

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ ЗАСОБІВ
РОЗВ'ЯЗАННЯ ЗАВДАНЬ ДИСКРЕТНОЇ ОПТИМІЗАЦІЇ
RESEARCH AND SOFTWARE IMPLEMENTATION OF TOOLS
FOR SOLVING THE DISCRETE OPTIMIZATION PROBLEMS

Виконав: студент 2 курсу, групи КНТ-112м

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ДЗУГАНЬ А.О.

(ПРИЗВИЩЕ та ініціали)

Керівник КОЛПАКОВА Т.О.

(ПРИЗВИЩЕ та ініціали)

Рецензент КОЦУР М.І.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
 Кафедра програмних засобів
 Ступінь вищої освіти магістр
 Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
 Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
 “ ” 2023 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ДЗУГАНЬ Андрія Олександровича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація засобів розв'язання завдань дискретної оптимізації. Research and Software Implementation of Tools for Solving the Discrete Optimization Problems

керівник проєкту (роботи) к.т.н., доцент, КОЛПАКОВА Тетяна Олексіївна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “14” листопада 2023 року № 437

2. Строк подання студентом проєкту (роботи) 01 грудня 2023 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали і методи. 3. Опис програми. 4. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	КОЛПАКОВА Т.О., доцент		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання “ 16 ” жовтня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки	2 тиждень	Розділ 2
4	Розробка структури програми.	3 тиждень	Розділ 3
5	Розробка програми.	4-7 тижні	Розділи 3,4
6	Тестування та експериментальне дослідження програмного забезпечення.	8 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	9-10 тижні	Додатки
8	Нормоконтроль та рецензування.	11 тиждень	
9	Захист роботи.	12 тиждень	

Студент(ка)

(підпис) Андрій ДЗУГАНЬ
(Імя ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Тетяна КОЛПАКОВА
(Імя ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
92 с., 35 рис., 3 табл., 3 дод., 17 джерел.

C++, QT CREATOR, ДИСКРЕТНА ОПТИМІЗАЦІЯ, ЕВОЛЮЦІЙНІ
АЛГОРИТМИ, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ.

Об'єкт дослідження – процес розробки програмного забезпечення для дискретної оптимізації.

Предмет дослідження – методи та програмні засоби для розв'язання завдань дискретної оптимізації.

Метою роботи є дослідження та розробка програмного забезпечення для розв'язання завдань дискретної оптимізації.

Матеріали, методи та технічні засоби: мова програмування C++, середовище розробки Qt Creator.

Результати. Розроблено програмне забезпечення для розв'язання завдань дискретної оптимізації за допомогою мови програмування C++ та середовища розробки Qt Creator.

Практична цінність роботи полягає у розробці програмного забезпечення, що реалізує методи дискретної оптимізації та дозволяє розв'язувати задачі пошуку оптимальних рішень.

Висновки. Виконано проєктування програмного забезпечення для розв'язання завдань дискретної оптимізації. Розроблено програмне забезпечення для розв'язання завдань дискретної оптимізації. Здійснено тестування розробленого програмного забезпечення для розв'язання завдань дискретної оптимізації.

Галузь використання – програмні засоби для пошуку оптимальних рішень.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 92 pages, 35 figures, 3 tables, 3 appendixes, 17 sources.

C++, QT CREATOR, DISCRETE OPTIMIZATION, EVOLUTIONARY ALGORITHMS, SOFTWARE.

The object of research is the process of developing software for discrete optimization.

The subject of the research is software for solving discrete optimization problems.

The purpose of this work is to develop the software for solving discrete optimization problems.

Materials, methods and technical tools: C++ programming language, Qt Creator development environment.

Results. The software tools for solving discrete optimization problems using C++ programming language and Qt Creator development environment have been developed.

The practical value of the work lies in the development of software that implements discrete optimization methods and allows you to solve problems of finding optimal solutions.

Conclusions. The software for solving discrete optimization problems has been designed. The software for solving discrete optimization problems has been developed. The developed software for solving discrete optimization problems has been tested.

Scope of use – software tools for finding optimal solutions.

ЗМІСТ

	С.
Перелік скорочень та умовних познач	8
Вступ	9
1 Аналіз предметної області	11
1.1 Поняття про дискретну оптимізацію	11
1.2 Аналіз програмного забезпечення для дискретної оптимізації	24
1.3 Висновки за розділом 1	34
2 Матеріали і методи	36
2.1 Вибір мови програмування	36
2.2 Вибір середовища розробки для створення програмного забезпечення для розв'язання задачі комівояжера	38
2.3 Розробка моделі взаємодії користувача з програмним забезпеченням розв'язання задачі комівояжера	41
2.4 Висновки за розділом 2	43
3 Опис програми	44
3.1 Структура програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації	44
3.2 Функціонування програмного забезпечення для розв'язання задачі комівояжера	45
3.3 Проєктування програмного забезпечення для розв'язання задачі комівояжера	47
3.4 Проєктування інтерфейсу програмного забезпечення для розв'язання задачі комівояжера	51
3.5 Висновки за розділом 3	55
4 Експлуатація, тестування та експериментальне дослідження програми	56
4.1 Призначення й умови застосування програми	56
4.2 Характеристики програми для розв'язання задачі комівояжера	57
4.3 Інструкція по експлуатації програми	57

4.3.1	Звернення до програми	57
4.3.2	Вхідні й вихідні дані	58
4.3.3	Повідомлення	58
4.4	Виконання програмного забезпечення для розв'язання задачі комівояжера	58
4.5	Тестування програмного забезпечення для розв'язання задачі комівояжера	62
4.6	Висновки за розділом 4	62
	Висновки	63
	Перелік джерел посилання	66
	Додаток А Технічне завдання	68
	Додаток Б Фрагмент тексту програми	72
	Додаток В Слайди презентації	84

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API – Application Programming Interface;

TSP – Traveling Salesman Problem;

ГА – генетичний алгоритм;

ЕА – еволюційний алгоритм;

ПЗ – програмне забезпечення.

Вступ

У наш час широко використовуються програмні засоби для розв'язання завдань дискретної оптимізації, що допомагають знаходити оптимальні рішення в умовах обмежених ресурсів. Програмні засоби забезпечують автоматизацію багатьох процесів у процесі дискретної оптимізації, що дозволяє відштовхуватися від задачі і шукати оптимальні рішення швидше та ефективніше у порівнянні з іншими методами, зокрема, методом повного перебору [1]-[6].

Програмне забезпечення дискретної оптимізації застосовуються в багатьох сферах, від технічних до соціальних, і має велике значення для вирішення проблем різного характеру [1]-[6]. Зокрема, використання програмного забезпечення дискретної оптимізації дозволяє зменшити витрати ресурсів, таких як час, кошти та енергія, завдяки знаходженню оптимальних рішень. Сучасні програмні рішення використовують потужні алгоритми та обчислювальні ресурси, що дозволяє знаходити оптимальні розв'язки швидко та з високою точністю [1]-[6].

Отже, у наш час програмні засоби для розв'язання завдань дискретної оптимізації є важливим інструментом для забезпечення ефективності, вдосконалення процесів та здійснення інновацій у різних сферах діяльності [1]-[6]. Проте деяке програмне забезпечення може бути досить дорогим або обмежено доступним, що може бути важливим фактором для більш широкого використання. Деякі програми можуть використовувати неефективні алгоритми, деякі використовувані моделі можуть враховувати ідеальні умови, які не завжди відображають реальність, що у деяких випадках може призвести до отримання некоректних результатів. Крім того, деякі оптимізаційні алгоритми, що можуть використовуватися у програмних засобах дискретної оптимізації, можуть вимагати значних обчислювальних ресурсів, що, у свою

чергу, може вимагати великого часу та обчислювальних ресурсів для вирішення практичних задач [1]-[6].

Тому актуальною є розробка програмного забезпечення для розв'язання завдань дискретної оптимізації.

У дипломній кваліфікаційній роботі магістра розв'язується актуальне завдання розробки програмного забезпечення для розв'язання завдань дискретної оптимізації.

Для досягнення поставленої мети у кваліфікаційній роботі магістра необхідно розв'язати такі задачі:

- виконати аналіз предметної області та програмних засобів для розв'язання завдань дискретної оптимізації;
- розробити модель взаємодії користувача з програмним забезпеченням для розв'язання завдань дискретної оптимізації;
- здійснити проектування програмного забезпечення для розв'язання завдань дискретної оптимізації;
- створити програмне забезпечення для розв'язання завдань дискретної оптимізації;
- виконати тестування розробленого програмного забезпечення для розв'язання завдань дискретної оптимізації.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Поняття про дискретну оптимізацію

Дискретна оптимізація є областю інформаційних технологій, яка фокусується на ефективному пошуку найкращих рішень серед обмеженого або визначеного набору можливих альтернатив [1]-[4].

Однією з відомих і класичних задач дискретної оптимізації є задача комівояжера, що полягає у пошуку найкоротшого маршруту серед заданого списку міст і відстаней між ними. Задачі дискретної оптимізації виникають у різних областях, таких як біоінформатика, проектування телекомунікаційних мереж, авіапланування, проектування схем та ефективне розподілення ресурсів. Ця галузь інформаційних технологій тісно пов'язана з іншими галузями, такими як алгоритми наближення, опукла та тропічна геометрія, теорія чисел, реальна алгебраїчна геометрія, теорія складності, квантові обчислення, машинне навчання та математична логіка [1]-[4].

Програмне забезпечення для дискретної оптимізації підтримує процес розробки та дослідження обчислювальних методів дискретної оптимізації для експериментального вирішення комбінаторних задач, що виникають у різних областях людської діяльності [1]-[4].

Дискретна оптимізація являє собою процес пошуку найкращого рішення з обмеженою кількістю можливих варіантів шляхом ефективних обчислювальних методів. Зазвичай кількість цих можливих варіантів є надзвичайно великою, і потрібно розумно вибирати, які з них розглядати, замість бездумної перевірки всіх. Щоб досягти успіху, необхідно розуміти структуру проблеми [1]-[4].

До завдань дискретної оптимізації відносять такі завдання [1]-[4]:

- пошук мінімальної кількості охоплюючих дерев;
- проблема найкоротшого шляху;
- визначення максимальних потоків у графах;

- пошук збігів;
- проблема про оптимальне розташування предметів у рюкзаку;
- оптимізації потоків мінімальних витрат;
- завдання цілочисельного програмування [1]-[4].

Дискретна оптимізація та відповідні програмні засоби знайшли широке використання у різних галузях економіки. Проаналізуємо практичні застосування програмних засобів дискретної оптимізації [1]-[4].

Програмні засоби як дискретні оптимізатори впроваджують стратегію планування на основі витрат. Це означає, що програмні засоби дискретної оптимізації аналізують всі можливі варіанти планів, спрямовуючись на пошук найбільш економічно вигідного підходу з точки зору загальних витрат [1]-[4].

План, запропонований дискретним оптимізатором, вважається прийнятним, якщо він відповідає всім обмеженням, які були встановлені у профілі дискретної оптимізації. Остаточне рішення може включати в себе порушення термінів виконання або обмежень щодо запасів безпеки. Терміни виконання та рівень страхового запасу розглядаються як гнучкі обмеження з можливістю порушення, внаслідок чого дискретний оптимізатор генерує лише ті плани, які порушують ці гнучкі обмеження, якщо це є найбільш ефективним з погляду витрат відповідно до даних, внесених у систему [1]-[4].

Дискретний оптимізатор та відповідне програмне забезпечення можуть використовуватися для прийняття рішень щодо ресурсів в рамках стратегії планування на основі дискретної оптимізації. Це означає, що він використовує витрати як основу для прийняття рішень стосовно виробництва, транспортування, заготовлення, зберігання та доставки продуктів та відповідних обсягів, використання ресурсів та моделей виробничих процесів або структур виробничих даних, дати та часу виробництва, транспортування, закупівлі, зберігання та доставки, місця для виробництва, заготівлі, зберігання, доставки, а також пункти відправлення та призначення для транспортування готової продукції [1]-[4].

З огляду на можливість розв'язання програмними засобами різних завдань дискретної оптимізації, як вхідні дані можуть використовуватися обсяги використання різних ресурсів виробництва. Таким чином, можна використовувати дискретну оптимізацію для планування обсягів виробничих партій. Програмні засоби у вигляді дискретних оптимізаторів також дозволяють проводити планування обсягу партій, коли замовлення групуються великими партіями через високі витрати на конфігурацію виробничих ліній [1]-[4].

Методи дискретної оптимізації також використовуються для розв'язання задач планування, що дозволяє врахувати всі важливі фактори одночасно та знаходити оптимальне рішення. Цей підхід стає більш складним, коли активуються більше обмежень, що призводить до збільшення часу, необхідного для вирішення оптимізаційної задачі. Зазвичай процес оптимізації за допомогою відповідних програмних засобів запускають як фонову задачу, оскільки він може зайняти досить велику кількість часу [1]-[4].

Завдання комбінаторної оптимізації можуть бути класифіковані на лінійні оптимізаційні проблеми та більш складні задачі дискретної оптимізації [1]-[4].

Для задач лінійної оптимізації у профілі дискретного оптимізатора часто доступні наступні методи: первинний симплекс-метод, подвійний симплексний метод та метод внутрішньої точки [1]-[4]. Усі ці методи дозволяють досягти оптимального рішення, проте час виконання може відрізнятися, що впливає на вибір методу. Зазвичай немає універсального правила для вибору найкращого методу для конкретної задачі дискретної оптимізації, за винятком тестування кожного методу окремо. Ефективним способом визначення оптимального методу дискретної оптимізації є порівняльний аналіз за допомогою тестових сценаріїв. Це пояснюється тим, що вибір найкращого методу дискретної оптимізації залежить від структури вхідних даних та постановки завдання [1]-[4].

Оптимізація у дискретному вигляді виникає як проблема у плануванні у випадках, коли модель містить такі особливості: використання цілих чисел для визначення розмірів партій для транспортування або в інших практичних завданнях, обмеження на дискретність у транспортуванні, потреба у дискретному збільшенні виробничої потужності, встановлення мінімальних розмірів партій для транспортування, використання поштучно лінійних функцій для оцінки витрат на транспортування, виробництво чи закупівлю, фіксація витрат матеріалу, врахування планування розміру партій за допомогою різних часових періодів [1]-[4].

Крім того, важливо зазначити, що кусково-лінійна функція витрат, яку можна визначити у вхідних даних для розв'язання завдань дискретної оптимізації, може приймати форму опуклої функції витрат, де вартість за одиницю збільшується зі збільшенням обсягів, наприклад, при моделюванні понаднормових або нічних змін, або увігнутої функції витрат, де вартість за одиницю зменшується зі збільшенням обсягів, наприклад, при моделюванні ставок фрахту [1]-[4]. Опуклі функції витрат не ускладнюють задачу дискретної оптимізації і можуть бути ефективно вирішені. Проте, їх також можна моделювати за допомогою альтернативних підходів без використання кусково-лінійних функцій вартості [1]-[4].

Програмне забезпечення дискретної оптимізації може виконувати пошук у дискретному стані при розв'язанні практичних задач планування, розрізняючи пріоритети між замовленнями клієнтів та передбаченим попитом. У випадку високого пріоритету замовлення клієнта, дискретний оптимізатор надає перевагу цьому, враховуючи пріоритет передбаченого виправленого попиту та передбаченого попиту. В межах кожного рівня пріоритету програмне забезпечення дискретної оптимізації приймає остаточне рішення, використовуючи всю доступну інформацію щодо витрат. При застосуванні дискретної оптимізації на основі вартості, оптимізатор використовує дані щодо

витрат у вигляді штрафних функцій, які містяться в основних даних продукту, для визначення оптимального рішення [1]-[4].

Методи дискретної оптимізації дозволяють знизити вимоги до обчислювальної потужності та обсягу пам'яті, необхідного для проведення оптимізації. Декомпозиція розв'язуваної практичної задачі може також бути способом для оптимізатора знайти оптимальне рішення, особливо в разі великих дискретних завдань [1]-[4].

Дискретна оптимізація може впливати на процес агрегатного планування, перетворюючи його у процес вертикального планування. Для зменшення обсягу моделі, яка піддається оптимізації, дискретний оптимізатор може обмежити планування на рівні групи продуктів та їх розташувань, за умови, що попит визначено на більш низькому рівні [1]-[4].

З іншого боку, дискретна оптимізація дозволяє планувати частину ланцюга поставок, перетворюючи сукупне планування на планування на побічному рівні. При використанні методів дискретної оптимізації для розв'язання практичних завдань, наприклад, завдань планування поставок, існує можливість обмежити певні характеристики, наприклад, продукти або місця, які враховуються під час оптимізації. Наприклад, якщо дискретна оптимізація проводиться на рівні заводу, але прогнози робляться на рівні клієнта, оптимізатор може використовувати вимоги на рівні заводу і застосовувати їх під час виконання оптимізації [1]-[4].

Поступова дискретна оптимізація є методом планування, який базується на методах дискретної оптимізації і застосовується до частини моделі або існуючих планів. Цей підхід може ставати складним завданням через те, що оптимізатор не може заздалегідь передбачити всі можливі надходження, які можуть виникнути через фіксоване розташування графіка попереднього виконання. Також може бути важко враховувати вхідні продукти або продукти, доступні для транспортування до вихідного пункту призначення через маршрут транспортування [1]-[4].

Отже, дискретна оптимізація включає набір методів, що дозволяють розв'язувати завдання, які вимагають вибору найкращого варіанту зі списку можливостей. Як одна з галузей інформаційних технологій, дискретна оптимізація є досить широко використовуваною. Дискретна оптимізація також є предметом вивчення в теоретичній інформатиці, де дослідження зосереджені на алгоритмах розв'язування різних завдань, ефективність яких доведено в умовах зростання розмірів задачі до нескінченності. В обох випадках актуальність використання методів дискретної оптимізації виходить з обчислювального боку цієї галузі, зокрема великої кількості завдань практичного і теоретичного значення, які можуть бути зведені до пошуку оптимального рішення у дискретному просторі станів. Оптимальна відповідність завдання стосується знаходження найкращого способу об'єднання різних об'єктів. Зазвичай такі моделі дискретної оптимізації визначаються на дискретних структурах, таких як мережі, графи і матриці. Обчислювальний аспект дискретної оптимізації є важливим у сфері дослідження операцій, де дискретна оптимізація є важливим інструментом для розв'язання завдань, що виникають у промисловості та торгівлі [1]-[4].

Одним з ключових завдань дискретної оптимізації є завдання пошуку найкоротшого маршруту, що включає у себе пошук швидкого способу переміщення від одного пункту до іншого через мережу доріг. Задача комівояжера полягає в пошуку найкоротшого маршруту для відвідування певного набору міст [1]-[4].

Завдання комівояжера (Traveling Salesman Problem, TSP) полягає у пошуку оптимального шляху при відвідуванні певних населених пунктів [5]-[7]. Однією з основних проблем є вибір оптимального маршруту для доставки з багатьма пунктами видачі. Відсутність найкоротшого шляху може призвести до зайвого часу, який кур'єр витратить на перевезення до кінцевого призначення. Іноді це може стати проблемою, особливо коли доступні різні варіанти маршрутів, і важко визначити, який з них є найбільш

ефективним [5]-[7]. У результаті, неефективне розв'язання задачі комівояжера може призвести до втрати часу та затримок у доставці [5]-[7].

При розв'язанні завдання комівояжера необхідно враховувати деякі особливості та проблеми, що виникають при роботі з реальними даними, зокрема, щоденно необхідно здійснювати кілька поставок протягом обмеженого часу, і, отже, існує низка часових обмежень. Іншою проблемою є ризик непередбачених змін, коли необхідно виконувати додаткові та термінові візити, а іноді деякі візити відкладаються або скасовуються через відсутність клієнтів. Як наслідок, задача комівояжера у сучасних умовах є задачею дискретної оптимізації, яка є складною математичною проблемою з необхідністю її розв'язання за допомогою спеціальних програмних засобів, оскільки вона вимагає врахування великої кількості змінних [5]-[7]. Отже, актуальною є розробка програмного забезпечення для розв'язання завдання комівояжера, яке дозволить створювати найкоротші маршрути, враховуючи низку можливих перестановок і комбінацій для ефективної доставки та здатності змінювати маршрути в останній момент [5]-[7].

У наш час використовуються різні методи для розв'язання завдання комівояжера. Метод найближчого сусіда (Nearest Neighbor) є одним з основних підходів для розв'язання задачі комівояжера. Він базується на припущенні, що комівояжер спочатку повинен відвідати найближче місто. Коли всі міста вже включені у маршрут, водій повертається до початкового пункту [5]-[7]. Для розв'язання задачі комівояжера за допомогою цього методу, користувач починає з випадкового міста, а потім переходить до найближчого міста, яке ще не було відвідане, і так далі. Коли всі міста будуть включені в маршрут, необхідно повернутися до початкового міста [5]-[7].

Метод розгалужень і меж (The Branch and Bound Algorithm) для задачі комівояжера використовує стратегію поділу задачі на менш складні задачі. Кожна з цих задач може мати кілька можливих рішень, і вибір рішення для однієї задачі може впливати на розв'язання наступних [5]-[7].

Метод грубої сили (The Brute Force Algorithm) для задачі комівояжера використовує метод динамічного програмування для розгляду всіх можливих перестановок маршруту з мінімальною вартістю. Спочатку обчислюється загальна відстань, а потім розглядаються всі можливі способи отримання цієї відстані через перестановки міст. Знаходиться відстань для кожного варіанту, і найкоротший маршрут вважається найкращим рішенням [5]-[7].

Програмне забезпечення для розв'язання задачі комівояжера знайшло широке використання у різних практичних сферах. Більшість компаній відчують на собі наростаючу складність проблеми комівояжера, особливо коли стає питання про останній етап доставки, відомий як доставка останньої милі. Цей термін означає доставку товарів від складу (або депо) до кінцевого пункту, де знаходиться клієнт. Управління ланцюгом постачання підкреслює, що це самий витратний етап у процесі доставки, особливості якого необхідно враховувати при постановці та розв'язанні задачі комівояжера. Високі витрати доставки останньої милі часто виникають через відсутність ефективного програмного забезпечення управління маршрутами транспорту, що допомагає знаходити оптимальні маршрути з метою уникнення надмірних оперативних витрат. Таким чином, використовуючи відповідне програмне забезпечення для розв'язання задачі комівояжера, можна уникнути проблем з доставкою та знизити витрати на доставку останньої милі [5]-[7].

Програмне забезпечення для управління доставкою використовує автоматизований процес розв'язання завдань дискретної оптимізації, який не потребує ручного втручання або розрахунків для визначення оптимальних маршрутів [5]-[7].

Задача комівояжера є завданням дискретної оптимізації, яке вивчається в теорії графів та галузі дослідження операцій. У цьому оптимізаційному завданні всі вузли або міста графа пов'язані за допомогою зупинок або прямих маршрутів. Вага кожного ребра показує відстань між двома містами [5]-[7]. Завдання полягає в тому, щоб знайти оптимальний маршрут, який відвідає

кожне місто один раз і повернеться до початкової точки та пункту прибуття після проходження всіх міст [5]-[7].

TSP часто досліджується у загальній версії, яка є проблемою маршрутизації транспортного засобу. Це одна з найбільш вивчених задач дискретної оптимізації та полягає у пошуку або створенні набору маршрутів для оптимізації термінів, вартості пального та доставки [5]-[7].

Багато рішень для задачі комівояжера і задачі маршрутизації автомобільного транспорту ґрунтуються на академічних концепціях дискретної оптимізації, що означає, що вони мають обмежену застосовність в реальному житті. Такі ситуації виникають, оскільки багато з цих рішень спрямовані на досягнення ідеальності і вимагають використання складних методів динамічного програмування. Отже, якщо компанії дійсно бажають покращити ефективність своєї діяльності, їм потрібно використовувати програмне забезпечення для дискретної оптимізації маршрутів із вбудованим засобом для розв'язання задачі комівояжера [5]-[7].

Програмні засоби для розв'язання задачі комівояжера повинні надавати інтегроване планування маршрутів і оптимізації, щоб забезпечити вчасну та ефективну доставку посилок. Крім того, це програмне забезпечення повинно використовувати ефективний алгоритм, який забезпечує отримання реалістичних рішень для задачі комівояжера так, щоб менеджер по доставці може легко створювати оптимальні маршрути за декілька хвилин [5]-[7].

Отже, узагальнено задачу комівояжера можна визначити як необхідність знайти оптимальний та найкоротший маршрут, який обов'язково включає в себе всі точки маршруту, включаючи початкову і кінцеву точки. Кількість можливих маршрутів зростає експоненціально зі збільшенням кількості місць, які необхідно відвідати. Пошук оптимального рішення стає обчислювально складним, навіть при помірному розмірі задачі [5]-[7].

Для розв'язання задачі комівояжера ефективно можуть використовуватися генетичні алгоритми дискретної оптимізації [8]-[10].

Генетичні алгоритми (ГА) є методами пошуку, що ґрунтуються на природній еволюції та є евристичним засобом розв'язання завдань дискретної оптимізації. Вони успішно використовуються для вирішення різноманітних практичних задач, що мають значну обчислювальну складність [8]-[10].

Генетичні алгоритми вдосконалено наслідують природну еволюцію та ґрунтуються на принципі Дарвіна. Вони використовують абстрактну версію еволюційних процесів для розробки рішень у вирішенні певних завдань. У кожному ГА використовується популяція штучних хромосом. Кожна хромосома представляє собою рішення задачі та має придатність, яка є числовим показником, що визначає, наскільки ефективним є це рішення для конкретної задачі [8]-[10].

Генетичний алгоритм розпочинає свою роботу з початкової популяції, яка формується шляхом випадкової генерації хромосом. Після цього ГА проводить відбір і рекомбінацію хромосом на основі їх придатності, створюючи нову популяцію наступного покоління. Під час рекомбінації генетичний матеріал батьківських хромосом обирається і комбінується, утворюючи таким чином дочірні хромосоми. Потім ці дочірні хромосоми передаються до наступних поколінь. Цей процес повторюється, утворюючи послідовні покоління, і середня придатність хромосом зростає, поки не буде досягнуто певного критерію зупинки. Таким чином, ГА поступово еволюціонує та знаходить найкраще рішення для задачі [8]-[10].

ГА складається з різних компонентів, і це особливість, яка дозволяє використовувати стандартні компоненти з мінімальними змінами для багатьох різних видів ГА. Це значно спрощує процес впровадження алгоритму. Основними принципами та компонентами ГА є кодування хромосом, визначення функцій придатності, відбір, рекомбінація та моделювання еволюційних процесів [8]-[10].

У генетичних алгоритмах кодується набір можливих рішень для конкретної задачі. Потім ці рішення піддаються комбінації та зміні (подібно до

природної еволюції), щоб створити нові рішення, які як спадкоємці розвиваються протягом кількох поколінь. Кожному рішенню присвоюється значення придатності (на основі його об'єктивної функціональної цінності), і більш придатні рішення мають більше шансів на створення нащадків у своєму поколінні [8]-[10].

Отже, генетичні алгоритми використовують еволюцію кращих рішень протягом поколінь, доки не досягнемо критерію зупинки [8]-[10].

Генетичні алгоритми в цілому є стохастичними за своєю природою, але працюють значно краще, ніж випадковий локальний пошук (коли просто випробовуються різні випадкові рішення і вибирається найкраще рішення на даний момент), оскільки вони також використовують інформацію про попередні покоління [8]-[10].

Генетичні алгоритми характеризуються численними перевагами, що робить їх дуже популярними в різних областях при розв'язанні завдань дискретної оптимізації. До цих переваг належать [8]-[10]:

- не потрібно мати інформацію про похідну цільової функції, що може бути недоступною у багатьох реальних задачах;
- ГА працюють, як правило, швидше та ефективніше, ніж традиційні методи дискретної оптимізації;
- високий потенціал для паралельних обчислень;
- можливість оптимізувати як безперервні, так і дискретні функції, а також вирішувати багатоцільові проблеми;
- розв'язком є список прийнятних рішень, а не єдине рішення;
- ГА постійно адаптуються та знаходять відповіді на проблеми, які можуть з часом покращитися;
- ці алгоритми особливо корисні, коли простір пошуку великий і містить багато параметрів [8]-[10].

Генетичні алгоритми можуть швидко надавати рішення, які вважаються задовільними та прийнятними. Це робить їх привабливими для вирішення завдань дискретної оптимізації [8]-[10].

Деякі задачі в інформатиці відносяться до класу складних, що означає, що навіть найпотужніші комп'ютерні системи потребують великої кількості часу (іноді, навіть років), щоб знайти оптимальні рішення. У таких випадках генетичні алгоритми можуть бути корисними, надаючи майже оптимальні рішення за короткий час [8]-[10].

При розробці генетичних алгоритмів для розв'язання конкретної задачі дискретної оптимізації існує велика кількість аспектів, які потрібно враховувати. Вибір способу кодування залежить від характеру проблеми, яку ви намагаєтеся вирішити. У сучасних ГА вже стали стандартом різні форми кодування, включаючи послідовності цілих чисел або чисел з плаваючою комою. Проте, коли кількість алелей у хромосомі збільшується, наприклад, у випадку рядків з числами з плаваючою комою, кількість можливих комбінацій стає значно вищою. Сучасні ГА використовують різні методи представлення, щоб забезпечити, що можливі хромосоми наявні в наборі допустимих рішень для проблеми [8]-[10].

Після вибору кодування, слід враховувати інші важливі аспекти, такі як: форма функції пристосованості; розмір популяції; параметри операторів схрещування та мутації та їхні оптимальні значення; вибір еволюційної стратегії; і умови зупинки або перезапуску алгоритму. Зазвичай, ефективний підхід до проектування включає комбінування здобутого досвіду, створення моделей для конкретної проблеми та експерименти з різними налаштуваннями алгоритму та іншими параметрами [8]-[10].

Стандартна модель генетичного алгоритму (GA) з використанням повної заміни традиційними генетичними операторами може бути описаним наступним чином [8]-[10]. Метод розпочинає свою роботу зі створення початкової популяції хромосом випадковим чином. Після цього розраховується

відповідність (fitness) для кожної хромосоми у початковій популяції. Потім створюється порожня наступна популяція, а потім повторюються наступні кроки до створення певної кількості хромосом у популяції [8]-[10]:

- за допомогою пропорційного вибору відбираються дві хромосоми з початкової популяції;
- застосовується точкове схрещування до обраних хромосом з певною ймовірністю, щоб отримати дочірню хромосому;
- застосовується рівномірна мутація до нової хромосоми з ймовірністю мутації;
- нова хромосома додається до наступної популяції [8]-[10].

Після цього виконується заміна початкову популяції на новостворену популяцію [8]-[10]. Якщо критерії зупинки не виконано, виконується повернення до другого етапу [8]-[10].

На практиці гнучкість та надійність генетичних алгоритмів означають, що реалізація ГА для розв'язання конкретної задачі дискретної оптимізації зазвичай є досить простою за допомогою використання стандартних еволюційних операторів. Вдосконалення генетичних алгоритмів до такого рівня реалізації, коли вони переважають інші доступні підходи для конкретної задачі дискретної оптимізації є більш складним завданням. Генетичні алгоритми виявилися особливо корисними для вирішення проблем, де існують обмеження і традиційні методи оптимізації на основі градієнта є не ефективними [8]-[10].

Визначено, що у наш час широко використовуються програмні засоби для розв'язання завдань дискретної оптимізації, які допомагають знаходити оптимальні рішення в умовах обмежених ресурсів. Показано, що дискретна оптимізація є областю інформаційних технологій, яка фокусується на ефективному пошуку найкращих рішень серед обмеженого або визначеного набору можливих альтернатив. Проаналізовано завдання дискретної оптимізації. Відзначено, що дискретна оптимізація та відповідні програмні

засоби знайшли широке використання у різних галузях економіки, проаналізовано практичні застосування програмних засобів дискретної оптимізації. Визначено, що важливою задачею дискретної оптимізації є задача комівояжера, що полягає у пошуку найкоротшого маршруту серед заданого списку міст і відстаней між ними. Проаналізовано особливості та проблеми, що виникають при роботі з реальними даними при розв'язанні задачі комівояжера. Досліджено методи розв'язання задачі комівояжера. Відзначено, що програмні засоби для розв'язання задачі комівояжера повинні надавати інтегроване планування маршрутів, щоб забезпечити вчасну та ефективну доставку посилок. Показано, що для розв'язання задачі комівояжера ефективно можуть використовуватися генетичні алгоритми дискретної оптимізації.

1.2 Аналіз програмного забезпечення для дискретної оптимізації

Проаналізуємо програмне забезпечення підтримки для дискретної оптимізації [11]-[13].

Програмне забезпечення Lingo є комплексним програмним інструментом, розробленим для створення оптимізаційних моделей, таких як лінійні, нелінійні, квадратичні, обмежені, конусні, стохастичні та дискретні моделі, з метою вирішення оптимізаційних задач, зокрема задач дискретної оптимізації. Програмне забезпечення Lingo (рис. 1.1) надає інтегрований пакет, який включає потужний мову для формулювання оптимізаційних завдань, повноцінне середовище для їх створення та редагування, а також набір швидких вбудованих рішень [11].

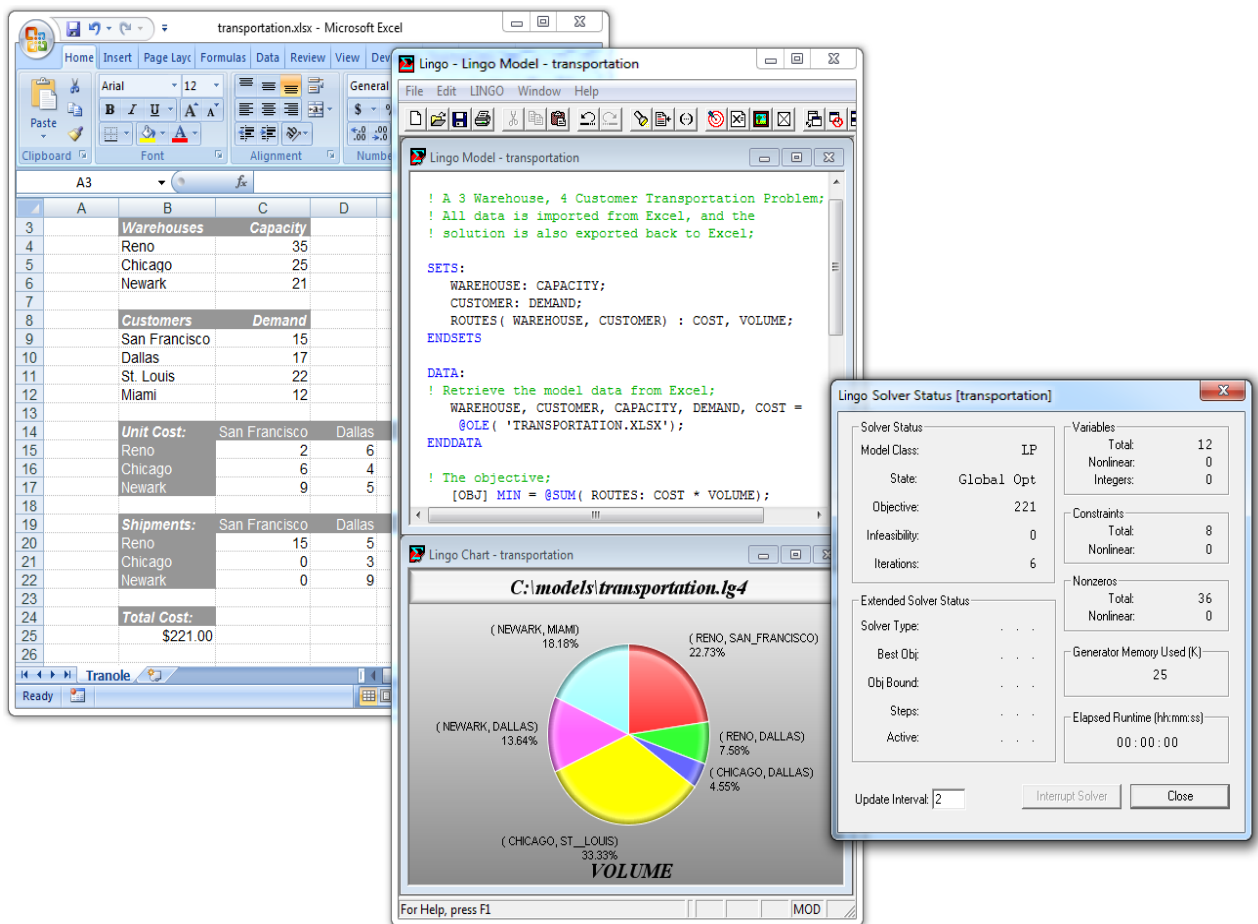


Рисунок 1.1 – Програмне забезпечення Lingo [11]

Серед особливостей програмного забезпечення Lingo для розв'язання завдань дискретної оптимізації можна виділити такі [11]:

– простий модельний вираз: програмне забезпечення Lingo спрощує розробку оптимізаційних моделей та вирішення оптимізаційних задач, дозволяючи легко формулювати математичні завдання у зрозумілому і зручному для читання форматі. Мова моделювання Lingo дає можливість виражати моделі інтуїтивно зрозумілим способом, використовуючи поняття сумування та змінні з індексами. Побудова та розуміння оптимізаційних моделей завдяки Lingo виконуються досить просто. Програмне забезпечення Lingo може використовувати кілька ядер центрального процесора для швидкої генерації моделей [11];

– зручне управління даними: програмне забезпечення Lingo виключає необхідність витрачати час і зусилля на управління даними. Це дозволяє створювати моделі, які отримують дані безпосередньо з баз даних і електронних таблиць. Також програмне забезпечення Lingo може виводити інформацію про рішення безпосередньо в базу даних або електронну таблицю, що робить процес створення звітів для розв’язуваної задачі дискретної оптимізації набагато зручнішим [11];

– Lingo володіє повним набором вбудованих швидких моделей для розв’язання різних видів оптимізаційних задач, включаючи лінійну, нелінійну, квадратичну, обмежену, конічну, стохастичну та дискретну оптимізацію [11];

– розширена документація та довідка: програмне забезпечення Lingo забезпечує необхідні інструменти для швидкого початку роботи з програмою. Довідка програмного забезпечення містить докладний текст про особливості моделювання, де розглядаються всі основні типи оптимізаційних задач, включаючи лінійні, цілочисельні та нелінійні. Також користувачам надається велика кількість реальних прикладів, які можна модифікувати та розширити для розв’язання конкретної задачі [11];

– засоби інтерактивного моделювання та створення готових застосунків: користувачу надається можливість створювати моделі та вирішувати практичні завдання за допомогою Lingo або викликати програмні модулі Lingo з зовнішньої програми. допомогою надає повне середовище для моделювання, де можна створювати, розв’язувати та аналізувати моделі в інтерактивному режимі. Для створення готових рішень Lingo має інтерфейси, які можна викликати з користувацьких програм. Крім того, існує можливість взаємодіяти з Lingo безпосередньо через макроси Excel або програми підтримки баз даних [11].

Серед недоліків програмного забезпечення Lingo можна відзначити те, що це програмне забезпечення є комерційним програмним продуктом, і вартість ліцензії може бути досить високою, що може ускладнювати

доступність програми для невеликих компаній та індивідуальних користувачів. Крім того, для користувачів, що не є фахівцями з розв'язання оптимізаційних завдань, може знадобитися певний час, щоб освоїти Lingo і ознайомитися з усіма його функціями. Деякі користувачі можуть відчувати певну складність при створенні та розв'язанні моделей за допомогою цього програмного забезпечення [11].

Програмне забезпечення TSPSG (рис. 1.2) розроблено для створення та вирішення задач комівояжера [12]. Програмне забезпечення TSPSG використовує метод розгалужень та границь для пошуку оптимального розв'язку. Вхідними даними для програмного забезпечення TSPSG є кількість міст та матриця вартостей подорожей між ними. Матрицю вартостей подорожей між містами для розв'язання завдання комівояжера за допомогою програмного забезпечення TSPSG можна заповнити випадковими значеннями заздалегідь встановленого діапазону (це може бути корисним для створення тестових завдань). Результатом роботи програмного забезпечення TSPSG є знайдений оптимальний маршрут, його вартість, докладні матриці кроків розв'язання та графік вирішення. Задачі комівояжера можуть бути збережені у внутрішньому бінарному форматі програмного забезпечення TSPSG та відкриті для перегляду пізніше. Отриманий результат можна надрукувати або зберегти у форматах PDF, HTML або ODF [12].

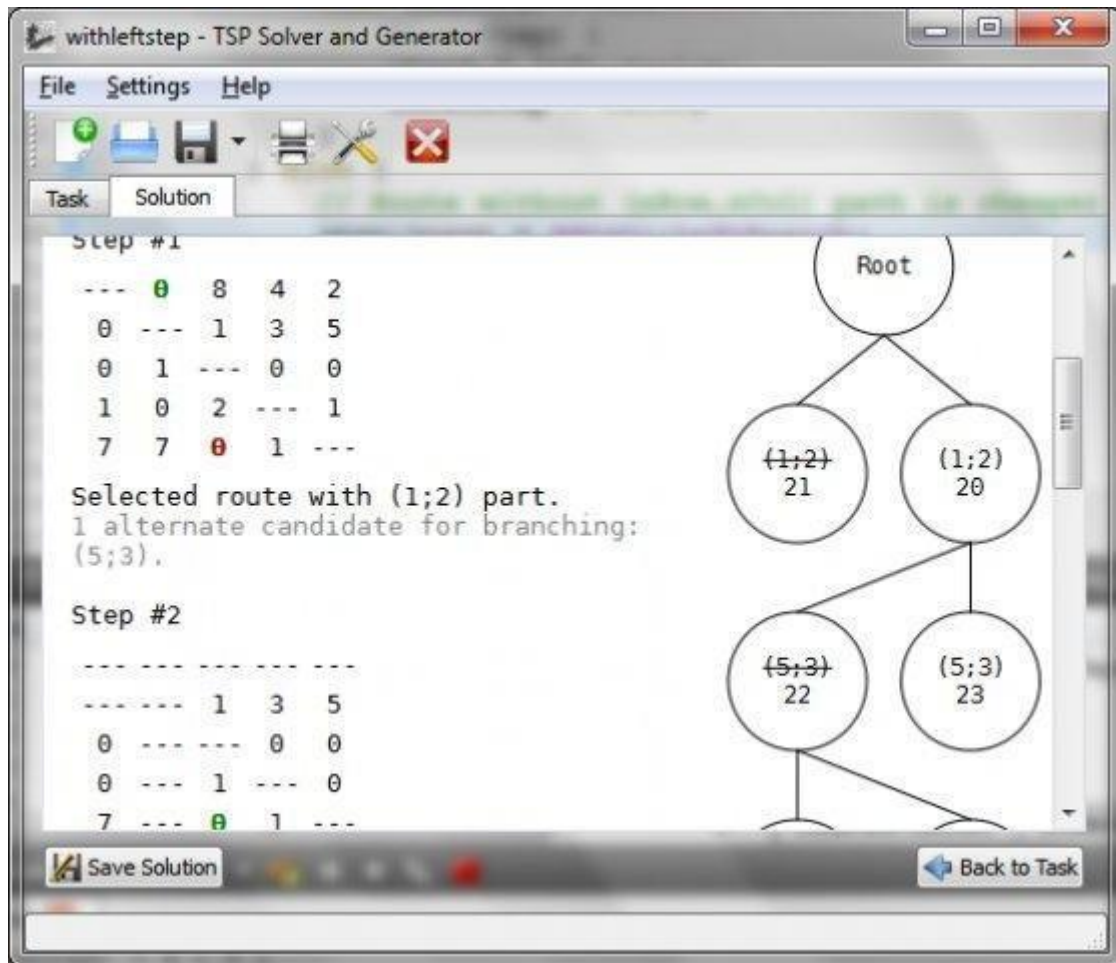


Рисунок 1.2 – Програмне забезпечення TSPSG [12]

Програмне забезпечення TSPSG може бути корисним для транспортних організацій при створенні оптимальних маршрутів. Крім того, ця програма може використовуватися створення тестових завдань, а також служити прикладом використання методу розгалуження та границь для вирішення конкретних завдань [12].

Серед особливостей програмного забезпечення TSPSG для розв'язання задачі комівояжера можна виділити такі [12]:

- зручні засоби введення вхідних даних. Користувач програмного забезпечення TSPSG може ввести кількість міст та матрицю вартостей подорожей між ними як вхідні дані для вирішення задачі комівояжера. Матриця може бути заповнена випадковими значеннями у встановленому діапазоні для створення випадкових завдань [12];

- розв'язання задач комівояжера: програмне забезпечення TSPSG спеціалізується на генерації та розв'язанні задач комівояжера, що відвідує всі задані міста і повертається до початкового пункту шляхом проходження найкоротшого маршруту [12];

- використання методу розгалуження та границь для ефективного пошуку оптимального розв'язку задачі комівояжера [12];

- засоби подання результатів: програмне забезпечення TSPSG надає користувачу інформацію про оптимальний маршрут, його вартість та покрокові матриці розв'язання для детального аналізу [12];

- збереження та відновлення завдань: конкретні завдання комівояжера можна зберегти у внутрішньому бінарному форматі та відкрити для перегляду або вирішення в майбутньому [12];

- експорт результатів: результати розв'язання завдання комівояжера можна надрукувати або зберегти у різних форматах, включаючи PDF, HTML та ODF [12];

- підтримка багатьох мов: програмне забезпечення TSPSG підтримує різні мови, що робить його доступним для користувачів з різних країн та мовних середовищ [12].

Загалом, програмне забезпечення TSPSG є корисним інструментом для вирішення задач комівояжера з можливістю створення, аналізу та збереження завдань дискретної оптимізації [12].

Серед недоліків програмного забезпечення TSPSG можна відзначити те, що програма обмежена у розв'язанні задач комівояжера з певною кількістю міст (до 50). Це може бути недостатньо для вирішення складних задач, де кількість міст значно більша. Іншим недоліком програмного забезпечення є те, що TSPSG спеціалізується на розв'язанні задачі комівояжера і може бути важко застосовуваним при розв'язанні інших задач дискретної оптимізації або аналізу даних. Крім того, програма TSPSG є платформи залежною, що означає,

що вона може працювати лише на певних операційних системах або середовищах [12].

Програмне забезпечення OptimoRoute (рис. 1.3) призначено для ефективного планування та оптимізації маршрутів доставки шляхом розв'язання завдань дискретної оптимізації. Програмне забезпечення OptimoRoute, вирішуючи завдання комівояжера, допомагає автоматично складати оптимальні маршрути для водіїв доставки або кур'єрів, щоб забезпечити ефективну доставку товарів чи послуг [13]. Програмне забезпечення OptimoRoute дозволяє прогнозувати попит та масштабувати кількість автомобілів відповідно до обсягу роботи та історичних даних [13].

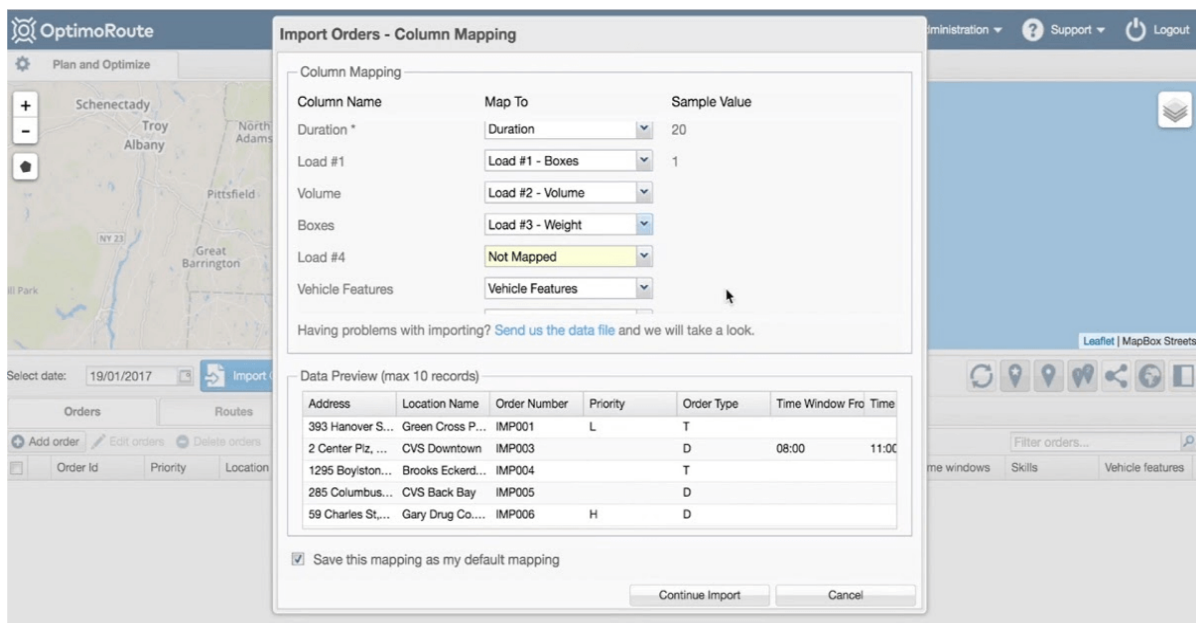


Рисунок 1.3 – Програмне забезпечення OptimoRoute [13]

Програмне забезпечення OptimoRoute дозволяє виконувати масовий імпорт замовлень: користувач може легко імпортувати списки замовлень з електронних таблиць Excel або файлів CSV, щоб швидко включити їх у планування [13]. Програма OptimoRoute дозволяє враховувати різноманітні обмеження, такі як час доставки, графік роботи водіїв, зони обслуговування, що робить планування більш точним і ефективним [13]. Програмне

забезпечення OptimoRoute надає історичні дані та аналітику, що допомагають покращити ефективність доставки та оптимізувати процеси. OptimoRoute допомагає зменшити витрати на логістику, збільшити продуктивність водіїв та знизити час доставки. Програма OptimoRoute робить парк автомобілів більш гнучким і дозволяє диспетчерам адаптуватися до змін у реальному часі [13].

Серед особливостей програмного забезпечення OptimoRoute для розв'язання задачі комівояжера можна виділити такі [13]:

- автоматичне планування маршрутів: програмне забезпечення OptimoRoute дозволяє автоматично створювати оптимальні маршрути на основі введених замовлень та обмежень [13];

- масштабування операцій: програма допомагає прогнозувати попит та масштабувати парк автомобілів відповідно до обсягу роботи та історичних даних [13];

- засоби імпорту даних: програмне забезпечення OptimoRoute надає можливість імпортувати списки замовлень з електронних таблиць Excel або файлів CSV [13];

- врахування обмежень: програмне забезпечення OptimoRoute бере до уваги різноманітні обмеження при розв'язанні задачі комівояжера, такі як час доставки, графік роботи водіїв, зони обслуговування та інші, щоб максимально точно планувати маршрути [13];

- аналітика та звітність: програма OptimoRoute надає доступ до історичних даних та аналітики, що допомагають покращити процеси та приймати обґрунтовані рішення [13];

- ефективне використання ресурсів: програмне забезпечення OptimoRoute допомагає зменшити витрати на логістику, збільшити продуктивність водіїв та оптимізувати час доставки [13];

- гнучкість та адаптація: програма OptimoRoute робить парк автомобілів більш гнучким та дозволяє диспетчерам адаптуватися до змін у реальному часі [13];

– зменшення залежності від ручного планування: програмне забезпечення OptimoRoute допомагає знизити потребу у великій кількості диспетчерів для ручного планування маршрутів [13];

– багатомовність: програма підтримує можливість роботи з різними мовами, що робить її доступною для користувачів із різних регіонів [13];

– інтеграція з іншими програмами: програмне забезпечення OptimoRoute може інтегруватися з іншими програмами та системами, що спрощує обмін даними та використання програми в складних системах [13].

Серед недоліків програмного забезпечення OptimoRoute можна відзначити те, що ця програма є комерційною, і вартість її використання може бути досить високою для малих підприємств з обмеженим бюджетом. Крім того, для ефективного використання OptimoRoute може знадобитися час на вивчення програми та її налаштування під конкретні потреби компанії [13].

Порівняльну характеристику програмного забезпечення для дискретної оптимізації наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для дискретної оптимізації

Критерій порівняння	Lingo [11]	TSPSG [12]	OptimoRoute [13]
Необхідність розуміння базових принципів програмування	+	+	–
Зручність розв’язання завдань комівояжера	+–	+	+
Можливість врахування додаткових умов при розв’язанні оптимізаційних завдань	+–	+	–
Засоби інтеграції з зовнішнім програмним забезпеченням	+	–	–

Можливість використання багатьох ядер процесора для синтезу оптимізаційних моделей	+	+	–
--	---	---	---

За результатами проведеного аналізу можна зробити висновок, що у наш час існує досить багато програмних засобів для розв’язання завдань дискретної оптимізації. Проте деякі програми для дискретної оптимізації можуть бути досить дорогим або обмежено доступним, що може бути важливим фактором для більш широкого використання. Інші програми дискретної оптимізації, зокрема, розв’язання задачі комівояжера, можуть не враховувати деяких обмежень, зокрема, час доставки, графік роботи водіїв, зони обслуговування та інші. Деякі програми можуть використовувати неефективні алгоритми, деякі використовувані моделі можуть враховувати ідеальні умови, які не завжди відображають реальність, що у деяких випадках може призвести до отримання некоректних результатів. Крім того, деякі оптимізаційні алгоритми, що можуть використовуватися у програмних засобах дискретної оптимізації, можуть вимагати значних обчислювальних ресурсів, що, у свою чергу, може вимагати великого часу та обчислювальних ресурсів для вирішення практичних задач. Тому актуальною є розробка програмного забезпечення для розв’язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації.

При розробці програмного забезпечення для розв’язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації необхідно забезпечити такі функціональні вимоги:

- можливість розв’язання задачі комівояжера;
- використання генетичних алгоритмів для розв’язання оптимізаційних завдань;
- можливість завантаження вхідних даних про пункти призначення комівояжера з текстових файлів;

- можливість зміни параметрів роботи генетичних алгоритмів;
- візуалізація побудованого оптимального шляху;
- виведення результатів дискретної оптимізації за допомогою генетичних алгоритмів у текстовому та графічному вигляді.

1.3 Висновки за розділом 1

Визначено, що у наш час широко використовуються програмні засоби для розв'язання завдань дискретної оптимізації, які допомагають знаходити оптимальні рішення в умовах обмежених ресурсів. Показано, що дискретна оптимізація є областю інформаційних технологій, яка фокусується на ефективному пошуку найкращих рішень серед обмеженого або визначеного набору можливих альтернатив. Проаналізовано завдання дискретної оптимізації. Відзначено, що дискретна оптимізація та відповідні програмні засоби знайшли широке використання у різних галузях економіки, проаналізовано практичні застосування програмних засобів дискретної оптимізації. Визначено, що важливою задачею дискретної оптимізації є задача комівояжера, що полягає у пошуку найкоротшого маршруту серед заданого списку міст і відстаней між ними. Проаналізовано особливості та проблеми, що виникають при роботі з реальними даними при розв'язанні задачі комівояжера. Досліджено методи розв'язання задачі комівояжера. Відзначено, що програмні засоби для розв'язання задачі комівояжера повинні надавати інтегроване планування маршрутів, щоб забезпечити вчасну та ефективну доставку посилок. Показано, що для розв'язання задачі комівояжера ефективно можуть використовуватися генетичні алгоритми дискретної оптимізації.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для дискретної оптимізації, зокрема, для розв'язання задачі комівояжера. Проте деякі програми для дискретної оптимізації можуть бути досить дорогим або обмежено доступним, що може

бути важливим фактором для більш широкого використання. Деякі програми можуть використовувати неефективні алгоритми, деякі використовувані моделі можуть враховувати ідеальні умови, які не завжди відображають реальність, що у деяких випадках може призвести до отримання некоректних результатів. Крім того, деякі оптимізаційні алгоритми, що можуть використовуватися у програмних засобах дискретної оптимізації, можуть вимагати значних обчислювальних ресурсів, що, у свою чергу, може вимагати великого часу та обчислювальних ресурсів для вирішення практичних задач. Тому актуальною є розробка програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації.

Сформульовано функціональні вимоги до програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування

При розробці програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації було використано мову програмування C++ [14], [15].

Мова C++ дозволяє використовувати різні підходи до програмування, такі як об'єктно-орієнтований стиль, процедурне програмування та загальне програмування. Крім того, вона наділяє розробників різними модулями налаштування, такими як обробка винятків, абстракція та робота з об'єктами даних, зокрема, з класами [14], [15].

Мова програмування C++ знайшла широке використання у сфері розробки програмного забезпечення та є однією із найбільш вживаних мов програмування. Вона застосовується для створення операційних систем, серверних сценаріїв, програмних додатків, драйверів пристроїв, вбудованих серверних рішень, високопродуктивних серверів та ігрових програм. У стандартних бібліотеках C++ містяться контейнери та алгоритми. Мова програмування C++ об'єднує в собі характеристики як мов високого, так і мов низького рівня [14], [15].

У порівнянні з іншими мовами, C++ має ряд важливих особливостей [14], [15]:

- це відкрита мова стандарту ISO C++, і її стандарт був затверджений комітетом ISO у 1998 році;
- вона підтримує як статичну, так і динамічну реєстрацію;
- C++ є мовою зі структурною організацією коду і вважається однією з найпотужніших мов програмування;
- вона надає можливість використання різноманітних методологій програмування;

– маючи високий ступінь вантажопідйомності, мова програмування C++ підтримується крос платформними компіляторами;

– ця мова сумісна з мовою C, що дозволяє використовувати бібліотеки C без потреби внесення змін у код бібліотеки [14], [15].

Мова програмування C++ також має певні недоліки, включаючи обмежену підтримку модулів, закриті процесори, нестрогі правила типізації, питання з безпекою та відсутність автоматичного звільнення пам'яті. Як і багато інших мов об'єктно-орієнтованого програмування, C++ підтримує три ключові концепції об'єктно-орієнтованого програмування: інкапсуляцію, успадкування та поліморфізм [14], [15].

Обґрунтування вибору мови програмування для розробки програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації наведено у таблиці 2.1.

Таблиця 2.1 – Обґрунтування вибору мови програмування для розробки програмного забезпечення для розв'язання завдання комівояжера

Критерій порівняння мов програмування	Мова програмування		
	C++	Matlab	Python
Відкритість коду	+	–	+
Велика спільнота розробників	+	–	+
Статична типізація	+	–	–
Ручне виділення пам'яті	+	–	–
Наявність бібліотек для вирішення задачі комівояжера	+	+	+-

Отже, для реалізації програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації обрано мову програмування C++, яка знайшла широке

використання у сфері розробки програмного забезпечення для дискретної оптимізації та є однією із найбільш вживаних мов програмування, крім того, ця мова дозволяє використовувати різні підходи до програмування, а також надає розробникам зручні інструменти для обробки винятків, абстракції та роботи з об'єктами даних, зокрема, з класами.

2.2 Вибір середовища розробки для створення програмного забезпечення для розв'язання задачі комівояжера

В якості середовища розробки для створення програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації було обрано Qt Creator [16], [17].

Для спрощення виконання проекту використовувався середовище розробки Qt Creator, що являє універсальний інструмент для створення програм, написаних на мові програмування C++. Середовище розробки Qt Creator дозволяє створювати програми для будь-якої операційної системи без необхідності змінювати код. Це середовище розробки легко компілює програми для більшості сучасних ОС. Qt Creator включає в себе всі основні класи, потрібні для розробки програмних застосунків, від елементів управління інтерфейсом користувача до мережевих баз даних. Ця бібліотека дозволяє об'єднувати різні програмні компоненти та отримувати доступ до файлів на різних платформах [16], [17].

Середовище розробки Qt Creator було обрано з декількох причин. По-перше, це середовище розробки має необхідні компоненти для розробки програми для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації. По-друге, це інтегрована середовище розробки є досить зручним для роботи [16], [17].

Середовище розробки Qt Creator є комплексним засобом розробки, розробленим для створення крос платформних застосунків на основі бібліотеки Qt [16], [17].

У середовище розробки Qt Creator включено графічний інтерфейс та інструменти для легкої створення візуальних інтерфейсів [16], [17].

Основна мета середовища розробки Qt Creator полягає в наданні допомоги у розробці застосунків, які працюють на різних платформах з використанням відповідного фреймворку [16], [17].

Середовище розробки програмного забезпечення Qt Creator має наступні функціональні можливості [16], [17]:

- включений редактор коду;
- постійне розширення можливостей;
- повний перегляд каталогів програмного проекту та підтримка багатьох мов програмування [16], [17].

Середовище розробки програмного забезпечення Qt Creator є інтегрованим засобом, призначеним для створення нових програм або модифікації вже існуючих [16], [17].

Середовище розробки Qt Creator забезпечує єдине середовище, що спрощує процес створення програм, оскільки воно об'єднує в собі різноманітні інструменти та ресурси. Розширена підтримка різних мов програмування: середовище розробки Qt Creator підтримує різноманітні мови програмування, такі як C++, Python та інші, що робить його універсальним інструментом для розробників [16], [17].

Середовище розробки програмного забезпечення Qt Creator є універсальним засобом для різних платформ. За допомогою Qt Creator розробник можете створювати крос платформні програми, що означає, що розроблюване програмне забезпечення буде працювати на різних операційних системах без необхідності внесення у створюване програмне забезпечення значних змін [16], [17].

Обґрунтування вибору середовища розробки для створення програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації наведено у таблиці 2.2.

Таблиця 2.2 – Обґрунтування вибору середовища розробки для створення програмного забезпечення для розв'язання задачі комівояжера

Критерій порівняння середовищ розробки	Середовище розробки		
	Qt Creator	Microsoft Visual Studio	C++Builder
Зручність розробки на мові C++	+	+-	+
Підтримка зовнішніх інструментів	+	+-	+-
Кросплатформність	+	+-	+-
Зручність для створення ПЗ для розв'язання задачі комівояжера	+	-	-
Вільна ліцензія	+	-	-

Таким чином, для створення програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації обрано середовище розробки Qt Creator, що являє собою універсальний інструмент для створення програм, написаних на мові програмування C++, дозволяє створювати програми для різних операційних систем без необхідності змінювати код та включає в себе основні класи, потрібні для розробки програмних застосунків, від елементів управління інтерфейсом користувача до роботи з базами даних.

2.3 Розробка моделі взаємодії користувача з програмним забезпеченням розв'язання задачі комівояжера

З метою можливості розробки програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації створимо відповідну модель.

При розробленні моделі взаємодії користувача з програмним забезпеченням для розв'язання задачі комівояжера враховуємо те, що користувачі програми мають можливості розв'язання задачі комівояжера, використання генетичних алгоритмів для розв'язання оптимізаційних завдань, завантаження вхідних даних про пункти призначення комівояжера з текстових файлів, зміни параметрів роботи генетичних алгоритмів, візуалізації побудованого оптимального шляху, виведення результатів дискретної оптимізації за допомогою генетичних алгоритмів у текстовому та графічному вигляді.

Основний успішний сценарій користувача при роботі з програмним забезпеченням для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації можна описати таким чином:

- користувач програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації запускає програму;

- після запуску програми для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації користувачу відображається головна форма;

- користувач програми для розв'язання задачі комівояжера бачить головну форму та завантажує файл з даними про пункти призначення комівояжера;

- користувач вводить параметри генетичного алгоритму для розв'язання задачі комівояжера;

- користувач здійснює розв’язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації шляхом натискання відповідної кнопки у програмі;
- користувач бачить результат роботи програми, зокрема карту з пунктами призначення та оптимальним шляхом;
- користувач завершує роботу з програмним забезпеченням для розв’язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Враховуючи сценарії роботи користувачів при роботі з програмним забезпеченням для розв’язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації, модель взаємодії користувача з програмним забезпеченням для розв’язання задачі комівояжера подамо у вигляді діаграми прецедентів, зображеної на рис. 2.1.

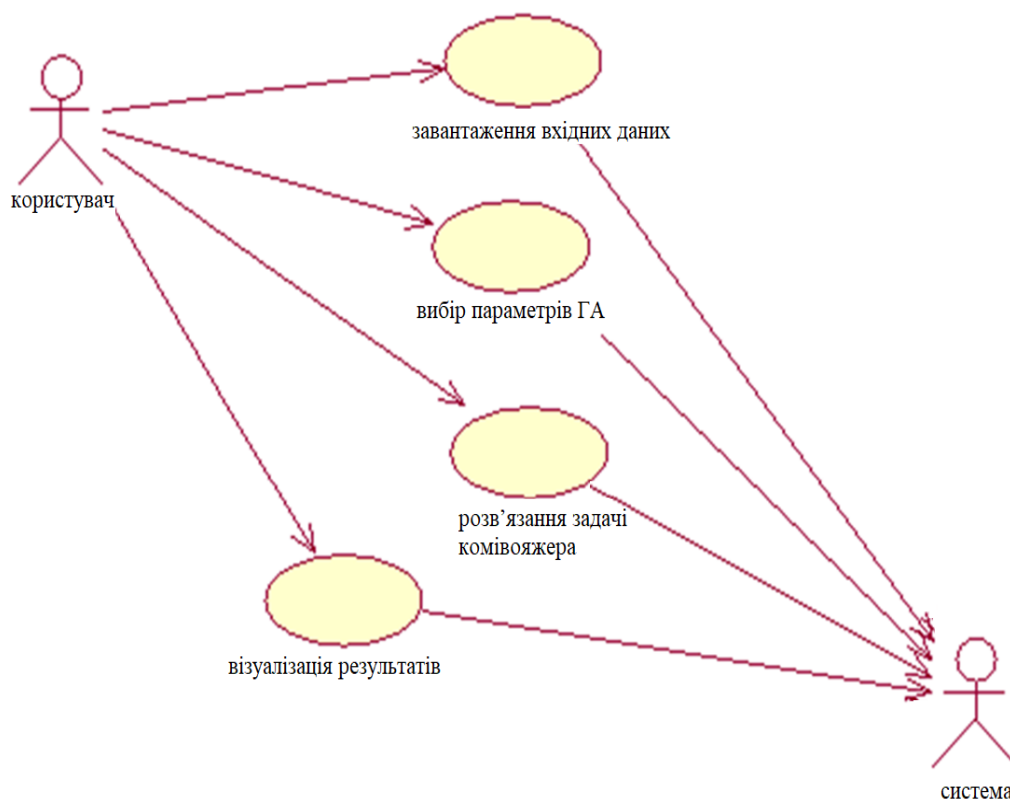


Рисунок 2.1 – Модель взаємодії користувача з програмним забезпеченням для розв’язання задачі комівояжера

Таким чином, створено модель взаємодії користувача з програмним забезпеченням для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації, яка подана у вигляді діаграми прецедентів, та дозволяє розроблювати програмне забезпечення для розв'язання задачі комівояжера.

2.4 Висновки за розділом 2

Для реалізації програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації обрано мову програмування C++, яка знайшла широке використання у сфері розробки програмного забезпечення для дискретної оптимізації та є однією із найбільш вживаних мов програмування, крім того, ця мова дозволяє використовувати різні підходи до програмування, а також надає розробникам зручні інструменти для обробки винятків, абстракції та роботи з об'єктами даних, зокрема, з класами.

Для створення програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації обрано середовище розробки Qt Creator, що являє собою універсальний інструмент для створення програм, написаних на мові програмування C++, дозволяє створювати програми для різних операційних систем без необхідності змінювати код та включає в себе основні класи, потрібні для розробки програмних застосунків, від елементів управління інтерфейсом користувача до роботи з базами даних.

Створено модель взаємодії користувача з програмним забезпеченням для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації, яка подана у вигляді діаграми прецедентів, та дозволяє розроблювати програмне забезпечення для розв'язання задачі комівояжера.

3 ОПИС ПРОГРАМИ

3.1 Структура програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації

Структуру програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації подано у вигляді діаграми класів (рис. 3.1).

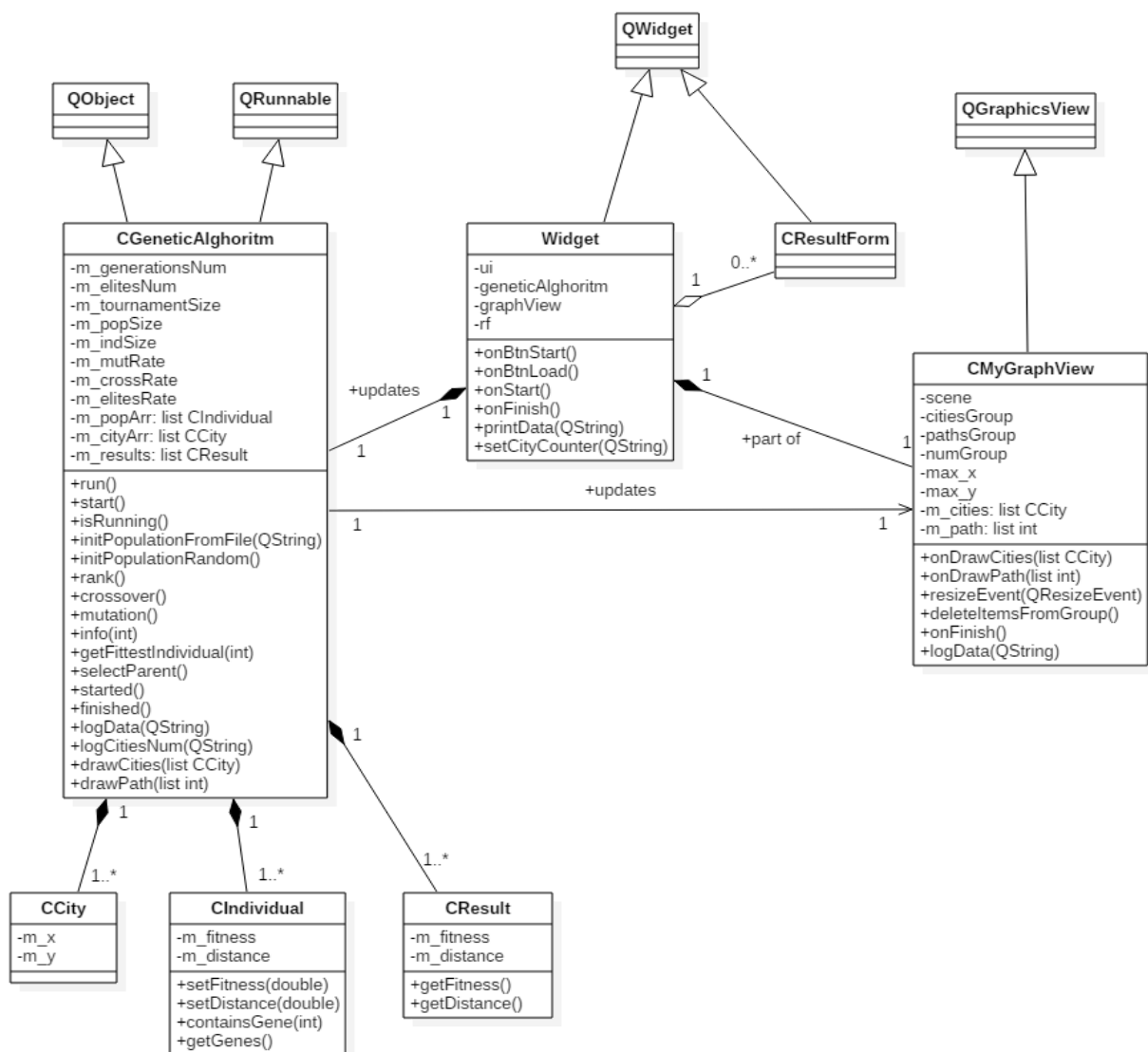


Рисунок 3.1 – Структура програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації

Як видно з рисунку, програмне забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації складається з таких основних класів: Генетичний алгоритм (GeneticAlgorithm), Графічне відображення (MyGraphView), Візуальний блок (Widget) та інші класи, які є похідними від вказаних. Також є класи, що визначають певні структури даних, зокрема, хромосоми (Individual), міста (City) та результати (Result).

3.2 Функціонування програмного забезпечення для розв'язання задачі комівояжера

Схема функціонування програмного забезпечення для розв'язання задачі комівояжера наведена на рис. 3.2.

На початку взаємодії користувача з програмою для розв'язання задачі комівояжера необхідно здійснити вибір параметрів та визначити вхідні дані. Серед цих параметрів важливо враховувати кількість ітерацій, розмір популяції, величину турніру, коефіцієнт елітності, операції схрещування та мутації. Крім того, користувач має можливість вибрати один із доступних методів для ініціалізації початкової популяції.

У якості вхідних даних програма для розв'язання задачі комівояжера очікує отримати текстовий файл із координатами міст, для яких потрібно розв'язати задачу комівояжера за допомогою генетичного алгоритму. Після завантаження цих даних та запуску алгоритму розпочнеться процес пошуку оптимального рішення. Варто відзначити, що пошук відбуватиметься у новому потоці, що дозволяє користувачу змінювати параметри налаштувань для нового запуску генетичного алгоритму.

Після завершення роботи генетичного алгоритму, користувач програмного забезпечення для розв'язання задачі комівояжера має можливість змінити параметри чи вхідні дані і продовжити роботу з програмою.

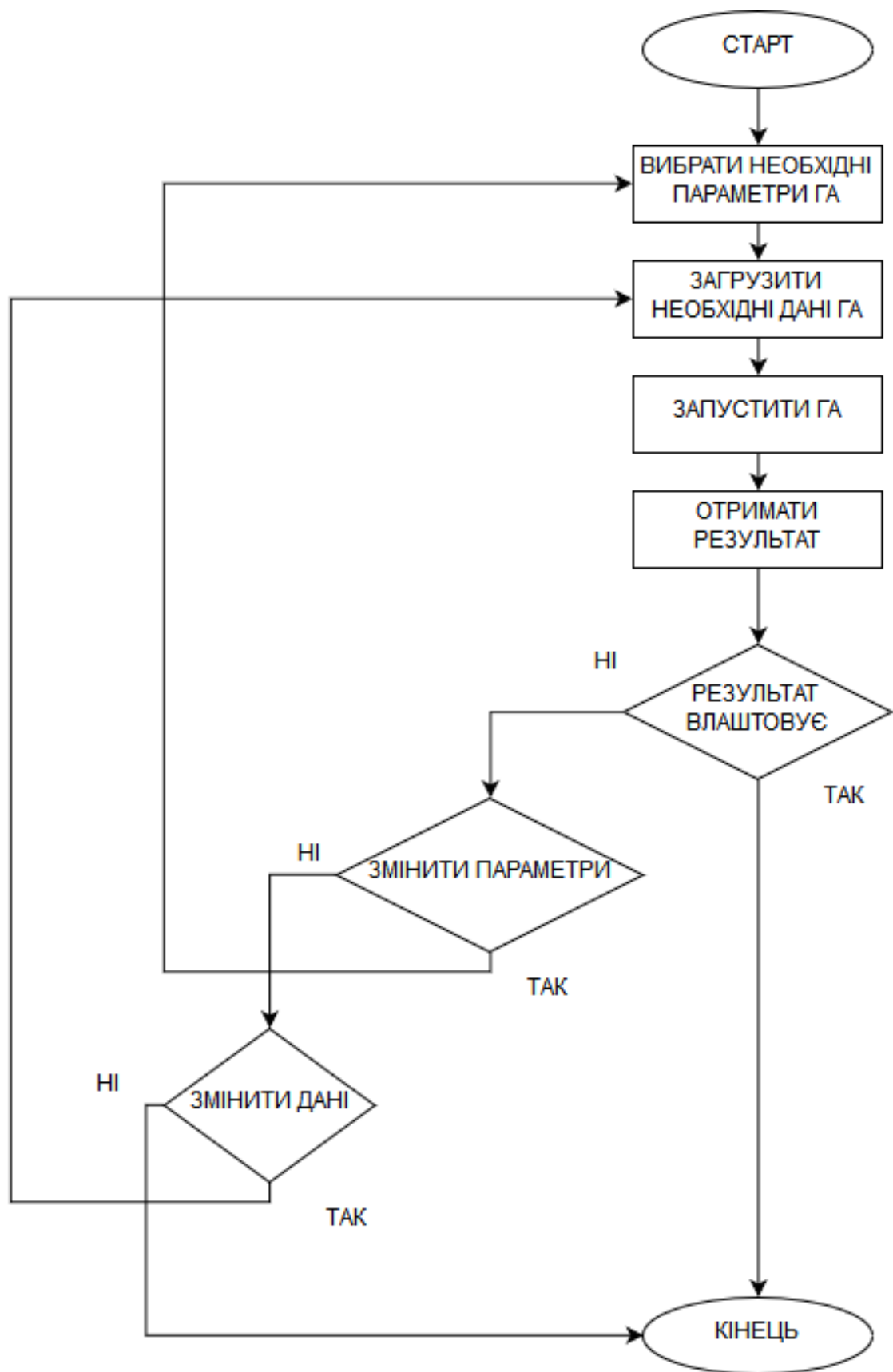


Рисунок 3.2 – Схема функціонування програмного забезпечення для розв’язання задачі комівояжера

3.3 Проектування програмного забезпечення для розв’язання задачі комівояжера

Загальна діаграма послідовностей програмного забезпечення для розв’язання задачі комівояжера наведена на рис. 3.3.

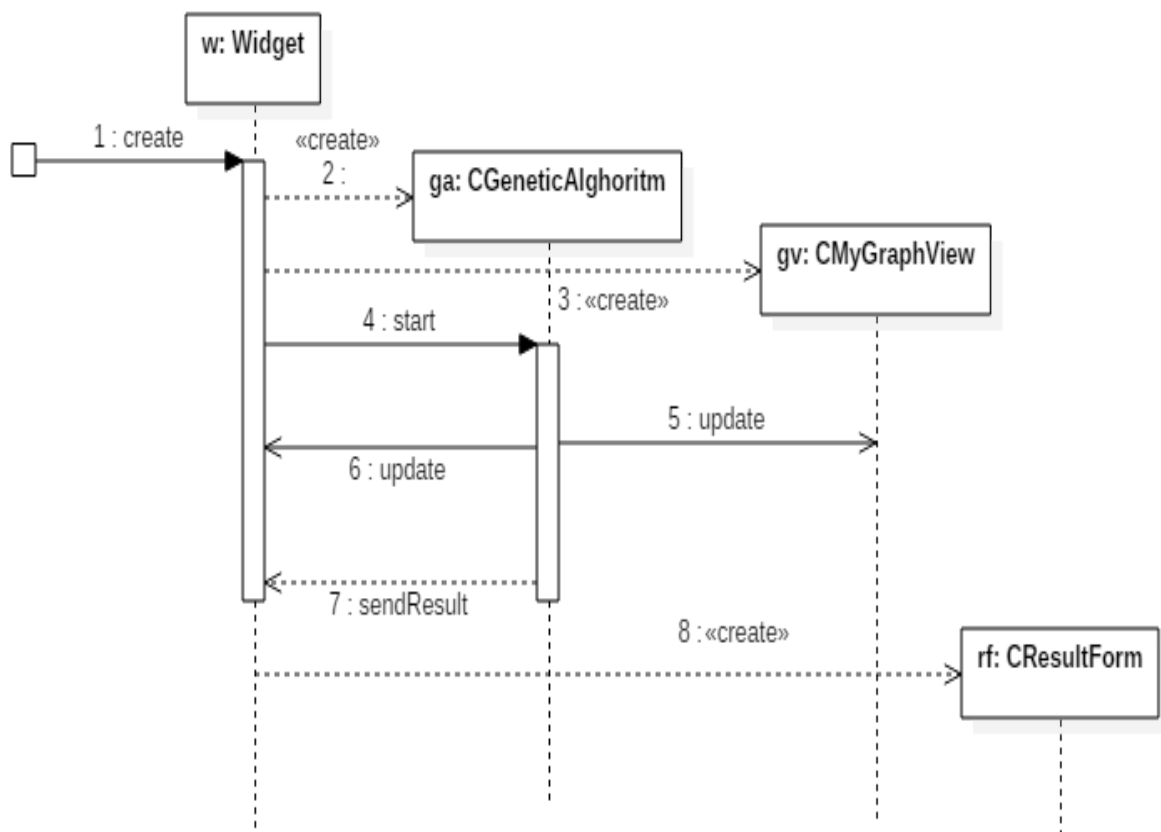


Рисунок 3.3 – Діаграма послідовностей програмного забезпечення для розв’язання задачі комівояжера

Як видно, спочатку створюється основне вікно програми, яке є екземпляром класу Widget, і також виконується на створення екземплярів класів GeneticAlhoritm і MyGraphView. Екземпляр GeneticAlhoritm

відповідає за виконання основної частини завдання з дискретної оптимізації та оновлення інтерфейсу програми. Після завершення своєї роботи він передає результати екземпляру класу Widget, що створює форму у вигляді екземпляра ResultForm для графічного відображення результатів.

Діаграма кооперацій програмного забезпечення для розв’язання задачі комівояжера наведена на рис. 3.4.

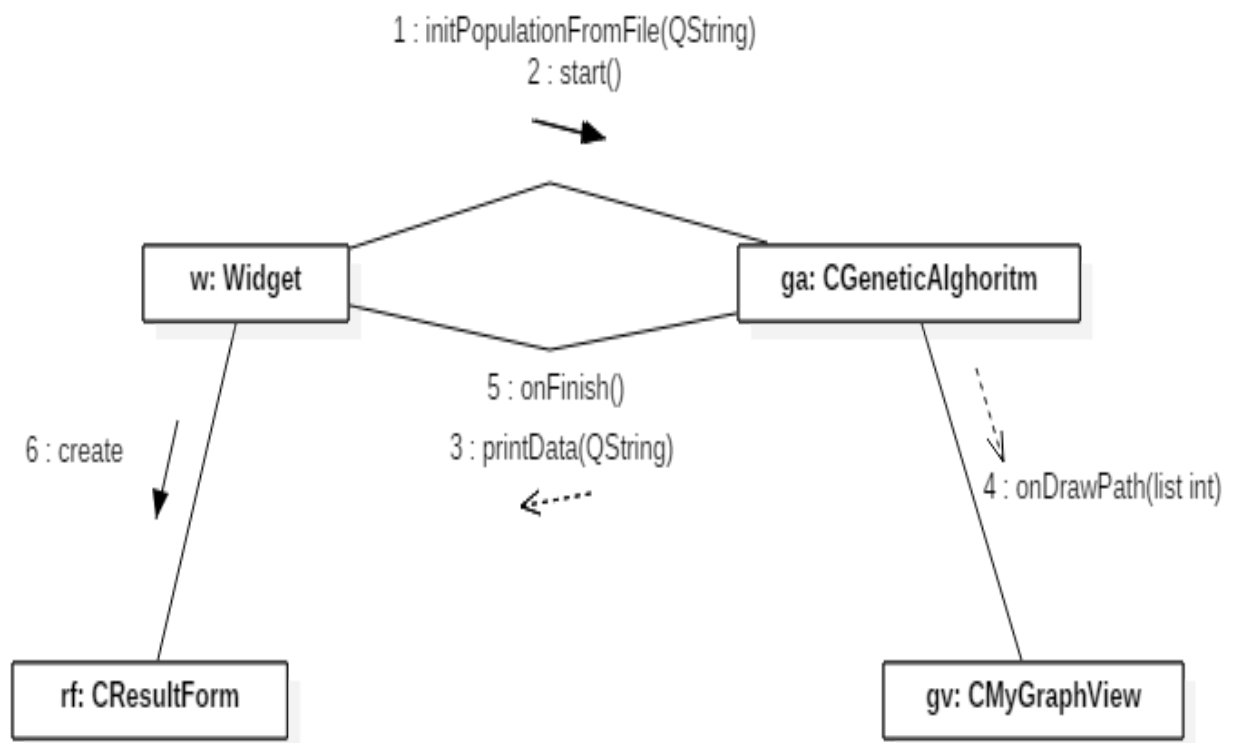


Рисунок 3.4 – Діаграма кооперацій програмного забезпечення для розв’язання задачі комівояжера

Графічна представлення, що наведено на рис. 3.4, є моментальним знімком процесу функціонування генетичного алгоритму, що виникає після запуску методу start(). На цьому зображенні видно різні етапи та події, такі як оновлення інтерфейсу (за допомогою функції printData), графічне відтворення роботи алгоритму (за допомогою функції onDrawPath), і представлення результатів обчислень (за допомогою функції create).

Діаграму станів програмного забезпечення для розв'язання задачі комівояжера наведена на рис. 3.5.

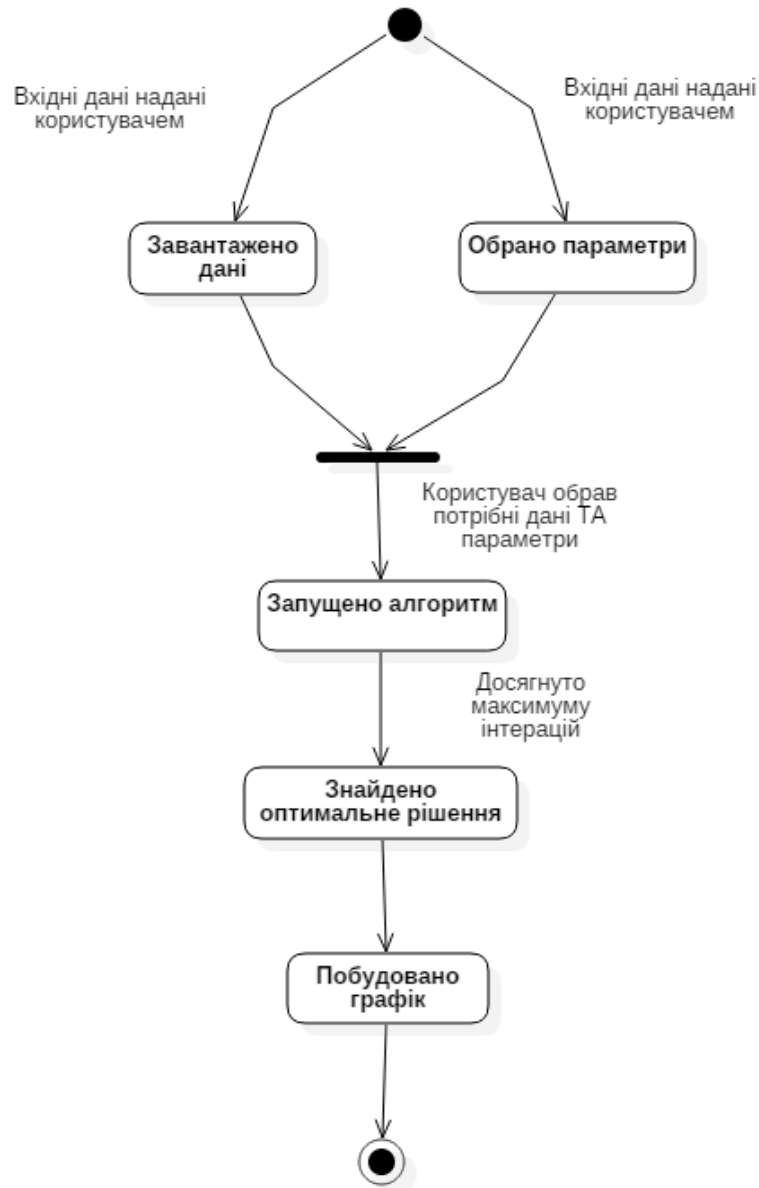


Рисунок 3.5 – Діаграма станів програмного забезпечення для розв'язання задачі комівояжера

На графіку станів відображено послідовність подій, що призводять до перетворення абстрактних значень та відношень у системі, інакше кажучи, її різних станів. Графік ілюструє, що для зміни певних станів, наприклад, для

активації алгоритму, потрібно, щоб система перебувала у двох попередніх станах.

Діаграму діяльності програмного забезпечення для розв'язання задачі комівояжера наведена на рис. 3.6.

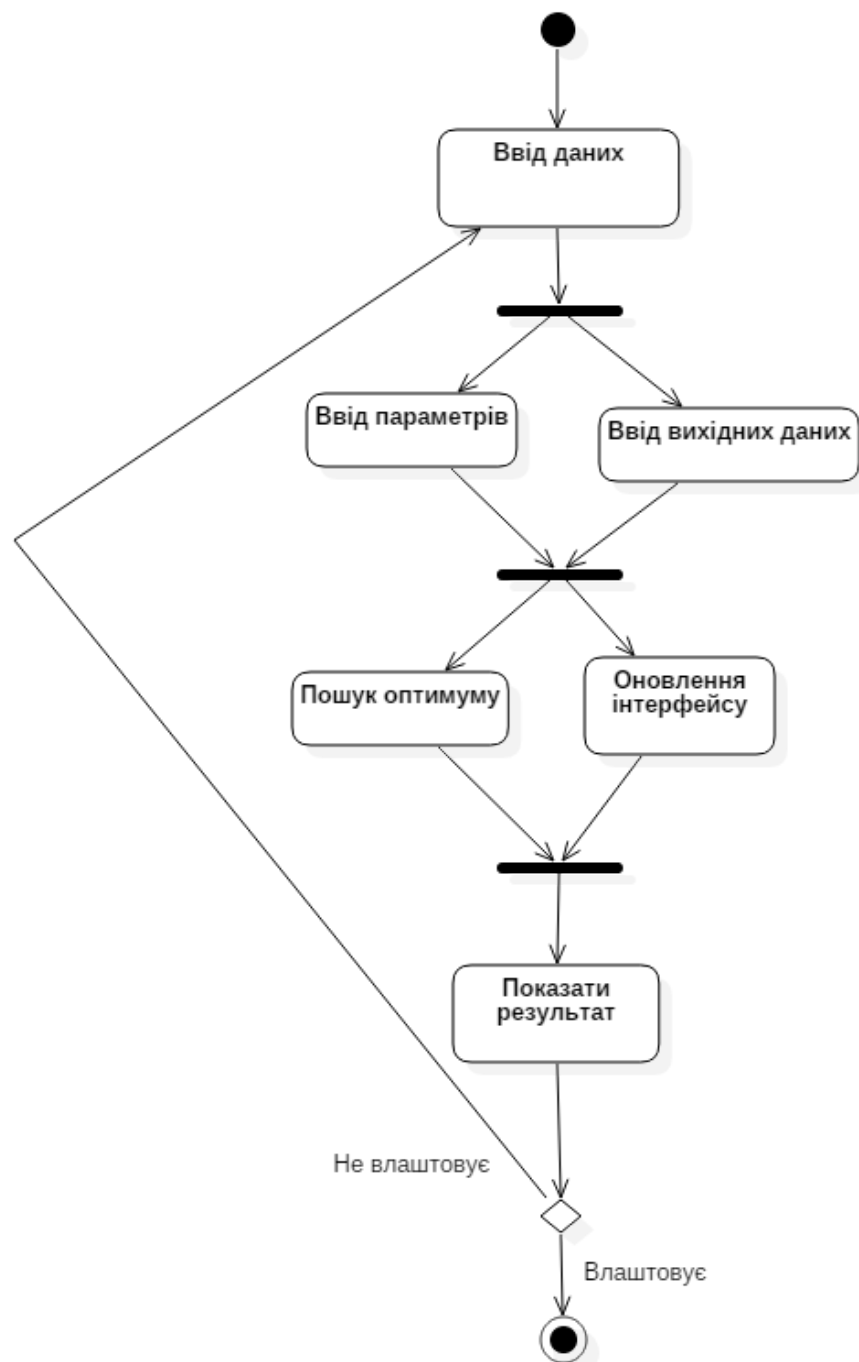


Рисунок 3.6 – Діаграма діяльності програмного забезпечення для розв'язання задачі комівояжера

На даній діаграмі активності відображена послідовність кроків, які ілюструють процес функціонування генетичного алгоритму. Ця діаграма ставить основний акцент на операціях та чітко демонструє потік управління. Операції можуть бути представлені одночасно.

3.4 Проєктування інтерфейсу програмного забезпечення для розв’язання задачі комівояжера

Проєктування інтерфейсу взаємодії користувача з програмним забезпеченням для розв’язання задачі комівояжера виконано з урахуванням вимог технічного завдання.

Форму головного вікна програмного забезпечення для розв’язання задачі комівояжера зображень наведено на рис. 3.7.

GA Parameters

Generations	1000	Elites rate	0.02
Population size	100	Crossover rate	0.9
Tournament size	5	Mutate rate	0.001

☐ Random population

☐ Optimal population

Data : 0 cities loaded.

Load data from file

Log console

Start Stop

Рисунок 3.7 – Форма головного вікна програмного забезпечення для розв’язання задачі комівояжера

Форму діалогового вікна вибору файлу з даними для розв’язання завдання комівояжера відображено на рис. 3.8.

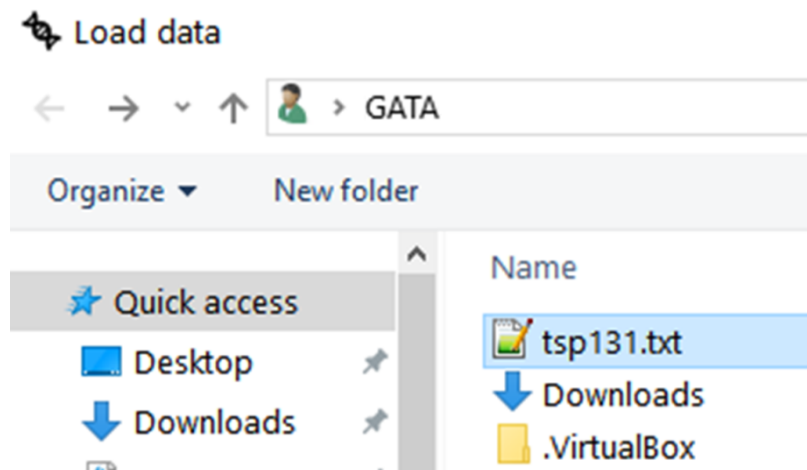


Рисунок 3.8 – Форма діалогового вікна вибору файлу з даними для розв’язання завдання комівояжера

Форму вікна з візуалізацією положення міст, які повинен пройти комівояжер, наведено на рис. 3.9.

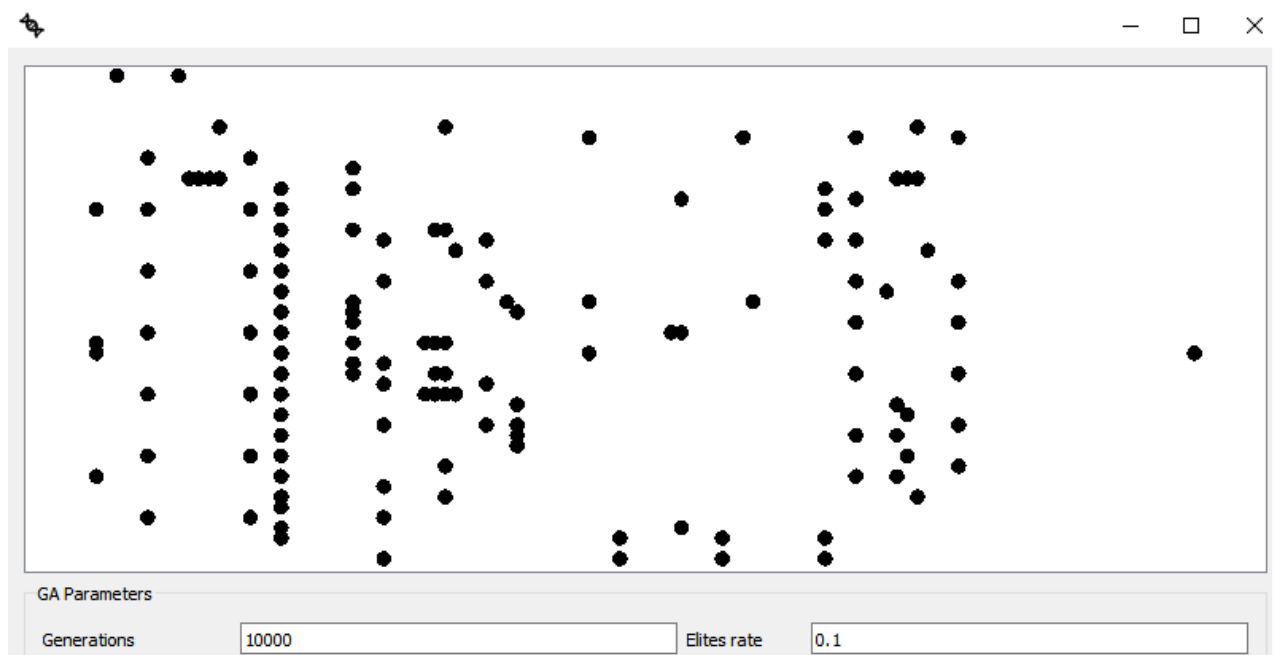


Рисунок 3.9 – Форма вікна з візуалізацією положення міст

Інтерфейс форми з візуалізацією знайденого оптимального шляху відображено на рис. 3.10.

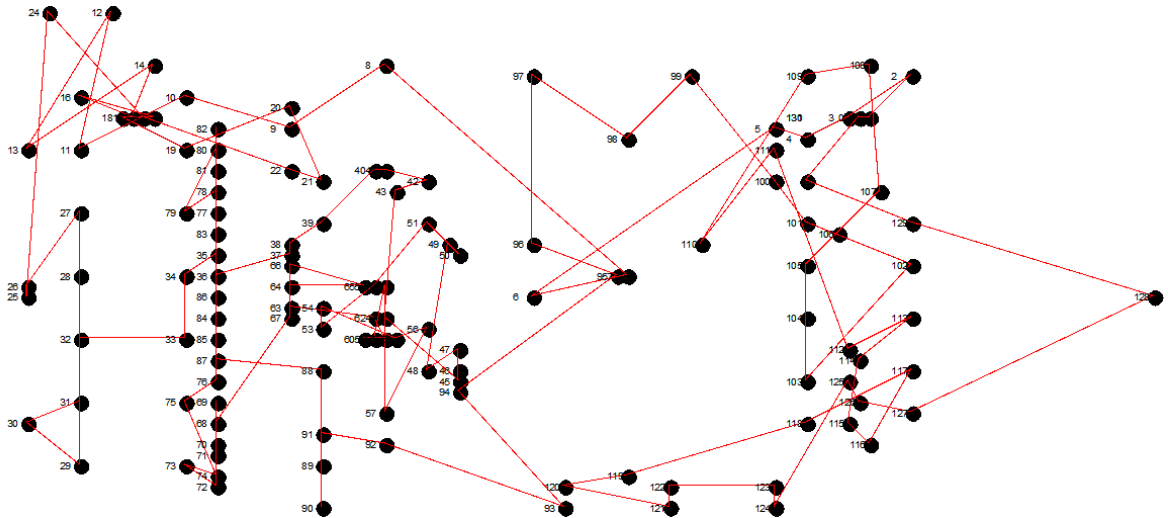


Рисунок 3.10 – Інтерфейс форми з візуалізацією знайденого оптимального шляху

Форму з візуалізацією результатів функціонування генетичного алгоритму відображено на рис. 3.11.

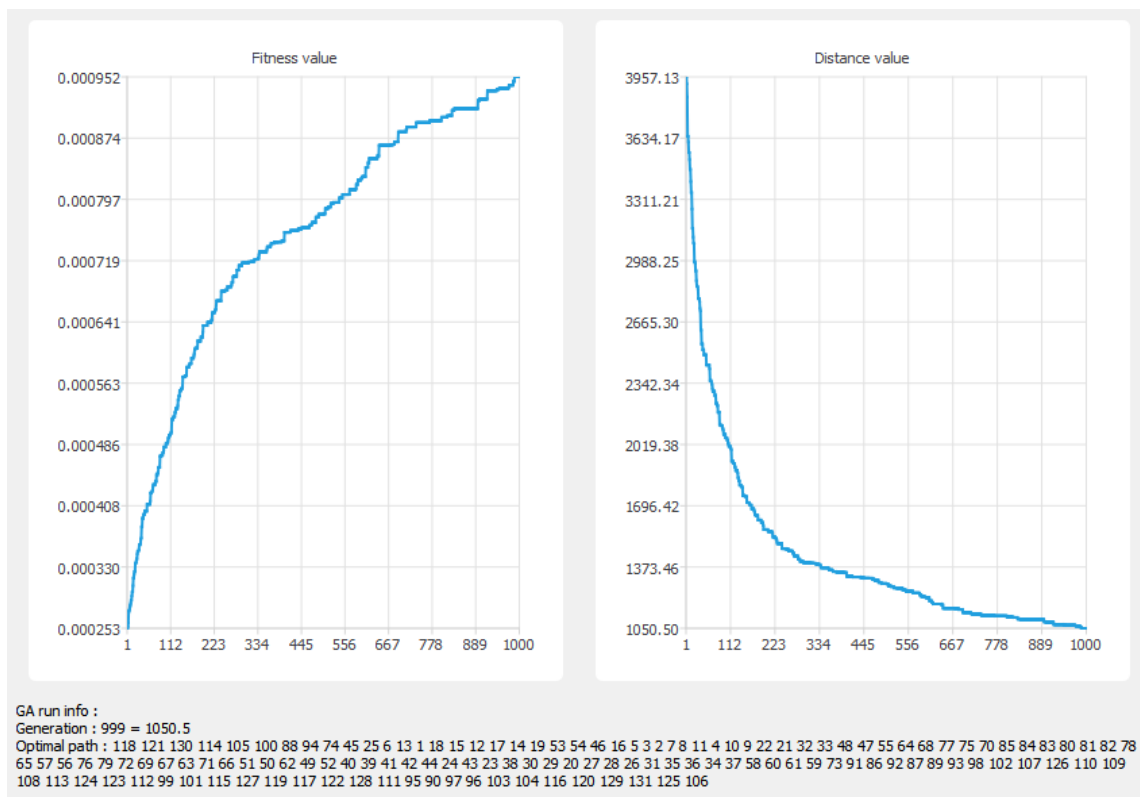


Рисунок 3.11 – Інтерфейс форми з візуалізацією результатів функціонування генетичного алгоритму

3.5 Висновки за розділом 3

Розроблено програмне забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації. Запропоновано структуру та описано основні модулі програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Описано функціонування програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації. Описано особливості реалізації програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації. Виконано проектування інтерфейсу взаємодії користувача з програмним забезпеченням для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення й умови застосування програми

Програма призначена для розв’язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації. Програма написана на мові C++, яка знайшла широке використання у сфері розробки програмного забезпечення для дискретної оптимізації та є однією із найбільш вживаних мов програмування, крім того, ця мова дозволяє використовувати різні підходи до програмування, а також надає розробникам зручні інструменти для обробки винятків, абстракції та роботи з об’єктами даних, зокрема, з класами. В якості середовища розробки для створення програми обрано Qt Creator, що являє собою універсальний інструмент для створення програм, написаних на мові програмування C++, дозволяє створювати програми для різних операційних систем без необхідності змінювати код та включає в себе основні класи, потрібні для розробки програмних застосунків, від елементів управління інтерфейсом користувача до роботи з базами даних.

Програмне забезпечення для розв’язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації забезпечує виконання таких функцій:

- можливість розв’язання задачі комівояжера;
- використання генетичних алгоритмів для розв’язання оптимізаційних завдань;
- можливість завантаження вхідних даних про пункти призначення комівояжера з текстових файлів;
- можливість зміни параметрів роботи генетичних алгоритмів;
- візуалізація побудованого оптимального шляху;
- виведення результатів дискретної оптимізації за допомогою генетичних алгоритмів у текстовому та графічному вигляді.

Для експлуатації програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять об'ємом 1Гб і більше;
- процесор із тактовою частотою 1,2 GHz і більше;
- вільне місце на диску 250 Мб або більше;
- встановлена операційна система.

Функціональні характеристики розробленого програмного забезпечення для розв'язання задачі комівояжера наведено у технічному завданні (додаток А). Фрагмент тексту програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації наведено у додатку Б.

4.2 Характеристики програми для розв'язання задачі комівояжера

Програмне забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації може працювати безперервно, в залежності від потреб користувача. Програмне забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації може використовуватись в будь-якій операційній системі. Для коректного функціонування програми для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації вміст директорії з програмним забезпеченням не можна змінювати.

4.3 Інструкція по експлуатації програми

4.3.1 Звернення до програми

Для запуску програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації

необхідно запустити головний файл, що знаходиться в кореневій директорії програми.

4.3.2 Вхідні й вихідні дані

Вхідними даними до програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації є файл з даними міст, які повинен обійти комівояжер, а також параметри генетичного алгоритму.

Вихідними даними програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації є оптимальний шлях комівояжера, а також графіки та текстова інформація, що відображають результати процесу функціонування генетичного алгоритму.

4.3.3 Повідомлення

Користувач програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації може отримати такі повідомлення: повідомлення про успішне завершення дискретної оптимізації з результатами, що містять послідовність оптимального шляху комівояжера.

4.4 Виконання програмного забезпечення для розв'язання задачі комівояжера

Після запуску програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації користувачу виводиться головна форма програми (рис. 4.1).

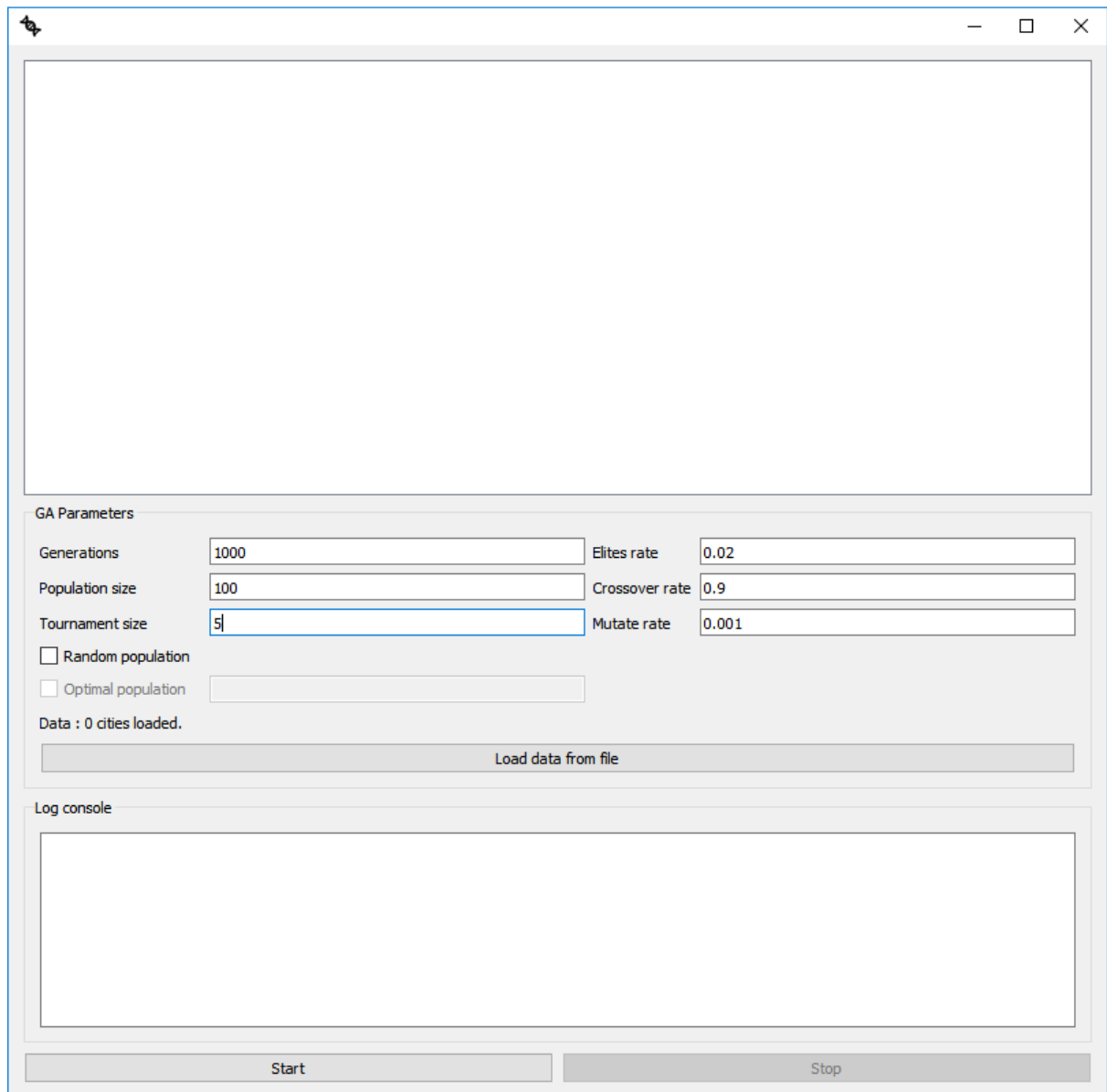


Рисунок 4.1 – Форма головного вікна програмного забезпечення для розв’язання задачі комівояжера

Як видно, користувач має можливість завантажити файлу з даними для розв’язання завдання комівояжера, обрати параметри генетичного алгоритму шляхом введення відповідних значень та отримати текстовий опис процесу генетичної оптимізації у відповідному полі головного вікна.

При натисканні на відповідну кнопку відображається форму діалогового вікна вибору файлу з даними для розв’язання завдання комівояжера (рис. 4.2).

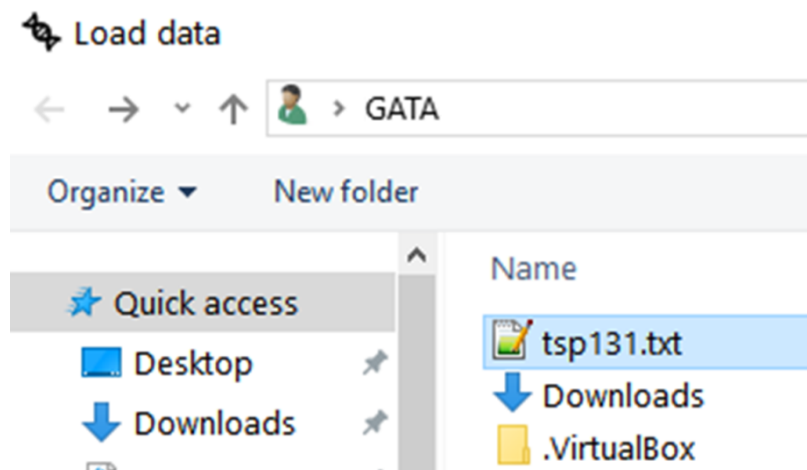


Рисунок 4.2 – Форма діалогового вікна вибору файлу з даними для розв’язання завдання комівояжера

Після вибору файлу з даними у головному вікні відображаються положення міст, які повинен пройти комівояжер (рис. 4.3).

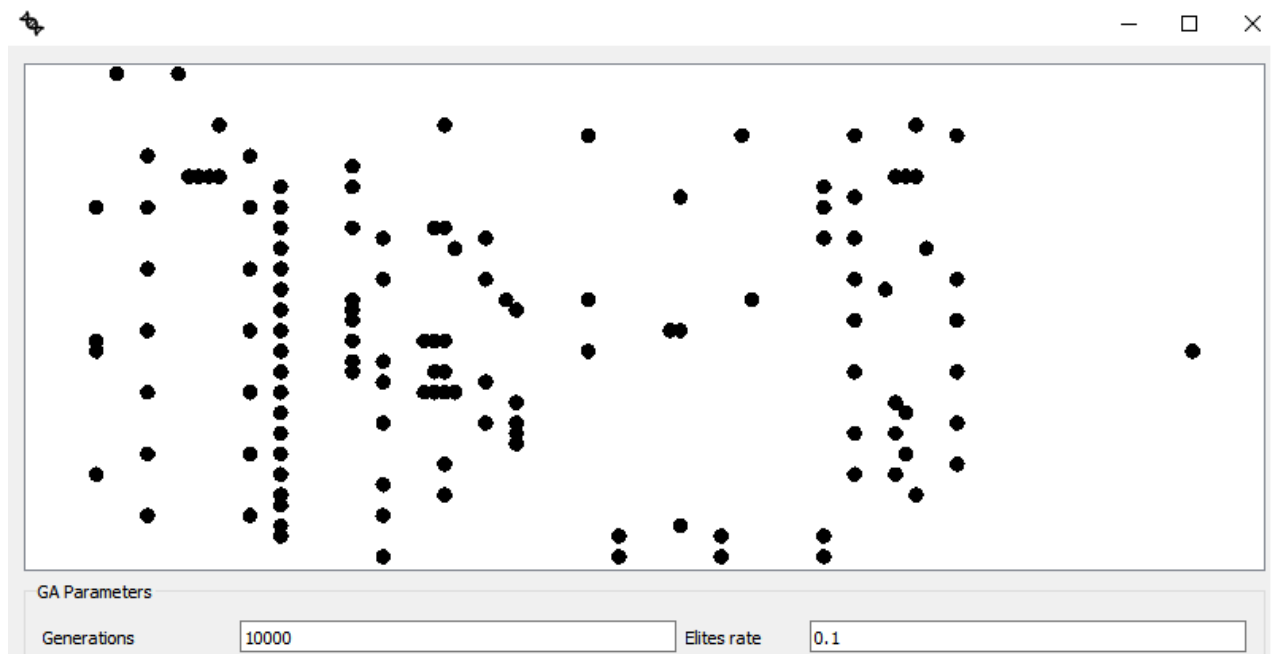


Рисунок 4.3 – Форма вікна з візуалізацією положення міст

Після натискання кнопки Start на головній формі розпочинається процес розв’язання задачі комівояжера за допомогою генетичного алгоритму.

Результатом генетичної оптимізації є знайдений оптимальний шлях. Інтерфейс форми з візуалізацією знайденого оптимального шляху відображено на рис. 4.4.

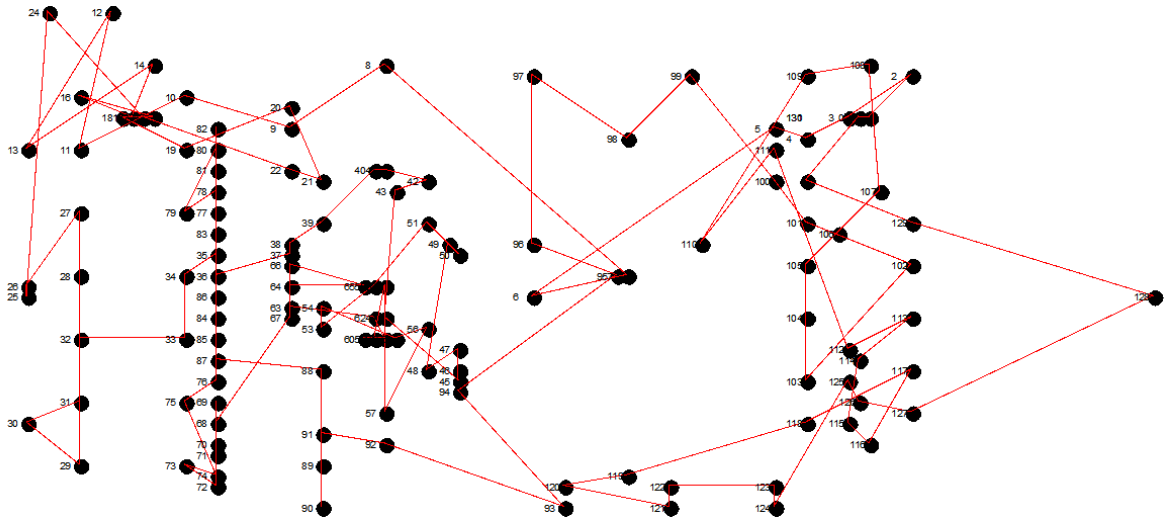


Рисунок 4.4 – Інтерфейс форми з візуалізацією знайденого оптимального шляху

Форму з візуалізацією результатів функціонування генетичного алгоритму відображено на рис. 4.5.

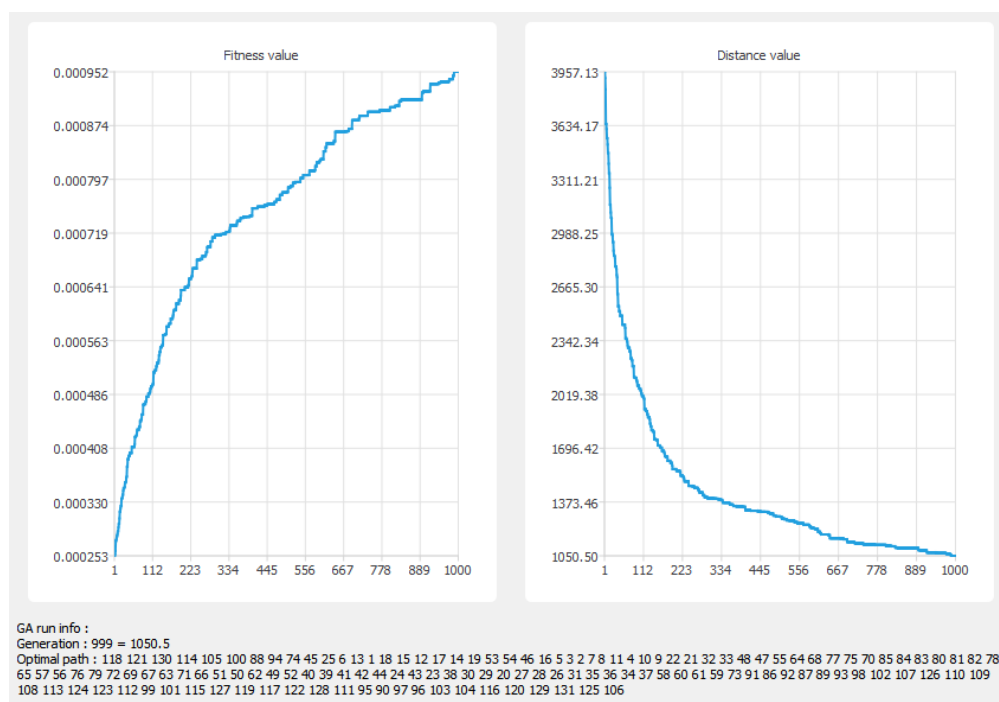


Рисунок 4.5 – Інтерфейс форми з візуалізацією результатів функціонування генетичного алгоритму

4.5 Тестування програмного забезпечення для розв’язання задачі комівояжера

Виконано тестування програмного забезпечення для розв’язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Виявлено, що програма для розв’язання задачі комівояжера функціонує правильно та злагоджено. Вона реалізує всі функціональні вимоги та успішно виконує свою основну задачу – розв’язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Розроблене програмне забезпечення дозволяє розв’язувати комівояжера за допомогою генетичних методів дискретної оптимізації.

4.6 Висновки за розділом 4

Описано програмне забезпечення для розв’язання задачі комівояжера. Виконано тестування розробленого програмного забезпечення для розв’язання задачі комівояжера. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє розв’язувати задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Висновки

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для дискретної оптимізації.

Визначено, що у наш час широко використовуються програмні засоби для розв'язання завдань дискретної оптимізації, які допомагають знаходити оптимальні рішення в умовах обмежених ресурсів. Показано, що дискретна оптимізація є областю інформаційних технологій, яка фокусується на ефективному пошуку найкращих рішень серед обмеженого або визначеного набору можливих альтернатив. Проаналізовано завдання дискретної оптимізації. Відзначено, що дискретна оптимізація та відповідні програмні засоби знайшли широке використання у різних галузях економіки, проаналізовано практичні застосування програмних засобів дискретної оптимізації. Визначено, що важливою задачею дискретної оптимізації є задача комівояжера, що полягає у пошуку найкоротшого маршруту серед заданого списку міст і відстаней між ними. Проаналізовано особливості та проблеми, що виникають при роботі з реальними даними при розв'язанні задачі комівояжера. Досліджено методи розв'язання задачі комівояжера. Відзначено, що програмні засоби для розв'язання задачі комівояжера повинні надавати інтегроване планування маршрутів, щоб забезпечити вчасну та ефективну доставку посилок. Показано, що для розв'язання задачі комівояжера ефективно можуть використовуватися генетичні алгоритми дискретної оптимізації.

За результатами проведеного аналізу зроблено висновок, що у наш час існує досить багато програмних засобів для дискретної оптимізації, зокрема, для розв'язання задачі комівояжера. Проте деякі програми для дискретної оптимізації можуть бути досить дорогим або обмежено доступним, що може бути важливим фактором для більш широкого використання. Деякі програми можуть використовувати неефективні алгоритми, деякі використовувані моделі

можуть враховувати ідеальні умови, які не завжди відображають реальність, що у деяких випадках може призвести до отримання некоректних результатів. Крім того, деякі оптимізаційні алгоритми, що можуть використовуватися у програмних засобах дискретної оптимізації, можуть вимагати значних обчислювальних ресурсів, що, у свою чергу, може вимагати великого часу та обчислювальних ресурсів для вирішення практичних задач. Тому актуальною є розробка програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації.

Сформульовано функціональні вимоги до програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації.

Для реалізації програмного забезпечення для розв'язання завдання комівояжера за допомогою генетичних методів дискретної оптимізації обрано мову програмування C++, яка знайшла широке використання у сфері розробки програмного забезпечення для дискретної оптимізації та є однією із найбільш вживаних мов програмування, крім того, ця мова дозволяє використовувати різні підходи до програмування, а також надає розробникам зручні інструменти для обробки винятків, абстракції та роботи з об'єктами даних, зокрема, з класами.

Для створення програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації обрано середовище розробки Qt Creator, що являє собою універсальний інструмент для створення програм, написаних на мові програмування C++, дозволяє створювати програми для різних операційних систем без необхідності змінювати код та включає в себе основні класи, потрібні для розробки програмних застосунків, від елементів управління інтерфейсом користувача до роботи з базами даних.

Новизна роботи полягає в тому, що запропоновано модель взаємодії користувача з програмним забезпеченням для розв'язання задачі комівояжера

за допомогою генетичних методів дискретної оптимізації, яка подана у вигляді діаграми прецедентів, та дозволяє розроблювати програмне забезпечення для розв'язання задачі комівояжера.

Практичне значення роботи полягає в тому, що розроблено програмне забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Запропоновано структуру та описано основні модулі програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Описано функціонування програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації. Описано особливості реалізації програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації. Виконано проектування інтерфейсу взаємодії користувача з програмним забезпеченням для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Виконано тестування розробленого програмного забезпечення для розв'язання задачі комівояжера. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє розв'язувати задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Discrete Optimization [Electronic resource]. – Access mode : <https://sites.math.washington.edu/~rothvoss/lecturenotes/DisOpt409-Spring2020.pdf>.
2. Discrete Optimization: Mathematics, Algorithms, and Computation [Electronic resource]. – Access mode : <https://icerm.brown.edu/programs/sp-s23/>.
3. Optimization-Based Planning [Electronic resource]. – Access mode : https://help.sap.com/doc/saphelp_scm700_ehp02/7.0.2/en-US/81/3ec95360267614e10000000a174cb4/content.htm.
4. A Practical Guide to Discrete Optimization [Electronic resource]. – Access mode : https://www.math.uwaterloo.ca/~bico/papers/comp_chapter1.pdf.
5. Explained: What is Traveling Salesman Problem [Electronic resource]. – Access mode : <https://www.upperinc.com/guides/travelling-salesman-problem/>.
6. Traveling salesman problem [Electronic resource]. – Access mode : https://optimization.cbe.cornell.edu/index.php?title=Traveling_salesman_problem.
7. The traveling salesman problem [Electronic resource]. – Access mode : <https://uwaterloo.ca/math/traveling-salesman-problem>.
8. Genetic algorithms for modelling and optimisation [Electronic resource]. – Access mode : <https://www.sciencedirect.com/science/article/pii/S0377042705000774>.
9. Genetic Algorithms - Introduction [Electronic resource]. – Access mode: https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_introduction.htm.
10. A review on genetic algorithm: past, present, and future [Electronic resource]. – Access mode : <https://link.springer.com/article/10.1007/s11042-020-10139-6>.

11. Lingo Optimization Modeling Software for Linear, Nonlinear, and Integer Programming [Electronic resource]. – Access mode : <https://www.lindo.com/index.php/products/lingo-and-optimization-modeling>.
12. TSP Solver and Generator [Electronic resource]. – Access mode : <https://sourceforge.net/projects/tspsg/>.
13. OptimoRoute [Electronic resource]. – Access mode : <https://optimoroute.com/travelling-salesman-problem/>.
14. C++ Compiler [Electronic resource]. – Access mode : <https://www.embarcadero.com/free-tools/ccompiler>.
15. Introduction to C++ Programming Language [Electronic resource]. – Access mode : <https://www.geeksforgeeks.org/introduction-to-c-programming-language/>.
16. Qt Creator IDE Review [Electronic resource]. – Access mode : <https://comparecamp.com/qt-creator-ide-review-pricing-pros-cons-features/>.
17. Qt Creator [Electronic resource]. – Access mode : <https://doc.qt.io/qtcreator/>.

Додаток А
Технічне завдання

Вступ

Програмне забезпечення може використовуватися для розв'язання задачі комівояжера.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Дослідження та програмна реалізація засобів розв'язання завдань дискретної оптимізації», затверджене наказом Національного університету «Запорізька політехніка» № 437 від 14 листопада 2023 р.

A.2 Призначення розробки

Програмний продукт призначений для забезпечення та підтримки процесу розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

A.3 Основні вимоги до програми, що розробляється

A.3.1 Вимоги до функціональних характеристик

Програмне забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації забезпечує виконання таких функцій:

- можливість розв'язання задачі комівояжера;
- використання генетичних алгоритмів для розв'язання оптимізаційних завдань;
- можливість завантаження вхідних даних про пункти призначення комівояжера з текстових файлів;

- можливість зміни параметрів роботи генетичних алгоритмів;
- візуалізація побудованого оптимального шляху;
- виведення результатів дискретної оптимізації за допомогою генетичних алгоритмів у текстовому та графічному вигляді.

A.3.2 Вимоги до інтерфейсу програми

Інтерфейс програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації повинен бути зручним для користувачів.

A.3.3 Вимоги до надійності

Програмне забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації **ПОВИННО** забезпечити надійне функціонування.

A.3.4 Умови експлуатації

Для експлуатації програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації необхідна наявність персонального комп'ютера у користувача.

A.3.5 Вимоги до складу та параметрів технічний засобів

Для експлуатації програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації необхідно таке апаратне та програмне забезпечення:

- оперативна пам'ять об'ємом 1Гб і більше;
- процесор із тактовою частотою 1,2 GHz і більше;
- вільне місце на диску 250 Мб або більше;
- встановлена операційна система.

А.3.6 Вимоги до маркування і пакування

Програма для розв'язання задачі комівояжера може бути записана на будь-якому носії інформації.

На пакуванні повинна бути назва програми – «Програмне забезпечення для розв'язання задачі комівояжера».

А.4 Вхідні дані до роботи

Вхідними даними до програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації є файл з даними міст, які повинен обійти комівояжер, а також параметри генетичного алгоритму.

Вихідними даними програмного забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації є оптимальний шлях комівояжера, а також графіки та текстова інформація, що відображають результати процесу функціонування генетичного алгоритму.

Додаток Б
Фрагмент тексту програми

```

Widget::Widget(QWidget *prnt) :
    QWidget(prnt),
    usrntrf(new Usrntrf::Widget)
{
    usrntrf->setupUsrntrf(this);
    usrntrf->btstp->setEnabled(false);
    lncalc = 0;
    rtf = nullptr;
    gntcalg = new CGAlgm();
    gqv = new Mgrvw();
    usrntrf->lGn->setText("500");
    usrntrf->lPp->setText("100");
    usrntrf->lTr->setText("5");
    usrntrf->lElt->setText("0.03");
    usrntrf->lCrs->setText("0.85");
    usrntrf->lMt->setText("0.001");
    usrntrf->vrtlt->insertWidget(0, gqv);

    connect(gntcalg, &CGAlgm::fnshd, gqv, &Mgrvw::onFinish);
    connect(gntcalg, &CGAlgm::fnshd, this, &Widget::onFinish);
        connect(usrntrf->btstrt, &Qpbtnclc::clicked, this,
&Widget::onBtnStart);
        connect(usrntrf->btstp, &Qpbtnclc::clicked, gntcalg,
&CGAlgm::stop);
    connect(this, &Widget::stopGNTCALG, gntcalg, &CGAlgm::stop);
        connect(usrntrf->btdtfl, &Qpbtnclc::clicked, this,
&Widget::onBtnLoad);
    connect(gntcalg, &CGAlgm::lgdt, this, &Widget::prndt);
    connect(gntcalg, &CGAlgm::lgctsnm, this, &Widget::stCtCntr);
    connect(gqv, &Mgrvw::lgdt, this, &Widget::prndt);
    connect(gntcalg, &CGAlgm::drcts, gqv, &Mgrvw::onDrcts);
    connect(gntcalg, &CGAlgm::drpth, gqv, &Mgrvw::onDrpth);

    connect(usrntrf->ckbxrnd, &QCheckBox::stateChanged, [=]() {
        bool chkd = usrntrf->ckbxrnd->isChecked();
        if(!chkd) {
            usrntrf->chkbxtm->setEnabled(false);
            usrntrf->chkbxtm->setChkd(false);
            usrntrf->lptnm->setEnabled(false);
            usrntrf->lptnm->clear();
        }
        else {
            usrntrf->chkbxtm->setEnabled(true);
            usrntrf->lptnm->setEnabled(true);
            usrntrf->lptnm->setText("200");
        }
    });
}

```

```

        showMaximized();
    }

Widget::~Widget()
{
    if(gntcalg->isRunning()) emit stopGNTCALG();
    QThreadPool::globalInstance()->cancel(gntcalg);
    delete usrntrf;
}

void Widget::onBtnStart()
{
    if(gntcalg->dataIsLoaded()){
        if(!gntcalg->isRunning()){
            if(lnclc > 0)
                gqv->clearSame();

            usrntrf->btdtfl->setEnabled(false);
            usrntrf->btstrt->setEnabled(false);
            usrntrf->btstp->setEnabled(true);

            gntcalg->stgnrnm(usrntrf->lGn->text().toInt());
            gntcalg->stppsz(usrntrf->lPp->text().toInt());
            gntcalg->sttrnmsz(usrntrf->lTr->text().toInt());

            gntcalg->stelnm(std::ceil(usrntrf->lElt->text().toDouble()) *
            gntcalg->getPpszrt());
            gntcalg->stmrtrt(usrntrf->lMt->text().toDouble());
            gntcalg->stcrtrt(usrntrf->lCrs->text().toDouble());

            bool randChkd = usrntrf->ckbxrnd->isChkd();
            gntcalg->strndpplt(randChkd);
            if(randChkd){
                gntcalg->stoptppn(usrntrf->chkbxtm->isChkd());
            }

            gntcalg->stoptppnClkst(usrntrf->lptnm->text().toInt());
        }

        QThreadPool::globalInstance()->start(gntcalg);
    }else{
        usrntrf->console->clear();
        prndt("GNTCALG is running ...");
    }
}else{
    usrntrf->console->clear();
    prndt("Дані не завантажено ...");
    return;
}

```

```

    }
}

void Widget::onBtnLoad()
{
    crpth = QFileDialog::getOpenFileName(this, "Завантаження
даних", QDir::homePath(), "Data Files (*.txt)");
    if(crpth.isEmpty()) return;
    usrntrf->lData->setText("... Дані завантажуються з [" + crpth
+ "]" with ["");
    if(lnclc > 0)
        gqv->clearNew();
    prndt("\nГенерація оптимального рішення...\n");
    bool randChkd = usrntrf->ckbxrnd->isChkd();
    gntcalg->strndpplt(randChkd);
    if(randChkd){
        gntcalg->stoptppn(usrntrf->chkbxtm->isChkd());
        gntcalg->stoptppnClkst(usrntrf->lptnm->text().toInt());
    }
    gntcalg->intppffl(crpth);
}

void Widget::prndt(QString pdata)
{
    usrntrf->console->append(pdata);
}

void Widget::stCtCntr(QString pdata)
{
    usrntrf->lData->setText(usrntrf->lData->text() + pdata + "]
cities ...");
    m_curDataSize = pdata.toInt();
}

CGAlgm::CGAlgm() :
    gnrNm(5000),
    ppSz(100),
    m_trntSz(5),
    m_elRt(0.03),
    m_mtrt(0.001),
    m_crRt(0.85),
    m_rndPp(false),
    m_soptp(false),
    m_ptpNm(200)
{
    setAutoDelete(false);
}

```

```

CGAlgm::CGAlgm(int gnNmrt, int ppszrt, int tnSzrt, double eRtt,
double mRtt, double cRtt, bool rndPprt, bool ptPtrt, int ptpNm) :
    gnrNm(gnNmrt),
    ppSz(ppszrt),
    m_trntSz(tnSzrt),
    m_elRt(eRtt),
    m_mtrt(mRtt),
    m_crRt(cRtt),
    m_rndPp(rndPprt),
    m_soptp(ptPtrt),
    m_ptpNm(200)
{
    setAutoDelete(false);
}

bool CGAlgm::intppffl(QString flpth)
{
    if(!ctMas.empty()){
        ctMas.clear();
    }
    QFile file(flpth);
    if (file.open(QIODevice::ReadOnly | QIODevice::Text)){
        emit lgdt("Завантаження даних ...\n");
        while (!file.atEnd()) {
            QStringList msto = QString(file.readLine()).split("
");
            emit lgdt(msto[1] +" "+ msto[2]);
            ctMas.push_back(CMsto(msto[1], msto[2]));
        }
        if(!ctMas.empty())
        {
            eltNms = std::ceil(m_elRt * ppSz);
            inSz = ctMas.size();
            emit lgctsnm(QString::clktber(inSz));
            emit drcts(ctMas);
            gntcRttNwppln();
        }
        return false;
    }
}

void CGAlgm::run()
{
    start();
}

void CGAlgm::gntcRttNwppln()

```

```

{
    if(m_rndPp)
    {
        int clkst = ppSz;
        if(m_soptp)
            clkst = m_ptpNm;

        std::vector<Indvl> tgmprl;
        for (int i = 0; i < clkst; ++i)
            tgmprl.push_back(Indvl(inSz, m_rndPp));
        for (int ndvl = 0; ndvl < clkst; ++ndvl) {
            double dstnc = 0.0;
            for (int gntc = 0; gntc < inSz-1; ++gntc) {
                dstnc +=
std::sqrt(std::pow(ctMas[tgmprl[ndvl].getGntcs()[gntc+1] - 1].m_x
- ctMas[tgmprl[ndvl].getGntcs()[gntc] - 1].m_x, 2) +

std::pow(ctMas[tgmprl[ndvl].getGntcs()[gntc+1] - 1].m_y -
ctMas[tgmprl[ndvl].getGntcs()[gntc] - 1].m_y, 2));
            }

            dstnc +=
std::sqrt(std::pow(ctMas[tgmprl[ndvl].getGntcs()[0] - 1].m_x -
ctMas[tgmprl[ndvl].getGntcs()[inSz-1] - 1].m_x, 2) +
            std::pow(ctMas[tgmprl[ndvl].getGntcs()[0] -
1].m_y - ctMas[tgmprl[ndvl].getGntcs()[inSz-1] - 1].m_y, 2));
            tgmprl[ndvl].setDstnc(dstnc);
            tgmprl[ndvl].setFitness(1.0/dstnc);
        }

        if(m_soptp){
            auto opt = getFtstNdvlidualIn(0, tgmprl);
            ppar.resize(ppSz, *opt);
        }else{
            ppar = tgmprl;
        }
    }
    else ppar.resize(ppSz, Indvl(inSz));

    int a = 5;
}

void CGAlgm::start()
{
    if(!mrslt.empty()){
        mrslt.clear();
        ppar.clear();
        gntcRttNwppln();
    }
}

```

```

emit strtd();
emit lgdt("\nРозпочинаю роботу ГА GNTCALG ...\n");
isrnng = true;
QElapsedTimer tmrd;
tmrd.start();
rnk();
for (int gntcratation = 0; gntcratation < gnrNm; ++gntcratation) {
    if(!isrnng) break;
    crsvr();
    mtnt();
    rnk();
    info(gntcratation);
}
emit snntmf(tmrd.elapsed()/500.0);
isrnng = false;
emit fnshd(mrslt);
emit lgdt("ГА завершує роботу ...\n");
}

void CGAlgm::stop()
{
    isrnng = false;
}

void CGAlgm::rnk()
{
    for (int ndvl = 0; ndvl < ppSz; ++ndvl) {
        double dstnc = 0.0;
        for (int gntc = 0; gntc < inSz-1; ++gntc) {
            dstnc +=
std::sqrt(std::pow(ctMas[ppar[ndvl].getGntcs()[gntc+1] - 1].m_x -
ctMas[ppar[ndvl].getGntcs()[gntc] - 1].m_x, 2) +
            std::pow(ctMas[ppar[ndvl].getGntcs()[gntc+1]
- 1].m_y - ctMas[ppar[ndvl].getGntcs()[gntc] - 1].m_y, 2));
        }
        dstnc +=
std::sqrt(std::pow(ctMas[ppar[ndvl].getGntcs()[0] - 1].m_x -
ctMas[ppar[ndvl].getGntcs()[inSz-1] - 1].m_x, 2) +
            std::pow(ctMas[ppar[ndvl].getGntcs()[0] - 1].m_y
- ctMas[ppar[ndvl].getGntcs()[inSz-1] - 1].m_y, 2));

        ppar[ndvl].setDstnc(dstnc);
        ppar[ndvl].setFitness(1.0/dstnc);
    }
}

void CGAlgm::crsvr()
{

```

```

std::vector<Indvl> nwppln(ppSz);
for (int ndvl = 0; ndvl < ppSz; ++ndvl) {
    auto prnt1 = getFtstNdvlidual(ndvl);
    if(m_crRt > dis(gnnt) && ndvl > eltNms){
        auto prnt2 = selectPrnt();
        std::vector<int> nshchdGntcs(inSz, -1);
        Indvl nshchd(nshchdGntcs);
        int tchk1 = std::ceil(dis(gnnt) * (inSz-1));
        int tchk2 = std::ceil(dis(gnnt) * (inSz-1));
        if(tchk1 > tchk2){
            int tgmprl = tchk1;
            tchk1 = tchk2;
            tchk2 = tgmprl;
        }
        for (int i = tchk1; i < tchk2; ++i) {
            nshchd.setGntc(i, prnt1->getGntc(i));
        }
        for (int i = 0; i < inSz; ++i) {
            int prnt2gntc = i + tchk2;
            if(prnt2gntc >= inSz){
                prnt2gntc -= inSz;
            }
        }

        if(!nshchd.containsGntc(prnt2->getGntc(prnt2gntc))){
            for (int j = 0; j < inSz; ++j) {
                if(nshchd.getGntc(j) == -1){
                    nshchd.setGntc(j,
prnt2->getGntc(prnt2gntc));
                    break;
                }
            }
        }
        nwppln[ndvl] = nshchd;
    } else {
        nwppln[ndvl] = *prnt1;
    }
}
ppar = nwppln;
}

void CGAlgm::mtnt()
{
    std::vector<Indvl> nwppln(ppSz);
    for (int ndvl = 0; ndvl < ppSz; ++ndvl) {
        auto ndvlidual = getFtstNdvlidual(ndvl);
        if(ndvl >= eltNms){
            for (int gntc = 0; gntc < inSz; ++gntc) {

```

```

        if(m_mtrt > dis(gnnt)){
            int newGntcTchk = std::ceil(dis(gnnt) *
(inSz-1));
            int gntc1 = ndvlidual->getGntc(newGntcTchk);
            int gntc2 = ndvlidual->getGntc(ndvl);
            ndvlidual->setGntc(ndvl, gntc1);
            ndvlidual->setGntc(newGntcTchk, gntc2);
        }
    }
}
nwppln[ndvl] = *ndvlidual;
}
ppar = nwppln;
}

void CGAlgm::info(int i)
{
    auto bestNdvl = getFtstNdvlidual(0);
    mrslt.push_back(Rsltvf(bestNdvl->getFitness(),
bestNdvl->getDstnc()));
    emit drpth(bestNdvl->getGntcs());
    emit lgdt("Gntcrations : " + QString::clkstber(i) + " = " +
QString::clkstber(bestNdvl->getDstnc()));
    QThread::msleep(2);
}

void CGAlgm::stgnrnm(int i)
{
    gnrNm = i;
}

int CGAlgm::getGntcrationsClkst() const
{
    return gnrNm;
}

void CGAlgm::stppsiz(int i)
{
    ppSz = i;
}

int CGAlgm::getPpsizrt() const
{
    return ppSz;
}

void CGAlgm::sttrnmsz(int i)

```

```

{
    m_trntSz = i;
}

int CGAlgm::getTrntSz() const
{
    return m_trntSz;
}

void CGAlgm::stelnm(int i)
{
    eltNms = i;
}

int CGAlgm::getElitesClkst() const
{
    return eltNms;
}

void CGAlgm::stmtrt(double i)
{
    m_mtrt = i;
}

int CGAlgm::getMutateRtt() const
{
    return m_mtrt;
}

void CGAlgm::stcrrt(double i)
{
    m_crRt = i;
}

int CGAlgm::getCrsvrRate() const
{
    return m_crRt;
}

bool CGAlgm::dataIsLoaded()
{
    return !ctMas.empty();
}

void CGAlgm::strndpplt(bool flag)
{
    m_rndPp = flag;
}

```

```

bool CGAlgm::gtrndppln()
{
    return m_rndPp;
}

void CGAlgm::stoptppn(bool flag)
{
    m_soptp = flag;
}

bool CGAlgm::gtoptppln()
{
    return m_soptp;
}

void CGAlgm::stoptppnClkst(int i)
{
    m_ptpNm = i;
}

int CGAlgm::gtoptpplnClkst()
{
    return m_ptpNm;
}

void Mgrvw::onDrcts(std::vector<CMsto>& c)
{
    ctcs = c;
    QPen penBlack(Qt::black);
    foreach (CMsto msto, ctcs){
        ctcgrp->adgrp(scene->addEllipse(msto.m_x, msto.m_y, 0.5,
0.5, penBlack, QBrush(Qt::black)));
    }
}

void Mgrvw::onDrpth(std::vector<int> pthd)
{
    mpth = pthd;
    dltpd(pthdsGroup);
    for (int i = 0; i < (int)mpth.size()-1; ++i) {
        pthdsGroup->adgrp(scene->addLine(ctcs[mpth[i] - 1].m_x,
ctcs[mpth[i] - 1].m_y, ctcs[mpth[i+1] - 1].m_x, ctcs[mpth[i+1] -
1].m_y, penRed));
    }
}

```

```
    pthdsGroup->adgrp(scene->addLine(ctcs[mpth[mpth.size()-1] -  
1].m_x, ctcs[mpth[mpth.size()-1] - 1].m_y, ctcs[mpth[0] - 1].m_x,  
ctcs[mpth[0] - 1].m_y, penRed));  
}
```

Додаток В
Слайди презентації

Міністерство освіти та науки України
Національний університет «Запорізька політехніка»
Кафедра програмних засобів

**ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ
ЗАСОБІВ РОЗВ'ЯЗАННЯ
ЗАВДАНЬ ДИСКРЕТНОЇ ОПТИМІЗАЦІЇ**

Виконав
ст. гр. КНТ-112м

Андрій ДЗУГАНЬ

Керівник
к. т. н., доцент

Тетяна КОЛПАКОВА

Рисунок В.1 – Слайд 1

Об'єктом дослідження є процес розробки програмного забезпечення для дискретної оптимізації.

Предмет дослідження – методи та програмні засоби для розв'язання завдань дискретної оптимізації.

Мета роботи – дослідження та розробка програмного забезпечення для розв'язання завдань дискретної оптимізації.

Рисунок В.2 – Слайд 2

Завдання роботи:

- виконати аналіз предметної області та програмних засобів для розв'язання завдань дискретної оптимізації;
- розробити модель взаємодії користувача з програмним забезпеченням для розв'язання завдань дискретної оптимізації;
- здійснити проектування програмного забезпечення для розв'язання завдань дискретної оптимізації;
- створити програмне забезпечення для розв'язання завдань дискретної оптимізації;
- виконати тестування розробленого програмного забезпечення для розв'язання завдань дискретної оптимізації.

3

Рисунок В.3 – Слайд 3

Дослідження програмних засобів для дискретної оптимізації

Критерій порівняння	Lingo [11]	TSPSG [12]	OptimoRoute
Необхідність розуміння базових принципів програмування	+	+	–
Зручність розв'язання завдань комівояжера	+-	+	+
Можливість врахування додаткових умов при розв'язанні оптимізаційних завдань	+-	+	–
Засоби інтеграції з зовнішнім програмним забезпеченням	+	–	–
Можливість використання багатьох ядер процесора для синтезу оптимізаційних моделей	+	+	–

4

Рисунок В.4 – Слайд 4

Вибір мови програмування

Критерій порівняння мов програмування	Мова програмування		
	C++	Matlab	Python
Відкритість коду	+	–	+
Велика спільнота розробників	+	–	+
Статична типізація	+	–	–
Ручне виділення пам'яті	+	–	–
Наявність бібліотек для вирішення задачі комівояжера	+	+	+–

5

Рисунок В.5 – Слайд 5

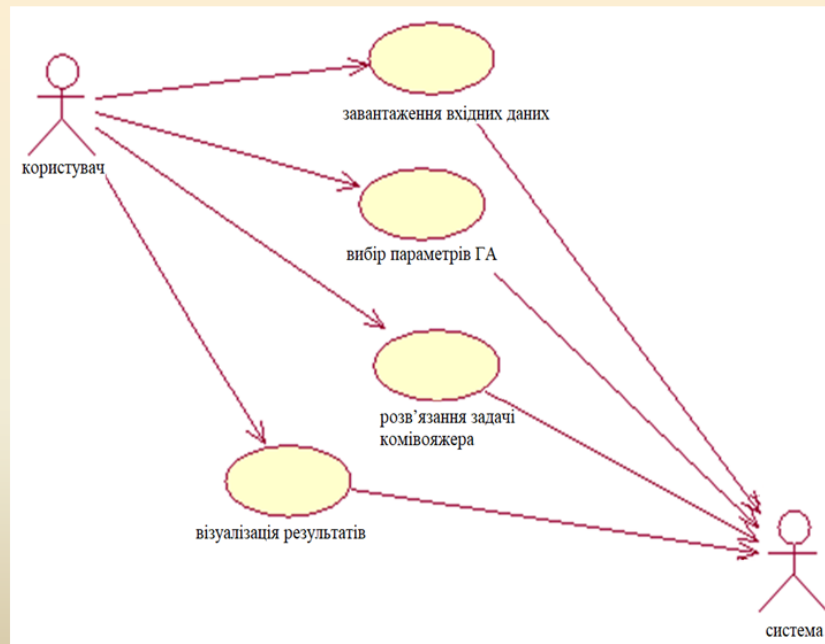
Вибір середовища розробки

Критерій порівняння середовищ розробки	Середовище розробки		
	Qt Creator	Microsoft Visual Studio	C++Builder
Зручність розробки на мові C++	+	+–	+
Підтримка зовнішніх інструментів	+	+–	+–
Кросплатформність	+	+–	+–
Зручність для створення ПЗ для розв'язання задачі комівояжера	+	–	–
Вільна ліцензія	+	–	–

6

Рисунок В.6 – Слайд 6

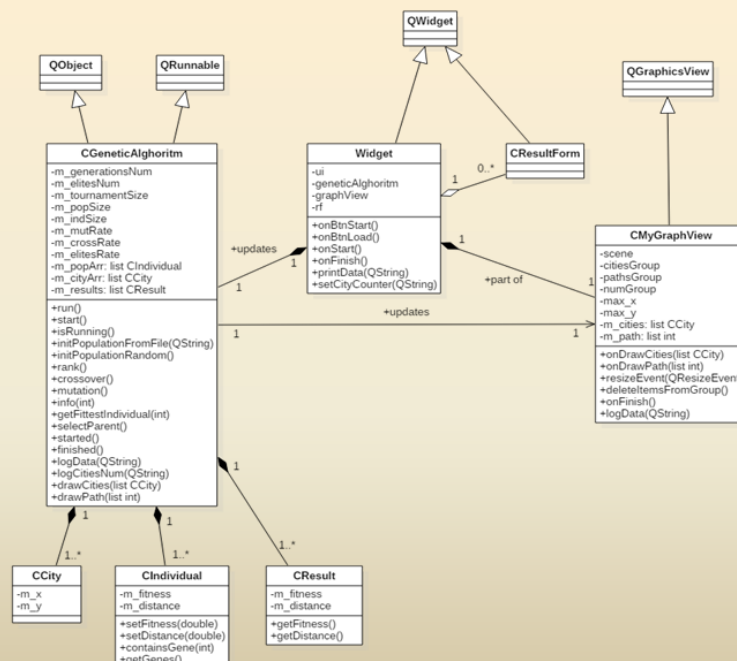
Модель взаємодії користувача з програмним забезпеченням



7

Рисунок В.7 – Слайд 7

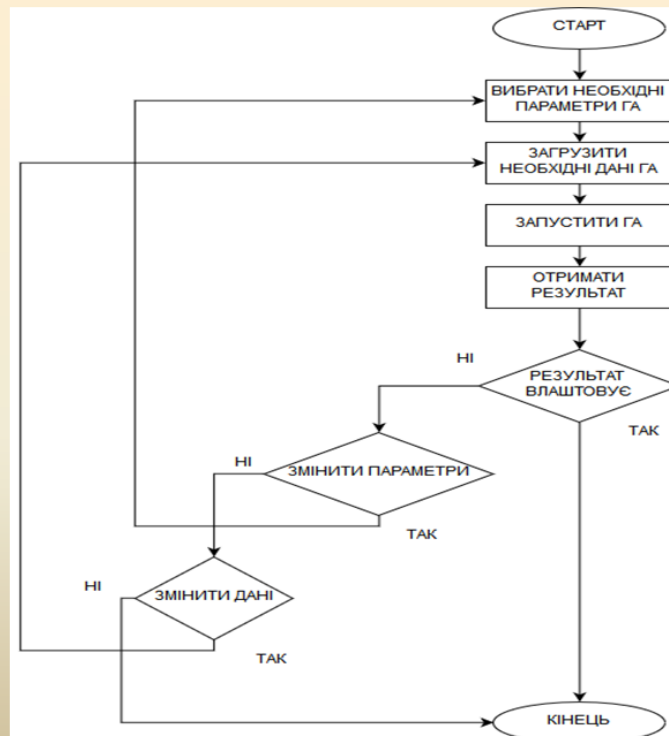
Структура програмного забезпечення



8

Рисунок В.8 – Слайд 8

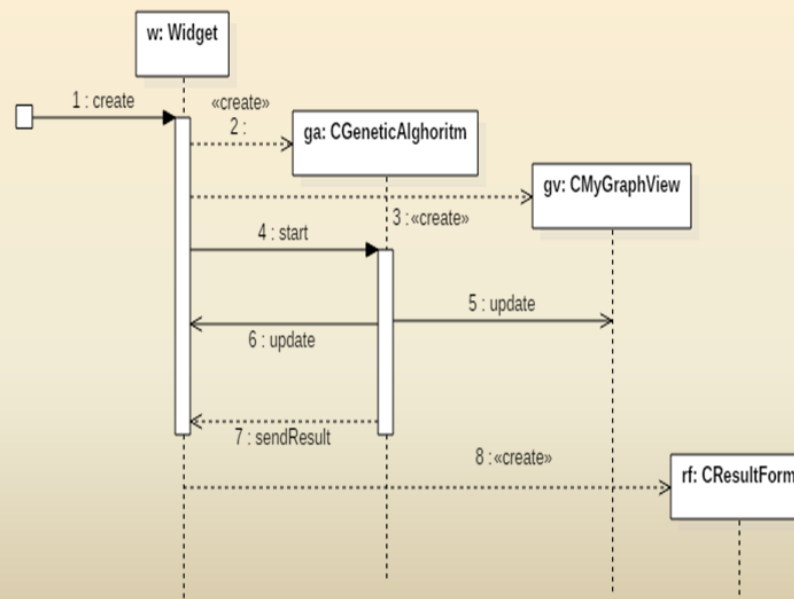
Узагальнена схема роботи програми



9

Рисунок В.9 – Слайд 9

Діаграма послідовностей програмного забезпечення для розв'язання задачі комівояжера



10

Рисунок В.10 – Слайд 10

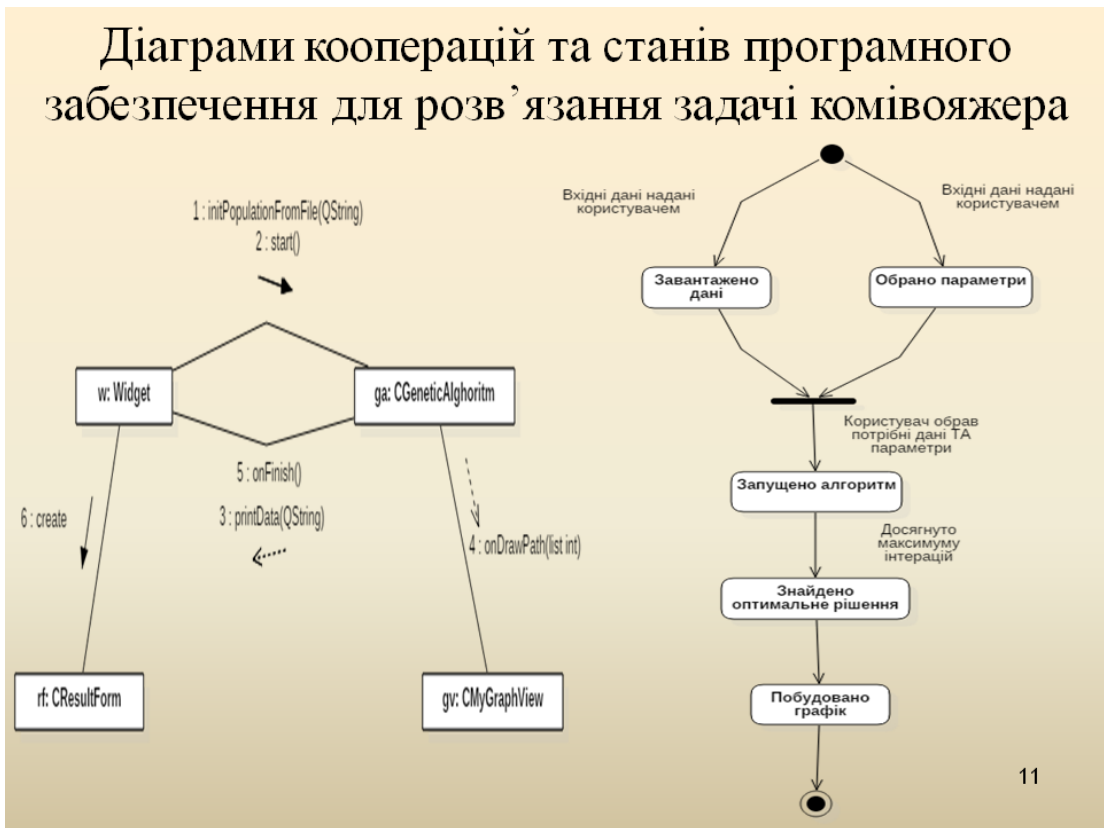


Рисунок В.11 – Слайд 11

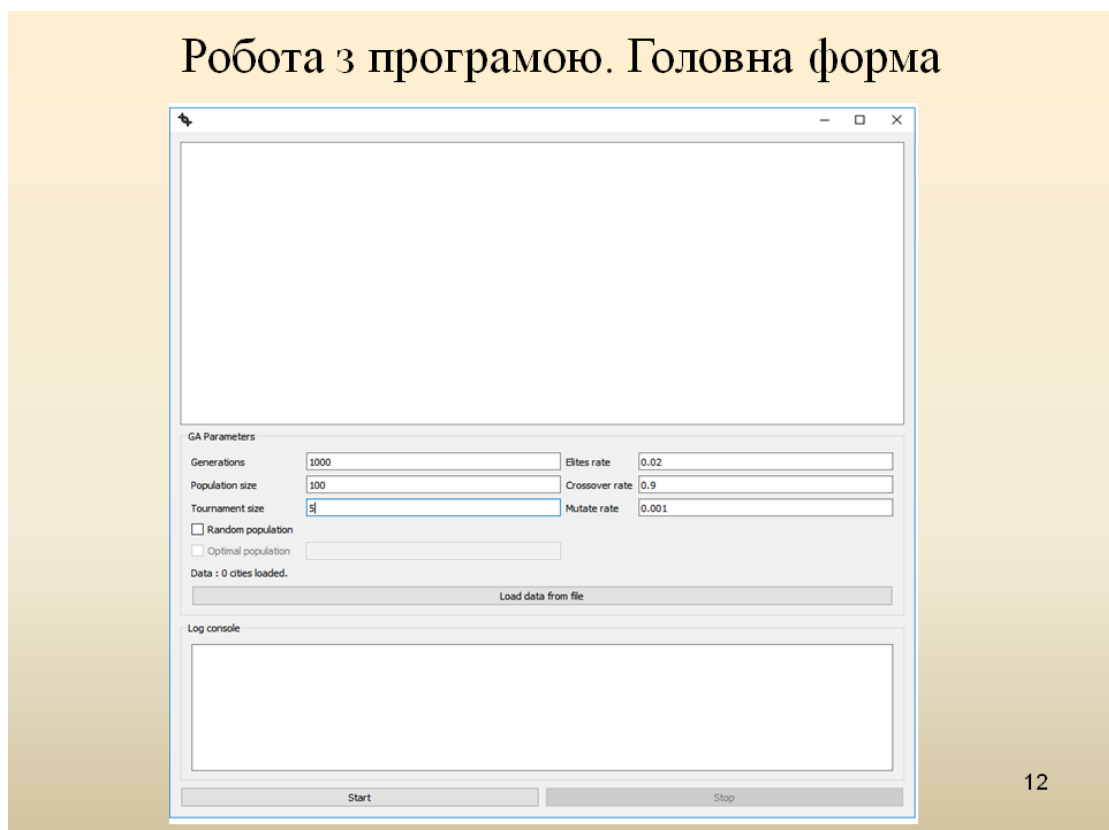
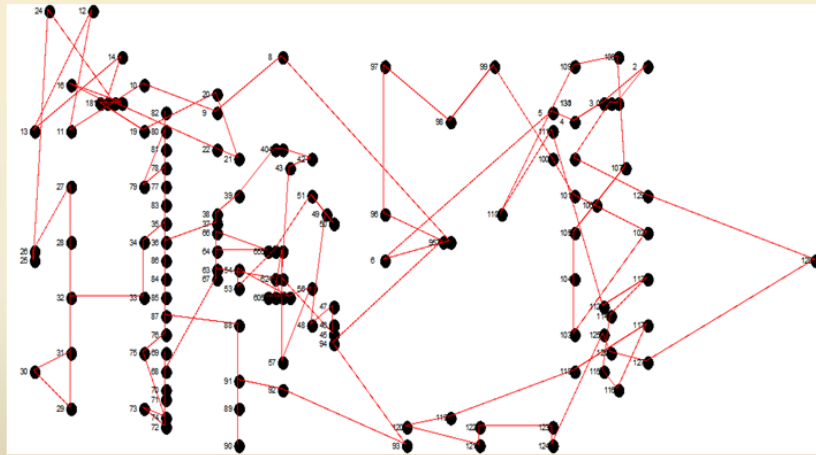


Рисунок В.12 – Слайд 12

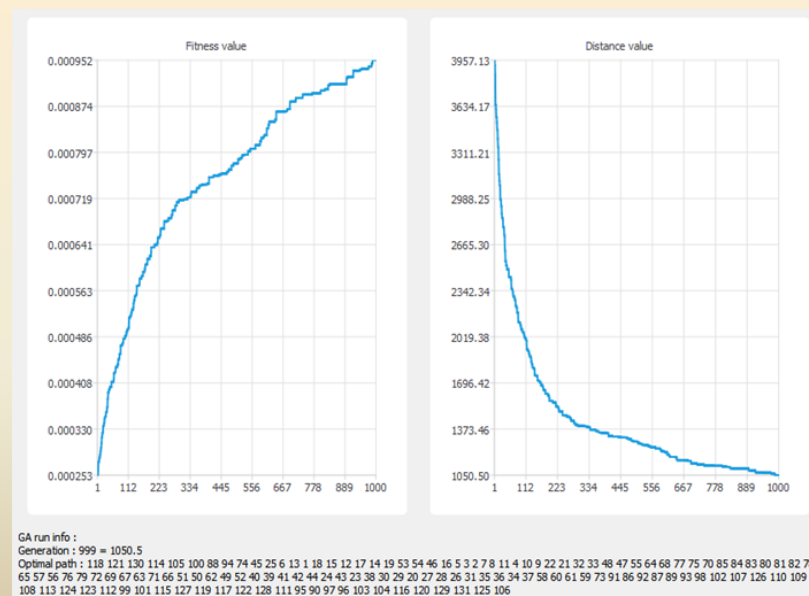
Робота з програмою. Вікно з візуалізацією знайденого оптимального шляху



13

Рисунок В.13 – Слайд 13

Робота з програмою. Результати функціонування генетичного алгоритму



14

Рисунок В.14 – Слайд 14

Висновки

В ході виконання дипломної кваліфікаційної роботи магістра було проаналізовано та досліджено процес розробки програмного забезпечення для дискретної оптимізації.

Новизна роботи полягає в тому, що запропоновано модель взаємодії користувача з програмним забезпеченням для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації, яка подана у вигляді діаграми прецедентів у нотації UML, та дозволяє розроблювати програмне забезпечення для розв'язання задачі комівояжера.

Практичне значення роботи полягає в тому, що розроблено програмне забезпечення для розв'язання задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

Виконано тестування розробленого програмного забезпечення. Результати тестування програмного забезпечення показали, що розроблена програма дозволяє розв'язувати задачі комівояжера за допомогою генетичних методів дискретної оптимізації.

15