Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних систем управління

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення

Виконав: студент 2 курсу, групи 2АКІТ-22м
спеціальності 151 – Автоматизація та
комп'ютерно-інтегровані технології

Юрій САВЧЕНКО
Ім'я ПРІЗВИЩЕ

Керівник: к.т.н., доцент каф. КСУ
посада, вчене, наукове
Олена НИКИТЕНКО
Ім'я ПРІЗВИЩЕ
«04» 12 2023 р.

Опонент: к.т.н., доцент каф. АІІТ
посада, вчене, наукове
Юрій ІВАНОВ
Ім'я ПРІЗВИЩЕ
«05» 12 2023 р.

Допущено до захисту
Т.в.о. зав. кафедри КСУ
Марія ЮХИМЧУК
«07» 12 2023

Вінниця ВНТУ – 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних систем управління
Рівень вищої освіти другий (магістерський)
Галузь знань – 15 – Автоматизація та приладобудування
Спеціальність – 151 – Автоматизація та комп'ютерно-інтегровані технології
Освітньо - професійна програма – Інтелектуальні комп'ютерні системи

ЗАТВЕРДЖУЮ
Т.в.о.Зав. кафедри КСУ

Марія ЮХИМЧУК

"09" жовтня 2023 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ

студенту Савченку Юрію Олександровичу .
(прізвище, ім'я, по батькові)

1. Тема роботи. Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення
керівник роботи Никитенко Олена Дмитрівна
затверджені наказом ВНТУ від "18" вересня 2023 року №247
2. Термін подання студентом роботи "1 " грудня 2023 року
3. Вихідні дані до роботи: Вихідними даними є візуалізація формування маршруту та зазначення відповідних декартових координат, які користувач може переглядати для кожного окремо обраного алгоритму та критерію, а також файл у форматі .json, що містить інформацію про маршрут.

4. Зміст текстової частини: вступ, аналіз предметної області та огляд аналогів, математична постановка задачі маршрутизації безпілотних літальних апаратів, розробка програмного та технічного забезпечення, дослідження ефективності розроблених методів.
5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень) класифікація задач управління маршрутами транспортних засобів класифікація наближених методів розв'язання задач управління маршрутами транспортних засобів,схема структурна діяльності,схема структурна діяльності, блок-схема методу найдешевшого включення, блок-схема гібридного алгоритму найдешевшого включення з проміжними покращеннями, схема структурна послідовності, діаграма класів, псевдокод для методу найдешевшого включення, приклад візуалізованих результатів виконання алгоритмів, аналіз результатів

1. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання	виконання
4	Бурєннікова Н. В., д.е.н., професор кафедри ЕПВМ.		

2. Дата видачі завдання "09" жовтня 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Огляд методів розв'язання задач маршрутизації транспортних засобів	04.10.2023		
2	Аналіз методів маршрутизації безпілотних літальних апаратів та математична постановка задач їх виконання	08.10.2023		
3	Розробка програмного та технічного забезпечення	22.10.2023		
4	Дослідження ефективності розроблених методів	06.11.2023		
5	Розрахунок економічної частини	14.11.2023		
6	Графічні матеріали	20.11.2023		

Студент Юрій САВЧЕНКО
(підпис) (*Ім'я ПРІЗВИЩЕ*)

Керівник роботи Олена НИКИТЕНКО
(підпис) (***Ім'я ПРІЗВИЩЕ***)

АНОТАЦІЯ

УДК 621.374.415

Савченко Ю. О. Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення. Магістерська кваліфікаційна робота зі спеціальності 151 – Автоматизація та комп'ютерно-інтегровані технології, освітня програма – Інтелектуальні комп'ютерні системи. Вінниця: ВНТУ, 2023. 105 с.

На укр. мові. Бібліогр.: 46 назв; рис.: 27; табл. 14.

Досліджено проблему маршрутизації транспортних засобів, зокрема безпілотників, в умовах наявності кількох складів, використовуючи алгоритми, такі як детермінований локальний пошук, заборонений пошук, алгоритм імітації відпалу та алгоритм прискореного імовірнісного моделювання. Робота ставить перед собою завдання розробити алгоритми для вирішення задачі маршрутизації дронів у ситуації, коли наявні кілька складів, включаючи детермінований локальний пошук, заборонений пошук, алгоритм імітації відпалу та алгоритм прискореного імовірнісного моделювання. Також передбачено розробку програмного забезпечення для оптимізації маршрутів і проведення дослідження ефективності розроблених алгоритмів.

У контексті переходу безпілотників з військового використання в цивільні сфери, зокрема в доставку пакетів, обговорено переваги безпілотників, такі як точність та ефективність. Результати дослідження можуть бути застосовані в різних галузях, включаючи пошуково-рятувальну діяльність та сільське господарство.

Ключові слова: маршрутизація транспортних засобів, безпілотники, детермінований локальний пошук, заборонений пошук, алгоритм імітації відпалу, програмне забезпечення, ефективність алгоритму.транспортних засобів, , детермінований локальний пошук.

ANNOTATION

UDC 621.374.415

Savchenko, Y. O. Optimal Route Control System for Aerial Vehicles Considering Landing Points. Master's Qualification Thesis in the field of 151 – Automation and Computer-Integrated Technologies, Educational Program – Intelligent Computer Systems. Vinnytsia: VNTU, 2023. 105 p.

In Ukrainian. Bibliography: 46 titles; figures: 27; tables: 14.

The research addresses the issue of routing for transportation vehicles, particularly unmanned aerial vehicles, in the presence of multiple depots, utilizing algorithms such as deterministic local search, forbidden search, simulated annealing, and accelerated probabilistic modeling algorithm. The thesis aims to develop algorithms for solving the drone routing problem in scenarios involving multiple depots, including deterministic local search, forbidden search, simulated annealing, and accelerated probabilistic modeling algorithm. Additionally, the development of software for route optimization and the investigation of the efficiency of the developed algorithms are outlined.

In the context of the transition of unmanned aerial vehicles from military applications to civilian sectors, specifically in package delivery, the advantages of drones, such as accuracy and efficiency, are discussed. The research findings can be applied in various fields, including search and rescue operations and agriculture.

Keywords: transportation vehicle routing, unmanned aerial vehicles, deterministic local search, forbidden search, simulated annealing algorithm, software, algorithm efficiency.

Зміст

ВСТУП

Проблема маршрутизації транспортних засобів (VRM) є дуже важливою та актуальною через її широке застосування на практиці. Джордж Данциг і Джон Рамсер вперше заговорили про ЗМТЗ у 1959 році [2]. Вони вручну виконують аналітичні розрахунки для оптимізації вантажних перевезень, використовуючи підхід на основі лінійного програмування. В результаті отримано близьке до оптимального рішення з 4 маршрутами і 12 споживачами. З того часу кількість досліджень у цьому напрямку значно зросла. Так, за запитом "проблема маршрутизації транспортного засобу" в Академії Google є приблизно 522 000 результатів за всі роки, 2017 - 18 900 результатів, 2018 - 20 600 результатів, 2019 - 2020 - 22 400 результатів - 2021 - 19 200 результатів - За 4 місяці ми отримали 3 850 результати. Як бачимо, ЗМЗТ зараз дуже популярний серед науковців і практиків. Це пов’язано з тим, що в деяких випадках, особливо при плануванні великих обсягів транспорту, навіть невеликі вдосконалення маршруту руху транспортних засобів (VT) можуть призвести до значної економії ресурсів.

Якщо наземний транспорт раніше вважався засобом пересування, використовувалися різні літальні апарати (ЛА), в тому числі безпілотні літальні апарати.У 1899 році Нікола Тесла сконструював і продемонстрував радіокерований міні-човен. У 1910 році американський військовий інженер Чарльз Кеттерінг запропонував використовувати літальні апарати без безпосередньої участі людини в управлінні. Їхнє призначення — використання у бойових діях: у певний момент літак ЛА має скласти крила й впасти, як бомба, на ворога. Під час Другої світової війни Лос-Анджелес пережив подальший агресивний розвиток. Вони використовуються як безпілотники-мішені, керовані бомби, радіокеровані літаки з вибухівкою, торпедні катери тощо. Згодом Лос-Анджелес почав переходити з військової

сфери на цивільну. Їх почали використовувати в мирних цілях. Лос-Анджелес почав створювати попит на промислові послуги доставки та перевезення вантажів, а також в інших сферах: оборона, пошук і порятунок, сільське господарство, екологічний моніторинг, картографування тощо. Наприклад, після нещодавнього технологічного прогресу в Лос-Анджелесі великі компанії, зацікавлені в доставці пакетів, такі як Amazon, DHL і FedEx, почали вивчати можливість включення доставки в Лос-Анджелесі в свої бізнес-послуги [3]. За принципами управління їх можна розділити на безпілотні, автоматичні та дистанційні. Після 2000 року використання новітніх літаків в авіаційній сфері стрімко зросло. Говорячи про них, вживають терміни «безпілотний літальний апарат» і «дрон».

Основними перевагами LA є: точність, значне зниження витрат на технічне обслуговування, висока ефективність, кореляція отриманих даних, широкий спектр застосування. Лос-Анджелес може вирішувати конкретні завдання, на виконання яких людям знадобляться тижні за кілька днів. Крім того, однією з переваг є безпечність використання. Наприклад, певні види робіт у шахтах є небезпечними та несуть загрозу життю працівників, і через Лос-Анджелес ці ризики можна мінімізувати.

Враховуючи те, що галузі, пов’язані з ЛА, зараз активно розвиваються, перед вченими постає нова проблема — використання ЛА для пошуку оптимального рішення для ЗМТЗ.

Метою цієї роботи є мінімізація часу огляду або витрат на маршрут, що є результатом огляду та/або обслуговування заданого набору цілей на землі за допомогою БПЛА, які можуть бути розташовані на кількох ділянках, але з певними додатковими обмеженнями.

Для досягнення мети необхідно виконати наступні завдання:

- Аналізувати поточний стан завдань маршрутизації транспортних засобів та завдань маршрутизації БПЛА;

- Огляд доступних методів вирішення проблем маршрутизації транспортних засобів за наявності кількох стоянок;

- Розробити алгоритми для вирішення задачі маршрутизації дронів за наявності кількох складів (детермінований локальний пошук, заборонений пошук, алгоритм імітації відпалу, алгоритм прискореного імовірнісного моделювання);

- розробляти програмне забезпечення;

- Провести дослідження ефективності розробленого алгоритму.

Об’єктом дослідження є процес побудови маршрутів за допомогою дронів.

Предметом дослідження є методи побудови маршруту у випадку багатостанційності.

Методи дослідження включають емпіричні методи та теоретичні методи.

Наукова новизна – класифікація задач маршрутизації транспортних засобів та розробка нових алгоритмів маршрутизації для дронів за наявності кількох складів.

Прикладне значення. Метод і програмне забезпечення можуть бути використані для вирішення багатобазових проблем маршрутизації безпілотників у пошуково-рятувальній діяльності, сільському господарстві, екологічному моніторингу, військовій справі, картографії та логістиці.

Апробація представлені в роботі результати апробовані в результаті участі в конференції LIII Науково-технічна конференція факультету інтелектуальних інформаційних технологій та автоматизації (2024)

Публікації Ю. О. Савченко наза: «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» Секція: Секція автоматизації та інтелектуальних інформаційних технологій «

LIII Науково-технічна конференція факультету інтелектуальних інформаційних технологій та автоматизації (2024)»

1 ОГЛЯД ЗАДАЧ МАРШРУТИЗАЦІЇ ТРАНСПОРТНИХ ЗАСОБІВ ТА МЕТОДІВ ЇХ РОЗВ'ЯЗУВАННЯ

1.1 Класифікація задач маршрутизації транспортних засобів

Проблема маршрутизації транспортного засобу (VRP) — це завдання пошуку найкращого маршруту транспортування товарів, об'єктів дослідження тощо від початкової точки до пункту призначення. Метою такого завдання є пошук набору маршрутів, які обслуговують клієнтів із певною кількістю ТЗ у заданому середовищі.

1.1.2 Задача маршрутизації в загальному вигляді

Класичний ЗМТЗ належить до класу комбінаторних задач [4]. Його можна виразити у вигляді графіка G(V,E),

Де V = {v0, v±,..., vn} - множина вершин (v0 - склад, v1;..., vn - точка доставки, клієнт); E - множина ребер, що відповідає шляху {(vi (vj)|i ф j});

C – матриця ваг ребер (вартості шляху) qj між вершинами, m – кількість транспортних засобів, які беруть участь у перевезенні вантажів;

Ri маршрут i-го автомобіля (i=1,m);

C(Ri) – вартість маршруту Ri;

qi кількість товару для відвантаження в i-ту точку доставки.

Кожна вершина vi пов'язана з певною кількістю товару, який необхідно доставити клієнту у відповідну точку доставки. Завдання маршрутизації полягає у визначенні набору маршрутів з мінімальною вартістю (відповідно до значення m - кількості ТК), щоб кожна вершина була відвідана лише один раз одним ТК. Усі маршрути мають починатися й закінчуватися на сайті v0.

Рішення проблеми полягає в тому, щоб розділити множину V на підмножини (шляхи) і вказати порядок обходу вершин на кожній підмножині (розташування вершин шляху). Рішення задачі є прийнятним, якщо всі шляхи

задовольняють додатковим обмеженням задачі. Цільовою функцією (TF) є загальна вартість перевезення:

$$F_{VRP} = \sum_{i=1}^{m} C(R_i), \quad (1.1)$$

де C(Ri) - сума вартості шляху (ваги країв) маршруту Ri.

У класичному варіанті ЗМТЗ необхідно знайти прийнятне рішення з мінімальними витратами. Часто в реальних ЗМТЗ виникає багато додаткових обмежень, що призводить до подібного набору проблем.

1.1.3 Найбільш поширені класи задач маршрутизації транспортних засобів

Однією з найпростіших ЗМТЗ є задача комівояжера (ТСП). Він полягає в знаходженні найкоротшого G-гамільтонового циклу на графіку. Тобто агент повинен один раз пройти всі вершини графа, а потім повернутися до початкової вершини з найменшою пройденою відстанню або найменшим витраченим часом.

Існує багато типів ЗМТЗ: обмеження пропускної здатності, «часові вікна», альтернативні динамічні склади, повернення та доставка товарів, різні транспорти, маршрути за розкладом тощо. У літературі існує багато різних класифікацій занять ЗМТЗ. Рисунок 1.1 показує одну з таких можливих ієрархій.

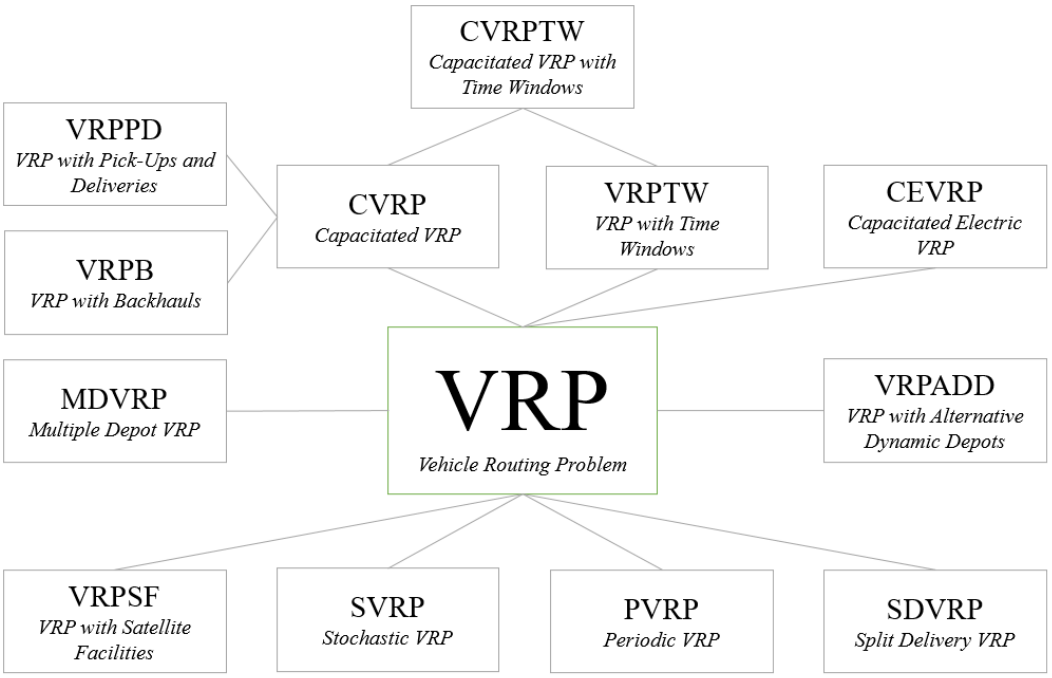


Рисунок 1.1 - Ієрархія основних класів ЗМТЗ

Розглянемо докладніше кожен із типів задач, зображених на рисунку 1.1.

Здатний VRP, CVRP [5]. Іншим обмеженням проблеми є те, що маса вантажу на кожному маршруті не повинна перевищувати заданого значення обсягу вантажу, який необхідно доставити в певну точку. Метою такого завдання є мінімізація загальної вартості перевезення. Задачі маршрутизації з обмеженнями навантаження можна розширити додатковими обмеженнями і отримати принципово нові класи задач.

VRP Pickup and Delivery (VRPPD) [6, 7]. Ця проблема відрізняється від класичної проблеми тим, що клієнт у пункті доставки може повернути товар, який транспортний засіб повинен привезти назад у двір. Умовою повернення товару є те, що транспортний засіб не має бути перевантаженим, тобто має бути врахована вантажопідйомність. Такі ЗМТЗ складно вирішувати, оскільки вони породжують багато проблем: можливість неефективного використання транспорту, збільшення вартості маршруту та загальної кількості транспортних засобів у депо тощо. Для простоти прийнято

розглядати задачі з додатковими обмеженнями. Наприклад, товар відправляється тільки в пункт видачі і тільки повертається на склад, тобто обмін товаром між пунктами видачі не відбувається. Ще один спосіб спростити – зняти обмеження про те, що всі точки доставки можна відвідувати лише один раз. Є й інший спосіб: припустимо, що вантажівки спочатку доставляють усі вантажі, а потім починають забирати їх із пункту доставки (VRP з Backhauls, VRPB, описано нижче). Ціль місії: мінімізація транспортних витрат і розміру автопарку. Обмеження: кількість товарів, які доставляються та забираються від клієнтів, не може перевищувати вантажопідйомність транспортного засобу в будь-якій точці маршруту. Однією зі складних проблем VRRPD є гетерогенна VRP зі змішаними транзитними та часовими вікнами (HVRPMBTW) [8]. Завдання характеризується обмеженим обсягом традиційних знань, різними можливостями та витратами.

VRP with backhaul (VRPB) [9, 10]. У попередньому завданні VRRPD вам потрібно було врахувати, що всі товари, які повертатиме клієнт, мають відповідати ТЗ. Але місія VRPB інша: усі товари мають бути доставлені, перш ніж клієнти зможуть їх повернути. Це пояснюється тим, що більшість ТК зазвичай завантажуються ззаду або зверху, утворюючи стопку. Тому переставляти товар під час кожної доставки економічно невигідно.

VRP з часовими вікнами (VRPTW) [11]. Основним обмеженням цього завдання є те, що ТЗ має доставити товар протягом певного часу. Такі завдання мають верхній і нижній ліміт часу. Тому, якщо товар буде доставлено в це місце після обмеженого часу, вони не зможуть прийняти товар, інакше його оштрафують. Якщо ТЗ доставить товар раніше нижньої межі, то він повинен почекати. Метою цього завдання є мінімізація загальної вартості транспортування, розміру автопарку та загального часу очікування. Такі завдання допомагають визначити найкращий час для від’їзду транспортного засобу зі стоянки та уникнути простою. Подальшим

розвитком проблеми VRPTW є розв’язання VRP за допомогою Backhauls and Time Windows (VRPBTW). Проблема включає пропускну здатність, окупність продукту, часові вікна та вирішується за допомогою гібридного метаевристичного алгоритму. Метою є мінімізація загальної відстані транспортування [12]. Інша проблема полягає в тому, що VRP забезпечує м’які часові вікна для багатьох типів традиційних знань і контролює забруднення навколишнього середовища від транспортування. Мета полягає в тому, щоб дослідити взаємозв'язок між витратами на розподіл, задоволеністю клієнтів і забрудненням навколишнього середовища. Для вирішення цієї проблеми був розроблений гібридний генетичний алгоритм, результати якого показали, що швидкість транспортного засобу має сильну кореляцію з експлуатаційними витратами та забрудненням навколишнього середовища, тоді як вантажопідйомність впливає на експлуатаційні витрати, задоволеність споживачів та забруднення навколишнього середовища [13].

Можливий VRP із часовими вікнами (CVRPTW). Якщо ми поєднаємо ЗМТЗ з обмеженнями потужності та «часовими вікнами», ми матимемо такий тип проблеми. Вони будуть враховувати вантажопідйомність транспортного засобу та часові обмеження, протягом яких товар повинен бути доставлений.

ЗМТЗ має кілька складів (багатоскладські ВРП, МДВРП) [14, 15, 16, 17, 18, 19, 20, 21, 22]. Це завдання вимагає розподілу клієнтів по різних складах найбільш вигідним способом і визначення найкращих маршрутів доступу до них. Кожен склад має власний парк транспортних засобів, кожен з яких обслуговує клієнтів, а потім повертається. Метою таких місій є мінімізація транспортних витрат і розміру флоту. У деяких випадках може бути кілька складів, які обслуговують клієнтів. Якщо клієнти зосереджені біля кожного складу, то завдання можна розділити на кілька незалежних завдань. Але якщо клієнти і склади знаходяться в будь-якому місці, то потрібно шукати рішення ЗМТЗ з декількома складами.

ЗМТЗ має можливість перезавантаження (ВРП із супутниковими засобами, ВРПСФ) [23]. Особливістю класичного ЗМТЗ є те, що транспортні засоби при проходженні маршруту повинні починати і закінчувати маршрут в депо. Це пов'язано з обмеженою вантажопідйомністю: вантажівка повинна бути перевантажена після того, як буде доставлений весь вантаж. Для деяких завдань вигідно перевантажувати машину безпосередньо на маршруті без повернення в депо. Цього можна досягти за допомогою додаткових традиційних знань. Типовою ситуацією є велика кількість клієнтів, які бажають регулярних поставок від центрального постачальника. Метою задачі є мінімізація витрат на транспортування протягом заданого часу. Можливим недоліком цієї проблеми є те, що загальна вартість вирішення проблеми є вищою в короткостроковій перспективі, ніж у випадку зі звичайним VRP через додаткові витрати на додаткові ТК.

ЗМТЗ (Random VRP, SVRP) з випадковими даними [24, 25]. Для цього типу ЗМТЗ характерна випадкова поведінка окремих компонентів задачі. Випадковість виражається як: випадкові клієнти - кожна точка розподілу є значенням ймовірності; Випадкові запити – кількість товарів, які потрібно доставити клієнту, і відстань між пунктами доставки є випадковими значеннями. Вирішення таких завдань відбувається в два етапи. На першому етапі рішення отримують без урахування інших змінних. На другому етапі попередньо отриманий розв'язок коригується за випадковими значеннями. Мета задачі – мінімізація загальної вартості транспортування. Якщо деякі точки доставки невідомі, повинні бути накладені додаткові обмеження: певні умови виконуються із заданою ймовірністю, або повинна бути побудована модель корекції, яка буде виконуватися, коли певні обмеження порушуються. Наприклад, у задачі СВРП про повернення товарів і обмеження вантажопідйомності транспортного засобу методом корекції буде: повернення в депо при перевантаженні транспортного засобу, повернення на маршрут після розвантаження, повернення в депо при перевантаженні

транспортного засобу і потім знову оптимізуйте маршрут, що залишився; навіть якщо транспортний засіб не повністю завантажений, заплануйте повернення в гараж якомога раніше. Для такої модифікованої моделі вирішення проблеми може залежати від кількості зібраного товару та відстані до складу. Якщо вантажопідйомність наступного клієнта перевищує вантажопідйомність транспортного засобу, транспортний засіб може повернутися в депо.

ЗМТЗ (Періодичний ВРП, ПВРП) з періодичною маршрутизацією [26]. Такі завдання характеризуються розширеними термінами планування від одного до кількох днів. Метою цієї місії є мінімізація транспортних витрат і розміру автопарку. Додаткові обмеження: транспортний засіб може бути повернено в гараж в іншу дату, ніж дата його вибуття; кожну точку доставки необхідно відвідати принаймні один раз протягом M-днів. Кожна заявка клієнта повинна бути виконана одним ТЗ за одне відвідування. Якщо плановий період M = 1, то задача спрощується до класичної ВРП. Кожна точка розподілу в періодичній задачі маршрутизації має бути відвідана k разів, і $1 < k < M$. У класичному PVRP щоденні замовлення клієнта є фіксованими. Тому таку задачу можна розглядати як задачу щоденного складання набору маршрутів, якщо маршрути задовольняють обмеженням і загальна вартість перевезення мінімізована.

ЗМТЗ має різні способи доставки (Split Delivery VRP, SDVRP) [27, 28]. Принципова відмінність таких завдань полягає в тому, що пункт доставки може обслуговуватися декількома ТЗ, якщо це зменшить загальну вартість транспортування. Типовим прикладом проблеми цього типу є великий обсяг замовлення при малій вантажопідйомності автомобіля. Метою цієї місії є мінімізація транспортних витрат і розміру автопарку.

VRP з альтернативним динамічним складом (VRPADD) [29, 30]. Сферою застосування для таких завдань є ЗМТЗ, де в якості ТЗ виступає дрон або інша мобільна роботизована система. Особливість цього виду місії

полягає в тому, що транспортний засіб може почати і завершити свій маршрут в мобільному гаражі. Такі завдання пов'язані з військовою і морською діяльністю. Вантажопідйомність не відіграє ключової ролі при оптимізації маршрутів (наприклад, при аерофотозйомці). Важливі обмеження на системні ресурси: кількість палива або ємність акумулятора;

ЗМТЗ використовує електротранспорт і має обмежену вантажопідйомність (Capacitated Electric VPR, CEVRP). З розвитком електромобілів кількість зарядних станцій і запас ходу електромобілів обмежені, а потім виникають проблеми з дорогою. У такому завданні необхідно враховувати не лише послідовність обслуговування клієнтів, але й графік зарядки електромобіля. Вирішити обидві проблеми одночасно досить складно. Тому її розглядають як задачу дворівневої оптимізації. На першому рівні розглядається задача ємнісної маршрутизації (CVRP), а на другому – задача зарядки транспортного засобу за відомим маршрутом. Для підзавдань першого рівня обмеження енергоспоживання ігноруються, а на другому рівні розробляються нові евристики для задоволення обмежень енергоспоживання згенерованих маршрутів.

На рисунку 1.2 показано розташування досліджуваної задачі маршрутизації в межах множини класів ЗМТЗ.

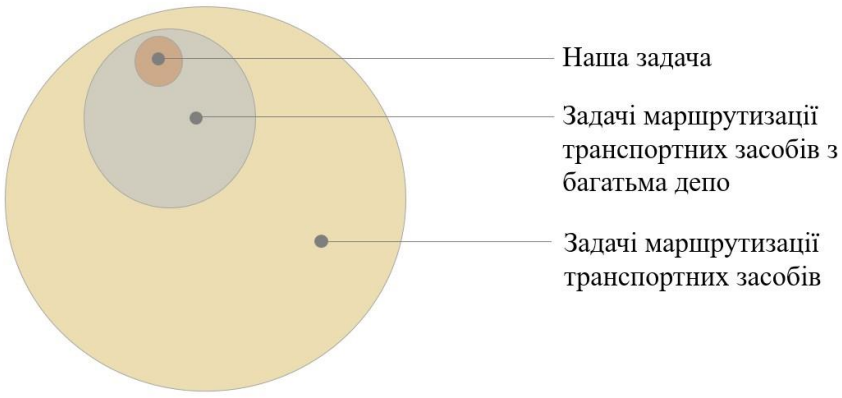


Рисунок 1.2 – Місце досліджуваної проблеми серед множини класів ЗМТЗ

Проаналізувавши подібні твердження, можна зробити висновок, що досліджувана проблема є проблемою MDVRP, яка характеризується наявністю кількох бібліотек із розподіленими між ними ТЗ.

1.2 Класифікація задач маршрутизації транспортних засобів для безпілотних літальних апаратів

Проблема маршрутизації LA перетворюється з нової теми на область досліджень, що розвивається. Розвиток сучасних технологій поступово призвів до того, що місцева логістика відіграє ключову роль у міській логістиці. Зараз проблема маршрутизації LA стала зрілою технологією і застосовується в провідних галузях промисловості. LA можуть значно скоротити фінансові витрати та час, необхідний для транспортування матеріалів, оскільки вони дешевші в обслуговуванні, ніж традиційні транспортні засоби, а LA можуть, наприклад, зменшити витрати на робочу силу, виконуючи завдання автономно. На рисунку 1.3 наведено класифікацію областей застосування маршрутних завдань ПС. Загалом їх можна розділити на дві категорії: цивільні та військові.

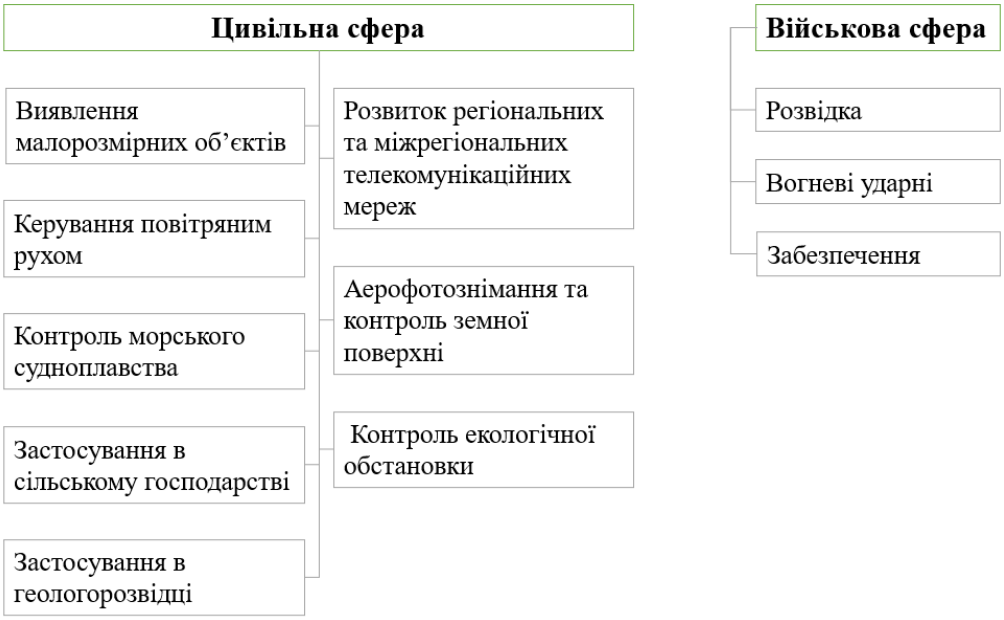


Рисунок 1.3 – Сфери застосування ЛА

Розглянемо сфери застосування ЛА більш детально, як показано на рисунку 1.3. Цивільна сфера включає наступні завдання:

- Виявлення малих об'єктів (повітря, вода, земля, пошук і порятунок, допомога в надзвичайних ситуаціях);

- управління повітряним рухом (тимчасові маршрути у важкодоступних районах, під час стихійних лих і аварій, при авіабудівництві);

- контроль морського судноплавства (пошук і виявлення суден, запобігання надзвичайним ситуаціям у портах, контроль морських кордонів, контроль правил рибальства);

- Застосування в сільському господарстві (властивості в аерокосмічній та хімічній інженерії);

- Застосування в геологорозвідці (визначення властивостей ґрунтів, розвідка корисних копалин, виявлення надр);

- Розвиток регіональних і міжрегіональних телекомунікаційних мереж (системи зв'язку, телебачення і радіомовлення, ретрансляції, системи навігації, реклами, телебачення, кіно);

- Аерофотозйомка та контроль земної поверхні (картографування, контроль водних та погодних умов, правоохоронна діяльність, виявлення лісових пожеж, спостереження за периметрами об'єктів, контроль залізничних колій);

- Контроль стану навколишнього середовища (радіаційний контроль, газохімічний контроль, контроль стану газонафтопроводів, контроль лавинного стану) [31].

Військова сфера включає наступні завдання:

- Розвідка (повітряна розвідка з малих і середніх висот в умовах сильного протистояння з природними умовами і складною радіоелектронною обстановкою, а також можливість передачі розвідувальної інформації, що надходить із захищених радіоканалів у режимі реального часу; виявлення та ідентифікація наземних, наземних і повітряних об’єктів). цілей, угруповань військ і техніки, вогневих точок і укріплень, виявлення активно запускаючих об'єктів);

- вогневе ураження (ураження техніки і живої сили противника шляхом ураження наземних і морських цілей; ураження повітряних цілей і знищення боєголовок балістичних ракет; дистанційна постановка мінних полів і розмінування);

- Постачання (перевезення вантажів у будь-яких бойових умовах, у тому числі під час бойових дій і на території, забруднені радіоактивними, хімічними та біологічними речовинами; боротьба з наземними, повітряними та морськими пожежами; оцінка результатів ударів по противнику; ретрансляція інформації та даних, пошук та евакуація втрат з поля бою) [31].

Ієрархію основних класів ZMTZ можна розширити шляхом додавання завдань маршрутизації за допомогою LA.

Завдання комівояжерів з використанням безпілотних літальних апаратів (TSP with Unmanned Aerial Vehicles, TSP-UAV) [32, 33]. Такі завдання характеризуються наявністю складу, ЛА та вартісними

показниками. LA повинен відвідати кожну ціль лише один раз і повернутися на склад. Мета задачі – мінімізація загальної вартості маршруту. Наприклад, критерієм виконання може бути пройдена відстань або час для виконання завдання.

Місія комівояжера з використанням кількох літальних апаратів (TSP with Multi Unmanned Aerial Vehicles, TSP-MUAV) [34, 35]. Ця проблема є узагальненням проблеми TSP-UAV. Принципова відмінність полягає в тому, що ціль доступна не одному LA, а кільком LA. Крім того, кожну ціль може відвідати лише один LA. Метою цієї задачі є визначення набору маршрутів, які мінімізують їх вартість.

ТСП з безпілотними літальними апаратами та вартість маршруту (ТСП-УАВКР). Ця проблема також є узагальненням проблеми TSP-UAV. Основна його відмінність полягає в тому, що для кожного маршруту в залежності від напрямку можуть бути встановлені різні ціни. Метою цієї проблеми також є мінімізація загальної вартості маршруту.

Класичний ЗМТЗ з використанням БПЛА (ВРП-БПЛА) [36, 37, 38, 39, 40, 41, 42]. Необхідно визначити набір маршрутів для флоту безпілотників, які будуть відвідувати ціль. Для класичного ЗМТЗ метою завдання є мінімізація загальної вартості маршрутизації.

VRP з безпілотниками та вікнами часу (VRPTW-UAV) [43, 44, 45, 46]. Ця проблема є узагальненням проблеми VRP-UAV. Фундаментальна відмінність цього завдання полягає в тому, що досягнення мети відбувається протягом заданого періоду часу, який називається «часовим вікном». Такі «часові вікна» встановлюються індивідуально для кожного об'єкта. Якщо LA досягає мети раніше, він повинен почекати, якщо пізніше, він повинен заплатити штраф. Метою цієї проблеми також є мінімізація загальної вартості маршруту.

ЗМТЗ має можливість використання авіації та перезавантаження (ВРП з БПЛА та супутниковими засобами, ВРПСФ-БПЛА) [47]. Ця місія

характеризується можливістю скоротити використання літака за рахунок перезавантаження. Метою також є мінімізація маршрутних витрат.

VRP з можливістю БПЛА, CVRP-UAV [48]. Однією з особливостей цієї місії є обмежена вантажопідйомність літака. При цьому кожен безпілотник може літати лише на стільки, скільки має достатньо потужності, і не підлягає дозаправці. Метою цієї проблеми також є мінімізація вартості маршруту.

ВРП з безпілотником і можливістю дозаправки, КВРППР-БПЛА. Ця місія є продовженням місії CVRP-UAV, але з можливістю повернення на базу для дозаправки. Таким чином, у Лос-Анджелесі можливі кілька рейсів.

ВРП з БПЛА та мультискладом, МДВРП-БПЛА. Ця проблема дуже схожа на звичайну MDVRP, за винятком того, що тут LA використовується як ТК. У цій місії є кілька складів, кілька цілей і кілька літаків. Кожен LA може відвідати лише один пункт призначення, а потім повинен повернутися до того самого місця, з якого він почав. Мета цієї задачі - знайти шлях з найменшими витратами для досягнення мети.

Як видно з наведеної вище класифікації, клас ZMTZ з використанням LA подібний до класу ZMTZ без використання LA, але все ж є певні відмінності.

1.3 Класифікація методів розв’язування задач комбінаторної оптимізації

Для простоти ми вважаємо, що задача комбінаторної оптимізації (COM) не має додаткових обмежень, тобто D ≠ X. Потрібно знайти x* = X таке, що

$$x_* = \arg \operatorname*{ext}_{x \in X} f(x). \tag{1.2}$$

Ми розглядаємо комбінаторний алгоритм оптимізації (COM) як специфічний процес для перетворення заданої підмножини X ⊂ X (початкового наближення) у множину X* ⊂ X:

$$AZ = X_*, \tag{1.3}$$

де X* – множина знайдених розв’язків задачі (1.2).

Існує багато методів розв’язування ЗМТЗ, які класифікуються за різними ознаками: за точністю, типом використання простору, структурою розрахункової схеми, отриманим рішенням тощо. Розглянемо дві можливі класифікації основних методів: за точністю та за типом розрахункової схеми.

1.3.1 Обчислювальна складність задач комбінаторної оптимізації

Проблеми поділяються на дві категорії: проблеми P і проблеми NP.

Задачі P-типу — це задачі, які можна розв’язати за поліноміальний час, а задачі типу No — це задачі, час розв’язання яких не обмежений поліномами, і точне розв’язання можна знайти лише шляхом обходу всіх варіантів.

Для задач типу NP точні алгоритми не рекомендуються, оскільки їх обчислювальна складність занадто висока. Для них використовуються наближені методи, такі як: евристики, метаевристики та ін.

3.2 Класи алгоритмів комбінаторної оптимізації за точністю

На рис.1.4 наведено основні методи розв’язування ЗМТЗ, класифіковані за точністю.

Рисунок 1.4 – Класифікація АКО за точністю

Exact IF знаходить глобальні рішення, для яких X* ⊂ Arg ext. Наближений АКО поділяється на алгоритми з апріорними оцінками точності та апостеріорними оцінками точності [4]. Для пошуку розв'язків проблеми комівояжера використовуються евристичні методи, засновані на розумних міркуваннях (наприклад, методи, які включають найближчі міста). Однак такі методи не забезпечують необхідної точності отриманих результатів.

На практиці дослідники часто об'єднують алгоритми оцінки наближеної точності та евристичні алгоритми в один клас, який просто називають «алгоритмами наближеного визначення». Тоді IF можна розділити на дві великі категорії: точні алгоритми та наближені алгоритми.

Точні алгоритми можна розділити на загальні методи та спеціальні алгоритми. Загальні методи включають метод повного перерахування, метод розгалуження та зв'язування, метод розгалуження та зв'язку, метод аналізу послідовності варіантів, метод динамічного програмування тощо. До спеціальних належать алгоритми, розроблені для конкретної задачі і тому мають більш вузьку сферу застосування.

Алгоритми апроксимації можна розділити на сім категорій (рис. 1.5).

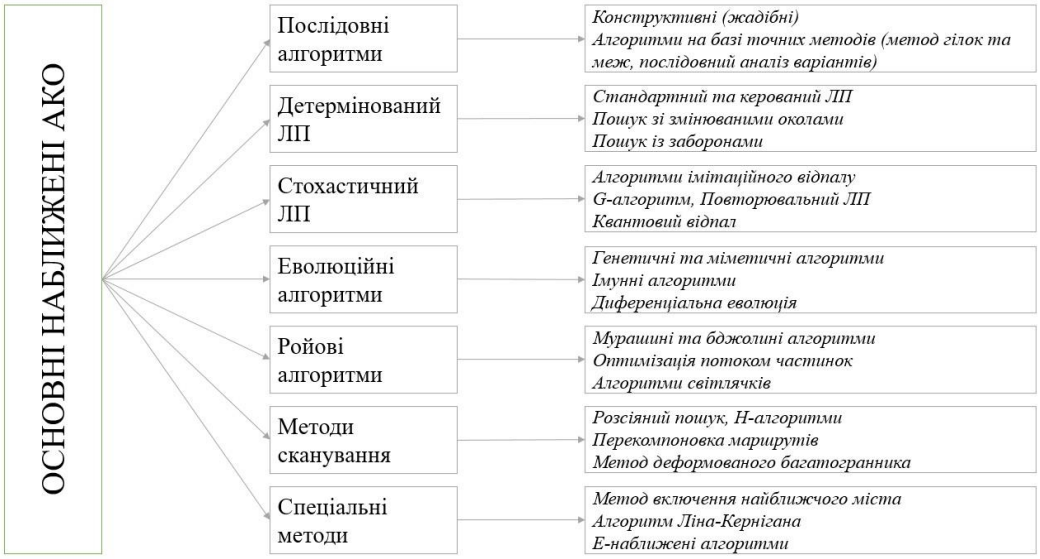


Рисунок 1.5 – Класифікація основних наближених АКО

Важливо відзначити, що в рамках даної класифікації точні алгоритми (такі як розгалуження та межі) використовуються для створення наближених розрахункових рішень. Звичайно, сьогодні АКО набагато більше, ніж показано на рисунку 1.5, але, як показує аналіз наукових публікацій у цьому напрямку, саме ці алгоритми використовуються найбільше при розв'язанні ЗКО.

Переважно лише наближені алгоритми використовуються для розв'язання складних задач, і це пов'язано з кількома причинами: Майже всі складні задачі є ІЧ-складними задачами, тому їх точні розв'язки є проблематичними, навіть якщо використовуються наближені алгоритми. Сучасні комп'ютери; зазвичай CF ZKO є багатоекстремальним, тобто існує кілька локальних мінімумів або максимумів; у багатьох прикладних задачах є певна помилка в наданих даних; ідеї, які є основою для розробки наближених АКО, дозволяють створення не тільки Алгоритмів, які можуть розв'язати задачу, а й розв'язати цілий клас наближень на основі формули ZKO.

1.3.3 Класи алгоритмів комбінаторної оптимізації за типом обчислювальної схеми

Залежно від типу розрахункової схеми АКО поділяють на конструктивні (прямі, послідовні) та ітераційні. Нехай у нас є певний набір осі Y X.

Конструктивні алгоритми — це алгоритми, де $Y \supset X$ ($Y \neq X$), тобто вони працюють у розширенні простору рішень X. Починаючи «з нуля» або інших фрагментів рішення, які поступово утворюють повне рішення. Розглянемо приклади таких алгоритмів для різних ЗКО.

Задача комівояжера розв'язується алгоритмом із залученням найближчого міста, у якому на кожному наступному кроці до маршруту додається вершина, відстань до якої серед усіх можливих відстаней є

найменшою. Це називається «перейти до найближчого», посилаючись на жадібний алгоритм.

Для квадратичної задачі призначення характерно розміщення об’єктів (елементів) із сильнішим потоком першими.

Пошук мінімального остовного дерева здійснюється за допомогою евристики, згідно з якою на кожному наступному кроці побудований сегмент остовного дерева включає вершини, загальна довжина яких мінімально збільшується і не утворюють підкільця.

Ітераційні алгоритми — це алгоритми, які обчислюють «повне» рішення на кожному кроці, тобто простір пошуку Y ≠ X. Починаючи з деякого x0 ∈ X, ітераційний алгоритм намагається покращити його крок за кроком:

$$x^{(h+1)} = A^{(h)}x^{(h)}, h = 0,1, \ldots,$$

де A(h) — ітераційний процес, який у більшості випадків не залежить від кроку h, тобто A(h) = A.

Існує два типи ітераційних алгоритмів: траєкторні алгоритми та загальні алгоритми. Ітеративний підхід, який оперує одним (поточним) рішенням на кожному кроці, називається траєкторією. Ітераційні методи на основі популяції — це методи, у яких на кожній ітерації одночасно обчислюється декілька рішень замість одного. Отже, для алгоритму траєкторії ||Z|| = ||X*|| = 1, для загального ||Z|| > 1, ||X*|| > 1.

Тому в першому розділі розглянуто ЗМТЗ і встановлено, що вони бувають різних типів: з обмеженнями вантажопідйомності, «часовими вікнами», альтернативними динамічними складами, поверненням і доставкою товару, різними видами транспорту, періодичної маршрутизації, з повторним перевезенням. Можливість завантаження тощо. Це дає класифікацію найбільш поширених ЗМТЗ.

Після аналізу подібних пропозицій визначається приналежність досліджуваної задачі до класу ЗМТЗ з декількома складами та показано

положення досліджуваної задачі в множині класу ЗМТЗ. Під час аналізу поточного стану ЗМТЗ було виявлено, що в багатьох випадках рекомендується використовувати ЛА замість штатних транспортних засобів. Розвиток сучасних технологій поступово призвів до того, що LA відіграє ключову роль у різноманітних логістичних рішеннях. Зараз проблема маршрутизації LA стала зрілою технологією і застосовується в провідних галузях промисловості. Аналіз сфер застосування ЛА показує, що в цілому її можна розділити на дві категорії: цивільну та військову. LA можуть значно скоротити фінансові витрати та час, необхідний для доставки матеріалів, оскільки вони дешевші в обслуговуванні, ніж традиційні ТК.

Як і звичайні ЗМТЗ, зараз існує багато типів ЗМЗТ з використанням ЛА: завдання продавця з використанням кількох ЛА, завдання продавця з використанням ЛА та вартості маршруту тощо. При цьому також розглядається метод розв’язання CSI, аналізується обчислювальна складність CSI та виділяються основні категорії CSI за точністю та типом схеми розрахунку. Тому за точністю АКО поділяють на точні, наближені (з оцінкою точності) і евристичні, а за типом розрахункової схеми — на конструктивні та ітераційні. В ітераційних алгоритмах також виділяють дві категорії: траєкторії та популяції.

2 МАТЕМАТИЧНА ПОСТАНОВКА ЗАДАЧІ МАРШРУТИЗАЦІЇ БЕЗПІЛОТНИХ ЛІТАЛЬНИХ АПАРАТІВ

2.1 Змістовна постановка задачі

Загалом розглядається проблема планування місії для гетерогенної групи літальних апаратів. Кількість досліджуваних об’єктів, час огляду та координати площини вважаються заданими; кількість аеропортів та їх координати, кількість літаків, швидкість кожного літака, вартість одиниці відстані польоту та ресурси польоту. Мета полягає в тому, щоб мінімізувати загальну вартість або час виконання завдання.

Проблема полягає в тому, щоб досліджувати та/або підтримувати заданий набір об’єктів у цьому районі. При цьому ЛА не закріплюється за конкретним складом – місцем розташування бази. Мета полягає в тому, щоб мінімізувати час або вартість маршруту з використанням мінімальної кількості повітряних суден або будь-якої кількості доступних літаків (з урахуванням певних додаткових обмежень):

- До кожного об'єкта може бути доступний тільки один LA;

- кожен літальний апарат має обмежені льотні ресурси;

- Довжина маршруту кожного ПС не повинна перевищувати його льотний ресурс;

- Після завершення місії кожен LA повинен повернутися на станцію свого початкового маршруту.

Якщо ресурсів LA недостатньо для перевірки та/або обслуговування всіх об’єктів, проблема нерозв’язана. У цьому випадку необхідно збільшити кількість задіяних літаків або використовувати літаки з підвищеними параметрами ресурсу.

2.2 Критерії оцінювання розв’язку

Даній місії призначається дві категорії: ті, які покращують заданий критерій за кількістю залучених літаків, і ті, які мінімізують кількість літаків шляхом подальшого вдосконалення критерію. При цьому критерії такі: час виконання завдання та вартість маршруту. Отже, у нас чотири різні завдання.

Користувачі мають можливість вибрати одну з двох категорій завдань. Якщо вибрано клас з найменшим часом або вартістю виконання, то завдання алгоритму полягає в тому, щоб покращити критерій незалежно від кількості ЛА. Якщо категорії вибрано для мінімізації кількості LA та подальшого уточнення критерію, першим завданням алгоритму є знайти мінімальну кількість LA перед тим, як критерій потрібно буде уточнити. Перший тип трапляється, коли в Лос-Анджелесі є парк автомобілів, тоді потрібно лише знайти найкращий маршрут з точки зору часу чи вартості. Друга категорія місій корисна для клієнтів, яким потрібно придбати літаки/збільшити свій парк перед місією, тому їм важливо знати мінімальну кількість літаків, необхідних для побудови оптимального маршруту.

Розглянемо критерій мінімізації часу виконання завдання. Для кожного повітряного судна повинні бути вказані його швидкість і ресурс польоту, а для кожного об'єкта - час огляду. Знайти час виконання завдання як час виконання максимального маршруту. Час виконання маршруту розраховується як сума добутку загальної довжини маршруту і швидкості літака на загальний час вимірювання об'єкта.

Розглянемо критерій мінімізації маршрутних витрат. Для кожного літака встановіть ресурс польоту та вартість одиниці пройденої відстані. Вартість маршруту є добутком його довжини та одиничної вартості ЛА, що виконує маршрут.

2.3 Математична модель

Для заданої задачі оптимізації можна запропонувати цілі (булеві) і комбінаторні формули. Давайте спочатку розглянемо модель з булевими змінними. Для цього введемо позначення:

M = [n + 1,..., n + m} - складська множина, m - кількість складів;

N = [1,..., n} - множина цілей, які необхідно відвідати, n - їх кількість;

P = M ∪ N - множина точок (склад і ціль);

B = [1, ...,b} - множина LA;

б——кількість ЛА;

D = dij, i,j = 1,n + m - матриця відстаней між ділянками та об'єктами;

T = ti, a ε N - час перевірки i-ї цілі;

C = ck,k ∈ B – питома вартість траєкторії польоту k-го літака;

V = vk,k ∈ B - середня швидкість k-го літака;

R = rk,k ∈ B – ресурс польоту k-го літака.

Переліт k-го літака від пункту (ангару, об’єкта) до пункту (ангару, об’єкта) задається змінною hijk, розрахованою за формулою (2.1):

$$x_{ijk} = \begin{cases} 0 - \text{переліт не виконується} \\ 1 - \text{переліт виконується} \end{cases}, \; i,j \in P, k \in B. \tag{2.1}$$

Відповідно до стандарту завдання полягає в мінімізації однієї з цільових функцій: загальної вартості завдання (розраховується за формулою (2.2)) або часу виконання завдання (розраховується за формулою (2.3)):

$$F_1 = \sum_{k\in B} \sum_{i\in P} \sum_{j\in P} c_k d_{ij} x_{ijk}; \tag{2.2}$$

$$F_2 = \max_{k\in B} \left(\sum_{i\in P} \sum_{j\in P} \frac{d_{ij} x_{ijk}}{v_k} + \sum_{i\in M} \sum_{j\in P} t_j x_{ijk} \right). \tag{2.3}$$

Обмеження такі:

$$\sum_{k\in B}\sum_{i\in P} x_{ijk} = 1, j \in N; \quad (2.4)$$

$$\sum_{k\in B}\sum_{j\in P} x_{ijk} = 1, i \in N; \quad (2.5)$$

$$\sum_{i\in M}\sum_{j\in N} x_{ijk} = 1, k \in B; \quad (2.6)$$

$$\sum_{i\in N}\sum_{j\in M} x_{ijk} = 1, k \in B; \quad (2.7)$$

$$\sum_{k\in B}\sum_{i\in M}\sum_{j\in M} x_{ijk} = 0; \quad (2.8)$$

$$\sum_{k\in B}\sum_{i\in P} x_{iik} = 0; \quad (2.9)$$

$$\sum_{i\in P}\sum_{j\in P} d_{ij}x_{ijk} \leq r_k, k \in B. \quad (2.10)$$

Цільова функція визначає загальну вартість планування маршруту польоту над об’єктом (як показано у рівнянні (2.2)) або час для виконання завдання огляду об’єкта (як показано у рівнянні (2.3)), на основі різних критеріїв.

Область визначення змінної задачі задається формулою (2.1). Кожна ЛА відвідує певну кількість об'єктів, але дохід і вихід з кожного об'єкта виконується лише один раз однією ЛА, що визначає формули (2.4) і (2.5). Рівняння (2.4) вказує, що лише один LA залишає об’єкт, а рівняння (2.5) вказує, що лише один LA досягає об’єкта. Рівняння (2.6) задає умови для вильоту літака з аеропорту, а рівняння (2.7) задає умови для повернення літака в аеропорт. Рівняння (2.8) відображає вимогу не літати безпосередньо з одного аеропорту в той чи інший аеропорт. Рівняння (2.10) показує обмеження льотних ресурсів літака.

Розглянемо комбінаторну модель задачі. Компактніше, програмна реалізація зручніша. Оскільки:

n - кількість об'єктів дослідження;

m - кількість складів;

б - кількість ЛА;

Ti - тривалість перевірки i-го об’єкта, i = 1,n;

Rk - Ресурс польоту k-го літака, k = 1,b;

Sk - швидкість k-го літака, k=1,b;

Ck – вартість одиниці дальності польоту k-го літака, k = 1,b.

Нехай V = {v1,vn} – множина об’єктів дослідження, E = [e1,em} – множина складів. Розглянемо розбиття множини V на спеціальні розбиття Vi, i = 1,b, які є впорядкованими послідовностями елементів V і задовольняють умови (2.11) і (2.12):

$$\bigcup_{i=1}^{b} V^i = V; \tag{2.11}$$

$$\bigcap_{i=1}^{b} V^i = \emptyset. \tag{2.12}$$

Ці спеціальні розділи визначають елементи кожного маршруту LA та порядок їх обходу. Маршрут і-го ЛА задається вектором Xi = (хи1,...хил), и = 1,b, 1 = 1, |Xl|, де хи1 = хил = еи, еи ∈ E - вихідна ділянка № .i LAs;(xi,2, ..,xi,l-1) = Vi, a = 1,b - одне із зазначених упорядкованих розділів. Ми використовуємо X = {X1, ...,Xb} для представлення набору всіх маршрутів. Серед них довжина шляху і-го ЛА розраховується за формулою (2.13):

$$L(X^i) = \begin{cases} \sum_{j=1}^{|X^i|} d(x_j^i, x_{j+1}^i), \text{якщо } V^i \neq \emptyset, i = \overline{1,b} \\ 0, \text{в іншому випадку,} \end{cases} \tag{2.13}$$

де d(xi,j, xi,j+1) – відстань між відповідними точками (складами, приміщеннями).

Відповідно до стандарту, завдання полягає в мінімізації однієї з цільових функцій. Вартість маршруту розраховується як добуток вартості одиниці шляху, пройденого повітряним судном, на довжину маршруту Цільова функція обчислюється за формулою (2.14):

$$F_1(X) = \sum_{i=1}^{m} C^i L(X^i). \tag{2.14}$$

Тривалість маршруту розраховується як сума відношення довжини маршруту до швидкості літака і сумарного часу обслуговування всіх об'єктів на маршруті. Цільова функція критерію часу виконання завдання розраховується за такою формулою

$$F_2(X) = \max_{i=1,b} \left(\frac{1}{S^i} L(X^i) + \sum_{j \in V^i} T^j \right). \tag{2.15}$$

Виконання умов достатності ресурсу польоту визначається формулою (2.16):

$$\sum_{j=1}^{|X^i|-1} d\left(x_j^i, x_{j+1}^i\right) < R^i, i = \overline{1,b}. \quad (2.16)$$

Виконання умови одноразового обстеження всіх об’єктів та обстеження кожного об’єкту в результаті виконання місії описується формулами (2.11) та (2.12).

2.4 Вхідні дані задачі

Усі вхідні дані можна розділити на дві категорії: введення документа та введення користувача. Розглянемо кожну категорію докладніше.

Вхідним документом є файл у форматі json, що містить інформацію про об’єкт дослідження, склад та ЛА. На рисунку 2.1 показано фрагмент вхідного файлу, що містить інформацію про об’єкт дослідження.


```
{
    "customers": [
        {
            "idx": 1,
            "duration": 2.0,
            "x": -37.67,
            "y": 53.89
        },
        {
            "idx": 2,
            "duration": 4.0,
            "x": -1.32,
            "y": -33.5
        },
        {
            "idx": 3,
            "duration": 3.0,
            "x": -7.91,
            "y": 1.86
        },
```


Рисунок 2.1 - Фрагмент вхідного файлу з інформацією про об’єкти

Кожне поле містить такі властивості:

- idx - ідентифікатор об'єкта дослідження;
- Тривалість - час обстеження суб'єкта;
- x——Координата абсцис об’єкта дослідження на площині;

- y——Координата ординати об’єкта дослідження на площині.

На рисунку 2.2 показано фрагмент вхідного файлу, що містить інформацію про репозиторій програмного забезпечення.


```
"depots": [
    {
        "idx": 151,
        "x": 98.3,
        "y": -11.39
    },
    {
        "idx": 152,
        "x": -29.36,
        "y": 67.89
    },
    {
        "idx": 153,
        "x": -63.26,
        "y": -85.15
    }
],
```


Рисунок 2.2 – Фрагмент вхідного файлу з інформацією про депо

Кожне поле містить такі властивості:

- idx——ідентифікатор складу;

- x——Абсциса відрізка літака;

- y - ордината аеропорту.

На рисунку 2.3 показано фрагмент вхідного файлу, що містить інформацію про ЛА.


```
"uavs": [
    {
        "idx": 1,
        "speed": 30.0,
        "travel_cost": 5.0,
        "max_distance": 500
    },
    {
        "idx": 2,
        "speed": 50.0,
        "travel_cost": 6.0,
        "max_distance": 300
    },
    {
        "idx": 3,
        "speed": 40.0,
        "travel_cost": 4.0,
        "max_distance": 700
    },
```

Рисунок 2.3 – Фрагмент вхідного файлу з інформацією про ЛА

Кожне поле містить такі властивості:

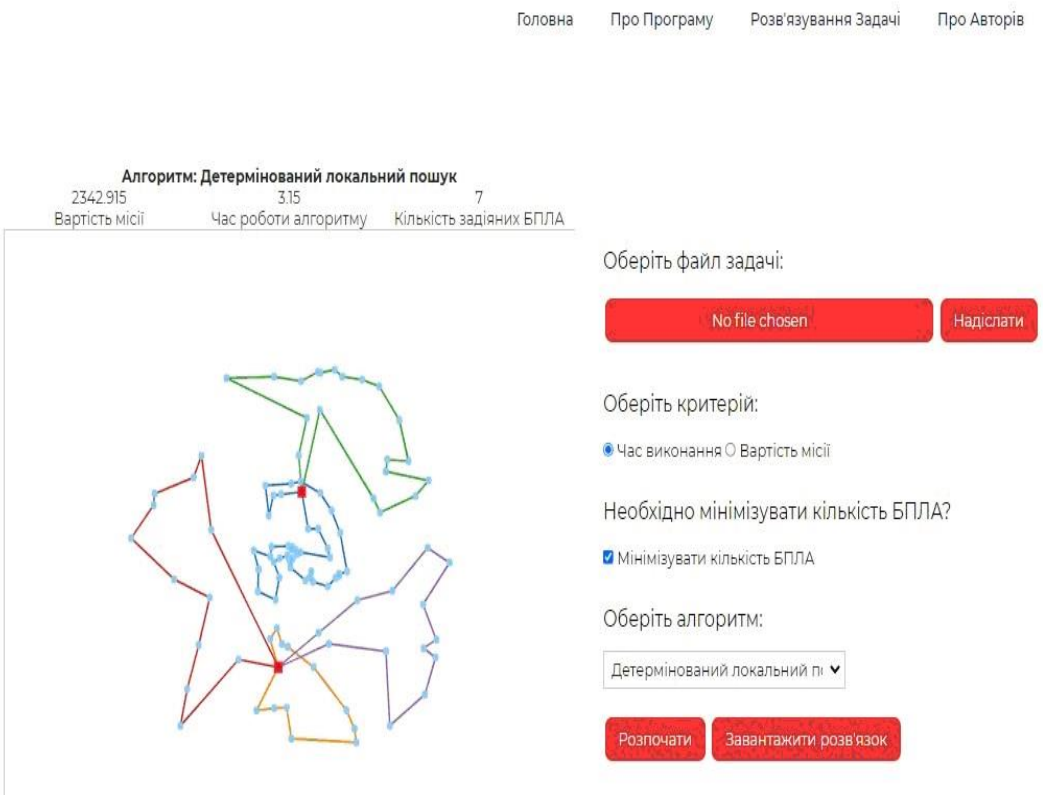
- idx - ідентифікатор Лос-Анджелеса;
- Speed - швидкість літака;
- Travel_cost – вартість одиниці пройденої відстані;
- max_distance – рейсові ресурси Лос-Анджелеса.

Вхід користувача - це вибір варіантів розв'язання проблеми та критеріїв. Користувач може вибрати один із запропонованих алгоритмів (детермінований локальний пошук (DLP), заборонений пошук (пошук табу, TP), алгоритм імітованого відпалу (AIV) або алгоритм прискореного ймовірнісного моделювання) та один із критеріїв (час виконання завдання або вартість маршрутизації). і необхідність мінімізації кількості ЛА).

2.5 Вихідні дані задачі

Вихідними даними для задачі є візуалізація виконання алгоритму на декартовій координатній площині, користувачі можуть переглядати кожен окремо обраний алгоритм і критерій, а також файл у форматі .json, що містить інформацію про маршрут.

На рисунку 2.4 показаний приклад результатів візуалізації DLP на основі критеріїв загальної вартості місії та необхідності мінімізації кількості залучених літаків.



0

Рисунок 2.4 – Приклад візуалізованих результатів виконання алгоритмів

Основні результати програми також представлені у вигляді таблиці, де вказано загальну вартість маршрутів, час роботи алгоритму та кількість маршрутів, тобто кількість задіяних літаків. Після створення маршруту користувачі можуть зберегти результати у файлі .json, що містить інформацію про маршрут. На малюнку 2.5 показано приклад вихідного файлу.

```
{
    "routes": [
        {
            "uav": 1,
            "depot": 152,
            "customers": [23, 58, 104, 3, 148, 2, 10]
        },
        {
            "uav": 3,
            "depot": 151,
            "customers": [76, 102, 16, 30, 20, 4, 111, 98, 79]
        },
        {
            "uav": 6,
            "depot": 153,
            "customers": [22, 101, 70, 17, 1, 11, 68]
        },
```


Рисунок 2.5 – Приклад вихідного файлу з інформацією про маршрути

Кожне поле містить такі властивості:

- Дрон - ідентифікатор літака;

- Warehouse - ідентифікатор складу;

- Клієнти - Список ідентифікаторів суб'єктів дослідження.

У другій частині дається змістовна постановка проблеми. Розглянемо проблему планування місії для неоднорідної групи літальних апаратів, яка передбачає вимірювання та/або обслуговування заданого набору об’єктів на місцевості. При цьому ЛА не є апріорі прив’язаними до конкретної ділянки, а метою є мінімізація витрат часу або вартості маршруту з використанням мінімальної кількості ЛА або будь-якої кількості доступних ЛА за певних обставин. Додаткові обмеження.

Заданій задачі відносять дві категорії: задачу мінімізації критерію із заданою кількістю ЛА та задачу мінімізації кількості ЛА шляхом подальшого вдосконалення критерію. Також враховуйте такі критерії: час виконання завдання та вартість маршруту. У результаті було отримано чотири різні питання. Завдання алгоритму полягає в тому, щоб покращити критерій для заданої кількості LA, якщо вибрано клас, який мінімізує час або вартість виконання. Якщо категорії вибрано для мінімізації кількості LA та подальшого уточнення критерію, першим завданням алгоритму є знайти мінімальну кількість LA перед тим, як критерій потрібно буде уточнити. Для

даної задачі оптимізації запропоновано булеві та комбіновані формули. Однак комбінована формула більш компактна і її легше програмно реалізувати. У процесі побудови цих тверджень вводяться основні позначення, описуються обмеження та формулюються дві КФ: мінімізація загальної вартості завдання або часу його виконання.

Усі введення поділяються на дві категорії: введення документів і введення користувача. Вхідним документом є файл .json, який містить інформацію про об’єкт дослідження, склад та ЛА. Введення користувача - це вибір рішень проблеми та критеріїв. Вихідними даними завдання є візуалізація формування маршруту та зазначення відповідних декартових координат, які користувач може переглядати для кожного окремо обраного алгоритму та критерію, а також файл у форматі .json, що містить інформацію про маршрут.

3 МЕТОДИ МАРШРУТИЗАЦІЇ БЕЗПІЛОТНИХ ЛІТАЛЬНИХ АПАРАТІВ

3.1 Обґрунтування методів розв’язування

Важливо відзначити, що з точки зору програмування реалізація алгоритму не залежить від вибору стандарту. Тому стандартні «час виконання завдання» і «вартість маршруту» можна розглядати як одне завдання з різними CF. Оскільки ЗМТЗ є NP-комплексом за наявності кількох складів, то для нього рекомендується використовувати апроксимаційні алгоритми. Більшість таких алгоритмів складається з двох етапів – генерації початкового наближення та його вдосконалення. 3.1 зображено загальну покрокову схему алгоритму.

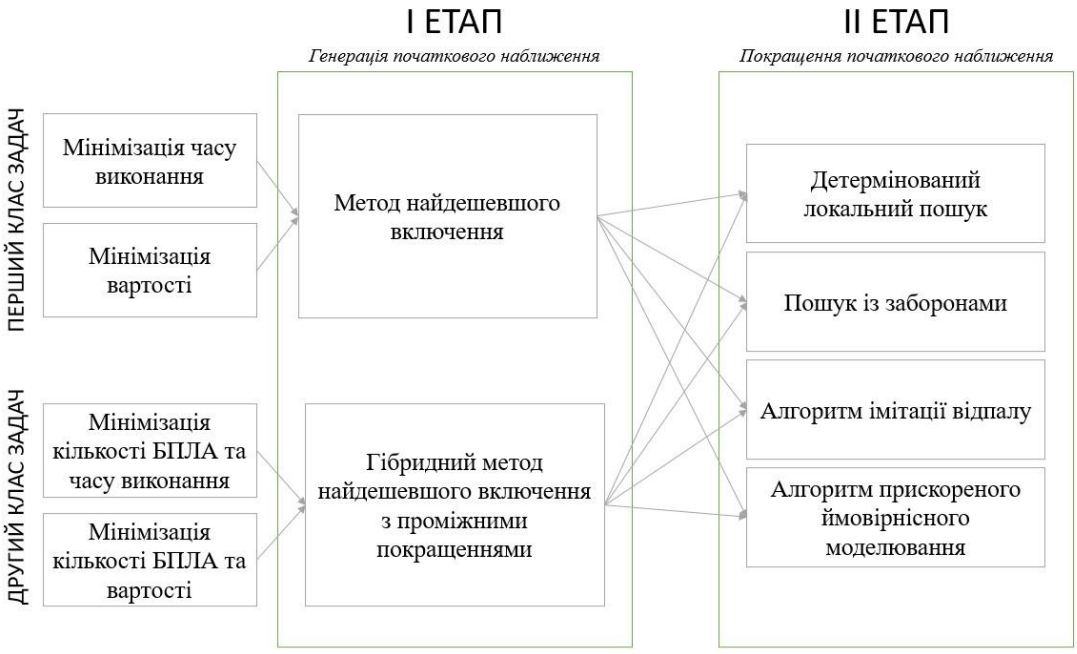


Рисунок 3.1 – Загальна поетапна схема роботи алгоритмів маршрутизації ЛА за наявності декількох депо

Перший етап методу залежить від категорії проблеми та критеріїв, другий етап залежить лише від критеріїв. Ми бачимо, що на першому етапі

або досягається мінімізація кількості ЛА разом з наступною мінімізацією КФ, або завдання мінімізації КФ не залежить від кількості ЛА. На другому етапі спостерігається лише збільшення CF, але не збільшення кількості LA.

3.2 Генерація початкового наближення

Розглянемо генерацію початкових наближень для двох різних класів задач: мінімізації кількості ЛА та КФ. Для задач першого типу якість початкового наближення мало впливає на кінцевий результат, тому рекомендується використовувати найдешевший алгоритм включення, оскільки він має низьку обчислювальну складність і простий у реалізації. Для задач другого типу необхідно знайти мінімальну кількість ЛА на першому етапі, оскільки алгоритм другого етапу для цього не підходить. Тому ми розробляємо гібридний алгоритм на основі найдешевшого методу включення та DLP для генерації початкових наближень.

3.2.1 Метод найдешевшого включення

Найдешевшим методом включення є конструктивний жадібний алгоритм. В ілюстративному матеріалі наведено структурну схему алгоритму.

Опишемо покрокову реалізацію найдешевшого способу включення.

Крок 1. Створюємо найкоротший шлях, що складається з сайту та об'єкта.

Крок 2. Визначаємо найдешевше включення об'єкта в існуючий маршрут. Крок 3. Визначити найдешевший спосіб створення нового маршруту.

Якщо створити маршрут дешевше, ніж включити його, перейдіть до кроку 4. Крок 4. Створюємо найдешевший маршрут.

В іншому випадку крок 5.

Крок 5. Виконуємо найдешевше включення об'єкта в існуючий шлях. Якщо є об'єкти, які не входять в маршрут, переходьте до кроку 2.

В іншому випадку крок 6.

Крок 6. Алгоритм завершується.

На малюнку 3.2 показано псевдокод для найдешевшого методу включення.

```
procedure НайдешевшеВключення
    невикористані об'єкти := всі об'єкти
    створити найдешевший маршрут
    видалити об'єкт зі списку невикористаних
    while (існують невикористані об'єкти)
        x := ціна найдешевшого можливого включення
        y := ціна створення найдешевшого маршруту
        if (x > y)
            здійснити найдешевше можливе включення
        else
            створити найдешевший маршрут
        endelse
        видалити об'єкт зі списку невикористаних
    endwhile
end
```

Рисунок 3.2 – Псевдокод для методу найдешевшого включення

Створення найдешевшого маршруту передбачає пошук найближчого до складу об'єкта і формування маршруту від складу до об'єкта і назад. Найдешевшим можливим включенням є включення об'єкта на маршрут з найнижчим значенням CF і для якого не вичерпані польотні ресурси повітряного судна, що летить за цим маршрутом.

3.2.2 Гібридний алгоритм найдешевшого включення з проміжними покращеннями

Гібридний алгоритм включення з мінімальною вартістю з проміжними вдосконаленнями є комбінацією алгоритму включення з мінімальною вартістю та DLP. В ілюстративному матеріалі наведено структурну схему алгоритму.

Опишемо покрокову реалізацію.

Крок 1. Створюємо найкоротший шлях, що складається з сайту та об'єкта.

Крок 2. Шукаємо найдешевший маршрут серед існуючих. Якщо є найдешевше включення, переходьте до кроку 3.

Крок 3. Ми робимо найдешевше включення та переходимо до кроку 2. В іншому випадку крок 4.

Крок 4. Перевіряємо, чи всі об'єкти включені в маршрут. Якщо всі об'єкти включені в маршрут, переходьте до кроку 6.

В іншому випадку крок 5.

Крок 5. Впроваджуємо DLP на існуючі маршрути. Якщо включено існуючий маршрут, перейдіть до кроку 2. В іншому випадку перейдіть до кроку 1.

Крок 6. Алгоритм завершується.

На рисунку 3.3 показаний псевдокод гібридного алгоритму включення з мінімальною вартістю з проміжними вдосконаленнями.

```
procedure НайдешевшеВключенняЗПокращенням
    невикористані об'єкти := всі об'єкти
    while (існують невикористані об'єкти)
        створити найдешевший маршрут
        while (існують можливі включення в маршрути що існують)
            виконати найдешевше включення
            if (немає можливих включень)
                виконати ДЛП для існуючих маршрутів
            endif
        endwhile
    endwhile
end
```

Рисунок 3.3 – Псевдокод гібридного алгоритму включення мінімальної вартості з проміжними вдосконаленнями

Проміжний покращений найдешевший гібридний алгоритм стримування — це модифікація найкращого алгоритму стримування, у якому

новий маршрут створюється лише в тому випадку, якщо стримування не може бути виконане в межах існуючого маршруту, який було вдосконалено за допомогою DLP.

3.3 Покращення початкового наближення

Удосконалення початкового наближення є другим етапом роботи апроксимаційного алгоритму, який включає оптимізацію КФ. Розглянемо чотири найефективніші за швидкістю і точністю обчислень методи розв'язування ЗКО: DLP, TP, AIV, G-алгоритм. Ми використовуємо ці алгоритми для вирішення поставленої задачі.

3.3.1 Детермінований локальний пошук

DLP є одним із найпростіших алгоритмів локального пошуку. З точки зору точності DLP належить до наближеного АКО, а з точки зору типу схеми розрахунку – до ітеративно-траєкторного алгоритму. Суть DLP полягає в пошуку кращих рішень навколо поточних рішень. На малюнку 3.4 показано псевдокод для DLP.

```
procedure ДЛП
    x := початкове наближення
    d := 1
    while d > 0 do
        X := окіл x
        for i=0 to |X| do
            d := Δ(x, X[i])
            if d > 0 and do
                x := X[i]
            endif
        endfor
    endwhile
end
```

Рисунок 3.4 – Псевдокод для ДЛП

Робота DLP починається з початкового наближення x. На кожній ітерації формується околиця O(x) поточного рішення. Для критерію вартості

завдання околицею рішень x є множина допустимих рішень, утворена переміщенням об’єкта в іншу позицію на поточному шляху або в будь-яку позицію на іншому шляху. Для критерію часу виконання завдання околиця рішення x — це набір прийнятних рішень, утворених шляхом розміщення об’єкта від найдовшого маршруту до іншого місця на тому ж або іншому маршруті. В околі O(x) шукається розв’язок y ∈ O(x), у якому значення функції згідно з рівнянням (3.1) є максимальним.

$$\triangle = \triangle(x, y) = F(x) - F(y), \tag{3.1}$$

де F це ЦФ.

Після цього візьміть рішення y як поточне рішення (x = y). Якщо <0, пошук закінчується, якщо >0, то перехід до наступної ітерації.

3.3.2 Пошук із заборонами

ТП є різновидом DLP. За точністю відноситься до наближеного АКО, а за типом схеми розрахунку – до ітераційно-траєкторного алгоритму. Удосконалення ТР полягає в тому, що під час роботи алгоритму зберігається певна кількість попередніх рішень, і алгоритм не може повернутися до них на наступній ітерації. Це зменшує ймовірність передчасної зупинки алгоритму на локальному мінімумі. На малюнку 3.5 показано псевдокод ТР.

```
procedure ПошукІзЗаборонами
    s := початкове наближення
    x := початкове наближення
    tabu_list := []
    while не виконується умова завершення do
        X := окіл x
        for i=0 to |X| do
            d = Δ(x, X[i])
            if d > 0 and X[i] ∉ tabu_list do
                x := X[i]
            endif
        endfor
        if Δ(s, x) > 0 do
            s := x
        endif
        tabu_list.append(x)
        if |tabu_list| > maxTabuSize do
            tabu_list.removeFirst()
        endif
    endwhile
end
```

Рисунок 3.5 – Псевдокод для ТП

Робота ТП починається з початкового наближення x і порожнього забороненого списку. s=x як поточне найкраще рішення. На кожній ітерації формується околиця O(x) поточного рішення. В околицях O(x) шукати розв'язки для y ∈ O(x), у яких значення Δ-функції (3.1) є максимальним і не входить до списку заборонених. Після цього візьміть рішення y як поточне рішення (x = y). Якщо x краще за s, то s = x. Поточне рішення x додається в кінець списку заборонених. Якщо довжина забороненого списку перевищує максимальну, перший елемент видаляється зі списку заборонених. Якщо умова завершення не виконується, потрібна наступна ітерація. Умова завершення: якщо поточне найкраще рішення s не покращується протягом максимальної кількості ітерацій, алгоритм завершується. Результат роботи – найкраще рішення.

3.3.3 Алгоритм імітації відпалу

AIV — це випадковий локальний пошук. За точністю відноситься до наближеного АКО, а за типом схеми розрахунку – до ітераційно-траєкторного алгоритму. AIV подібний до фізичного процесу контрольованого охолодження (також називається «відпал») і використовує впорядкований випадковий пошук. З термодинаміки відомо, що ймовірність переходу системи з одного стану в інший розраховується за формулою Больцмана-Гіббса (3.2):

$$p = e^{-\frac{\triangle E}{kT}}, \tag{3.2}$$

У формулі ΔE – зміна узагальненої енергії;

T——температура;

k — постійна Больцмана.

При повільному контрольованому охолодженні розплавленого металу кристалізація розплаву супроводжується загальним зниженням його енергії

E. Однак існують обставини, що дозволяють тимчасово збільшити його. Завдяки можливості короткочасного підвищення енергії можна уникнути локальних мінімумів, які виникають під час реалізації процесу. Зниження температури до абсолютного нуля унеможливлює самостійне збільшення енергії.

Для даної задачі аналогією для фізичних систем є ZKO, а аналогією для енергії є CF. Імовірність переходу розраховується за формулою (3.3):

$$p(\triangle, T) = e^{-\frac{\triangle}{T}}; \tag{3.3}$$

$$\text{де } \triangle = F(y) - F(x). \tag{3.4}$$

При високих температурах значення ймовірності переходу при A < 0 досить високі, тому робота алгоритму схожа на випадковий пошук. Зі зниженням температури ймовірність переходу для збільшення CF зменшується. На малюнку 3.6 показано псевдокод для AIV.

```
procedure ІмітаціяВідпалу
    x := початкове наближення
    T := початкова температура
    s := x
    while не виконується умова завершення do
        while не досягнута рівновага do
            for i = 0 to k do
                y := випадковий елемент з O(x)
                d = Δ(x, y)
                P := min(1, p(d, T))
                if P > random(0, 1) then
                    x := y
                    if Δ(s, x) > 0 do
                        s := x
                    endif
                endif
            endfor
        endwhile
        T := нове значення тепмератури
    endwhile
end
```

Рисунок 3.6 - Псевдокод для AIB

AIV працює з початковим наближенням x. І поточне найкраще рішення s = x. Встановіть початкову температуру T і запустіть цикл, що складається з k ітерацій. Виберемо випадковий розв'язок y ∈ 0(x). Функція Δ обчислюється за формулою (3.4), а ймовірність переходу P = min (1, p (Δ , T)), а p (Δ, T) обчислюється за формулою (3.3). Якщо P > random[0,1],

прийняти x = y як поточне рішення, а якщо $\Delta(s, x) > 0$, прийняти s = x як поточне найкраще рішення. Після кожного проходу необхідно перевіряти, чи досягнута рівновага. Якщо так, перевірте умови завершення. Якщо умова завершення не виконується, температура T знижується і починається наступний цикл. В іншому випадку алгоритм припиняє роботу, і результатом операції є поточне найкраще рішення s.

Розподіл температури є геометричним рядом, α ε [0,5; 0,99], тобто таке значення температури обчислюється за рівнянням (3.5):

$$T_{i+1} = \alpha T_i. \tag{3.5}$$

Розглянемо умову рівноваги. Після кожного прогону перевіряється зважена зміна ЦФ відносно кожного попереднього прогону за формулою (3.6):

$$d = \frac{|f_i - f_k|}{f_i}, \tag{3.6}$$

де fi – значення CF i-го прогону;

fk — поточне поточне значення CF.

Рівновага вважається досягнутою, якщо ∃ i = 0,k-1: d < епсилон.

Алгоритм вважається завершеним, якщо результати не покращуються протягом n>1 ітерацій.

3.3.4 Алгоритм прискореного ймовірнісного моделювання

Іншим представником стохастичних алгоритмів локального пошуку є алгоритм прискореного ймовірнісного моделювання, або алгоритм G. У порівнянні з AIV його обчислювальна складність нижча, а результати більш стабільні. Як і AIV, алгоритм G шукає навколо поточного наближення. Найкраще наближення відразу приймається як поточне значення, тоді як найгірше наближення також приймається як поточне значення з деякою ймовірністю.

Однак, на відміну від AIV, ймовірність переходу до гіршого варіанту залежить лише від зміни CF, а обмежуючим фактором переходу при збільшенні CF є поріг, який знижується під час роботи алгоритму. На малюнку 3.7 показано псевдокод алгоритму G.

```
procedure Gалгоритм
    x := початкове наближення
    μ := 0
    s := x
    while O(x) не обстежений повністю do
        while не досягнута рівновага do
            for i = 0 to k do
                y := випадковий елемент з O(x)
                P := p(x, y)
                ξ = μ + random[0, 1] * (1 - μ)
                if P > ξ then
                    x := y
                    if Δ(s, x) > 0 do
                        s := x
                    endif
                endif
            endfor
        endwhile
        μ := наступне значення
    endwhile
end
```

Рисунок 3.7 – Псевдокод для G-алгоритму

Робота алгоритму O починається з початкового наближення x. s=x як поточне найкраще рішення. Встановіть початкову температуру T і запустіть цикл, що складається з k ітерацій. Виберемо випадковий розв'язок y ∈ O(x). Імовірність того, що p переходить від x до y, обчислюється за формулою (3.7):

$$p(x,y) = \begin{cases} \min\{1, \varphi(x,y)\}, \varphi(x,y) \geq 0, \\ 0, \text{у іншому разі,} \end{cases} \tag{3.7}$$

$$\text{де } \varphi(x,y) = 1 - \frac{F(y)-F(x)}{F(x)}. \tag{3.8}$$

Поріг для переходу < f розраховується згідно з рівнянням (3.9):

$$\xi = \mu + random[0,1](1-\mu), \tag{3.9}$$

де miu — це значення, додане на кожній ітерації, а його наступне значення формується за допомогою дельта-функції на основі попереднього значення, як показано у формулі (3.10):

$$\mu_{i+1} = G(\mu_i). \tag{3.10}$$

Типовою функцією G є (3.11):

$$G_i(x) = \left(x^{\frac{1}{i}} + b\right)^k, i \in \mathbb{N}, 0 < b < 1. \quad (3.11)$$

Якщо ймовірність переходу p > ξ візьміть x = y як поточне рішення, а якщо F(s) - F(x) >0, візьміть поточне найкраще наближення s = x. Після кожного запуску перевіряйте, чи досягнуто рівноваги. Якщо так, перевірте умову припинення. Якщо умова завершення не виконується, потрібно змінити значення μ на наступне та почати наступний запуск. В іншому випадку алгоритм припиняє роботу, і результатом алгоритму є поточне найкраще рішення 5.

^-Умови рівноваги та критерії завершення алгоритму такі самі, як і AIV. Вони описані в розділі 3.3.3.

У розділі 3 визначено, що для ЗМТЗ рекомендовано використовувати наближений алгоритм через його складність за наявності кількох складів. Більшість апроксимаційних алгоритмів складається з двох етапів – генерації початкового наближення та його вдосконалення.

Генерація початкових наближень розглядається для двох різних класів задач: мінімізації кількості ЛА та КФ. Для задач першого типу якість початкового наближення мало впливає на кінцевий результат, тому рекомендується використовувати найдешевший алгоритм включення, оскільки він має низьку обчислювальну складність і простий у реалізації. Для задач другого типу необхідно знайти мінімальну кількість ЛА на першому етапі, оскільки алгоритм другого етапу для цього не підходить. Тому для генерації початкових наближень був розроблений гібридний алгоритм, заснований на найдешевшому методі включення та DLP. Для кожного алгоритму наведено його робочу блок-схему, покрокову реалізацію та псевдокод.

Удосконалення початкового наближення є другим етапом роботи апроксимаційного алгоритму, який включає оптимізацію КФ. Розроблено чотири найефективніших за швидкістю та точністю обчислень методів розв’язання ZKO: DLP, TP, AIV, O алгоритм. Для кожного алгоритму надається покрокова реалізація та псевдокод.

4 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

4.1 Засоби розробки

Програмний продукт розроблено з використанням операційної системи Ubuntu 20.04, текстового редактора Sublime Text 3, мов програмування Python і JavaScript, мови веб-розмітки HTML5, таблиць стилів CSS3 та інструментів розробки, таких як бібліотеки NumPy, Flask і WebGL.

Ubuntu — це операційна система для робочих станцій, комп'ютерів і серверів; найпопулярніший у світі дистрибутив Linux. Це операційна система на основі ядра Linux, придатна для використання на ПК, ноутбуках і серверах. Він містить усі необхідні програми: для роботи в Інтернеті, пакети офісних програм для роботи з текстами, електронними таблицями та презентаціями, програми для спілкування в Інтернеті тощо.[49]

Sublime Text 3 — швидкий кросплатформенний текстовий редактор. Він має потужний API Python, який дозволяє плагінам розширювати вбудовані функції. Контроль пакетів можна налаштувати за допомогою панелі команд, забезпечуючи легкий доступ до тисяч пакетів, створених спільнотою [50].

Python — це інтерпретована об'єктно-орієнтована мова сценаріїв високого рівня загального призначення з динамічною семантикою. Розширені вбудовані структури даних у поєднанні з динамічною типізацією та динамічним зв'язуванням роблять його дуже привабливим для швидкої розробки додатків і для використання в якості сценаріїв або для склеювання існуючих компонентів [51].

JavaScript — це мова програмування високого рівня, розроблена Netscape Communication і Sun. JavaScript побудований на основі Java, але не є строго типізованим. У той же час він підтримує багато синтаксичних конструкцій Java, але не має поняття класів і використовує лише невеликий набір типів даних: числа, рядки та логічні значення [52].

HTML 5 є вдосконаленою версією стандарту HTML, започаткованого WHATWG у червні 2004 року, і його специфікація була названа «Веб-додатки 1.0» [53]. HTML – це мова веб-розмітки, яка використовується для форматування вмісту.

CSS — це специфікація, розроблена W3C для розширення можливостей форматування документів XML і HTML для спрощення керування їх відображенням у веб-браузерах. У створеній об’єктній моделі документа останній уявляється як такий, що складається з набору об’єктів. Усі об’єкти в документі взаємозалежні та об’єднані спільною структурою дерева. Використовуйте CSS, щоб відокремити структуру та вміст документа від рівня, на якому він представлений користувачам. Таблицю можна використовувати в багатьох документах [52].

NumPy — це програмний пакет, який додає підтримку Python для великих багатовимірних масивів і матриць, а також велику бібліотеку розширених математичних функцій для роботи з цими масивами. Основними програмними пакетами, які доповнюють NumPy є: SciPy і Matplotlib [54].

Flask — це платформа для створення веб-додатків, які діють як веб-сервери. Він не вимагає підключення додаткових бібліотек і розроблений на мові Python.

WebGL — це реалізація інтерфейсу для обробки графічних елементів мови JavaScript. WebGL є частиною HTML 5, побудованої на стандарті OpenGL ES 2.0 і підтримується всіма сучасними браузерами, такими як Chrome, Opera, Mozilla Firefox, Safari, Chromium та іншими.

4.2 Архітектура програмного забезпечення

Наведемо діаграми розгортання структури, послідовності, дії та класи, щоб зрозуміти логіку програми. Ми також опишемо специфікації функцій в окремих файлах.

4.2.1 Схема структурна розгортання

На графічному матеріалі показані варіанти конструктивного розміщення. Опишемо його докладніше. Програма складається з серверної та клієнтської частин.

Ієрархія така:

– Сервер, включаючи серверну операційну систему, яка в свою чергу містить веб-сервер, який складається з двох компонентів: Api і Routing;

– Клієнт, який включає клієнтську операційну систему; яка, у свою чергу, включає клієнтський браузер.

4.2.2 Схема структурна послідовності

На графічному матеріалі подано структурну схему послідовності. Опишемо послідовність операцій докладніше:

– З клієнта на портал: «Завантажити файл з умовами завдання»;

– Від клієнта до порталу: «Вибрати параметри завдання»;

– Від клієнта до порталу: «Вибір критеріїв завдання»;

- З порталу на сервер: «Надіслати запит для пошуку рішення»;

– На сервері: «Почати пошук вирішення проблеми»;

– З порталу на сервер: «Надіслати запит на найкраще поточне рішення»;

– Від порталу до клієнта: «показати найкраще поточне рішення»;

- Від сервера до порталу: «Відправити остаточне рішення проблеми»;

– Від порталу до клієнта: «Показати кінцеве рішення»;

– Від клієнта до порталу: «Надіслати запит на завантаження файлу з остаточним рішенням»;

- Від порталу до клієнта: «Надіслати файл, що містить остаточне рішення».

4.2.3 Схема структурна діяльності

На графічному матеріалі подано структурну діаграму діяльності. попит замовника:

– Завантажувати файли з вхідними даними;

– Виберіть алгоритм: DLP, TP, AIV або G-алгоритм;

– Виберіть критерій: час виконання або вартість маршруту;

- Виберіть кількість LA, яку потрібно мінімізувати.

Щоб реагувати на дії клієнта, портал повинен надати остаточне вирішення проблеми. Після цього клієнт може переглянути та завантажити отримане рішення, на цьому процес завершується.

4.2.4 Діаграма класів

Діаграми класів наведені в матеріалах рисунків. У верхній частині діаграми є базовий клас Node, від якого успадковуються класи Customer і Depot. Клас Node, у свою чергу, успадковує клас Position. Клас Graph містить список об’єктів класу Depot, Customer і UAV. Клас Route містить об'єкт класу UAV і список об'єктів класу Customer. Клас Solution містить список об'єктів класу Route.

– Клас Position має поля x і y з плаваючою комою та метод distance, який повертає значення з плаваючою комою.

– Клас Node має метод distance, який повертає значення з плаваючою комою.

– Класи Depot і Customer є нащадками класу Node і не мають власних полів і методів.

– Клас Graph має поля customers (список об’єктів класу Customer), depots (список об’єктів класу Depot), uavs (список об’єктів класу UAV) та методи, які повертають об’єкти irc_greedy, irc_hybrid validate. класу Solution, який повертає логічне значення.

– У класі UAV є depot (об’єкт класу Depot), ресурси типу float і поля швидкості.

– Клас Route має поля targets (список об’єктів класу Customer), uav (об’єкт класу UAV) і методи equals (повертає логічне значення), length і score (повертає значення з плаваючою комою), update (виконання не повертає жодного значення).

– Клас Solution має поле routes (список об’єктів класу Routes), метод DLP (не повертає значення), equals (повертає логічне значення), Galgorithm (не повертає значення), is_valid (не повертає значення), length (повертає значення з плаваючою комою), SA (не повертає значення), fraction (повертає значення з плаваючою комою), TabuSearch (не повертає значення).

4.2.5 Специфікація функцій

Ці функції згруповані в окремі файли відповідно до їх призначення: irc.py, search_utils.py, algorithms.py, solution.py, graph.py, app.py, api.py. Розглянемо кожен файл і функції в ньому окремо. irc.py Таблиця 4.1 надає детальний опис функцій у файлі irc.py, який містить функції, що використовуються для створення початкових наближень.

Таблиця 4.1 – Опис функцій із файлу irc.py

Назва функції	Опис
_get_uav()	Функція знаходження найкращого БПЛА для створення маршруту
_get_best_insertion()	Функція знаходження найкращого включення в існуючий маршрут
less_uavs()	Метод найдешевшого включення
shorter_routes()	Метод найдешевшого включення без оптимізації кількості БПЛА
get_initial_solution()	Інтерфейс для створення початкового наближення

Таблиця 4.2 містить детальний опис функцій у файлі search_utils.py, який містить допоміжні функції для алгоритму покращення початкового наближення.

Таблиця 4.2 – Опис функцій із файлу search_utils.py

Назва функції	Опис
get_benefit()	Зменшення ЦФ при видаленні об’єкта з маршруту
get_cost()	Збільшення ЦФ при додаванні об’єкту до маршруту
get_random_neighbour()	Вибір випадкового наближення з околу поточного

Таблиця 4.3 містить детальний опис функцій у файлі algorithm.py, що містить алгоритм, який використовується для покращення початкового наближення.

Таблиця 4.3 - Опис функцій із файлу algorithmspy

Назва функції	Опис
local_search()	ДЛП
local_search.get_better_solution()	Функція знаходження найкращого розв’язку з околу поточного
tabu_search()	ТП
tabu_search.get_better_solution()	Функція знаходження найкращого розв’язку з околу поточного з урахуванням списку заборон
simulated_annealing()	АІВ
g_algorithm()	G-алгоритм

Таблиця 4.4 містить детальний опис функцій у файлі Solution.py, який містить класи рішень і допоміжні класи.

Таблиця 4.4 - Опис функцій із файлу solution.py

Назва функції	Опис
solution.score()	Значення ЦФ для поточного розв’язку
solution.update()	Оновлення ЦФ поточного розв’язку
solution.for_render()	Представлення розв’язку для відображення
route.update()	Значення ЦФ для маршруту
route.score()	Оновлення ЦФ маршруту
node.distance()	Відстань між точками графу

Таблиця 4.5 містить детальний опис функцій у файлі graph.py, який містить клас graph.

Таблиця 4.5 - Опис функцій із файлу graph.py

Назва функції	Опис
graph.from_json()	Зчитування графу з файлу формату .json
graph.for_render()	Представлення графу для відображення

Таблиця 4.6 містить детальний опис функцій у файлі app.py, який містить функції, які використовує веб-сервер.

Таблиця 4.6- Опис фукцій із файлу app.py

solver()	Функція для відображення сторінки з побудовою маршрутів
authors()	Функція для відображення сторінки з інформацією про авторів
solution()	Функція, що повертає найкращий поточний розв'язок

Таблиця 4.7 містить детальний опис функцій у файлі api.py, який містить функції, що використовуються для зв'язку між веб-сервером і алгоритмом.

Таблиця 4.7- Опис функцій із файлу api.py

Назва функції	Опис
start_solving()	Функція для початку розв'язання задачі
get_current_solution()	Функція, що повертає найкращий поточний розв'язок для відображення
get_graph()	Функція, що повертає граф для відображення

Сторінки HTML для побудови маршруту мають функцію plot(), яка відображає об'єкти, зупинки та маршрути в декартовій системі координат.

4.3 Настанова з використання

Щоб користуватися системою, користувачі повинні мати мінімальні навички роботи з комп'ютером. Для полегшення розуміння функцій програмного продукту на рисунку 4.1 зображено діаграму переходів між сторінками.

Рисунок 4.1 – Схема переходу між сторінками програмного продукту

Використання програми починається з головної сторінки. Звідси користувач може перейти на сторінку з можливостями вирішення проблем, на сторінку з порівнянням продуктивності алгоритмів і на сторінку з інформацією про програму та авторів.

Опишемо більш детально основні операції, які користувачі можуть виконувати із запропонованою системою. Головна сторінка цього програмного продукту представлена на рисунку 4.2.

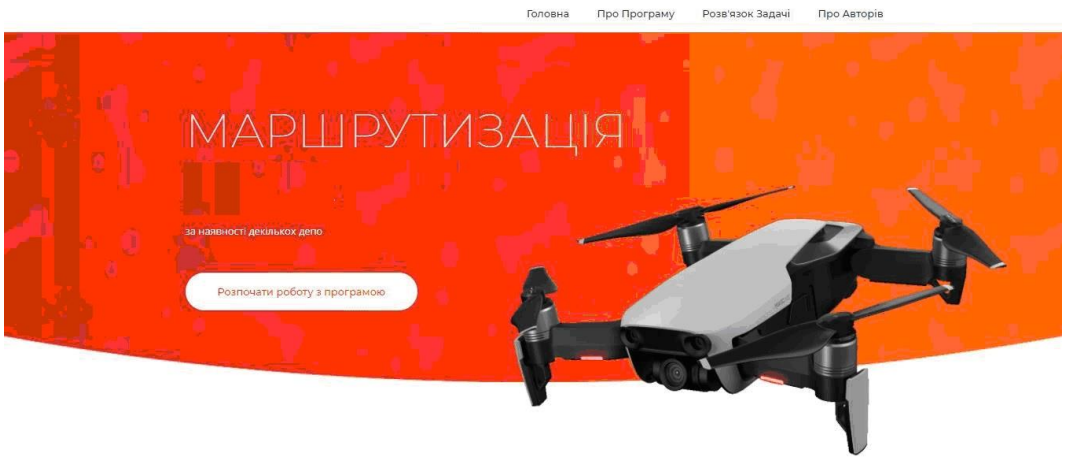


Рисунок 4.2 – Головна сторінка

На домашній сторінці є меню, яке використовується на всіх сторінках, назва програмного продукту та кнопка, яка спрямовує вас на сторінку

усунення несправностей. Розглянемо головне меню програмного продукту (рис. 4.3).

Рисунок 4.3 – Головне меню

Реакція системи на дії користувача при використанні головного меню наведена в таблиці 4.8.

Таблиця 4.8 – Реакція системи на дії користувача на головному меню

№	Опис-реакція системи
1	При натисканні на логотип система відкриває головну сторінку.
2	При натисканні на кнопку «Головна» система відкриває головну сторінку.
3	При натисканні на кнопку «Про програму» система відкриває сторінку, де користувач має змогу ознайомитися з інформацією про запропонований програмний продукт
4	При натисканні на кнопку «Розв'язування задачі» система відкриває сторінку, де користувач має змогу розв'язати свою задачу.
5	При натисканні на кнопку «Про авторів» система відкриває сторінку, де користувач має змогу ознайомитися з інформацією про авторів.

На малюнку 4.4 показано сторінку з інформацією про програму. На цій сторінці користувачі можуть ознайомитися з інформацією про запропоновані програмні продукти.

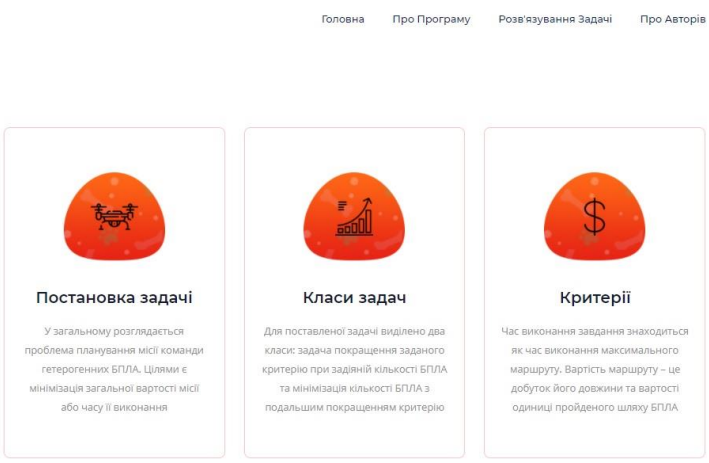


Рисунок 4.4 – Сторінка з інформацією про програму

На сторінці «Розв’язання задачі» (рис. 4.5) користувач може завантажити свою задачу, розв’язати її за одним із запропонованих алгоритмів за одним із критеріїв, а також переглянути візуалізований розв’язок у декартовій системі координат і, наприклад, за потреби завантажити. файли з результатами завдання на свій комп’ютер.

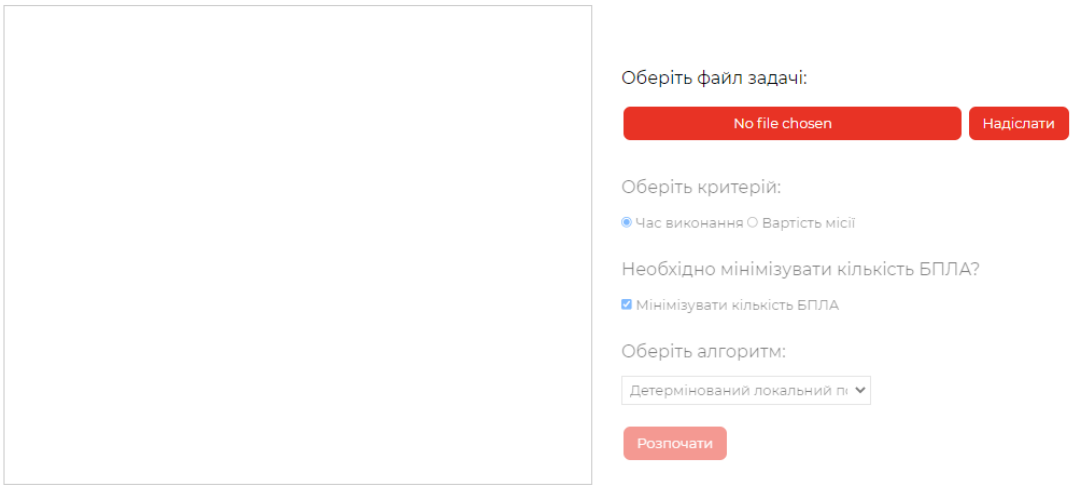


Рисунок 4.5 – Сторінка «Розв’язування задачі»

Після натискання кнопки «Файл не вибрано» система дозволяє користувачеві завантажити необхідні файли .json, необхідні для запуску програми. Після натискання кнопки «Відправити» відобразиться карта з

об'єктами і складами в декартовій системі координат. Після цього поле стає активним, і користувачеві потрібно вибрати критерій (час місії або вартість місії), що вказує на необхідність мінімізації кількості літальних апаратів і вибрати алгоритм (DLP, TP, AIV або G-алгоритм). Якщо жоден критерій не вибрано, рішення за замовчуванням використовує критерій часу виконання. Якщо не вибрано жоден алгоритм, рішення за замовчуванням використовує DLP. Якщо рішення щодо мінімізації не прийнято, за замовчуванням вибрано параметр «Так». Після того як користувач заповнив усі поля, йому необхідно натиснути кнопку «Пуск», після чого система видасть результати роботи. Після завершення роботи програми з'являється кнопка «Завантажити рішення», яка дозволяє користувачеві зберегти файл .json, що містить інформацію про маршрутизацію комп'ютера. На рисунку 4.6 наведено приклад результатів програми, коли користувач вибирає критерій «час виконання», вирішує, що необхідно мінімізувати кількість ЛА, і вибирає алгоритм вирішення проблеми DLP.

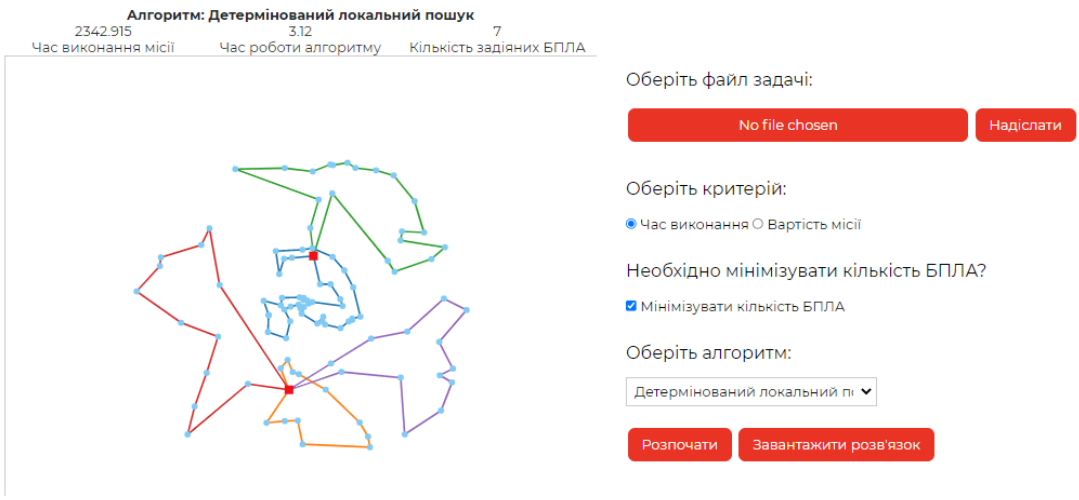


Рисунок 4.6 – Приклад результату роботи програми

Результати роботи програми подаються у вигляді: назва алгоритму, загальний час виконання завдання, час виконання алгоритму, кількість маршрутів, тобто кількість задіяних літаків, візуалізація результатів.

Розділ 4 досліджує основні підходи до розробки та дає їх конкретні характеристики. Тому при написанні програми використовувалася операційна система Ubuntu 20.04, текстовий редактор Sublime Text 3, мови програмування Python і JavaScript, мова веб-розмітки HTML5, таблиці стилів CSS3 і бібліотеки NumPy, Flask і WebGL.

Архітектура програмного забезпечення наведена за допомогою структурних діаграм розгортання, діаграм послідовності, діаграм активності та діаграм класів, які наведені в графічному матеріалі. Програма складається з серверної та клієнтської частин.

Описано функціональні характеристики. Так, функції згруповані в окремі файли відповідно до їх призначення: irc.py, search_utils.py, algorithms.py, solution.py, graph.py, app.py, api.py. HTML-сторінка конструктора маршрутів має окремі функції для відображення об’єктів, зупинок і маршрутів у декартовій системі координат.

Щоб користуватися системою, користувачі повинні мати мінімальні навички роботи з комп’ютером. Цей програмний продукт складається з 4 сторінок, і користувачі можуть вільно перемикатися між сторінками. Програма запускається з головної сторінки. Звідти користувач може перейти на сторінку, яка містить інформацію про програму, можливості вирішення проблеми та інформацію про автора програми. на сторінці

«Вирішити задачу» дозволяє користувачам завантажити файл, що містить їхню задачу, вибрати необхідні критерії, необхідність мінімізації кількості ЛА та алгоритм. Після цього система починає розрахунок і виводить результати у вигляді візуалізації в декартовій системі координат. Користувач також може завантажити на свій комп’ютер файл у форматі .json, що містить інформацію про маршрутизацію.

5 ДОСЛІДЖЕННЯ ЕФЕКТИВНОСТІ РОЗРОБЛЕНИХ МЕТОДІВ

5.1 Умови проведення досліджень

Усі дослідження проводилися на ноутбуці HP ProBook 4540s із такими характеристиками:

- процесор Intel® Core™ i5-3210M з двома ядрами і чотирма потоками, тактова частота ядра 2,6 ГГц;

- Об'єм оперативної пам'яті - 8 Гб.

Для проведення дослідів було сформовано 9 завдань:

- 3 місії, 2 склади і 100 об'єктів;

- 3 місії, 3 склади і 150 об'єктів;

- 3 місії, 4 склади і 200 об'єктів.

При порівнянні ефективності всіх алгоритмів використовували середнє значення CF (розраховане як середнє гармонічне) і час виконання 10 прогонів алгоритму. Середнє гармонічне обчислюється за формулою (5.1):

$$H = \frac{n}{\frac{1}{x_1}+\frac{1}{x_2}+\cdots+\frac{1}{x_n}}, x_1, x_2, \ldots, x_n > 0. \quad (5.1)$$

Рекомендується використовувати середнє гармонічне, оскільки воно більш чутливе до менших значень, ніж середнє арифметичне та середнє геометричне. Це дозволяє розглядати стабільність алгоритму, оскільки алгоритми з менш стабільними результатами (більшою дисперсією) матимуть нижче середнє арифметичне значень CF.

Для всіх алгоритмів максимальний час роботи обмежений 60 секундами. Якщо алгоритм працює довше, то результат його роботи вважається найкращим поточним рішенням на той момент. Використовуйте локальний пошук для вибору параметрів алгоритму. Експерименти проводились окремо для різних критеріїв: Критерій 1 – час виконання завдання Критерій 2 – вартість маршруту.

5.2 Дослідження впливу параметрів на роботу алгоритмів

Різні алгоритми мають різні параметри. Наприклад, DLP взагалі не має параметрів, але параметри алгоритмів TP, AIV і G будуть розглянуті в наступних підрозділах. Використовуйте DLP для встановлення параметрів.

5.2.1 Параметри пошуку із заборонами

TP має два параметри: довжину забороненого списку та максимальну кількість ітерацій, які не покращують результати. В якості простору для локального пошуку вибираються значення параметрів з певним розміром кроку:

- Довжина списку банів: від 5 до 50 з кроком 5;
- Максимальна кількість ітерацій без покращення результатів: від 10 до 100 з кроком по 10.

В якості вихідного параметра алгоритму вибирається середнє значення в заданому діапазоні. Початкові та кінцеві значення параметрів ТП наведені в таблиці 5.1.

Таблиця 5.1 – Початкові та кінцеві параметри ТП

Параметр	Початкове значення	Кінцеве значення Критерію 1	Кінцеве значення Критерію 2
Довжина списку заборон	30	35	35
Максимальна кількість ітерацій без покращення результату	50	70	60

Для TP середнє значення CF для критерію 1 при виконанні всіх завдань з початковими значеннями параметрів склало 345,62, а для кінцевих значень параметрів

– 337,02. При виконанні всіх завдань при початкових значеннях параметрів середнє значення CF для критерію 2 склало 2434,35, тоді як середнє значення CF для кінцевих значень параметрів склало 2423,72.

5.2.2 Параметри алгоритму імітації відпалу

AIV має п’ять параметрів: початкова температура, тривалість циклу, коефіцієнт зниження температури, поріг рівноваги та максимальна кількість ітерацій без покращення результатів. В якості простору для локального пошуку вибираються значення параметрів з певним розміром кроку:

- Початкова температура: від 1 до 10, розмір кроку 1;

- Тривалість виконання: від 5 до 30 з кроком 5;

——Коефіцієнт пониження температури: від 0,9 до 0,99, крок 0,01;

- Поріг для досягнення рівноваги: від 0,001 до 0,01 з кроком 0,001;

- Максимальна кількість ітерацій без покращення результатів: від 10 до 100 з кроком по 10.

Початкові і кінцеві значення параметрів AIB наведені в таблиці 5.2.

Таблиця 5.2 – Початкові та кінцеві параметри AIB

Параметр	**Початкове значення**	**Кінцеве значення Критерію 1**	**Кінцеве значення Критерію 2**
Початкова температура	5	7	7
Тривалість прогону	20	20	15
Коефіцієнт зниження температури	0,95	0,98	0,99
Поріг досягнення рівноваги	0,005	0,01	0,01
Максимальна кількість ітерацій без покращення результату	50	30	40

Для AIV середнє значення CF для критерію 1 становило 334,23 під час виконання всіх завдань із початковими значеннями параметрів, а для кінцевих значень параметрів

- 319,78. При виконанні всіх завдань при початкових значеннях параметрів середнє значення CF для критерію 2 склало 2421,45, тоді як середнє значення CF для кінцевих значень параметрів склало 2372,12.

5.2.3 Параметри алгоритму прискореного ймовірнісного моделювання

Алгоритм G має чотири параметри: довжину циклу, коефіцієнт k дельта-функції, поріг для досягнення рівноваги та максимальну кількість ітерацій без покращення результатів. В якості простору для локального пошуку вибираються значення параметрів з певним розміром кроку:

- Довжина прогону: від 5 до 30 з кроком 5;
- Коефіцієнт k дельта-функції: від 0,0001 до 0,001 з кроком 0,0001;
- Поріг для досягнення рівноваги: від 0,001 до 0,01 з кроком 0,001;
- Максимальна кількість ітерацій без покращення результатів: від 10 до 100 з кроком по 10.

Початкові і кінцеві значення параметрів алгоритму G наведені в таблиці 5.3.

Таблиця 5.3 – Початкові та кінцеві параметри G-алгоритму

Параметр	Початкове значення	Кінцеве значення Критерію 1	Кінцеве значення Критерію 2
Довжина прогону	20	15	20
Коефіцієнт b для G-функції	0,0005	0,001	0,001
Поріг досягнення рівноваги	0,005	0,003	0,002
Максимальна кількість ітерацій без покращення результату	50	40	30

Для алгоритму G середнє значення CF для критерію 1 при виконанні всіх завдань при початкових значеннях параметрів склало 329,5, а для кінцевих значень параметрів - 315,61. При виконанні всіх завдань з початковими значеннями параметрів середнє значення CF для критерію 2 склало 2407,77, тоді як середнє значення CF для кінцевих значень параметрів склало 2316,43.

5.2.4 Результати налаштування параметрів алгоритмів

Оскільки для встановлення параметрів використовується алгоритм локального пошуку, значення CF зменшується. У таблиці 5.4 і 5.5 наведено

результати коригування параметрів алгоритму для Критерію 1 і Критерію 2 відповідно.

Таблиця 5.4 – Результати налаштування параметрів алгоритмів для Критерію 1

Алгоритм	ТП	AIB	G-алгоритм
Початкове значення ЦФ	345,62	334,23	329,5
Кінцеве значення ЦФ	337,02	319,78	315,61
Відсоток покращення	2,55%	4,5%	4,4%

Таблиця 5.5 – Результати налаштування параметрів алгоритмів для Критерію 2

Алгоритм	ТП	AIB	G-алгоритм
Початкове значення ЦФ	2434,35	2421,45	2407,77
Кінцеве значення ЦФ	2423,72	2372,12	2316,43
Відсоток покращення	0,43%	2,08%	3,9%

Після коригування параметрів алгоритму результати для критерію 1 покращилися в середньому на 3,8%, а для критерію 2 в середньому на 2,1%.

5.3 Порівняння результатів роботи алгоритму

Кожен алгоритм потрібно запустити для кожного з 9 файлів. Ефективність алгоритму оцінюється за двома критеріями оцінки алгоритму: якість кінцевого рішення та час роботи алгоритму. Ми перевіряємо кожен алгоритм за цими критеріями.

5.3.1 Дослідження кінцевих результатів

Вивчимо значення CF для критерію часу виконання завдання (критерій 1) і кількості ЛА, які необхідно мінімізувати. У таблиці 5.6 наведені значення CF, мінімізовані LA за критерієм 1.

Таблиця 5.6 – Значення ЦФ для Критерію 1 з мінімізацією по ЛА

Задача	Кількість об'єктів	Кількість депо	ДЛП	ТП	AIB	G-алгоритм
1	100	2	492,76	425,58	485,78	405,31
2	100	2	438,27	438,27	430,29	414,48
3	100	2	466,85	431,76	455,80	408,11
Середнє значення ЦФ			465,96	431,87	457,25	409,3
4	150	3	460,04	437,13	438,20	420,17
5	150	3	473,74	448,92	459,84	421,08
6	150	3	488,39	462,48	486,12	460,98
Середнє значення ЦФ			474,06	449,51	461,39	434,08
7	200	4	484,77	453,59	469,47	426,39
8	200	4	474,23	466,43	466,90	444,70
9	200	4	477,65	443,57	474,70	422,89
Середнє значення ЦФ			478,88	454,53	470,36	431,33
Загальне середнє значення ЦФ			472,99	445,30	463,01	424,90

Як видно з таблиці 5.6, алгоритм G має найнижче значення CF серед загальних середніх значень CF, тому він є найбільш придатним для типу поставленої задачі. Щоб наочно проілюструвати отримані результати, побудуємо графік залежності середнього CF від кількості об'єктів для різних алгоритмів (рис. 5.1), як критерій часу виконання завдання та необхідної мінімізації кількості об'єктів LA.

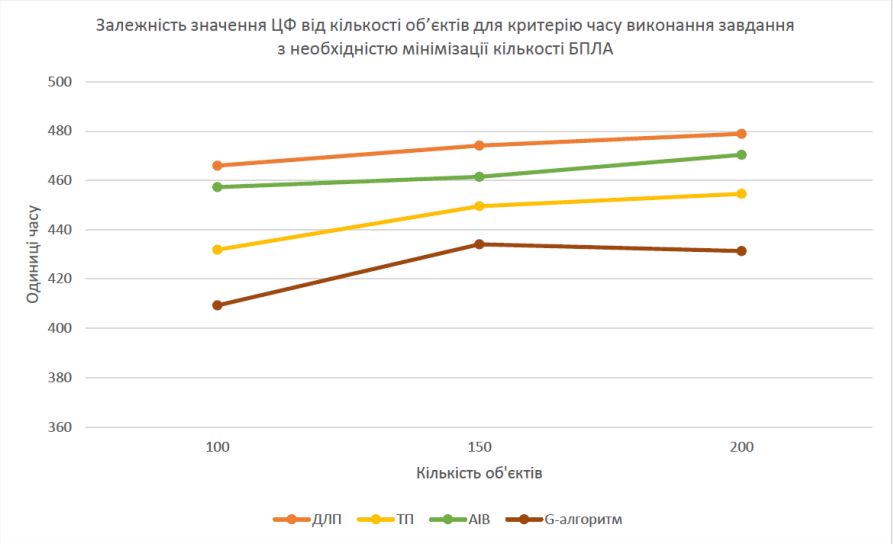


Рисунок 5.1 – Залежність значення CF від кількості об'єктів за критерієм часу виконання завдання та кількості літаків, які необхідно мінімізувати

На малюнку 5.1 показано графік залежності CF від кількості об’єктів для різних алгоритмів. DLP позначено тут помаранчевим кольором, TP — жовтим, AIV — зеленим, а алгоритм G — коричневим. По осі абсцис позначається кількість об’єктів, які розв’язали задачу, по осі ординат – значення CF. Обчисліть середнє значення CF для кожного алгоритму для різної кількості об’єктів і побудови графіків. З малюнка ми бачимо, що алгоритм G має найкращі результати для будь-якої кількості об’єктів, за ним йде TP, AIV на третьому місці, а результати DLP є найгіршими.

Ми проведемо дослідження щодо значення CF критерію вартості маршруту (критерій 2) та необхідності мінімізації кількості літаків. У таблиці 5.7 наведені значення CF, мінімізовані LA за критерієм 2.

Таблиця 5.7 – Значення ЦФ для Критерію 2 з мінімізацією по ЛА

Задача	Кількість об’єктів	Кількість депо	ДЛП	ТП	АІВ	G-алгоритм
1	100	2	2327,18	2327,18	2246,18	2178,08
2	100	2	2342,91	2342,91	2307,08	2269,46
3	100	2	2339,84	2334,54	2226,85	2227,90
Середнє значення ЦФ			2336,64	2334,87	2260,03	2225,14
4	150	3	2977,90	2950,72	2865,42	2802,85
5	150	3	3079,57	3079,57	3011,28	2942,18
6	150	3	2926,06	2921,68	2783,88	2758,31
Середнє значення ЦФ			2994,51	2983,99	2886,86	2834,44
7	200	4	3300,84	3300,84	3218,72	3296,69
8	200	4	3298,21	3298,21	3182,59	3148,07
9	200	4	3231,36	3229,56	3119,30	3045,26
Середнє значення ЦФ			3276,80	3276,20	3173,53	3163,34
Загальне середнє значення ЦФ			2869,32	2865,02	2773,48	2740,98

Як видно з таблиці 5.7, алгоритм G має найнижче значення CF серед загальних середніх значень CF, тому він є найбільш придатним для типу поставленої задачі.

Побудуємо графіки залежності значення CF для різних алгоритмів від кількості об’єктів за критерієм вартості маршруту (рис. 5.2) та необхідності

мінімізації кількості ПС. Обчисліть середнє значення CF для кожного алгоритму для різної кількості об’єктів і побудови графіків.

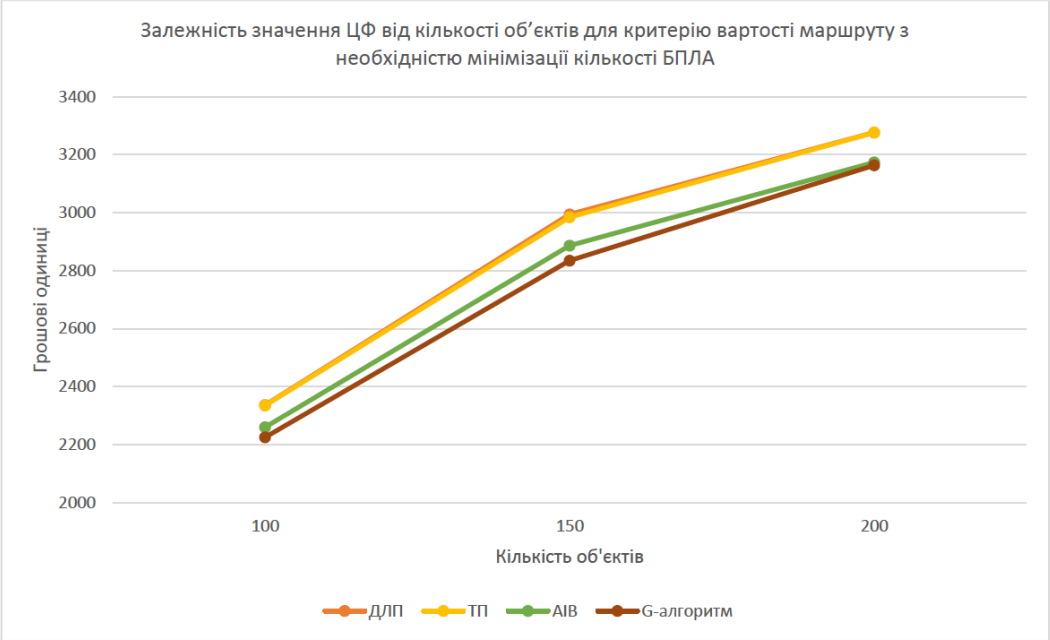


Рисунок 5.2 – Залежність величини CF від кількості об’єктів критерію вартості маршруту та необхідності мінімізації кількості ПС

На рисунку 5.2 зображено графік залежності CF від кількості об’єктів для різних алгоритмів. DLP позначено тут помаранчевим кольором, TP — жовтим, AIV — зеленим, а алгоритм G — коричневим. По осі абсцис позначається кількість об’єктів, які розв’язали задачу, по осі ординат – значення CF. На малюнку ми бачимо, що алгоритм G має кращі результати для будь-якої кількості об’єктів, за ним слідує AIV, тоді як результати DLP і TP приблизно однакові. Для 150 об’єктів TP працює трохи краще, але це покращення незначне.

Ми будемо вивчати значення CF для критерію часу виконання завдання (критерій 1) без мінімізації кількості LA. У таблиці 5.8 наведено значення CF, які не вдалося мінімізувати LA за критерієм 1.

Таблиця 5.8 – Значення ЦФ для Критерію 1 без мінімізації по ЛА

Задача	Кількість об'єктів	Кількість депо	ДЛП	ТП	AIB	G-алгоритм
1	100	2	374,94	339,12	306,98	320,83
2	100	2	279,37	332,99	327,48	259,11
3	100	2	347,75	281,65	347,48	324,26
Середнє значення ЦФ			334,02	317,92	327,32	301,4
4	150	3	289,38	272,26	327,59	254,89
5	150	3	347,97	300,76	309,81	304,64
6	150	3	375,92	306,14	327,94	300,76
Середнє значення ЦФ			337,75	293,05	321,78	286,76
7	200	4	358,92	358,9	359,7	331,19
8	200	4	369,76	356,88	289,08	269,51
9	200	4	298,07	299,78	304,67	265,81
Середнє значення ЦФ			342,25	338,52	317,82	288,84
Загальне середнє значення ЦФ			338,01	316,50	322,30	292,33

Як видно з таблиці 5.8, серед загальних середніх значень CF алгоритм G має найменше значення CF, тому він є найбільш придатним для типу поставленої задачі.

Побудуємо графік залежності КФ різних алгоритмів від кількості об'єктів (рис. 5.3) для критерію часу виконання завдання без мінімізації кількості ЛА. Обчисліть середнє значення CF для кожного алгоритму для різної кількості об'єктів і побудови графіків.

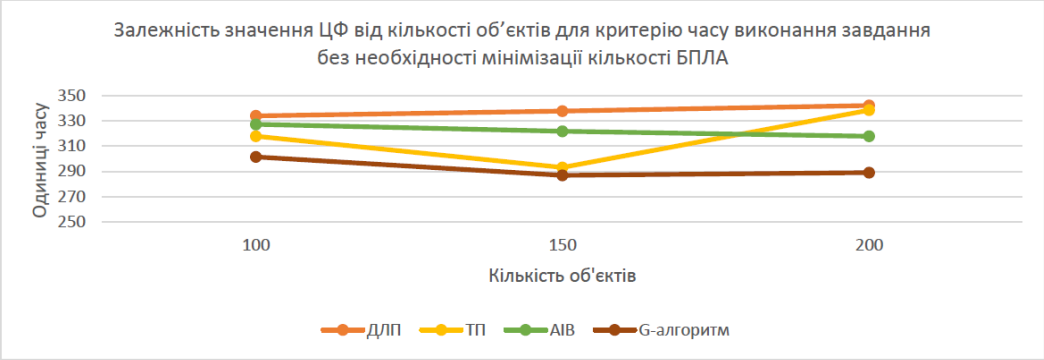


Рисунок 5.3 – Залежність значення CF від кількості об'єктів за критерієм часу виконання завдання без мінімізації кількості ЛА

На рисунку 5.3 зображено графік залежності CF від кількості об'єктів для різних алгоритмів. DLP позначено тут помаранчевим кольором, TP —

жовтим, AIV — зеленим, а алгоритм G — коричневим. По осі абсцис позначається кількість об'єктів, які розв'язали задачу, по осі ординат – значення CF. На малюнку ми бачимо, що алгоритм G має кращі результати для будь-якої кількості об'єктів, за яким слідує TP, але його продуктивність для великих даних набагато гірша, ніж AIV. Для невеликих даних AIV займає третє місце, а DLP виконує найгірше серед усіх завдань.

Ми будемо вивчати значення CF критерію вартості маршруту (критерій 2) без мінімізації кількості повітряних суден. У таблиці 5.9 наведено значення CF, які не вдалося мінімізувати LA за критерієм 2.

Таблиця 5.9 – Значення ЦФ для Критерію 2 без мінімізації по ЛА

Задача	Кількість об'єктів	Кількість депо	ДЛП	ТП	AIB	G-алгоритм
1	100	2	1825,99	1681,76	1413,79	1347,71
2	100	2	1555,31	1857,02	1760,62	1425,45
3	100	2	1745,28	1661,92	1372,53	1370,36
Середнє значення ЦФ			1708,86	1733,57	1515,65	1381,17
4	150	3	2156,4	1972,43	1929,99	1900,13
5	150	3	2287,51	1991,2	2252,11	2318,53
6	150	3	1890,92	2198,33	1716,51	1726,87
Середнє значення ЦФ			2111,61	2053,99	1966,20	1981,84
7	200	4	2275,6	1992,13	2152,77	2041,46
8	200	4	2036,06	2230,55	2454,21	2166,70
9	200	4	1985,95	2471,23	2338,88	1943,27
Середнє значення ЦФ			2099,20	2231,30	2315,29	2050,48
Загальне середнє значення ЦФ			1973,22	2006,29	1932,38	1804,50

Як видно з таблиці 5.9, серед загальних середніх значень CF алгоритм G має найнижче значення CF, тому він є найбільш придатним для типу поставленої задачі.

Побудуємо графік залежності CF різних алгоритмів від кількості об'єктів (рис. 5.4) для критерію вартості маршруту без мінімізації кількості повітряних суден. Обчисліть середнє значення CF для кожного алгоритму для різної кількості об'єктів і побудови графіків.

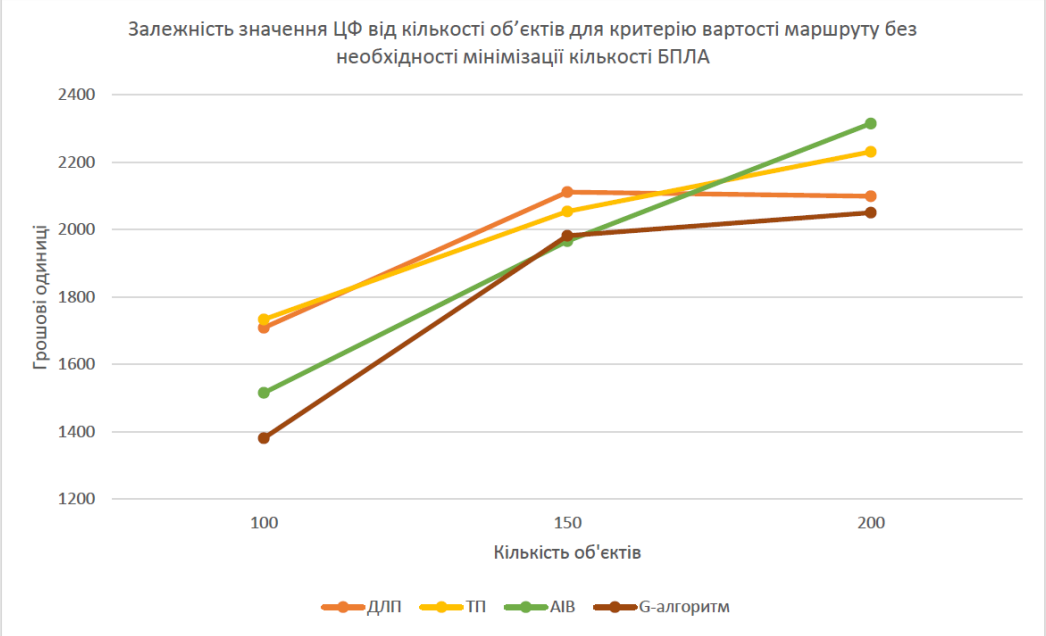


Рисунок 5.4 – Залежність значення CF від кількості об'єктів для критеріїв вартості маршруту без мінімізації кількості повітряних суден

На рисунку 5.4 зображено графік залежності CF від кількості об’єктів для різних алгоритмів. DLP позначено тут помаранчевим кольором, TP — жовтим, AIV — зеленим, а алгоритм G — коричневим. По осі абсцис позначається кількість об’єктів, які розв’язали задачу, по осі ординат – значення CF. З малюнка ми бачимо, що різні алгоритми працюють краще для різної кількості об’єктів. Таким чином, для невеликої кількості об’єктів алгоритм G є найкращим, а TP – найгіршим. Для середньої кількості об’єктів AIV є найкращим, а DLP – найгіршим. Для великої кількості об’єктів алгоритм G є найкращий, а AIV – найгірший. Тим часом AIV добре працює з малими та середніми даними, але дає найгірші результати з великими даними.

5.3.2 Дослідження швидкодії

Ми будемо вивчати значення швидкості алгоритму за критерієм часу виконання завдання (критерій 1) і вимагати мінімізації кількості ЛА. У

таблиці 5.10 наведено швидкість виконання алгоритму через мінімізацію LA за критерієм 1.

Таблиця 5.10 – Значення швидкості роботи алгоритму для Критерію 1 з мінімізацією по ЛА

Задача	Кількість об'єктів	Кількість депо	ДЛП	ТП	АІВ	G-алгоритм
1	100	2	2,4	5,9	6,1	5,0
2	100	2	5,7	6,9	6,3	6,5
3	100	2	4,9	8,9	5,4	6,2
Середнє значення швидкості			4,3	7,2	5,9	5,9
4	150	3	6,9	12,3	7,1	10,9
5	150	3	7,7	29,3	11,1	11,4
6	150	3	5,0	13,7	11,5	14,2
Середнє значення швидкості			6,5	18,4	9,9	12,2
7	200	4	9,8	21,9	22,3	20,8
8	200	4	19,3	28,3	19,0	15,6
9	200	4	10,9	29,0	16,6	14,8
Середнє значення швидкості			13,3	26,4	19,3	17,1
Загальне середнє значення			8,1	17,4	11,7	11,7

Як видно з таблиці 5.10, серед загальних середніх швидкостей алгоритму DLP має найнижче значення швидкості алгоритму.

Щоб наочно показати отримані результати дослідження, побудуємо графіки залежності швидкодійних значень алгоритмів різних алгоритмів від кількості об'єктів (рис. 5.5), як критерію часу виконання завдання від вимоги мінімізації кількості ЛА.

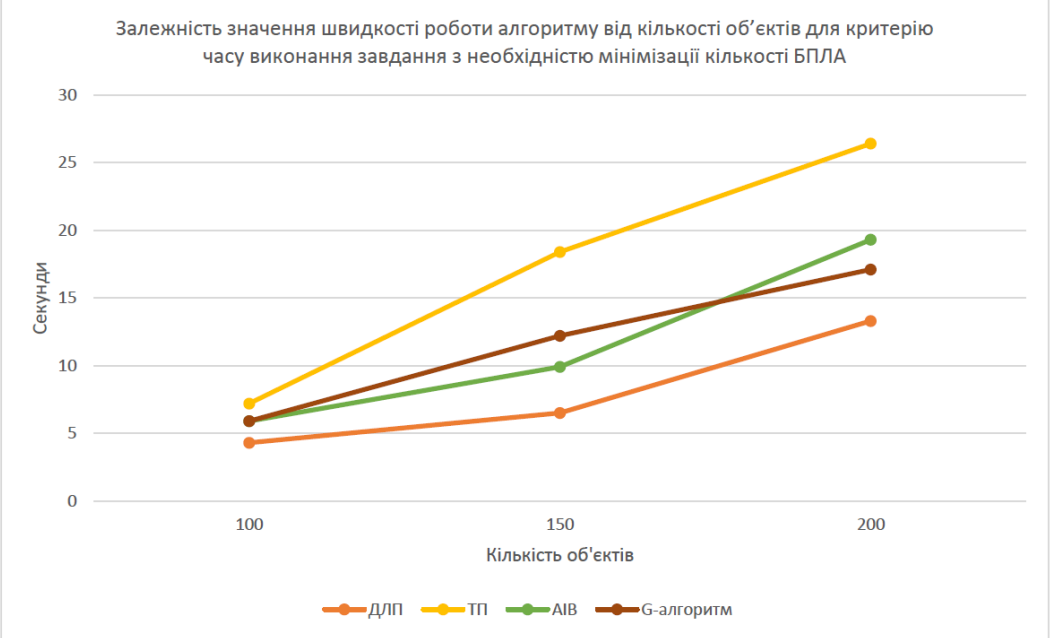


Рисунок 5.5 – Залежність значення швидкості алгоритму від кількості об’єктів для критерію часу виконання завдання та необхідності мінімізації кількості літальних апаратів.

На рисунку 5.5 зображено графік залежності CF від кількості об’єктів для різних алгоритмів. DLP позначено тут помаранчевим кольором, TP — жовтим, AIV — зеленим, а алгоритм G — коричневим. По осі абсцис відкладено кількість об’єктів, розв’язаних задачею, по осі ординат – швидкість роботи алгоритму. Для побудови графіка було розраховано середнє значення швидкості алгоритму для різної кількості об’єктів і для кожного алгоритму. З малюнка ми бачимо, що DLP має найкращі результати для будь-якої кількості об’єктів, тоді як TP має найдовший час виконання. Для великих даних AIV перевершує алгоритм G, але для середніх даних все навпаки.

Ми будемо вивчати алгоритмічні значення швидкості для критерію вартості маршруту (критерій 2), який вимагає мінімізації кількості повітряних суден. У таблиці 5.11 наведено швидкість виконання алгоритму через мінімізацію LA за критерієм 2.

Таблиця 5.11 – Значення швидкості роботи алгоритму для Критерію 2 з мінімізацією по ЛА

Задача	Кількість об'єктів	Кількість депо	ДЛП	ТП	АІВ	G-алгоритм
1	100	2	9,0	20,3	25,2	27,8
2	100	2	4,9	16,1	17,5	23,2
3	100	2	8,4	26,3	24,1	25,1
Середнє значення швидкості			7,4	20,9	22,3	25,4
4	150	3	11,9	58,4	30,8	40,2
5	150	3	18,6	48,6	42,0	39,9
6	150	3	17,2	45,9	29,9	34,6
Середнє значення швидкості			15,9	51,0	34,2	38,2
7	200	4	54,0	60,0	50,3	53,5
8	200	4	28,1	60,0	47,4	45,1
9	200	4	47,1	60,0	60,0	60,0
Середнє значення швидкості			43,1	60,0	52,6	52,9
Загальне середнє значення			22,1	44,0	36,4	38,8

Як видно з таблиці 5.11, серед загальних середніх швидкостей алгоритму DLP має найнижче значення швидкості алгоритму.

Побудуємо графік залежності алгоритмічних значень швидкості для різних алгоритмів від кількості об'єктів (рис. 5.6), як критерій вартості маршруту та необхідності мінімізації кількості літаків. Для побудови графіка було розраховано середнє значення швидкості алгоритму для різної кількості об'єктів і для кожного алгоритму.

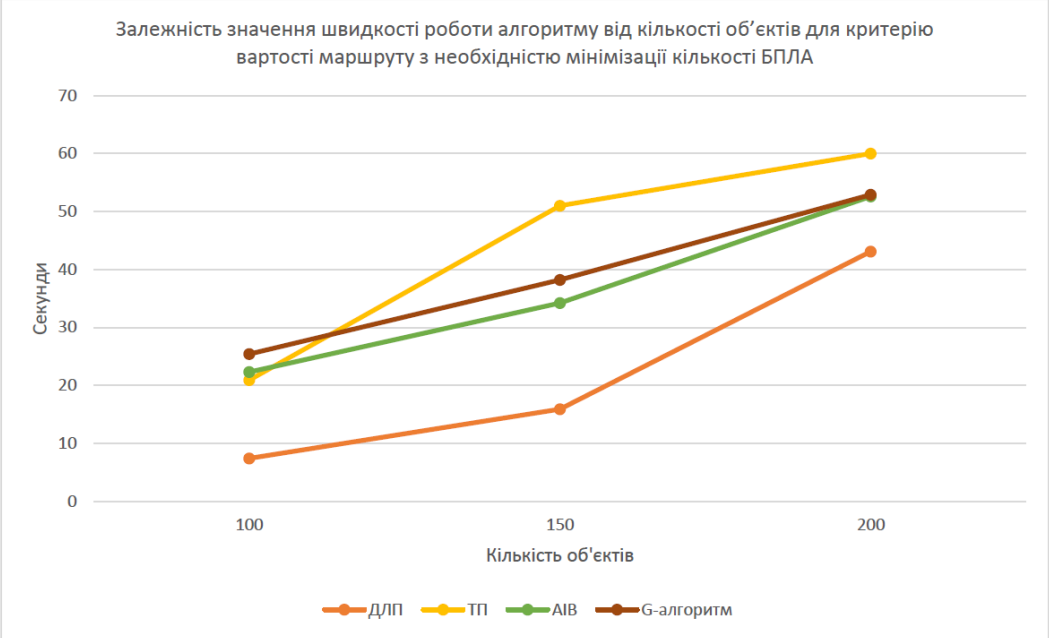


Рисунок 5.6 – Залежність значення швидкості алгоритму від кількості об’єктів, як критерій вартості маршруту кількість повітряних суден необхідно мінімізувати

На малюнку 5.6 показано швидкість алгоритму в залежності від кількості об’єктів для різних алгоритмів. DLP позначено тут помаранчевим кольором, TP — жовтим, AIV — зеленим, а алгоритм G — коричневим. По осі абсцис відкладено кількість об’єктів, розв’язаних задачею, по осі ординат – швидкість роботи алгоритму. Як видно з малюнка, DLP має найкращі результати для будь-якої кількості об'єктів. Крім того, алгоритми AIV і G приблизно однакові. Однак алгоритм G працює повільно з даними малого та середнього розміру. TP працює швидше на невеликих даних, але має найгірший ефект на середніх і великих даних.

Ми будемо вивчати значення швидкодії алгоритму за критерієм часу виконання завдання (критерій 1) без мінімізації кількості ЛА. У таблиці 5.12 наведено швидкість роботи алгоритму без мінімізації ЛА за критерієм 1.

Таблиця 5.12 – Значення швидкості роботи алгоритму для Критерію 1 без мінімізації по ЛА

Задача	Кількість об’єктів	Кількість депо	ДЛП	ТП	АІВ	G-алгоритм
1	100	2	1,1	2,5	3,0	1,8
2	100	2	2,2	2,9	2,6	2,5
3	100	2	2,3	3,2	2,5	2,5
Середнє значення швидкості			1,9	2,9	2,7	2,3
4	150	3	2,3	4,8	2,3	3,4
5	150	3	3,8	11,2	4,8	4,8
6	150	3	2,5	6,4	4,3	6,4
Середнє значення швидкості			2,8	7,2	3,8	4,9
7	200	4	4,7	8,5	9,8	7,7
8	200	4	5,8	10,2	7,9	5,1
9	200	4	4,7	9,9	7,4	4,5
Середнє значення швидкості			5,1	9,5	8,4	5,8
Загальне середнє значення			3,3	6,6	4,9	4,3

Як видно з таблиці 5.12, серед загальних середніх швидкостей алгоритму DLP має найнижче значення швидкості алгоритму.

Побудуємо графік залежності швидкості роботи алгоритму від кількості об’єктів (рис. 5.7) як критерію часу виконання завдання без мінімізації кількості літальних апаратів. Для побудови графіка було розраховано середнє значення швидкості алгоритму для різної кількості об’єктів і для кожного алгоритму.

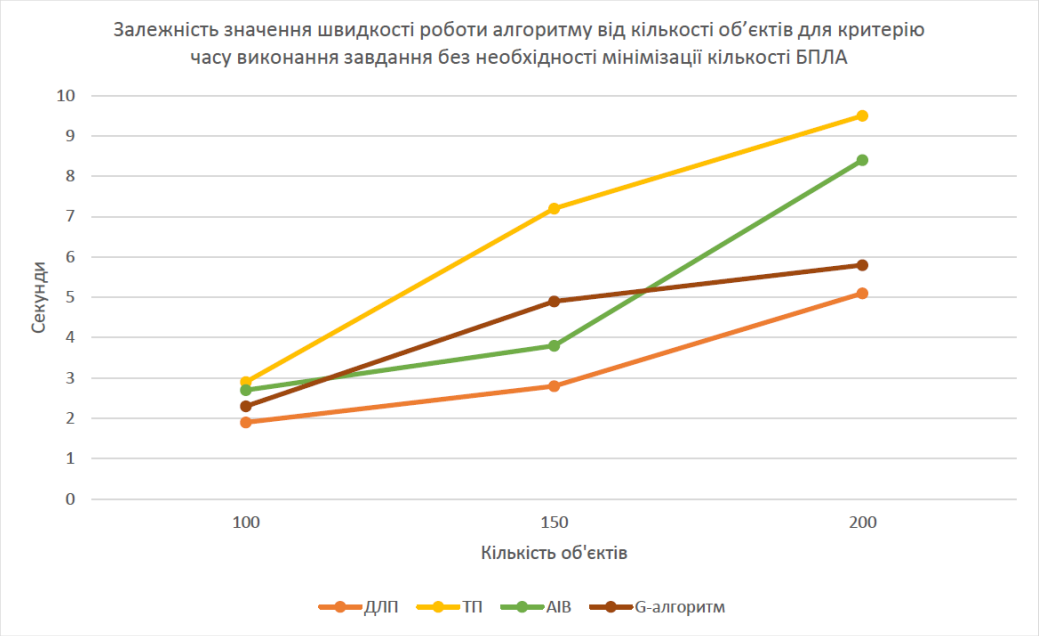


Рисунок 5.7 – Співвідношення значень швидкодії алгоритму з кількістю об'єктів для критеріїв часу виконання завдання без мінімізації кількості ЛА

На малюнку 5.7 показано швидкість алгоритму в залежності від кількості об’єктів для різних алгоритмів. DLP позначено тут помаранчевим кольором, TP — жовтим, AIV — зеленим, а алгоритм G — коричневим. По осі абсцис відкладено кількість об’єктів, розв’язаних задачею, по осі ординат – швидкість роботи алгоритму. Як видно з малюнка, DLP має найкращі результати для будь-якої кількості об'єктів. Найгірший по швидкості - ТП. Між цими двома алгоритмами знаходяться алгоритми AIV і G. У той же час ефект AIV на малих даних, особливо великих даних, гірший, ніж алгоритм G. Навпаки, алгоритм G дає швидше результати на середніх даних.

Ми будемо вивчати алгоритмічні значення швидкості для критерію вартості маршруту (критерій 2), який не потребує мінімізації кількості повітряних суден. У таблиці 5.13 наведено швидкість виконання алгоритму без мінімізації ЛА за критерієм 2.

Таблиця 5.13 – Значення швидкості роботи алгоритму для Критерію 2 без мінімізації по ЛА

Задача	Кількість об'єктів	Кількість депо	ДЛП	ТП	АІВ	G-алгоритм
1	100	2	4,2	9,7	11,3	8,8
2	100	2	2,3	7,4	6,9	9,9
3	100	2	3,6	12,4	8,5	10,9
Середнє значення швидкості			3,4	9,8	8,9	9,9
4	150	3	4,6	23,3	11,4	12,1
5	150	3	8,1	20,7	19,6	16,7
6	150	3	7,3	21,6	13,4	16,0
Середнє значення швидкості			6,3	21,9	14,8	14,9
7	200	4	23,6	27,1	19,2	20,4
8	200	4	12,6	32,4	17,8	16,8
9	200	4	21,3	29,2	26,6	32,7
Середнє значення швидкості			19,2	29,6	21,2	23,3
Загальне середнє значення			9,7	20,4	15,0	16,0

Як видно з таблиці 5.13, із загальної середньої швидкості алгоритму DLP має найнижче значення швидкості алгоритму.

Побудуємо графіки алгоритмічних значень швидкості для різних алгоритмів від кількості об'єктів (рис. 5.8) як критерій вартості маршруту без мінімізації кількості літаків. Для побудови графіка було розраховано середнє значення швидкості алгоритму для різної кількості об'єктів і для кожного алгоритму.

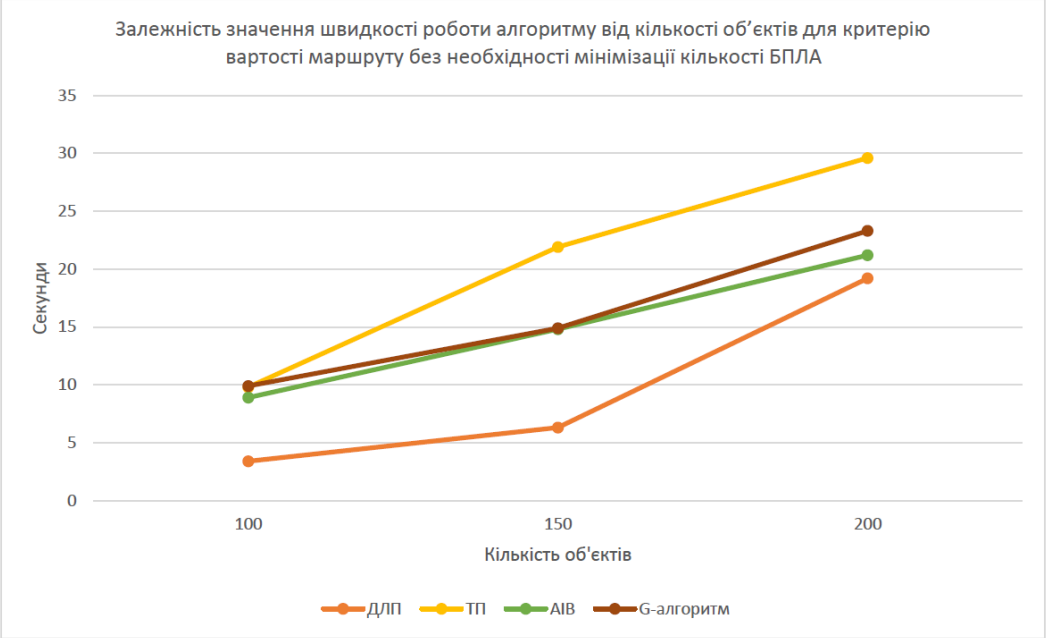


Рисунок 5.8 – Залежність алгоритмічних значень швидкості від кількості об'єктів для критеріїв вартості маршруту без мінімізації кількості повітряних суден

На малюнку 5.8 показано швидкість алгоритму в залежності від кількості об’єктів для різних алгоритмів. DLP позначено тут помаранчевим кольором, TP — жовтим, AIV — зеленим, а алгоритм G — коричневим. По осі абсцис позначається кількість об’єктів, які розв’язали задачу, по осі ординат – значення CF. З малюнка ми бачимо, що найкращі результати для будь-яких даних забезпечує DLP. AIV займає друге місце, алгоритм G займає третє місце, а TP займає останнє місце.

Розділ 5 вивчає ефективність розробленого методу, тобто: описує умови дослідження, вивчає вплив параметрів на роботу алгоритму та порівнює результати алгоритму.

Було сформовано 9 експериментальних задач: 3 задачі для 2 складів і 100 об’єктів, 3 задачі для 3 складів і 150 об’єктів, 3 задачі для 4 складів і 200 об’єктів. При порівнянні ефективності всіх алгоритмів використовували середнє значення CF і часу виконання алгоритму за 10 прогонів. Для всіх

алгоритмів максимальний час роботи обмежений 60 секундами. Використовуйте локальний пошук для вибору параметрів алгоритму. Експерименти проводились окремо за різними критеріями: час виконання завдання та вартість маршруту.

Різні алгоритми мають різні параметри. Наприклад, для DLP немає параметрів взагалі, але для алгоритмів TP, AIV і G параметри є. TP має два параметри: довжину забороненого списку та максимальну кількість ітерацій, які не покращують результати. AIV має п’ять параметрів: початкова температура, тривалість циклу, коефіцієнт зниження температури, поріг рівноваги та максимальна кількість ітерацій без покращення результатів. Алгоритм G має чотири параметри: довжину циклу, коефіцієнт k дельта-функції, поріг для досягнення рівноваги та максимальну кількість ітерацій без покращення результатів. В результаті налаштування параметрів за допомогою алгоритму локального пошуку встановлено, що значення CF зменшується. Після коригування параметрів алгоритму результати для критерію 1 покращилися в середньому на 3,8%, а для критерію 2 в середньому на 2,1%.

Продуктивність алгоритму оцінюється за двома критеріями оцінки алгоритму: якість кінцевого рішення та час роботи алгоритму. Кожен алгоритм перевіряється на відповідність цим критеріям. При дослідженні остаточних результатів було встановлено, що алгоритм G мав найнижче значення CF серед усіх критеріїв і був досить швидким. Під час дослідження швидкості було встановлено, що DLP працює найшвидше, але має найгіршу точність.

6 ЕКОНОМІЧНА ЧАСТИНА

Виконання науково-дослідної роботи завжди передбачає отримання певних результатів і вимагає відповідних витрат. Результати виконаної роботи завжди дають нам нові знання, які в подальшому можуть бути використані для удосконалення та/або розробки (побудови) нових, більш продуктивних зразків техніки, процесів та програмного забезпечення.

Дослідження на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» може бути віднесено до фундаментальних і пошукових наукових досліджень і спрямоване на вирішення наукових проблем, пов’язаних з практичним застосуванням. Основою таких досліджень є науковий ефект, який виражається в отриманні наукових результатів, які збільшують обсяг знань про природу, техніку та суспільство, які розвивають теоретичну базу в тому чи іншому науковому напрямку, що дозволяє виявити нові закономірності, які можуть використовуватися на практиці.

Для цього випадку виконаємо такі етапи робіт:

1) здійснимо проведення наукового аудиту досліджень, тобто встановлення їх наукового рівня та значимості;

2) проведемо планування витрат на проведення наукових досліджень;

3) здійснимо розрахунок рівня важливості наукового дослідження та перспективності, визначимо ефективність наукових досліджень.

6.1 Оцінювання наукового ефекту

Основними ознаками наукового ефекту науково-дослідної роботи є новизна роботи, рівень її теоретичного опрацювання, перспективність, рівень розповсюдження результатів, можливість реалізації. Науковий ефект НДР на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» можна охарактеризувати двома

показниками: ступенем наукової новизни та рівнем теоретичного опрацювання.

Значення показників ступеня новизни і рівня теоретичного опрацювання науково-дослідної роботи в балах наведені в табл. 6.1 та 6.2.

Таблиця 6.1 – Показники ступеня новизни науково-дослідної роботи виставлені експертами

Ступінь новизни	Характеристика ступеня новизни	Значення ступеня новизни, бали		
		Експерти (ПІБ, посада)		
		1		
Принципово нова	Робота якісно нова за постановкою задачі і ґрунтується на застосуванні оригінальних методів дослідження. Результати дослідження відкривають новий напрям в даній галузі науки і техніки. Отримані принципово нові факти, закономірності; розроблена нова теорія. Створено принципово новий пристрій, спосіб, метод	-		
Нова	Отримана нова інформація, яка суттєво зменшує невизначеність наявних значень (по-новому або вперше пояснені відомі факти, закономірності, впроваджені нові поняття, розкрита структура змісту). Проведено суттєве	8	4	6

	вдосконалення, доповнення і уточнення раніше досягнутих результатів			
Відносно нова	Робота має елементи новизни в постановці задачі і методах дослідження. Результати дослідження систематизують і узагальнюють наявну інформацію, визначають шляхи подальших досліджень; вперше знайдено зв'язок (або знайдено новий зв'язок) між явищами. В принципі відомі положення розповсюджені на велику кількість об'єктів, в результаті чого знайдено ефективне рішення. Розроблені більш прості способи для досягнення відомих результатів. Проведена часткова раціональна модифікація (з ознаками новизни)	-	0	
Традиційна	Робота виконана за традиційною методикою. Результати дослідження мають інформаційний характер. Підтверджені або поставлені під сумнів відомі факти та твердження, які потребують перевірки. Знайдено новий варіант рішення, який не дає суттєвих переваг в порівнянні з існуючим	-		
Не нова	Отримано результат, який раніше зафіксований в інформаційному полі, та не був відомий авторам	-		

Середнє значення балів експертів	48,0

Згідно отриманого середнього значення балів експертів ступінь новизни характеризується як нова, тобто отримана нова інформація, яка суттєво зменшує невизначеність наявних знань (по-новому або вперше пояснені відомі факти, закономірності, впроваджені нові поняття, розкрита структура змісту) та проведено суттєве вдосконалення, доповнення і уточнення раніше досягнутих результатів.

Таблиця 6.2 – Показники рівня теоретичного опрацювання науково-дослідної роботи виставлені експертами

Характеристика рівня теоретичного опрацювання	Значення показника рівня теоретичного опрацювання, бали		
	Експерт (ПІБ, посада)		
		2	3
Відкриття закону, розробка теорії	-	-	-
Глибоке опрацювання проблеми: багатоаспектний аналіз зв’язків, взаємозалежності між фактами з наявністю пояснень, наукової систематизації з побудовою евристичної моделі або комплексного прогнозу	60	-	-
Розробка способу (алгоритму, програми), пристрою, отримання нової речовини	-	58	58
Елементарний аналіз зв’язків між фактами та наявною гіпотезою, класифікація, практичні рекомендації для окремого випадку тощо	-	-	-
Опис окремих елементарних фактів, викладення	-	-	-

досвіду, результатів спостережень, вимірювань тощо			
Середнє значення балів експертів	58,7		

Згідно отриманого середнього значення балів експертів рівень теоретичного опрацювання науково-дослідної роботи характеризується як розробка способу (алгоритму, програми), пристрою, отримання нової речовини.

Показник, який характеризує рівень наукового ефекту, визначаємо за формулою [Козловський, Лесько, Кавецький]:

$$E_{нау} = 0,6 \cdot k_{нов} + 0,4 \cdot k_{теор}, \quad (6.1)$$

де $k_{нов}, k_{теор}$ - показники ступеня новизни та рівня теоретичного опрацювання науково-дослідної роботи, $k_{нов} = 48,0, k_{теор} = 58,7$ балів;

0,6 та 0,4 – питома вага (значимість) показників ступеня новизни та рівня теоретичного опрацювання науково-дослідної роботи.

$$E_{нау} = 0,6 \cdot k_{нов} + 0,4 \cdot k_{теор} = 0,6 \cdot 48,0 + 0,4 \cdot 58,67 = 52,27 \text{ балів.}$$

Визначення характеристики показника $E_{нау}$ проводиться на основі висновків експертів виходячи з граничних значень, які наведені в табл. 4.3.

Таблиця 4.3 – Граничні значення показника наукового ефекту

Досягнутий рівень показника	Кількість балів
Високий	70…100
Середній	50…69
Достатній	15…49
Низький (помилкові дослідження)	1…14

Відповідно до визначеного рівня наукового ефекту проведеної науково-дослідної роботи на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення», даний рівень становить 52,27 балів і відповідає статусу - середній рівень. Тобто у даному випадку можна вести мову про потенційну фактичну ефективність науково-дослідної роботи.

6.2 Розрахунок витрат на здійснення науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

6.2.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою **[Козловський, Лесько, Кавецький]**:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (6.2)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=22 дні.

$$З_o = 18200{,}00 \cdot 48 / 22 = 39709{,}09 \; грн.$$

Проведені розрахунки зведемо до таблиці.

Таблиця 6.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник науково-дослідної роботи	18200,00	827,27	48	39709,09
Науковий співробітник	17500,00	795,45	48	38181,82
Консультант (оператор БПЛА)	18000,00	818,18	5	4090,91
Інженер-розробник ПЗ	18000,00	818,18	44	36000,00
Технік	8110,00	368,64	22	8110,00
Всього				126091,82

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» розраховуємо за формулою:

$$З_p = \sum_{i=1}^{n} C_i \cdot t_i , \qquad (6.3)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}} , \qquad (6.4)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо *Мм*=6700,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) **[Козловський, Лесько, Кавецький]**;

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об’єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

T_p – середнє число робочих днів в місяці, приблизно T_p = 22 дн;

$t_{зм}$ – тривалість зміни, год.

C_1 = 6700,00 · 1,10 · 1,35 / (22 · 8) = 56,53 *грн.*

$З_{p1}$ = 56,53 · 8,00 = 452,25 *грн.*

Таблиця 6.5 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи,	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна	Величина оплати на робітника

	год			ставка, грн	грн
Установка обладнання для проведення пошукових робіт	8,00	2	1,10	56,53	452,25
Підготовка робочого місця наукового співробітника	5,00	3	1,35	69,38	346,90
Підготовка робочого місця інженера-розробника програмних засобів	4,50	4	1,50	77,09	346,90
Інсталяція програмного забезпечення для моделювання та розробки	6,00	4	1,50	77,09	462,53
Всього					1608,57

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (6.5)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод}$ = (126091,82 + 1608,57) · 11 / 100% = 14047,04 грн.

6.2.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (6.6)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (126091,82 + 1608,57 + 14047,04) · 22 / 100% = 31184,44 грн.

6.2.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення».

Витрати на матеріали на даному етапі проведення досліджень в основному пов’язані з використанням моделей елементів та моделювання роботи і досліджень за допомогою комп’ютерної техніки та створення експериментальних математичних моделей або програмного забезпечення, тому дані витрати формуються на основі витратних матеріалів характерних для офісних робіт.

Витрати на матеріали ($М$), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}\,, \quad (6.7)$$

де $Н_j$ – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

K_j – коефіцієнт транспортних витрат, (K_j = *1,1 … 1,15*);

B_j – маса відходів *j*-го найменування, кг;

$Ц_{вj}$ – вартість відходів *j*-го найменування, грн/кг.

$M_1 = 2 \cdot 186{,}00 \cdot 1{,}1 - 0 \cdot 0 = 409{,}20$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 6.6 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Офісний папір VERDE Ultra	186,00	2	0	0	409,20
Папір для записів VERDE Light A5	99,00	3	0	0	326,70
Органайзер офісний VERDE OFFICE	172,00	2	0	0	378,40
Канцелярське приладдя (набір офісного працівника)	202,00	4	0	0	888,80
Картридж для принтера Canon LBP6000	1208,00	1	0	0	1328,80
Диск	25,0	3	0	0	82,50

оптичний NewCODE CD-R	0				
Flesh-пам'ять Kingston 32 GB	239,00	1	0	0	262,90
Тека для паперів CARBONIX BOX-ZX	114,00	2	0	0	250,80
Всього					3928,10

6.2.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення», розраховуємо, згідно з їхньою номенклатурою, за формулою:

$$К_в = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j \qquad (4.8)$$

де $Н_j$ – кількість комплектуючих *j*-го виду, шт.;

$Ц_j$ – покупна ціна комплектуючих *j*-го виду, грн;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ *= 1,1 ... 1,15*).

$К_в = 2 \cdot 2799,00 \cdot 1,05 = 5877,90$ грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 4.7 – Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Сенсори для вимірювання погодних умов.	2	2799,00	5877,90
GPS-пристрої та системи навігації	2	6899,00	14487,90
Всього			20365,80

6.2.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \ , \qquad (6.9)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань устаткування.

$B_{спец} = 26057,00 \cdot 1 \cdot 1,05 = 27359,85$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 6.8 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Сервер ARTLINE Business R17 v14	1	26057,00	27359,85
Мережеве сховище Synology 4BAY DS423+ (DS423+)	1	21780,00	22869,00
Всього			50228,85

6.2.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (6.10)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань програмних засобів.

$B_{прг} = 12800{,}00 \cdot 1 \cdot 1{,}01 = 12928{,}00$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 6.9 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Спеціалізоване програмне забезпечення для планування маршрутів та оптимізації	1	12800,00	12928,00
Всього			12928,00

6.2.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (6.11)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (22820{,}00 \cdot 3) / (3 \cdot 12) = 1901{,}67$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 6.10 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Персональний комп'ютер HP ProBook 4540s	22820,00	3	3	1901,67
Обчислювально-графічна система програмної розробки	42888,00	3	3	3574,00
Робоче місце розробника програмного забезпечення	8599,00	5	3	429,95
Пристрій виводу текстової інформації	6500,00	4	3	406,25
Оргтехніка	8925,00	4	3	557,81
Приміщення лабораторії	520000,00	25	3	5200,00

Всього	12069,68

6.2.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (6.12)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 7,50 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,20 \cdot 360,0 \cdot 7,50 \cdot 0,95 / 0,97 = 540,00$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 6.11 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Персональний комп'ютер HP ProBook 4540s	0,20	360,0	540,00
Обчислювально-графічна система програмної розробки	0,40	360,0	1080,00
Робоче місце розробника програмного	0,07	360,0	189,00

забезпечення			
Пристрій виводу текстової інформації	0,22	3,2	5,28
Оргтехніка	0,45	1,6	5,40
Сервер ARTLINE Business R17 v14	0,25	100,0	187,50
Мережеве сховище Synology 4BAY DS423+ (DS423+)	0,10	100,0	75,00
Всього			2082,18

6.2.9 Службові відрядження

До статті «Службові відрядження» дослідної роботи на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов’язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з’їзди, конференції, наради, пов’язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{H_{св}}{100\%}, \quad (6.13)$$

де $H_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 20%.

$B_{св}$ = (126091,82 + 1608,57) · 20 / 100% = 25540,08 грн.

6.2.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (6.14)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$B_{сп}$ = (126091,82 + 1608,57) · 30 / 100% = 38310,12 грн.

6.2.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$І_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (6.15)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$І_в$ = (126091,82 + 1608,57) · 50 / 100% = 63850,19 грн.

6.2.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (6.16)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$B_{нзв}$ = (126091,82 + 1608,57) · 100 / 100% = 127700,39 грн.

Витрати на проведення науково-дослідної роботи на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + B_{спец} + B_{прг} + A_{обл} + B_е + B_{св} + B_{сп} + I_в + B_{нзв}. \quad (6.17)$$

$B_{заг}$= 126091,82 +1608,57 +14047,04 +31184,44 +3928,10 +20365,80 +50228,85 + 12928,00 + 12069,68 +2082,18 +25540,08 +38310,12 +63850,19 +127700,39 = 529935,25 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (6.18)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,95.

ЗВ = 529935,25 / 0,95 = 557826,58 грн.

6.3 Оцінювання важливості та наукової значимості науково-дослідної роботи

Оцінювання та доведення ефективності виконання науково-дослідної роботи фундаментального чи пошукового характеру є достатньо складним процесом і часто базується на експертних оцінках, тому має вірогідний характер.

Для обґрунтування доцільності виконання науково-дослідної роботи на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» використовується спеціальний комплексний показник, що враховує важливість, результативність роботи, можливість впровадження її результатів у виробництво, величину витрат на роботу.

Комплексний показник K_P рівня науково-дослідної роботи може бути розрахований за формулою:

$$K_P = \frac{I^n \cdot T_C \cdot R}{B \cdot t}, \qquad (6.19)$$

де I – коефіцієнт важливості роботи. Приймемо $I = 4$;

n – коефіцієнт використання результатів роботи; $n = 0$, коли результати роботи не будуть використовуватись; $n = 1$, коли результати роботи будуть використовуватись частково; $n = 2$, коли результати роботи будуть використовуватись в дослідно-конструкторських розробках; $n = 3$, коли результати можуть використовуватись навіть без проведення дослідно-конструкторських розробок. Приймемо $n=3$;

T_C – коефіцієнт складності роботи. Приймемо $T_C = 3$;

R – коефіцієнт результативності роботи; якщо результати роботи плануються вище відомих, то $R = 4$; якщо результати роботи відповідають відомому рівню, то $R = 3$; якщо нижче відомих результатів, то $R = 1$. Приймемо $R = 4$;

B – вартість науково-дослідної роботи, тис. грн. Приймемо B =557826,58 грн;

t – час проведення дослідження. Приймемо t = 0,17 років, (2 міс.).

Визначення показників I, n, T_C, R, B, t здійснюється експертним шляхом або на основі нормативів [Козловський, Лесько, Кавецький].

$$K_P = \frac{I^n \cdot T_C \cdot R}{B \cdot t} = 4^3 \cdot 3 \cdot 4 / 557{,}8 \cdot 0{,}17 = 8{,}26.$$

Якщо $K_P > 1$, то науково-дослідну роботу на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» можна вважати ефективною з високим науковим, технічним і економічним рівнем.

6.4 Висновок до розділу

Витрати на проведення науково-дослідної роботи на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» складають 557826,58 грн. Відповідно до проведеного аналізу та розрахунків рівень наукового ефекту проведеної науково-дослідної роботи на тему «Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення» є середній, а дослідження актуальними, рівень доцільності виконання науково-дослідної роботи $K_P > 1$, що свідчить про потенційну ефективність з високим науковим, технічним і економічним рівнем.

Література: Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

ВИСНОВКИ

Джерела інформації узагальнено та класифіковано. Розглянуто та класифіковано найпоширеніші категорії задач маршрутизації ПС. Запропоновано математичну формулу задачі про маршрут літака при наявності кількох ангарів. Два CF були розроблені шляхом визначення двох критеріїв для двох типів проблем та їх рішень.

Перевірено метод розв’язання та розроблено новий алгоритм розв’язання задачі маршрутизації літака з кількома ангарами, а саме:

- Формування початкових наближень: методу найдешевшого включення та проміжного вдосконаленого гібридного алгоритму найдешевшого включення;

— Покращено початкове наближення: алгоритми DLP, TP, AIV і O.

Розроблено програмне забезпечення, яке вирішує проблему дослідження.

Налаштовано параметри алгоритму та досліджено ефективність розробленого алгоритму.

Продуктивність алгоритму оцінюється за двома критеріями оцінки: якість кінцевого рішення та час роботи алгоритму. При дослідженні кінцевих результатів було встановлено, що G-алгоритм може знайти мінімальне значення CF для всіх критеріїв і є досить швидким. Під час дослідження швидкості було встановлено, що DLP працює найшвидше, але має найгіршу точність.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Гуляницький Л. Ф., Коткова А. А. До класифікації задач маршрутизації транспортних засобів. Науковий вісник Ужгородського університету. 2020. № 1 (36). С. 73–84.

2. Dantzig G. B., Ramser J. H. The truck dispatching problem. Management Science. 1959. Vol. 6, No 1. P. 80–91.

3. Radzki G., Thibbotuwawa, A., Bocewicz, G. UAVs flight routes optimization in changing weather conditions – Constraint programming approach. Applied Computer Science. 2019. Vol. 15, No 3. P. 5–17.

4. Гуляницький Л. Ф., Мулеса О. Ю. Прикладні методи комбінаторної оптимізації: навч. посіб. Київ : Видавничо-поліграфічний центр «Київський університет», 2016. 146 с.

5. Uchoa E., Pecin D., Pessoa A. New benchmark instances for the Capacitated Vehicle Routing Problem. European Journal of Operational Research. 2017. Vol. 257, No 3. P. 845–858.

6. Avci M., Topaloglu S. An adaptive local search algorithm for vehicle routing problem with simultaneous and mixed pickups and deliveries. Computers & Industrial Engineering. 2015. Vol. 83. P. 15–29.

7. Yao B., Yu B., Hu P. An improved particle swarm optimization for carton heterogeneous vehicle routing problem with a collection depot. Annals of Operations Research. 2016. Vol. 242, No 2. P. 303–320.

8. Berghida M., Boukra A. Quantum Inspired Algorithm for a VRP with Heterogeneous Fleet Mixed Backhauls and Time Windows. International Journal of Applied Metaheuristic Computing. 2016. Vol. 7, No 4. P. 18–38.

9. Koc C., Laporte G. Vehicle routing with backhauls: Review and research perspectives. Computers & Operations Research. 2018. Vol. 91. P. 79–91.

10. Reil S., Bortfeldt A., Monch L. Heuristics for vehicle routing problems with backhauls, time windows, and 3D loading constraints. European Journal of Operational Research. 2018. Vol. 266, No 3. P. 877–894.

11. Bruglieri M., Pezzella F., Pisacane O. A Variable Neighborhood Search Branching for the Electric Vehicle Routing Problem with Time Windows. Electronic Notes in Discrete Mathematics. 2015. Vol. 47. P. 221–228.

12. Kucukoglu I., Ozturk N. An advanced hybrid meta-heuristic algorithm for the vehicle routing problem with backhauls and time windows. Computers & Industrial Engineering. 2015. Vol. 86. P. 60–68.

13. Bin Y., Zhi-Hua H. Routing with time-windows for multiple environmental vehicle types. Computers & Industrial Engineering. 2015. Vol. 89. P. 150–161.

14. Cantu-Funes R., Angelica Salazar-Aguilar M., Boyer V. Multi-depot periodic vehicle routing problem with due dates and time windows. Journal of the Operational Research Society. 2016. Vol. 69. P. 296–306.

15. Montoya-Torres J., Franco J., Isaza S. A literature review on the vehicle routing problem with multiple depots. Computers & Industrial Engineering. 2015. Vol. 79. P. 115– 129.

16. De Oliveira F., Enayatifar R., Sadaei H. A cooperative coevolutionary algorithm for the Multi-Depot Vehicle Routing Problem. Expert Systems with Applications. 2016. Vol. 43. P. 117–130.

17. Lalla-Ruiz E., Exposito-Izguierdo Ch., Taheripour Sh. An improved formulation for the multi-depot open vehicle routing problem. OR Spectrum. 2016. Vol. 38, No 1. P. 175–187.

18. Leungsubthawee K., Saranwong S., Likasiri Ch. Multiple Depot Vehicle Routing Problems on Clustering Algorithms. Thai Journal of Mathematics. 2018. P. 205– 216.

19. Ramos T., Gomes M., Povoa A. Multi-depot vehicle routing problem: a comparative study of alternative formulations. International Journal of Logistics Research and Applications. 2019. P. 1–18.

20. Chávez J., Escobar J. A Multi-objective Pareto and Colony Algorithm for the Multi-depot Vehicle Routing Problem with Backhauls. International Journal of Industrial Engineering Computations. 2016. Vol. 7. P. 35–48.

21. Zhang Y., Qi M., Mia L., Wu G. A Generalized Multi-Depot Vehicle Routing Problem with Replenishment Based on LocalSolver. International Journal of Industrial Engineering Computations. 2015. Vol. 6. P. 81–98.

22. Braekers K., Ramaekers K., Nieuwenh I. The Vehicle Routing Problem: State of the Art Classification and Review. Computers & Industrial Engineering. 2016. Vol. 99. P. 300–313.

23. Markov I., Varone S., Bierlaire M. Integrating a heterogeneous fixed fleet and a flexible assignment of destination depots in the waste collection VRP with intermediate facilities. Transportation Research Part B: Methodological. 2016. Vol. 84. P. 256–273.

24. Ritzinger U., Puchinger J. A survey on dynamic and stochastic vehicle routing problems. International Journal of Production Research. 2015. Vol. 54. P. 215–231.

25. Sarasola B., Doerner K., Schmid V. Variable neighborhood search for the stochastic and dynamic vehicle routing problem. Annals of Operations Research. 2016. Vol. 236, No 2. P. 425–461.

26. Hernandez F., Gendreau M. Potvin J. Heuristics for tactical time slot management: a periodic vehicle routing problem view. International Transactions in Operational Research. 2017. Vol. 24, No 6. P. 1233–1252.

27. Lee J., Kim B. Industrial ship routing problem with split delivery and two types of vessels. Expert Systems with Applications. 2015. Vol. 42, No 22. P. 9012–9023.

28. Chu J., Yan S., Huang H. A Multi-Trip Split-Delivery Vehicle Routing Problem with Time Windows for Inventory Replenishment Under Stochastic Travel Times. Networks and Spatial Economics. 2017. Vol. 17, No 1. P. 41–68.

29. Горбулін В. П., Гуляницький Л. Ф., Сергієнко І. В. Постановки та математичні моделі проблем оптимізації маршрутів літальних апаратів із динамічними депо. Управляючі системи й машини. 2019. № 1. С. 3–10.

30. Гуляницький Л. Ф., Сторчевий В. В. Одна спеціальна задача маршрутизації ЛА. Науковий вісник Ужгородського університету. Серія математика і інформатика. 2019. № 34 (1). С. 60–78.

31. Міночкін А. І., Сова О. Я. Аналіз використання безпілотних літальних апаратів у якості ретрансляторів тактичних мобільних радіомереж. Збірник наукових праць ВІТІ. 2017. № 1. С. 61–70.

32. Yu K., Budhiraja A. Algorithms and experiments on routing of unmanned aerial vehicles with mobile recharging stations. Field robotics. 2019. Vol. 36, No 3. P. 602– 616.

33. Dae-Sung J., Hyeok-Joo Ch., Han-Lim Ch. Optimal control-based UAV path planning with dynamically-constrained TSP with neighborhoods. 17th International Conference on Control, Automation and Systems (ICCAS), Jeju, Korea (South), 18- 21 October 2017. Jeju, 2017. P. 373–378.

34. Schwarzrock J., Zacarias I. Solving task allocation problem in multi Unmanned Aerial Vehicles systems using Swarm intelligence. Engineering Applications of Artificial Intelligence. 2019. Vol. 72. P. 10–20.

35. Yilmaz B., Sueda N. Multi UAV Based Traffic Control in Smart Cities. 1th International Conference on Computing, Communication and Networking Technologies (ICCCNT). IEEE, 1-3 July 2020. P. 1–7.

36. Zhen l. A vehicle routing problem arising in unmanned aerial monitoring. Computers & Operations Research. 2019. Vol. 105. P. 1–11.

37. Rojas Viloria D., Solano-Charnis E. Unmanned aerial vehicles/drones in vehicle routing problems: a literature review. International Transactions in Operational Research. 2021. Vol. 28, No 4. P. 1626–1657.

38. Khoufi I., Laouiti A. A survey of recent extended variants of the traveling salesman and vehicle routing problems for unmanned aerial vehicles. Drones. 2019. Vol. 3, No 3. P. 66.

39. Sacramento D., Pisinger D. An adaptive large neighborhood search metaheuristic for the vehicle routing problem with drones. Transportation Research Part C: Emerging Technologies. 2019. Vol. 102. P. 289–315.

40. Alotaibi K., Rosenbarger J. Unmanned aerial vehicle routing in the presence of threats. Computers & Industrial Engineering. 2018. Vol. 115. P. 190–205.

41. Wang Z., Sheu J. Vehicle routing problem with drones. Transportation research part B: methodological. 2019. Vol. 122. P. 350–364.

42. Li M., Zhen L. Unmanned aerial vehicle scheduling problem for traffic monitoring. Computers & Industrial Engineering. 2018. Vol. 122. P. 15–23.

43. Song M., Li J. Metaheuristics for solving the vehicle routing problem with the time windows and energy consumption in cold chain logistics. Applied Soft Computing. 2020. Vol. 95.

44. Semiz F., Polat F. Solving the area coverage problem with UAVs: A vehicle routing with time windows variation. Robotics and Autonomous Systems. 2020. Vol. 126.

45. Jia Z., Yu J. Cooperative multiple task assignment problem with stochastic velocities and time windows for heterogeneous unmanned aerial vehicles using a genetic algorithm. Aerospace Science and Technology. 2018. Vol. 76. P. 112–125.

46. Coutinho W., Battarra M. The unmanned aerial vehicle routing and trajectory optimization problem, a taxonomic review. Computers & Industrial Engineering. 2018. Vol. 120. P. 116–128.

47. Li H., Wang H. Two-echelon vehicle routing problem with time windows and mobile satellites. Transportation Research Part B: Methodological. 2020. Vol. 138. P. 179– 201.

48. Thibbotuwawa A., Nielsen P. Energy consumption in unmanned aerial vehicles: A review of energy consumption models and their relation to the UAV routing. In International Conference on Information Systems Architecture and Technology, Springer, Cham, September 2018. P. 173–184.

49. Вовкодав О. В., Ліп’яніна Х. В. Сучасні інформаційні технології : навч. посіб. Тернопіль : ТНЕУ, 2017. 500 с.

50. A sophisticated text editor for code, markup and prose. URL: http://www.sublimetext.com/

51. Анісімов А. В., Дорошенко А. Ю. Програмування числових методів мовою Python : навч. посіб. Київ : Видавничо-поліграфічний центр «Київський університет», 2014. 640 с.

52. Півняк Г. Г., Бусигін Б. С. Дівізінюк М. М. Тлумачний словник з інформатики. Дніпропетровськ : Національний гірничий університет, 2010. 600 с.

53. Кириченко А. В., Хрусталев А. А HTML5+CSS3. Основы современного web-дизайна. СПб. : Наука и Техника, 2018. 352 с.

54. Демків Т. М., Демків Л. С. Пакети Python для моделювання фізичних процесів. FLOSS Lviv 2015 : матеріали V міжнародної науково-практичної конференції, м. Львів, 23-26 квітня 2015 р. Львів, 2015. С. 21–22.

50. A sophisticated text editor for code, markup and prose. URL: http://www.sublimetext.com/

51. Анісімов А. В., Дорошенко А. Ю. Програмування числових методів мовою Python : навч. посіб. Київ : Видавничо-поліграфічний центр «Київський університет», 2014. 640 с.

52. Півняк Г. Г., Бусигін Б. С. Дівізінюк М. М. Тлумачний словник з інформатики. Дніпропетровськ : Національний гірничий університет, 2010. 600 с.

53. Кириченко А. В., Хрусталев А. А HTML5+CSS3. Основы современного web-дизайна. СПб. : Наука и Техника, 2018. 352 с.

54. Демків Т. М., Демків Л. С. Пакети Python для моделювання фізичних процесів. FLOSS Lviv 2015 : матеріали V міжнародної науково-практичної конференції, м. Львів, 23-26 квітня 2015 р. Львів, 2015. С. 21–22.

ДОДАТКИ

ДОДАТОК А

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: **«Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення»**

Тип роботи: Магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ КСУ, ФІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 87,6% Схожість 12,4%

Аналіз звіту подібності (відмітити потрібне)

v Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на розгляд експертної комісії кафедри.

☐ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Володимир ДУБОВОЙ
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unichek щодо роботи.

Автор роботи _____ Юрій САВЧЕНКО
(підпис) (прізвище, ініціали)

Керівник роботи _____ Олена НИКИТЕНКО
(підпис) (прізвище, ініціали)

ДОДАТОК Б

ВНТУ

ЗАТВЕРДЖЕНО
Т.в.о. Зав. кафедри КСУ ВНТУ,
д.т.н., доцент
_____ Марія ЮХИМЧУК
"06" 10 2023 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

«Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення »

08-33.МКР.13.00.000 ТЗ
номер

Студент групи 2АКІТ-22м

Підпис _____ Юрій САВЧЕНКО
Ім'я ПРІЗВИЩЕ

Керівник к.т.н., доцент кафедри КСУ

Підпис _____ Олена НИКИТЕНКО
Ім'я ПРІЗВИЩЕ

Вінниця 2023

1. Назва та галузь застосування

1.1. Назва – Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення

1.2. Галузь застосування – авіаційна промисловість

2. Підстава для проведення розробки.

Тема магістерської кваліфікаційної роботи затверджена наказом по ВНТУ від “18” вересня 2023 року №247

3. Мета та призначення розробки.

Метою магістерської кваліфікаційної роботи є мінімізація часу огляду або витрат на маршрут, що є результатом огляду та/або обслуговування заданого набору цілей на землі за допомогою БПЛА, які можуть бути розташовані на кількох ділянках, але з певними додатковими обмеженнями.

4. Джерела розробки.

Магістерська кваліфікаційна робота виконується вперше. В ході проведення розробки повинні використовуватись такі документи:

.1 Гуляницький Л. Ф., Коткова А. А. До класифікації задач маршрутизації транспортних засобів. Науковий вісник Ужгородського університету. 2020. № 1 (36). С. 73–84.

2. Dantzig G. B., Ramser J. H. The truck dispatching problem. Management Science. 1959. Vol. 6, No 1. P. 80–91.

3. Radzki G., Thibbotuwawa, A., Bocewicz, G. UAVs flight routes optimization in changing weather conditions – Constraint programming approach. Applied Computer Science. 2019. Vol. 15, No 3. P. 5–17.

5. Вимоги до розробки.

5.1. Перелік головних функцій:

- Аналіз поточного стану завдань маршрутизації транспортних засобів та завдань маршрутизації БПЛА;
- Огляд доступних методів вирішення проблем маршрутизації транспортних засобів при наявності кількох стоянок:;
- Розробка алгоритмів для вирішення задачі маршрутизації дронів при наявності кількох складів;
-Проведення дослідження ефективності розробленого алгоритму;

5.2. Основні технічні вимоги до розробки.

5.2.1. Вимоги до програмної платформи:

– WINDOWS 10;
- Використання відкритих бібліотек та фреймворків для швидкого розроблення та інтеграції.

5.2.2. Умови експлуатації системи:

- робота на мобільних та веб-додатках;
- можливість цілодобового функціонування системи;
- дані оновлюються і є актуальними.

6. Стадії та етапи розробки.

6.1 Пояснювальна записка:

1. Аналіз методів, принципів, підходів і засобів реалізації задачі автоматизації процесами в об’єкті управління відповідно до теми кваліфікаційної роботи. Постановка задач дослідження «03»_09__ 2023 р.
2. Удосконалення технології прийняття рішень при автоматизації об’єкту управління «_5»___09__ 2023 р.
3. Визначення технічних характеристик системи «_5»___09__ 2023р.
4. Розробка програмного забезпечення системи «22»_10 2023р.

6.2 Графічні матеріали:

1. Розробка UML-діаграм системи «12»_11 2023р.
2. Розробка моделі бази даних системи «20»__11__ 2023р.

3. Тестування програмного забезпечення «1»___11__ 2023 р.

7. Порядок контролю і приймання.

7.1. Хід виконання роботи контролюється керівником роботи. Рубіжний контроль провести до «11»__12_ 2023 р.

7.2. Атестація МКР здійснюється на попередньому захисті. Попередній захист магістерської кваліфікаційної роботи провести до «11»__12_ 2023 р.

7.3. Підсумкове рішення щодо оцінки якості виконання роботи приймається на засіданні ЕК. Захист магістерської кваліфікаційної роботи провести до «20»__12_ 2023 р.

ДОДАТОК В

Елементи програмного коду

Створення даних

```
def create_data_model():
  """Stores the data for the problem."""
  data = {}
  data["distance_matrix"] = [
    # fmt: off
    [0, 548, 776, 696, 582, 274, 502, 194, 308, 194, 536, 502, 388, 354, 468, 776, 662],
    [548, 0, 684, 308, 194, 502, 730, 354, 696, 742, 1084, 594, 480, 674, 1016, 868, 1210],
    [776, 684, 0, 992, 878, 502, 274, 810, 468, 742, 400, 1278, 1164, 1130, 788, 1552, 754],
    [696, 308, 992, 0, 114, 650, 878, 502, 844, 890, 1232, 514, 628, 822, 1164, 560, 1358],
    [582, 194, 878, 114, 0, 536, 764, 388, 730, 776, 1118, 400, 514, 708, 1050, 674, 1244],
    [274, 502, 502, 650, 536, 0, 228, 308, 194, 240, 582, 776, 662, 628, 514, 1050, 708],
    [502, 730, 274, 878, 764, 228, 0, 536, 194, 468, 354, 1004, 890, 856, 514, 1278, 480],
    [194, 354, 810, 502, 388, 308, 536, 0, 342, 388, 730, 468, 354, 320, 662, 742, 856],
    [308, 696, 468, 844, 730, 194, 194, 342, 0, 274, 388, 810, 696, 662, 320, 1084, 514],
    [194, 742, 742, 890, 776, 240, 468, 388, 274, 0, 342, 536, 422, 388, 274, 810, 468],
    [536, 1084, 400, 1232, 1118, 582, 354, 730, 388, 342, 0, 878, 764, 730, 388, 1152, 354],
    [502, 594, 1278, 514, 400, 776, 1004, 468, 810, 536, 878, 0, 114, 308, 650, 274, 844],
    [388, 480, 1164, 628, 514, 662, 890, 354, 696, 422, 764, 114, 0, 194, 536, 388, 730],
    [354, 674, 1130, 822, 708, 628, 856, 320, 662, 388, 730, 308, 194, 0, 342, 422, 536],
    [468, 1016, 788, 1164, 1050, 514, 514, 662, 320, 274, 388, 650, 536, 342, 0, 764, 194],
    [776, 868, 1552, 560, 674, 1050, 1278, 742, 1084, 810, 1152, 274, 388, 422, 764, 0, 798],
    [662, 1210, 754, 1358, 1244, 708, 480, 856, 514, 468, 354, 844, 730, 536, 194, 798, 0],
    # fmt: on
  ]
  data["num_vehicles"] = 4
  data["depot"] = 0
  return data
```

Визначаэмо координати

```
[(456, 320), # location 0 - the depot
(228, 0),    # location 1
(912, 0),    # location 2
(0, 80),     # location 3
(114, 80),   # location 4
(570, 160),  # location 5
```

```
(798, 160),  # location 6
(342, 240),  # location 7
(684, 240),  # location 8
(570, 400),  # location 9
(912, 400),  # location 10
(114, 480),  # location 11
(228, 480),  # location 12
(342, 560),  # location 13
(684, 560),  # location 14
(0, 640),    # location 15
(798, 640)]  # location 16

def distance_callback(from_index, to_index):
  """Returns the distance between the two nodes."""
  # Convert from routing variable Index to distance matrix NodeIndex.
  from_node = manager.IndexToNode(from_index)
  to_node = manager.IndexToNode(to_index)
  return data["distance_matrix"][from_node][to_node]

transit_callback_index = routing.RegisterTransitCallback(distance_callback)
routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)
dimension_name = "Distance"
routing.AddDimension(
  transit_callback_index,
  0,  # no slack
  3000,  # vehicle maximum travel distance
  True,  # start cumul to zero
  dimension_name,
)
distance_dimension = routing.GetDimensionOrDie(dimension_name)
distance_dimension.SetGlobalSpanCostCoefficient(100)
def create_data_model():
  """Stores the data for the problem."""
  data = {}
  data["distance_matrix"] = [
    # fmt: off
   [0, 548, 776, 696, 582, 274, 502, 194, 308, 194, 536, 502, 388, 354, 468, 776, 662],
   [548, 0, 684, 308, 194, 502, 730, 354, 696, 742, 1084, 594, 480, 674, 1016, 868, 1210],
```

```
    [776, 684, 0, 992, 878, 502, 274, 810, 468, 742, 400, 1278, 1164, 1130, 788, 1552, 754],
    [696, 308, 992, 0, 114, 650, 878, 502, 844, 890, 1232, 514, 628, 822, 1164, 560, 1358],
    [582, 194, 878, 114, 0, 536, 764, 388, 730, 776, 1118, 400, 514, 708, 1050, 674, 1244],
    [274, 502, 502, 650, 536, 0, 228, 308, 194, 240, 582, 776, 662, 628, 514, 1050, 708],
    [502, 730, 274, 878, 764, 228, 0, 536, 194, 468, 354, 1004, 890, 856, 514, 1278, 480],
    [194, 354, 810, 502, 388, 308, 536, 0, 342, 388, 730, 468, 354, 320, 662, 742, 856],
    [308, 696, 468, 844, 730, 194, 194, 342, 0, 274, 388, 810, 696, 662, 320, 1084, 514],
    [194, 742, 742, 890, 776, 240, 468, 388, 274, 0, 342, 536, 422, 388, 274, 810, 468],
    [536, 1084, 400, 1232, 1118, 582, 354, 730, 388, 342, 0, 878, 764, 730, 388, 1152, 354],
    [502, 594, 1278, 514, 400, 776, 1004, 468, 810, 536, 878, 0, 114, 308, 650, 274, 844],
    [388, 480, 1164, 628, 514, 662, 890, 354, 696, 422, 764, 114, 0, 194, 536, 388, 730],
    [354, 674, 1130, 822, 708, 628, 856, 320, 662, 388, 730, 308, 194, 0, 342, 422, 536],
    [468, 1016, 788, 1164, 1050, 514, 514, 662, 320, 274, 388, 650, 536, 342, 0, 764, 194],
    [776, 868, 1552, 560, 674, 1050, 1278, 742, 1084, 810, 1152, 274, 388, 422, 764, 0, 798],
    [662, 1210, 754, 1358, 1244, 708, 480, 856, 514, 468, 354, 844, 730, 536, 194, 798, 0],
    # fmt: on
  ]
  data["num_vehicles"] = 4
  data["depot"] = 0
  return data


[(456, 320), # location 0 - the depot
(228, 0),    # location 1
(912, 0),    # location 2
(0, 80),     # location 3
(114, 80),   # location 4
(570, 160),  # location 5
(798, 160),  # location 6
(342, 240),  # location 7
(684, 240),  # location 8
(570, 400),  # location 9
(912, 400),  # location 10
(114, 480),  # location 11
(228, 480),  # location 12
(342, 560),  # location 13
(684, 560),  # location 14
(0, 640),    # location 15
(798, 640)]  # location 16
```

```
def distance_callback(from_index, to_index):
    """Returns the distance between the two nodes."""
    # Convert from routing variable Index to distance matrix NodeIndex.
    from_node = manager.IndexToNode(from_index)
    to_node = manager.IndexToNode(to_index)
    return data["distance_matrix"][from_node][to_node]

transit_callback_index = routing.RegisterTransitCallback(distance_callback)
routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)
```

Добавить измерение расстояния

Чтобы решить этот VRP, вам нужно создать измерение расстояния, которое вычисляет совокупное расстояние, пройденное каждым транспортным средством на своем маршруте. Затем вы можете установить стоимость, пропорциональную максимальному общему расстоянию по каждому маршруту. Программы маршрутизации используют измерения для отслеживания количеств, которые накапливаются на маршруте транспортного средства. См. Размеры для более подробной информации.

Следующий код создает измерение расстояния с помощью метода AddDimension решателя. Аргумент transit_callback_index — это индекс для Distance_Callback .

питон
C++
Джава
C#

```
dimension_name = "Distance"
routing.AddDimension(
    transit_callback_index,
    0,  # no slack
    3000,  # vehicle maximum travel distance
    True,  # start cumul to zero
    dimension_name,
)
distance_dimension = routing.GetDimensionOrDie(dimension_name)
distance_dimension.SetGlobalSpanCostCoefficient(100)
```

Метод SetGlobalSpanCostCoefficient задает большой коэффициент (100) для глобального промежутка маршрутов, который в данном примере является максимальным из расстояний маршрутов. Это делает глобальный отрезок преобладающим фактором в целевой функции, поэтому программа минимизирует длину самого длинного маршрута.

Добавьте принтер решения

Функция, которая печатает решение, показана ниже.

питон

C++

Джава

C#

```
def print_solution(data, manager, routing, solution):
    """Prints solution on console."""
    print(f"Objective: {solution.ObjectiveValue()}")
    max_route_distance = 0
    for vehicle_id in range(data["num_vehicles"]):
        index = routing.Start(vehicle_id)
        plan_output = f"Route for vehicle {vehicle_id}:\n"
        route_distance = 0
        while not routing.IsEnd(index):
            plan_output += f" {manager.IndexToNode(index)} -> "
            previous_index = index
            index = solution.Value(routing.NextVar(index))
            route_distance += routing.GetArcCostForVehicle(
                previous_index, index, vehicle_id
            )
        plan_output += f"{manager.IndexToNode(index)}\n"
        plan_output += f"Distance of the route: {route_distance}m\n"
        print(plan_output)
        max_route_distance = max(route_distance, max_route_distance)
    print(f"Maximum of the route distances: {max_route_distance}m")

"""Simple Vehicles Routing Problem (VRP).

    This is a sample using the routing library python wrapper to solve a VRP
    problem.
    A description of the problem can be found here:
```

```
  http://en.wikipedia.org/wiki/Vehicle_routing_problem.

  Distances are in meters.
"""

from ortools.constraint_solver import routing_enums_pb2
from ortools.constraint_solver import pywrapcp


def create_data_model():
  """Stores the data for the problem."""
  data = {}
  data["distance_matrix"] = [
    # fmt: off
    [0, 548, 776, 696, 582, 274, 502, 194, 308, 194, 536, 502, 388, 354, 468, 776, 662],
    [548, 0, 684, 308, 194, 502, 730, 354, 696, 742, 1084, 594, 480, 674, 1016, 868, 1210],
    [776, 684, 0, 992, 878, 502, 274, 810, 468, 742, 400, 1278, 1164, 1130, 788, 1552, 754],
    [696, 308, 992, 0, 114, 650, 878, 502, 844, 890, 1232, 514, 628, 822, 1164, 560, 1358],
    [582, 194, 878, 114, 0, 536, 764, 388, 730, 776, 1118, 400, 514, 708, 1050, 674, 1244],
    [274, 502, 502, 650, 536, 0, 228, 308, 194, 240, 582, 776, 662, 628, 514, 1050, 708],
    [502, 730, 274, 878, 764, 228, 0, 536, 194, 468, 354, 1004, 890, 856, 514, 1278, 480],
    [194, 354, 810, 502, 388, 308, 536, 0, 342, 388, 730, 468, 354, 320, 662, 742, 856],
    [308, 696, 468, 844, 730, 194, 194, 342, 0, 274, 388, 810, 696, 662, 320, 1084, 514],
    [194, 742, 742, 890, 776, 240, 468, 388, 274, 0, 342, 536, 422, 388, 274, 810, 468],
    [536, 1084, 400, 1232, 1118, 582, 354, 730, 388, 342, 0, 878, 764, 730, 388, 1152, 354],
    [502, 594, 1278, 514, 400, 776, 1004, 468, 810, 536, 878, 0, 114, 308, 650, 274, 844],
    [388, 480, 1164, 628, 514, 662, 890, 354, 696, 422, 764, 114, 0, 194, 536, 388, 730],
    [354, 674, 1130, 822, 708, 628, 856, 320, 662, 388, 730, 308, 194, 0, 342, 422, 536],
    [468, 1016, 788, 1164, 1050, 514, 514, 662, 320, 274, 388, 650, 536, 342, 0, 764, 194],
    [776, 868, 1552, 560, 674, 1050, 1278, 742, 1084, 810, 1152, 274, 388, 422, 764, 0, 798],
    [662, 1210, 754, 1358, 1244, 708, 480, 856, 514, 468, 354, 844, 730, 536, 194, 798, 0],
    # fmt: on
  ]
  data["num_vehicles"] = 4
  data["depot"] = 0
  return data
```

```
def print_solution(data, manager, routing, solution):
    """Prints solution on console."""
    print(f"Objective: {solution.ObjectiveValue()}")
    max_route_distance = 0
    for vehicle_id in range(data["num_vehicles"]):
        index = routing.Start(vehicle_id)
        plan_output = f"Route for vehicle {vehicle_id}:\n"
        route_distance = 0
        while not routing.IsEnd(index):
            plan_output += f" {manager.IndexToNode(index)} -> "
            previous_index = index
            index = solution.Value(routing.NextVar(index))
            route_distance += routing.GetArcCostForVehicle(
                previous_index, index, vehicle_id
            )
        plan_output += f"{manager.IndexToNode(index)}\n"
        plan_output += f"Distance of the route: {route_distance}m\n"
        print(plan_output)
        max_route_distance = max(route_distance, max_route_distance)
    print(f"Maximum of the route distances: {max_route_distance}m")



def main():
    """Entry point of the program."""
    # Instantiate the data problem.
    data = create_data_model()

    # Create the routing index manager.
    manager = pywrapcp.RoutingIndexManager(
        len(data["distance_matrix"]), data["num_vehicles"], data["depot"]
    )

    # Create Routing Model.
    routing = pywrapcp.RoutingModel(manager)

    # Create and register a transit callback.
    def distance_callback(from_index, to_index):
```

```
    """Returns the distance between the two nodes."""
    # Convert from routing variable Index to distance matrix NodeIndex.
    from_node = manager.IndexToNode(from_index)
    to_node = manager.IndexToNode(to_index)
    return data["distance_matrix"][from_node][to_node]

transit_callback_index = routing.RegisterTransitCallback(distance_callback)

# Define cost of each arc.
routing.SetArcCostEvaluatorOfAllVehicles(transit_callback_index)

# Add Distance constraint.
dimension_name = "Distance"
routing.AddDimension(
    transit_callback_index,
    0,  # no slack
    3000,  # vehicle maximum travel distance
    True,  # start cumul to zero
    dimension_name,
)
distance_dimension = routing.GetDimensionOrDie(dimension_name)
distance_dimension.SetGlobalSpanCostCoefficient(100)

# Setting first solution heuristic.
search_parameters = pywrapcp.DefaultRoutingSearchParameters()
search_parameters.first_solution_strategy = (
    routing_enums_pb2.FirstSolutionStrategy.PATH_CHEAPEST_ARC
)

# Solve the problem.
solution = routing.SolveWithParameters(search_parameters)

# Print solution on console.
if solution:
    print_solution(data, manager, routing, solution)
else:
    print("No solution found !")
```

```
if __name__ == "__main__":
    main()
def create_distance_matrix(data):
  addresses = data["addresses"]
  API_key = data["API_key"]
  # Distance Matrix API only accepts 100 elements per request, so get rows in multiple requests.
  max_elements = 100
  num_addresses = len(addresses) # 16 in this example.
  # Maximum number of rows that can be computed per request (6 in this example).
  max_rows = max_elements // num_addresses
  # num_addresses = q * max_rows + r (q = 2 and r = 4 in this example).
  q, r = divmod(num_addresses, max_rows)
  dest_addresses = addresses
  distance_matrix = []
  # Send q requests, returning max_rows rows per request.
  for i in range(q):
    origin_addresses = addresses[i * max_rows: (i + 1) * max_rows]
    response = send_request(origin_addresses, dest_addresses, API_key)
    distance_matrix += build_distance_matrix(response)

  # Get the remaining remaining r rows, if necessary.
  if r > 0:
    origin_addresses = addresses[q * max_rows: q * max_rows + r]
    response = send_request(origin_addresses, dest_addresses, API_key)
    distance_matrix += build_distance_matrix(response)
  return distance_matrix
```

ДОДАТОК Д

ІЛЮСТРАТИВНА ЧАСТИНА

Система оптимального управління маршрутами літальних засобів з урахуванням точок приземлення
(тема)

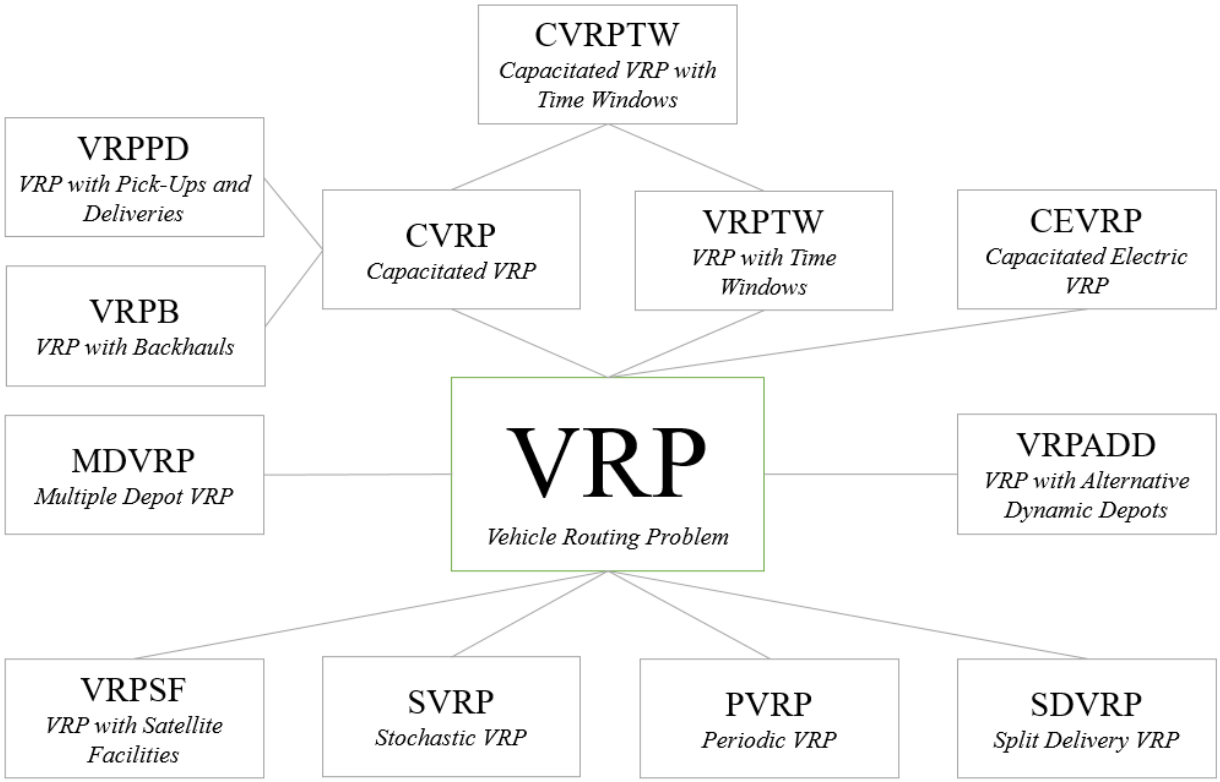
1. Класифікація задач управління маршрутами транспортних засобів
2. Класифікація наближених методів розв'язання задач управління маршрутами транспортних засобів
3. Схема структурна діяльності
4. Блок-схема методу найдешевшого включення
5. Блок-схема гібридного алгоритму найдешевшого включення з проміжними покращеннями
6 Схема структурна розгортання
7 Схема структурна послідовності
8 Діаграма класів
9 Загальна поетапна схема роботи алгоритмів маршрутизації
10 Псевдокод для методу найдешевшого включення
11. Головна сторінка
12 Сторінка з інформацією про програму
13 Приклад візуалізованих результатів виконання алгоритмів
14 Аналіз результатів
15 Швидкодія роботи програмного забезпечення

Студент групи 2АКІТ-22м

Юрій САВЧЕНКО
Підпис *Ім'я ПРІЗВИЩЕ*

Керівник к.т.н., доцент кафедри КСУ
Олена НИКИТЕНКО
Підпис *Ім'я ПРІЗВИЩЕ*

Класифікація задач управління маршрутами транспортних засобів



Класифікація наближених методів розв’язання задач управління маршрутами транспортних засобів

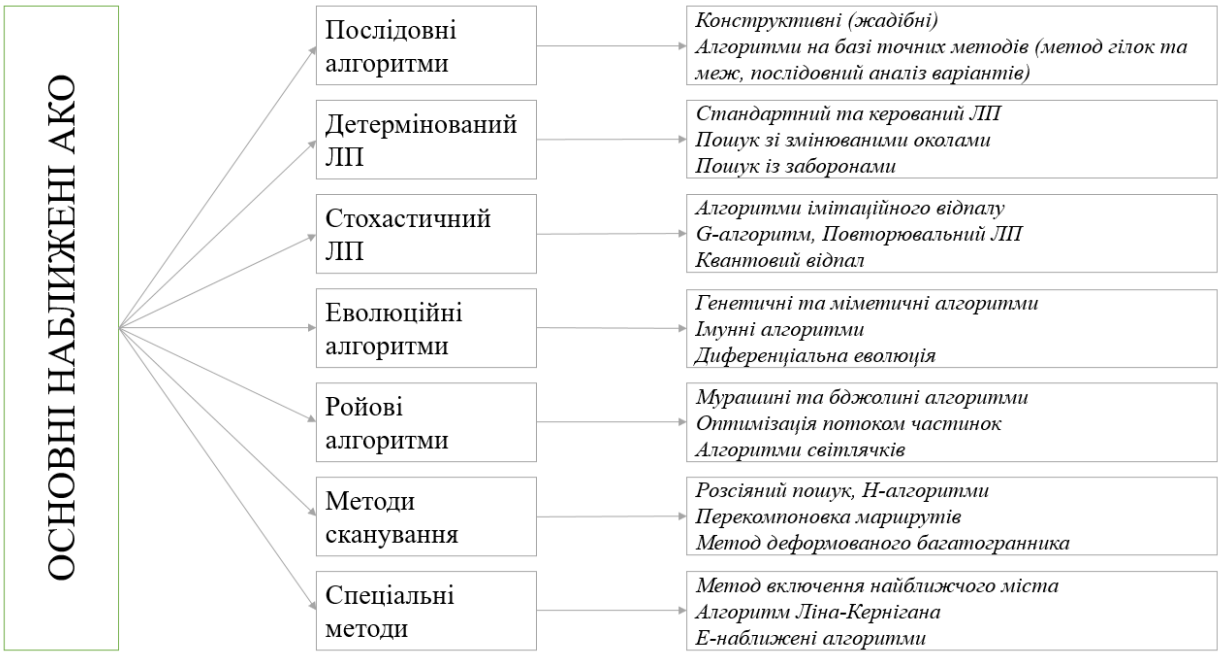
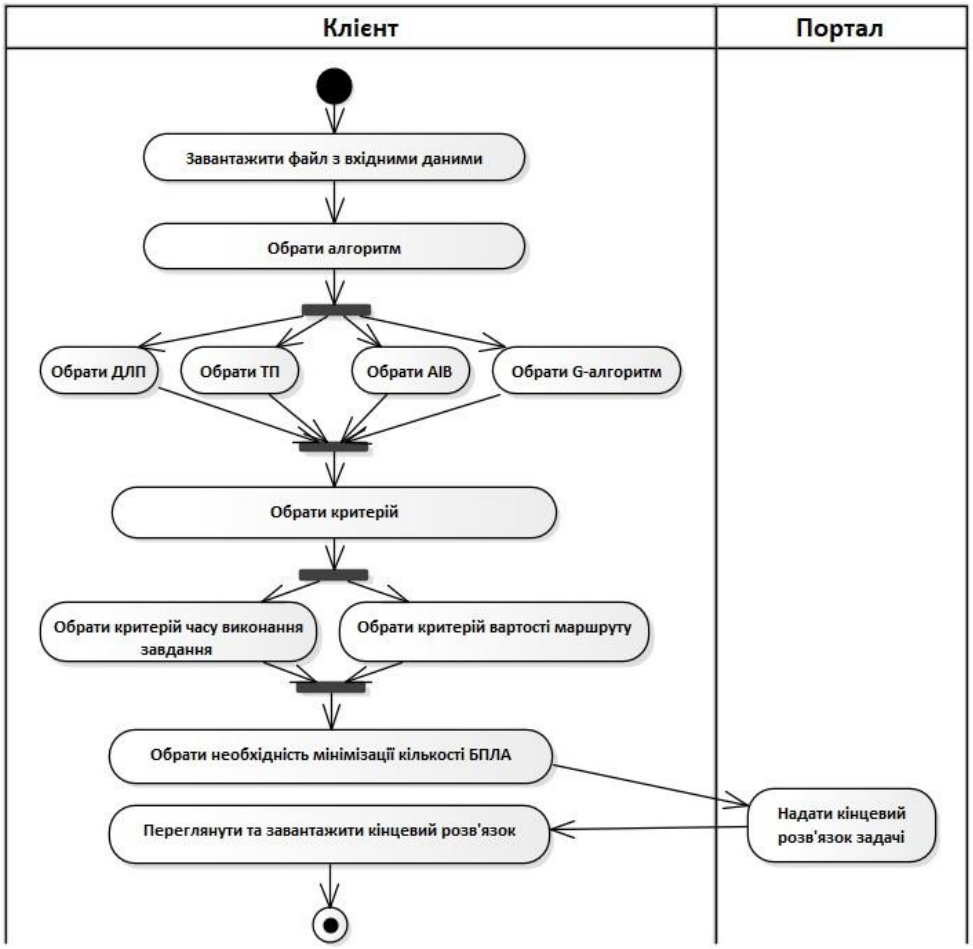


Схема структурна діяльності



Блок-схема методу найдешевшого включення

Блок-схема гібридного алгоритму найдешевшого включення з проміжними покращеннями

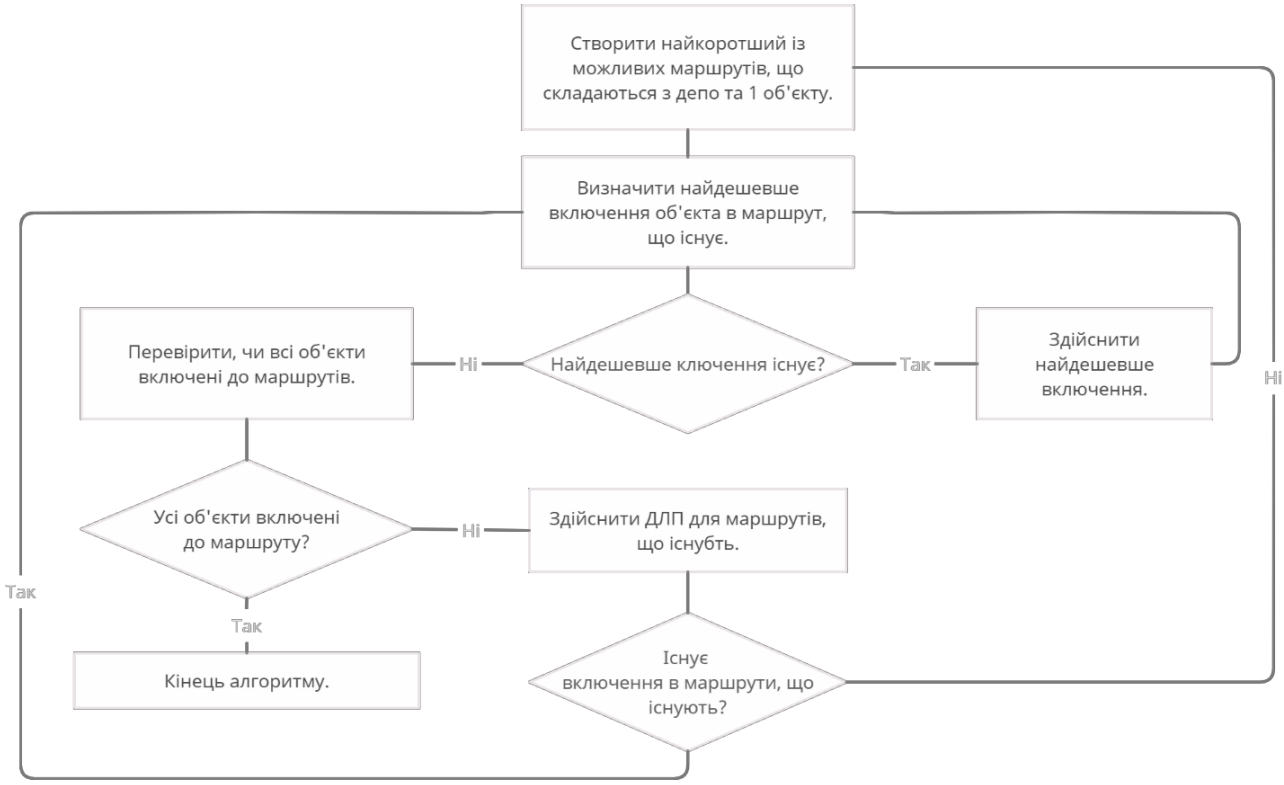


Схема структурна розгортання

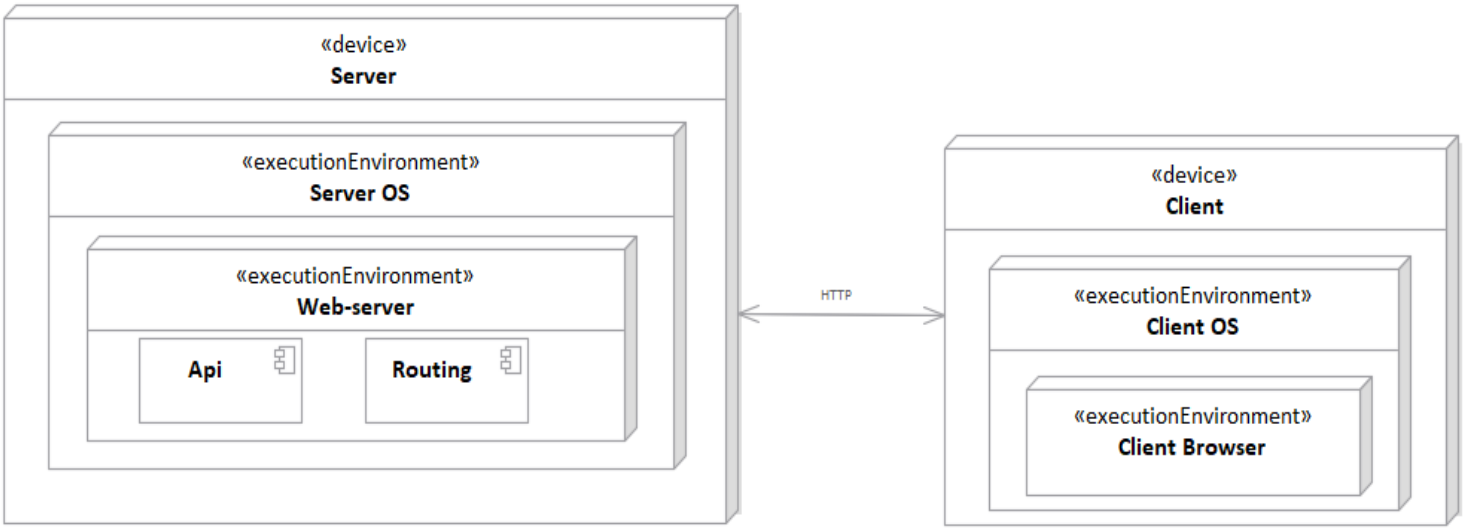
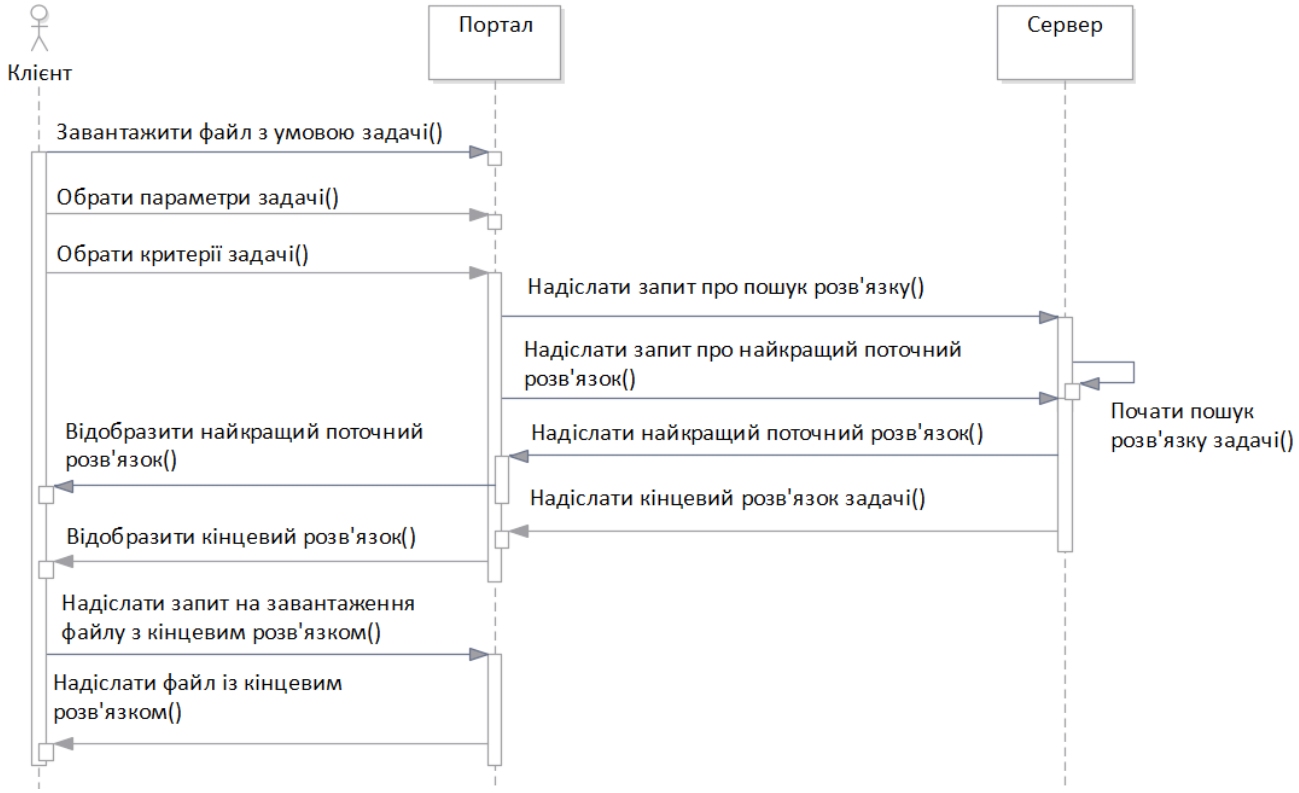
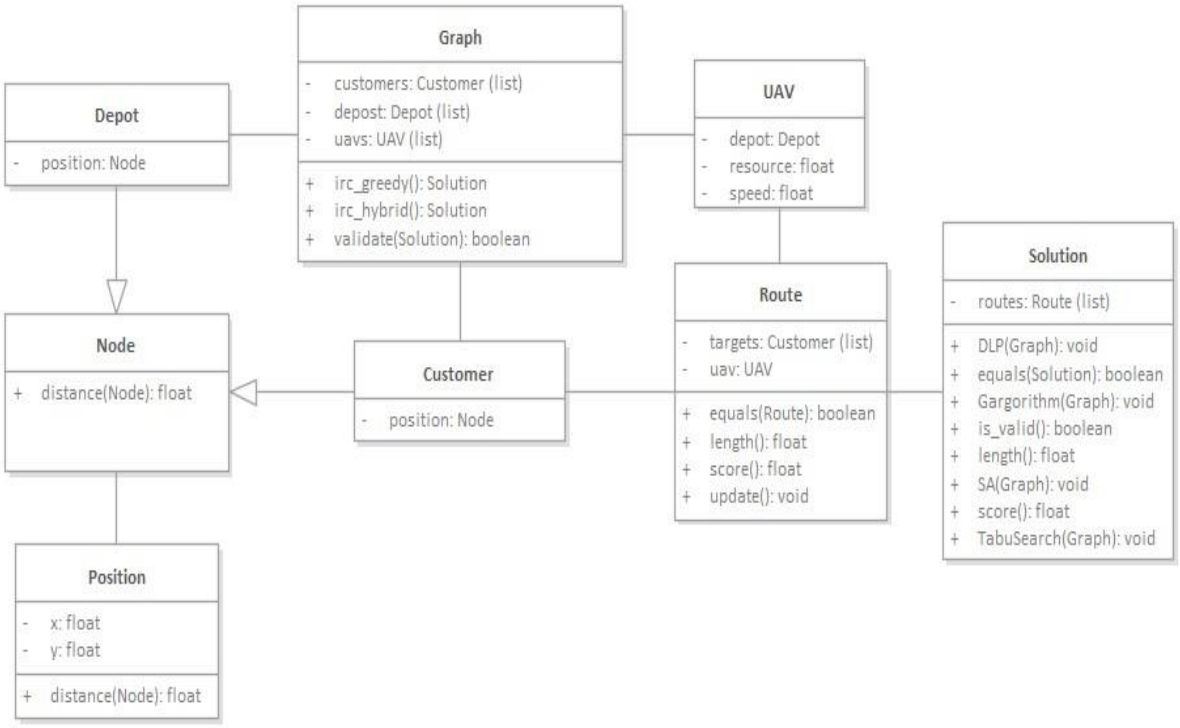


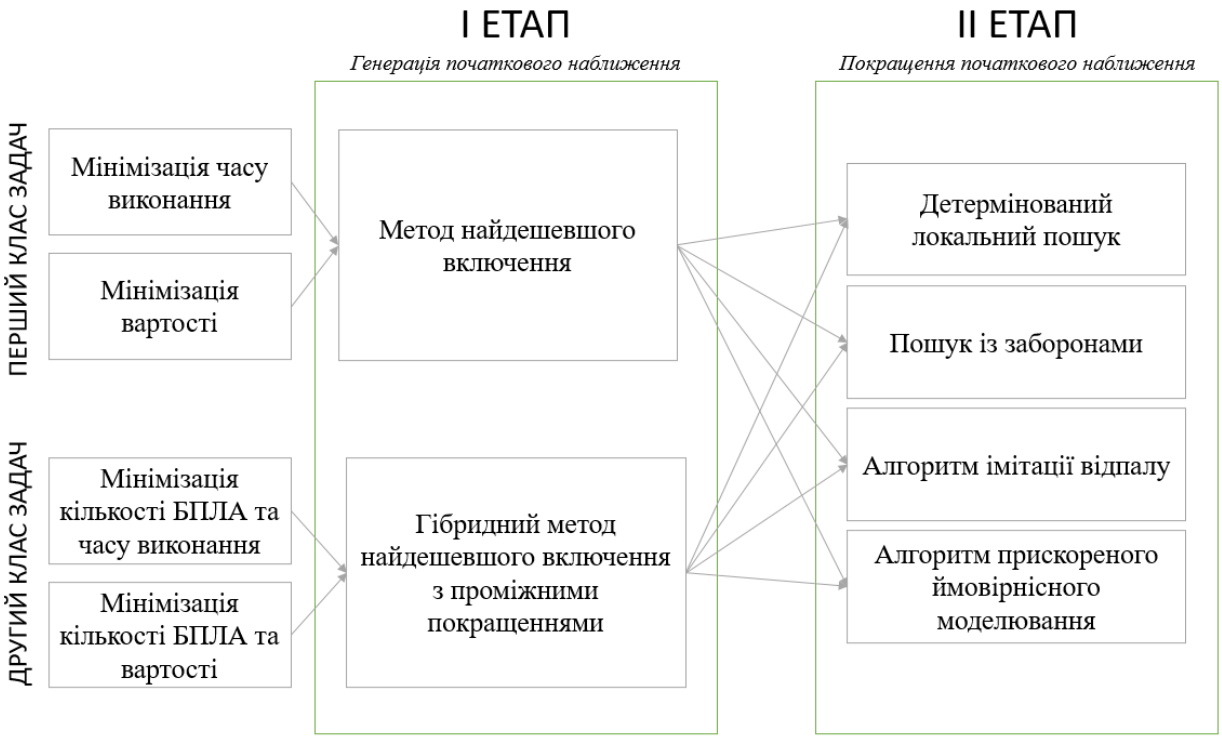
Схема структурна послідовності



Плакат 6 Діаграма класів



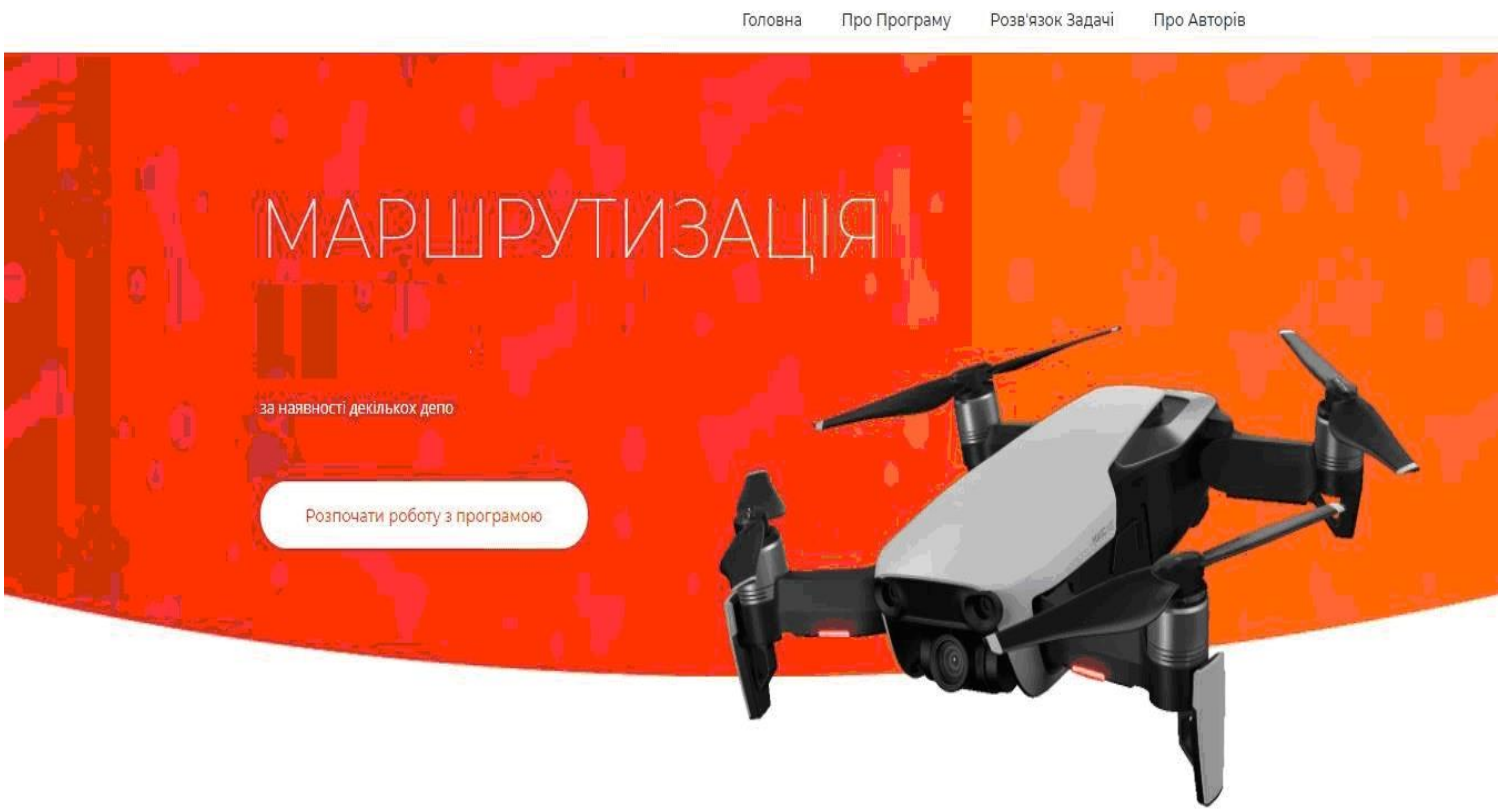
Загальна поетапна схема роботи алгоритмів маршрутизації



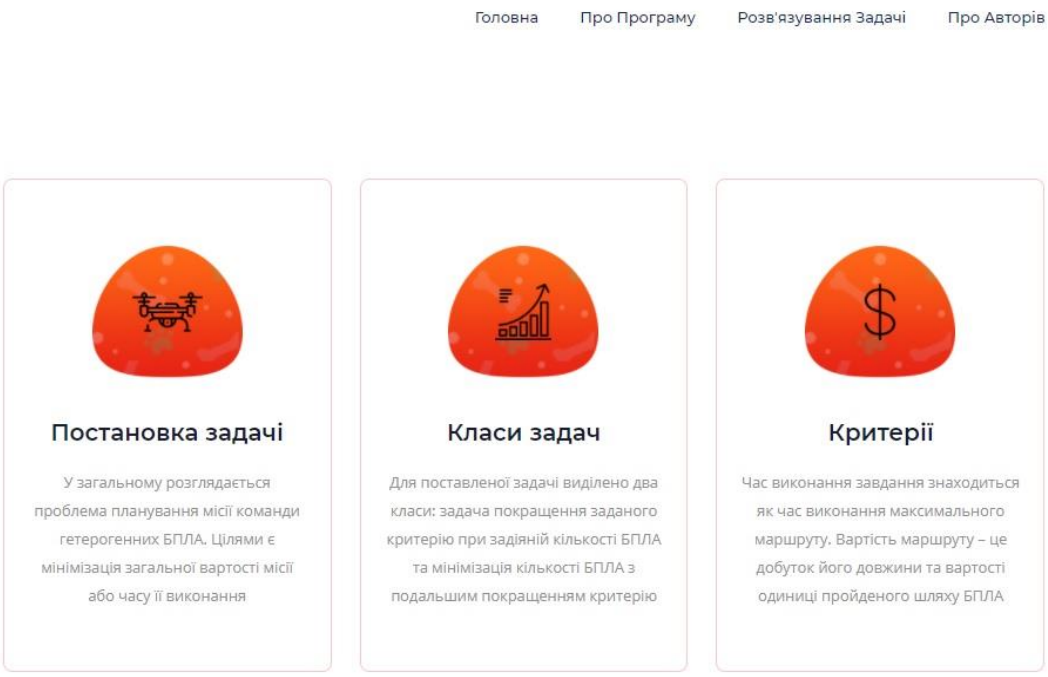
Псевдокод для методу найдешевшого включення

```
procedure НайдешевшеВключення
    невикористані об'єкти := всі об'єкти
    створити найдешевший маршрут
    видалити об'єкт зі списку невикористаних
    while (існують невикористані об'єкти)
        x := ціна найдешевшого можливого включення
        y := ціна створення найдешевшого маршруту
        if (x > y)
            здійснити найдешевше можливе включення
        else
            створити найдешевший маршрут
        endelse
        видалити об'єкт зі списку невикористаних
    endwhile
end
```

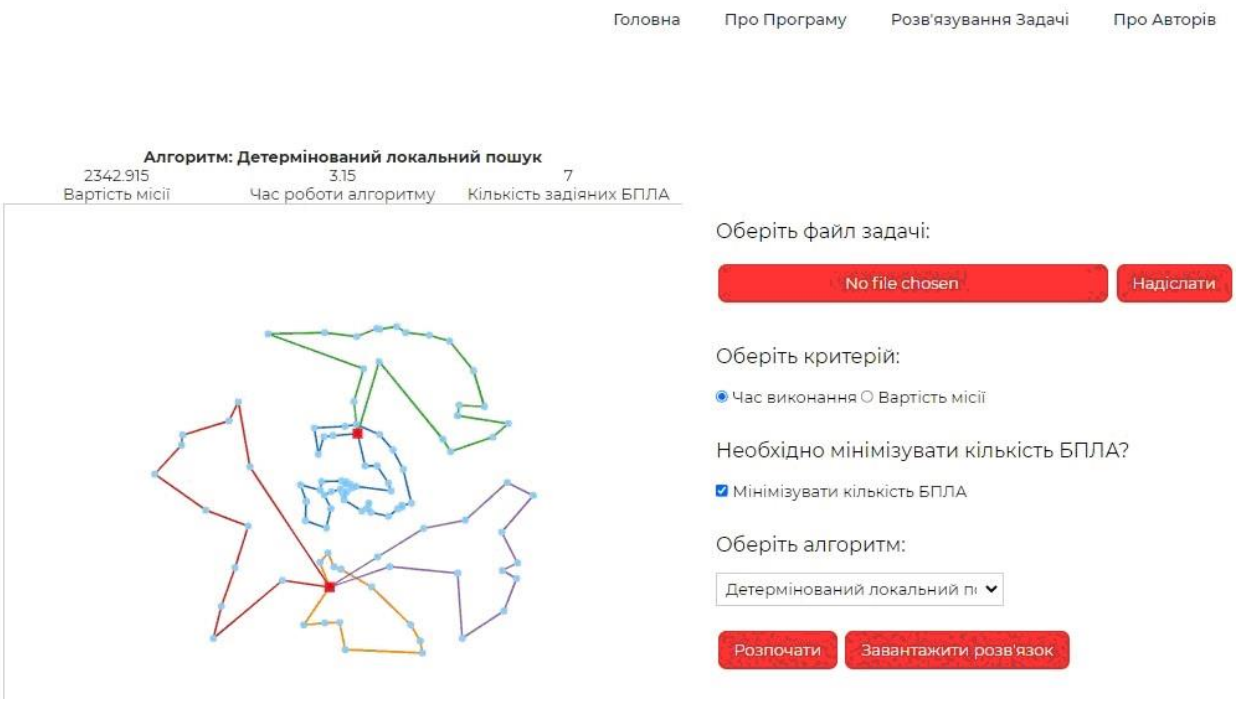
Головна сторінка



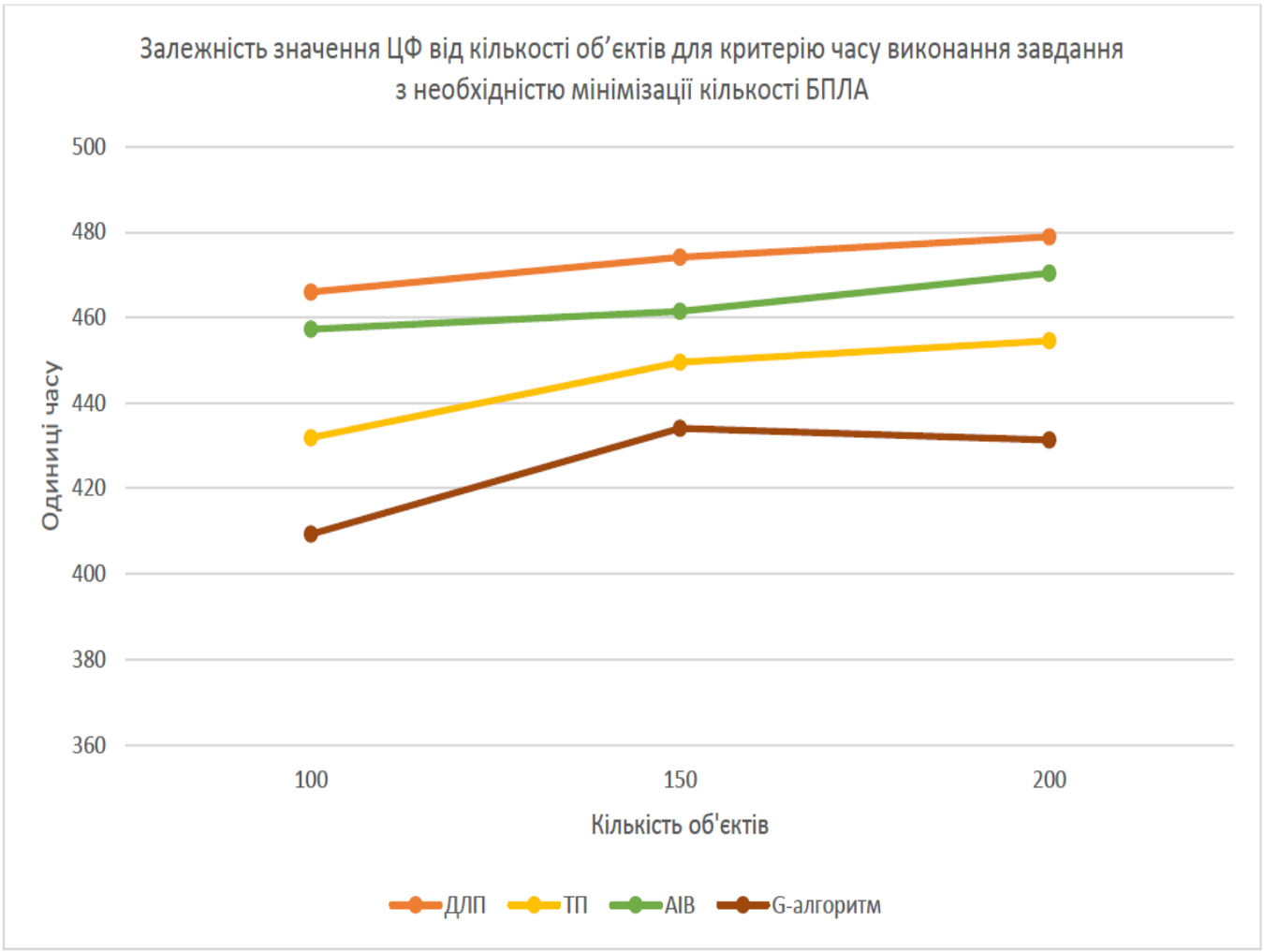
Сторінка з інформацією про програму



Приклад візуалізованих результатів виконання алгоритмів



Аналіз результатів



Швидкодія роботи програмного забезпечення