектування та документування систем. Діаграми класів відображають структуру програмної системи, її об’єкти (або класи) та взаємозв’язки між ними. Цей тип діаграм дозволяє розглядати систему з точки зору її складових частин, а також описувати основні характеристики цих частин.

Кожен клас на діаграмі класів представляє собою абстракцію певного

об’єкта чи елемента системи. Класи визначають атрибути (змінні) та методи (функції) цих об’єктів. Атрибути описують стан класу, тобто дані, які він зберігає. Методи визначають поведінку класу, тобто операції, які можна виконати над ним.

Взаємозв’язки між класами на діаграмі класів вказують на те, як об’єкти цих класів взаємодіють між собою. Діаграми класів є потужним інструментом для аналізу та проектування систем, оскільки вони дозволяють визначити структуру системи, ідентифікувати основні компоненти та їх взаємодію, а також створити базу для подальшої розробки програмного забезпечення.

На рисунку 3.2 наведена діаграма класів розроблювального додатку.

Можна виділити такі основні класи:

– клас TspApplication слугує для запуску веб-додатку, і він є точкою входу для виконання програми. Він ініціалізує всі необхідні компоненти та налаштовує середовище для виконання додатку;

– клас MainPaigeController відповідає за обробку запитів, що надходять до веб-додатку, та відсилає відповіді на сторінки додатку. Він взаємодіє з клієнтами та передає їхні запити класу MainPaigeService для подальшої обробки;

– клас MainPaigeService містить логіку обробки запитів від MainPaigeController. Він виконує необхідні операції та обчислення для обробки запитів користувачів та підготовки відповідей;

– клас TspGaRestrict реалізує генетичний алгоритм для розв’язання задачі MTSP;

– клас TspBnbRestrict реалізує метод гілок та меж для розв’язання задачі MTSP;

– клас RestrictionChecker виконує перевірку маршруту на відповідність обмеженням задачі MTSP. Він допомагає визначити, чи виконані всі необхідні обмеження задачі;

– клас City представляє собою структуру даних для зберігання інформації про пункти маршруту. Кожен об’єкт цього класу містить

інформацію про координати пункту та його індекс;

– клас CityFromFile відповідає за зчитування задачі з файлу. Він дозволяє завантажити дані задачі з зовнішнього джерела;

– клас CostMatrixFromPoints слугує для перетворення набору пунктів в матрицю відстаней. Він обчислює відстані між всіма парами пунктів для подальшого використання в алгоритмах;

– клас CostMatrixExtender розширює матрицю відстаней для зведення MTSP задачі до TSP.

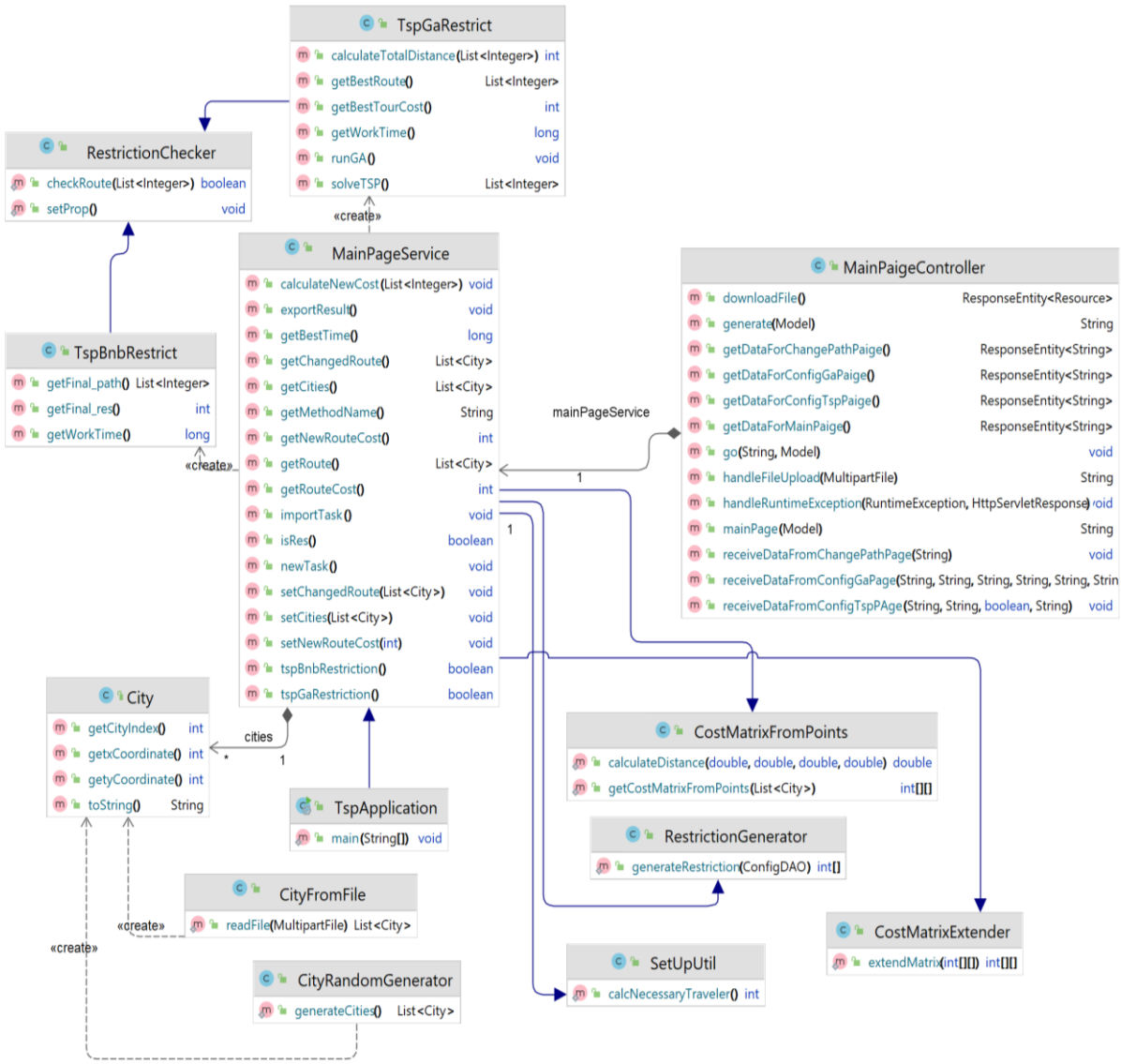


Рисунок 3.2 – Діаграма класів розроблювального додатку

### 3.1.3 Вибір технологій програмування

Для розробки додатку були вибрані наступні технології:

– мова програмування Java для бекенду сайту: Java є мовою програмування з високою продуктивністю та надійністю. Вона дозволяє створювати ефективні серверні додатки, що важливо для оптимальної обробки запитів та роботи з базою даних [31]. Spring Boot, як фреймворк для розробки на Java, надає зручний спосіб створення масштабованих та легко підтримуваних веб-додатків. Крім того, Java має велику спільноту розробників, що сприяє доступності підтримки та багатофункціональних бібліотек. Ця комбінація ресурсів та інструментів дозволяє розробникам швидко створювати надійні та потужні веб-додатки, які можуть легко масштабуватися та підтримуватися протягом тривалого часу;

– мова програмування JavaScript (JS), технологія FreeMarker, HTML та CSS для фронтенду: Використання JS для фронтенду дозволяє створювати динамічні та інтерактивні інтерфейси, що забезпечують кращий досвід користувача [32]. Технологія FreeMarker дозволяє розширювати можливості виводу даних на сторінках та створювати шаблони для їхнього відображення. Використання HTML та CSS дозволяє створювати структуру та стилізацію веб-сторінок з легкістю. Комбінація цих технологій забезпечує зручний та привабливий інтерфейс для користувачів.

Такий вибір технологій дозволив створити додаток, який поєднує високу продуктивність та надійність на бекенді з інтерактивним та привабливим фронтендом для забезпечення якісного користувацького досвіду.

## 3.2 Опис основного функціоналу розробленого програмного додатку

У рамках дослідження було розроблено програмний засіб у вигляді веб-додатку, який реалізує функції, які наведені нижче.

Вигляд інтерфейсу веб-додатку наведений на рисунку 3.3.

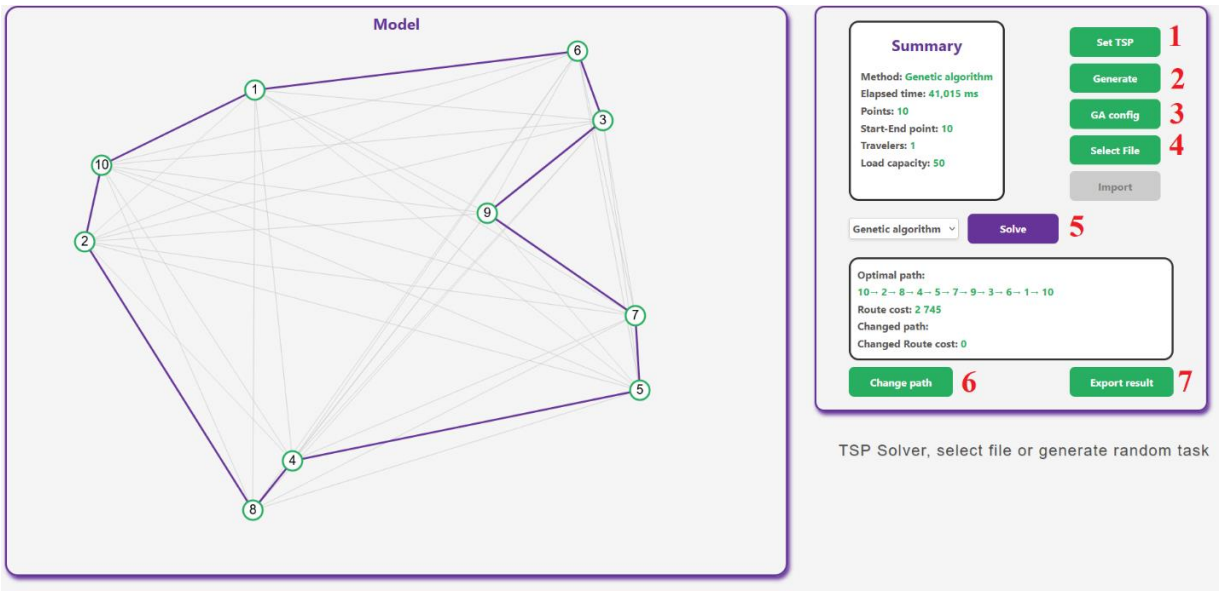


Рисунок 3.3 – Головна сторінка веб додатку

Функція постановки задачі MTSP. Вона включає в себе налаштування параметрів (рисунок 3.3, кнопка 1) для випадкової генерації задачі (рисунок 3.3, кнопка 2) та зчитування задачі з файлу (рисунок 3.3, кнопка 4). Вікно налаштування задачі наведено на рисунку 3.4. Приклад вхідного файлу наведено на рисунку 3.5.

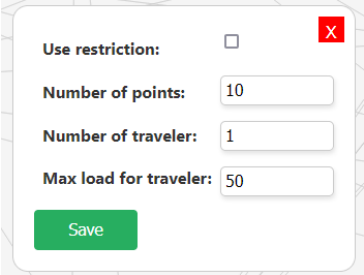


Рисунок 3.4 – Вікно налаштування параметрів задачі

Функція побудови оптимального маршруту (рисунок 3.3, кнопка 5) включає в себе вибір методу розв’язання (генетичний алгоритм чи метод гілок та меж), налаштування параметрів ГА (рисунок 3.3, кнопка 3). Вікно налаштувань ГА наведено на рисунку 3.6. Приклад відображення оптимального маршруту для 50 пунктів призначення наведений на рисунку 3.7.

```
{xCoordinate=517, yCoordinate=799, cityIndex=1};
{xCoordinate=643, yCoordinate=596, cityIndex=2};
{xCoordinate=270, yCoordinate=169, cityIndex=3};
{xCoordinate=202, yCoordinate=171, cityIndex=4};
{xCoordinate=798, yCoordinate=458, cityIndex=5};
{xCoordinate=172, yCoordinate=798, cityIndex=6};
{xCoordinate=557, yCoordinate=792, cityIndex=7};
{xCoordinate=295, yCoordinate=768, cityIndex=8};
{xCoordinate=516, yCoordinate=393, cityIndex=9};
{xCoordinate=448, yCoordinate=505, cityIndex=10};
{xCoordinate=744, yCoordinate=672, cityIndex=11};
{xCoordinate=581, yCoordinate=179, cityIndex=12};
{xCoordinate=496, yCoordinate=160, cityIndex=13};
{xCoordinate=362, yCoordinate=733, cityIndex=14};
{xCoordinate=385, yCoordinate=247, cityIndex=15};
RestrictionsArray=0,15,6,7,14,14,14,10,9,5,5,11,15,11,13
```

Рисунок 3.5 – Приклад файлу з описом задачі

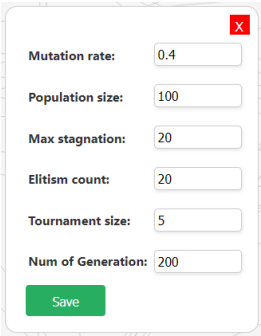


Рисунок 3.6 – Вікно налаштувань параметрів ГА

Після знаходження оптимального маршруту інформація про рішення виводиться на екран, приклад наведено на рисунку 3.8 та стає доступною функція зміни маршруту (рисунок 3.3, кнопка 6). Вікно зміни маршруту наведено на рисунку 3.9.

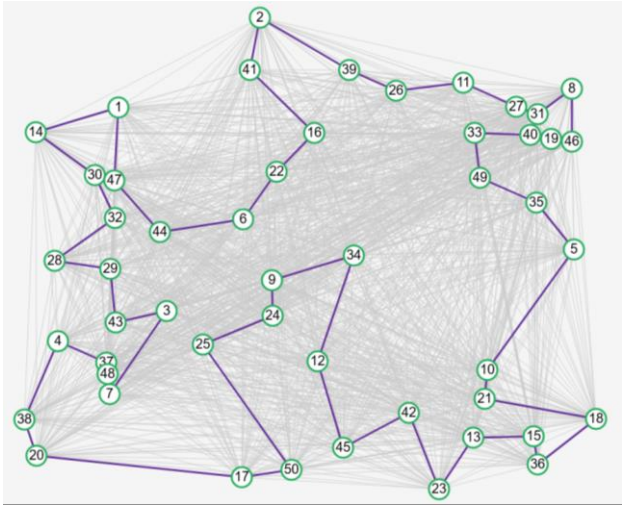

Рисунок 3.7 – Відображення рішення задачі TSP

Функція експорту результатів у вигляді txt файлу (рисунок 3.3, кнопка 7). Вона надає користувачам можливість зберігати результати своєї роботи у форматі текстового файлу, що дозволяє подальшу обробку та аналіз даних за допомогою зовнішніх інструментів або їх збереження для подальшого використання. Приклад звіту наведений на рисунку 3.10.

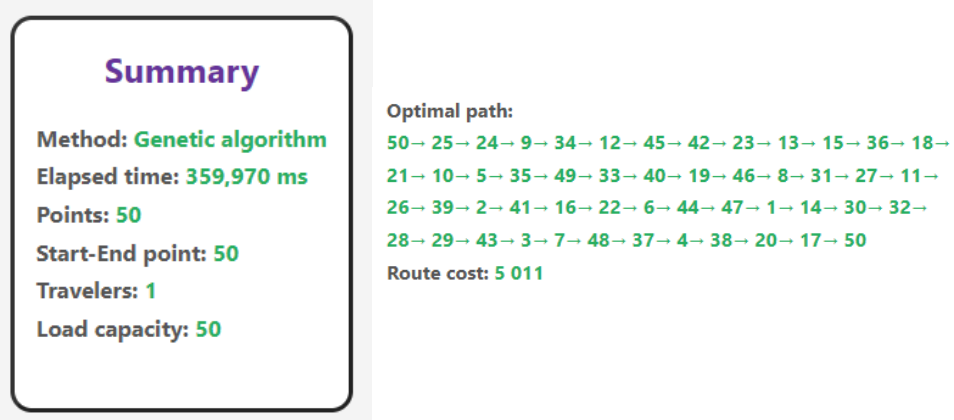


Рисунок 3.8 – Інформація про маршрут

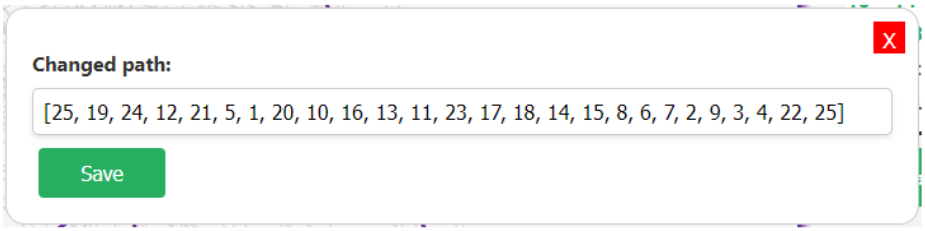


Рисунок 3.9 – Вікно зміни маршруту

```
Hello
Num of points: 50
Num of traveler: 1
Optimal route:[50, 36, 18, 42, 45, 26, 21, 35, 32, 39, 28,
20, 48, 25, 29, 13, 44, 15, 33, 12, 31, 6, 34, 4, 9, 24,
46, 10, 40, 49, 3, 11, 5, 23, 43, 38, 1, 2, 41, 17, 30,
47, 14, 16, 8, 19, 37, 27, 22, 7, 50]
Method: Genetic algorithm
MutationRate: 0.4
NumGenerations: 200
MaxStagnation: 20
PopulationSize: 100
TournamentSize: 5
ElitismCount: 20
Elapsed time: 372,193 ms

Bye bye
```

Рисунок 3.10 – Приклад звіту

## 3.3 Експерименти та аналіз результатів

Проведемо серію експериментів для встановлення закономірностей роботи реалізованих у застосунку алгоритмів та надання рекомендацій щодо їх використання. Ці експерименти дозволять краще розуміти ефективність цих алгоритмів у конкретних умовах, а також визначити оптимальні налаштування для досягнення найкращих результатів.

В якості середи виконання експериментів використовувався ПК з операційною системою Microsoft Windows 10 LTSB, та такими технічними характеристиками:

– RAM 12.00 GB;
– процесор Intel Core i5-8250U, 2,5 ГГц;
– графічний адаптер Nvidia GeForce GTX 1050.

### 3.3.1 Порівняння генетичного алгоритму з методом гілок та меж

Проведемо ряд експериментів для заміру часу роботи алгоритмів та відносної похибки розв’язання генетичним алгоритмом при різних сценаріях використання. Порівнюючи ці два методи, слід зазначити, що:

– генетичний алгоритм є ймовірнісним методом, який може знайти глобальний оптимум в складних задачах, але не гарантує цього. При використанні ГА важливо проводити дослідження налаштовування параметрів, щоб збільшити ймовірність знаходження глобального оптимуму в складних задачах.;
– метод гілок та меж є точним методом, який завжди знаходить глобальний оптимум, але може бути обмеженим у використанні для складних задач з великою кількістю варіантів.

Вибір між цими двома методами залежить від конкретної задачі та обмежень, а також від бажаного балансу між точністю та обчислювальною складністю.

Експеримент 1: порівняння часу роботи і відносної похибки генетичного алгоритму. Матриця відстаней генерується випадково, задача розв'язується без обмежень для одного комівояжера. Результати зібрані для 50 прогонів, та зведені у таблиці 3.1.

Таблиця 3.1 – Результати першого експерименту

| Кількість вершин | BnB | GA | Відносна похибка розв'язання, % |
|---|---|---|---|
| | Час роботи, мс | Час роботи, мс | |
| 5 | 0,019 | 2,8671 | 0 |
| 6 | 0,029 | 3,176 | 0 |
| 7 | 0,074 | 3,269 | 0 |
| 8 | 0,247 | 4,149 | 0,039 |
| 9 | 0,847 | 4,504 | 0,098 |
| 10 | 2,475 | 5,281 | 0,11275 |
| 11 | 9,003 | 6,128 | 0,255 |
| 12 | 33,128 | 6,467 | 0,3227 |
| 13 | 68,954 | 6,939 | 0,555 |
| 14 | 313,666 | 7,883 | 0,87 |
| 15 | 789,591 | 9,023 | 1,09 |
| 16 | 2282,626 | 9,749 | 1,21 |
| 17 | 3606,225 | 11,146 | 1,382 |
| 18 | 7443,031 | 12,337 | 1,459 |
| 19 | 14856,389 | 14,831 | 1,681 |
| 20 | 63212,131 | 16,953 | 1,98 |

На рисунку 3.11 наведено графік залежності часу роботи алгоритмів від розмірності задачі (ось ординат представлена у вигляді логарифмічної шкали). Можна зробити висновок що алгоритм BnB показує кращі результати в плані швидкості виконання при розмірності задачі менше 11. Експоненціальна залежність часу виконання від кількості вершин для алгоритму BnB свідчить про значну чутливість обчислювального часу до збільшення розмірності задачі. Для великих розмірностей задачі зростання часу обчислень алгоритму BnB стає дуже швидким і неефективним, тоді як алгоритм GA, хоча і потребує більше часу на початкових етапах, залишається ефективним для обчислень навіть при великих значеннях $N$.

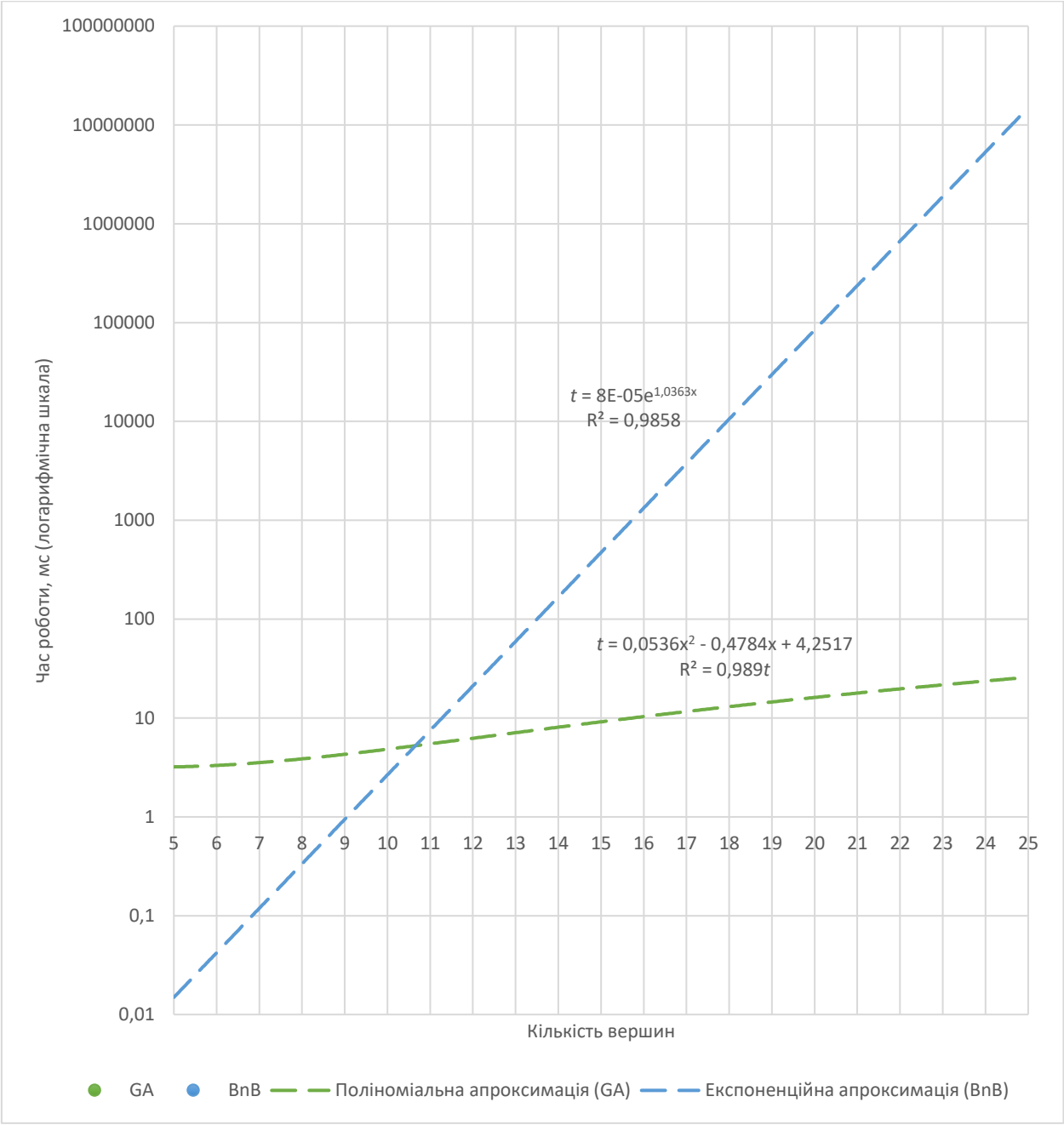


Рисунок 3.11 – Залежність часу роботи алгоритмів від розмірності задачі (один комівояжер)

На рисунку 3.12 наведено графік залежності відносної похибки розв’язання генетичного алгоритму від розмірності задачі. Можна зробити висновок, що відносна похибка генетичного алгоритму зростає з підвищенням складності задачі, що свідчить про зменшення точності алгоритму при вирішенні більш складних задач, проте, для задач з розмірністю до 25, похибка розв’язання не перевищую 4%, що є цілком прийнятним.

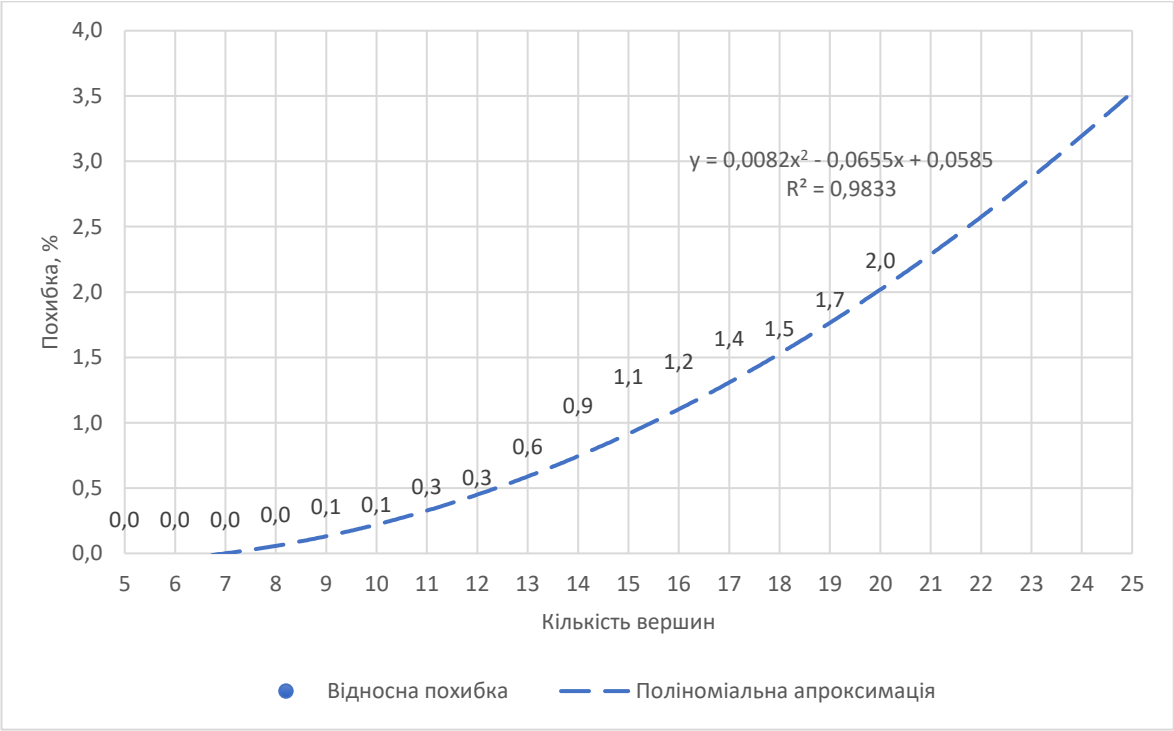


Рисунок 3.12 – Залежність відносної похибки генетичного алгоритму від розмірності задачі (один комівояжер)

Експеримент 2: порівняння часу роботи та відносної похибки алгоритмів є важливою частиною дослідження при розв'язанні задачі декількох комівояжерів. У цьому дослідженні, матриця відстаней генерується випадково, і задача розв'язується без обмежень, що дозволяє оцінити продуктивність алгоритмів в різних умовах.

Для забезпечення достовірних результатів, дослідження включає 20 прогонів. Отримані дані з цих прогонів зведені в таблиці 3.2, яка відображає результати для кожного алгоритму, включаючи середнє значення часу виконання ($t$) та відносної похибки ($\delta$).

На рисунку 3.13 представлено графік залежності часу роботи алгоритмів від розмірності задачі та кількості комівояжерів. Можна зробити висновок, що для кожного алгоритму, збільшення кількості комівояжерів $m$ збільшує час виконання, що є очікуваним, оскільки більша кількість комівояжерів веде до більшої розмірності задачі, згідно (2.12). Генетичний алгоритм показує більшу ефективність зі збільшенням $m$ ніж метод гілок та меж.

Таблиця 3.2 – Результати другого експерименту

| $N$ | $m$=2 | | | $m$=3 | | | $m$=4 | | |
|---|---|---|---|---|---|---|---|---|---|
| | BnB | GA | $\delta$, % | BnB | GA | $\delta$, % | BnB | GA | $\delta$, % |
| | $t$, мс | $t$, мс | | $t$, мс | $t$, мс | | $t$, мс | $t$, мс | |
| 5 | 0,076 | 2,925 | 0 | 0,208 | 2,98 | 0 | 0,894 | 3,714 | 0 |
| 6 | 0,096 | 3,057 | 0 | 0,499 | 3,153 | 0,01 | 3,323 | 3,932 | 0,038 |
| 7 | 0,267 | 3,84 | 0,06 | 1,563 | 4,805 | 0,02 | 16,517 | 4,66 | 0,045 |
| 8 | 0,75 | 4,75 | 0,09 | 5,04 | 4,98 | 0,27 | 64,12 | 4,856 | 0,49 |
| 9 | 3,532 | 5,616 | 0,27 | 20,98 | 5,181 | 0,34 | 308,08 | 4,985 | 0,925 |
| 10 | 10,55 | 5,268 | 0,29 | 63,86 | 5,716 | 0,58 | 777,51 | 6,016 | 0,72 |
| 11 | 30,16 | 6,562 | 0,38 | 240,6 | 6,614 | 0,75 | 1998,1 | 6,857 | 1,42 |
| 12 | 62,12 | 7,414 | 0,42 | 329,8 | 7,868 | 0,92 | 3244,0 | 8,459 | 1,64 |
| 13 | 87,73 | 8,573 | 0,72 | 693,5 | 8,912 | 1,44 | 4974,2 | 8,966 | 1,96 |
| 14 | 251,8 | 9,279 | 0,98 | 2743 | 9,514 | 1,92 | 7408,9 | 9,644 | 2,38 |
| 15 | 422,2 | 9,92 | 1,18 | 3774 | 10,57 | 2,34 | 22363 | 12,14 | 2,98 |
| 16 | 854,2 | 11,4 | 1,24 | 7543 | 11,85 | 2,85 | 99095 | 13,05 | 3,52 |
| 17 | 2130 | 12,38 | 1,48 | 10524 | 12,94 | 3,12 | - | 14,88 | - |
| 18 | 7213 | 13,77 | 1,87 | 21445 | 14,88 | 3,67 | - | 15,42 | - |
| 19 | 19143 | 15,24 | 2,14 | - | 16,60 | - | - | 18,11 | - |
| 20 | - | 16,24 | - | - | 18,94 | - | - | 21,40 | - |

Експеримент 3: порівняння часу роботи і відносної похибки алгоритмів при розв’язанні задачі декількох комівояжерів з обмеженнями. Матриця відстаней генерується випадково, задача розв’язується для трьох комівояжерів. Результати зібрані для 20 прогонів, та зведені у таблиці 3.3.

Таблиця 3.3 – Результати третього експерименту

| Кількість вершин | BnB | GA | Відносна похибка розв’язання, % |
|---|---|---|---|
| | Час роботи, мс | Час роботи, мс | |
| 1 | 2 | 3 | 4 |
| 5 | 1,25 | 1,106 | 0 |
| 6 | 4,327 | 2,549 | 0 |
| 7 | 5,971 | 3,729 | 0 |
| 8 | 10,057 | 4,629 | 0,21 |
| 9 | 26,162 | 5,984 | 0,489 |
| 10 | 89,355 | 6,039 | 0,612 |
| 11 | 315,6 | 7,193 | 0,854 |

Продовження таблиці 3.3

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| 12 | 673,918 | 7,692 | 1,077 |
| 13 | 2476,045 | 9,589 | 1,15 |
| 14 | 4986,962 | 9,996 | 1,268 |
| 15 | 6546,188 | 10,665 | 1,488 |
| 16 | 9058,724 | 11,793 | 1,983 |
| 17 | 23472,351 | 13,82 | 2,17 |
| 18 | 79167,54 | 16,609 | 2,42 |
| 19 | - | 17,419 | - |
| 20 | - | 18,68 | - |

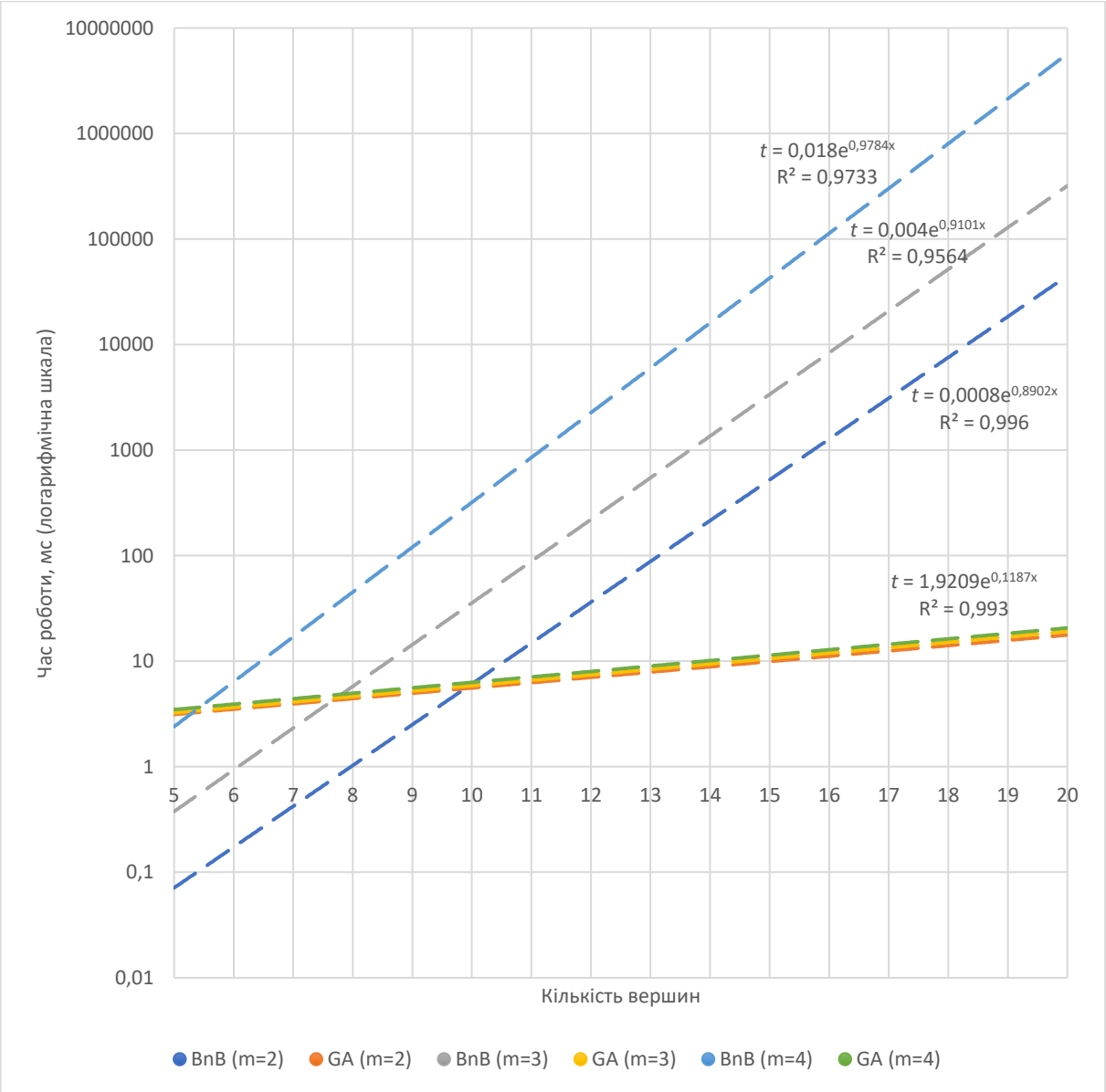


Рисунок 3.13 – Залежність часу роботи алгоритмів від розмірності задачі та кількості комівояжерів

На рисунку 3.14 наведено графік залежності часу роботи алгоритмів від розмірності задачі з обмеженнями. Можна зроби висновок, що час розв'язання для обох методів більше для задач з обмеженнями. Генетичний алгоритм виявляється більш масштабованим порівняно з алгоритмом гілок та меж, що вказує на його придатність для розв'язання задач з обмеженнями. Хоча BnB може бути кращим для менших розмірів задач через його здатність знаходити точні рішення, його ефективність знижується при збільшенні розміру задачі через експоненціальне зростання часу обчислень.

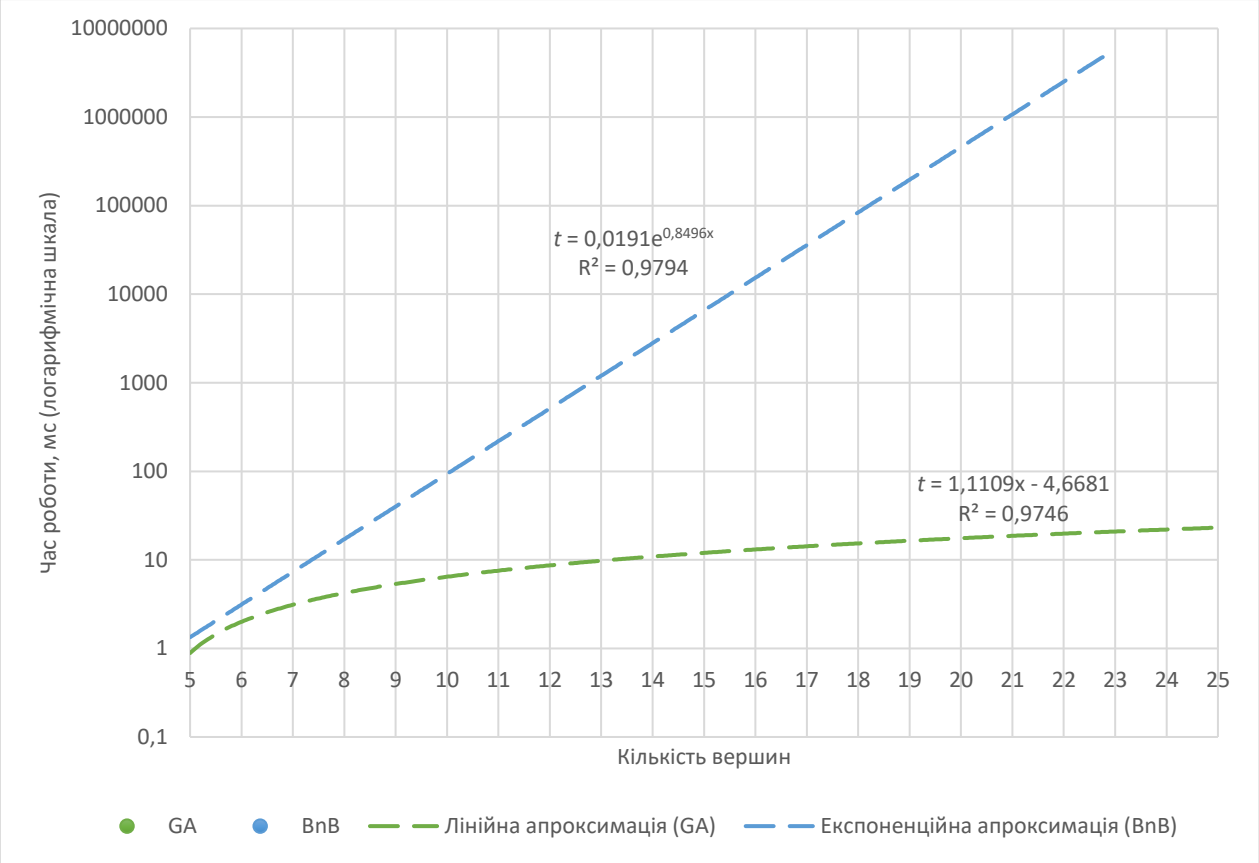


Рисунок 3.14 – Залежність часу роботи алгоритмів від розмірності задачі з обмеженнями (три комівояжера)

На рисунку 3.15 представлено графік залежності відносної похибки генетичного алгоритму від розмірності задачі з обмеженнями. Можна зробити висновок, що похибка є більшою, ніж для розв'язання задач без обмежень, проте, для задач з розмірністю до 25, похибка розв'язання не перевищую 5%.

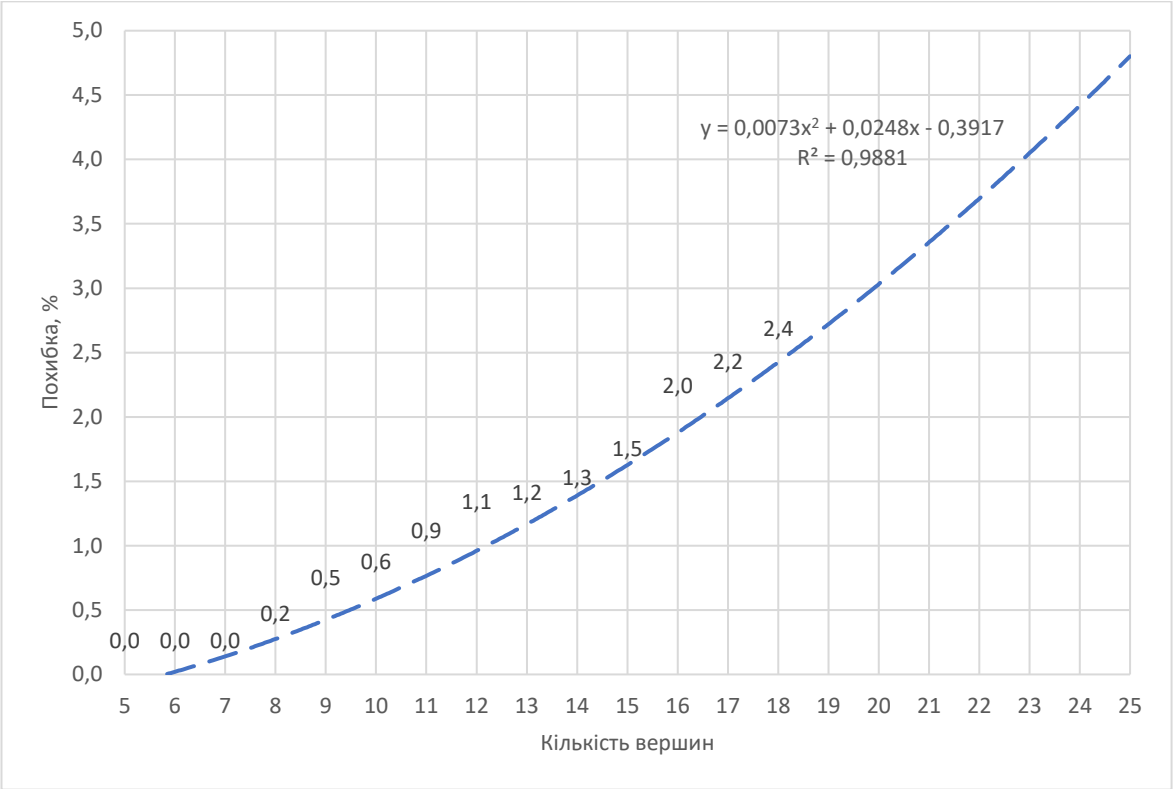


Рисунок 3.15 – Залежність відносної похибки генетичного алгоритму від розмірності задачі з обмеженнями (три комівояжера)

Експеримент 4: визначення часу роботи генетичного алгоритму для задач з великою розмірністю. Матриця відстаней генерується випадково, задача розв’язується без обмежень для одного комівояжера. Результати зібрані для 100 прогонів, та зведені у таблиці 3.4.

Таблиця 3.4 – Результати четвертого експерименту

| Кількість вершин | Час роботи, мс |
|---|---|
| 20 | 32,773 |
| 40 | 56,364 |
| 60 | 151,572 |
| 80 | 280,908 |
| 100 | 413,185 |
| 120 | 492,5 |
| 140 | 588,961 |
| 160 | 716,515 |
| 180 | 799,611 |
| 200 | 928,196 |

На рисунку 3.16 наведено графік залежності часу роботи генетичного алгоритму від розмірності задачі. Можна зробити висновок, що час роботи генетичного алгоритму зростає квадратично зі збільшенням кількості вершин, що вказує на збільшення обчислювальної складності з ростом розміру задачі

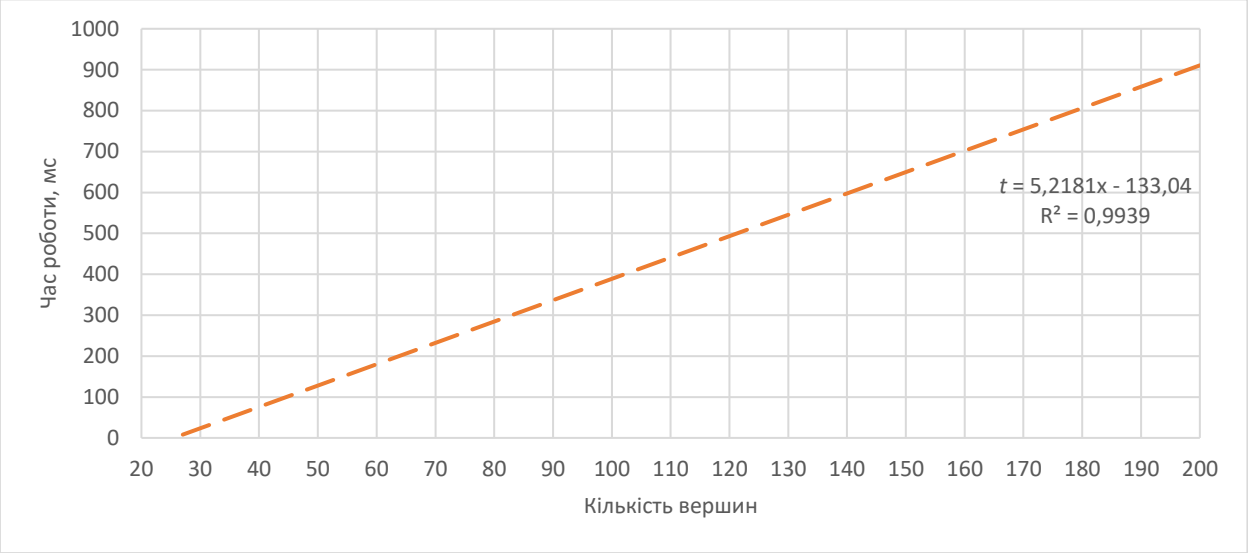


Рисунок 3.16 – Залежність часу роботи генетичного алгоритму від розмірності задачі (один комівояжер)

### 3.3.2 Вплив параметрів генетичного алгоритму на його роботу

Вплив параметрів генетичного алгоритму на його роботу визначає ефективність та якість знайдених рішень. Коректний вибір параметрів може покращити продуктивність алгоритму та забезпечити оптимальні результати. Нижче ми розглянемо основні параметри генетичного алгоритму та їх вплив на його роботу.

Розмір популяції (Population Size): цей параметр визначає кількість індивідів в популяції. Зазвичай більша популяція дозволяє збільшити різноманітність та робить алгоритм менш схильним до застрягання в локальних оптимумах.

Інтенсивність мутації (Mutation Rate): цей параметр визначає ймовірність мутації кожного гену в індивідуальних рішеннях. Мутація

допомагає в алгоритмі рухатися вздовж простору пошуку.

Кількість поколінь (Number of Generations): цей параметр визначає, скільки поколінь індивідів будуть створені перед завершенням роботи алгоритму.

Відсоток елітизму (Elitism Rate): цей параметр визначає, яку частку найкращих індивідів з поточної популяції буде беззмінно передано в наступне покоління. Цей параметр служить для збереження найкращих рішень і запобігання їх втраті в процесі еволюції.

Розмір турніру (Tournament Size): цей параметр визначає, скільки індивідів буде обрано для участі в турнірі на кожному кроці вибору батьків для наступного покоління.

Проведемо серію експериментів для встановлення залежностей впливу цих параметрів на роботи генетичного алгоритму.

Експеримент 5: дослідження впливу зміни ймовірності мутації на час роботи генетичного алгоритму та вартість маршруту. Матриця відстаней розмірністю 100 пунктів згенерована випадково, задача розв’язується з обмеженнями для трьох комівояжерів. Результати зібрані для 20 прогонів, та зведені у таблиці 3.5.

Таблиця 3.5 – Результати п’ятого експерименту

| Ймовірність мутації | Час розв’язання, мс | Найкраща довжина | Середня довжина |
|---|---|---|---|
| 0,1 | 425,871 | 8987 | 10026 |
| 0,2 | 425,626 | 8244 | 8897 |
| 0,3 | 430,65 | 7559 | 8373 |
| 0,4 | 447,877 | 7553 | 8008 |
| 0,5 | 466,344 | 7321 | 7692 |
| 0,6 | 469,247 | 7243 | 7757 |
| 0,7 | 477,941 | 7174 | 7746 |
| 0,8 | 482,978 | 7176 | 7619 |
| 0,9 | 493,919 | 7409 | 8100 |
| 1 | 498,235 | 8600 | 9987 |

На рисунку 3.17 подано графік, що відображає вплив ймовірності мутації на час виконання генетичного алгоритму. Можна зробити висновок, що зі збільшенням ймовірності мутації спостерігається збільшення часу, необхідного для виконання алгоритму. Це пояснюється тим, що зі збільшенням ймовірності мутації зростає кількість мутаційних операцій, які потрібно виконати, що впливає на тривалість виконання алгоритму.

На рисунку 3.18 представлений графік, який демонструє вплив ймовірності мутації на вартість маршруту, яка є результатом роботи генетичного алгоритму. З аналізу цього графіка можна зробити висновок, що оптимальне значення ймовірності мутації розташовується в діапазоні від 0.4 до 0.8. Однак, враховуючи час виконання алгоритму, оптимальне значення ймовірності мутації обмежується діапазоном від 0.4 до 0.5.

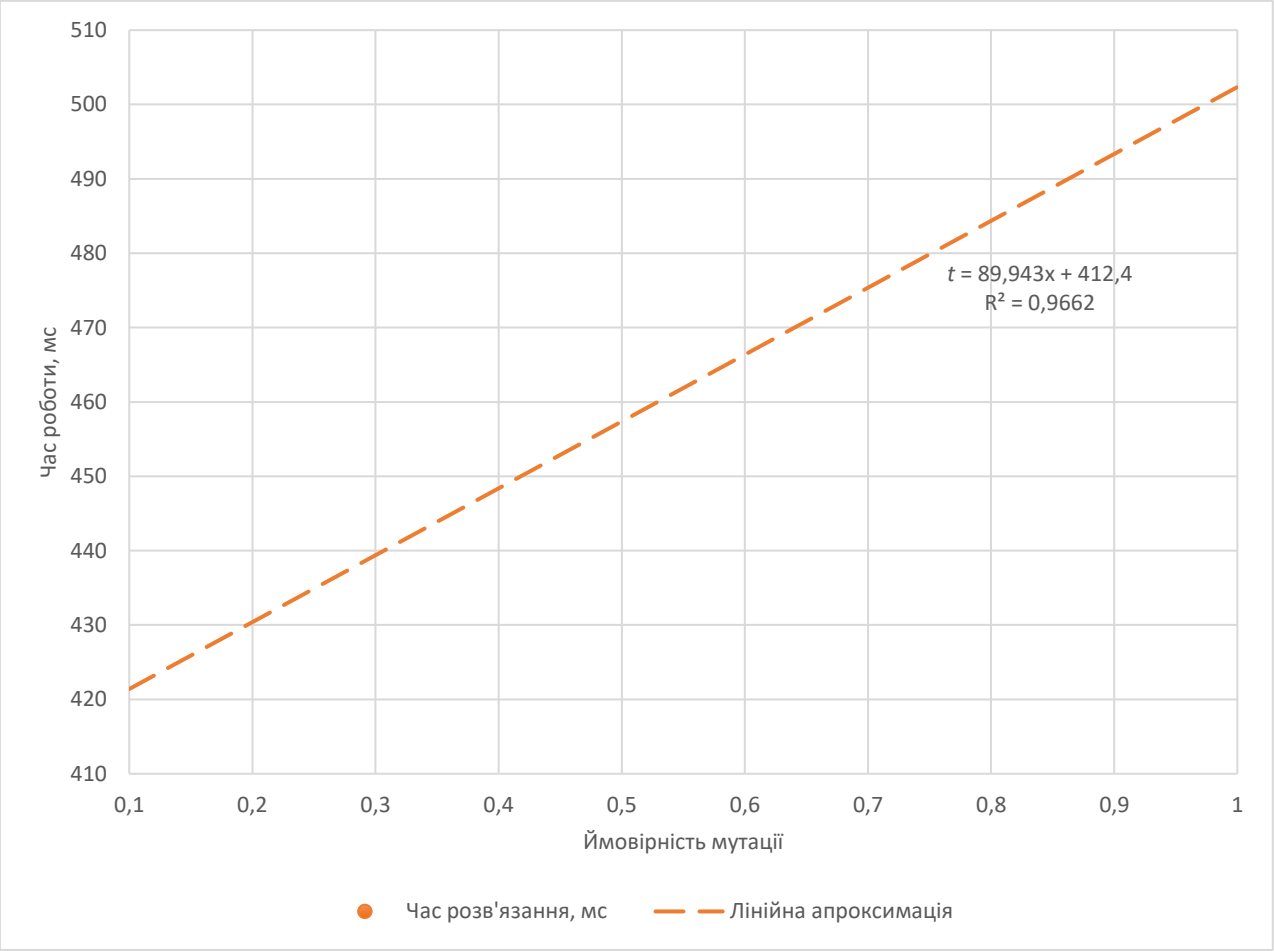


Рисунок 3.17 – Залежність часу роботи генетичного алгоритму від ймовірності мутації

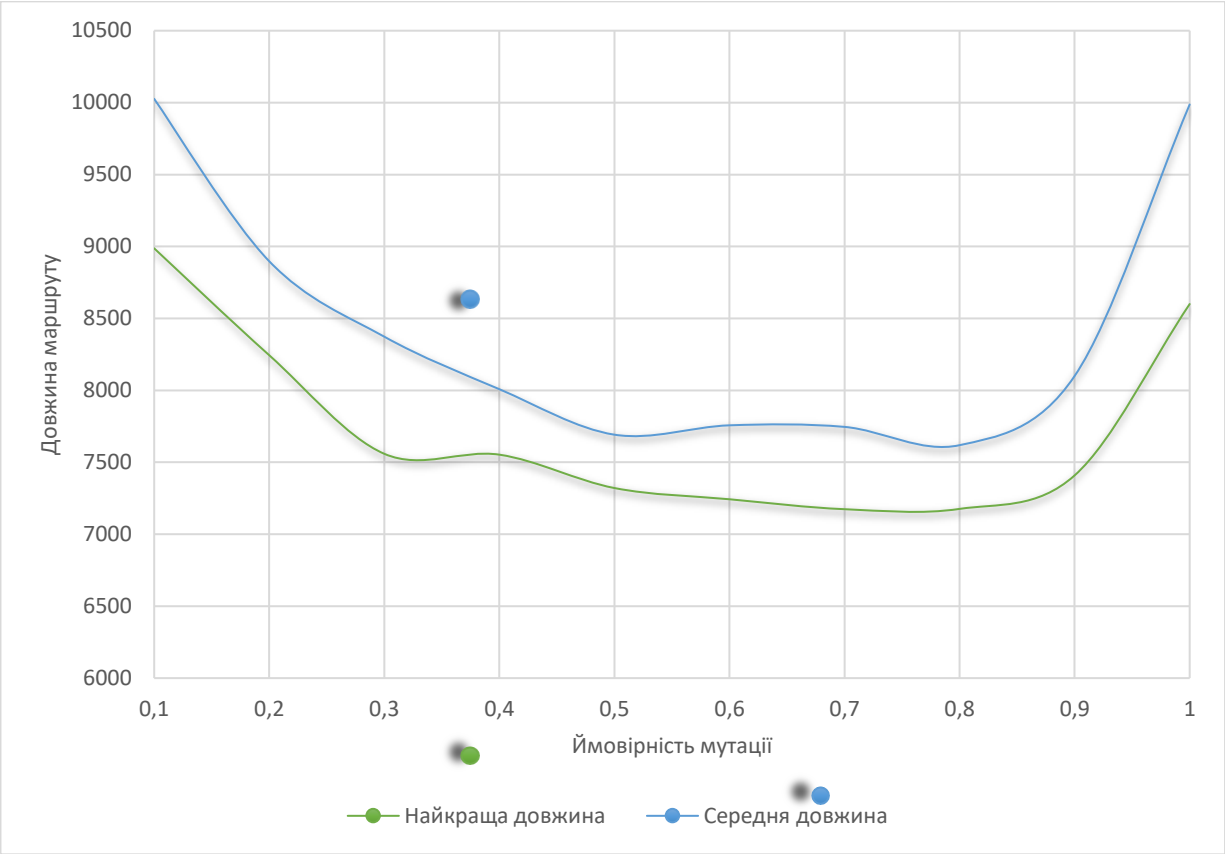


Рисунок 3.18 – Залежність вартості маршруту від ймовірності мутації

Експеримент 6: дослідження впливу зміни розміру популяції на час роботи генетичного алгоритму та вартість маршруту. Матриця відстаней розмірністю 100 пунктів згенерована випадково, задача розв’язується з обмеженнями для трьох комівояжерів. Результати зібрані для 20 прогонів, та зведені у таблиці 3.6.

Таблиця 3.6 – Результати шостого експерименту

| Розмір популяції | Час розв’язання, мс | Найкраща довжина | Середня довжина |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 50 | 283,084 | 8048 | 8592 |
| 100 | 545,958 | 7723 | 8228 |
| 150 | 853,575 | 7541 | 8062 |
| 200 | 1114,578 | 7550 | 7975 |
| 250 | 1266,636 | 7601 | 7960 |
| 300 | 1476,586 | 7540 | 7858 |
| 350 | 1820,481 | 7561 | 7814 |

Продовження таблиці 3.6

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| 400 | 1989,966 | 7509 | 7820 |
| 450 | 2307,518 | 7479 | 7809 |
| 500 | 2464,024 | 7517 | 7794 |
| 550 | 2684,729 | 7537 | 7800 |
| 600 | 2889,574 | 7481 | 7791 |
| 650 | 3165,971 | 7557 | 7806 |
| 700 | 3438,547 | 7482 | 7783 |
| 750 | 3639,471 | 7448 | 7766 |
| 800 | 3861,325 | 7558 | 7791 |
| 850 | 4002,323 | 7448 | 7770 |
| 900 | 4264,62 | 7530 | 7709 |
| 950 | 4578,751 | 7442 | 7719 |
| 1000 | 4884,161 | 7338 | 7652 |

На рисунку 3.19 представлений графік, який демонструє вплив зміни розміру популяції на час виконання генетичного алгоритму. Можна зробити висновок, що зі збільшенням розміру популяції час виконання генетичного алгоритму збільшується. Ця залежність пояснюється зростанням кількості операцій кросоверу та мутації над популяцією.

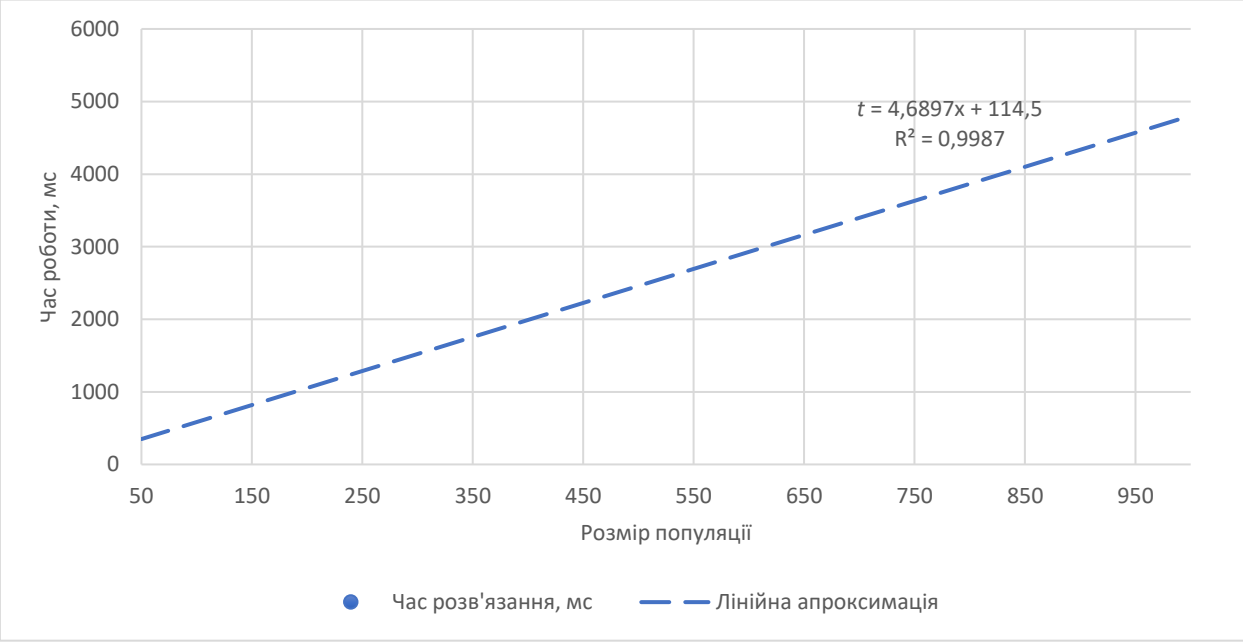


Рисунок 3.19 – Залежність часу роботи генетичного алгоритму від розміру популяції

На рисунку 3.20 показано графік, який відображає залежність вартості маршруту від розміру популяції. З цього графіку можна зробити висновок, що зі збільшенням розміру популяції вартість маршруту зменшується незначною мірою. Отже, розглядаючи обидва графіки, можна стверджувати, що оптимальний розмір популяції складає близько 200 осіб. Проте, враховуючи зростання розмірності задачі, оптимальне значення розміру популяції також збільшуватиметься.

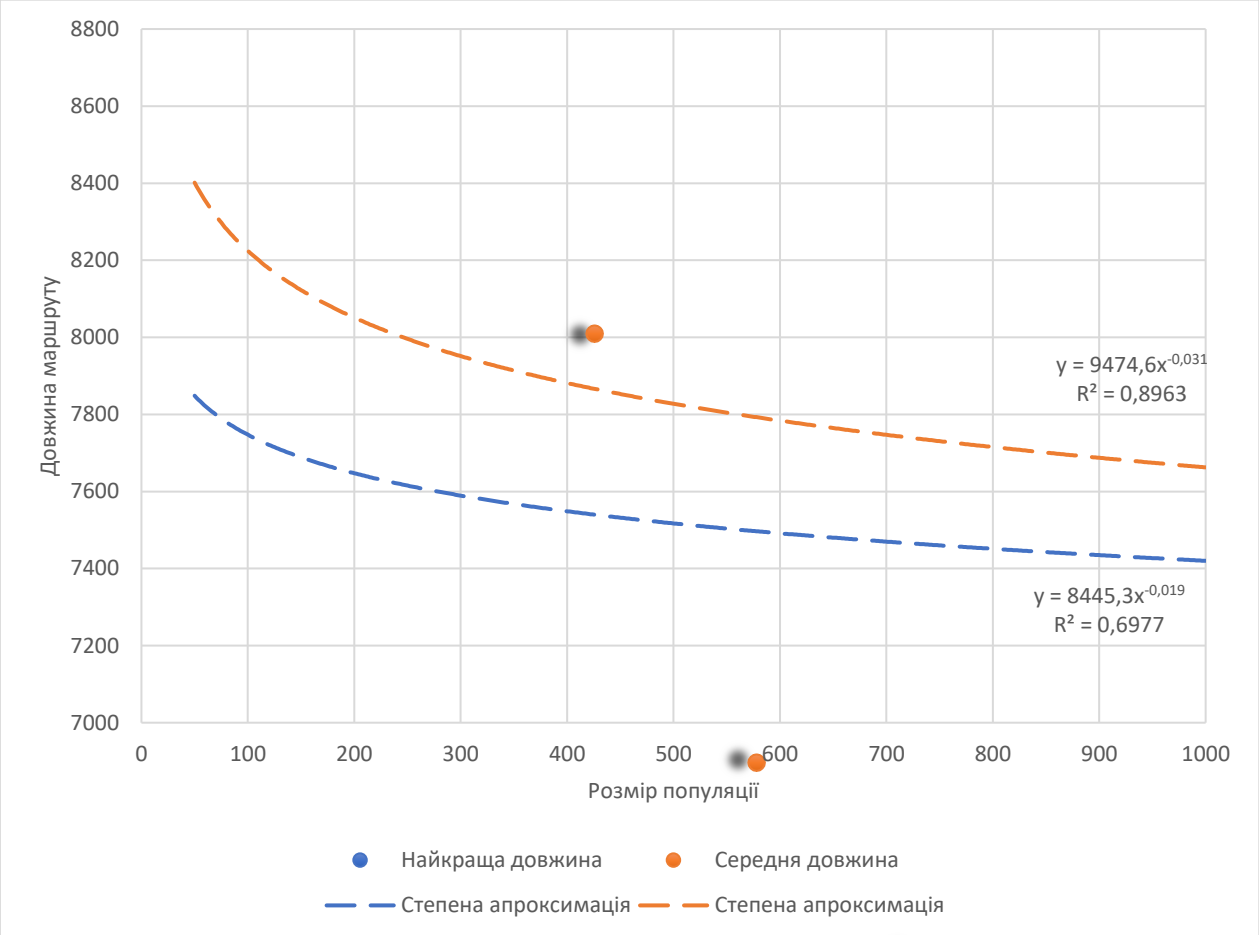


Рисунок 3.20 – Залежність вартості маршруту від розміру популяції

Експеримент 7: дослідження впливу зміни відсотку елітизму на час роботи генетичного алгоритму та вартість маршруту. Матриця відстаней розмірністю 100 пунктів згенерована випадково, задача розв’язується з обмеженнями для трьох комівояжерів. Результати зібрані для 20 прогонів, та зведені у таблиці 3.7.

Таблиця 3.7 – Результати сьомого експерименту

| Відсоток елітизму | Час розв’язання, мс | Найкраща довжина | Середня довжина |
|---|---|---|---|
| 0,1 | 1626,791 | 6986 | 7290 |
| 0,2 | 1460,831 | 7061 | 7328 |
| 0,3 | 1332,19 | 6901 | 7438 |
| 0,4 | 1228,572 | 7031 | 7437 |
| 0,5 | 1046 | 7242 | 7655 |
| 0,6 | 849,247 | 7256 | 7689 |
| 0,7 | 660,698 | 7707 | 8156 |
| 0,8 | 488,723 | 8149 | 8969 |
| 0,9 | 279,714 | 10970 | 11870 |

На рисунку 3.21 представлений графік, що відображає залежність часу виконання генетичного алгоритму від зміни відсотку елітизму. З цього графіку можна зробити висновок, що зі збільшенням відсотку елітизму час виконання генетичного алгоритму зменшується. Це можна пояснити тим, що збільшення відсотку елітизму призводить до зменшення кількості операцій кросоверу, які необхідно виконати для підтримання заданого розміру популяції.

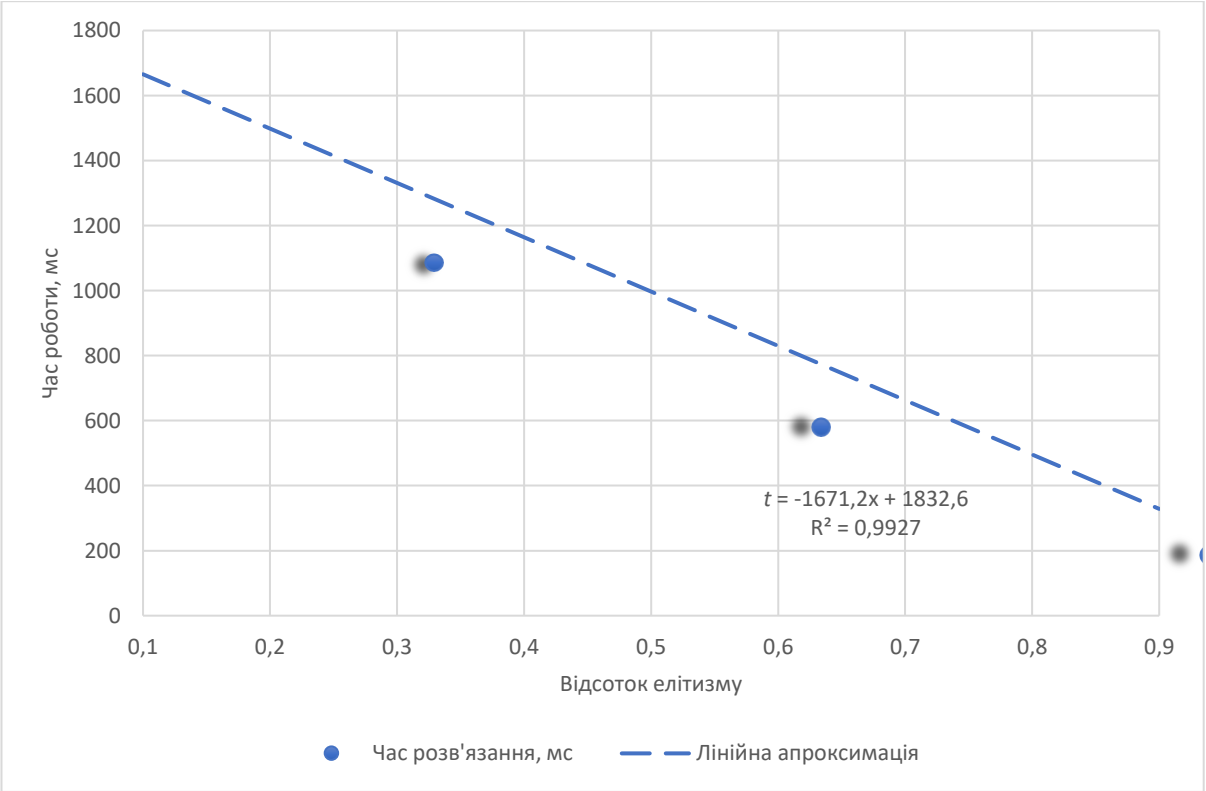


Рисунок 3.21 – Залежність часу роботи ГА від зміни відсотку елітизму

На рисунку 3.22 показаний графік, який відображає залежність вартості маршруту від зміни відсотку елітизму. З цього графіку можна зробити висновок, що вартість маршруту різко зростає при великих значеннях відсотку елітизму. Отже, розглядаючи обидва графіки, можна зазначити, що оптимальний відсоток елітизму становить приблизно 0.65.

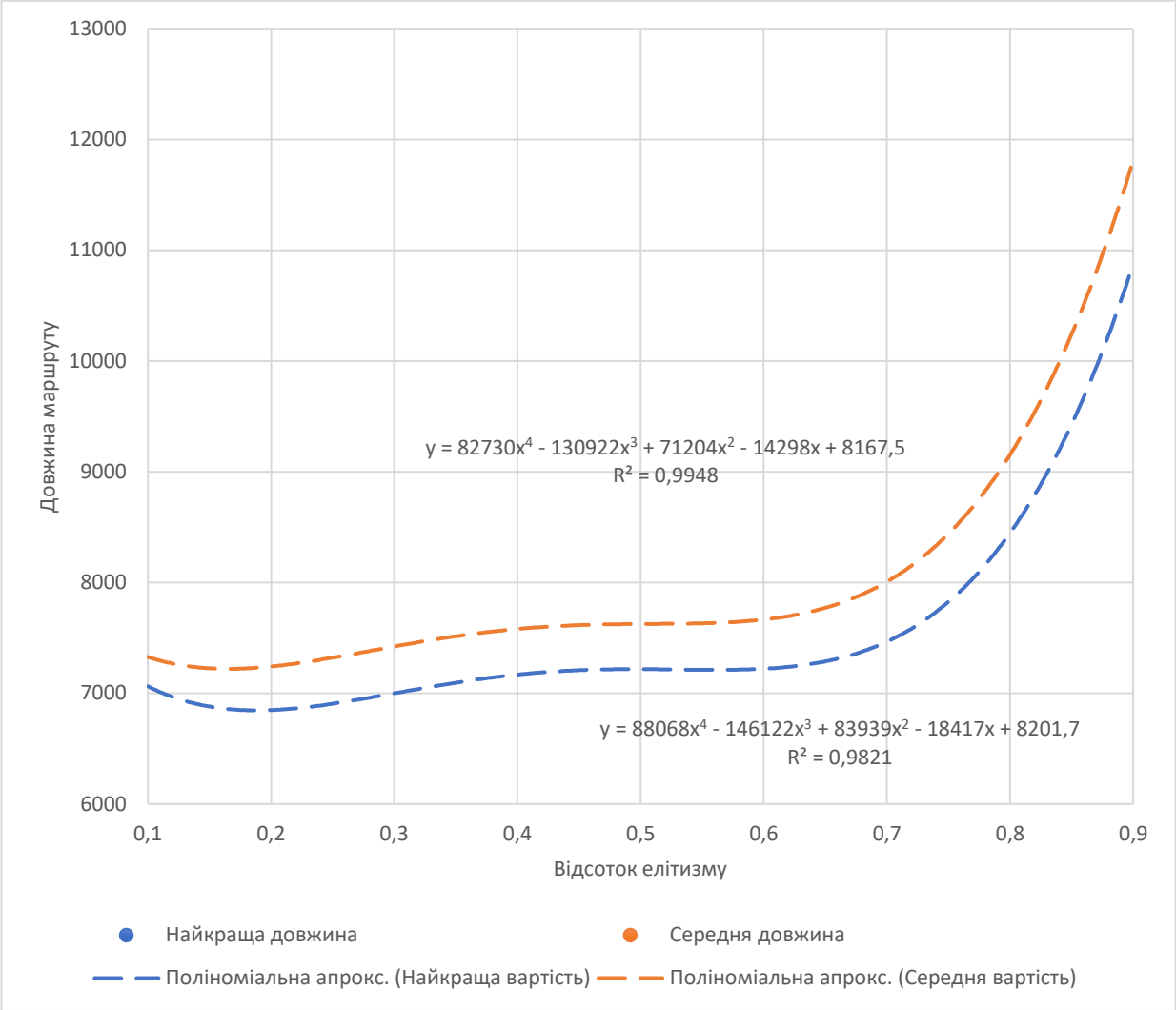


Рисунок 3.22 – Залежність вартості маршруту від зміни відсотку елітизму

Експеримент 8: Дослідження впливу зміни розміру турніру на час роботи генетичного алгоритму та вартість маршруту. Матриця відстаней розмірністю 100 пунктів згенерована випадково, задача розв’язується з обмеженнями для трьох комівояжерів. Результати зібрані для 20 прогонів, та зведені у таблиці 3.8.

Таблиця 3.5 – Результати восьмого експерименту

| Розмір турніру | Час розв’язання, мс | Найкраща довжина | Середня довжина |
|---|---|---|---|
| 5 | 1488,87 | 7333 | 7753 |
| 10 | 1660,122 | 7351 | 7673 |
| 15 | 1776,53 | 7284 | 7627 |
| 20 | 2059,915 | 7215 | 7689 |
| 25 | 2441,972 | 7314 | 7723 |
| 30 | 2566,972 | 7262 | 7747 |
| 35 | 2895,328 | 7365 | 7719 |
| 40 | 3078,589 | 7419 | 7681 |
| 45 | 3312,473 | 7405 | 7720 |
| 50 | 3625,495 | 7320 | 7699 |

На рисунку 3.23 представлено графік, що показує вплив розміру турніру на час виконання генетичного алгоритму. З цього графіку можна зробити висновок, що час роботи генетичного алгоритму зростає зі збільшенням розміру турніру.

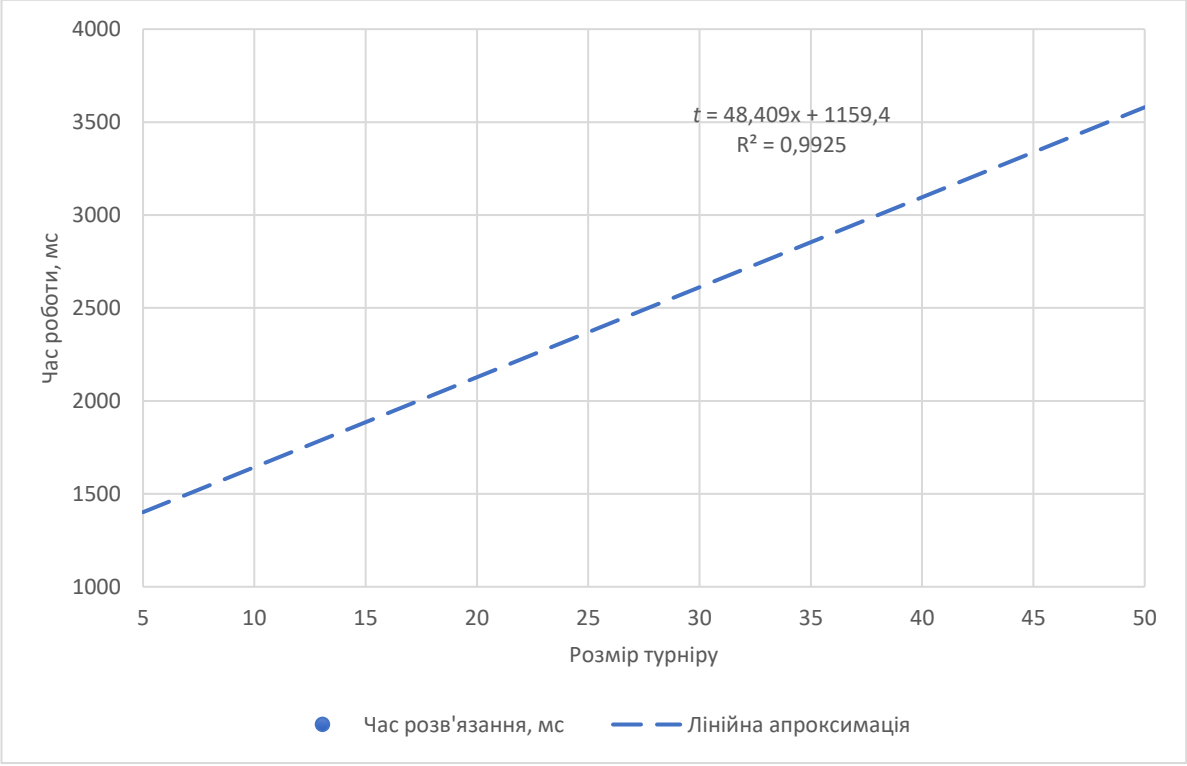


Рисунок 3.23 – Залежність часу роботи генетичного алгоритму від розміру турніру

На рисунку 3.24 наведено графік, який відображає залежність вартості маршруту від розміру турніру. З цього графіку можна зробити висновок, що розмір турніру майже не має впливу на вартість маршруту. Отже, менший розмір турніру може бути оптимальним вибором через зменшення часу роботи алгоритму.

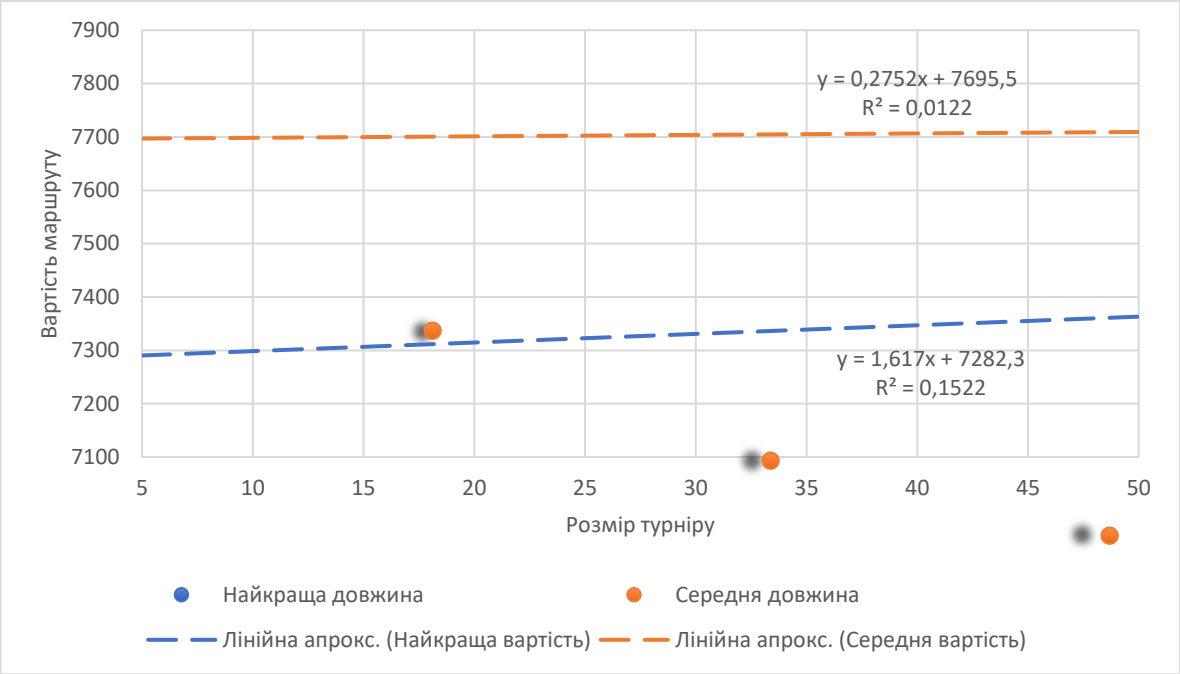


Рисунок 3.24 – Залежність вартості маршруту від розміру турніру

## 3.4 Висновки до третього розділу

В цьому розділі детально описані вимоги до розроблювального програмного засобу, обґрунтовано вибір ООП як підходу до розробки. Особливу увагу приділено використанню мови моделювання UML для системного моделювання, включаючи створення діаграми прецедентів та діаграми класів. Обґрунтовано вибір технології для розробки додатку.

Описано основний функціонал розробленого додатку, включаючи задання задачі, вибір та налаштування алгоритмів розв’язання задачі, побудову оптимального маршруту та експортування результатів.

Проведені ряд експериментів для порівняння генетичного алгоритму з

методом гілок та меж на задачах різною розмірністю. Ґрунтуючись на результатах експериментів, можна зробити наступні висновки:

– час роботи алгоритмів. Результати вказують на те, що алгоритм BnB є найшвидшим для розмірностей задачі менше 11, але стає неефективним для великих задач через експоненціальний ріст часу обчислень. Генетичний алгоритм залишається ефективним навіть для великих розмірностях задачі. Тому, в залежності від конкретної задачі, може бути обрано один з цих алгоритмів;

– розв’язання задач великої розмірності та задач з обмеженнями. З результатів видно, що зі збільшенням розмірності задачі час виконання зростає для всіх алгоритмів. Також, генетичний алгоритм виявляється більш масштабованим для розв’язання задач з обмеженнями порівняно з алгоритмом гілок та меж.

Щодо значень оптимальних налаштувань генетичного алгоритму можна зробити наступні висновки:

– ймовірність мутації. Зі збільшенням ймовірності мутації час виконання збільшується. Оптимальне значення ймовірності мутації знаходиться в діапазоні від 0.4 до 0.8. Проте, для швидкості виконання, оптимальне значення зазвичай обмежується діапазоном від 0.4 до 0.5;

– розмір популяції. Зі збільшенням розміру популяції час виконання GA збільшується незначною мірою. Оптимальний розмір популяції складає близько 200 осіб, але це значення може змінюватися в залежності від розмірності задачі;

– відсоток елітизму. Зі збільшенням відсотку елітизму час виконання GA зменшується. Оптимальний відсоток елітизму становить приблизно 0.65;

– розмір турніру майже не впливає на вартість маршруту, але зі його збільшенням зростає час виконання.

Отже, що вибір алгоритму та його параметрів повинен враховувати конкретні умови задачі, її розмірність. обмеження та ліміти по часу виконання.

# 4 ЗАХОДИ З ОХОРОНИ ПРАЦІ

Сучасний розвиток технологічного стану інфраструктури підприємств сприяє зростанню автоматизації та оптимізації виробничих процесів. Це зокрема стосується роботи з комп'ютерами, яка вимагає відповідності будівель проектній документації та забезпечення умов, що відповідають стандартам санітарного освітлення, мікроклімату, вібрації, шуму, вогнестійкості, а також контролю електромагнітного, ультрафіолетового та інфрачервоного випромінювання [33].

Суб'єктом в роботі розглядається працівник підприємства, який основі, використовуючи розроблений додаток здійснює побудову оптимальних логістичних маршрутів на виробництві.

Оскільки знаходження оптимальних маршрутів здійснюється за допомогою персонального комп'ютера, то в процесі виконання посадових обов'язків на працівника можуть впливати шкідливі та небезпечні виробничі фактори, які призводять до серйозних проблем фізичного та психологічного характеру.

## 4.1 Вимоги до приміщення

В рамках охорони праці, дотримання вимог до приміщення є невід'ємним аспектом забезпечення безпеки та здоров'я працівників. Перш за все, важливо забезпечити, що робоче місце буде достатньо просторим. Це означає не лише наявність достатнього місця для виконання трудових завдань, але й забезпечення вільного доступу до евакуаційних шляхів та необхідних ресурсів. Просторість також сприяє кращому розподілу робочих зон, що важливо для підтримки порядку та ефективності.

Робоче середовище користувачів персональних комп'ютерів вимагає особливої уваги до зорового комфорту, тому правильне освітлення є критично

важливим. Згідно «Правил охорони праці під час експлуатації електроннообчислюваних машин», рекомендується комбінація природного та штучного освітлення. Природне світло має надходити через вікна, орієнтовані переважно на північ або північний схід, із можливістю регулювання інтенсивності за допомогою жалюзі, штор чи зовнішніх козирків.

У деяких випадках, як у лабораторіях, де співробітник виконує свою роботу, може спостерігатися недолік природного освітлення, особливо якщо робоче місце розташоване далеко від джерела світла. В таких умовах штучне освітлення стає необхідністю не тільки у виробничих та побутових приміщеннях з обмеженим доступом до природного світла, але й для освітлення приміщень під час темного часу доби. Штучне освітлення, яке включає систему загального рівномірного освітлення та освітлення через екрани комп'ютерів, є важливим.

Згідно з Державними будівельними нормами (ДБН-В.2.5-28-2006) щодо природного і штучного освітлення, освітленість на робочому місці інженера-дослідника має бути у межах 300 – 750 Лк. Однак, часто фактичне рівень освітлення, який становить 220 – 280 Лк, не відповідає нормативним показникам. Для роботи з ПК, що класифікується як робота середньої точності, нормативний рівень освітлення повинен становити 400 Лк. В цьому контексті LED-лампи є оптимальним вибором для штучного освітлення, оскільки вони забезпечують один з найвищих показників світловіддачі.

Температурний режим та рівень вологості також мають велике значення. Надто висока або низька температура може не тільки знизити продуктивність, але й призвести до теплових або холодових шоків. Регулювання температури та вологості має бути адаптоване до специфіки роботи та індивідуальних потреб працівників.

Ергономічність робочих місць полягає у створенні умов, які мінімізують фізичний та психологічний стрес. Меблі, обладнання та розташування робочих зон мають бути адаптовані до фізіологічних та психологічних потреб працівників.

Захист від шуму є важливим, особливо у виробничих умовах. Використання звукоізоляційних матеріалів, регулярне обслуговування обладнання та забезпечення працівників засобами індивідуального захисту сприяє зниженню рівня шуму та запобіганню втрати слуху.

Забезпечення відповідності цим вимогам, постійний моніторинг їхнього виконання та адаптація до змінюваних умов та потреб працівників є ключовими для створення безпечного, здорового та продуктивного робочого середовища.

## 4.2 Заходи пожежної безпеки

Заходи пожежної безпеки є невід’ємною складовою загальних заходів з охорони праці та забезпечення безпеки на виробництві та в офісних приміщеннях. Вони спрямовані на запобігання виникнення пожеж, мінімізацію наслідків можливих пожежних інцидентів та підвищення рівня готовності до дій у разі пожежі.

Першочерговим є визначення потенційних джерел займання та розробка заходів щодо їх усунення або мінімізації ризику їх виникнення. Це може включати контроль за станом електропроводки, обладнання, утримання матеріалів, що легко займаються, у безпечних умовах, регулярні перевірки стану технічних засобів і систем.

Ретельне дотримання технологічних процесів, правил експлуатації обладнання та використання лише сертифікованих матеріалів та засобів є обов’язковими для запобігання виникненню пожеж.

Наступним важливим кроком є облаштування робочих місць системами пожежогасіння та сигналізації. Встановлення датчиків диму, систем автоматичного пожежогасіння, наявність вогнегасників, а також чітко облаштованих та відмічених шляхів евакуації є обов’язковими елементами забезпечення пожежної безпеки.

Організація навчань та інструктажів з пожежної безпеки для всіх

працівників є критично важливою. Працівники повинні знати основні правила поведінки у випадку пожежі, методи первинного пожежогасіння, порядок евакуації, а також вміти користуватися вогнегасниками та іншими засобами пожежогасіння.

Регулярне проведення перевірок, аудиту стану пожежної безпеки на підприємстві, а також оновлення планів евакуації і пожежогасіння з урахуванням всіх змін у виробничих процесах, структурі приміщень та обладнанні є необхідними для підтримки належного рівня захисту від пожеж.

Заходи пожежної безпеки мають бути частиною комплексного підходу до забезпечення безпеки працівників і мають виконуватися на постійній основі. Вони включають не тільки технічні та організаційні заходи, але й формування відповідального ставлення до питань безпеки серед усіх працівників.

# ВИСНОВКИ

Результатом виконання кваліфікаційної роботи стало підвищення ефективність роботи усього комплексу внутрішньозаводської логістики за рахунок розробки програмного засобу для прокладання оптимальних логістичних маршрутів з урахуванням координат пунктів призначень, кількості необхідного вантажу та характеристик транспортних засобів.

Для формалізація проблеми оптимізації логістичних маршрутів, було обрано розглядати її у контексті задачі комівояжера. Розглянуто різні варіації ЗК, та обрано ту, яка найбільш точно описує проблему оптимізації маршрутів на підприємстві – задача декількох комівояжерів з обмеженнями (MTSPC).

Проаналізовано найбільш перспективні методи вирішення ЗК, включаючи метод гілок та меж, метод динамічного програмування, та генетичний алгоритм, розглянуто їх переваги та недоліки. Для подальшої реалізації у програмному додатку обрано варіацію методу гілок та меж – метод Літтла та генетичний алгоритм.

Було створено програмний додаток у вигляді веб-сайту, в якому реалізовано згадані вище методи та інший функціонал, а саме: постановка задачі оптимізації логістичних маршрутів, вибір та налаштування алгоритмів розв’язання задачі, побудову оптимального маршруту та експортування результатів. Заплановано та проведено машинні експерименти для оцінки ефективність запропонованих методів та підбору оптимальних параметрів генетичного алгоритму.

З експериментальних даних, можна зробити висновок, що метод гілок та меж швидший для невеликих розмірностей задачі (менше 11) але зі збільшенням розмірності, програє генетичному алгоритму, через експоненціальний ріст часу обчислень. Також, генетичний алгоритм виявляється більш масштабованим для розв’язання задач з обмеженнями порівняно з алгоритмом гілок та меж, тому вибір методу розв’язання ЗК залежить від специфіки конкретної задачі.

Експерименти з параметрами ГА, показали, що найбільш точний розв’язок задачі отримуємо при значеннях ймовірності мутації від 0.4 до 0.8, проте з її збільшенням зростає час обчислень, тому оптимальним буде значення у діапазоні від 0.4 до 0.5. Зі збільшенням розміру популяції час виконання ГА збільшується незначною мірою. Оптимальний розмір популяції складає близько 200 осіб, але це значення може змінюватися в залежності від розмірності задачі. Також, встановлено, що зі збільшенням відсотку елітизму час виконання ГА зменшується. Оптимальний відсоток елітизму становить приблизно 0.65. Розмір турніру майже не впливає на точність знайденого маршруту.

Практичне використання отриманих результатів дозволить підвищити ефективність систем підтримки прийняття рішень у внутрішньозаводській логістиці за рахунок автоматизації процедур побудови оптимальних логістичних маршрутів.

За результатами дослідження опубліковано статтю у збірнику студентських наукових статей «Автоматизація та приладобудування» ADED-2023(2) [6] (Харківський національний університет радіоелектроніки), підготовлено тези доповіді на двадцять сьомий Міжнародний молодіжний форум «Радіоелектроніка та молодь у XXI столітті», конференція «Автоматизовані системи та комп’ютеризовані технології радіоелектронного приладобудування» [7] (Харківський національний університет радіоелектроніки) та Всеукраїнську науково-практичну конференцію «Комп’ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві» [8] (Харківський національний автомобільно-дорожній університет, 22 листопада 2023 р.).

# ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Logistic costs of manufacturing enterprises as objects of accounting and control. The institute of accounting, control and analysis in the globalization circumstances. 2019. No. 1-2.

2. Beskorovainyi V., Sudik A. Optimization of topological structures of centralized logistics networks in the process of reengineering. Innovative technologies and scientific solutions for industries. 2021. No. 1 (15). P. 23–31.

3. Гуляницький Л. Ф. Розробка моделей і наближених методів комбінаторної оптимізації та їх застосування в інформаційних технологіях : автореф. дис. д-ра техн. наук. Київ, 2005. 32 с.

4. Методичні вказівки з підготовки та захисту кваліфікаційної роботи здобувачами другого (магістерського) рівня вищої освіти спеціальності 151 Автоматизація та комп'ютерно-інтегровані технології, освітньо-професійних програм: «Автоматизоване управління технологічними процесами», «Комп'ютерно-інтегровані технологічні процеси і виробництва», «Комп'ютеризовані та робототехнічні системи» / Упоряд. І. Ш. Невлюдов, Р. В. Артюх, В. В. Безкоровайний, Н. П. Демська, В. В. Євсєєв, О. І. Филипенко, О. М. Цимбал. – Харків: ХНУРЕ, 2021. – 55 с.

5. ДСТУ 3008: 2015. Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлення. К.: ДП "УкрНДНЦ". 2016. 30 с.

6. Чернишенко О. В. Оптимізація маршрутів в логістичних мережах виробничого процесу. Автоматизація та Приладобудування («Automation and Development of Electronic Devices» ADED-2023) [Електронний ресурс] : збірник студентських наукових статей / Харківський національний університет радіоелектроніки ; [редкол.: І.Ш. Невлюдов та ін.]. – Харків : ХНУРЕ, 2023. – Вип. 2. – 408с.

7. Чернишенко О. В. Формалізація задачі реінжинірингу внутрішньозаводської логістики. Радіоелектроніка та молодь у XXI столітті:

Матеріали 27-го міжнар. молодіж. форуму. Харків: ХНУРЕ, 10 трав. 2023 р. С. 39–40.

8. Чернишенко О. В., Безкоровайний В. В. Формалізація задачі оптимізації маршрутів логістичної мережі виробничого процесу. Комп'ютерно-інтегровані технології автоматизації технологічних процесів на транспорті та у виробництві. Матеріали всеукраїнської науково-практичної конференції здобувачів вищої освіти і молодих учених. – Харків, ХНАДУ, 2023. – 320 с.

9. Крикавський Є. В., Похильченко О. А. Концептуальні орієнтири маркетингово-логістичного управління ланцюгами поставок. 2016.

10. Невлюдов І., Пономарьова Г., Кітченко Д. Автоматизація логістичних бізнес-процесів цифрового підприємства. ББК: у 290-21. С. 104.

11. Чикуркова А., Білянський Ю., Іванишин О. Види і функції логістики. Сучасні виклики та перспективи розвитку економіки, підприємництва та біржової діяльності. 2022. С. 122–124.

12. Маций О. Б. Огляд завдань маршрутизації, що зводяться до задачі комівояжера. Open information and computer integrated technologies. 2020. № 86. С. 152–159.

13. Hoffman K. L., Padberg M., Rinaldi G. Traveling salesman problem. Encyclopedia of operations research and management science. Boston, MA, 2013. P. 1573–1578.

14. Корпало А., Козачко О. Дослідження методів розв'язання задачі комівояжера на основі сучасних інтелектуальних технологій. Вінницький національний технічний університет. 2019.

15. Zhang J. Comparison of various algorithms based on TSP solving. Journal of physics: conference series. 2021. Vol. 2083, no. 3. P. 032007.

16. Louis P. Manufacturing execution systeme (MES). Manufacturing execution systems. Wiesbaden, 2009. P. 7–49.

17. Mullen T. D. Material flow control in complex manufacturing systems. 1991.

18. Osaba E., Yang X.-S., Del Ser J. Traveling salesman problem: a perspective review of recent research and new results with bio-inspired metaheuristics. Nature-Inspired computation and swarm intelligence. 2020. P. 135 - 164.

19. The multi-visit traveling salesman problem with multi-drones / Z. Luo et al. Transportation research part C: emerging technologies. 2021. Vol. 128. P. 103 - 122.

20. An algorithm for the traveling salesman problem / J. D. C. Little et al. Operations research. 1963. Vol. 11, no. 6. P. 972–989.

21. Poikonen S., Golden B., Wasil E. A. A branch-and-bound approach to the traveling salesman problem with a drone. INFORMS journal on computing. 2019. Vol. 31, no. 2. P. 335–346.

22. Fachini R. F., Armentano V. A. Exact and heuristic dynamic programming algorithms for the traveling salesman problem with flexible time windows. Optimization letters. 2018. Vol. 14, no. 3. P. 579–609.

23. Маринюк В. В. Розв"язування задачі оптимального інвестування методом динамічного програмування. Студентський вісник Національного університету водного господарства та природокористування. 2017. Вип. 2 (8). С. 91–93.

24. Cherkas D., Krasnoshlyk N. Research of genetic algorithms of solution of optimization problems. Cherkasy university bulletin: applied mathematics. informatics. 2020. № 2. С. 34–43.

25. Cheikhrouhou O., Khoufi I. A comprehensive survey on the multiple traveling salesman problem: applications, approaches and taxonomy. Computer science review. 2021. Т. 40. С. 100369.

26. Applying a genetic algorithm to a m-tsp: case study of a decision support system for optimizing a beverage logistics vehicles routing problem / D. E. Gomes et al. Electronics. 2021. Vol. 10, no. 18. P. 2298.

27. П’ятикоп А. Г. Адаптація генетичного алгоритму для рішення задачі комівояжера в порівнянні з алгоритмом "Мурахи" : master's thesis. 2020.

28. Bryant K. Genetic algorithms and the travelling salesman problem. 2000.

29. Singh N., Chouhan S. S., Verma K. Object oriented programming: concepts, limitations and application trends. 2021 5th international conference on information systems and computer networks (ISCON), Mathura, India, 22–23 October 2021. 2021.

30. Ozkaya M., Erata F. A survey on the practical use of UML for different software architecture viewpoints. Information and software technology. 2020. Vol. 121. P. 275.

31. Reis A. J. D. Introduction to Programming Using Java. Jones & Bartlett Learning, LLC, 2011.

32. Radjenovic J., Milosavljevic B., Surla D. Modelling and implementation of catalogue cards using FreeMarker. Program. 2009. Vol. 43, no. 1. P. 62–76.

33. Інструкція з охорони праці при роботі з комп'ютером, принтером, ксероксом та іншою оргтехнікою [Електронний ресурс]. – 2018. – Режим доступу до ресурсу: https://osvita-docs.com/node/41.