УДК 004.021

Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*
(повна назва)
Кафедра *Системотехніки*
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Дослідження методів розв'язання багатокритеріальних задач транспортної маршрутизації

(тема)

Виконав:
Студент 2 курсу, групи *СПРм-22-2*
Спеціальність *122 – Комп'ютерні науки*
(код і повна назва напряму)
Тип програми *освітньо-наукова*
(освітньо-професійна або освітньо-наукова)
Освітня програма *Системне проєктування*
(повна назва освітньої програми)

Ісакій К.Г.
(прізвище, ініціали)

Керівник *д.т.н., професор Гребеннік І.В.*
(посада, прізвище, ініціали)

Допускається до захисту
Зав. кафедри _____ *Гребеннік І. В.*
(підпис) (прізвище, ініціали)

2024 р.

Кваліфікаційна робота не містить відомостей заборонених до відкритого опублікування.

Кваліфікаційна робота виконана у відповідності до стандартів, що діють в Україні.

Попередній захист проведений «10» червня 2024 р.

Керівник кваліфікаційної роботи д.т.н., професор Гребеннік І.В.

Харківський національний університет радіоелектроніки

Факультет *Комп'ютерних наук*

(повна назва)

Кафедра *Системотехніки*

(повна назва)

Рівень вищої освіти другий (магістерський)

Спеціальність *122 – Комп'ютерні науки*

(код і повна назва)

Тип програми *освітньо-наукова*

(освітньо-професійна або освітньо-наукова)

Освітня програма *Системне проєктування*

(повна назва)

ЗАТВЕРДЖУЮ:

Зав. кафедри _____

(підпис)

«____»_____ 20___ р.

ЗАВДАННЯ

НА АТЕСТАЦІЙНУ РОБОТУ

студентові Ісакію Костянтину Григоровичу

(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження методів розв'язання багатокритеріальних задач транспортної маршрутизації*

затверджена наказом по університету від «01» квітня 2024 р. № 259 Ст

2. Термін подання студентом роботи (проекту) 13 червня 2024 р.

3. Вихідні дані до роботи (проекту) Функція: визначення оптимального маршруту транспортної маршрутизації з найкоротшою відстанню та найменшим часом в дорозі з використанням часових вікон на базі генетичного алгоритму та алгоритму мурашиних колоній для даних з датасету r206 Solomon benchmark.

4. Зміст пояснювальної записки (перелік питань, що потрібно розробити) Вступ, Аналіз предметної галузі та постановка задачі, Актуальність теми, Огляд і аналіз сучасного стану проблеми маршрутизації транспорту, Постановка задачі, Теоретико-методологічні основи аналізу транспортної маршрутизації, Поняття та класифікація задач транспортної маршрутизації, Capacitated vehicle routing problem, Mehicle routing problem with time windows, Multi-depot vehicle routing problem, Heterogeneous vehicle routing problem, Dynamic vehicle routing problem, Vehicle routing problem with driver working hours, Vehicle routing problem

with pick-up and delivery, Теоретичні основи транспортної маршрутизації, Класичні методи вирішення задач транспортної маршрутизації , Евристичні та метаевристичні методи, Моделювання транспортних систем, Оцінка ефективності транспортної маршрутизації, Висновки за розділом, Аналіз та експериментальна оцінка задачі маршрутизації транспортних засобів з часовими вікнами, Математична постановка задачі маршрутизації транспортних засобів з часовими вікнами, Опис та застосування генетичного алгоритму (GA), Опис та застосування алгоритму мурашиних колоній (ACO), Порівняння результатів, Висновки за розділом 3, Висновки, Перелік джерел посилання.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслеників, плакатів): Класифікація характеристик VRP, Класифікація найпоширеніших алгоритмів VRP, Візуалізація оптимального маршруту побудованого генетичним алгоритмом, Візуалізація оптимального маршруту побудованого ACO.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1.	Отримання завдання на атестаційне проектування	01.04.2024	
2.	Аналіз завдання та пошук літератури з теми роботи	05.04.2024	
3.	Опрацювання літератури та аналіз об'єкту дослідження	12.04.2024	
4.	Вибір апаратного набору та його збірка	26.04.2024	
5.	Розробка алгоритмів	17.05.2024	
6.	Аналіз отриманих результатів	31.05.2024	
7.	Оформлення пояснювальної записки та документації	06.06.2024	
8.	Оформлення презентаційних матеріалів	07.06.2024	
9.	Представлення на рецензування	10.06.2024	
10.	Представлення атестаційної роботи	13.06.2024	

Дата видачі завдання *01 квітня 2024 р.*

Студент _____ *Ісакій К.Г*

(підпис)

Керівник роботи _____ д.т.н., професор Гребеннік І.В.

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи: 73 с., 7 рис., 3 табл., 3 додатки, 25 джерел інформації.

ТРАНСПОРТНА ЗАДАЧА, ЗАВДАННЯ МАРШРУТИЗАЦІЇ ТРАНСПОРТУ, ЗАДАЧА VRP, ЗАДАЧА VRPTW, ЧАСОВІ ВІКНА, МЕТАЕВРИСТИЧНІ АЛГОРИТМИ

Об’єктом дослідження в роботі є планування та оптимізація транспортних перевезень.

Предметом досліджень є алгоритми планування та методи оптимізації транспортних перевезень.

Мета роботи - підвищення ефективності системи планування та оптимізації транспортних перевезень.

Методи дослідження – методи теорії ймовірності, математичної статистики та математичне імітаційне моделювання.

Сфера застосування – покращення ефективності транспортних систем, зниженні витрат, покращенні якості обслуговування та зменшенні викидів шкідливих речовин, що сприяє підвищенню конкурентоспроможності підприємств.

ABSTRACT

Qualification work: 73 p., 7 pic., 3 tables, 25 sources, 3 applications.

TRANSPORTATION PROBLEM, TRANSPORT ROUTING PROBLEM, VRP PROBLEM, VRPTW PROBLEM, TIME WINDOWS, METAHEURISTIC ALGORITHMS

The object of the study is the planning and optimization of transportation.

The subject of the study is planning algorithms and methods for optimizing transportation.

The purpose of the work is to improve the efficiency of the planning and optimization system for transportation.

Research methods – methods of probability theory, mathematical statistics, and mathematical simulation modeling.

Application area – improving the efficiency of transport systems, reducing costs, improving service quality, and reducing harmful emissions, which contributes to increasing the competitiveness of enterprises.

ЗМІСТ

ВСТУП

Вирішення задач транспортної маршрутизації є однією з ключових проблем в галузі логістики та управління транспортними системами. Сучасні транспортні системи стикаються з численними викликами, такими як зростаючий трафік, змінні умови дорожнього руху, різноманітні вимоги клієнтів та необхідність зменшення впливу на навколишнє середовище. Традиційні методи вирішення задач транспортної маршрутизації не завжди здатні врахувати ці динамічні фактори, що може призводити до збільшення витрат на перевезення та зниження рівня обслуговування клієнтів.

Однією з основних проблем є те, що традиційні методи маршрутизації часто базуються на статичних моделях, які не враховують реальні умови дорожнього руху та динамічні зміни в попиті на перевезення. У сучасному світі, де обсяги даних зростають експоненціально, використання великих даних стає необхідним для більш точного та ефективного планування маршрутів. Це дозволяє враховувати динамічні зміни в дорожньому русі, погодних умовах та інших факторах, що впливають на ефективність транспортних систем.

Актуальність роботи обумовлена необхідністю підвищення ефективності транспортних систем в умовах динамічних змін. Використання великих даних та сучасних алгоритмів, таких як генетичні алгоритми та глибоке навчання, дозволяє значно покращити точність прогнозування та оптимізацію маршрутів. Це, в свою чергу, сприяє зниженню витрат на перевезення, підвищенню рівня обслуговування клієнтів та зменшенню негативного впливу на навколишнє середовище.

Зростаюча конкуренція в галузі логістики та транспортних послуг вимагає від компаній постійного вдосконалення своїх процесів. Впровадження інноваційних підходів до маршрутизації транспорту дозволяє не тільки зменшити операційні витрати, але й підвищити конкурентоспроможність компаній на ринку.

Об’єктом дослідження є процеси транспортної маршрутизації в логістичних системах.

Предметом дослідження є методи та алгоритми розв'язання багатокритеріальних задач транспортної маршрутизації.

Метою даної роботи є дослідження методів розв'язання багатокритеріальних задач транспортної маршрутизації з використанням аналізу великих даних та сучасних алгоритмів. Для досягнення цієї мети необхідно вирішити наступні завдання:

- Провести огляд літератури з традиційних та сучасних методів вирішення задач транспортної маршрутизації.
- Розробити математичну модель багатокритеріальної задачі маршрутизації.
- Впровадити генетичні алгоритми для оптимізації маршрутів.
- Використати методи глибокого навчання для прогнозування попиту.
- Провести експерименти для оцінки ефективності запропонованих методів.

Для досягнення поставлених цілей використовуються такі методи дослідження:
аналіз та синтез літературних джерел, математичне моделювання, генетичні алгоритми, методи глибокого навчання, експериментальні дослідження.

Наукова новизна дослідження полягає у розробці та впровадженні нових методів розв'язання багатокритеріальних задач транспортної маршрутизації з використанням аналізу великих даних та сучасних алгоритмів, що дозволяє підвищити ефективність та точність прийнятих рішень.

Практичне значення одержаних результатів полягає в можливості застосування запропонованих методів для підвищення ефективності роботи транспортних систем у різних галузях, таких як логістика, перевезення вантажів та пасажирів. Використання сучасних алгоритмів та великих даних дозволяє знизити витрати на перевезення, підвищити рівень обслуговування клієнтів та зменшити негативний вплив на навколишнє середовище.

Результати дослідження можуть бути використані для вдосконалення існуючих систем транспортної маршрутизації, а також для розробки нових підходів до управління транспортними потоками в умовах динамічних змін. Основні результати за темою магістерської роботи пройшли апробацію:

– на хакатоні «Open Data Campus» (м. Харків), що відбувся 25-26 січня 2020 р. у рамках відкритого проекту USAID / UK «Прозорість та підзвітність у державному управлінні та послугах» під егідою Міністерства цифрової трансформації України [1-2];

– на міжнародній науково-практичній конференції «MODERN RESEARCH IN SCIENCE AND EDUCATION» (м.Чикаго, США) [3].

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1 Актуальність теми

В умовах сучасного розвитку логістичних систем та транспортних мереж ефективне планування та оптимізація транспортних маршрутів є надзвичайно важливими завданнями. Зростаючий обсяг перевезень, змінні умови дорожнього руху, підвищені вимоги до точності та швидкості доставки, а також екологічні обмеження вимагають від логістичних компаній впровадження новітніх технологій та методів для забезпечення конкурентоспроможності та стійкості на ринку.

Основні аспекти, що обумовлюють актуальність дослідження теми багатокритеріальної задачі транспортної маршрутизації, включають:

Зростання обсягів перевезень та урбанізація: Збільшення населення великих міст та розвиток електронної комерції призводять до значного зростання обсягів вантажних та пасажирських перевезень. Це створює додаткове навантаження на транспортні системи, що потребує ефективних рішень для маршрутизації та планування.

Змінні умови дорожнього руху: Динамічні зміни в дорожньому русі, такі як затори, аварії, погодні умови, вимагають від транспортних компаній оперативного реагування та перерозподілу ресурсів. Традиційні статичні методи не здатні адекватно враховувати ці фактори, що знижує ефективність перевезень.

Підвищені вимоги до якості обслуговування: Сучасні клієнти очікують високого рівня обслуговування, включаючи точність та своєчасність доставки. Це потребує від логістичних компаній забезпечення максимальної ефективності маршрутів, мінімізації часу доставки та оптимізації використання ресурсів.

Екологічні вимоги та сталий розвиток: Зменшення впливу на навколишнє середовище стає важливим критерієм при плануванні транспортних перевезень. Використання оптимізованих маршрутів дозволяє знизити споживання палива, викиди шкідливих речовин та вплив на екологію, що сприяє досягненню цілей

сталого розвитку.

Розвиток технологій великих даних та інтелектуальних алгоритмів: Застосування технологій великих даних, генетичних алгоритмів та методів глибокого навчання відкриває нові можливості для вдосконалення процесів маршрутизації. Використання реальних даних про дорожній рух, прогнози попиту та аналіз історичних даних дозволяє створювати більш точні та ефективні моделі для планування маршрутів.

Таким чином, актуальність теми дослідження полягає у необхідності підвищення ефективності транспортних систем шляхом впровадження інноваційних методів та технологій, що дозволяють враховувати динамічні умови та забезпечувати високу якість обслуговування клієнтів. Результати дослідження можуть мати значний вплив на розвиток логістики та транспортних систем, сприяючи зниженню витрат, покращенню екологічних показників та підвищенню конкурентоспроможності компаній.

1.2 Огляд і аналіз сучасного стану проблеми маршрутизації транспорту

Проблема оптимальної маршрутизації транспорту є однією з ключових задач у сфері логістики та управління транспортними системами. Сучасні методи та підходи до вирішення задач транспортної маршрутизації включають як класичні математичні моделі, так і новітні інтелектуальні алгоритми, що базуються на аналізі великих даних та машинному навчанні. У цьому розділі розглядаються основні підходи до вирішення задач маршрутизації транспорту, їхні переваги та недоліки, а також аналізується сучасний стан проблеми.

Класичні підходи до вирішення задач маршрутизації:

Лінійне та цілочисельне програмування:

Одними з найпоширеніших методів вирішення задач транспортної маршрутизації є методи лінійного та цілочисельного програмування. Ці методи дозволяють сформулювати задачу маршрутизації як математичну модель, яка може бути розв'язана за допомогою оптимізаційних алгоритмів. Основною перевагою таких підходів є їхня строгість та математична обґрунтованість. Однак, для великих

та складних задач ці методи можуть бути непридатними через високі обчислювальні витрати та необхідність спрощення моделі.

Евристичні та метаевристичні алгоритми:

Для вирішення складних задач маршрутизації, які не можуть бути ефективно розв'язані за допомогою лінійного програмування, часто використовуються евристичні та метаевристичні алгоритми. До таких методів належать:

– генетичні алгоритми: Імітують процес природного відбору та еволюції для знаходження оптимальних рішень. Генетичні алгоритми є ефективними для вирішення задач з великою кількістю змінних та обмежень, але можуть вимагати значного налаштування параметрів;

– алгоритми табу-пошуку: Використовують механізми запам'ятовування попередніх рішень для уникнення повторного відвідування вже розглянутих варіантів. Цей підхід дозволяє ефективно обробляти великі простори пошуку, але може бути менш ефективним для задач з високою динамічністю;

– алгоритми імітаційного відпалу: Моделюють процес охолодження металу для знаходження глобального оптимуму. Цей метод є ефективним для задач з численними локальними екстремумами, але потребує значного обчислювального часу.

Сучасні підходи до вирішення задач маршрутизації:

Використання великих даних

З розвитком технологій збору та аналізу великих даних з'явилися нові можливості для покращення процесів транспортної маршрутизації. Великі дані дозволяють враховувати динамічні зміни в дорожньому русі, погодних умовах та інших факторах, що впливають на ефективність транспортних систем. Завдяки аналізу великих даних можна отримати більш точні прогнози попиту та оптимальні маршрути, що знижують витрати на перевезення та підвищують рівень обслуговування клієнтів.

Використання великих даних дозволяє також інтегрувати інформацію з різних джерел, таких як GPS-трекери, сенсори дорожнього руху та соціальні мережі, що забезпечує більш повну та точну картину стану транспортної системи в реальному

часі. Це, в свою чергу, дозволяє транспортним компаніям швидко реагувати на зміни та коригувати маршрути для підвищення ефективності.

Методи глибокого навчання

Глибоке навчання, зокрема нейронні мережі, показують високу ефективність у вирішенні задач прогнозування та класифікації. Застосування методів глибокого навчання до задач транспортної маршрутизації дозволяє враховувати складні залежності між різними параметрами та отримувати більш точні прогнози попиту.

– нейронні мережі: Використовуються для аналізу великих обсягів даних та виявлення прихованих закономірностей. Нейронні мережі можуть бути налаштовані для вирішення конкретних задач, таких як прогнозування трафіку або попиту на перевезення;

– рекурентні нейронні мережі (RNN): Ефективні для обробки послідовних даних, що дозволяє їх використовувати для прогнозування часу прибуття транспорту або аналізу історичних даних про дорожній рух;

– глибокі нейронні мережі (DNN): Можуть бути використані для створення складних моделей, які враховують велику кількість факторів, що впливають на транспортну маршрутизацію;

– інтеграція традиційних методів та новітніх підходів

Сучасний стан досліджень у галузі транспортної маршрутизації демонструє, що найкращі результати досягаються при поєднанні традиційних методів оптимізації з новітніми підходами, що базуються на аналізі великих даних та глибокому навчанні. Це дозволяє враховувати динамічні умови та забезпечувати високу ефективність транспортних систем.

Гібридні моделі: Поєднують переваги традиційних алгоритмів оптимізації з можливостями аналізу великих даних та інтелектуальних алгоритмів. Наприклад, генетичні алгоритми можуть бути використані для знаходження початкових рішень, які потім оптимізуються за допомогою методів глибокого навчання.

Інтеграція реальних даних: Використання реальних даних про дорожній рух, погодні умови та інші фактори дозволяє створювати більш точні моделі та підвищувати ефективність транспортних систем.

На сьогоднішній день, значна частина досліджень та практичних впроваджень у сфері транспортної маршрутизації спрямована на поєднання класичних методів оптимізації з новітніми підходами, що базуються на аналізі великих даних та машинному навчанні. Це дозволяє враховувати динамічні умови та забезпечувати високу ефективність транспортних систем.

Основними викликами, що залишаються актуальними, є необхідність обробки великих обсягів даних у реальному часі, інтеграція різних джерел даних та забезпечення стійкості алгоритмів до змінних умов. Впровадження сучасних підходів, таких як глибоке навчання та генетичні алгоритми, дозволяє вирішувати ці задачі більш ефективно, проте потребує додаткових досліджень та оптимізації.

– обробка великих обсягів даних: Необхідність обробки великих обсягів даних у реальному часі є одним з основних викликів сучасних транспортних систем. Це вимагає розробки нових методів та алгоритмів, які можуть ефективно працювати з великими даними.

– інтеграція різних джерел даних: Для створення точних моделей транспортних систем необхідно інтегрувати дані з різних джерел, таких як GPS-трекери, сенсори дорожнього руху та соціальні мережі. Це дозволяє отримати більш повну та точну картину стану транспортної системи в реальному часі.

– стійкість алгоритмів до змінних умов: Сучасні алгоритми повинні бути стійкими до змінних умов, таких як погодні умови, затори або аварії. Це вимагає розробки адаптивних моделей, які можуть швидко реагувати на зміни.

Сучасний стан проблеми транспортної маршрутизації вимагає інтеграції класичних методів оптимізації з новітніми підходами, що базуються на аналізі великих даних та інтелектуальних алгоритмах. Це дозволяє підвищити ефективність транспортних систем, знизити витрати на перевезення та підвищити рівень обслуговування клієнтів. Подальші дослідження мають бути спрямовані на вдосконалення цих підходів та їх адаптацію до реальних умов експлуатації транспортних систем.

Розробка та впровадження нових методів транспортної маршрутизації, що враховують динамічні умови та базуються на аналізі великих даних, дозволяють

значно покращити ефективність транспортних систем. Це сприяє зниженню витрат, підвищенню якості обслуговування та зменшенню негативного впливу на навколишнє середовище, що є важливими аспектами для досягнення сталого розвитку та підвищення конкурентоспроможності компаній.

1.3 Постановка задачі

В умовах сучасного розвитку логістичних систем та транспортних мереж ефективне планування та оптимізація транспортних маршрутів є надзвичайно важливими завданнями. Зростаючий обсяг перевезень, підвищені вимоги до точності та швидкості доставки, а також екологічні обмеження вимагають від логістичних компаній впровадження новітніх технологій та методів для забезпечення конкурентоспроможності та стійкості на ринку.

Основна задача магістерської роботи полягає в розробці та впровадженні методів та алгоритмів розв’язання багатокритеріальних задач транспортної маршрутизації, що дозволяють підвищити ефективність транспортних систем за рахунок використання аналізу великих даних та сучасних алгоритмів. Для досягнення визначеної мети необхідно:

– провести огляд літератури з традиційних та сучасних методів вирішення задач транспортної маршрутизації.

– дослідити математичну модель багатокритеріальної задачі маршрутизації, яка враховує основні фактори, що впливають на ефективність перевезень.

– впровадити генетичні алгоритми для оптимізації маршрутів, що дозволяють знайти ефективні рішення в умовах великої кількості змінних та обмежень.

– використати методи глибокого навчання для прогнозування попиту на транспортні перевезення, що дозволяє враховувати динамічні зміни в попиті.

– провести експерименти для оцінки ефективності запропонованих методів на основі реальних даних про дорожній рух та попит на перевезення.

– розробити рекомендації щодо впровадження запропонованих методів в реальні транспортні системи для підвищення їх ефективності.

2 ТЕОРЕТИКО-МЕТОДОЛОГІЧНІ ОСНОВИ АНАЛІЗУ ТРАНСПОРТНОЇ МАРШРУТИЗАЦІЇ

2.1 Поняття та класифікація задач транспортної маршрутизації

Задачі транспортної маршрутизації (Vehicle Routing Problems, VRP) є одними з найважливіших і найскладніших задач в області логістики та транспортних систем. Вони полягають у визначенні оптимальних маршрутів для транспортних засобів, що здійснюють перевезення товарів або пасажирів, з метою мінімізації витрат, часу доставки, покращення якості обслуговування та зменшення впливу на навколишнє середовище (Рисунок 2.1). У цьому розділі розглянемо основні поняття та класифікацію задач транспортної маршрутизації.

Рисунок 2.1. Класифікація характеристик VRP

Задача транспортної маршрутизації (VRP) – це оптимізаційна задача, яка полягає у визначенні набору маршрутів для автотранспортних засобів, що мають

обслуговувати певну кількість клієнтів. Основною метою задачі є мінімізація загальних витрат на перевезення, які можуть включати відстань, час, витрати палива та інші ресурси.

Основні елементи задачі транспортної маршрутизації:

– транспортні засоби: транспортні засоби, які мають обмежену вантажопідйомність або обсяг.

– клієнти: точки, які мають бути обслуговуванні транспортними засобами. Кожен клієнт має свої вимоги до обслуговування, наприклад, кількість вантажу або часове вікно.

– депо: початкова та кінцева точка для кожного маршруту, де розташовані транспортні засоби.

– маршрути: послідовність точок, які має відвідати кожен транспортний засіб для обслуговування клієнтів.

Задачі транспортної маршрутизації можуть бути класифіковані за різними ознаками залежно від специфіки завдання, обмежень та цілей оптимізації. Основні види задач VRP включають (Рисунок 2.2):

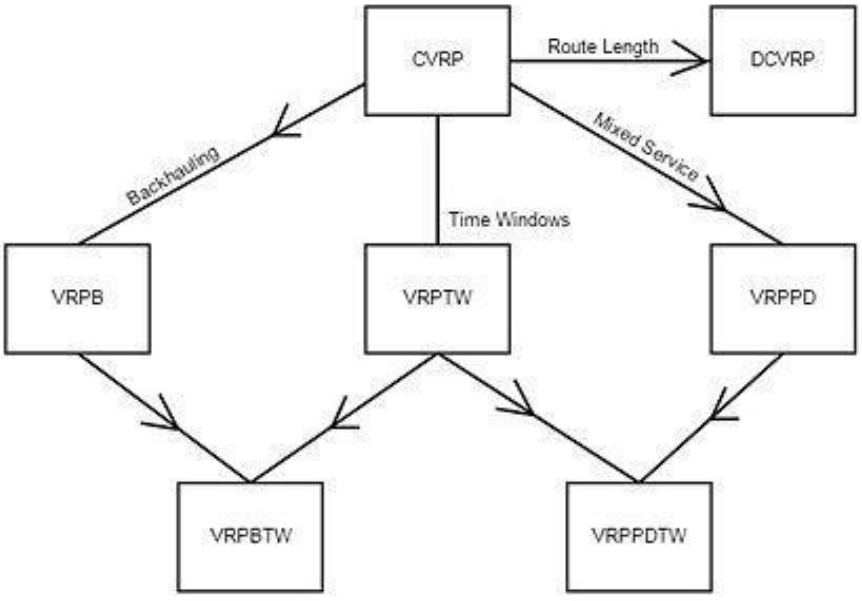

Рисунок 2.2. Основні види задач VRP

2.1.1 Capacitated Vehicle Routing Problem

Класична задача транспортної маршрутизації (Capacitated Vehicle Routing Problem, CVRP) є однією з найважливіших і найпоширеніших задач в області транспортної логістики та оптимізації. CVRP полягає в розробці оптимальних маршрутів транспортних засобів, які починають і закінчують свої маршрути в одному депо, забезпечуючи доставку товарів до клієнтів, при цьому кожен транспортний засіб має обмежену вантажопідйомність. Основною метою є мінімізація загальної вартості, відстані маршрутів при дотриманні всіх обмежень [4].

Основні компоненти CVRP

Транспортні засоби:

– кожен транспортний засіб має фіксовану вантажопідйомність Q.

– транспортні засоби починають і закінчують свої маршрути в одному депо.

Клієнти:

– кожен клієнт має певний попит q_i, який повинен бути задоволений одним транспортним засобом.

– всі клієнти повинні бути обслуговуванні тільки один раз.

Депо:

– відправна і кінцева точка для всіх маршрутів транспортних засобів.

Маршрути:

– послідовність точок (клієнтів), які має відвідати кожен транспортний засіб.

Математична модель CVRP може бути представлена наступним чином:

– набір клієнтів: V={1, 2, ..., n}

– депо: 0

– матриця відстаней: C=[c_{ij}], де c_{ij} - відстань або вартість між клієнтами i та j

– попит клієнтів: q_i для кожного клієнта i.

2.1.2 Vehicle Routing Problem with Time Windows

Задача маршрутизації транспортних засобів з часовими вікнами (Vehicle Routing Problem with Time Windows, VRPTW) є розширенням класичної задачі транспортної маршрутизації, яка включає додаткове обмеження на часові інтервали, протягом яких клієнти можуть бути обслуговуванні. Ця задача є критично важливою в багатьох галузях, таких як доставка товарів, надання послуг та логістика, де своєчасність обслуговування є ключовим фактором успіху.

Основні компоненти VRPTW включають транспортні засоби, клієнтів, депо та маршрути. Кожен транспортний засіб має фіксовану вантажопідйомність і починає та закінчує свої маршрути в одному депо. Клієнти мають певний попит, який повинен бути задоволений одним транспортним засобом, і визначений часовий інтервал, протягом якого вони можуть бути обслуговуванні.

Метою VRPTW є мінімізація загальної відстані або вартості маршрутів при дотриманні всіх обмежень. Математична модель VRPTW включає набір клієнтів, депо, матрицю відстаней, попит клієнтів та часові інтервали. Змінні рішення включають двійкові змінні, які вказують, чи проходить транспортний засіб від одного клієнта до іншого, та час прибуття транспортного засобу до кожного клієнта.

Основні обмеження задачі включають забезпечення того, що кожен клієнт відвідується лише один раз, дотримання вантажопідйомності транспортних засобів та дотримання часових інтервалів обслуговування клієнтів. Важливим аспектом є також безперервність маршрутів, тобто забезпечення того, що транспортні засоби не повертаються в депо до завершення всіх запланованих доставок.

Методи вирішення VRPTW включають точні методи, такі як метод гілок і меж та динамічне програмування, а також евристичні та метаевристичні підходи. До останніх належать жадібні алгоритми, конструктивні евристики, генетичні алгоритми, табу-пошук та імітаційний відпал. Кожен з цих методів має свої переваги та обмеження і може бути обраний в залежності від специфіки задачі та доступних ресурсів. Задача маршрутизації транспортних засобів з множинними депо (MDVRP)

є задачею оптимізації маршрутів, яка передбачає вибір найбільш ефективного маршруту для доставки товарів або послуг від кількох депо до групи клієнтів [5].

2.1.3 Multi-Depot Vehicle Routing Problem

Задача маршрутизації транспортних засобів з множинними депо (Multi-Depot Vehicle Routing Problem, MDVRP) є складнішою версією класичної задачі транспортної маршрутизації (CVRP), в якій транспортні засоби можуть починати і закінчувати свої маршрути в різних депо. Це обмеження значно ускладнює планування маршрутів, але одночасно робить задачу більш реалістичною для великих логістичних мереж, де є кілька складів або розподільчих центрів.

Метою MDVRP є оптимізація маршрутів таким чином, щоб мінімізувати загальні витрати або відстань, пройдену транспортними засобами, при цьому враховуючи обмеження на вантажопідйомність транспортних засобів, розташування депо і довжину маршрутів. Ця задача є особливо важливою для логістичних та транспортних компаній, де ефективне управління автопарком є ключовим фактором успіху.

Основними компонентами MDVRP є транспортні засоби, клієнти, депо та маршрути. Кожен транспортний засіб має фіксовану вантажопідйомність і може починати та закінчувати свій маршрут у будь-якому з доступних депо. Клієнти мають певний попит, який повинен бути задоволений одним транспортним засобом. Депо є початковими і кінцевими точками для маршрутів транспортних засобів.

Вирішення MDVRP ускладнюється з ростом кількості депо і транспортних засобів, оскільки необхідно забезпечити скоординовану роботу всієї системи, щоб всі клієнти були обслуговуванні своєчасно та з мінімальними витратами. Це вимагає розробки ефективних алгоритмів та методів оптимізації, які можуть враховувати множинні обмеження і варіанти рішень.

Методи вирішення MDVRP включають точні методи, такі як метод гілок і меж та динамічне програмування, а також евристичні та метаевристичні підходи. Евристичні методи, такі як жадібні алгоритми та конструктивні евристики,

дозволяють швидко знаходити наближені рішення, тоді як метаевристичні методи, такі як генетичні алгоритми, табу-пошук та імітаційний відпал, дозволяють покращувати ці рішення та знаходити глобальні оптимуми. Задача маршрутизації з обмеженнями на кількість транспортних засобів (VRP with Limited Fleet) передбачає обмеження на кількість транспортних засобів, що можуть бути використані для обслуговування клієнтів. Метою є оптимізація маршрутів з урахуванням цього обмеження [6].

2.1.4 Heterogeneous Vehicle Routing Problem

Задача маршрутизації транспортних засобів з гетерогенними транспортними засобами (Heterogeneous Vehicle Routing Problem, HVRP) є різновидом класичної задачі транспортної маршрутизації (CVRP), яка враховує різноманітність характеристик транспортних засобів. У HVRP транспортні засоби можуть мати різну вантажопідйомність, вартість експлуатації, паливну ефективність та інші параметри, що додає додаткову складність до задачі оптимізації маршрутів.

Метою HVRP є оптимізація маршрутів таким чином, щоб мінімізувати загальні витрати або відстань, пройдену транспортними засобами, враховуючи різноманітні характеристики та обмеження для кожного типу транспортного засобу. Це завдання є надзвичайно важливим для логістичних і транспортних компаній, де ефективне управління різними типами транспортних засобів є ключовим для зниження витрат і підвищення продуктивності.

Основними компонентами HVRP є транспортні засоби, клієнти, депо та маршрути. Можуть бути транспортні засоби з різною вантажопідйомністю, вартістю експлуатації та іншими характеристиками. Кожен клієнт має певний попит, який повинен бути задоволений одним із доступних транспортних засобів. Депо є початковими і кінцевими точками для маршрутів транспортних засобів.

Вирішення HVRP ускладнюється через необхідність врахування різноманітних характеристик транспортних засобів і забезпечення оптимального розподілу

ресурсів. Це вимагає розробки ефективних алгоритмів та методів оптимізації, які можуть враховувати множинні обмеження і варіанти рішень.

Методи вирішення HVRP включають точні методи, такі як метод гілок і меж та лінійне програмування, а також евристичні та метаевристичні підходи. Евристичні методи, такі як жадібні алгоритми та конструктивні евристики, дозволяють швидко знаходити наближені рішення, тоді як метаевристичні методи, такі як генетичні алгоритми, табу-пошук та імітаційний відпал, дозволяють покращувати ці рішення та знаходити глобальні оптимуми [7].

2.1.5 Dynamic Vehicle Routing Problem

Динамічна задача маршрутизації транспортних засобів (Dynamic Vehicle Routing Problem, DVRP) є розширенням класичної задачі транспортної маршрутизації (CVRP), яка враховує динамічні зміни в системі, такі як нові замовлення, зміни в дорожніх умовах, затори та інші фактори, що можуть змінюватися в реальному часі. На відміну від статичних задач маршрутизації, де всі дані відомі заздалегідь, DVRP потребує постійного оновлення інформації та адаптації маршрутів у відповідь на нові дані.

Основними компонентами DVRP є транспортні засоби, клієнти, депо та маршрути. Кожен транспортний засіб має фіксовану вантажопідйомність і починає та закінчує свої маршрути в одному депо. Клієнти можуть подавати замовлення в реальному часі, і кожне замовлення має певний попит, який повинен бути задоволений одним із транспортних засобів.

Метою DVRP є оптимізація маршрутів таким чином, щоб мінімізувати загальні витрати або відстань, пройдену транспортними засобами, при цьому враховуючи зміни в системі в реальному часі. Це завдання є надзвичайно важливим для логістичних і транспортних компаній, де ефективне управління автопарком в умовах невизначеності та змін є ключовим для зниження витрат і підвищення продуктивності.

Вирішення DVRP ускладнюється через необхідність постійного оновлення інформації та адаптації маршрутів. Це вимагає розробки ефективних алгоритмів та методів оптимізації, які можуть швидко реагувати на зміни і забезпечувати оптимальні рішення в реальному часі.

Методи вирішення DVRP включають точні методи, такі як метод гілок і меж та динамічне програмування, а також евристичні та метаевристичні підходи. Евристичні методи, такі як жадібні алгоритми та конструктивні евристики, дозволяють швидко знаходити наближені рішення, тоді як метаевристичні методи, такі як генетичні алгоритми, табу-пошук та імітаційний відпал, дозволяють покращувати ці рішення та знаходити глобальні оптимуми.

Особливу роль у вирішенні DVRP відіграють методи машинного навчання та аналізу великих даних, які дозволяють прогнозувати попит і змінювати маршрути в режимі реального часу. Використання технологій IoT (Internet of Things) також сприяє підвищенню точності і швидкості збору даних, що є критично важливим для динамічного планування маршрутів [8].

2.1.6 Vehicle Routing Problem with Driver Working Hours

Задача маршрутизації транспортних засобів з обмеженнями на робочий час водіїв (Vehicle Routing Problem with Driver Working Hours, VRP-DWH) є розширенням класичної задачі транспортної маршрутизації (CVRP), яка враховує додаткові обмеження на час роботи водіїв. Це включає максимальну тривалість робочого дня, час перерв та інші нормативні обмеження, що регулюють робочий час водіїв. Ці обмеження є критично важливими для забезпечення безпеки водіїв та дотримання законодавчих вимог.

Основними компонентами VRP-DWH є транспортні засоби, клієнти, депо, маршрути та розклад роботи водіїв. Кожен транспортний засіб має фіксовану вантажопідйомність і починає та закінчує свої маршрути в одному депо. Клієнти мають певний попит, який повинен бути задоволений одним із транспортних засобів.

Водії мають обмеження на максимальну тривалість робочого дня, включаючи час на перерви та відпочинок.

Метою VRP-DWH є оптимізація маршрутів таким чином, щоб мінімізувати загальні витрати або відстань, пройдену транспортними засобами, при цьому враховуючи обмеження на робочий час водіїв. Це завдання є надзвичайно важливим для логістичних і транспортних компаній, де ефективне управління автопарком і дотримання правил робочого часу є ключовими для забезпечення безпеки та підвищення продуктивності [24].

Вирішення VRP-DWH ускладнюється через необхідність врахування додаткових обмежень на робочий час водіїв. Це вимагає розробки ефективних алгоритмів та методів оптимізації, які можуть забезпечити оптимальні рішення, враховуючи всі обмеження і вимоги.

Методи вирішення VRP-DWH включають точні методи, такі як метод гілок і меж та динамічне програмування, а також евристичні та метаевристичні підходи. Евристичні методи, такі як жадібні алгоритми та конструктивні евристики, дозволяють швидко знаходити наближені рішення, тоді як метаевристичні методи, такі як генетичні алгоритми, табу-пошук та імітаційний відпал, дозволяють покращувати ці рішення та знаходити глобальні оптимуми.

Особливу роль у вирішенні VRP-DWH відіграють також методи машинного навчання та аналізу великих даних, які дозволяють прогнозувати попит і змінювати маршрути з урахуванням робочого часу водіїв. Використання технологій IoT (Internet of Things) сприяє підвищенню точності і швидкості збору даних, що є критично важливим для динамічного планування маршрутів [5].

2.1.7 Vehicle Routing Problem with Pick-Up and Delivery

Задача маршрутизації з розподілом і збором (Vehicle Routing Problem with Pick-Up and Delivery, VRPDP) є складною версією класичної задачі транспортної маршрутизації, яка передбачає не тільки доставку товарів до клієнтів, але і збір вантажів у клієнтів. Ця задача має велике значення для логістичних та транспортних

компаній, оскільки дозволяє ефективно поєднувати процеси доставки і збору, зменшуючи загальні витрати на перевезення.

В основі VRPDP лежить необхідність оптимального планування маршрутів для транспортних засобів, які здійснюють як доставку, так і збір вантажів. Кожен транспортний засіб має обмежену вантажопідйомність, і всі вони починають і закінчують свої маршрути в одному або декількох депо. Клієнти можуть мати потребу як у доставці вантажів, так і у зборі, і для кожного клієнта визначено часовий інтервал, протягом якого він може бути обслугований.

Метою задачі VRPDP є мінімізація загальної відстані або вартості маршрутів, необхідної для обслуговування всіх клієнтів. Це включає забезпечення своєчасної доставки і збору вантажів, дотримання обмежень на вантажопідйомність транспортних засобів і часові інтервали обслуговування клієнтів.

Для вирішення задачі VRPDP використовуються різні методи, включаючи точні, евристичні та метаевристичні підходи.

Точні методи, такі як метод гілок і меж, розбивають задачу на підзадачі і поступово виключають неконкурентоспроможні рішення, знаходячи оптимальні маршрути для обслуговування клієнтів. Динамічне програмування також може бути ефективним для вирішення задач середнього розміру, де можливо розбити задачу на менші підзадачі і об'єднувати їхні рішення.

Евристичні методи включають жадібні алгоритми та конструктивні евристики, які використовують прості правила для побудови початкових рішень. Жадібні алгоритми на кожному кроці обирають найближчого клієнта для обслуговування, тоді як конструктивні евристики поступово додають елементи до рішення, доки воно не буде завершеним.

Метаевристичні методи, такі як генетичні алгоритми, алгоритми табу-пошуку та імітаційний відпал, дозволяють знаходити наближені до оптимальних рішення для великих і складних задач. Генетичні алгоритми імітують природний відбір і еволюцію, використовуючи операції кросоверу та мутації для створення нових рішень. Алгоритми табу-пошуку запам'ятовують попередні рішення і забороняють повернення до них у найближчі ітерації, уникаючи локальних мінімумів. Імітаційний

відпал базується на процесі охолодження металу, де випадкові зміни в рішенні допускаються з певною ймовірністю, яка зменшується з часом.

Загалом, задача маршрутизації з розподілом і збором є важливою для оптимізації транспортних процесів у сучасній логістиці. Вона дозволяє ефективно поєднувати процеси доставки і збору, знижуючи витрати та покращуючи якість обслуговування клієнтів. Використання різних методів вирішення забезпечує гнучкість і ефективність у досягненні високих показників продуктивності транспортних систем.

Поняття та класифікація задач транспортної маршрутизації охоплюють широкий спектр проблем, що виникають у процесі планування та оптимізації транспортних перевезень. Кожен тип задачі має свої специфічні особливості та вимоги, що впливають на вибір методів та алгоритмів для їх вирішення. Розуміння цих аспектів є ключовим для ефективного вирішення задач транспортної маршрутизації та покращення загальної ефективності логістичних систем [9-11].

2.2 Теоретичні основи транспортної маршрутизації

Транспортна маршрутизація (Vehicle Routing) є важливою галуззю дослідження в логістиці та операційному менеджменті. Вона включає в себе планування оптимальних маршрутів для транспортних засобів з метою мінімізації витрат, підвищення ефективності обслуговування клієнтів та зменшення негативного впливу на навколишнє середовище. У цьому розділі розглянемо основні теоретичні аспекти транспортної маршрутизації, включаючи математичні моделі, критерії оптимальності та підходи до формалізації задач.

Математичні моделі задач транспортної маршрутизації

Математична модель задачі транспортної маршрутизації зазвичай складається з наступних основних компонентів:

Множини та змінні:

V – множина всіх вершин (клієнтів) у графі.

E – множина всіх ребер (маршрутів) між вершинами.

C – вартість (або відстань) проїзду між вершинами.

Q – вантажопідйомність транспортного засобу.

q_i – обсяг вантажу, що має бути доставлений до клієнта

Змінні рішення:

x_{ij} – двійкова змінна, яка дорівнює 1, якщо маршрут між клієнтами i та j використовується, і 0 – в іншому випадку.

y_i – кількість вантажу, що перевозиться транспортним засобом після обслуговування клієнта i.

Обмеження:

Вантажопідйомність: $\sum_{i \in V} q_i \leq Q$.

Маршрутизація: кожен клієнт повинен бути обслуговований тільки один раз.

Забезпечення безперервності маршруту для кожного транспортного засобу.

Функція цілі: мінімізація загальних витрат на перевезення: $\sum_{i \in V} \sum_{j \in V} C_{ij} x_{ij}$.

Підходи до формалізації задач транспортної маршрутизації

Існує декілька підходів до формалізації задач транспортної маршрутизації:

Лінійне програмування (LP): Основний підхід до вирішення задач оптимізації, де функція цілі та обмеження представлені у вигляді лінійних рівнянь та нерівностей.

Цілочисельне програмування (IP): Використовується для задач, де змінні рішення повинні приймати цілочисельні значення (наприклад, кількість транспортних засобів або кількість поїздок).

Динамічне програмування (DP): Підхід, який використовується для вирішення задач, де рішення приймаються послідовно з урахуванням попередніх рішень.

Основні критерії оптимальності, які використовуються в задачах транспортної маршрутизації, включають:

– мінімізація загальної відстані або часу: основний критерій для задач, де основним завданням є зменшення витрат на паливо або зменшення часу доставки.

– мінімізація загальних витрат: включає витрати на паливо, амортизацію транспортних засобів, оплату праці водіїв та інші операційні витрати.

– максимізація рівня обслуговування клієнтів: забезпечення своєчасної доставки та високого рівня обслуговування клієнтів, враховуючи часові вікна та інші вимоги клієнтів.

– мінімізація впливу на навколишнє середовище: зменшення викидів шкідливих речовин та інших екологічних впливів.

Теоретичні основи транспортної маршрутизації включають в себе широкий спектр методів та підходів до вирішення задач оптимізації транспортних маршрутів. Використання математичних моделей, евристичних та метаевристичних методів, а також технологій великих даних та глибокого навчання дозволяє розробляти ефективні рішення для покращення логістичних систем та підвищення ефективності транспортних перевезень.

2.3 Класичні методи вирішення задач транспортної маршрутизації

Класичні методи вирішення задач транспортної маршрутизації (Рисунок 2.3) є основоположними підходами, які широко використовуються в теорії оптимізації та практиці логістичного планування.

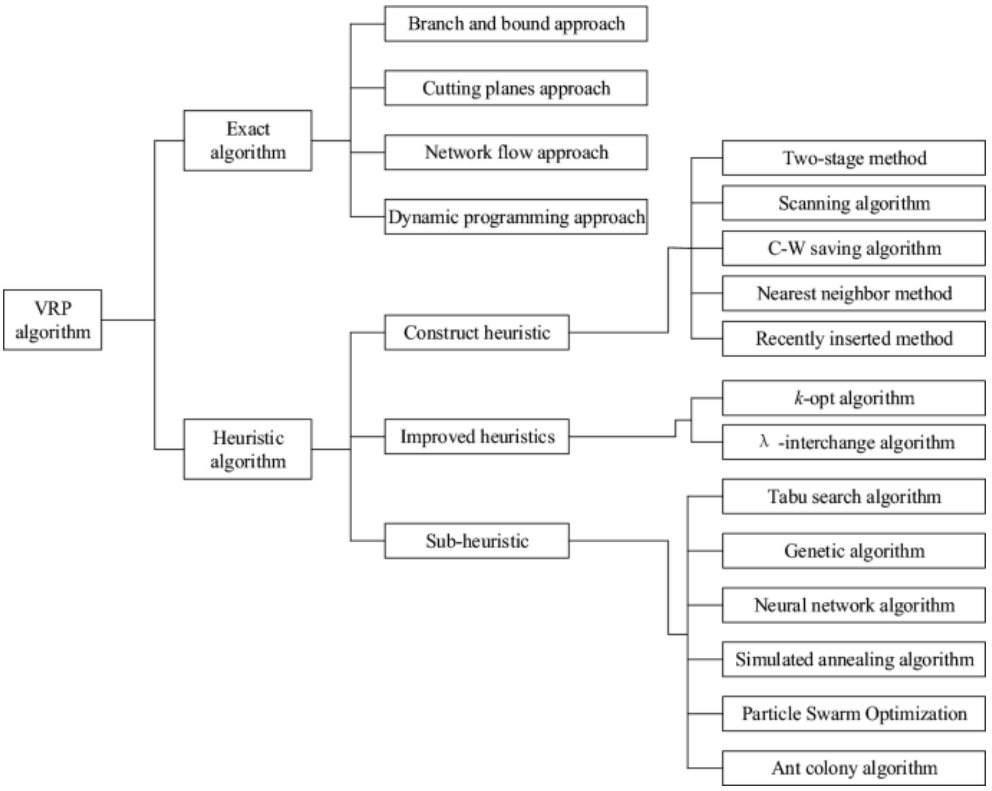

Рисунок 2.3. Основні методи вирішення VRP задач

Одним із найпоширеніших методів є метод гілок і меж, який застосовується для розв'язання задач цілочисельного програмування. Суть методу полягає в поступовому розбитті початкової задачі на підзадачі шляхом додавання обмежень (гілок), з подальшою оцінкою (межами) для кожної підзадачі, щоб відсіяти підзадачі, які не можуть містити оптимального рішення. Цей підхід дозволяє ефективно зменшити розмір простору пошуку та знайти оптимальне рішення шляхом послідовного розділення задачі на менші та простіші частини [14].

Основні етапи методу гілок і меж:

– ініціалізація: Встановлюється початкова верхня межа (наприклад, значення функції цілі для якогось допустимого рішення) і початкова нижня межа (наприклад, значення функції цілі для релаксації задачі).

– розбиття: Вибирається підзадача з найменшою межею та розбивається на менші підзадачі. Це може бути зроблено шляхом вибору певного ребра або вершини в графі та поділу на два випадки: включення цього ребра або вершини в розв'язок та виключення.

– оцінка: Для кожної нової підзадачі обчислюються нові межі. Якщо нижня межа для підзадачі перевищує поточну верхню межу, ця підзадача відкидається.

– оновлення: Якщо знайдено нове допустиме рішення з меншою вартістю, оновлюється верхня межа.

– завершення: Процес триває, доки всі підзадачі не будуть розглянуті або відкинуті.

Цей метод дозволяє ефективно скоротити розмір простору пошуку, що робить його придатним для вирішення задач середньої складності.

Графові методи, такі як методи Дейкстри та Беллмана-Форда, використовуються для знаходження оптимальних шляхів у графах.

Метод Дейкстри є ефективним для знаходження найкоротших шляхів від однієї вершини до всіх інших вершин графа з невід'ємними вагами ребер.

Алгоритм:

– ініціалізуйте всі відстані від початкової вершини до інших вершин як нескінченність, а відстань до самої себе як 0.

– виберіть вершину з найменшою відстанню, яка ще не була оброблена.

– для кожного сусіда обраної вершини обчисліть нову відстань і, якщо вона менша за поточну відстань, оновіть її.

– повторюйте кроки 2 і 3, доки всі вершини не будуть оброблені.

Метод Беллмана-Форда, у свою чергу, здатний знаходити найкоротші шляхи навіть у графах з від'ємними вагами ребер, що робить його більш універсальним. Обидва методи є основними інструментами для вирішення задач маршрутизації, оскільки дозволяють знаходити оптимальні маршрути в транспортних мережах.

Алгоритм:

– ініціалізуйте всі відстані від початкової вершини до інших вершин як нескінченність, а відстань до самої себе як 0.

– для кожного ребра в графі, якщо нова відстань до кінцевої вершини через це ребро менша за поточну відстань, оновіть її.

– повторіть крок 2 (V-1) разів, де V – кількість вершин.

– перевірте наявність циклів негативної ваги, повторивши крок 2 ще раз; якщо відстані зміняться, то існує цикл негативної ваги.

Метод динамічного програмування є ще одним потужним інструментом для вирішення задач транспортної маршрутизації, зокрема задачі комівояжера. Цей метод ефективний для задач, де рішення приймаються послідовно, з урахуванням попередніх рішень. Основна ідея методу полягає в розбитті задачі на менші підзадачі, які вирішуються незалежно, з подальшим об'єднанням отриманих результатів для знаходження оптимального рішення. Метод динамічного програмування дозволяє значно зменшити обчислювальні витрати та час, необхідні для вирішення складних задач маршрутизації.

Алгоритм:

– визначити стан кожного підбору міст, які вже відвідані, та останнього відвіданого міста.

– рекурсивно обчислити мінімальну вартість для кожного стану.

– використовуйте таблицю для зберігання результатів підзадач, щоб уникнути повторних обчислень.

Загалом, класичні методи вирішення задач транспортної маршрутизації базуються на чітких математичних моделях і алгоритмах, що дозволяють ефективно знаходити оптимальні рішення для широкого спектра логістичних задач. Використання методу гілок і меж, графових методів та динамічного програмування забезпечує високу точність та надійність рішень, що є важливими для підвищення ефективності транспортних систем та зниження витрат на перевезення [25].

2.4 Евристичні та метаевристичні методи

Евристичні та метаевристичні методи є важливими підходами до вирішення задач транспортної маршрутизації, особливо коли точні методи стають неефективними через високу обчислювальну складність або масштаб задачі. Евристичні методи використовують правила та підходи, які дозволяють знайти хороші, хоча й не завжди оптимальні рішення за обмежений час. Метаевристичні методи, в свою чергу, є більш загальними підходами, які поєднують кілька евристичних стратегій для досягнення кращих результатів.

Одним із найбільш відомих евристичних методів є генетичні алгоритми (Genetic Algorithms), які імітують процес природного відбору та еволюції. Генетичні алгоритми починаються з випадкової популяції можливих рішень, які поступово еволюціонують через операції кросоверу та мутації. Вибір найкращих рішень для наступного покоління базується на їхній пристосованості до заданої функції цілі. Цей підхід дозволяє знаходити хороші рішення для складних задач з великою кількістю змінних та обмежень [13, 16].

Етапи:

– ініціалізація: Створення початкової популяції рішень.

– оцінка: Кожен індивід оцінюється за допомогою функції пристосованості (fitness function).

– відбір: Вибір індивідів для створення наступного покоління.

– кросовер: Поєднання частин двох батьківських рішень для створення нових індивідів.

– мутація: Випадкові зміни в індивідах для підтримки генетичної різноманітності.

Переваги: Можуть знайти близькі до оптимальних рішення для великих та складних задач.

Недоліки: Вимагають багато обчислювальних ресурсів і часу.

Жадібні алгоритми (Greedy Algorithms)

Жадібні алгоритми використовують просту стратегію: на кожному кроці вони обирають локально оптимальне рішення з надією, що воно призведе до глобально оптимального рішення. Наприклад, у задачі комівояжера жадібний алгоритм може починати з будь-якого міста і завжди переходити до найближчого не відвіданого міста, доки всі міста не будуть відвідані [15].

Переваги: Простота реалізації та швидке виконання.

Недоліки: Можуть призводити до субоптимальних рішень, оскільки не завжди обирають глобально оптимальний шлях.

Конструктивні евристики (Constructive Heuristics)

Конструктивні евристики поступово будують рішення, додаючи один елемент за раз, поки не буде досягнуто повне рішення. Один з прикладів - алгоритм Найближчого сусіда (Nearest Neighbor), де починають з випадкового міста і кожного разу обирають найближчого сусіда.

Переваги: Швидке побудова початкових рішень.

Недоліки: Рішення можуть бути далеко від оптимальних.

Алгоритми табу-пошуку є ще одним ефективним метаевристичним методом, який використовує механізми запам'ятовування попередніх рішень для уникнення повторного відвідування вже розглянутих варіантів. Це дозволяє алгоритму уникати локальних мінімумів та досліджувати нові області простору пошуку. Табу-пошук починається з певного початкового рішення, яке поступово покращується шляхом обміну елементів маршруту з використанням "таблиці табу" для запам'ятовування заборонених кроків.

Етапи:

– ініціалізація: Початкове рішення та створення табу-списку.

– пошук сусідніх рішень: Генерація нових рішень шляхом незначних змін.

– оцінка та оновлення: Вибір найкращого сусіднього рішення, яке не знаходиться в табу-списку, та оновлення табу-списку.

Ройові алгоритми (Ant Colony Optimization), такі як алгоритм мурашиних колоній та алгоритм бджолиного рою, також широко використовуються для вирішення задач транспортної маршрутизації. Вони базуються на поведінці соціальних комах і використовують кооперативний пошук для знаходження оптимальних рішень. Алгоритм мурашиних колоній, наприклад, імітує поведінку мурах, які залишають феромонні сліди на своєму шляху, щоб інші мурахи могли слідувати за цими слідами до оптимального рішення [17-19].

Етапи:

– ініціалізація: Розподіл мурах по різним точкам.

– пошук: Мурахи рухаються по графу, залишаючи феромони.

– оновлення: Інтенсивність феромонів оновлюється на основі якості знайдених рішень.

– випаровування: Зменшення інтенсивності феромонів з часом для уникнення застою.

Переваги: Ефективний для складних комбінаторних задач, добре масштабований.

Недоліки: Може потребувати значних обчислювальних ресурсів.

Загалом, евристичні та метаевристичні методи забезпечують гнучкі та потужні інструменти для вирішення складних задач транспортної маршрутизації. Використання генетичних алгоритмів, табу-пошуку та ройових алгоритмів дозволяє знаходити високоякісні рішення за обмежений час, що є особливо важливим у реальних логістичних системах з великою кількістю змінних та обмежень.

2.5 Моделювання транспортних систем

Моделювання транспортних систем є важливим етапом у процесі планування та оптимізації логістичних операцій. Моделювання дозволяє створити віртуальну копію реальної транспортної системи, що дозволяє досліджувати її поведінку під різними умовами, оцінювати ефективність різних стратегій та приймати обґрунтовані рішення. У цьому розділі розглядаються основні принципи моделювання транспортних систем, види моделей, методи моделювання та програмне забезпечення, яке використовується для цієї мети.

Принципи моделювання транспортних систем

Моделювання транспортних систем базується на ряді принципів, які визначають структуру, компоненти та взаємодію елементів системи. Основні принципи включають:

– системний підхід: Розгляд транспортної системи як комплексу взаємозалежних компонентів, таких як транспортні засоби, інфраструктура, маршрути, логістичні центри та клієнти.

– ієрархічність: Розподіл моделі на різні рівні деталізації, починаючи від загальних логістичних стратегій до конкретних операційних рішень.

– динамічність: Можливість врахування змінних умов та параметрів системи, таких як трафік, погодні умови, попит на перевезення та інші фактори.

– адаптивність: Здатність моделі адаптуватися до нових даних та змін у системі, забезпечуючи актуальність та точність прогнозів.

Види моделей транспортних систем

Моделі транспортних систем можна класифікувати за різними критеріями, включаючи рівень деталізації, типи рішень та методи моделювання. Основні види моделей включають:

– аналогові моделі: Включають фізичні моделі транспортних систем, такі як моделі доріг, мостів та інших елементів інфраструктури.

– дискретні моделі: Використовуються для моделювання систем з чітко визначеними станами та переходами між ними, наприклад, моделі черг, моделі руху транспортних засобів.

– безперервні моделі: Використовуються для моделювання систем з постійними змінами параметрів, такі як моделі потоків транспорту.

– гібридні моделі: Поєднують елементи дискретних та безперервних моделей для більш точного відтворення поведінки складних систем.

Методи моделювання транспортних систем

– імітаційне моделювання: Використовується для відтворення реальної поведінки транспортної системи під різними умовами. Цей метод дозволяє проводити експерименти з моделлю для оцінки впливу різних факторів та стратегій на ефективність системи.

– математичне програмування: Включає лінійне програмування, цілочисельне програмування та інші методи для оптимізації рішень у транспортних системах. Використовується для визначення оптимальних маршрутів, розподілу ресурсів та інших задач.

– агентне моделювання: Моделювання поведінки окремих агентів (наприклад, водіїв, транспортних засобів) та їх взаємодії у системі. Це дозволяє враховувати індивідуальні рішення та їх вплив на загальну систему.

– моделювання на основі правил: Використовує набір правил для визначення поведінки системи під різними умовами. Це підходить для задач, де необхідно враховувати складні взаємодії між елементами системи.

Моделювання транспортних систем є невід'ємною частиною сучасного управління логістикою та транспортними перевезеннями. Воно дозволяє створювати віртуальні копії реальних систем, досліджувати їх поведінку під різними умовами та приймати обґрунтовані рішення для підвищення ефективності транспортних операцій. Використання різних видів моделей, методів моделювання та спеціалізованого програмного забезпечення забезпечує точність та надійність отриманих результатів, що сприяє оптимізації логістичних процесів та зниженню витрат.

2.6 Оцінка ефективності транспортної маршрутизації

Оцінка ефективності транспортної маршрутизації дозволяє виявити слабкі місця в існуючих процесах, визначити можливості для їх покращення та підвищити загальну продуктивність транспортних систем. Цей процес включає аналіз різних критеріїв ефективності, використання інструментів та методів для збору та обробки даних, а також розробку рекомендацій для вдосконалення маршрутів. У цьому розділі розглядаються основні методи оцінки ефективності транспортної маршрутизації, критерії оцінки та інструменти, що використовуються для цієї мети.

Критерії оцінки ефективності

Для оцінки ефективності транспортної маршрутизації використовуються різні критерії, які можна розділити на економічні, екологічні та соціальні:

Економічні критерії:

– загальні витрати на перевезення: включають витрати на паливо, оплату праці водіїв, амортизацію транспортних засобів, ремонт та обслуговування.

– відстань маршрутів: загальна відстань, яку проходять транспортні засоби під час виконання маршрутів.

– час доставки: загальний час, необхідний для доставки вантажів до клієнтів, включаючи час на завантаження, розвантаження та транспортні затримки.

– використання ресурсів: ефективність використання транспортних засобів, включаючи вантажопідйомність та обсяги перевезень.

Екологічні критерії:

– викиди CO_2: кількість викидів вуглекислого газу, що утворюються під час перевезень.

– споживання палива: обсяг палива, використаного під час виконання маршрутів.

– енергетична ефективність: співвідношення між обсягом перевезених вантажів та витраченими енергоресурсами.

Соціальні критерії:

– рівень обслуговування клієнтів: задоволеність клієнтів точністю та своєчасністю доставки.

– безпека на дорогах: кількість аварій та інцидентів, що сталися під час виконання маршрутів.

– робочі умови водіїв: дотримання норм робочого часу, умов праці та відпочинку.

Методи оцінки ефективності

Аналіз витрат та вигод (Cost-Benefit Analysis, CBA): метод, що дозволяє оцінити економічну ефективність транспортної маршрутизації шляхом порівняння витрат на перевезення з вигодами, які отримуються від покращення маршрутів. Цей метод включає в себе аналіз прямого та непрямого впливу змін у маршрутизації на загальну вартість логістичних операцій.

Аналіз ефективності з використанням ключових показників ефективності (KPI): використання набору ключових показників ефективності для оцінки продуктивності транспортних операцій. До основних KPI відносяться витрати на перевезення, відстань маршрутів, час доставки та рівень обслуговування клієнтів. Аналіз KPI дозволяє виявити відхилення від встановлених норм та стандартів і приймати своєчасні коригувальні дії [23].

Імітаційне моделювання: метод, що дозволяє створити віртуальну копію транспортної системи та провести експерименти для оцінки впливу різних факторів на ефективність маршрутів. Імітаційне моделювання дозволяє враховувати динамічні зміни в системі та оцінювати різні сценарії. Цей підхід забезпечує гнучкість та можливість тестування різних стратегій без ризику для реальної системи.

Математичне програмування: використання методів лінійного та цілочисельного програмування для знаходження оптимальних рішень з урахуванням обмежень та критеріїв ефективності. Цей метод дозволяє визначити оптимальні маршрути та мінімізувати загальні витрати. Математичне програмування забезпечує

високу точність і надійність рішень, що є важливими для ефективного управління транспортними системами.

Аналіз чутливості: метод, що дозволяє оцінити вплив змін у вхідних параметрах на результативні показники моделі. Аналіз чутливості допомагає виявити критичні фактори, що впливають на ефективність транспортної системи, і розробити стратегії для мінімізації ризиків та підвищення стійкості системи до зовнішніх змін.

Інструменти для оцінки ефективності

Для оцінки ефективності транспортної маршрутизації використовуються різні програмні інструменти та платформи, які дозволяють збирати, обробляти та аналізувати дані:

Геоінформаційні системи (GIS):

Використовуються для візуалізації маршрутів, аналізу географічних даних та оцінки ефективності транспортних мереж. GIS дозволяє враховувати просторові аспекти при плануванні маршрутів та оцінювати вплив географічних особливостей на ефективність перевезень [20].

Програмне забезпечення для імітаційного моделювання:

Інструменти, такі як AnyLogic, MATLAB, SUMO, дозволяють створювати моделі транспортних систем та проводити експерименти для оцінки їх ефективності. Це програмне забезпечення забезпечує високий рівень деталізації та точності моделювання, що дозволяє аналізувати різні сценарії та оптимізувати маршрути.

Системи управління транспортом (TMS):

Інформаційні системи, що дозволяють автоматизувати процеси планування, моніторингу та аналізу транспортних операцій. TMS забезпечує ефективне управління ресурсами, зниження витрат та підвищення точності планування маршрутів. Ці системи також надають можливість інтеграції з іншими інформаційними системами та базами даних, що підвищує загальну ефективність управління транспортними операціями.

Аналітичні платформи:

Інструменти для аналізу великих даних, такі як Power BI, Tableau, які дозволяють візуалізувати дані, проводити аналіз та оцінку ефективності транспортних операцій. Аналітичні платформи забезпечують інтерактивні можливості для аналізу даних, що дозволяє швидко отримувати необхідну інформацію та приймати обґрунтовані рішення.

Рекомендації щодо підвищення ефективності транспортної маршрутизації

Використання сучасних технологій:

Впровадження новітніх технологій, таких як IoT (Internet of Things), GPS-моніторинг, Big Data та машинне навчання, для покращення збору та аналізу даних, що дозволяє більш точно планувати маршрути та оперативно реагувати на зміни.

Оптимізація процесів:

Вдосконалення процесів планування та управління маршрутами шляхом автоматизації рутинних операцій, використання алгоритмів оптимізації та імітаційного моделювання. Це дозволяє знизити витрати, покращити точність планування та підвищити рівень обслуговування клієнтів.

Підвищення кваліфікації персоналу:

Організація навчання та тренінгів для співробітників, відповідальних за управління транспортними операціями, з метою підвищення їх кваліфікації та ознайомлення з новітніми методами та інструментами управління.

Інтеграція з іншими системами:

Інтеграція систем управління транспортом з ERP (Enterprise Resource Planning) системами, CRM (Customer Relationship Management) системами та іншими інформаційними системами для забезпечення цілісного підходу до управління логістичними операціями.

Екологічна стійкість:

Розробка та впровадження екологічно стійких стратегій, що включають оптимізацію маршрутів для зниження викидів $CO2$, використання енергоефективних транспортних засобів та впровадження відновлюваних джерел енергії.

Оцінка ефективності транспортної маршрутизації є ключовим аспектом управління логістичними операціями, що дозволяє підвищити продуктивність транспортних систем, знизити витрати та покращити рівень обслуговування клієнтів. Використання різних критеріїв оцінки, методів та інструментів забезпечує комплексний підхід до аналізу та оптимізації маршрутів, що сприяє досягненню високих показників ефективності та стійкості транспортних систем. Розробка та впровадження рекомендацій щодо підвищення ефективності транспортної маршрутизації дозволяє забезпечити конкурентоспроможність логістичних компаній на ринку та сприяє сталому розвитку транспортних систем.

2.7 Висновки за розділом 2

У другому розділі даної роботи було проведено аналіз теоретико-методологічних основ багатокритеріальної транспортної маршрутизації. Було розглянуто різні підходи до вирішення задач транспортної маршрутизації, включаючи класичні методи, евристичні та метаевристичні методи, а також точні методи оптимізації.

Для подальшого експертного дослідження було обрано задачу маршрутизації транспортних засобів з часовими вікнами (VRPTW), яка є однією з найважливіших і найскладніших задач у логістиці. В рамках цього було обрано два підходи для подальшого проведення експертної оцінки та порівняння: генетичний алгоритм (GA) та алгоритм мурашиних колоній (ACO).

Алгоритм GVNS розглянуто ефективність у знаходженні високоякісних рішень для складних задач транспортної маршрутизації завдяки своїй здатності систематично змінювати околиці та уникати локальних мінімумів. Застосування цього метаевристичного методу дозволяє знайти близькі до оптимальних маршрути за обмежений час, що є особливо важливим у динамічних умовах реального світу.

GA забезпечує високу гнучкість і адаптивність завдяки використанню еволюційних механізмів природного відбору, кросоверу та мутації. Ці механізми дозволяють алгоритму знаходити нові рішення шляхом комбінування та зміни існуючих рішень. Крім того, GA має високу здатність до глобальної оптимізації, що

робить його ефективним у знаходженні глобальних оптимумів у великих і складних просторах пошуку, характерних для VRPTW. Еволюційний підхід GA дозволяє знаходити наближені оптимальні рішення навіть у ситуаціях з багатьма змінними та обмеженнями.

ACO, з іншого боку, моделює поведінку мурах у пошуку оптимальних шляхів, використовуючи феромонні сліди для направлення пошуку. Це природне моделювання дозволяє ACO ефективно вирішувати комівояжерські задачі та задачі маршрутизації транспортних засобів. Стихійний розподіл пошуку за допомогою феромонних слідів дозволяє ACO адаптивно знаходити оптимальні або наближені до оптимальних маршрути, що робить його особливо ефективним у розв'язанні задач з часовими вікнами. Феромонні сліди допомагають алгоритму швидко збирати інформацію про найкращі маршрути, що значно покращує ефективність пошуку.

Обидва алгоритми демонструють високу ефективність та адаптивність у вирішенні складних оптимізаційних задач, таких як VRPTW, завдяки своїм унікальним підходам до пошуку та оптимізації рішень. GA використовує еволюційні механізми для комбінування та модифікації рішень, тоді як ACO використовує колективну поведінку мурах і феромонні сліди для направлення пошуку, що забезпечує високу ефективність і адаптивність в умовах багатьох змінних та обмежень.

3 АНАЛІЗ ТА ЕКСПЕРИМЕНТАЛЬНА ОЦІНКА ЗАДАЧІ МАРШРУТИЗАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ З ЧАСОВИМИ ВІКНАМИ

3.1 Математична постановка задачі маршрутизації транспортних засобів з часовими вікнами

Задача маршрутизації транспортних засобів з часовими вікнами (Vehicle Routing Problem with Time Windows, VRPTW) (Рисунок 3.1) полягає у визначенні оптимальних маршрутів для транспортних засобів, які мають обслуговувати набір клієнтів з часовими вікнами для прийому товарів. Метою є мінімізація загальних витрат або відстані, при цьому дотримуючись всіх обмежень на час та вантажопідйомність транспортних засобів.

Рисунок 3.1. Vehicle Routing Problem with Time Windows

Основні компоненти математичної моделі:

$$\sum_{i=0}^{n}\sum_{j=0,j\neq i}^{n}\sum_{k=1}^{K} c_{ij}x_{ijk} \rightarrow min, \tag{2.1.1}$$

$$\sum_{i=0}^{n}\sum_{j=0,j\neq i}^{n}\sum_{k=1}^{K} t_{ij}\,x_{ijk} \rightarrow min, \tag{2.1.2}$$

$$\sum_{k=1}^{K}\sum_{j=1}^{n} x_{0jk} \leq 1, \tag{2.2}$$

$$\sum_{j=1}^{n} x_{0jk} = \sum_{j=1}^{n} x_{j0k} \leq 1, \text{ для } k\epsilon\overline{1,K}, \tag{2.3}$$

$$\sum_{k=1}^{K}\sum_{j=0,j\neq i}^{n} x_{ijk} = 1, \text{ для } i\epsilon\overline{1,n}, \tag{2.4}$$

$$\sum_{k=1}^{K}\sum_{i=0,i\neq j}^{n} x_{ijk} = 1, \text{ для } j\epsilon\overline{1,n}, \tag{2.5}$$

$$\sum_{i=1}^{N} q_i \sum_{j=0,j\neq i}^{N} x_{ijk} \leq Q, \quad \text{для } k\epsilon\overline{1,K}, \tag{2.6}$$

$$\sum_{i=1}^{n}\sum_{j=0,j\neq i}^{n} x_{ijk}\left(w_i + s_i + t_{ij}\right) \leq T_k, \text{ для } k\epsilon\overline{1,K}, \tag{2.7}$$

$$t_0 = w_0 = s_0 = 0, \tag{2.8}$$

$$x_{ijk}\left(t_i + w_i + s_i + t_{ij}\right) \leq t_j, \text{ для } i\epsilon\overline{1,n}, j\epsilon\overline{1,n}, (\ i \neq j), k\epsilon\overline{1,K}, \tag{2.9}$$

$$e_i \leq (t_i + w_i) \leq l_i, \text{ для } i \in \overline{1,n}, \tag{2.10}$$

Де відомими є:

c_{ij} – вартість проїзду з i в j, $(i,j) \in N$;

t_{ij} – тривалість переїзду з i в j, $(i,j) \in N$;

q_i – потреби в товарі пункту i, $i \in N$;

s_i – тривалість обслуговування в пункті i, $i \in N$;

e_i – початок часового вікна в пункті i, $i \in N$;

l_i – кінець часового вікна в пункті i, $i \in N$;

K – загальна кількість транспортних засобів;

T_k – максимальна тривалість (час) маршруту для k-го транспортного засобу, $k \in K$;

Q – місткість ТЗ;

$N, n+1$ – кількість пунктів, включаючи депо.

Змінними є:

t_i – час прибуття в пункт i, $i \in N$;

w_i – час очікування в пункті i, $i \in N$.

$$x_{ijk} = \begin{cases} 1, & \text{якщо маршрут } (i,j) \text{ був обслугован } k-\text{им ТЗ} \\ 0, & \text{в іншому випадку} \end{cases}.$$

Таким чином, зміст складових моделі такий:

(2.1.1) – цільова функція, що мінімізує витрати на перевезення;

(2.1.2) – цільова функція, що мінімізує час на перевезення;

(2.2) – k-ий ТЗ може виїхати із депо не більше одного разу;

(2.3) – якщо k-ий ТЗ виїхав з депо, то він має туди повернутися;

(2.4) – із i-го пункту k-им ТЗ можна виїхати лише один раз;

(2.5) – в j-ий пункт k-им ТЗ можна в'їхати лише один раз;

(2.6) – місткості ТЗ;

(2.7) – загальний час перебування в маршруті k-ого ТЗ не може перевищувати максимальної тривалості маршруту для k-го ТЗ;

(2.8) – початкові умови;

(2.9) – добуток часу приїзду в i-ий пункт, часу очікування в i-ому пункті, часу обслуговування в i-ому пункті та часу переїзду з i-го пункту в j-ий не може бути більшою за час приїзду в j-ий пункт;

(2.10) – час приїзду в i-ий пункт і час очікування в i-ому пункті мають попадати в часове вікно i-го пункту [5].

Пояснення математичної моделі

Математична модель VRPTW включає кілька ключових компонентів, кожен з яких відіграє важливу роль у формулюванні та вирішенні задачі.

Функція цілі: Метою є мінімізація загальної відстані або вартості, яку транспортні засоби повинні подолати для обслуговування всіх клієнтів. Це дозволяє знизити витрати на паливо, час роботи водіїв та загальні логістичні витрати.

Обмеження на відвідування: Ці обмеження гарантують, що кожен клієнт буде відвіданий лише один раз, що є критичним для забезпечення ефективного обслуговування. Вони дозволяють уникнути ситуацій, коли транспортний засіб повторно відвідує того самого клієнта, що могло б збільшити загальні витрати та неефективність логістичних операцій.

Обмеження на депо: Забезпечують, що кожен транспортний засіб починає і закінчує свій маршрут в депо, що є важливим для планування логістичних операцій та управління транспортними засобами. Це обмеження дозволяє забезпечити координацію роботи та ефективне використання транспортних засобів.

Обмеження на вантажопідйомність: Ці обмеження гарантують, що транспортні засоби не перевищуватимуть свою максимальну вантажопідйомність, забезпечуючи безпеку та ефективність перевезень. Вони враховують обсяг товарів, які можна перевозити кожним транспортним засобом, що є важливим для уникнення перевантаження та зниження ризиків, пов'язаних з транспортними перевезеннями.

Часові вікна: Забезпечують обслуговування кожного клієнта в рамках заданого часового вікна. Це особливо важливо для клієнтів з жорсткими часовими обмеженнями на прийом товарів. Врахування часових вікон дозволяє оптимізувати розклад доставки, уникнути запізнень та забезпечити високу якість обслуговування.

Час обслуговування і пересування: Ці обмеження забезпечують, що транспортний засіб прибуває до клієнта в правильний час, враховуючи час обслуговування та пересування між клієнтами.

Безперервність маршруту: Використовується для уникнення підтурів у маршрутах, що може призвести до неефективного використання транспортних засобів та збільшення загальних витрат. Це обмеження гарантує, що маршрути будуть логічно послідовними та ефективними, знижуючи загальні витрати на перевезення.

Завантаження транспортного засобу: Ці обмеження гарантують, що завантаження транспортного засобу враховується при обслуговуванні кожного клієнта. Це дозволяє ефективно управляти обсягами перевезень і уникати перевантаження, забезпечуючи безпеку та ефективність логістичних операцій [21].

Для проведення експериментального порівняння використовуємо данні з Solomon benchmark. Це стандартний набір тестових даних, створеним М. Соломоном у 1987 році для оцінки ефективності алгоритмів вирішення задач маршрутизації транспортних засобів з часовими вікнами (VRPTW). Цей набір даних став широко використовуваним у науковій спільноті для порівняння різних методів і алгоритмів, що вирішують VRPTW, завдяки своїй структурі та реалістичності.

Таблиця 3.1 – Дані з датасету R206 Solomon benchmark

CUSTNO	XCOORD	YCOORD	DEMAND	READY TIME	DUE DATE	SERVICE TIME
1	35	35	0	0	1000	0
2	41	49	10	0	974	10
3	35	17	7	0	972	10
4	55	45	13	0	967	10
5	55	20	19	620	860	10
6	15	30	26	0	969	10
7	25	30	3	345	585	10
8	20	50	5	0	968	10
9	10	43	9	323	563	10
10	55	60	16	329	569	10
11	30	60	16	485	725	10
12	20	65	12	146	386	10
13	50	35	19	0	975	10
14	30	25	23	639	879	10
15	15	10	20	32	272	10
16	30	5	8	118	358	10
17	10	20	19	203	443	10
18	5	30	2	682	922	10

Продовження таблиці 3.1 – Дані з датасету R206 Solomon benchmark

19	20	40	12	286	526	10
20	15	60	17	204	444	10
21	45	65	9	504	744	10
22	45	20	11	0	971	10
23	45	10	18	332	572	10
24	55	5	29	146	386	10
25	65	35	3	656	896	10
26	65	20	6	716	956	10
27	45	30	17	0	978	10
28	35	40	16	60	300	10
29	41	37	16	65	305	10
30	64	42	9	132	372	10
31	40	60	21	187	427	10
32	31	52	27	0	972	10
33	35	69	23	599	839	10
34	53	52	11	24	264	10
35	65	55	14	0	953	10
36	63	65	8	630	870	10
37	2	60	5	41	281	10
38	20	20	8	0	968	10
39	5	5	16	234	474	10
40	60	12	31	33	273	10
41	40	25	9	279	519	10
42	42	7	5	334	574	10
43	24	12	5	25	265	10
44	23	3	7	543	783	10
45	11	14	18	167	407	10
46	6	38	16	29	269	10
47	2	48	1	452	692	10
48	8	56	27	48	288	10
49	13	52	36	0	962	10
50	6	68	30	401	641	10

Параметри датасету:

- CUSTNO. (Customer Number): Номер клієнта, який використовується для ідентифікації кожного клієнта в датасеті.
- XCOORD (X Coordinate): X координата місцезнаходження клієнта на карті або площині.
- YCOORD (Y Coordinate): Y координата місцезнаходження клієнта на карті або площині.
- DEMAND (Demand): Кількість товару або вантажу, яку потрібно доставити цьому клієнту.
- READY TIME: Час, коли клієнт готовий прийняти доставку. Це мінімальний час, коли можна почати обслуговування клієнта.
- DUE DATE: Кінцевий термін, до якого має бути здійснена доставка. Порушення цього терміну може призвести до штрафних санкцій.
- SERVICE TIME: Час, необхідний для обслуговування клієнта на місці, включаючи час розвантаження товару.

Для розгляду саме багатокритеріального аналізу будемо розглядати дві цільові функції:

- мінімалізація пройденого шляху автомобілем.
- мінімізація часу в дорозі.

Дані в полях часу будемо сприймати як час в хвилинах. Час виїзду з депо припускаємо як 00:00. Середня швидкість автомобілів - 60 км/год.

Цільова функція встановленої задачі - знайти найкоротші та найшвидші маршрути переміщення автомобілів.

3.2 Опис та застосування генетичного алгоритму (GA)

Генетичні алгоритми (GA) є одним з найпопулярніших методів метаевристичної оптимізації, заснованих на принципах природного відбору і генетики. Вони використовуються для вирішення складних комбінаторних задач, де традиційні методи оптимізації можуть бути неефективними. У випадку задачі VRPTW (Vehicle Routing Problem with Time Windows), генетичний алгоритм може

знайти оптимальні або наближенні до оптимальних маршрути для транспортних засобів з урахуванням обмежень по часу для кожного клієнта [16].

Основні етапи генетичного алгоритму для VRPTW:

Завантаження та підготовка даних

Для демонстрації використання генетичного алгоритму для VRPTW задачі з використанням даних Solomon benchmark, скористаємося набором даних R206. Давайте завантажимо цей набір даних і підготуємо його до використання. Додамо ще депо з якого має виїжджати та повертатися транспорт.

Лістинг 3.1 - Програмний код функції з завантаження та підготовка даних

```
import numpy as np
import random

# Завантаження даних R206 з файлу
def load_solomon_data(filename):
    with open(filename, 'r') as file:
        lines = file.readlines()

    customers = []
    for line in lines[9:]:
        parts = line.split()
        customer = {
            'id': int(parts[0]),
            'x': float(parts[1]),
            'y': float(parts[2]),
            'demand': float(parts[3]),
            'ready_time': float(parts[4]),
            'due_time': float(parts[5]),
            'service_time': float(parts[6])
        }
        customers.append(customer)

    return customers

filename = 'R206.txt'
customers = load_solomon_data(filename)
depot = {'id': 0, 'x': 0, 'y': 0, 'demand': 0, 'ready_time': 0, 'due_time': 1000, 'service_time': 0}
```

Ініціалізація популяції

Генерується початкова популяція можливих рішень (маршрутів). Кожне рішення кодується у вигляді хромосоми. Хромосома представляє собою послідовність клієнтів, яких потрібно обслуговувати, з урахуванням вікон часу для кожного з них. Початкова популяція може бути згенерована випадковим чином або за допомогою евристичних методів.

Лістинг 3.2 - Програмний код функції ініціалізації популяції

```
def generate_initial_population(population_size, customers, num_vehicles):
    population = []
    for _ in range(population_size):
        routes = [[] for _ in range(num_vehicles)]
        for i, customer in enumerate(random.sample(customers, len(customers))):
                vehicle = i % num_vehicles  # Рівномірний розподіл клієнтів між транспортними засобами
            routes[vehicle].append(customer)
        for route in routes:
            route.insert(0, depot)  # Додавання депо на початку маршруту
            route.append(depot)     # Додавання депо в кінці маршруту
        population.append(routes)
    return population
```

Оцінка фітнес-функції

Фітнес-функція оцінює якість кожного маршруту. Вона може включати такі критерії, як загальна довжина маршруту, дотримання часових обмежень, загальна вартість маршруту тощо. Фітнес-функція може бути визначена, наприклад, як сума довжини маршруту та штрафу за порушення вікон часу:

Лістинг 3.3 - Програмний код функції розрахунку фітнес-функції

```
def calculate_distance(c1, c2):
    return np.sqrt((c1['x'] - c2['x'])**2 + (c1['y'] - c2['y'])**2)

def calculate_travel_time(distance, speed):
    return (distance / speed) * 60  # Переведення часу в хвилини

def calculate_fitness(chromosome, num_vehicles, speed, alpha=0.5):
    total_distance = 0
```

```
        total_travel_time = 0
        for vehicle in range(num_vehicles):
            route = chromosome[vehicle]
            if not route:  # Пропустити порожній маршрут
                continue
            current_time = 0
            for i in range(len(route) - 1):
                customer = route[i]
                next_customer = route[i + 1]
                travel_time = calculate_travel_time(calculate_distance(customer, next_customer), speed)
                current_time += travel_time
                if current_time < next_customer['ready_time']:
                    current_time = next_customer['ready_time']
                if current_time > next_customer['due_time']:
                    total_travel_time += 1000  # Великий штраф за порушення вікна часу
                current_time += next_customer['service_time']
                total_distance += calculate_distance(customer, next_customer)
                total_travel_time += travel_time + next_customer['service_time']
        # Фітнес-функція з урахуванням вагових коефіцієнтів для відстані та часу
        return alpha * total_distance + (1 - alpha) * total_travel_time
```

Вибір

Вибір батьків для створення нового покоління здійснюється на основі їх фітнес-значень. Найбільш поширені методи вибору включають рулетку (Roulette Wheel Selection) та турнірний відбір (Tournament Selection). У турнірному відборі випадковим чином обираються кілька хромосом, і з них вибирається найкраща.

Лістинг 3.4 - Програмний код функції вибору

```
def tournament_selection(population, fitnesses, tournament_size):
    selected = random.sample(list(zip(population, fitnesses)), tournament_size)
    selected.sort(key=lambda x: x[1])
    return selected[0][0]
```

Кросовер (схрещування)

Обрані батьківські хромосоми комбінуються для створення нових хромосом (потомків). Найбільш поширені оператори кросовера включають одноточковий (Single-Point Crossover) та двоточковий (Two-Point Crossover) кросовер. У

одноточковому кросовері вибирається одна точка на обох батьківських хромосомах, де вони розрізаються і обмінюються частинами.

Лістинг 3.5 - Програмний код функції схрещування

```
def single_point_crossover(parent1, parent2, num_vehicles):
    children = [[] for _ in range(num_vehicles)]
    for vehicle in range(num_vehicles):
        if len(parent1[vehicle]) > 1 and len(parent2[vehicle]) > 1:
            point = random.randint(1, min(len(parent1[vehicle]), len(parent2[vehicle])) - 1)
                child1 = parent1[vehicle][:point] + [x for x in parent2[vehicle] if x not in parent1[vehicle][:point]]
                child2 = parent2[vehicle][:point] + [x for x in parent1[vehicle] if x not in parent2[vehicle][:point]]
        else:
            child1 = parent1[vehicle]
            child2 = parent2[vehicle]
        child1.insert(0, depot)  # Додавання депо на початку маршруту
        child1.append(depot)     # Додавання депо в кінці маршруту
        child2.insert(0, depot)  # Додавання депо на початку маршруту
        child2.append(depot)     # Додавання депо в кінці маршруту
        children[vehicle] = child1
    return children
```

Мутація

Внесення невеликих змін до нових хромосом для підтримання генетичної різноманітності. Основні оператори мутації включають обмін (Swap Mutation) та інверсію (Inversion Mutation). При обміні дві позиції в хромосомі обмінюються місцями.

Лістинг 3.6 - Програмний код функції мутації

```
def swap_mutation(chromosome):
    for vehicle in range(len(chromosome)):  # Переконуємось, що обробляємо всі транспортні засоби в хромосомі
        if len(chromosome[vehicle]) > 3:  # Більше ніж 3, оскільки є два депо
            point1, point2 = random.sample(range(1, len(chromosome[vehicle]) - 1), 2) # Уникаємо депо
                chromosome[vehicle][point1], chromosome[vehicle][point2] = chromosome[vehicle][point2], chromosome[vehicle][point1]
    return chromosome
```

Заміна

Нові хромосоми замінюють частину старих хромосом у популяції. Можливі стратегії заміни включають повну заміну (Full Replacement) та часткову заміну з елітизмом (Elitism). Елітизм передбачає збереження деякої кількості найкращих хромосом зі старої популяції.

Лістинг 3.7 - Програмний код функції заміни

```
def replace_population(population, offspring, elitism_count, num_vehicles):
    population.sort(key=lambda x: calculate_fitness(x, num_vehicles))
    next_generation = population[:elitism_count] + offspring
    return next_generation
```

Зупинка

Алгоритм завершується, коли досягається задана кількість поколінь або не покращується фітнес-значення протягом певної кількості поколінь. Критерії зупинки можуть бути різними, залежно від специфіки задачі та вимог до точності результату [22].

Лістинг 3.7 - Програмний код функції зупинки

```
def genetic_algorithm(customers, population_size, generations, elitism_count, tournament_size, num_vehicles, speed, alpha=0.5):
    population = generate_initial_population(population_size, customers, num_vehicles)
    for generation in range(generations):
        fitnesses = [calculate_fitness(chromosome, num_vehicles, speed, alpha) for chromosome in population]
        new_population = []
        for _ in range(population_size // 2):
            parent1 = tournament_selection(population, fitnesses, tournament_size)
            parent2 = tournament_selection(population, fitnesses, tournament_size)
            children = single_point_crossover(parent1, parent2, num_vehicles)
            for child in children:
                if len(child) == num_vehicles:  # Переконатися, що кількість транспортних засобів правильна
                    mutated_child = swap_mutation(child)
                    new_population.append(mutated_child)
        population = replace_population(population, new_population, elitism_count, num_vehicles, speed, alpha)
        best_solution = min(population, key=lambda x: calculate_fitness(x, num_vehicles, speed, alpha))
    return best_solution
```

Візуалізація результатів

Для візуалізації здійснюється за допомогою бібліотеки matplotlib.pyplot. Будуємо графік на якому відображені точки клієнтів (X та Y координати). Та побудовані оптимальні маршрути для кожного з автомобілів.

Лістинг 3.8 - Програмний код функції візуалізації результатів оптимального маршруту

```
import matplotlib.pyplot as plt
def calculate_route_distance(route):
    total_distance = 0
    for i in range(len(route) - 1):
        total_distance += calculate_distance(route[i], route[i + 1])
    return total_distance

def calculate_route_time(route, speed):
    total_time = 0
    current_time = 0
    for i in range(len(route) - 1):
        travel_time = calculate_travel_time(calculate_distance(route[i], route[i + 1]),
speed)
        current_time += travel_time
        next_customer = route[i + 1]
        if current_time < next_customer['ready_time']:
            current_time = next_customer['ready_time']
        if current_time > next_customer['due_time']:
            current_time += 1000  # Великий штраф за порушення вікна часу
        current_time += next_customer['service_time']
        total_time += travel_time + next_customer['service_time']
    return total_time / 60  # Конвертувати в години

def plot_solution(solution, num_vehicles):
    colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k']
    plt.figure(figsize=(10, 10))
    for vehicle in range(num_vehicles):
        x_coords = [customer['x'] for customer in solution[vehicle]]
        y_coords = [customer['y'] for customer in solution[vehicle]]
        plt.plot(x_coords, y_coords, colors[vehicle % len(colors)] + 'o-')
        for i, customer in enumerate(solution[vehicle]):
            plt.text(customer['x'], customer['y'], str(customer['id']), fontsize=12)
    plt.xlabel('X координати')
    plt.ylabel('Y координати')
```

```
plt.title('Оптимальний маршрут VRPTW')
plt.show()
```

Результати побудови оптимальних маршрутів для 5 автомобілів відображені у таблиці 3.2 та Рисунку 3.2.

Таблиця 3.2 – Вихідні дані застосування алгоритму генетичного алгоритму

Автомобіль №	Маршрут	Довжина маршруту (км)	Час маршруту (годин)
1	[0, 6, 39, 48, 8, 42, 47, 36, 20, 25, 34, 0]	493.10	13.99
2	[0, 7, 35, 9, 45, 49, 31, 33, 30, 12, 46, 0]	406.20	11.82
3	[0, 23, 24, 19, 26, 21, 2, 13, 28, 27, 29, 0]	331.64	9.96
4	[0, 43, 50, 10, 38, 18, 40, 4, 17, 3, 22, 0]	434.07	12.52
5	[0, 15, 41, 37, 32, 11, 5, 14, 16, 1, 44, 0]	317.44	9.44
Загальна довжина		1982.45	57.73

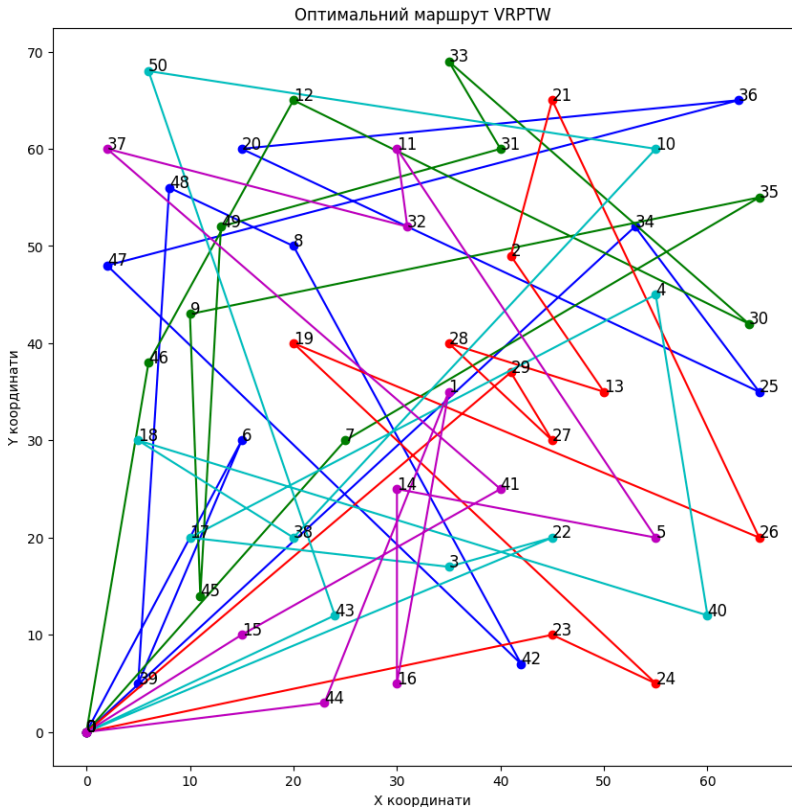

Рисунок 3.2 – Візуалізація оптимального маршруту побудованого генетичним алгоритмом

Генетичні алгоритми (GA) є ефективним методом для вирішення складних задач оптимізації, зокрема багатокритеріальних задач маршрутизації транспортних засобів з часовими вікнами (VRPTW). Вони імітують природний процес еволюції, використовуючи механізми селекції, схрещування та мутації для пошуку оптимальних або близьких до оптимальних рішень. Однією з головних переваг генетичних алгоритмів є здатність до глобального пошуку в просторі рішень, що допомагає уникнути застрягання в локальних мінімумах. Це особливо важливо для складних задач, де простір пошуку має багато локальних оптимумів.

Генетичні алгоритми також легко паралелізуються, оскільки оцінка придатності кожного індивідуума в популяції може виконуватися незалежно. Це робить їх придатними для великих і складних задач, таких як VRPTW. Вони можуть адаптуватися до різних типів задач та їх специфічних вимог завдяки використанню операцій мутації та схрещування. Ця властивість робить їх універсальним інструментом для різних проблем оптимізації. Генетичні алгоритми можуть бути інтегровані з іншими алгоритмами оптимізації, такими як локальні пошуки або методи ройового інтелекту, для покращення якості рішень та прискорення процесу оптимізації.

Проте, генетичні алгоритми мають і свої недоліки. Вони вимагають налаштування багатьох параметрів, таких як розмір популяції, ймовірності схрещування та мутації, що може бути складним і часозатратним процесом. Крім того, генетичні алгоритми можуть бути обчислювально інтенсивними, особливо для великих популяцій і складних задач, що потребує значних ресурсів для їх ефективного виконання. Незважаючи на ці виклики, генетичні алгоритми залишаються потужним і гнучким інструментом для оптимізації, здатним знаходити високоякісні рішення для складних задач у різних сферах застосування.

3.3 Опис та застосування алгоритму мурашиних колоній (ACO)

Алгоритм оптимізації колонії мурах (ACO) є потужним інструментом для вирішення різноманітних задач комбінаторної оптимізації, включаючи задачі маршрутизації транспортних засобів з часовими вікнами (VRPTW). Ці задачі є особливо важливими в логістиці та управлінні транспортом, де необхідно ефективно планувати маршрути для транспортних засобів з урахуванням численних обмежень та критеріїв.

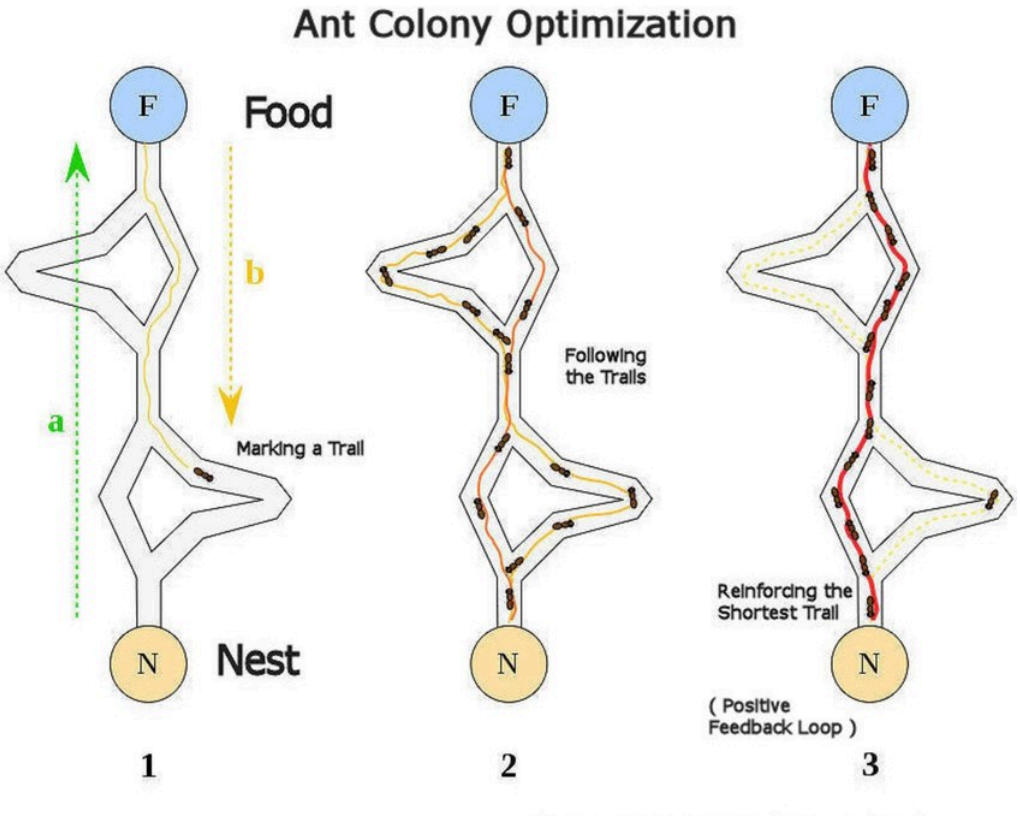

Рисунок 3.3 - Схематичне зображення мурашиних колоній

Основні компоненти ACO для VRPTW:

Представлення рішення:

Кожна мураха представляє можливе рішення у вигляді послідовності відвідуваних клієнтів. Маршрут визначається шляхом вибору ребер на графі, де вершини представляють клієнтів, а ребра – можливі шляхи між ними [17].

Феромонні сліди:

Мурахи залишають феромонні сліди на маршрутах, які вони використовують, що допомагає іншим мурахам знаходити ефективні шляхи. Кількість феромону збільшується на маршрутах з кращими характеристиками (менша відстань, відповідність часовим вікнам).

Функція цільової оцінки:

Включає різні критерії, такі як загальна відстань, дотримання часових вікон, мінімізація витрат та інших обмежень. Можлива багатокритеріальна оцінка, яка дозволяє враховувати декілька цілей одночасно.

Правила переходу:

Визначають ймовірність переходу мурахи від одного клієнта до іншого, враховуючи феромонні сліди та евристичну інформацію (наприклад, відстань між клієнтами, час у дорозі).

Оновлення феромонів:

Здійснюється шляхом випаровування старих феромонів та додавання нових на основі якості знайдених рішень. Враховується також вплив найкращих рішень для підвищення ефективності пошуку.

Основні етапи алгоритму мурашиних колоній:

Для демонстрації використання генетичного алгоритму для VRPTW задачі з використанням даних Solomon benchmark, скористаємося набором даних R206. Давайте завантажимо цей набір даних і підготуємо його до використання. Додамо ще депо з якого має виїжджати та повертатися транспорт.

Лістинг 3.9 - Програмний код функції з завантаження та підготовка даних

```
import numpy as np
import random
# Завантаження даних R206 з файлу
def load_solomon_data(filename):
    with open(filename, 'r') as file:
        lines = file.readlines()

    customers = []
    for line in lines[9:]:
```

```
            parts = line.split()
            customer = {
                'id': int(parts[0]),
                'x': float(parts[1]),
                'y': float(parts[2]),
                'demand': float(parts[3]),
                'ready_time': float(parts[4]),
                'due_time': float(parts[5]),
                'service_time': float(parts[6])
            }
            customers.append(customer)
        return customers
    filename = 'R206.txt'
    customers = load_solomon_data(filename)
    depot = {'id': 0, 'x': 0, 'y': 0, 'demand': 0, 'ready_time': 0, 'due_time': 1000, 'service_time': 0}
```

Ініціалізація

Першим етапом алгоритму є ініціалізація параметрів та феромонів. Тут ми визначаємо кількість мурах, кількість ітерацій, швидкість випаровування феромонів, а також вагові коефіцієнти для феромонів та евристики. Ми також ініціалізуємо матрицю феромонів, де кожен елемент представляє феромон на ребрі між двома клієнтами. Початкові значення феромонів встановлюються на деяке значення, зазвичай 1.0, що забезпечує рівні шанси для всіх маршрутів на початку.

Лістинг 3.10 - Програмний код функції з ініціалізації феромонів

```
    def calculate_distance(c1, c2):
        return np.sqrt((c1['x'] - c2['x'])**2 + (c1['y'] - c2['y'])**2)
    def calculate_travel_time(distance, speed):
        return (distance / speed) * 60  # Переведення часу в хвилини
    def initialize_pheromones(num_customers, initial_pheromone_level=1.0):
                    pheromones = np.full((num_customers, num_customers), initial_pheromone_level)
        np.fill_diagonal(pheromones, 0)
        return pheromones
    # Параметри алгоритму мурашиних колоній
    num_vehicles = 5  # Вибір кількості машин
    speed = 40  # Швидкість у одиницях на годину
    alpha = 1.0  # Вплив феромонів
    beta = 2.0  # Вплив евристики
```

```
decay = 0.5  # Швидкість випаровування феромонів
iterations = 100  # Кількість ітерацій
num_ants = 10  # Кількість мурах
# Ініціалізація феромонів
num_customers = len(customers) + 1
pheromones = initialize_pheromones(num_customers)
```

Розрахунок рішення

На цьому етапі кожна мураха будує своє рішення, починаючи з депо та поступово додаючи клієнтів до маршруту. Для цього використовується ймовірнісне правило переходу, яке враховує феромонові сліди та евристичну інформацію, таку як відстань до наступного клієнта. Ймовірність переходу до наступного клієнта розраховується на основі феромонових слідів та оберненої відстані, що дозволяє мурахам досліджувати різні шляхи та знаходити оптимальні маршрути.

Лістинг 3.11 - Програмний код функції з розрахунок рішень

```
def select_next_customer(current_customer, unvisited, pheromones, distances, alpha, beta):
    pheromone_levels = np.array([pheromones[current_customer['id']][i['id']] for i in unvisited])
    heuristic_values = np.array([1.0 / distances[current_customer['id']][i['id']] for i in unvisited])
    probabilities = (pheromone_levels ** alpha) * (heuristic_values ** beta)
    probabilities /= probabilities.sum()
    return np.random.choice(unvisited, p=probabilities)

def construct_solution(customers, depot, pheromones, distances, num_vehicles, speed, alpha, beta):
    vehicles = [[] for _ in range(num_vehicles)]
    unvisited = customers[:]
    for vehicle in vehicles:
        current_time = 0
        current_customer = depot
        vehicle.append(current_customer)
        while unvisited:
            next_customer = select_next_customer(current_customer, unvisited, pheromones, distances, alpha, beta)
            travel_time = calculate_travel_time(calculate_distance(current_customer, next_customer), speed)
            arrival_time = current_time + travel_time
```

```
                if arrival_time < next_customer['ready_time']:
                    arrival_time = next_customer['ready_time']
                if arrival_time > next_customer['due_time']:
                    break
                current_time = arrival_time + next_customer['service_time']
                vehicle.append(next_customer)
                unvisited.remove(next_customer)
                current_customer = next_customer
            vehicle.append(depot)
            # Якщо залишились необслуговані клієнти, додати їх до останнього маршруту
        for customer in unvisited:
            vehicles[-1].insert(-1, customer)
        return vehicles
    distances = np.array([[calculate_distance(i, j) for j in [depot] + customers] for i in [depot] + customers])
    solutions = [construct_solution(customers, depot, pheromones, distances, num_vehicles, speed, alpha, beta) for _ in range(num_ants)]
```

Оновлення феромонів

Після побудови рішень усіма мурахами феромонові сліди оновлюються. Це включає випаровування феромонів на всіх маршрутах, що зменшує їх значення на певний коефіцієнт, а також додавання феромонів на маршрути, пройдені мурахами, пропорційно до якості знайдених рішень. Цей процес забезпечує підсилення кращих рішень, сприяючи їх подальшому вибору іншими мурахами у наступних ітераціях.

Лістинг 3.12 - Програмний код функції оновлення феромонів

```
    def update_pheromones(pheromones, solutions, distances, decay, Q=100):
        pheromones *= (1 - decay)
        for solution in solutions:
            for route in solution:
                route_distance = sum(calculate_distance(route[i], route[i + 1]) for i in range(len(route) - 1))
                pheromone_deposit = Q / route_distance
                for i in range(len(route) - 1):
                    pheromones[route[i]['id']][route[i + 1]['id']] += pheromone_deposit
    update_pheromones(pheromones, solutions, distances, decay)
```

Ітерації

Цей етап включає повторення процесу конструкції рішень та оновлення феромонів до досягнення критерію зупинки (наприклад, кількість ітерацій або час виконання). Кожна ітерація включає побудову нових маршрутів мурахами на основі оновлених феромонів та евристики, а також оновлення феромонів на основі нових знайдених рішень. Цей цикл продовжується до досягнення оптимального або близького до оптимального рішення.

Лістинг 3.13 - Програмний код функції ітерації

```
def aco_algorithm(customers, depot, num_vehicles, speed, alpha=1.0, beta=2.0, decay=0.5, iterations=100, num_ants=10):
    num_customers = len(customers) + 1
    pheromones = initialize_pheromones(num_customers)
    distances = np.array([[calculate_distance(i, j) for j in [depot] + customers] for i in [depot] + customers])
    best_solution = None
    best_distance = float('inf')
    for iteration in range(iterations):
        solutions = []
        for ant in range(num_ants):
            solution = construct_solution(customers, depot, pheromones, distances, num_vehicles, speed, alpha, beta)
            solutions.append(solution)
            route_distance = sum(sum(calculate_distance(solution[i][j], solution[i][j + 1]) for j in range(len(solution[i]) - 1)) for i in range(num_vehicles))
            if route_distance < best_distance:
                best_distance = route_distance
                best_solution = solution
        update_pheromones(pheromones, solutions, distances, decay)
    return best_solution
best_solution = aco_algorithm(customers, depot, num_vehicles, speed, alpha, beta, decay, iterations, num_ants)
```

Візуалізація результатів

Після завершення алгоритму оптимальне знайдене рішення можна візуалізувати для оцінки його ефективності. Візуалізація включає графічне представлення маршрутів кожної машини на площині, що дозволяє побачити, як мурахи обирали свої шляхи та як ці шляхи змінювалися протягом ітерацій.

Результати побудови оптимальних маршрутів для 5 автомобілів відображені у таблиці 3.3 та Рисунку 3.4.

Лістинг 3.14 - Програмний код функції візуалізації

```
def plot_solution(solution, num_vehicles):
    colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k']
    plt.figure(figsize=(10, 10))
    for vehicle in range(num_vehicles):
        x_coords = [customer['x'] for customer in solution[vehicle]]
        y_coords = [customer['y'] for customer in solution[vehicle]]
        plt.plot(x_coords, y_coords, colors[vehicle % len(colors)] + 'o-')
        for i, customer in enumerate(solution[vehicle]):
            plt.text(customer['x'], customer['y'], str(customer['id']), fontsize=12)
    plt.xlabel('X координати')
    plt.ylabel('Y координати')
    plt.title('Оптимальний маршрут VRPTW')
    plt.show()
plot_solution(best_solution, num_vehicles)
def calculate_route_distance(route):
    total_distance = 0
    for i in range(len(route) - 1):
        total_distance += calculate_distance(route[i], route[i + 1])
    return total_distance
def calculate_route_time(route, speed):
    total_time = 0
    current_time = 0
    for i in range(len(route) - 1):
        travel_time = calculate_travel_time(calculate_distance(route[i], route[i + 1]),
speed)
        current_time += travel_time
        next_customer = route[i + 1]
        if current_time < next_customer['ready_time']:
            current_time = next_customer['ready_time']
        if current_time > next_customer['due_time']:
            current_time += 1000  # Великий штраф за порушення вікна часу
        current_time += next_customer['service_time']
        total_time += travel_time + next_customer['service_time']
    return total_time / 60  # Конвертувати в години
# Виведення оптимальних маршрутів для кожної автівки та їх довжини
total_distance_all_routes = 0
total_time_all_routes = 0
for vehicle in range(num_vehicles):
    route = [customer['id'] for customer in best_solution[vehicle]]
```

```
        route_distance = calculate_route_distance(best_solution[vehicle])
        route_time = calculate_route_time(best_solution[vehicle], speed)
        total_distance_all_routes += route_distance
        total_time_all_routes += route_time
        print(f"Маршрут для автівки {vehicle + 1}: {route} з довжиною маршруту
{route_distance:.2f} одиниць та часом в дорозі {route_time:.2f} годин")
    # Виведення загальної довжини всіх маршрутів та загального часу роботи
    print(f"Загальна довжина всіх маршрутів: {total_distance_all_routes:.2f}
одиниць")
    print(f"Загальний час роботи всіх маршрутів: {total_time_all_routes:.2f} годин")
```

Таблиця 3.3 – Вихідні дані застосування алгоритму мурашиних колоній

Автомобіль №	Маршрут	Довжина маршруту (км)	Час маршруту (годин)
1	[0, 39, 45, 17, 6, 7, 19, 9, 47, 49, 8, 0]	149.04	5.39
2	[0, 15, 18, 0]	70.80	2.10
3	[0, 44, 38, 14, 3, 0]	99.98	3.17
4	[0, 43, 16, 42, 23, 24, 25, 13, 2, 0]	190.80	6.10
5	[0, 40, 41, 22, 26, 5, 27, 1, 4, 10, 11, 12, 20, 21, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 46, 48, 50, 0]	604.57	19.28
Загальна довжина		1115.19	36.05

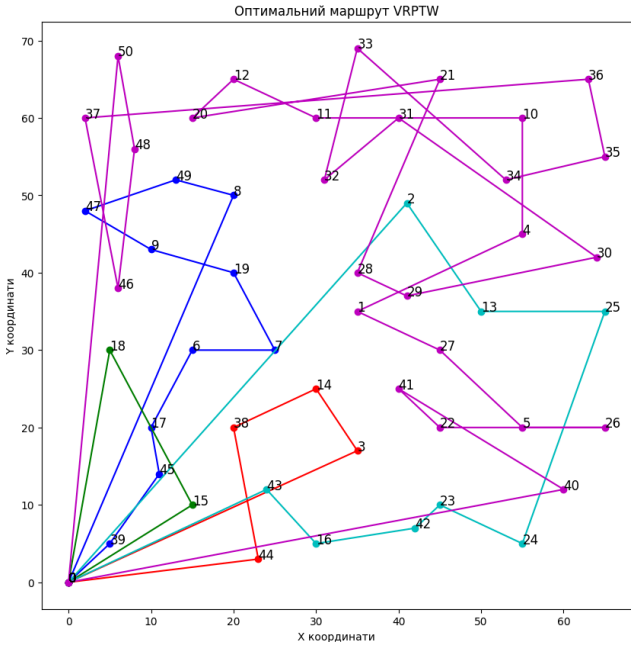

Рисунок 3.3 – Візуалізація оптимального маршруту побудованого ACO

Алгоритм оптимізації колонії мурах (ACO) є ефективним методом для вирішення складних задач оптимізації, зокрема багатокритеріальних задач маршрутизації транспортних засобів з часовими вікнами (VRPTW). Цей алгоритм натхненний поведінкою реальних мурах, які використовують феромонові сліди для знаходження найкоротших шляхів до джерел їжі. Однією з головних переваг ACO є його здатність до глобального пошуку в просторі рішень, що дозволяє уникнути застрягання в локальних оптимумах. Це досягається завдяки механізму випаровування феромонів, який запобігає надмірному накопиченню феромонів на одному маршруті і стимулює дослідження нових шляхів.

Алгоритм ACO також відзначається своєю гнучкістю та адаптивністю. Його можна легко налаштувати для різних типів задач, змінюючи параметри, такі як коефіцієнти феромону і евристичної інформації. Крім того, ACO може бути поєднаний з іншими методами оптимізації, такими як локальні пошуки або генетичні алгоритми, для покращення якості рішень і підвищення швидкості збіжності. Завдяки паралельному характеру обчислень, ACO може ефективно використовуватися в складних задачах, де потрібно швидке знаходження рішень.

Проте алгоритм ACO має і свої недоліки. Вибір відповідних параметрів, таких як коефіцієнт випаровування феромонів та вагові коефіцієнти, може бути складним і вимагати експериментальної наладки. Крім того, алгоритм може бути обчислювально інтенсивним, особливо для великих проблемних областей, що потребує значних ресурсів для його виконання. Також існує ризик передчасної збіжності до субоптимальних рішень, якщо феромонові сліди занадто швидко концентруються на одному маршруті.

Загалом, алгоритм оптимізації колонії мурах є гнучким інструментом для вирішення задач оптимізації. Його здатність до глобального пошуку, адаптивність та можливість паралельного виконання роблять його ефективним методом для знаходження високоякісних рішень у складних і багатокритеріальних задачах, таких як VRPTW. Незважаючи на деякі виклики, пов'язані з налаштуванням і обчислювальними витратами, ACO залишається одним з провідних підходів у сфері комбінаторної оптимізації.

3.4 Порівняння результатів

Після проведення експериментальних досліджень генетичного алгоритму та алгоритму мурашиних колоній можна зробити такі висновки:

Загальна довжина маршрутів:

Генетичний алгоритм показав загальну довжину маршрутів 1982.45 км, що значно перевищує результати алгоритму мурашиних колоній, де загальна довжина маршрутів становила 1115.19 км. Це свідчить про те, що ACO більш ефективно мінімізував загальну довжину маршрутів у цій задачі.

Загальний час маршруту:

Генетичний алгоритм також показав значно більший загальний час маршруту, який становить 57.73 години, порівняно з 36.05 годинами для алгоритму мурашиних колоній. Це ще раз підкреслює перевагу ACO у знаходженні більш оптимальних рішень щодо загального часу в дорозі.

Розподіл маршрутів між транспортними засобами:

Обидва алгоритми розподілили клієнтів між п'ятьма автомобілями, але ACO показав кращу збалансованість маршрутів. Найдовший маршрут у ACO мав довжину 604.57 км і займав 19.28 години, тоді як генетичний алгоритм мав значно довші окремі маршрути, наприклад, 493.10 км і 13.99 години для першого автомобіля.

Ефективність та збіжність:

ACO продемонстрував кращу ефективність і швидшу збіжність до оптимальних рішень у цій задачі. Це пов'язано з механізмом феромонів, який дозволяє швидше адаптуватися до оптимальних маршрутів, тоді як генетичний алгоритм може вимагати більше ітерацій для досягнення подібних результатів.

3.5 Висновки за розділом 3

Аналіз та експериментальна оцінка задачі маршрутизації транспортних засобів з часовими вікнами (VRPTW) за допомогою генетичного алгоритму (GA) та

алгоритму мурашиних колоній (ACO) дозволяють зробити важливі висновки про відмінності, переваги та недоліки кожного з методів.

Відмінності між методами

Механізми оптимізації:

Генетичний алгоритм (GA): Заснований на принципах природної еволюції, GA використовує механізми селекції, схрещування та мутації для створення нових поколінь рішень. Вибір індивідуумів базується на їхній придатності (фітнес-функції).

Алгоритм мурашиних колоній (ACO): Імітує поведінку мурах, які залишають феромонові сліди на шляху до їжі. Мурахи вибирають наступні кроки на основі ймовірностей, які залежать від феромонових слідів і евристичної інформації (наприклад, відстані).

Підхід до пошуку рішень:

GA: Виконує глобальний пошук за рахунок створення різноманітних рішень і їх еволюції через багато поколінь.

ACO: Виконує як глобальний, так і локальний пошук, використовуючи феромонові сліди для підсилення перспективних маршрутів.

Переваги та недоліки:

Генетичний алгоритм (GA)

Переваги:

— Глобальний пошук: Завдяки еволюційним механізмам GA здатний знаходити глобальні оптимуми, уникаючи застрягання в локальних мінімумах.

— Гнучкість: Легко адаптується до різних типів задач оптимізації шляхом налаштування фітнес-функції та операторів генетичних маніпуляцій (селекція, схрещування, мутація).

— Інтеграція з іншими методами: GA можна інтегрувати з іншими оптимізаційними методами, такими як локальні пошуки, для покращення результатів.

Недоліки:

— Обчислювальна інтенсивність: GA вимагає значних обчислювальних ресурсів, особливо для великих популяцій і складних задач.

— Налаштування параметрів: Вибір відповідних параметрів (розмір популяції, ймовірності схрещування та мутації) може бути складним і вимагати численних експериментів.

— Швидкість збіжності: Може вимагати багато поколінь для досягнення оптимальних рішень, що впливає на швидкість збіжності.

Алгоритм мурашиних колоній (ACO)

Переваги:

— Ефективність у комбінаторних задачах: ACO особливо ефективний для задач, де потрібно знайти найкоротший шлях, таких як VRPTW.

— Швидка збіжність: Механізм феромонових слідів забезпечує швидку адаптацію до оптимальних маршрутів.

— Гнучкість: ACO можна легко адаптувати до різних умов задач, змінюючи параметри феромонів і евристичної інформації.

— Локальний та глобальний пошук: Завдяки феромонам ACO забезпечує ефективний баланс між локальним і глобальним пошуком.

Недоліки:

— Налаштування параметрів: Потребує налаштування кількох параметрів, таких як коефіцієнт випаровування феромонів, вагові коефіцієнти для феромонів і евристики.

— Передчасна збіжність: Існує ризик передчасної збіжності до субоптимальних рішень, якщо феромонові сліди занадто швидко концентруються на одному маршруті.

— Обчислювальні витрати: Як і у випадку з GA, обчислювальні витрати можуть бути високими для великих і складних задач.

Порівняння результатів GA і ACO для задачі VRPTW показало, що ACO досягає кращих результатів з точки зору загальної довжини маршрутів та загального часу в дорозі. ACO продемонстрував більш ефективний розподіл клієнтів між

транспортними засобами, що призвело до зниження загальних витрат. Генетичний алгоритм, хоча і є потужним інструментом, виявився менш ефективним у цій задачі, вимагаючи більше часу та ітерацій для досягнення подібних результатів.

Загалом, алгоритм мурашиних колоній виявився більш ефективним та швидким у вирішенні задачі VRPTW порівняно з генетичним алгоритмом. ACO забезпечує кращі результати за менший час, що робить його переважним вибором для задач маршрутизації з численними обмеженнями. Водночас генетичний алгоритм залишається потужним інструментом, який може бути корисним у поєднанні з іншими методами для покращення результатів у складних задачах оптимізації.

ВИСНОВКИ

Проведений аналіз предметної галузі показав, що транспортна маршрутизація є ключовою складовою логістичних процесів, яка має вирішальний вплив на ефективність та економічність перевезень. Основні критерії, які враховуються при розв'язанні задач транспортної маршрутизації, включають мінімізацію загальної відстані маршрутів, скорочення часу перевезень, оптимальне використання транспортних засобів, дотримання часових вікон для обслуговування клієнтів, а також мінімізацію витрат на паливо та експлуатацію транспортних засобів. На основі цього аналізу було сформульовано задачу багатокритеріальної оптимізації транспортної маршрутизації з часовими вікнами (VRPTW), яка враховує наведені критерії та обмеження.

У теоретико-методологічних основах було розглянуто два основні методи розв'язання задач транспортної маршрутизації: генетичний алгоритм (GA) та алгоритм мурашиних колоній (ACO). Генетичний алгоритм, заснований на еволюційних принципах, використовує механізми селекції, схрещування та мутації для пошуку оптимальних рішень. Алгоритм мурашиних колоній імітує поведінку мурах, які залишають феромонові сліди на шляху до їжі, що допомагає знаходити найкоротші маршрути. Обидва методи мають свої переваги та недоліки, які були детально проаналізовані для визначення їхньої ефективності в задачах VRPTW.

За проведеним аналізом та експериментальною оцінкою було встановлено, що алгоритм мурашиних колоній (ACO) демонструє кращу ефективність у розв'язанні задач VRPTW порівняно з генетичним алгоритмом (GA). Результати показали, що ACO досягає меншої загальної довжини маршрутів (1115.19 км) та часу маршруту (36.05 годин) порівняно з GA, де загальна довжина маршрутів склала 1982.45 км, а час маршруту - 57.73 годин. Це свідчить про те, що ACO більш ефективно використовує феромонові сліди для швидкої збіжності до оптимальних рішень, забезпечуючи кращий розподіл клієнтів між транспортними засобами.

Генетичний алгоритм, хоча і є потужним інструментом для глобального пошуку, вимагає більше часу та обчислювальних ресурсів для досягнення аналогічних результатів. Налаштування параметрів GA, таких як розмір популяції та ймовірності схрещування та мутації, є складним процесом, який потребує ретельного підходу та численних експериментів.

На основі проведених досліджень та експериментальних результатів можна зробити висновок, що алгоритм мурашиних колоній є більш придатним та ефективним методом для вирішення задач транспортної маршрутизації з численними обмеженнями. Він забезпечує швидшу збіжність до оптимальних рішень, менші загальні витрати та краще збалансовані маршрути. Генетичний алгоритм, незважаючи на свою універсальність та можливість інтеграції з іншими методами, показав менш ефективні результати у цій задачі.

Практичне застосування розроблених методів та алгоритмів дозволить підвищити ефективність логістичних процесів, знизити витрати на перевезення та покращити обслуговування клієнтів. Результати дослідження можуть бути використані у транспортних та логістичних компаніях для оптимізації маршрутів та підвищення конкурентоспроможності на ринку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Міністерство та Комітет цифрової трансформації України. Перший у 2020 році OpenData Campus відбувся у Харкові [Електронний ресурс] / Міністерство та Комітет цифрової трансформації України. – 2024. – Режим доступу до ресурсу: https://thedigital.gov.ua/news/pershiy-u-2020-rotsi-opendata-campus-vidbuvsya-u-kharkovi.

2. TAPAS. Open Data Campus. Харків [Електронний ресурс] / TAPAS. – 2024. – Режим доступу: http://tapas.org.ua/media/open-data-campus-kharkiv/.

3. Ісакій К.Г. Дослідження методів багатокритеріальних задач транспортної маршрутизації. // World science: problems, prospects and innovations. Abstracts of the 3rd International scientific and practical conference. Perfect Publishing. Toronto, Canada. 2024. Pp. 322-327. URL: https://sci-conf.com.ua/x-mizhnarodna-naukovo-praktichna-konferentsiya-modern-research-in-science-and-education-29-31-05-2024-chikago-ssha-arhiv/.

4. Eduardo, U., Diego, P. (2017). New benchmark instances for the Capacitated Vehicle Routing Problem. European Journal of Operational Research.

5. Brian Kallehauge. VEHICLE ROUTING PROBLEM WITH TIME WINDOWS / Brian Kallehauge, Jesper Larsen., 2007.

6. J. Li, Y. Li, P. M. Pardalos. Multi-depot vehicle routing problem with time windows under shared depot resources. // Journal of Combinatorial Optimization. – 2016.

7. C.-Y. Liong. Vehicle routing problem: Models and solutions / C.-Y. Liong, Khairuddin Omar. // Journal of Quality Measurement and Analysis. – 2008.

8. Li, J., Li, Y., & Panos, M. P. (2016). Multi-depot vehicle routing problem with time windows under shared depot resources. Journal of Combinatorial Optimization

9. R. Dupas, I. Grebennik, I. Litvinchev, T. Romanova, O. Chorna Solution Strategy for One-to-One Pickup and Delivery Problem Using the Cyclic Transfer Approach // EAI Endorsed Transactions on Energy Web, Special issue on Energy

Conservation, Information Technologies and Large Scale Optimization, Issue 27, 2020, e5 Scopus https://eudl.eu/doi/10.4108/eai.13-7-2018.164110.

10. G. Desaulniers, J. Desproseiers, VRP with Pickup and Delivery. // The Vehicle Routing Problem. – 2002. – pp. 225-242.

11. Y. Wang, X. Ma, Y. T. Lao, H. Y. Yu. Two-stage heuristic method for vehicle routing problem with split deliveries and pickups. – 2016. – pp. 202-203.

12. Burak Eksioglu. The vehicle routing problem: A taxonomic review [Електронний ресурс] / Burak Eksioglu, Arif Volkan Vural, Arnold Reisman – 2024 – Режим доступу до ресурсу: https://staff.fmi.uvt.ro/~daniela.zaharie/ma2018/projects/biblio/applications/VehicleRouting/VRP_taxonomicReview.pdf.

13. Manar Ibrahim Hosny. Investigating Heuristic and Meta-Heuristic Algorithms for Solving Pickup and Delivery Problems : дис. докт. філос. наук / Manar Ibrahim Hosny. – Cardiff, 2010. – 242 с.

14. N. A. El-Sherbeny. Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods // Journal of King Saud University - Science. – 2010, Jul. – vol. 22. – no. 3 . – pp. 123–131.

15. Emmanouil, E., & Christos, D. (2016). The Vehicle Routing Problem with Simultaneous Pick- ups and Deliveries and Two-Dimensional Loading Constraints. European Journal of Operational Research.

16. A. Salah. Genetic algorithms – 2020. – pp. 4-13

17. M. Dorigo, S. Member, L. M. Gambardella. Ant Colony System : A Cooperative Learning Approach to the Traveling Salesman Problem. – 1997. – vol. 1. – no. 1. – pp. 53–66.

18. T. Stützle, M. Dorigo. ACO Algorithms for the Traveling Salesman Problem // Evolutionary Algorithms in Engineering and Computer Science: Recent Advances in Genetic Algorithms, Evolution Strategies, Evolutionary Programming, Genetic Programming and Industrial Applications. – 1990 . – p.

19. S. Katiyar, I. Nasiruddin, A. Q. Ansari. Ant Colony Optimization: A Tutorial Review. – 2015.

20. B. Yu, Z. Z. Yang, B. Z. Yao. A hybrid algorithm for vehicle routing problem with time windows // Expert Systems with Applications. – 2011, Jan. – vol. 38. –no. 1. – pp. 435–441.

21. IBM [Електроний ресурс]: Sterling Transportation Management System. – 2024. – Режим доступу http://www.01.ibm.com/software/commerce/ products/transportation-management/.

22. M. M. Solomon. Algorithms for the VR and SP rith TM Constraints // vol. 35 . –1987. – no. 2. –pp. 254–265.

23. G. A. P. Kinderwater, M. W. Savelsbergh. Vehicle routing: handling edge exchanges // In E.H.L. Aarts and J. K. Lenstra (eds), Local Search in Combinatorial Optimization. –1997.

24. Braekers, K., Ramaekers, K., & Nieuwenh, I. V. (2016). The Vehicle Routing Problem: State of the Art Classification and Review. Computers & Industrial Engineering.

25. Ehsan, T., & Vahid, K. (2016). Enhanced intelligent water drops and cuckoo search algorithms for solving the capacitated vehicle routing problem. Information Sciences.