НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

"На правах рукопису"
УДК 519.854.2 + 004.021

До захисту допущено
Завідувач кафедри

_____ Олександр РОЛІК
)
« » 20 22 р.

МАГІСТЕРСЬКА ДИСЕРТАЦІЯ

на здобуття ступеня магістра
за освітньо-професійною програмою
«Інформаційне забезпечення робототехнічних систем»
зі спеціальності 126 «Інформаційні системи та технології»
на тему:

«Інформаційна система з підтримки дослідження задачі складання маршрутів БПЛА з рухомою базою обслуговування»

Виконав:
студент VI курсу, групи **ІК-11мп**
Шинягін Максим Дмитрович _____

Керівник:
професор каф. ІСТ, д.т.н., с.н.с.,
Гуляницький Леонід Федорович _____

Рецензент:
старший викладач каф. ОТ, к.т.н.,
Антонюк Андрій Іванович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – другий (магістерський)
Спеціальність – 126 «Інформаційні системи та технології»
Освітньо-професійна програма «Інформаційне забезпечення робототехнічних систем»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Олександр РОЛІК
«___»_____2022 р.

ЗАВДАННЯ
на магістерську дисертацію студенту
Шинягіну Максиму Дмитровичу

1. Тема дисертації «Інформаційна система з підтримки дослідження задачі складання маршрутів БПЛА з рухомою базою обслуговування», науковий керівник дисертації Гуляницький Леонід Федорович, доцент, д.т.н., затверджений наказом по університету від «09» 11 2022 р. № 4115-с
2. Термін подання студентом дисертації «12» 12 2022 р.
3. Об’єкт дослідження

Задача складання маршрутів БПЛА з рухомою базою обслуговування

4. Вихідні дані

Результати роботи алгоритмів та досліджень задачі складання маршрутів БПЛА з рухомою базою обслуговування

5. Перелік завдань, які потрібно розробити

- дослідити предметну область;
- розробити алгоритми для складання маршрутів БПЛА з рухомою базою обслуговування;
- розробити програмний продукт;
- провести аналіз отриманих результатів;

6. Орієнтовний перелік графічного (ілюстративного) матеріалу
7. Орієнтовний перелік публікацій
8. Дата видачі завдання «05» 09 2022 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Дослідження предметної області	08.09.2022	
2	Огляд існуючих рішень та методів	15.09.2022	
3	Розробка алгоритмів	22.09.2022	
4	Розробка програмного продукту	13.10.2022	
5	Тестування програмного продукту	20.10.2022	
6	Створення документації та аналіз отриманих результатів	27.10.2022	
7	Подання роботи на основний захист	12.12.2022	

Студент Максим ШИНЯГІН

Науковий керівник Леонід ГУЛЯНИЦЬКИЙ

РЕФЕРАТ

Магістерська дисертація: 102 с., 23 рис., 37 табл., 9 додатків, 49 джерел.

Актуальність. Особливий інтерес до планування маршруту БПЛА виявляють військові відомства різних країн. Крім того БПЛА активно використовуються в цивільній сфері, тож обрана тема є актуальною на поточний момент.

Метою роботи є розробка алгоритмів для дослідження задачі складання маршрутів БПЛА з рухомою базою обслуговування та дослідження їх ефективності.

Об’єктом дослідження знаходження маршрутів БПЛА для задачі, а предмет дослідження – методи знаходження маршрутів.

Завдання дослідження:

— провести аналіз предметної області;

— сформулювати математичну модель досліджуваної задачі;

— провести аналіз методів розв’язання задачі;

— розробити алгоритми розв’язання досліджуваної задачі;

— розробити інформаційну систему з підтримки проведення досліджень ефективності розроблених алгоритмів;

— провести аналіз результатів експериментів.

Наукова новизна отриманих результатів полягає в розробці нових алгоритмів вирішення поставлених завдань.

Практичне значення одержаних результатів. Розроблені алгоритми можна використовувати в системах складання маршрутів БПЛА, що працюють в реальних умовах.

АЛГОРИТМИ КОМБІНАТОРНОЇ ОПТИМІЗАЦІЇ, ПЛАНУВАННЯ, МАРШРУТИЗАЦІЯ, БЕЗПІЛОТНИЙ ЛІТАЛЬНИЙ АПАРАТ, ПАРАМЕТРИ АЛГОРИТМІВ, НАЛАШТУВАННЯ ПАРАМЕТРІВ, ПЛАНУВАННЯ ОПЕРАЦІЇ, VRP

ABSTRACT

Master's thesis: 102 pages, 23 figures, 37 tables, 9 appendices, 49 sources.

Topicality. The military departments of various countries show special interest in planning the UAV route. In addition, UAVs are actively used in the civilian sphere, so the chosen topic is relevant at the current moment.

The purpose of the work is the development of algorithms for the study of the task of constructing UAV routes with a mobile service base and the study of their effectiveness.

The object of research is finding UAV routes for the task, and the subject of research is route finding methods.

Objectives of the study:

— conduct an analysis of the subject area;

— formulate a mathematical model of the researched problem;

— conduct an analysis of problem solving methods;

— develop algorithms for solving the researched problem;

— develop an information system to support conducting research on the effectiveness of the developed algorithms;

— analyze the results of experiments.

The scientific novelty of the obtained results lies in the development of new algorithms for solving the tasks.

Practical significance of the obtained results. The developed algorithms can be used in UAV route planning systems operating in real conditions.

COMBINATORIAL OPTIMIZATION ALGORITHMS, SCHEDULING, ROUTING, UAV, ALGORITHM PARAMETERS, SETTING PARAMETERS, OPERATION PLANNING, VRP

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ТЗ - транспортний засіб

БПЛА - безпілотний літальний апарат

VRP - Vehicle Routing Problem (задача маршрутизації транспортних засобів)

ЗМТЗ - задача маршрутизації транспортного засобу

Система - інформаційна система з підтримки дослідження задачі складання маршруту польоту БПЛА з рухомою базою обслуговування

ВСТУП

Задачі маршрутизації транспортних засобів (VRP) є популярним класом комбінаторних задач у логістиці через їх застосовність і зв’язок із розподілом останньої милі, де детерміновані VRP є одними з найбільш часто використовуваних у реальних додатках, враховуючи наявність методів їх. Детерміновані VRP розглядають ідеальні параметри без невизначеності, але багато програм піддаються різним типам невизначеності, що часто означає, що існує набір можливих рішень, які залежать від моделі/методу їх вирішення. Таким чином, невизначені VRP (стохастичні, інтервальні, нечіткі тощо) виявляють більшу складність, вимагаючи передових моделей і спеціалізованих методів вирішення.

Використання безпілотних літальних апаратів (БПЛА) зростає. Завдяки широкому спектру функцій дронів і підвищеній зручності використання все більше компаній починають використовувати їх у своїй діяльності: ЖКГ – для енергоаудиту будівель, лісове господарство – для боротьби з незаконними рубками, охоронні компанії – пошуку порушників, у сільському господарстві – для поливу, забезпечення оптимальної температури, догляду за структурою посівних площ та контролю землекористування тощо. Найважливішою сферою застосування дронів є військова сфера, де вони використовуються для тактичної та стратегічної повітряної розвідки.

Сьогодні актуальною є проблема мінімізації енерговитрат при виконанні операцій з безпілотними літальними апаратами та скорочення тривалості таких операцій. Одним із завдань є побудова маршруту між заданою множиною точок. Побудова маршруту БПЛА здійснюється заздалегідь, тому що під час польоту можуть виникнути різноманітні перешкоди радіозв'язку, і оператор втратить контроль над літаком, що призведе до збою польоту, тому маршрут БПЛА повинен бути чітко визначений та запрограмований під час планування.

У даній роботі досліджується задача формування маршрутів БПЛА з використанням мобільних сервісних баз, для чого пропонується спеціальний алгоритм мурашиної колонії, оскільки ефективність мурашиного алгоритму доведена досвідом вирішення різноманітних задач транспортної маршрутизації.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд існуючих досліджень

Мета цього підрозділу полягає в тому, щоб розглянути, які типи VRP є актуальними сьогодні, описати їх застосування та визначити, які обмеження та застереження найчастіше з’являлися у постановках задач за останні кілька років.

На тему планування маршруту безпілотних літальних апаратів (БПЛА) було проведено багато досліджень. У 2004 році була сформована загальна модель задачі оптимізації маршруту БПЛА [1]. У наступних дослідженнях були введені нові обмеження: вперше розглянуто обмеження на вантаж в експлуатації [2], встановлено умови існування часових вікон [3], введено обмеження ресурсу польоту [4] . Також розглядаються різні цільові функції: мінімізація загального часу місії [5], мінімізація загальної довжини маршруту [6], мінімізація кількості дронів [7], мінімізація ресурсів польоту [8].

Однією з найважливіших сфер застосування БПЛА є військова сфера, де БПЛА використовуються для повітряної розвідки – як тактичної, так і стратегічної [9,10]. У [2] бойова авіація використовується для ураження намічених цілей з обмеженими боєзапасами. Крім того, [11] запропонував алгоритм вирішення проблеми збору інформації за найкоротший час за допомогою парку дронів. Проблема планування маршруту в умовах уникнення загроз (виявлення радарів, перешкод, зіткнення з іншими БПЛА) розглядається в [12].

Сфери застосування БПЛА з кожним роком розширюються, що призводить до активного створення нових постановок задач. У роботі Ines Khouf «A Survey of Recent Extended Variants of the Traveling Salesman and Vehicle Routing Problems for Unmanned Aerial Vehicles» [13], проблема оптимізації маршруту БПЛА детально аналізується у двох класах задач: проблема комівояжера (TSP) і проблема маршрутизації транспортного засобу (VRP). У цій статті розглядаються спільні місії між дронами та вантажівками. Вантажівка служить платформою для транспортування завантажених вантажем безпілотників, які, у свою чергу, доставляють товари клієнтам. Для цього завдання було обрано п’ять можливих конфігурацій:

— кількість дронів =1, використовується лише один дрон;

— кількість безпілотників = m, парк розгорнутих безпілотників;

— кількість дронів = 1, кількість вантажівок = 1, один дрон працює з однією вантажівкою;

— кількість дронів = m, кількість вантажівок = 1, парк дронів працює з однією вантажівкою;

— кількість дронів = m, кількість вантажівок = n, парк дронів працює з багатьма вантажівками.

Усі існуючі задачі маршрутизації безпілотників, опубліковані за останні кілька років, є варіантами задачі комівояжера або задачі маршрутизації транспортного засобу. У [13] наведена така класифікація.

Розширені варіанти задачі комівояжера

TSP-D - проблема комівояжера з використанням дронів. Дано набір локацій, склад, дрон, індикатор вартості. Метою TSP-D є визначення маршруту таким чином, щоб кожна ціль була відвідана лише один раз, і в кінці БПЛА повертався на базу, мінімізуючи загальну вартість маршруту (наприклад, пройдену відстань або час на проходження).

mTSP-D є узагальненням TSP-D з використанням кількох дронів. Завдання mTSP-D полягає в тому, щоб ідентифікувати набір з m маршрутів таким чином, щоб загальна вартість усіх маршрутів була мінімізована, а кожну ціль відвідував лише один БПЛА.

ATSP-D – також узагальнення TSP-D, за винятком того, що вартість між вершинами може змінюватися залежно від напрямку.

CETSP-D є узагальненням TSP-D, де кожна ціль ідентифікується за її околицями. Мета полягає в тому, щоб визначити маршрут БПЛА, який відвідує околиці кожної цілі лише один раз і повертається на склад, мінімізуючи загальну вартість маршруту.

Розширені варіанти задачі маршрутизації транспортних засобів

Основні класи задач маршрутизації транспортних засобів наведено на рисунку 1.1.

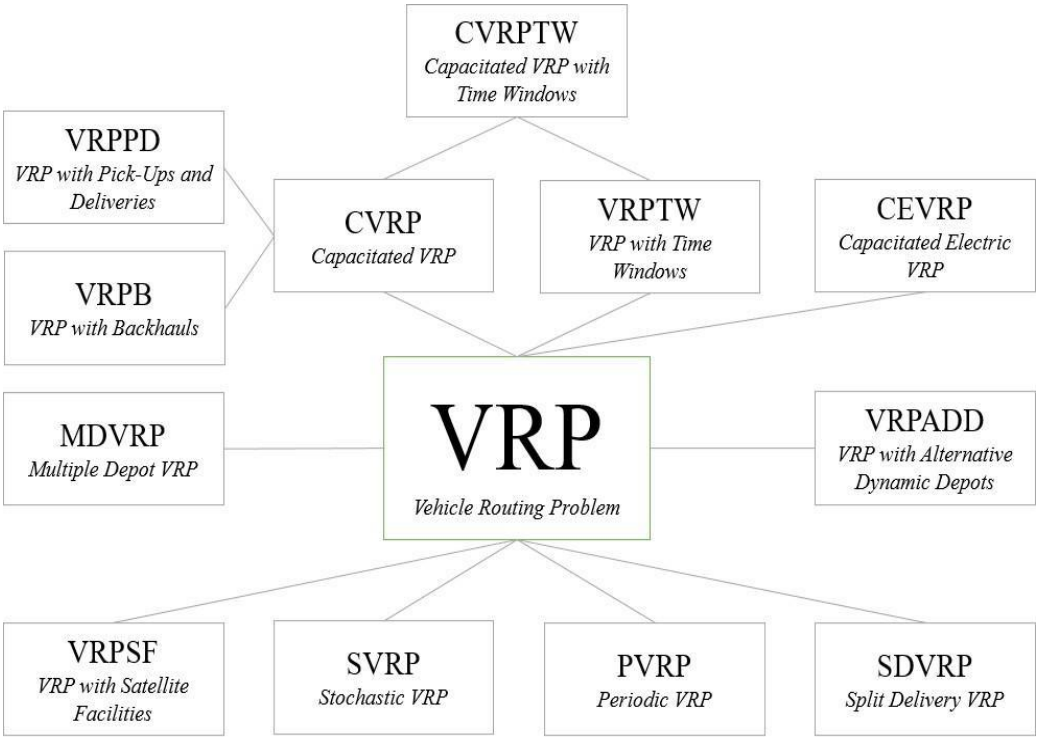


Рисунок 1.1 – Ієрархія основних класів задач маршрутизації транспортних засобів [13]

Розглянемо більш детально деякі з них.

VRP-D – завдання з наведення техніки з безпілотниками. VRP-D має на меті визначити набір маршрутів для парку безпілотників, які відвідують набір цілей, мінімізуючи загальну вартість маршрутів.

VRPTW-D є узагальненням VRP-D, де обслуговування кожного клієнта починається в певний час, який називається часовим вікном. Вікно встановлюється для кожного клієнта і визначається часовим інтервалом.

CVRPTW-D – це узагальнення VRPTW-D, де кожен дрон має певну потужність (наприклад, корисне навантаження), а загальний попит, який обслуговує кожен дрон, не перевищує його потужності.

GVRP-D є варіантом VRP-D, метою якого є зменшення кількості транспортних засобів, які використовуються для заправки безпілотників.

MTVRP-D — це варіант GVRP-D, призначений для подолання обмеженого корисного навантаження, що дозволяє БПЛА повертатися на базу та виконувати кілька польотів.

Безпілотники можна використовувати для наступних цілей:

— доставка та перевезення: доставка посилок, товарів, реклами тощо;

— зв'язок: БПЛА можна використовувати як точку ретрансляції, що літає, для відновлення зв'язку;

— спостереження, стеження: дрони, оснащені датчиками, можуть використовуватися для того, щоб слідкувати за певними територіями;

— логістичний процес: дрони можна використовувати для інвентаризації складів;

— лиха: дрони можна використовувати для гасіння пожеж або пошуку людей чи тварин.

Крім того, багато уваги приділено алгоритмам оптимізації маршрутів.

Для вирішення надскладних задач, таких як планування маршруту БПЛА, використовується окремий клас методів оптимізації для вирішення задач комбінаторної оптимізації – метаевристики. Вони покладаються на вдосконалення локальних рішень протягом кількох ітерацій для отримання квазіоптимальних рішень. Останніми прикладами метаевристичних алгоритмів для планування траєкторії БПЛА є імітований відпал [14], генетичний алгоритм [15], диференціальна еволюція [16], оптимізація потоку частинок [17], оптимізація колонії мурах [18], алгоритм колонії бджіл [19], Алгоритм жертва-хижак [20], алгоритм сірого вовка [21] тощо. Кожен із цих підходів пропонує унікальні стратегії для модифікації та вдосконалення рішень, і їх ефективність часто залежить від поточної проблеми.

Питання, розглянуті в цьому проєкті, відрізняються від розглянутих вище тим, що:

— БПЛА знаходиться на базі мобільного обслуговування;

— дрони мають можливість відновлювати запаси енергії на базі обслуговування;

— склад – це транспортний засіб, який може пересуватися незалежно від дрона.

Використання баз мобільних служб для визначення місцезнаходження дронів особливо актуальне сьогодні. Програма «Гремліни» Агентства передових оборонних досліджень США (DARPA) розроблялася кілька років. Програма передбачає запуск БПЛА з існуючих великих літальних апаратів, таких як бомбардувальники або транспортні літаки, а також винищувачів, стаціонарних платформ [22].

Проблема маршрутизації БПЛА перетворюється з нової теми на зростаючу сферу досліджень. Сучасні технологічні розробки поступово рухаються в напрямку того, що дрони відіграють ключову роль у міській логістиці. На даний момент маршрутні місії БПЛА стали зрілою технологією, яка є лідером у галузі. Безпілотники можуть значно скоротити фінансові витрати та час, необхідний для доставки товарів, оскільки вони дешевші в обслуговуванні, ніж традиційні транспортні засоби. Наприклад, безпілотники можуть зменшити витрати на робочу силу, виконуючи завдання автономно. На рисунку 1.2 показано таксономію прикладних областей завдань маршрутизації БПЛА. Загалом їх можна розділити на дві категорії: цивільні та військові.

Рисунок 1.2 – Сфери застосування БПЛА

Розглянемо більш детально область застосування дронів, показану на рисунку 1.2.

Цивільна сфера включає такі завдання:

— виявлення малих об'єктів (повітря, вода, земля, пошук і порятунок, допомога в надзвичайних ситуаціях);
— управління повітряним рухом (у важкодоступних районах, під час стихійних лих і аварій, на тимчасових маршрутах під час аеронавігаційних робіт);
— контроль морського руху (пошук і виявлення суден, запобігання надзвичайним ситуаціям у портах, контроль морських кордонів, контроль правил рибальства);
— застосування в сільському господарстві (продуктивність в аерокосмічній та хімічній техніці);
— застосування в геологорозвідці (визначення властивостей ґрунтів, розвідка корисних копалин, дослідження земних надр);
— розвиток регіональних і міжрегіональних телекомунікаційних мереж (системи зв'язку, мовлення, ретрансляції, системи навігації, реклама, телебачення, кіно);
— аерофотозйомка та контроль земної поверхні (картографія, контроль стану води та погоди, правоохоронна діяльність, виявлення лісових пожеж, спостереження за периметром об'єктів, контроль залізничних колій);
— контроль стану довкілля (радіаційний контроль, газохімічний контроль, контроль стану газонафтопроводів, контроль лавинного стану) [31].

Військова сфера включає такі завдання:

— розвідка (повітряна розвідка із середніх і малих висот в умовах, сильно природнх лих і складній радіоелектронній обстановці з можливістю оперативної передачі отриманої розвідувальної інформації із захищених радіоканалів; виявлення та ідентифікація наземних, надводних і повітряних цілей, угруповання військ і засобів, вогневі точки та укріплення; виявлення об'єктів, що активно випромінюють);

— вогневі удари (ураження техніки і живої сили противника шляхом ураження наземних і морських цілей; ураження повітряних цілей і знищення боєголовок балістичних ракет; дистанційне створення мінних полів, очищення території);

— постачання (перевезення вантажів у різних бойових ситуаціях, у тому числі в період інтенсивних бойових дій і на території, забрудненій радіоактивними, хімічними та біологічними речовинами; управління вогневими засобами на землі, в повітрі та на морі; оцінка результатів атак противника; доставка повідомлень і даних; пошук та евакуація поранених з поля бою) [23].

Перелік основних класів ЗМТЗ можна доповнити, додавши до неї завдання маршрутизації за допомогою дронів.

Завдання комівояжера з використанням безпілотних літальних апаратів (TSP with Unmanned Aerial Vehicles, TSP-UAV) [24, 25]. Такі місії характеризуються наявністю складу, дрона та показниками вартості. Безпілотнику потрібно відвідати кожну ціль лише один раз, перш ніж повернутися на базу. Метою цієї проблеми є мінімізація загальної вартості шляху. Критеріями виконання можуть бути, наприклад, відстань або час виконання завдання.

Задача комівояжера з використанням декількох безпілотних літальних апаратів (TSP with Multi Unmanned Aerial Vehicles, TSP-MUAV) [26, 27]. Ця проблема є узагальненням проблеми TSP-UAV. Її принципова відмінність полягає в тому, що ціль може відвідувати не один дрон, а кілька. Крім того, кожну ціль може відвідати лише один дрон. Метою цього типу задач є визначення набору маршрутів таким чином, щоб мінімізувати витрати.

Задача комівояжера з використанням безпілотних літальних апаратів та врахуванням вартості маршруту (TSP with Unmanned Aerial Vehicles and the Cost of the Route, TSP-UAVCR). Ця проблема також є узагальненням проблеми TSP-UAV. Основна її 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.

Класична ЗМТЗ (VRP with Unmanned Aerial Vehicles, VRP-UAV) [28, 29, 30, 31, 32, 33, 34] з використанням БПЛА. Необхідно визначити набір маршрутів для флоту безпілотників, які будуть відвідувати ціль. Що стосується класичного ЗМТЗ, то мета цієї задачі – мінімізація загальної вартості маршруту.

ЗМТЗ, що використовує БПЛА та «часові вікна» (VRP with Unmanned Aerial Vehicles and Time Windows, VRPTW-UAV) [35, 36, 37, 38]. Ця проблема є узагальненням проблеми VRP-UAV. Фундаментальна відмінність цього завдання полягає в тому, що відвідувати цілі можна лише протягом заданого періоду часу, який називається «часовим вікном». Це все «Часове вікно» встановлюється індивідуально для кожного об'єкта. Якщо дрон досягає цілі раніше, то має чекати, а якщо пізніше, то має заплатити штраф. Метою задачі також є мінімізація загальної вартості маршруту.

ЗМТЗ з використанням БПЛА та можливістю перезарядки (VRP with Unmanned Aerial Vehicles and Satellite Facilities, VRPSF-UAV) [39]. Ця місія характеризується можливістю скорочення використання безпілотників за рахунок перезавантаження. Метою також є мінімізація витрат на маршрутизацію.

ЗМТЗ, що використовує БПЛА та має обмеження на вантажопідйомність (Capacitated VRP with Unmanned Aerial Vehicles, CVRP-UAV) [40]. Особливістю цієї місії є обмежена пропускна здатність безпілотника. При цьому кожен безпілотник може літати лише на таку відстань, на скільки має достатню потужність, і не може дозаправлятися. Метою цієї проблеми також є мінімізація вартості маршруту.

ЗМТЗ, що використовує БПЛА, має можливість дозаправки, а також обмеження на вантажопідйомність (Capacitated VRP with Unmanned Aerial Vehicles and the Possibility of Refueling, CVRPPR-UAV). Ця місія є розширенням місії CVRP-UAV, але з можливістю повернення на базу для дозаправки. Це дозволяє дрону виконувати кілька польотів.

ЗМТЗ з використанням безпілотними літальними апаратами та багатьма депо (VRP with Unmanned Aerial Vehicles and Multi Depo, MDVRP-UAV). Ця місія дуже

схожа на звичайну MDVRP, за винятком того, що тут як транспортні засоби використовуються БПЛА. Місія характеризується наявністю кількох складів, кількох цілей і кількох безпілотників. Кожен дрон може відвідати лише одну ціль, після чого повинен повернутися на ту ж базу, з якої злетів. Мета задачі - знайти шлях до мети з найменшими витратами.

З класифікації вище ми бачимо, що клас ЗМТЗ, який використовує безпілотники, схожий на клас ЗМТЗ, який не включає безпілотники, але все ж є певні відмінності.

Зробимо підсумки огляду задач маршрутизації транспортних засобів.

Особливо актуальним сьогодні є завдання планування маршруту безпілотного літального апарату. Особливий інтерес до нього виявляють військові відомства різних країн. Створюються різноманітні наукові проєкти, які знаходять нові методи та розробляють способи запуску та посадки дронів, їх інтеграції з літаками, автоматичної дозаправки та безпілотних посадок. Також зверніть увагу на створення маршрутів для дронів, оскільки керування дронами в режимі реального часу може бути ускладнене різними перешкодами, наприклад, глушінням радіозв’язку. Тому весь маршрут операції потрібно визначити заздалегідь, і дрон повинен літати автоматично.

Усі існуючі задачі маршрутизації безпілотників, опубліковані за останні кілька років, є варіантами задачі комівояжера або задачі маршрутизації транспортного засобу.

Тому для розв’язання задачі про знаходження шляху дронів використовуються ті ж методи, що й у цих двох типах задач, а саме генетичний алгоритм, оптимізація потоку частинок, оптимізація колонії мурашок, алгоритм колонії бджіл тощо.

Проблема, розглянута в цьому проєкті, є актуальною, оскільки враховує необхідність обслуговування БПЛА на рухомих базах.

1.2 Методи розв’язання

Метою цього підрозділу є розглянути існуючі підходи до вирішення VRP, зокрема визначити, які з них були найбільш актуальними в останні роки.

Для простоти ми розглядаємо задачі комбінаторної оптимізації (ЗКО) без додаткових обмежень, а саме $D \equiv X$. Треба знайти $x_* = X$ таке, щоб

$$x_* = arg\frac{ext}{x\epsilon X} f(x). \qquad (1.1)$$

Будемо вважати, алгоритм комбінаторної оптимізації (АКО) — це специфічна процедура перетворення заданої підмножини $Z \subset X$ (початкового наближення) у множину $X_* \subset X$:

$$AZ = X_*,$$

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

Для розв’язування ЗМТЗ використовують багато методів, класифікуючи їх за різними ознаками: за точністю, типом використання простору, структурою розрахункової схеми, отриманими рішеннями тощо. Розглянемо дві можливі основні класифікації методів: за точністю та за типом розрахункової схеми.

Обчислювальна складність задач комбінаторної оптимізації

Існує два класи задач: P і NP.

Задачі P-типу — це задачі, які можна розв’язати за поліноміальний час, а задачі NP-типу — це задачі, час розв’язання яких не обмежений поліномами, і лише шляхом обходу всіх варіантів можна знайти точне розв’язання.

Для NP-подібних задач точні алгоритми не підходять, оскільки їх обчислювальна складність занадто велика. Для них використовуються наближені методи: евристика, метаевристика та ін.

Класи алгоритмів комбінаторної оптимізації за точністю

На рисунку 1.3 показано основні методи розв’язування ЗМТЗ за категоріями точності.

Рисунок 1.3 – Класифікація АКО за точністю

Точні АКО знаходять глобальне рішення, для них $X_* \subset Arg\ ext$. Наближений АКО поділяється на алгоритми з апріорною та апостеріорною оцінками точності [41]. Евристика базується на розумних міркуваннях і, наприклад, методи, що включають найближче місто, використовуються для пошуку рішення проблеми комівояжера. Однак такі методи не забезпечують необхідної точності отриманих результатів.

Зазвичай на практиці дослідники поєднують алгоритми оцінки наближеної точності та евристичні алгоритми в простий клас під назвою «наближені алгоритми». Тоді АКО можна розділити на дві категорії: точний алгоритм і наближений алгоритм.

Точні алгоритми можна розділити на загальні методи та спеціальні алгоритми. Зазвичай використовуються такі методи: метод повного перерахування, метод гілок та меж, гілок та відтинань, метод послідовного аналізу варіантів, метод динамічного програмування тощо. Спеціальні алгоритми включають алгоритми, розроблені для конкретних завдань і тому мають вузьке застосування.

Наближені алгоритми можна розділити на сім категорій (рис. 1.4).

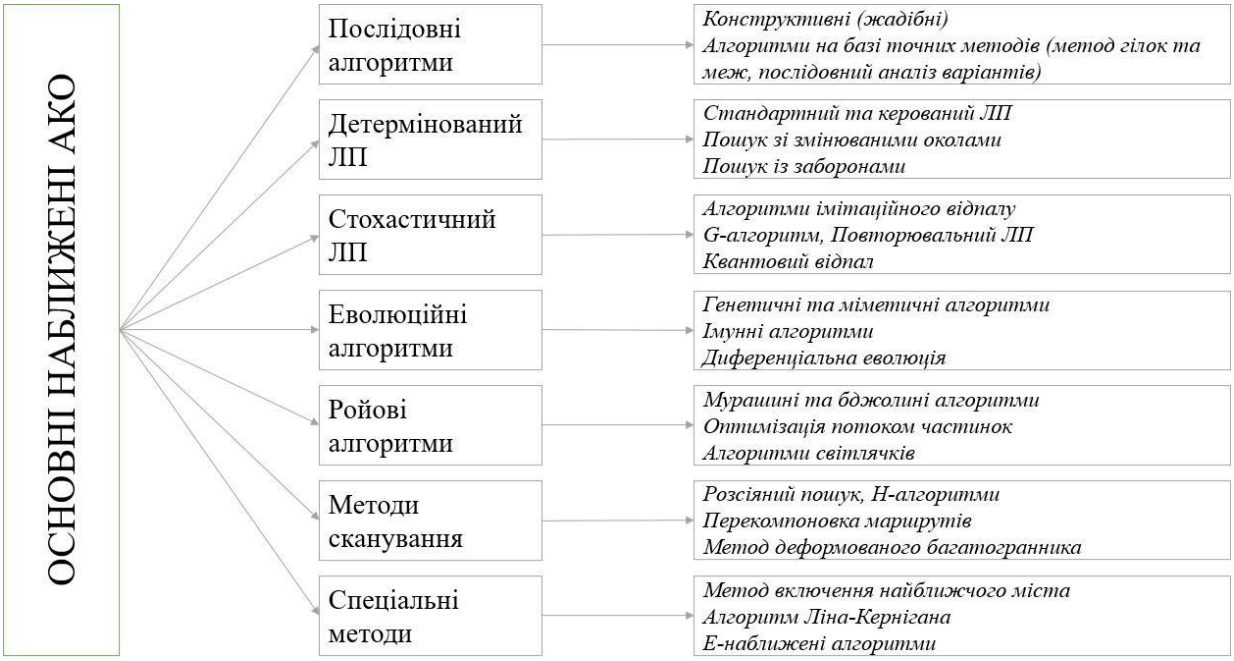


Рисунок 1.4 – Класифікація основних наближених АКО

Важливо відзначити, що в рамках даної класифікації точні алгоритми, такі як гілок та меж, використовуються для створення наближених схем обчислення. Звичайно, сьогодні АКО набагато більше, ніж показано на рисунку 1.4, але, як показує аналіз наукових публікацій у цій галузі, саме ці алгоритми найбільше використовуються при розв’язуванні ЗКО.

У переважній більшості випадків для розв’язування ЗКО використовуються лише наближені алгоритми, що пов’язано з кількома причинами: майже всі ЗКО відносяться до класу NP-складних задач, тому навіть їх точне розв’язування на сучасних комп’ютерах є проблематичним; зазвичай ЦФ ЗКО є багатоекстремальною, тобто існує кілька локальних мінімумів або максимумів; у багатьох прикладних задачах наведені дані мають певну похибку; ідея, покладена в основу розробки наближених АКО, дозволяє створювати алгоритми, які можуть вирішувати не тільки одну задачу, а цілий клас схожих задач.

Класи алгоритмів комбінаторної оптимізації за типом обчислювальної схеми

За типом обчислювальної схеми АКО поділяють на конструктивні (прямі, послідовні) та ітераційні. Нехай у нас є конкретна множина $Y \supseteq X$.

Конструктивні алгоритми - це алгоритми з $Y \supset X$ $(Y \neq X)$, тобто вони працюють у просторі, який є розширенням простору рішень X. Починають «з нуля» або з якогось іншого фрагмента рішення і поступово розробляють повне рішення. Розглянемо приклади таких алгоритмів для різних ЗКО.

Задача комівояжера вирішується алгоритмом, що включає найближче місто, де кожен крок додає до маршруту вершину, відстань якої є найменшою з усіх можливих відстаней. Це називається «перехід до найближчого», посилаючись на жадібний алгоритм.

Для задач квадратичного розподілу характерно, що першими розміщуються об'єкти (елементи) з більш інтенсивним потоком.

Пошук мінімального кістякового дерева здійснюється за допомогою евристики, згідно з якою на кожному наступному кроці побудований фрагмент кістякового дерева містить вершини, які мінімально збільшують його загальну довжину і не утворюють підциклів.

Ітераційні алгоритми – це ті, які обчислюють «повне» рішення, тобто свій простір пошуку $Y \equiv X$, на кожному кроці. Починаючи з деякого $x^0 \in X$, ітераційний алгоритм намагається покращити його крок за кроком:

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

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

Існує два класи ітераційних алгоритмів: траєкторні та популяційні. Ітеративний підхід, який оперує (поточним) рішенням на кожному кроці, називається траєкторним. Ітераційні методи на основі популяції — це ті, які одночасно обчислюють декілька рішень, а не одне рішення в кожній ітерації. Отже, для алгоритму траєкторії $\left|\left|Z\right|\right| = \left|\left|X_*\right|\right| = 1$, для алгоритму популяції $\left|\left|Z\right|\right| > 1$, $\left|\left|X_*\right|\right| > 1$.

1.3 Існуюче програмне забезпечення

Routific Route Optimization API

Routific — це компанія, яка надає програмне забезпечення з API оптимізації маршрутів, що дозволяє розв’язувати VRP [42]. Продуктовий портфель компанії включає наступні варіанти використання:

— доставка посилок;

— вихідне обслуговування;

— логістика розподілу;

— квітковий магазин;

— транспортування медичного обладнання;

— утилізація відходів.

Розробники наполягають на тому, що їхній алгоритм може заощадити до 40 відсотків часу доставки або пального та скоротити час, необхідний для планування маршрутів, на 95 відсотків. Програмний продукт, який зараз використовується більш ніж у 900 містах, може враховувати такі обмеження, як часові вікна, місткість транспортного засобу, години роботи, численні зупинки, відкриті маршрути, різні типи транспортних засобів і заміну водія.

Google OR-Tools

Google OR-Tools [43] — це складний програмний продукт з відкритим кодом, який надає можливість вирішувати проблеми оптимізації. Ця програма надає бібліотеку для вирішення VRP у Java, C#, Python і C++. Крім класичного VRP, цей застосунок дозволяє вирішити проблему обмеження ємності і часового вікна.

AIMMS

AIMMS — це платформа, яка дозволяє створювати та розповсюджувати програми, пов’язані з прийняттям рішень [44]. Середовище розробки поєднує унікальні можливості моделювання та інструменти проєктування, щоб ви могли швидко створювати нові програмні продукти. У рамках своєї платформи AIMMS

надає бібліотеку для вирішення проблеми маршрутизації з обмеженням пропускної здатності (CVRP).

1.4 Висновки до розділу

У першому розділі було розглянуто ЗМТЗ і визначено, що в даний час існує багато різновидів: з обмеженнями по потужності, «часовими вікнами», альтернативними динамічними складами, поверненнями і постачаннями товарів, різними видами транспорту, періодичним маршрутом, з можливістю повторного завантаження і т.д. Тож наведено класифікацію найпоширеніших ЗМТЗ.

При аналізі поточного стану ЗМТЗ виявилося, що в багатьох випадках замість традиційної техніки пропонують використовувати безпілотники. Розвиток сучасних технологій поступово рухається в напрямку того, що безпілотники відіграють ключову роль у різних логістичних сценаріях. На даний момент маршрутні місії БПЛА стали зрілою технологією, яка є лідером у галузі. Аналіз сфери застосування БПЛА показує, що її можна розділити на дві категорії: цивільну та військову. Безпілотники можуть значно скоротити фінансові витрати та час, необхідний для доставки матеріалів, оскільки вони дешевші в обслуговуванні, ніж традиційні транспортні засоби.

Як і звичайні задачі пошуку маршруту, проблеми маршрутизації на основі безпілотних літальних апаратів тепер мають багато різновидів: місії Комівояжера з використанням кількох безпілотників, місії Комівояжера з використанням безпілотників і врахуванням вартості маршруту, та інші. Розглянуто також методи розв’язання ЗКО, проаналізовано обчислювальну складність АКО та виділено основні категорії АКО за точністю та типом обчислювальних схем. Тому за точністю АКО поділяють на точні, орієнтовні (з оцінкою точності) та евристичні, а за типом розрахункової схеми – на конструктивні та ітераційні. В ітераційних алгоритмах також виділяють два інших класи: траєкторії та популяції.

2 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Змістовна постановка задачі

Розглянемо випадок, коли є автомобіль або інший транспортний засіб (ТЗ), який може переміщатися з початкової локації (бази) в іншу локацію з дроном на борту. Перед безпілотником, який може злітати з ТЗ і повертатися до нього (або поруч з ним), стоїть завдання облетіти заданий набір цільових точок (цілей) з урахуванням часових обмежень ресурсів польоту, при цьому кожну ціль потрібно відвідати лише один раз. Поповнення льотного ресурсу (заміна акумуляторної батареї або дозаправка) здійснюється тільки в місці базового розташування транспортного засобу (базування) на його маршруті, який вважається заздалегідь визначеним.

Якщо під час формування курсу потрібне поповнення ресурсів, дрон має повернутися на відповідну базу. У цьому випадку база включається у відповідний відрізок траєкторії дрона з урахуванням часу обслуговування дрона для повного відновлення ресурсів і продовження процесу формування траєкторії дрона.

Тому зроблено наступні припущення.

а) БПЛА відвідує кожну ціль один раз і тільки один раз.

б) транспортні засоби не беруть участь у службових цілях.

в) технічне обслуговування дронів здійснюється на базі (пункті стикування транспортних засобів).

г) опорні позиції на даному маршруті автомобіля вважаються фіксованими, і транспортний засіб відвідує їх послідовно, не повертаючись.

д) встановлено час прибуття транспортного засобу на кожну базу, при цьому транспортний засіб може чекати дрон протягом певного часу.

е) вважається, що резервів для поповнення ресурсів дронів достатньо.

ж) енергоспоживання БПЛА приймається лінійним, тобто можливе перевищення споживання під час зльоту чи посадки БПЛА не враховується.

з) маршрут БПЛА може складатися з сегментів (підциклів), кожен з яких починається і закінчується на вибраній базі маршруту транспортного засобу,

і включає час обстеження для цілей, які не порушують обмеження ресурсу польоту.

2.2 Математична постановка задачі

Розглянемо задачу I, що є базовою.

Задача I (задача з однією базою)

У нас є безпілотний літальний апарат (БПЛА), кілька цілей для обстеження та транспортний засіб на базі B_1. Дрон повинен облетіти цілі, починаючи з бази B_1. У зв'язку з обмеженим польотним ресурсом БПЛА він повинен повернутися на базу для обслуговування (поповнення польотного ресурсу) до закінчення повного обльоту і продовжити політ за цілями.

Завдання полягає у такому визначенні множини підмаршрутів БПЛА, які починаються та закінчуються на базі B_1, та часових інтервалів обслуговування на базі, щоб провести обстеження усіх цілей за мінімальний час з урахуванням обмежених ресурсів польоту БПЛА.

Вхідні дані:

— координати бази B_1;

— координати цілей;

— t – ресурс БПЛА у часовому вимірі (час у польоті без поповнення);

— τ – час відновлення ресурсу (обслуговування) на базі;

— v – швидкість БПЛА.

Проміжні дані:

— D – матриця відстаней як між цілями, так і між цілями та базою;

— C – матриця часів переміщення БПЛА між пунктами.

Вихідні дані:

— кількість підмаршрутів обльоту цілей;

— порядок обльоту цілей в кожному підмаршруті;

— загальний час виконання обстеження цілей;

— часовий розклад поповнення ресурсу.

На рисунку 2.1 зображено графічну ілюстрацію задачі з 8 цілями та один із можливих варіантів обльоту цих цілей дроном.

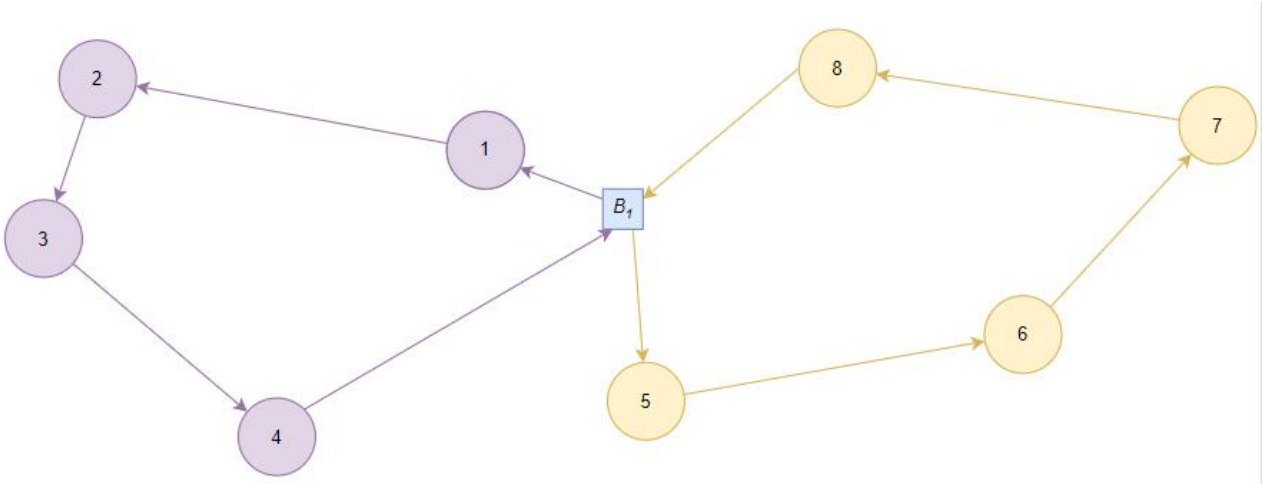


Рисунок 2.1 – Графічна ілюстрація варіанту розв'язку задачі із вісьма цілями та однією базою

Відповідно рисунку 2.1 кількість підмаршрутів дорівнює двом і при цьому БПЛА облітає задані цілі в такій послідовності:$B_1 - 1 - 2 - 3 - 4 - B_1 - 5 - 6 - 7 - 8 - B_1$.

Математична модель задачі I

Дано

$J = \{1,2 \dots n\}$ – множина цілей БПЛА

$\{0\}$ - місце розміщення бази B_1

D_{kl} $(k, l \in \{0,1,2 \dots n\})$ – матриця відстаней як між цілями, так і між цілями та базою.

c_{kl} $(k, l \in \{0,1,2 \dots n\})$ – матриця часів переміщення БПЛА між пунктами k, l $(c_{kl} = \frac{D_{kl}}{v})$.

Необхідно розбити множину J на R упорядкованих підмножин, що відповідають підмаршрутам обльоту цілей:

$$\{J_1 = \left\{j_1^1, j_2^1 \dots j_{n_1}^1\right\} \dots J_R = \left\{j_1^R, j_2^R \dots j_{n_R}^R\right\}$$

таких що

$$\cup_{r=1}^{R} J_r = J \quad (\sum_{r=1}^{R} n_r = n)$$

$$J_i \cap J_j = \emptyset, \text{де } i \neq j$$

Сумарний час кожного підмаршруту повинен бути не більшим ніж ресурс БПЛА у часовому вимірі без поповнення:

$$\mathrm{C}_r = c_{0j_1^r} + \sum_{q=1}^{n_r-1} c_{j_q^r j_{q+1}^r} + c_{j_{n_r}^r 0} \leq t, \text{де } r = 1, .., R$$

Цільова функція

Кількість маршрутів повинна бути мінімальною:

$$z = R \to min$$

Або сума часів усіх маршрутів повинна бути мінімальною:

$$z = (R-1)\tau + \sum_{r=1}^{R} C_r \to min$$

Якщо кількість маршрутів $R = 1$, то $z = \sum_{r=1}^{R} C_r$, оскільки обслуговування БПЛА на базі не відбувається.

Примітка 1. Зауважимо, що тут і далі:

$n_r = |J_r|, r = 1, \ldots, R,$

$\sum_{r=1}^{R} n_r = n.$

Примітка 2. Часовий розклад поповнення ресурсу визначається за результатами розв'язання задачі, а саме з урахуванням кількості підмаршрутів обльоту цілей та порядку обльоту цілей в кожному підмаршруті.

Розглянемо задачу II.

Задача II (задача з синхронізацією БПЛА та ТЗ з двома базами, де одна з баз є основною)

Маємо транспортний засіб, один БПЛА, задану множину цілей для обстеження та місцезнаходження двох баз B_1 і B_2 разом з часом переміщення транспортного засобу між ними. БПЛА стартує з транспортного засобу, що знаходиться на базі B_1, а транспортний засіб при цьому вирушає до бази B_2, тому приземлення чи

обслуговування далі може відбуватися уже лише на базі B_2. Завдання полягає у такому визначенні фрагментів маршруту БПЛА та часових інтервалів синхронного обслуговування на базі B_2, щоб провести обстеження цілей за мінімальний час. Сенс синхронізації полягає у тому, що час переміщення транспортного засобу до бази B_2 має бути не меншим, ніж час проходження першого фрагмента маршруту БПЛА, тобто щоб транспортний засіб прибув до B_2 раніше, ніж БПЛА, і чекав на нього.

Таким чином, задача розбивається на дві підзадачі:

— пошук такого маршруту БПЛА, щоб він дістався до B_2 за час, не менший, ніж час ТЗ в дорозі та максимально використав свій ресурс;

— визначення фрагментів маршруту БПЛА на базі B_2, щоб провести обстеження цілей за мінімальний час.

На рисунку 2.2 зображений один із можливих варіантів обльоту цілей для задачі II. А саме почерговий обліт БПЛА 8 цілей в такій послідовності: $B_1 - 8 - 7 - 6 - 5 - B_2 - 4 - 3 - 2 - 1 - B_2$.

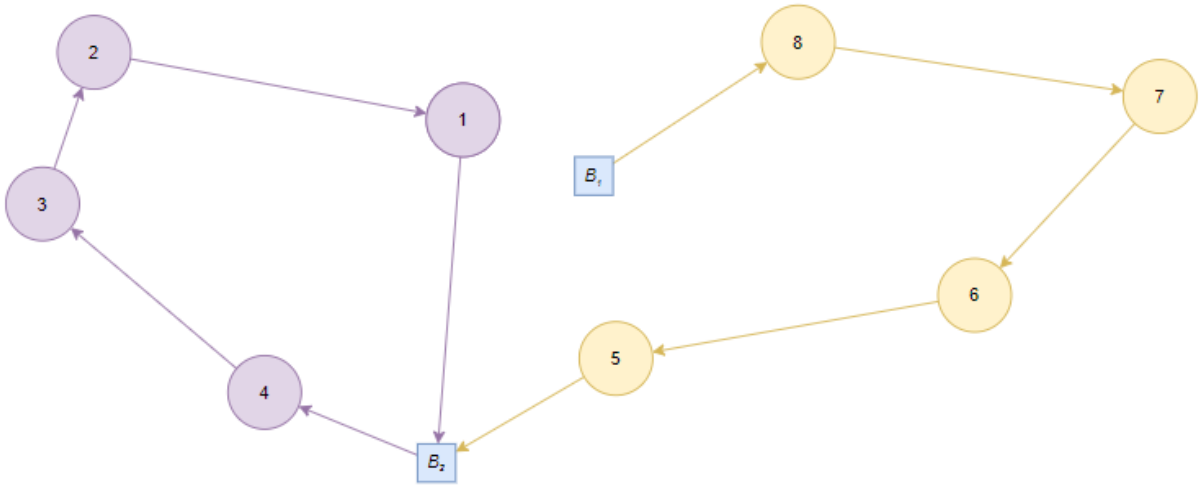


Рисунок 2.2 – Графічна ілюстрація варіанту розв’язку задачі із вісьма цілями та двома базами

Вхідні дані:

— координати баз B_1, B_2;

— t – ресурс БПЛА у часовому вимірі (час у польоті без поповнення);

— τ – час відновлення ресурсу (обслуговування) на базі;

— v – швидкість БПЛА;

— Δ – час переміщення транспортного засобу від бази B_1 до бази B_2.

Проміжні дані:

— D – матриця відстаней як між цілями, так і між цілями та базою;

— C – матриця часів переміщення БПЛА між пунктами.

Вихідні дані:

— кількість підмаршрутів обльоту цілей;

— порядок обльоту цілей в кожному підмаршруті;

— загальний час виконання обстеження цілей;

— часовий розклад поповнення ресурсу.

Математична модель

$J = \{1,2 \dots n\}$ – множина цілей БПЛА

0 - місце розміщення бази B_1, n + 1 – бази B_2

D_{kl} $(k, l \in \{0,1,2 \dots n\})$ – матриця відстаней як між цілями, так і між цілями та базою.

c_{kl} $(k, l \in \{0,1,2 \dots n, n+1\})$ – матриця часів переміщення БПЛА між пунктами k, l $(c_{kl} = \frac{D_{kl}}{v})$.

Необхідно розбити множину J на R упорядкованих підмножин, що відповідають підмаршрутам обльоту цілей:

$$\{J_1 = \left\{j_1^1, j_2^1 \dots j_{n_1}^1\right\} \dots J_R = \left\{j_1^R, j_2^R \dots j_{n_R}^R\right\}$$

таких що

$\cup_{r=1}^{R} J_r = J \quad (\sum_{r=1}^{R} n_r = n)$

$$J_i \cap J_j = \emptyset, \text{де } i \neq j$$

Сумарний час першого підмаршруту повинен бути не більшим ніж час у польоті БПЛА без поповнення:

$$\mathrm{C}_1 = c_{0j_1^1} + \sum_{q=1}^{n_1-1} c_{j_q^1 j_{q+1}^1} + c_{j_{n_1}^1(n+1)} \le t$$

Крім того, час першого підмаршруту БПЛА (з двома базами) повинен бути не меншим за час переміщення транспортного засобу від першої до другої бази:

$$\mathrm{C}_1 = c_{0j_1^1} + \sum_{q=1}^{n_1-1} c_{j_q^1 j_{q+1}^1} + c_{j_{n_1}^1(n+1)} \ge \Delta$$

Сумарний час усіх наступних підмаршрутів повинен бути не більшим ніж час у польоті БПЛА без поповнення:

$\mathrm{C}_r = c_{(n+1)j_1^r} + \sum_{q=1}^{n_r-1} c_{j_q^r j_{q+1}^r} + c_{j_{n_r}^r(n+1)} \le t$, де $r = 2,..,R$,

для підмаршрутів, що починаються і закінчуються в базі B_2.

Цільова функція

Кількість маршрутів повинна бути мінімальною:

$$z = R \to min$$

Або сума часів усіх маршрутів повинна бути мінімальною.

2.3 Опис розроблених алгоритмів

2.3.1 Модифікований алгоритм Дейкстри

У [45] Дейкстра описує алгоритм, який розв’язує проблему найкоротшого шляху з невід’ємними вагами ефективніше, ніж LP. Цей алгоритм тепер називається алгоритмом Дейкстри і добре задокументований. Його недолік полягає в тому, що він не може обробляти графи, що містять ребра з від’ємними вагами, що не зашкодить у нашому випадку, оскільки відстань між точками не може бути від’ємною.

Кожен вузол зберігає два атрибути v. Перший – це довжина найкоротшого шляху від s до v, знайденого на даний момент, а другий – вузол-предка v на цьому шляху. Алгоритм буде ітеративно будувати оптимальний шлях, покращуючи рішення

на кожному кроці. Після цього ви отримаєте найкоротші шляхи від вихідного вузла до всіх інших вузлів на графі.

Алгоритм

Для опису інтерпретації алгоритму необхідні такі визначення:

— A: = набір вузлів v, де було знайдено найкоротший шлях від s до v (оброблений набір вершин);

— X: = набір вузлів v, шлях від s до v залишається невизначеним (набір вершин у черзі на обробку);

— $g'(v)$: = довжина найкоротшого шляху від s до v, знайденого на поточній ітерації (це можна розглядати як оцінку або верхню межу найкоротшого шляху від s до v);

— $g'(s)$: = 0.

Алгоритм ініціалізується такими значеннями:

— $A = \emptyset$;

— $X = \{s\}$.

Повторюйте наступні кроки алгоритму, поки X не дорівнюватиме порожньому набору:

а) Знайдіть таку вершину v з множини X, що

$g'(v) := \min_{u \in X} g'(u)$, (1.2)

видалити v з X та додати до A.

б) Для кожної вершини u, для яка виконує $(v, u) \in E$ (кожна суміжна вершина v):

1) Якщо A містить u, нічого не робіть;

2) Якщо X містить u, і якщо $g'(v) + w(v, u) < g'(u)$, то нехай $g'(u) = g'(v) + w(v, u)$ та $parent(u) = v$;

3) Якщо u не входить ні в A, ні в X, додайте u до X і встановіть $g'(u) = g'(v) + w(v, u)$ та $parent(u) = v$.

Після завершення роботи алгоритму всі вузли, доступні з s, будуть у A (включаючи t), і проблему буде вирішено. Щоб заощадити час на обчислення, алгоритм можна зупинити, щойно t буде додано до A, оскільки в цей момент найкоротший шлях до t буде знайдено.

Алгоритмічна ефективність

Алгоритм Дейкстри завжди знаходить оптимальний шлях. На рис. 2.3 показано, як на прикладі графу набір A розширюється навколо вихідного вузла s, тоді як кожен вузол всередині межі знаходиться ближче до s, ніж будь-який вузол поза межею. А продовжує розширюватися, поки цільовий вузол не увійде до А.

Ми визначаємо v_k як вузол, доданий до A на кроці k. Використовуючи індукцію, ми покажемо, що v_k - це вузол, найкоротший шлях для якого $g(v_k)$ є найменшим для всіх вузлів, що не належать А.

Це тривіально для кроку 1, оскільки s додається до А і $g(s) = 0$.

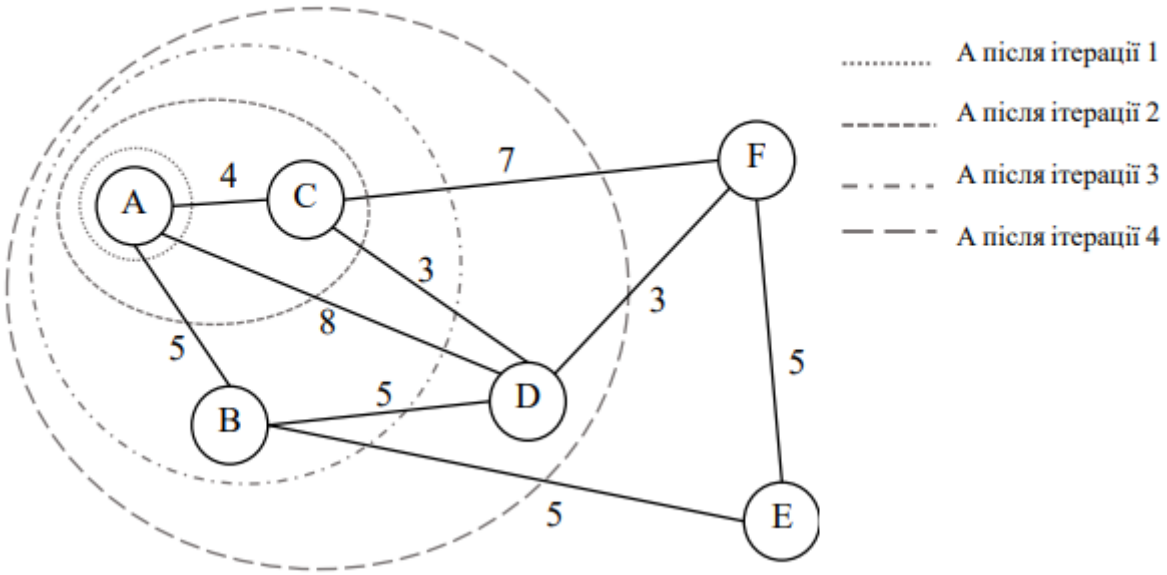


Рисунок 2.3 - Розширення множини A в алгоритмі Дейкстри

Припустимо, що ця умова зберігається до деякого кроку k, після якого будь-який вузол не в A буде далі від s, ніж будь-який вузол в A. Обов'язково всі сусіди всіх вузлів у наборі A будуть у наборі X; будь-який вузол, який не знаходиться ні в A, ні в

X, знаходиться далі від s, ніж принаймні один вузол у X, тому що він має пройти через вузол A, щоб досягти s. Це означає, що вузол v_{k+1}, який потрібно додати до A на кроці k+1, буде в X і, таким чином, буде знайдений алгоритмом.

Оскільки це працює для k = 1, це доводить, що це також працює для будь-якого $k \geq 1$ або будь-якого кроку алгоритму. Оскільки найкоротший шлях до v_k коротший за будь-який вузол поза A, цей шлях не містить жодних вузлів поза A, тоді знайдений найкоротший шлях є найкоротшим шляхом, або $\hat{g}(v_k) = g(v_k)$. Це означає, що v_k можна додати до A. Формальний доказ алгоритму Дейкстри наведено нижче.

Коли вузол додається до A, знайдено найкоротший шлях, алгоритм більше не розглядатиме цей вузол. Щоб довести, що шлях Дейкстри знаходить справжній найкоротший шлях для кожного вузла, нам потрібно довести для кожної ітерації k, що жоден шлях до v_k не є коротшим за $\hat{g}(v_k)$, або $\hat{g}(v_k) = g(v_k)$. Для цього використовується доведення леми 1, наведене в [46].

Лема 1

Для будь-якого оптимального шляху P від s до деякого вузла $u \notin A$, існує вузол $v \in X$ на шляху P, де $\hat{g}(v) = g(v_k)$.

Лема 1 стверджує, що для кожної ітерації існує вузол $v_k \in X$, де $\hat{g}(v_k) = g(v_k)$, поки існує вузол $u \notin A$. Коли цей вузол u не існує, X дорівнює порожньому набору, і алгоритм зупиняється. Це доводить, що для кожного вузла, доданого до A, знайдено найкоротший шлях, тому для кожного вузла $v \in A$ алгоритм Дейкстри знаходить найкоротший шлях від s до v.

Алгоритм модифікується наступним чином:

— вершини графу, що є базами, можуть бути відвідані необмежену кількість разів;

— враховані обмеження на ємність БПЛА та необхідність її поновлення;

— враховані обмеження, пов’язані з часовими вікнами (транспортний засіб має прибути до бази B_2 не пізніше БПЛА).

На рисунку 2.4 зображено графічну інтерпретацію алгоритму Дейкстри.

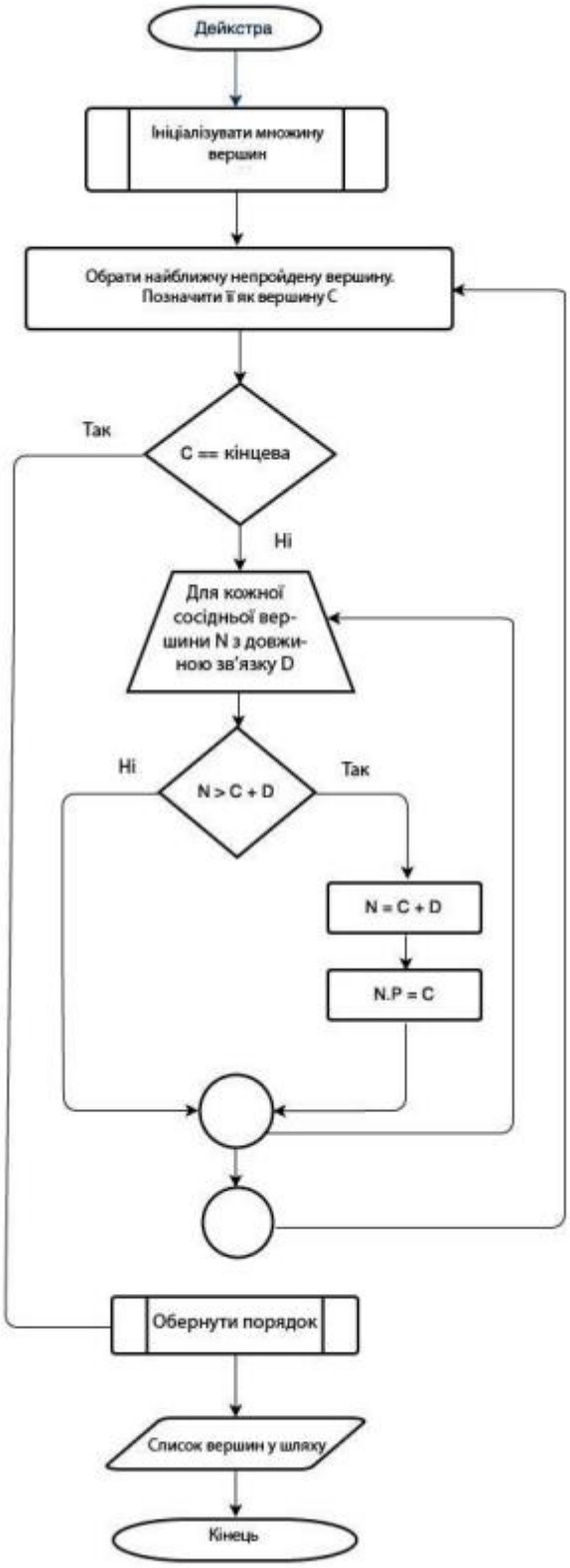


Рисунок 2.4 – Блок-схема алгоритма Дейкстри

2.3.2 Алгоритм на основі алгоритму оптимізації мурашиними колоніями

У всіх класичних алгоритмах оптимізації колонії мурах кожна мураха отримує початкове місто. Починаючи з цього міста, мурахи вибирають наступне місто за правилами алгоритму. Відвідавши всі клієнтські міста один раз, мураха повертається до вихідного міста. Мурахи можуть рухатися одночасно або послідовно. Кожна мураха відкладає на своєму шляху певну кількість феромону. Кількість феромону залежить від якості шляху мурахи: коротші шляхи зазвичай дають більше феромону. Відкладені феромони випаровуються.

Ці алгоритми використовують різні правила, щоб вибрати наступне місто для переміщення, випаровування та зберігання феромону.

Кошмаром кожного алгоритму оптимізації є назавжди застрягти в якомусь локальному оптимальному циклі.

Для різних алгоритмів оптимізації мурашиних колоній були розроблені різні методи обходу таких циклів.

Спочатку кожен шлях між містами має деяку початкову кількість феромонів. Кожна мураха починає з випадково призначеного міста і переходить від одного міста до іншого, поки не відвідає всі міста рівно один раз. З останнього відвіданого міста мураха повертається до початкового міста.

Мураха в місті i вибирає наступне місто для відвідування, обчислюючи ймовірність:

$$p(c_{ij}|s^p) = \frac{{\tau_{ij}}^{\alpha} * {\eta_{ij}}^{\beta}}{\sum_{c_{ij} \in N(s^p)} {\tau_{il}}^{\alpha} * {\eta_{il}}^{\beta}}, \forall c_{ij} \in N(s^p), \qquad (1)$$

де

— s^p - частковий розв'язок;

— N - множина всіх шляхів від міста i до всіх сусідніх міст, які ще не були відвідані мурахою;

— c_{ij} - шлях від міста i до міста j;

— p - ймовірність;

— τ_{ij} - кількість феромону на шляху;

— η_{ij}- евристичний фактор, зазвичай , де d_{ij} - відстань між містами i та j, Q - константа, $\eta_{ij} = \frac{Q}{d_{ij}}$;

— α і β - параметри алгоритму.

Мурахи порівнюють $p(c_{ij}|s^p)$ для кожного j, який потрібно передати деякому випадковому числу $n_j \subset [0,1)$. Якщо $p(c_{ij}|s^p) \geq n_j$, мураха негайно переміщується до міста j. Це означає, що мурахи не завжди вибирають шлях із найбільшою кількістю феромонів, що зменшує ймовірність увійти в локальну петлю.

Імовірність того, що умова $p(c_{ij}|s^p) \geq n_j$ ніколи не буде виконана, завжди дуже і дуже мала. Тому в цій роботі використовується модифікація. Для даного міста i все ще генерується випадкове число $n_j \subset [0,1)$, але воно порівнюється з ковзною сумою $\sum_j p(c_{ij}|s^p)$. Сума оновлюється з кожним новим обчисленням $p(c_{ij}|s^p)$. Перше j, яке виводить мураха з $n_j \leq \sum_j (c_{ij}|s^p)\, p$ є індексом наступного міста, до якого потрібно переїхати. Насправді ця модифікація була доречною лише на ранніх етапах тесту, коли феромони, що осідали на шляху подорожі, перебували в безпосередній близькості один від одного. Під кінець тесту мурахи зазвичай проходять максимальний шлях p. Також, виходячи з умови задачі алгоритм був модифікований так, що всі мурахи розпочинають свій шлях з одного міста (бази).

Мурахи подорожують одночасно. Це означає, що корекції феромонів не буде, доки всі мурахи не повернуться до початкового міста.

Загальна схема алгоритмів оптимізації мурашиними колоніями наведена на рисунку 2.5.

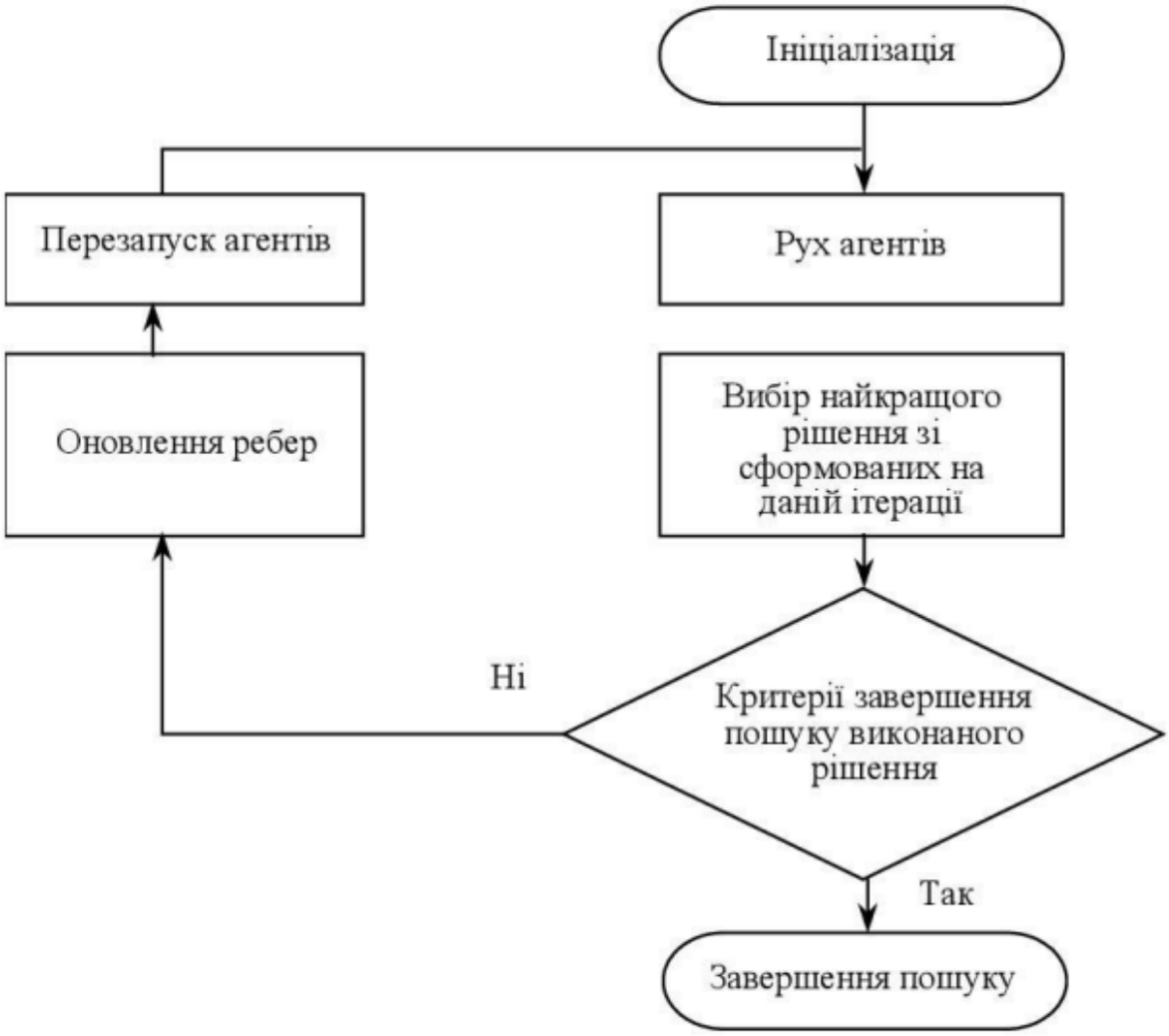


Рисунок 2.5 – Схема роботи алгоритму оптимізації мурашиними колоніями

Обчислювальна схема алгоритмів оптимізації мурашиними колоніями наведена на рисунку 2.6.

```
Ініціалізація параметрів алгоритму α, β, e, Q, τ0.
m = n          {Кількість мурах дорівнює кількості міст}
For i = 1 to n
   For j = 1 to n        {Для кожного ребра}
      If i <> j
            η(i, j) =1/D(i, j)    {Видимість}
            τ(i, j) = τ0     {Феромон}
      Else t(i, j) = 0
      End
   End
End
For k = 1 to m
  Розмістити мураху k у випадково обране місто.
End
Обрати умовно найкоротший маршрут T+ та обчислити його довжину L+.
{Головна програма}
For t = 1 to tmax          {Кількість ітерацій алгоритму}
   For k = 1 to m      {Для кожної мурахи}
      Побудувати маршрут T^k(t) за правилом (1) та обчислити його довжину L^k(t).
   End
   If «Кращий розв'язок знайдено»
     Оновити T+ та L+.
   End
   For i = 1 to n
      For j = 1 to n        {Для кожного ребра}
         Оновити сліди феромону за правилом (2).
      End
   End
End
Вивести найкоротший маршрут T+ та його довжину L+.
```

Рисунок 2.6 – Обчислювальна схема алгоритмів оптимізації мурашиними колоніями [49]

Оновлення феромонів

Феромони на шляхах, якими з'єднані міста,мурахи оновлюють за наступною формулою:

$$\tau_{ij} \leftarrow \tau_{ij} + \sum_{k=1}^{m} \Delta {\tau_{ij}}^{k},$$

де

— m - кількість мурах;

— $\Delta {\tau_{ij}}^{k} = \frac{Q}{L_k}$, якщо мураха **k** пройшла шлях ${c_{ij}}^{k}$ між містами і та j; Q — константа, а L_k - довжина шляху k-ї мурахи;

— інакше $\Delta\tau_{ij}{}^{k} = 0$.

Існують дуже різні пропозиції щодо значень α, β, ρ, початкового значення феромону та кількості мурах (ρ – коефіцієнт випаровування, див. нижче). Деякі джерела рекомендують α = 1, β = 2, ρ = 0,5 і початковий феромон $t = \frac{1}{m}$, де m – кількість міст. Ви можете експериментувати з цими значеннями для кращих результатів.

Кількість мурах, здається, не має значення, поки є достатньо випробувань, але в даній роботі було досліджено, що кількість мурах напряму впливає на якість розв'язку.

Випаровування

Після того, як подорож *n* завершена всіма мурахами, до всіх шляхів між містами буде застосовано випаровування:

$$\tau_{ij}{}^{n} \leftarrow (1 - \rho) * \tau_{ij}{}^{n} \qquad (3)$$

Коефіцієнт випаровування $\rho \subseteq (0; 1]$.

Виникає питання, коли доречно застосовувати глобальне випаровування: після того, як усі мурахи завершили свою подорож, чи після того, як усі феромони були оновлені на краях графіка (шляхи між точками)? Випаровування перед оновленням є, по суті, посиленням феромонів оновлення. У літературі деякі автори застосовують множник (1 - ρ) до всіх феромонів, а інші ні.

Було визначено, що виникненню локальних неоптимальних петель іноді запобігає випаровування після оновлення, тому в цій роботі глобальне випаровування застосовується в останню чергу до країв графіка, після оновлення феромонів.

Модифікації для даного алгоритму аналогічні до модифікацій попереднього алгоритму Дейкстри, а саме:

— вершини графу, що є базами, можуть бути відвідані необмежену кількість разів;

— враховані обмеження на ємність БПЛА та необхідність її поновлення;

— враховані обмеження, пов'язані з часовими вікнами (транспортний засіб має прибути до бази B_2 не пізніше БПЛА.

2.4 Висновки до розділу

У рамках цього розділу визначено змістовні постановки задач і описано Задачу II, дослідження якої є головною метою проєкту. Щоб отримати загальне уявлення про процес розв’язання, наведено математичну модель Задачі I, що є підзадачею Задачі II, та описи вхідних, проміжних і вихідних даних, які є основою для обраної предметної області.

Також описано розроблені алгоритми розв’язання Задачі II, а саме: алгоритм, модифікація алгоритму Дейкстри, і алгоритм, заснований на методі оптимізації колонії мурашок.

Обидва алгоритми були розроблені з урахуванням умови Задачі, а саме:

— вершини графу, що є базами, можуть бути відвідані необмежену кількість разів;

— враховані обмеження на ємність БПЛА та необхідність її поновлення;

— враховані обмеження, пов’язані з часовими вікнами (транспортний засіб має прибути до бази B_2 не пізніше БПЛА.

3 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Вимоги до системи

Для виконання всіх цілей розробки, що наведені в додатку А, необхідно записати та проаналізувати всі вимоги до системи.

На рисунку 3.1 показано існуючі типи системних вимог, які можна згрупувати в такі категорії:

— бізнес-вимоги - опишіть бізнес-вимоги в цілому;

— потреби користувача - опишіть, чого користувачі можуть досягти за допомогою програмного продукту. Як правило, вимоги групи документуються у формі історій користувачів, випадків використання або сценаріїв;

— вимоги до продукту – опишіть, як має працювати система, щоб задовольнити потреби бізнесу та користувачів. Вони включають функціональні та нефункціональні вимоги.

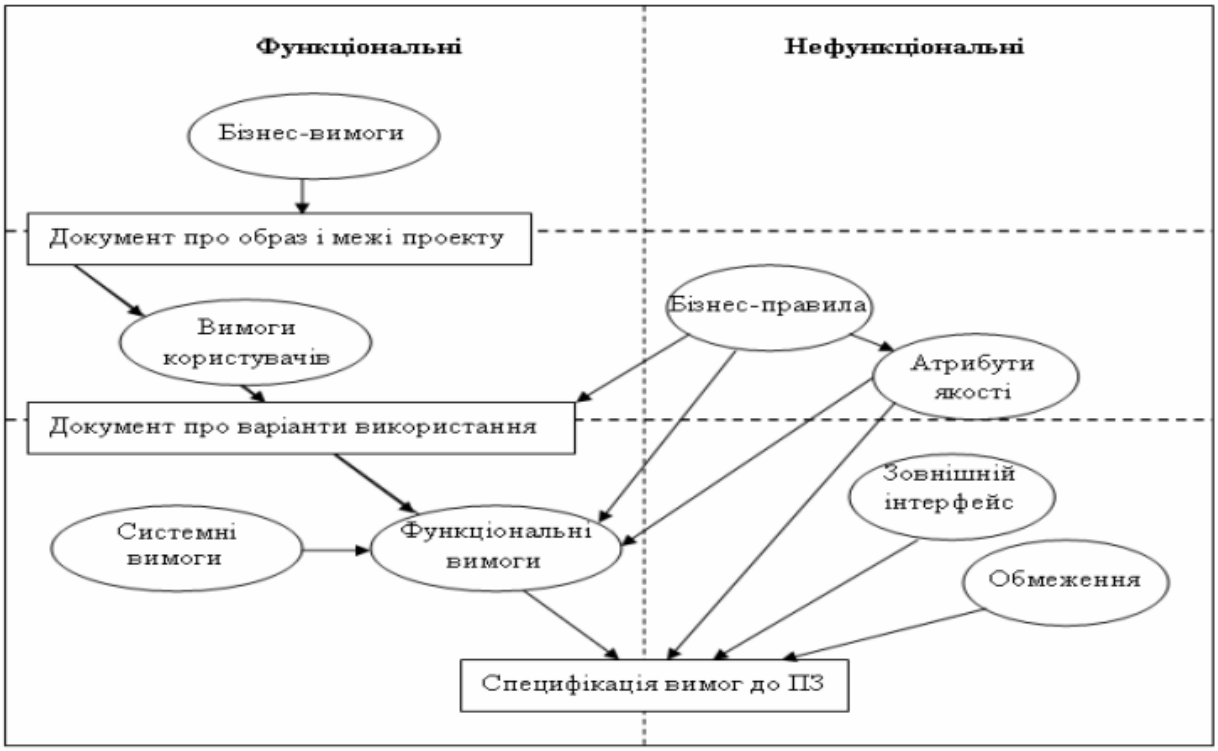


Рисунок 3.1 – Види вимог

3.2 Функціональні вимоги

Запис функціональних вимог має ряд важливих переваг, а саме:

— клієнти та розробники мають єдиний погляд на завдання, що допомагає уникнути непорозумінь та економить час спілкування;

— проєкти стають більш передбачуваними, що дозволяє точніше оцінювати час розробки.

Функціональні вимоги мають бути простими, зрозумілими та однозначними. У Таблиці 3.1 представлені функціональні вимоги розроблюваної системи у вигляді користувацьких історій.

Таблиця 3.1 – Опис функціональних вимог

Користувацька історія	Функціональні вимоги
Як користувач, я хочу мати можливість вирішити задачу.	Система повинна забезпечувати можливість введення параметрів задачі.
	Система повинна забезпечувати можливість обрати алгоритм.
	Система повинна забезпечувати можливість імпортувати координати баз та цілей з файлу CSV.
	Система повинна забезпечувати можливість налаштувати випадкову генерацію базових і цільових координат шляхом визначення числових обмежень для згенерованих даних.
	У разі введення непідтримуваних даних система повинна повідомити користувача.
	Система має відобразити рішення проблеми у вигляді списку підшляхів.

	Система повинна графічно відображати введені координати баз та цілей.
	Система повинна графічно відобразити рішення задачі.
Як користувач, я хочу мати можливість переглядати, зберігати та видаляти вирішені задачі.	Система повинна забезпечувати можливість зберігати задачі та їх розв’язок після їх вирішення
	Система повинна забезпечувати можливість перегляду збережених завдань, тобто всіх вхідних і вихідних даних.
	Система повинна забезпечувати можливість експорту вхідних даних завдання у форматі csv.
	Система повинна забезпечувати можливість експорту вихідних даних збережених завдань у формат csv.
	Система повинна графічно відображати координати збережених цілей і баз задачі.
	Система повинна графічно відобразити рішення задачі.
	Система повинна забезпечувати можливість видалення збережених завдань.
Як користувач, я хотів би мати можливість проводити експерименти, порівнюючи продуктивність алгоритмів.	Система повинна забезпечувати можливість налаштування випадкової генерації базових і цільових координат.
	Система повинна забезпечувати можливість введення параметрів місії.

	У разі введення непідтримуваних даних система повинна повідомити про це користувача.
	Система повинна представляти експериментальні результати в табличній формі.
Як користувач, я хотів би мати можливість експериментально вивчати одну задачу чи алгоритм.	Система повинна забезпечувати можливість введення вхідних параметрів завдання.
	Система повинна забезпечувати можливість імпорту вхідних координат бази та цілі у форматі csv.
	Систему слід налаштувати для випадкового генерування вхідних координат для цілей і баз.
	Система повинна мати можливість встановити числове значення розміру кроку, на який будуть змінюватися вибрані параметри задачі на кожній ітерації.
	У разі введення непідтримуваних даних система повинна повідомити користувача.
	Система повинна забезпечувати можливість перегляду результатів експериментів у графічній формі.
Як користувач, я хотів би мати можливість переглядати, зберігати та видаляти експерименти.	Система повинна забезпечувати можливість збереження експерименту після його завершення.

	Система повинна забезпечувати можливість перегляду проведених експериментів, тобто всіх вхідних і вихідних даних.
	Система повинна забезпечувати можливість експорту вхідних даних завдання у форматі csv.
	Система повинна забезпечувати можливість експорту необроблених даних збережених експериментів у форматі csv.
	Система повинна графічно відобразити результати збереженого експерименту.
	Система повинна забезпечувати можливість видалення збережених експериментів.
Як користувач, я хотів би мати можливість переглядати всі збережені експерименти та задачі.	Система повинна забезпечувати можливість переглядати всі збережені завдання та експерименти в списку.
	Система повинна надавати можливість фільтрувати список збережених проєктів, щоб показувати лише збережені завдання або лише експерименти.
	У системі має бути передбачена можливість пошуку збережених об'єктів за назвою.
	Система повинна забезпечувати можливість видалення збережених об'єктів.

3.3 Нефункціональні вимоги

Нефункціональні вимоги визначені менш чітко, ніж функціональні. Їх мета полягає в тому, щоб створити обмеження на програмні продукти з точки зору швидкості, масштабованості, надійності, переносимості, безпеки тощо.

В рамках роботи, що виконується, нефункціональними вимогами системи є те, що швидкість реакції інтерфейсу застосунку на дії клієнта не повинна бути вищою за задану межу.

3.4 Опис бізнес-логіки системи

Сценарії використання системи, що була розроблена, наведені в додатках В та Г. Вони доводять до відома, що робить система, але не надають подробиць того, як вона це робить. Такі сценарії корисні як клієнтам, так і розробникам. З точки зору клієнта, це корисний інструмент, щоб побачити, чи правильно розробник розуміє всі вимоги. Водночас для розробників це зручне та структуроване представлення потоку подій, які мають відбуватися в програмному продукті. На рисунку 3.2 показано як проводяться експерименти за допомогою діаграми діяльності.

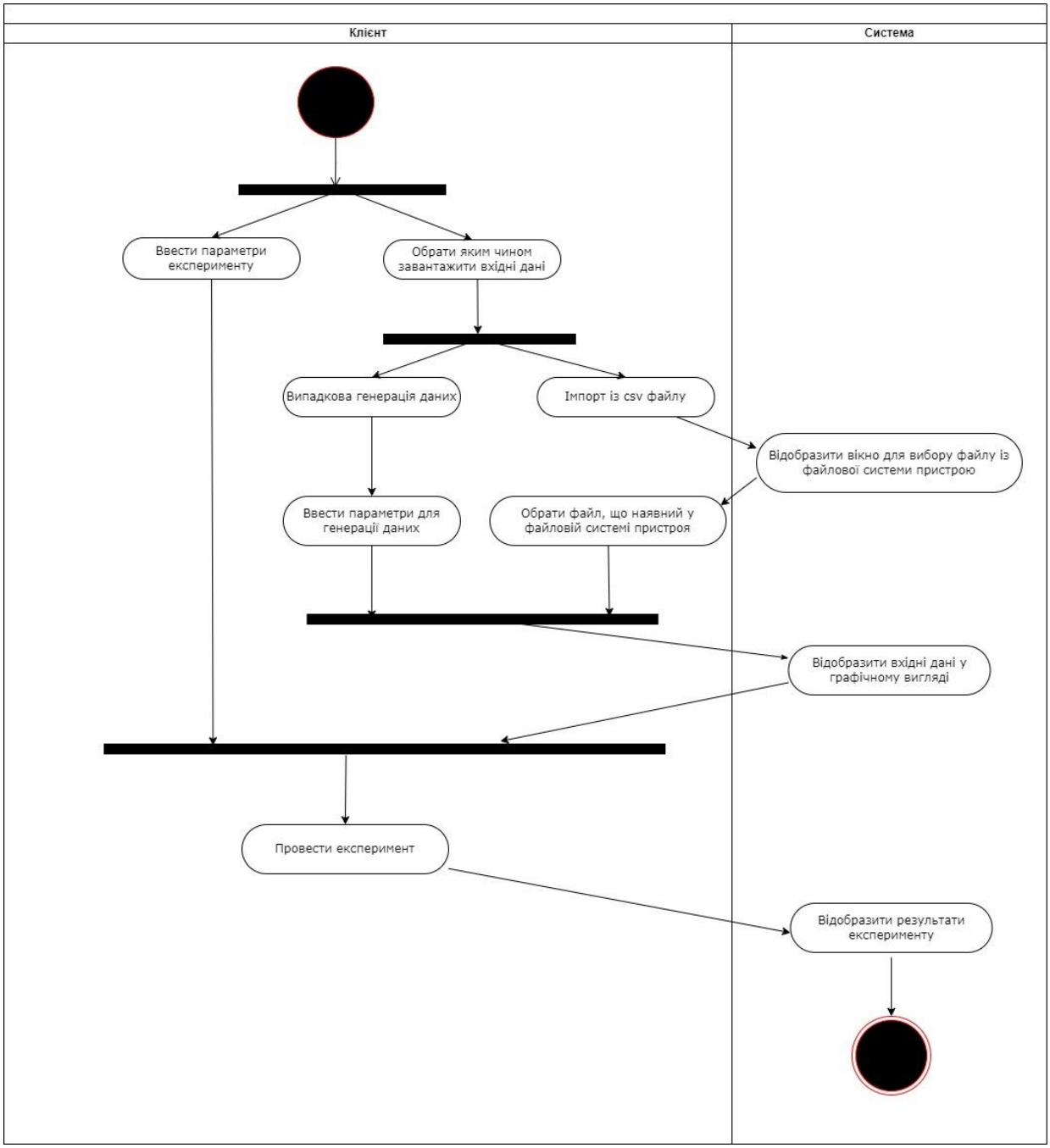


Рисунок 3.2 – Діаграма діяльності проведення експериментів

3.5 Вибір та обґрунтування елементів та технологій

Після того, як всі вимоги визначені та записані, необхідно, використовуючи отримані знання, вибрати технології для розробки системи.

В рамках роботи, що виконується, було вирішено реалізувати інтерфейс програмного продукту за допомогою ігрового двигуна Unity. Такий вибір значною

мірою зумовлений необхідністю графічного представлення рішення задачі. Двигун широко розповсюджений і на даний момент має більше 2,5 мільйонів зареєстрованих розробників з різних кутків світу [47]. Він має наступні переваги:

— підтримка великої кількості різних платформ;

— здатність працювати з різними видами графіки;

— безкоштовність для некомерційних проєктів;

— сильна спільнота;

— вбудовані інструменти для аналізу дій користувачів.

Оскільки C# є основною мовою розробки для Unity, було вирішено взяти за основу саме цю мову програмування. C# - це сучасна мова програмування, від компанії Microsoft, яку застосовують щоб розплутати безліч різноманітних завдань. На рисунку 3.3 показано основні переваги C#.

Рисунок 3.3 – Основні переваги мови програмування C#

Для зберігання даних використовується база даних SQLite. SQLite — це бібліотека, що містить реалізацію механізму SQL. Це база даних, яка не потребує попереднього налаштування адміністратором і не має жодних залежностей від третіх сторін. Крім того, вона безсерверна, і дані зберігаються в одному файлі на диску з файловою системою.

На рисунку 3.4 показано, як програма зазвичай підключається до такої бази даних. Транзакції SQLite сумісні з ACID. Отже, враховуючи вищесказане, SQLite добре підходить для цього проєкту.

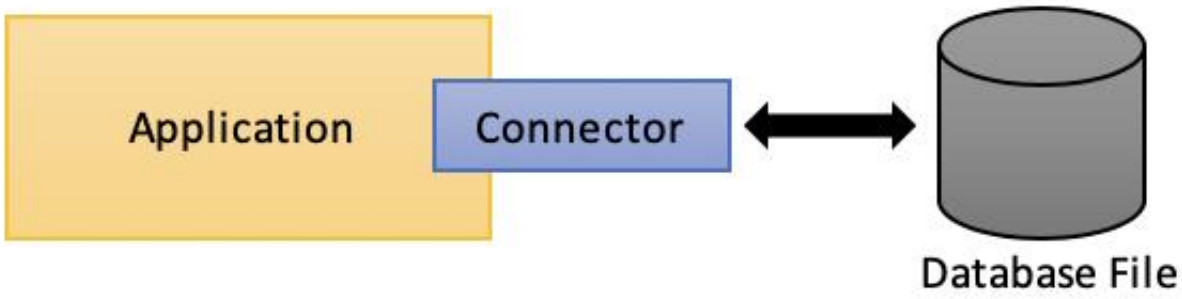


Рисунок 3.4 – Схема комунікації між застосунком та базою даних SQLite

Щоб коректно розробити архітектуру системи звернемось до діаграм послідовності, що наведені у додатках Е, Ж та И.

3.6 Архітектура системи

Схема компонентів наведена у додатку Д. Далі у таблиці 3.2 Розглянемо більш детально усі компоненти системи.

Таблиця 3.2 - Компоненти системи

Назва компонента	Основні функції
SimpleUITable	Відображення таблиці в інтерфейсі користувача
SimpleFileBrowser	Відобразити вікно вибору файлу з файлової системи пристрою, який використовує користувач
RandomDataGenerator	Випадково генеруйте координати цілі та опорні координати
CSVFileManager	Експортуйте та імпортуйте файли з розширенням .csv

AlgoSolutionLib	Містить реалізації алгоритмів розв’язування задач і методів проведення експериментів
DbService	Взаємодіяти зі сховищем збережених завдань і експериментів

Після короткого опису обов'язків компонентів системи, зупинимося на таких модулях, як AlgoSolutionLib і RandomDataGenerator більш детально, адже саме ці компоненти відіграють найважливішу роль у роботі даного програмного продукту.

Модуль RandomDataGenerator, що відповідає за генерацію випадкових вхідних даних, складається з методу generator, функціонал якого описаний у таблиці 3.3.

Таблиця 3.3 – Опис функціоналу методу generator

Назва параметра	Тип параметра	Опис
lowLineX	double	Параметр визначає нижню границю для генерації випадкових вхідних даних параметра x.
upLineX	double	Параметр визначає верхню границю для генерації випадкових вхідних даних параметра x.
lowLineY	double	Параметр визначає нижню границю для генерації випадкових вхідних даних параметра y.
upLineY	double	Параметр визначає

		верхню границю для генерації випадкових вхідних даних параметра y.
ammount	int	Параметр визначає кількість точок, що будуть згенеровані.
outList	List<point>	Параметр, що повертається функцією generator. Являє собою список об’єктів структури point, що описана нижче.
tempX	double	Допоміжний параметр, що використовується для генерації випадкових вхідних даних.
tempY	double	Допоміжний параметр, що використовується для генерації випадкових вхідних даних.
ScaleX	double	Допоміжний параметр, що використовується для генерації випадкових вхідних даних.
ScaleY	double	Допоміжний параметр, що використовується для

		генерації випадкових вхідних даних.
rnd	Random	Параметр, що відповідає за генерацію випадкових чисел. Використовує в якості параметру розподілу випадкових значень поточний час.

Структура даних point, яку повертає метод generator модуля RandomDataGenerator описана в таблиці 3.4.

Таблиця 3.4 – Структура даних point, яку повертає метод generator

Назва параметра	Тип параметра	Опис
x	double	Параметр структури даних point, що відповідає за значення координати точки x.
y	double	Параметр структури даних point, що відповідає за значення координати точки y.
visited	bool	Параметр структури даних point, що надає інформацію про те, чи відвідана точка. За замовчуванням

		встановлено значення false.

Модуль AlgoSolutionLib, що відповідає за роботу алгоритмів, поділений на дві бібліотеки: Deykstra та AntAlgoritm, опис основних класів яких наведено у таблиці 3.5 та таблиці 3.6 відповідно.

Таблиця 3.5 – Опис основних класів бібліотеки Deykstra

Назва класу	Опис
Dijkstra	Клас, в якому реалізований пошук найкоротшого шляху за допомогою модифікованого алгоритму Дейкстри.
Graph	Клас описує граф вцілому. Включає в себе методи додавання вершин та ребер.
GraphEdge	Клас, що описує ребро графу
GraphVertex	Клас описує вершину графу. Є можливість додати список зв’язаних вершин.
GraphVertexInfo	Допоміжний клас для опису інформації про вершини графу.

Таблиця 3.6 – Опис основних класів бібліотеки AntAlgoritm

Назва класу	Опис
AntCity	Клас представляє точку на маршруті
AntRoute	Клас представляє рішення для мурахи – маршрут, який буде починатися бази і

	проходить через усі точки, за потреби повторно відвідує базу, далі повертається до кінцевої бази.
Ant	Клас описує алгоритм проходження маршруту для мурахи
AntColonyOptmz	Клас описує методи для застосуваня алгоритму оптимізації мурашиними колоніями для кожної мурахи
DriverAnt	Основний клас, в якому зберігається масив точок та баз маршруту та результати роботи програми

Опис ключових функцій для реалізації алгоритму оптимізації мурашиними колоніями наведено в таблиці 3.7.

Таблиця 3.7 – Опис ключових функцій бібліотеки AntAlgoritm

Назва функції	Опис
call()	Метод call() класу Ant будує маршрут для кожної мурахи
adjustPheromoneLevel ()	Метод adjustPheromoneLevel () класу Ant регулює рівень феромону після кожного пройденого мурахою маршруту
getY ()	Метод int getY () класу Ant будує оптимальний маршрут на основі ймовірності переходу між містами
getTransitionProbabilities ()	Метод getTransitionProbabilities класу

	Ant обчислює ймовірність переходу мурахи між містами залежно від рівня феромону

Діаграма класів для алгоритму на основі оптимізації мурашиними колоніями наведена у додатку К.

3.7 Розробка бази даних

Реляційна схема бази даних наведена у Додатку Б. У таблицях 3.8 - 3.12 представлений опис таблиць бази даних.

Таблиця 3.8 – Опис таблиці Problems

Назва поля	Тип поля	Опис
Id (PK)	INTEGER	Унікальна ознака об’єкту (первинний ключ)
InputCoordinatedFilePath	TEXT	Шлях до вхідного файлу
ParameterSetId	INTEGER	Зовнішній ключ до таблиці ParameterSets

Таблиця 3.9 – Опис таблиці Algorithms

Назва поля	Тип поля	Опис
Id (PK)	INTEGER	Унікальна ознака об’єкту (первинний ключ)
AlgorithmName	TEXT	Ім’я алгоритму
AlgorithmDescription	TEXT	Алгоритмічний опис

Таблиця 3.10 – Опис таблиці Solutions

Назва поля	Тип поля	Опис
Id (PK)	INTEGER	Унікальна ознака об’єкту (первинний ключ)
Name	TEXT	Ім’я розв’язку задачі
Description	TEXT	Опис розв’язку задачі
OutputSolutionFilePath	TEXT	Шлях до файлу, що містить знайдене рішення задачі
CreatedAt	TEXT	Дата збереження розв’язку задачі
ProblemId (FK)	INTEGER	Зовнішній ключ до таблиці Problems
AlgorithmId (FK)	INTEGER	Зовнішній ключ до таблиці Algorithms

Таблиця 3.11 – Опис таблиці ParameterSets

Назва поля	Тип поля	Опис
Id (PK)	INTEGER	Унікальна ознака об’єкту (первинний ключ)
DroneSpeed	REAL	Швидкість дрона
DroneCapacity	REAL	Місткість дрона
RechargeSpeed	REAL	Час обслуговування дрона
AutoTime	REAL	Час пересування транспортного засобу між базами

Таблиця 3.12 – Опис таблиці Experiments

Назва поля	Тип поля	Опис
Id (PK)	INTEGER	Унікальна ознака об’єкту (первинний ключ)
Name	TEXT	Назва дослідження
Description	TEXT	Опис дослідження
MinX	REAL	Мінімальне значення x-координати всіх задач в дослідженні
MaxX	REAL	Максимальна x-координата всіх задач в дослідженні
MinY	REAL	Мінімальне значення y-координати всіх задач в дослідженні
MaxY	REAL	Максимальна y-координата всіх задач в дослідженні
ProblemCount	INTEGER	Кількість задач в дослідженні
OutputExperimentalFilePath	TEXT	Шлях до файлу, що містить вихідні дані результатів дослідження
CreatedAt	TEXT	Дата збереження дослідження
ParameterSetId	INTEGER	Зовнішній ключ до таблиці ParameterSets

3.8 Інструкція користувача

Схема структури сторінки інтерфейсу програмного продукту наведена на рисунку 3.5. Схема допомагає орієнтуватися в інтерфейсі програмного продукту. Після запуску інтерфейсу програми користувач бачить головне меню.

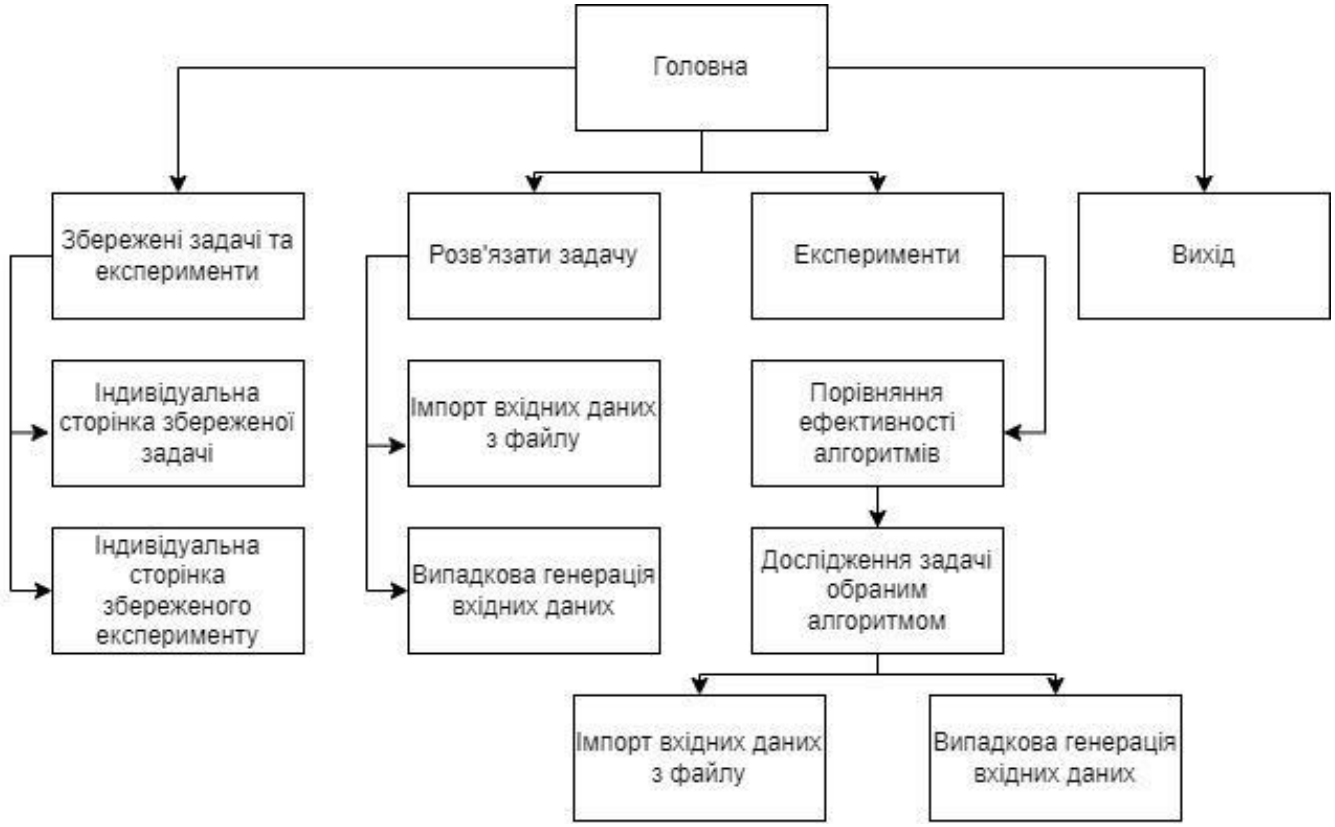


Рисунок 3.5 – Схема структури інтерфейсу застосунку

Щоб перейти до розв’язку задачі, в головному меню потрібно натиснути кнопку «Розв’язати задачу». Ця дія має перенаправити користувача на сторінку вирішення задачі. На цій сторінці ви можете вибрати як вводити координати своїх цілей і баз - з файлу з розширенням .csv або згенерувати випадкові дані. Щоб імпортувати файл csv, натисніть кнопку «Завантажити». Після цього відкриється вікно вибору файлу, де потрібно вибрати файл csv з допустимим форматом координат. Приклад дійсного формату координат представлений на рисунку 3.6. Для спрощення введення даних користувачем, будемо використовувати координати в форматі десяткових градусів[48].

	A	B	C	D
1	4.83	0.49	2.99	1.81
2	6.07	1.27	3.75	1.93
3	5.29	3.39		
4	1.19	3.09		
5	1.57	0.99		
6	4.71	0.53		
7	4.89	1.85		
8	7.23	1.97		
9	2.71	2.93		

Рисунок 3.6 – Файл з прикладом координат баз та цілей

Після вибору файлу бази та цілі будуть відображені у вікні зліва.

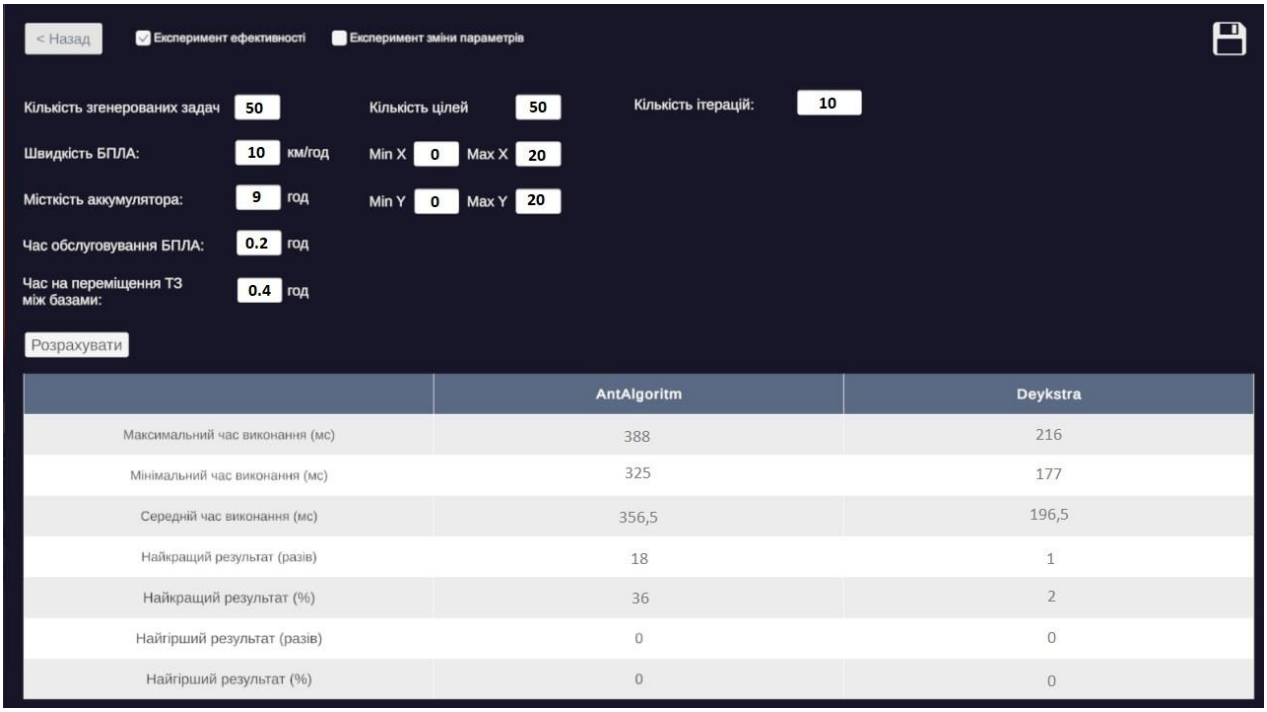
Далі вводимо параметри задачі. Після цього обраємо алгоритм, яким користувач хоче розв’язати задачу. Після цих дій можна натискати кнопку “Розрахувати”.

Щоб зберегти задачу, натисніть значок «Зберегти» у верхньому правому куті. Після цього з'явиться вікно, де потрібно ввести назву задачі та її опис.

Для випадкової генерації координат для баз і цілей необхідно зняти прапорець «Імпорт координат з файлу». Після цього вводимо параметри генерації координат і натискаємо кнопку «Show Data». Після цього у вікні зліва повинні з'явитися цілі та бази.

Для того, щоб провести експеримент необхідно із сторінки головного меню натиснути на кнопку “Експерименти”. Після цього відкриється сторінка проведення експерименту на порівняння ефективності алгоритмів.

Після введення усіх необхідних вхідних даних необхідно натиснути кнопку “Розрахувати”, щоб провести експеримент. Приклад результату експерименту можна побачити на рисунку 3.7.



	AntAlgoritm	Deykstra
Максимальний час виконання (мс)	388	216
Мінімальний час виконання (мс)	325	177
Середній час виконання (мс)	356,5	196,5
Найкращий результат (разів)	18	1
Найкращий результат (%)	36	2
Найгірший результат (разів)	0	0
Найгірший результат (%)	0	0

Рисунок 3.7 – Приклад результатів експерименту

3.9 Тестування системи

Окрім юніт та інтеграційних тестів системи, також виконується ручне тестування. Результати цього тесту наведено в таблиці 3.13 у вигляді тестових випадків. Тестовий випадок представляє набір операцій, які необхідно виконати в певному порядку, щоб перевірити правильність певної функції.

Таблиця 3.13 – Сценарії тестування системи

Номер	Опис	Кроки	Очікуваний результат	Висновок
1	Вирішення задачі з дійсними вхідними даними,	1. Перейдіть на сторінку вирішення проблеми; 2. Введіть дійсні параметри задачі;	Рішення задачі показано на графічному екрані в лівій частині сторінки.	Виконано

	експортованими з файлів CSV	3. Натисніть кнопку “Завантажити”; 4. У вікні вибору файлу виберіть файл csv із дійсними координатами; 5. У випадаючому меню виберіть один із запропонованих алгоритмів у полі; 6. Натисніть кнопку “Розрахувати”.	Рішення задачі показано у виляді тексту під кнопкою “Розрахувати”	
2	Вирішення задачі з прийнятними випадково згенерованими вхідними даними	1. Перейдіть на сторінку вирішення задачі; 2. Введіть дійсні параметри завдання; 3. Введіть дійсні параметри генерації випадкових координат для цілей та баз; 4. У випадаючому меню виберіть один із запропонованих алгоритмів;		Виконано

		5. Натисніть кнопку “Розрахувати”.		
3	Розв’язання задачі з недійсними введеними даними	1. Перейдіть на сторінку вирішення задачі; 2. Введіть недійсні параметри завдання; 3. Натисніть кнопку “Завантажити”.	У верхній частині графічного вікна з’являється повідомлення про те, що введені дані недійсні	Виконано
4	Збереження задачі з дійсними даними	1. Перейдіть на сторінку вирішення задачі; 2. Вирішіть задачу; 3. Натисніть на кнопку у вигляді значка “Зберегти”; 4. У вікні, що з’явиться, введіть назву та опис завдання; 5. Натисніть кнопку “Зберегти”.	Користувач успішно зберіг завдання	Виконано
5	Збереження задачі з недійсними даними	1. Перейдіть на сторінку вирішення задачі; 2. Вирішіть задачу;	Користувач отримує повідомлення про	Виконано

		3. Натисніть на значок “Зберегти”; 4. Залиште назву та опис завдання порожніми; 5. Натисніть кнопку “Зберегти”.	те, що введені дані недійсні	
6	Проведення дослідження на ефективність	1. Перейдіть на сторінку проведення досліджень на ефективність; 2. Заповніть усі обов'язкові поля дійсними даними; 3. Натисніть кнопку “Розрахувати”.	Експериментальні результати представлені у вигляді таблиці	Виконано
7	Проведення досліджень для вивчення варіацій параметрів	1. Перейдіть на сторінку проведення досліджень зміни параметрів; 2. Заповніть усі обов'язкові поля дійсними даними; 3. Натисніть кнопку “Розрахувати”.	Результати дослідження відображаються у вигляді графіків	Виконано

8	Збереження дослідження на ефективність з дійсними даними	1. Перейдіть на сторінку проведення досліджень на ефективність; 2. Проведіть дослідження; 3. Натисніть кнопку “Зберегти” у верхньому правому куті, яка виглядає як значок; 4. Введіть дійсну назву та опис експерименту у вікні, що з’явилось 5. Натисніть кнопку “Ок”.	Експеримент збережено успішно	Виконано
9	Пошук збережених завдань	1. Перейдіть на сторінку “Збережене”; 2. Виберіть опцію “Задачі” на панелі фільтрів 3. Введіть у пошук раніше збережену назву завдання;	Шукане завдання відображається на екрані в списку	Виконано

		4. Натисніть кнопку «Пошук» праворуч від поля пошуку, вона виглядає як значок.		

3.10 Висновки до розділу

У рамках даного розділу було висвітлено всі види вимог до системи, описана бізнес-логіка системи, були обґрунтовані елементи та технології, використані під час створення системи. Також наведено архітектуру системи та бази даних. Наведено інструкцію користувача та результати тестування продукту.

4 РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ

Дослідимо залежність якості рішення алгоритму від різноманітних параметрів.

Алгоритм проведення експериментів:

а) визначити алгоритми та параметри для дослідження;

б) провести експеримент:

1) відкрити вікно проведення експериментів;
2) ввести параметри проведення експериментів;
3) запустити алгоритм;
4) зберегти результати експерименту;
5) змінити досліджуваний параметр;
6) повторити пункти «в»-«д», поки не отримаємо достатню вибірку даних;
7) проаналізувати отримані результати роботи програми.

в) повторити пункт «б» для всіх досліджуваних параметрів.

Спосіб порівняння результатів роботи алгоритму – визначення впливу досліджуваних параметрів на показник кількості найкращих знайдених розв’язків у відсотках, що отримується шляхом підрахунку кількості знайдених в ході експерименту найкращих розв’язків для згенерованих задач.

Метрики, що досліджуються під час експериментів:

— t_max – максимальний час вирішення 1 задачі;

— t_min – мінімальний час вирішення 1 задачі;

— t_avg – середній час вирішення 1 задачі;

— best – кількість найкращих знайдених розв’язків;

— best_perc – кількість найкращих знайдених розв’язків у відсотках.

4.1 Експеримент 1

Дослідження роботи алгоритму на основі оптимізації мурашиними колоніями в залежності від кількості агентів.

Будемо змінювати кількість мурах від 100 до 500 з кроком в 100.

Базові параметри запуску алгоритму:

а) кількість випадкового згенерованих задач = 50;

б) кількість прогонів = 5;

в) швидкість БПЛА = 10;

г) місткість акумулятора = 9;

д) час обслуговування БПЛА = 0.2;

е) час на переміщення ТЗ між базами = 0.4;

ж) min X = 0;

з) min Y = 0;

и) max X = 20;

к) max Y = 20;

л) кількість ітерацій = 10;

м) кількість цілей = 50;

н) рівень випаровування феромону = 0.2.

Результати роботи алгоритму вказано в таблиці 4.1.

Таблиця 4.1 – Залежність якості роботи алгоритму від кількості мурах

Кількість мурах	t_max	t_min	t_avg	best	best_perc
100	388	325	356,5	18	36
200	496	479	487,5	21	42
300	617	583	600	19	38
400	863	812	837,5	16	33
500	1040	984	1012	26	52

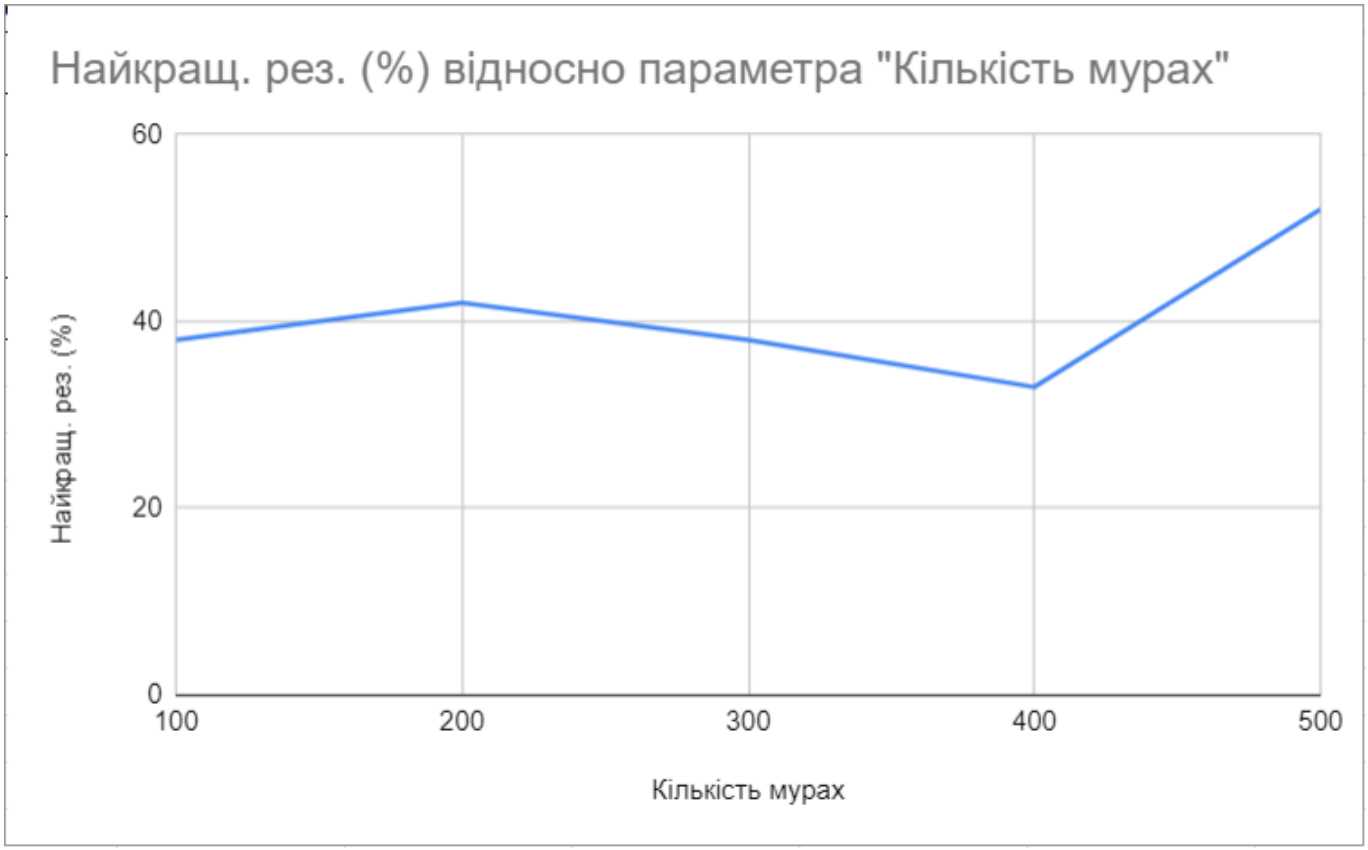


Рисунок 4.1 – Графік залежності кількості знайдених найкращих шляхів (у відсотках) від кількості агентів в алгоритмі

З поданого на рисунку 4.1 графіку ми можемо спостерігати, що зі збільшенням кількості мурах до 500 кількість найкращих знайдених розв’язків в процентному відношенні збільшилась. У випадку 400 агентів, кількість мурах не відчутно вплинула на результат, адже кількість оптимальних розв’язків була меншою, ніж у попередніх випадках.

4.2 Експеримент 2

Дослідження роботи алгоритму на основі оптимізації мурашиними колоніями в залежності від рівня випаровування феромону.

Посилаючись на рисунок 4.1, можна помітити, що при кількості агентів – 100 та коефіцієнту випаровування феромону = 0,2 відсоток найкращих результатів склав 38.

Тепер будемо змінювати рівень випаровування феромону від 0.3 до 0.7.

Базові параметри запуску алгоритму:

а) кількість випадкового згенерованих задач = 50;

б) кількість прогонів = 5;

в) швидкість БПЛА = 10;

г) місткість акумулятора = 9;

д) час обслуговування БПЛА = 0.2;

е) час на переміщення ТЗ між базами = 0.4;

ж) min X = 0;

з) min Y = 0;

и) max X = 20;

к) max Y = 20;

л) кількість ітерацій = 10;

м) кількість цілей = 50;

н) кількість мурах = 100.

Результати роботи алгоритму вказано в таблиці 4.2.

Таблиця 4.2 – Залежність якості роботи алгоритму від рівня випаровування феромону

Рівень випаровування феромону	t_max	t_min	t_avg	best	best_perc
0.2	388	325	356,5	18	36
0.3	342	306	324	19	38
0.45	294	272	283	30	60
0.7	281	245	263	32	64

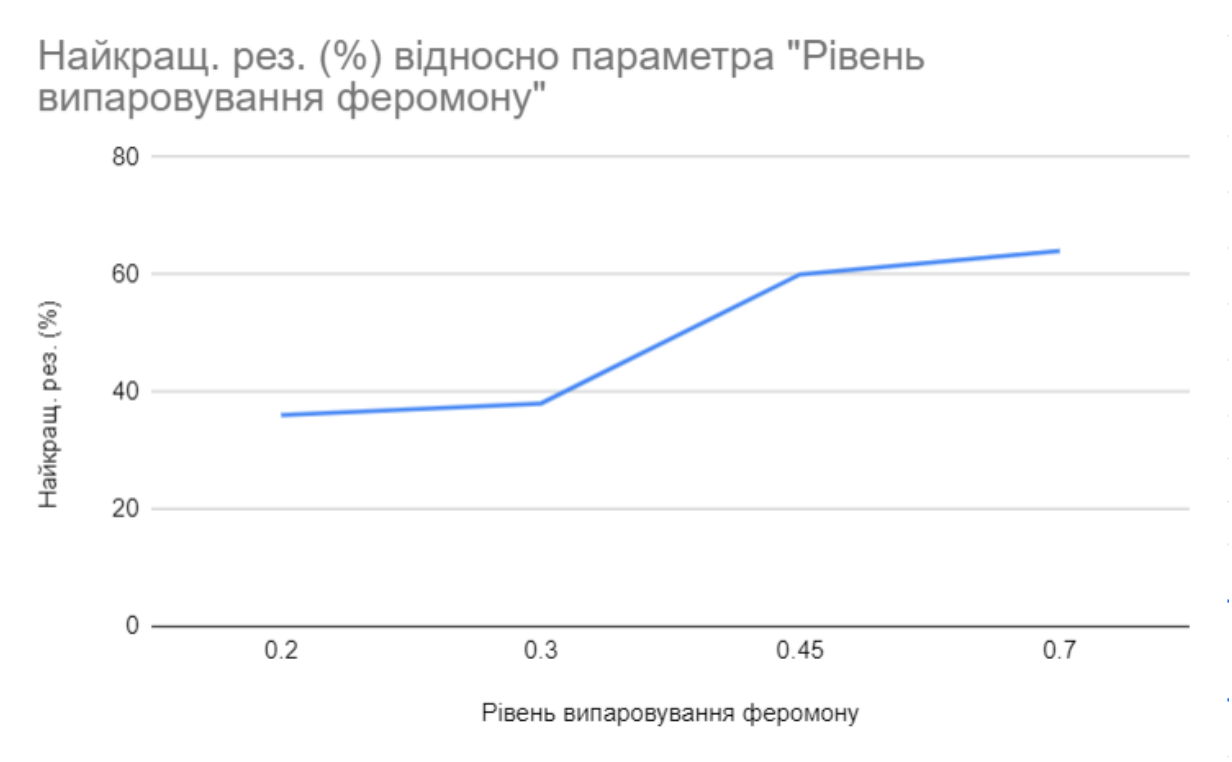


Рисунок 4.2 – Графік залежності кількості знайдених найкращих шляхів (у відсотках) від рівня випаровування феромону

З поданого на рисунку 4.2 графіку ми можемо спостерігати, що зі збільшенням рівня випаровування феромону кількість найкращих знайдених розв’язків в процентному відношенні збільшилась.

4.3 Експеримент 3

Дослідження роботи алгоритму на основі оптимізації мурашиними колоніями в залежності від кількості цілей для обльоту.

Будемо змінювати кількість цілей від 10 до 50 з кроком в 10.

Базові параметри запуску алгоритму:

а) кількість випадкового згенерованих задач = 50;

б) кількість прогонів = 5;

в) швидкість БПЛА = 10;

г) місткість акумулятора = 9;

д) час обслуговування БПЛА = 0.2;

е) час на переміщення ТЗ між базами = 0.4;

ж) min X = 0;

з) min Y = 0;

и) max X = 20;

к) max Y = 20;

л) кількість ітерацій = 10;

м) кількість мурах = 100;

н) рівень випаровування феромону = 0.2.

Результати роботи алгоритму вказано в таблиці 4.3.

Таблиця 4.3 – Залежність якості роботи алгоритму від кількості цілей

Кількість цілей	t_max	t_min	t_avg	best	best_perc
10	36	18	27	26	52
20	83	54	68,5	16	32
30	178	107	142,5	19	38
40	259	192	225,5	21	42
50	388	325	356,5	18	36

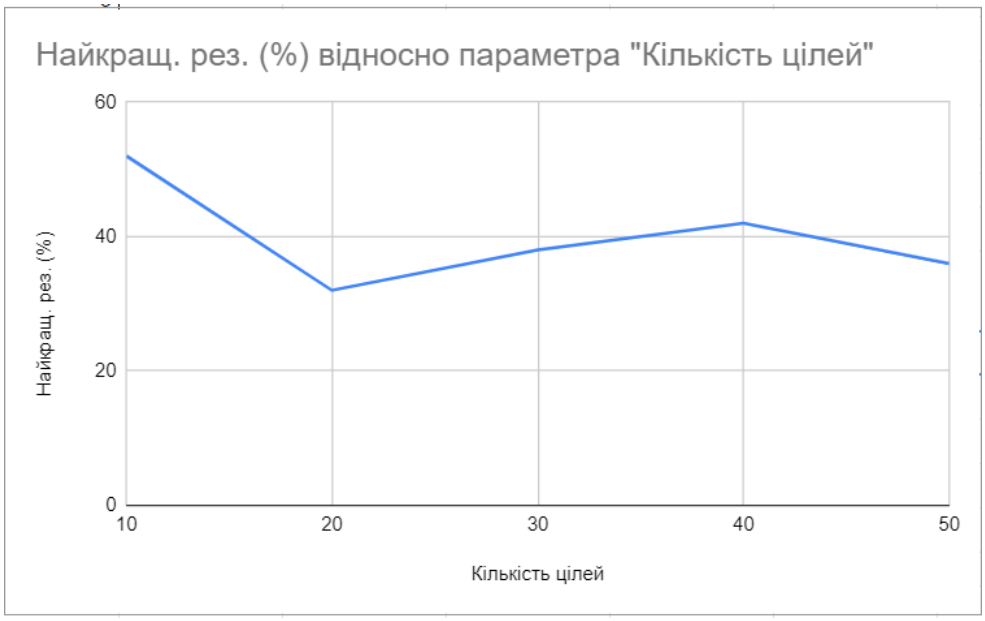


Рисунок 4.3 – Графік залежності кількості знайдених найкращих шляхів (у відсотках) від кількості цілей для обстеження

З поданого на рисунку 4.3 графіку ми можемо спостерігати, що зі збільшенням кількості цілей кількість найкращих знайдених розв’язків в процентному відношенні трохи зменшилась.

4.4 Експеримент 4

Дослідження роботи модифікованого алгоритму Дейкстри в залежності від кількості цілей для обльоту.

Будемо змінювати кількість цілей від 10 до 50 з кроком в 10.

Базові параметри запуску алгоритму:

а) кількість випадкового згенерованих задач = 50;

б) кількість прогонів = 5;

в) швидкість БПЛА = 10;

г) місткість акумулятора = 9;

д) час обслуговування БПЛА = 0.2;

е) час на переміщення ТЗ між базами = 0.4;

ж) min X = 0;

з) min Y = 0;

и) max X = 20;

к) max Y = 20;

л) кількість ітерацій = 10.

Результати роботи алгоритму вказано в таблиці 4.4.

Таблиця 4.4 – Залежність якості роботи алгоритму від кількості цілей

Кількість цілей	t_max	t_min	t_avg	best	best_perc
10	26	14	20	14	28
20	45	32	77	16	32
30	84	67	75,5	6	12
40	143	115	129	2	4

50	216	177	196,5	1	2

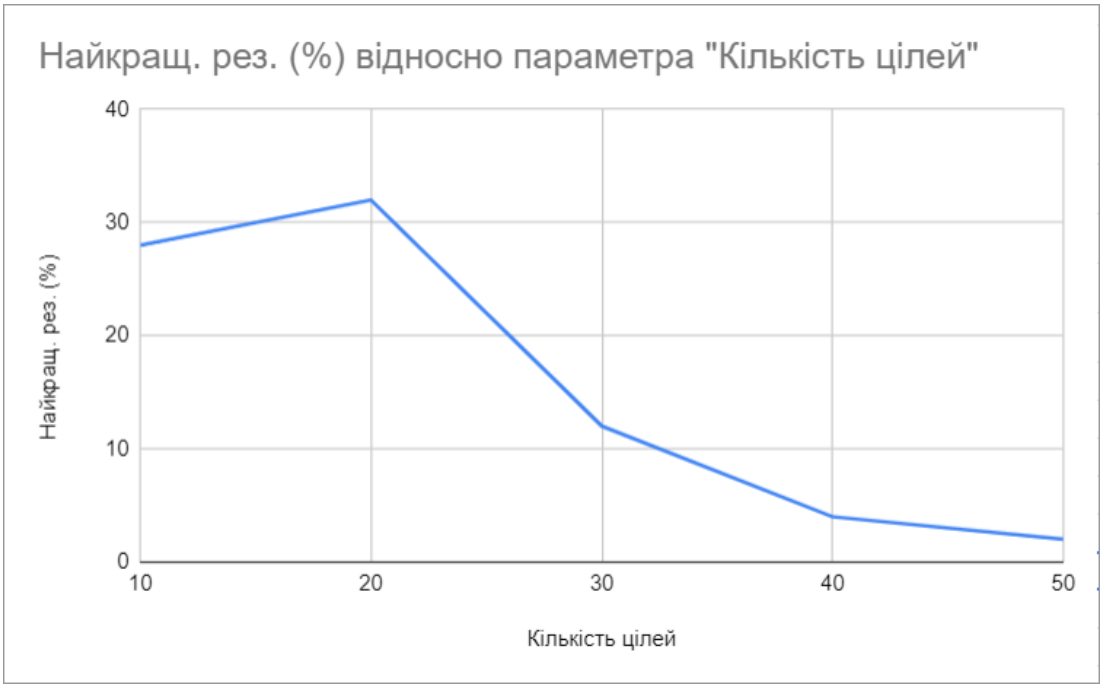
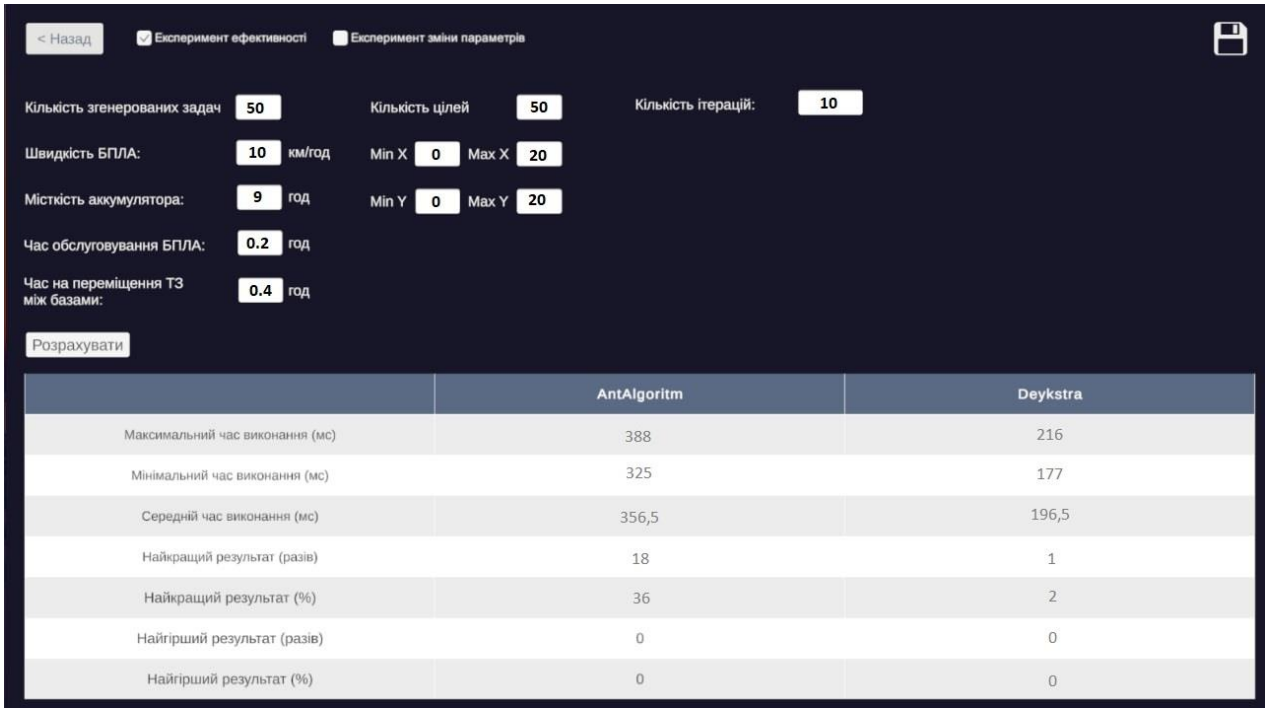


Рисунок 4.4 – Графік залежності кількості знайдених найкращих шляхів (у відсотках) від кількості цілей для обстеження

З поданого на рисунку 4.4 ми можемо спостерігати, що зі збільшенням кількості цілей кількість найкращих знайдених розв’язків в процентному відношенні значно зменшилась.

Набагато гірший результат експерименту, порівняно з попереднім алгоритмом, пояснюється наступним. Для коректної роботи алгоритма Дейкстри з обраною задачею, до алгоритма було додано обмеження, завдяки якому БПЛА прагнув максимально використати місткість свого акумулятора перш ніж повернутись на базу. Це погано вплинуло на ефективність алгоритму, бо максимальне використання ємності акумулятора не завжди призводить до кращого рішення.

На рисунку 4.5 бачимо скріншот порівняння двох алгоритмів під час розв’язання задачі із кількістю цілей 50.



	AntAlgoritm	Deykstra
Максимальний час виконання (мс)	388	216
Мінімальний час виконання (мс)	325	177
Середній час виконання (мс)	356,5	196,5
Найкращий результат (разів)	18	1
Найкращий результат (%)	36	2
Найгірший результат (разів)	0	0
Найгірший результат (%)	0	0

Рисунок 4.5 – Порівняння ефективності алгоритмів Дейкстри та оптимізації мурашиними колоніями

На рисунку 4.6 подано графічне представлення ефективності алгоритмів у випадку вирішення задачі з 50 цілями.

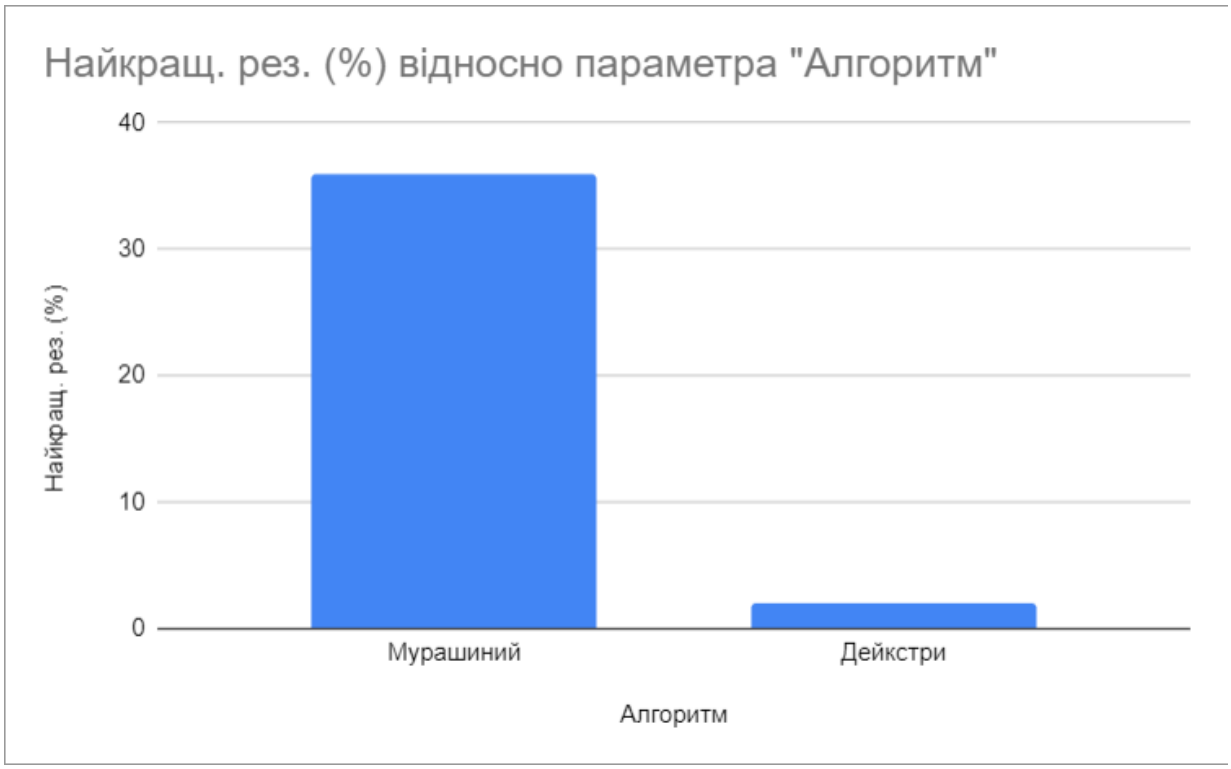


Рисунок 4.6 – Графік ефективності алгоритмів для задачі з 50 цілями

4.5 Висновки до розділу

У рамках даного розділу були проведені експериментальні дослідження розроблених алгоритмів.

Під час експериментів отримано наступні результати:

— при збільшенні кількості агентів у алгоритмі на основі оптимізації мурашиними колоніями кількість найкращих знайдених розв’язків в процентному відношенні збільшилась. У випадку 400 агентів, кількість агентів не відчутно вплинула на результат, адже кількість оптимальних розв’язків була меншою, ніж у попередніх випадках;

— зі збільшенням рівня випаровування феромону в алгоритмі на основі оптимізації мурашиними колоніями кількість найкращих знайдених розв’язків в процентному відношенні збільшилась;

— зі збільшенням кількості цілей в алгоритмі на основі оптимізації мурашиними колоніями кількість найкращих знайдених розв’язків в процентному відношенні трохи зменшилась;

— зі збільшенням кількості цілей у модифікованому алгоритмі Дейкстри кількість найкращих знайдених розв’язків в процентному відношенні значно зменшилась.

Отже, для вирішення поставленої задачі доцільно використовувати алгоритм на основі оптимізації мурашиними колоніями.

5 СТАРТАП ПРОЄКТ

5.1 Опис ідеї проєкту

Ідеєю даного проєкту є надання програмного забезпечення для підтримки дослідження задачі складання маршрутів БПЛА з рухомою базою обслуговування.

Таблиця 5.1 – Опис ідеї та сфери застосування проєкту

Зміст ідеї	Сфера застосування	Напрямки застосування
Надання програмного забезпечення для підтримки дослідження задачі складання маршрутів БПЛА з рухомою базою обслуговування	Військова	Розвідка
		Вогневі ударні
		Забезпечення
	Цивільна	Виявлення малорозмірних об’єктів
		Керування повітряним рухом
		Контроль морського судноплавства
		Застосування в сільському господарстві
		Застосування в геологорозвідці
		Розвиток регіональних та міжрегіональних телекомунікаційних мереж

		Аерофотознімання та контроль земної поверхні
		Контроль екологічної обстановки

5.2 Технологічний аудит ідеї проєкту

У рамках цього підрозділу розглядається технологія створення програмного продукту та визначається технічна можливість проєкту (див. таблицю 5.2).

Таблиця 5.2 – Можливість здійснення ідеї проєкту з технологічного боку

№ з/п	Ідея проєкту	За допомогою яких технологій реалізовано	Чи наявні технології	Чи доступні технології
1	Візуалізація маршрутів	Бібліотека LineRenderer	Наявна	Доступна
2	Збереження рішення задач та результатів досліджень	SQLite	Наявна	Доступна
3	Алгоритми побудови та оптимізації шляхів	Бібліотека AlgoSolutionLib	Наявна	Доступна
4	Візуалізація результатів	Бібліотека SimpleTableUI	Наявна	Доступна

	досліджень у табличному вигляді			
5	Візуалізація досліджень у графічному вигляді	Бібліотека Dynamic Line Chart	Наявна	Доступна
6	Імпорт початкових даних та експорт отриманих результатів за допомогою файлів csv	Бібліотека CSVFileManager	Наявна	Доступна
7	Генерація випадкових початкових даних	Бібліотека RandomDataGenerator	Наявна	Доступна

Як видно з попередньої таблиці, усі технології, необхідні для реалізації цього застосунку наявні на сьогодні, а також знаходяться у вільному доступі для комерційного використання. Тому реалізація цього проєкту цілком здійсненна.

5.3 Аналіз ринкових можливостей запуску стартап-проєкту

Щоб спланувати напрямок розвитку продукту, необхідно визначити ринкові можливості. Наявність попиту, його обсяг та динаміка розвитку наведені у таблиці 5.3.

Таблиця 5.3 – Попередня характеристика можливостей запуску стартапу для потенційного ринку

№ з/п	Показники ринкового стану (найменування)	Значення
1	Кількість головних гравців, од	5
2	Загальний обсяг продаж, грн/ум.од	Невідомий
3	Динаміка ринку	Зростає
4	Наявність обмежень для входу	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Юридичні
6	Середня норма рентабельності в галузі (або по ринку), %	50 %

Враховуючи отриману середню норму рентабельності, можна зробити висновок, що ринок привабливий для входу.

Таблиця 5.4 визначає приблизний перелік сегментів потенційних клієнтів, їх характеристики та вимоги до продукту для кожного сегмента.

Таблиця 5.4 - Характеристика потенційних цільових груп клієнтів стартапу

№ з/п	Ринок формується через потребу	Цільові групи клієнтів (сегменти ринку)	Поведінка потенційних цільових груп клієнтів	Вимоги користувачів до застосунку
1	Складання оптимальних маршрутів для БПЛА з рухомою базою обслуговування	Приватні компанії	Оскільки продуктом зазвичай користується ціла установа, необхідна підписка на декілька осіб	Зручний інтерфейс користувача, гнучке налаштування параметрів завдання, зручне введення даних і можливість збереження отриманих результатів

Після визначення потенційних груп клієнтів ми проведемо аналіз ринкового середовища. У таблиці 5.5 наведено перелік загроз реалізації проєкту. У таблиці 5.6 наведено фактори можливостей, які навпаки сприяють цій ситуації.

Таблиця 5.5 – Фактори загроз стартап-проєкту

№ з/п	Фактор загрози	Зміст загрози	Можливе рішення компанії
1	Політична ситуація у країнах розробки та впровадження	Війна або воєнний стан, революція, зміна державного ладу	Впровадити нові функції, скоригувати цінову політику
2	Епідеміологічна ситуація в країнах, що розробляють, і країнах, що впроваджують	Введення надзвичайного стану, введення обмежень на пересування	
3	Низький технічний рівень користувачів	Щоб використовувати методи оптимізації маршрутів, зазвичай потрібно базове розуміння змісту проблеми	
4	Посилення конкуренції	Поява нових конкурентів на ринку з схожим функціоналом	

Таблиця 5.6 – Фактори можливостей стартап-проєкту

№ з/п	Фактор можливості	Зміст можливості	Можливе рішення компанії
1	Розвиток технологій	Поява нових технологій або розвиток існуючих дозволить вдосконалювати програмні продукти - покращити швидкість роботи і зручність використання для клієнтів	Розширення команди для подальшого розвитку продукту
2	Розвиток наукових напрямів у галузі дослідження проблем маршрутизації та їх вирішення	Розвиток наукового напряму сприятиме створенню нових наукових праць з проблем маршрутизації. У свою чергу, поява нових типів проблем і рішень для них дозволить розширити програмні	

		продукти шляхом впровадження	
3	Комерційний і військовий розвиток	Попит на продукти, які вивчають проблеми маршрутизації, ймовірно, зросте через розвиток комерційних і військових технологій	

Результати ступеневого аналізу конкуренції на ринку наведені в таблиці 5.7.

Таблиця 5.7 – Аналіз конкуренції за ступенями на ринку

Особливості середовища конкуренції	Вияви поточної характеристики	Особливості середовища конкуренції
Тип конкуренції - чиста	Крім розглянутого програмного продукту, на ринку вже є кілька програмних рішень подібних проблем	Впровадити функції, яких наразі немає у конкурентів
За рівнем конкурентної боротьби - глобальний	Глобалізація полягає в тому, що продукти поширюються не лише в Україні	Охопити всесвітній ринок

За галузевою ознакою - міжгалузева	ІТ галузь	Розвивати ІТ технології
Конкуренція за видами товарів: товарно-видова	Подібні програмні продукти для оптимізації вже існують, але зазвичай вони призначені для вирішення конкретних типів проблем маршрутизації.	Прислухатись до побажань користувачів
За характером конкурентних переваг - нецінова	Замовники орієнтуються не на ціну, а на якість і зручність використання програмних продуктів	Підтримувати високий рівень якості програмних продуктів
За інтенсивністю - не марочна	Програмні продукти програми не мають широкого визнання	Просувати програмні продукти

Результати аналізу конкуренції в галузі за М. Портером наведені в таблиці 5.8.

Таблиця 5.8 - Аналіз існуючої конкуренції в обраній галузі за М. Портером

Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Споживачі	Товари, що можуть замінити
Routific Route Optimization	AIMMS	Відсутні	Наукові дослідники, приватні компанії	Часткова заміна присутня

API, Google Or-Tools				
Висновки		Конкуренти не мають постачальників		Конкуренти мають подібні можливості, але мають дуже вузький вибір методів вирішення проблем і дослідження

Проаналізувавши конкуренцію, можна зробити висновок, що шанси вивести продукт на ринок високі. Хоча в цій галузі вже є конкуренти, вони не охоплюють повного спектру можливостей, які можна реалізувати в контексті проблем маршрутизації транспорту та їх вирішення. Далі ми перелічимо фактори конкурентоспроможності в таблиці 5.9.

Таблиця 5.9 - Опис факторів конкурентоспроможності

№ з/п	Фактор конкурентоспроможності	Дослідження (наведення фактів, що пояснюють значущість фактора для порівняння конкурентних проєктів)
1	Цінова політика	Цінова політика є важливим фактором для потенційних користувачів програмних продуктів

2	Дизайн інтерфейсу	Продуманий і зручний дизайн інтерфейсу дозволяє економити час, витрачений користувачами на використання програмного продукту
3	Швидкодія роботи програмного продукту	Швидкість - ще основний фактор, який допомагає заощадити час
4	Впізнаваність компанії	Чим вище популярність компанії, тим більше потенційних клієнтів
5	Наявність додаткового функціоналу	Цей фактор допомагає позитивно виділитися серед конкурентів
6	Якість комунікації між користувачем та розробником	Якісне спілкування допомагає швидко виявляти проблеми в програмних продуктах і виправляти їх

На основі факторів, визначених у таблиці вище, ми проведемо порівняльний аналіз наявних слабких і сильних сторін продукту (таблиця 5.10).

Таблиця 5.10 - Порівняльний аналіз переваг та недоліків застосунку

№ з/п	Фактор конкурентоспроможності	Бали від 1 до 20	Рейтинг товарів, що конкурують						
			-3	-2	-1	0	+1	+2	+3
1	Цінова політика	15	+						
2	Зручність інтерфейсу	16			+				
3	Швидкість роботи застосунку	19					+		
4	Популярність компанії	18						+	
5	Наявність додаткових можливостей розв’язання задач предметної галузі	17				+			
6	Якість комунікації між клієнтом та постачальником	13		+					

На заключному етапі аналізу ринку ми складемо SWOT-аналіз, наведений у таблиці 5.11.

Таблиця 5.11 – Аналіз стартапу за SWOT

Переваги: цінова політика, лояльна до користувача, зручний інтерфейс	Недоліки: відсутність можливості працювати з різними платформами, невелика кількість алгоритмів для вирішення задач обраної предметної області
Можливості: залучення інвесторів для покращення стартапу	Загрози: швидкий розвиток конкурентів

Після проведення SWOT-аналізу необхідно визначити альтернативи ринкової поведінки, щоб вивести стартап на ринок. Ці альтернативи аналізуються в контексті визначення умов реалізації на ринку та можливості доступу до ресурсів (див. таблицю 5.12).

Таблиця 5.12 – Альтернативи впровадження стартапу на ринок

№ з/п	Альтернатива	Шанс отримання ресурсів для розвитку	Строки реалізації
1	Розповсюдження застосунку на інші платформи	Високий	6 місяців

5.4 Розроблення ринкової стратегії проєкту

Зазвичай при формуванні ринкової стратегії першим кроком є визначення стратегії охоплення ринку, тобто визначення потенційних споживачів і їх поділ на групи, як показано в таблиці 5.13.

Таблиця 5.13 - Визначення цільових груп потенційних клієнтів

№ з/п	Опис цільової групи можливих клієнтів	Готовність клієнтів сприйняти продукт	Приблизний попит серед цільової аудиторії (сегменту)	Рівень конкуренції в групі	Простота початку роботи у сегменті
1	Логістичні компанії	90%	100%	Високий	Низька

2	Військові логістичні та розвідувальні підрозділи	70%	100%	Низький	Висока
3	Наукові структури	80%	100%	Середній	Середня
Які цільові групи обрано: усі (1, 2, 3)					

За результатами визначення цільової групи потенційних споживачів було вирішено працювати з усіма сегментами ринку одночасно, тобто з використанням масового маркетингу. Далі визначимо базову стратегію розвитку у таблиці 5.14.

Таблиця 5.14 – Опис базового плану розвитку застосунку

№ з/п	Альтернативна можливість розвитку стартапу	Стратегія опанування ринку	Ключові позиції згідно з обраною альтернативою	Базовий план розвитку
1	Зосередження на науковому сегменті	Масовий маркетинг	Висока швидкодія, зручний інтерфейс, різноманіття методів	Стратегія диференціації

Далі у таблиці 5.15 визначимо базову стратегію конкурентної поведінки.

Таблиця 5.15 - Опис базового плану конкурентної поведінки

№ з/п	Чи є застосунок першим в своєму сегменті на ринку?	Стратегія розширення бази клієнтів	Стратегія розширення функціоналу застосунку	Стратегія конкурентної поведінки
1	Ні	Компанія планує шукати нових клієнтів та переманювати існуючих у конкурентів	Компанія не планує копіювати характеристики товару конкурента.	Стратегія лідера

У таблиці 5.16 визначимо стратегію позиціонування.

Таблиця 5.16 - Опис плану позиціонування

№ з/п	Вимоги до застосунку цільової групи можливих клієнтів	Базовий план розвитку	Ключові позиції власного стартапу	Вибір факторів, які формують комплексну позицію стартапу
1	Зручний інтерфейс	Стратегія диференціації	Слідування правилам UI/UX дизайну, швидкодія,	Лояльна цінова політика
2	Різноманіття алгоритмів			
3	Зручний функціонал аналізу алгоритмів			

4	Швидкодія програмного продукту		широкий спектр функціоналу	

5.5 Розробка маркетингової програми стартап-проєкту

В рамках цього блоку необхідно сформувати уявлення про продукт, яким буде користуватися користувач. З цією метою в таблиці 5.17 наведено основні переваги концепцій потенційного продукту.

Таблиця 5.17 - Опис головних переваг концепції застосунку

№ з/п	Завдання	Вигода, яку пропонує застосунок	Сильні сторони, порівняно з конкурентами
1	Підтримка дослідження задачі складання маршруту польоту БПЛА з рухомою базою обслуговування	Вивчення задач маршрутизації та методів їх розв’язання	Наявність інтерфейсу, візуалізація результатів, наявність декількох алгоритмів розв’язання
2	Складання маршруту польоту БПЛА з рухомою	Зменшення часу на обслуговування клієнтів	

	базою обслуговування		

Трирівневий опис моделі товару наведено в таблиці 5.18.

Таблиця 5.18 – Представлення рівнів моделі товару

Рівні об’єкту дослідження	Пояснення та складові об’єкту дослідження		
I. Товар за задумом	Застосунок пропонує функціонал з підтримки дослідження задачі складання маршруту польоту БПЛА з рухомою базою обслуговування		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. зручний інтерфейс; 2. доволі низька ціна; 3. складання маршрутів дронів; 4. дослідження задачі оптимізації транспортних засобів; 5. швидкість роботи.	-	-
	Якість: перевірка якості була здійснена за допомогою різних типів тестування, а саме: інтеграційні тести, юніт-тести, мануальне тестування		
	Комплектація: документація в електронному вигляді (інструкція користувача)		
	Марка: UAVRP		
III. Товар із підкріпленням	До продажу: Бета-версія з обмеженою розмірністю задачі		
	Після продажу: розвиток застосунку шляхом додавання нових методів розв’язання та функціоналу, підтримка користувачів		

Захист від копіювання буде здійснюватись за допомогою ліцензійних ключів.

Формування систем збуту описане у таблиці 5.19.

Таблиця 5.19 – Створення системи збуту

№ з/п	Особливості поведінки клієнтів під час закупівлі	Обов'язки збуту, які виконує розробник товару	Глибина каналу збуту	Найкраща система збуту
1			Однорівневий	Власна та залучена

Як останній крок у таблиці 5.20 представлена концепція маркетингових комунікацій.

Таблиця 5.20 – Концепція маркетингових зв'язків

№ з/п	Особливості поведінки цільової аудиторії	Канали зв'язку, якими користується цільова аудиторія	Головні позиції, що були обрані для позиціонування	Завдання рекламної компанії	Основа рекламного звернення
1	Орієнтація на зручність і швидкість програмних продуктів	Месенджер, соціальна мережа, електронна пошта	Створення довіру до компанії та продукту через спілкування	Завдання – донести до замовника, яку проблему може вирішити	Підкреслити переваги програмного продукту перед конкурентами

				програмний продукт	

5.6 Висновки до розділу

Проаналізувавши наявні потреби, рентабельність, динаміку ринку, потенційних клієнтів і конкурентів, можна зробити висновок, що існує можливість комерціалізації проєкту на ринку, перспективи реалізації досить високі, оскільки на даний момент конкуренції в галузі майже немає. Попит на цей вид продукції зростає з кожним роком, тому подальша реалізація цього проєкту є цілком доцільною. При розробці маркетингових комунікацій було вирішено зробити акцент на функціональності програмного продукту та спектрі завдань, які він може вирішити.

ВИСНОВКИ

В ході роботи було розглянуто ЗМТЗ і встановлено, що на даний момент існує багато різновидів: з обмеженнями вантажопідйомності, «часовими вікнами», альтернативними динамічними складами, поверненнями та доставками, різними видами транспорту, регулярним маршрутом, можливими перевантаженнями тощо. Тому наведено класифікацію найбільш поширених ЗМТЗ.

Крім того, визначено змістовну постановку проблеми та описано Задачу ІІ, вирішення якої є головною метою проєкту. Щоб отримати загальне уявлення про процес розв'язання, наведено математичну модель Задачі І, що є підзадачею Задачі ІІ, та описи вхідних, проміжних і вихідних даних, які є основою для обраної предметної області.

Також визначено та описано рішення Задачі ІІ. Для пошуку рішення було обрано два підходи: алгоритм, заснований на алгоритмі Дейкстри, і алгоритм, заснований на методі оптимізації колонії мурашок.

Алгоритм Дейкстри модифікується таким чином:

— вершини графу, що є базами, можуть бути відвідані необмежену кількість разів;

— враховані обмеження на ємність БПЛА та необхідність її поновлення;

— враховані обмеження, пов'язані з часовими вікнами (транспортний засіб має прибути до бази В2 не пізніше БПЛА.

Така ж модифікація застосована до другого алгоритму (на основі методу оптимізації колонії мурашок).

Далі висвітлюються всі типи вимог до системи, описується бізнес-логіка системи, а також уточнюються елементи та прийоми, використані при створенні системи. Також наведено архітектуру системи та бази даних. Надано посібник користувача та результати тестування продукту.

Після цього розроблені алгоритми досліджуються експериментально. Під час дослідження отримано такі результати:

— коли кількість агентів в алгоритмі оптимізації мурашиної колонії збільшується, відсоток найкращих знайдених розв'язків збільшується. У випадку 400 агентів

кількість агентів не має істотного впливу на результати, оскільки кількість найкращих знайдених розв’язків менше, ніж у попередніх випадках;

- зі збільшенням ступеня випаровування феромонів в алгоритмі оптимізації мурашиної колонії збільшується відсоток кількості найкращих знайдених розв’язків;
- зі збільшенням кількості цілей в алгоритмі оптимізації мурашиної колонії відсоток кількості найкращих знайдених розв’язків дещо зменшується;
- зі збільшенням кількості цілей у модифікованому алгоритмі Дейкстри значно зменшується відсоток кількості найкращих знайдених розв’язків.

Тому для вирішення поставленої задачі рекомендовано використовувати алгоритм, заснований на оптимізації колонії мурашок.

Проаналізовано існуючий попит, прибутковість, динаміку ринку, потенційних клієнтів і конкурентів. Визначено, що проєкт має потенціал для комерціалізації на ринку, а перспективи реалізації досить високі, оскільки конкуренція в галузі зараз незначна. Попит на даний вид продукції зростає з кожним роком, тому подальша реалізація проєкту є цілком доцільною. При розробці маркетингових комунікацій було вирішено зробити акцент на функціональності програмного продукту та спектрі завдань, які він може вирішувати.

ЛІТЕРАТУРА

1. R. W. Harder, R. R. Hill, and J. T. Moore, ―A Java universal vehicle router for routing unmanned aerial vehicles,‖ International Transactions in Operational Research, vol. 11, no. 3, pp. 259–275, 2004.
2. V. K. Shetty, M. Sudit, and R. Nagi, ―Priority-based assignment and routing of a feet of unmanned combat aerial vehicles,‖ Computers & Operations Research, vol. 35, no. 6, pp. 1813–1828, 2008.
3. A. M. Ham, ―Integrated scheduling of m-truck, m-drone, and m-depot constrained by time-window, drop-pickup, and mvisit using constraint programming,‖ Transportation Research Part C: Emerging Technologies, vol. 91, pp. 1–14, 2018.
4. Multi-uav resource constrained online monitoring of large-scale spatio- temporal environment with homing guarantee.
5. Avellar, G.S.C., Pereira, G.A.S., Pimenta, L.C.A., Iscold, P., 2015. Multi-UAV routing for area coverage and remote sensing with minimum time. Sensors 15 (11), 27783–27803.
6. Torres, M., Pelta, D. A., Verdegay, J. L., Coverage path planning with unmanned aerial vehicles for 3D terrain reconstruction. Expert Systems with Applications. 2016. Vol. 55. C. 441–451.
7. Guerriero, F., Surace, R., Loscr, V., Natalizio, E., 2014. A multi- objective approach for unmanned aerial vehicle routing problem with soft time windows constraints. Appl. Math. Model. 38 (3), 839–852.
8. Di Puglia Pugliese, L., Guerriero, F., Zorbas, D., Razafindralambo, T., 2016. Modelling the mobile target covering problem using flying drones. Optim. Lett. 10 (5), 1021–1052.
9. Ponda, S. S., Johnson, L. B., Geramifard, A. Cooperative mission planning for multi-UAV teams. In: Handbook of Unmanned Aerial Vehicles. Dordrecht: Springer. 2015. P. 1447-1490.

10. Liu, Y., Liu, Z., Shi, J. Optimization of Base Location and Patrol Routes for Unmanned Aerial Vehicles in Border Intelligence, Surveillance, and Reconnaissance. Journal of Advanced Transportation. 2019. Vol. 2019. C. 1–13.
11. G. S. C. Avellar, G. A. S. Pereira, L. C. A. Pimenta, and P. Iscold, ―Multi-UAV routing for area coverage and remote sensing with minimum time,‖ Sensors, vol. 15, no. 11, pp. 27783–27803, 2015.
12. Alotaibi, K.A., Rosenberger, J.M., Mattingly, S.P., Punugu, R.K., Visoldilokpun, S., 2018. Unmanned aerial vehicle routing in the presence of threats. Comput. Ind. Eng. 115, 190–205.
13. Ines Khouf ―A Survey of Recent Extended Variants of the Traveling Salesman and Vehicle Routing Problems for Unmanned Aerial Vehicles‖.
14. Colnarič, M., Behnck, L. P., Doering, D., Pereira, C. E., and Rettberg, A., ―A Modified Simulated Annealing Algorithm for SUAVs Path Planning,‖ in 2nd IFAC Conference on Embedded Systems, Computer Intelligence and Telematics, Maribor, Slovenia, 2015, vol. 48, pp. 63–68.
15. Ergezer, H. and Leblebicioglu, K., ―Path Planning for UAVs for Maximum Information Collection,‖ IEEE Trans. Aerosp. Electron. Syst., vol. 49, no. 1, pp. 502–520, Jan. 2013.
16. Zhang, X. and Duan, H., ―An improved constrained differential evolution algorithm for unmanned aerial vehicle global route planning,‖ Appl. Soft Comput., vol. 26, pp. 270–284, Jan. 2015.
17. Yangguang Fu, Mingyue Ding, and Chengping Zhou, ―Phase Angle- Encoded and Quantum-Behaved Particle Swarm Optimization Applied to ThreeDimensional Route Planning for UAV,‖ Syst. Man Cybern. Part Syst. Hum. IEEE Trans. On, vol. 42, no. 2, pp. 511–526, 2012.
18. Cekmez, U., Ozsiginan, M., and Sahingoz, O. K., ―A UAV path planning with parallel ACO algorithm on CUDA platform,‖ in 2014 International Conference on Unmanned Aircraft Systems (ICUAS), 2014, pp. 347–354.

19. Xu, C., Duan, H., and Liu, F., ―Chaotic artificial bee colony approach to Uninhabited Combat Air Vehicle (UCAV) path planning,‖ Aerosp. Sci. Technol., vol. 14, no. 8, pp. 535–541, Dec. 2010.
20. Zhu, W. and Duan, H., ―Chaotic predator–prey biogeography-based optimization approach for UCAV path planning,‖ Aerosp. Sci. Technol., vol. 32, no. 1, pp. 153–161, Jan. 2014.
21. Zhang, S., Zhou, Y., Li, Z., and Pan, W., ―Grey wolf optimizer for unmanned combat aerial vehicle path planning,‖ Adv. Eng. Softw., vol. 99, pp. 121– 136, Sep. 2016.
22. Gremlins [Електронний ресурс] – Режим доступу до ресурсу: https://www.darpa.mil/program/gremlins.
23. 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.
24. 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.
25. 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.
26. 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.
27. 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.
28. Zhen l. A vehicle routing problem arising in unmanned aerial monitoring. Computers & Operations Research. 2019. Vol. 105. P. 1–11.

29. 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.
30. 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.
31. 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.
32. Alotaibi K., Rosenbarger J. Unmanned aerial vehicle routing in the presence of threats. Computers & Industrial Engineering. 2018. Vol. 115. P. 190–205.
33. Wang Z., Sheu J. Vehicle routing problem with drones. Transportation research part B: methodological. 2019. Vol. 122. P. 350–364.
34. Li M., Zhen L. Unmanned aerial vehicle scheduling problem for traffic monitoring. Computers & Industrial Engineering. 2018. Vol. 122. P. 15–23.
35. 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.
36. 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.
37. 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.
38. 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.
39. 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.

40. 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.
41. Гуляницький Л. Ф., Мулеса О. Ю. Прикладні методи комбінаторної оптимізації: навч. посіб. Київ : Видавничо-поліграфічний центр «Київський університет», 2016. 146 с.
42. Routific Route Optimization API [Електронний ресурс] – Режим доступу до ресурсу: https://dev.routific.com/ .
43. Google OR-Tools [Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/optimization .
44. AIMMS [Електронний ресурс] – Режим доступу до ресурсу: https://www.aimms.com/ .
45. E. Dijkstra, “A note on two problems in connexion with graphs,” in Numerische Mathematik, vol. 1. 1959, P. 269–271.
46. P. E. Hart, N. J. Nilsson and B. Raphael, “A Formal Basis for the Heuristic Determination of Minimum Cost Paths,” in IEEE Transactions on Systems Science and Cybernetics SSC4, vol. 2. 1968, P. 100–107.
47. What Makes Unity So Popular in Game Development? [Електронний ресурс] – Режим доступу до ресурсу: https://www.arnia.com/what-makes-unity-so-popular-in-game-development/ .
48. Перетворення градусів, хвилин, секунд у десяткові градуси [Електронний ресурс] – Режим доступу до ресурсу: https://www.rapidtables.org/uk/convert/number/degrees-minutes-seconds-to-degrees.html .
49. Garey, M. R., & Johnson, D. S. (1979). Computers and intractability: A guide to the theory of NP-completeness. New York, USA: W.H. Freeman & Co Ltd.