УДК 004.652.4+004.827

В. В. Литвин[1], Д. І. Угрин[2], О. Д. Іллюк[2], С. В. Білоус[3], З. Л. Рибчак[1]
[1]Національний університет "Львівська політехніка",
кафедра інформаційних систем та мереж;
[2]Чернівецький факультет НТУ "Харківський політехнічний інститут",
кафедра інформаційних систем,
[3]Компанія Ceit.dev

СИСТЕМА ОПТИМІЗАЦІЇ МАРШРУТІВ ТУРИЗМУ НА ОСНОВІ МОДИФІКАЦІЇ ГЕНЕТИЧНОГО ТА МУРАШИНОГО АЛГОРИТМІВ

Запропоновано використовувати модифіковані оператори ініціалізації та схрещування мурашиного та генетичного алгоритмів для розв'язування транспортної задачі у сфері туризму. На основі аналізу поведінки мурашиних колоній, а саме пошуку найкоротшого маршруту виділенням феромонів та функції схрещування двох рішень генетичного алгоритму, розроблено метод та алгоритм виконання таких операцій: пошук оптимального маршруту, розрахунок витрат ресурсів, пошук дистанції, час маршруту, запам'ятовування виконаних маршрутів. У роботі описано створену систему для мобільних телефонів під операційну систему IOS, що виконує всі перераховані вище операції. Проведено тестування мобільного додатка методом "спочатку тест".

Ключові слова: мурашині колонії, генетичний алгоритм, туристичні маршрути, мобільний додаток, тестування.

The article offers operators use a modified initialization and ant crossing and genetic algorithms to solve the transport problem in tourism. By analyzing the behavior of ant colonies, such as finding the shortest route through the provision of pheromone function and crossing two solutions genetic algorithm developed methods and algorithms such operations: search for the optimal route, costing resources, search distance, time, route, storing executed routes. In the present work description created system for mobile phones operating system IOS, which performs all above listed transactions. Testing mobile app by "At first test".

Key words: ant colony genetic algorithm, hiking trails, a mobile application testing.

Вступ

Останніми роками для розв'язування комбінаторних задач оптимізації все ширше застосовують алгоритми обчислювального інтелекту. Такі алгоритми мають низку переваг – практичне застосування, гнучкість у налаштуванні, дають змогу отримати доволі ефективні результати, які допомагають знайти оптимальне розв'язування за короткий проміжок часу. До таких алгоритмів належать мурашині та генетичні алгоритми.

Метою статті є об'єднання мурашиних та генетичних алгоритмів у гібрид із власною модифікацією для того, щоб збільшити ефективність роботи нового алгоритму для розв'язування транспортних задач саме для туристичної сфери [1–3].

Також однією зі сфер дослідження є пошук оптимального розподілу станцій, на яких скупчуватимуться туристи, так, щоб витрати ресурсів не впливали негативно на ефективність знаходження оптимального шляху й одночасно були раціональними. Під час дослідження генетичного алгоритму виникла потреба використання функції схрещування для оптимізації пошуку оптимального маршруту для туристичної сфери.

Актуальність проблеми

Потреби людей у перевезенні збільшуються разом із розвитком інфраструктури держави загалом. Услід за цим зростає актуальність подорожей. Цьому можна знайти цілком логічне

пояснення. По-перше, інфраструктура населених пунктів і сполучень між ними розширюється, дороги постійно оновлюються. По-друге, перевезення автотранспортом здійснюються доволі швидко і завжди існуватиме потреба пришвидшувати процес перевезення. Тому актуальною є необхідність постійно формувати та створювати умови для зручного й ефективного виконання транспортних задач із використанням сучасних технологій, які істотно полегшать визначення маршрутів перевезення та оптимізують всі витрати ресурсів і знайдуть оптимальний шлях.

Сьогодні пошук найкоротшого маршруту вже розроблено, проте таке розв'язування не завжди доцільне, зокрема враховуючи специфіку туристичної діяльності. Тому знаходження найкоротшого шляху не завжди є ефективним вирішенням, особливо у сфері туризму. Основним завданням задачі цього типу є автоматизація маршрутів, згідно із потребами максимально швидко та ефективно доправити туристів до місць зупинок. Потрібно мінімізувати витрати транспорту на виконання поставлених завдань, враховуючи: дороги, туристичні місця, зупинки, посадку та висадку туристів. Також необхідно врахувати типи транспорту та їх можливості, а саме: кількість посадкових місць, швидкість транспорту.

Автомобільний транспорт є найпоширенішим засобом пересування. Це і перевезення людей автобусами, й оренда переважно легкових автомобілів для особистої подорожі та відпочинку. Особливо популярні автобусні тури для відвідування декількох міст, визначних місць, туристичних баз.

Транспортні перевезення, туристичні бази відпочинку, інформаційні системи і технології у туризмі тісно пов'язані між собою. Тому, використовуючи всі перераховані аспекти, можна створити якісну систему обслуговування туристів. Мобільні технології мають велике значення у інформаційних системах, які використовують у транспортній сфері. Існує тенденція використання мобільних пристроїв для виконання маршрутів, які вже поширені у суспільстві. Тому розроблення мобільного додатка, який адаптує процес обслуговування туристів, є актуальним, доступним, потрібним та, що важливо, зручним у застосуванні інструментом.

Розроблення такої системи може допомогти туристичним фірмам та туристам скласти подорож чи тур правильно і грамотно та максимально ефективно розподілити свій час, витрати та розклад туру чи власної подорожі.

Об'єкти та методи дослідження

Під час реалізації цих методів використовується моделювання та проектування із обмеженням за часом та витратами, основане на моделюванні колективного інтелекту, до них належать: метод мурашиних колоній (Ant Colony Optimization, ACO), метод схрещування двох рішень генетичного алгоритму (Crossbreeding solutions genetic algorithm, CSGA), модифікації операторів (modification operators, MO) та інші методи. Ці методи вже ефективно застосовуються для розв'язання різних задач: ACO застосовується для розв'язання задач пошуку найкоротшого маршруту, CSGA використовується для розв'язування задач транспортування, MO застосовується для кластеризації даних та об'єктів. У сукупності перераховані вище методи застосовують для створення гібридного алгоритму.

Аналіз літературних даних та постановка проблеми

Під час розроблення нового алгоритму необхідно було зрозуміти роботу мурашиного алгоритму, а саме як колонії мурах, щоб забезпечити себе їжею, шукають найкоротший шлях між гніздом та джерелом їжі без видимих активних механізмів координації. Дослідження виявили хаотичну діяльність мурах, проте як тільки джерело їжі було знайдено, все більше і більше мурах рухалось організовано по найкоротшому шляху. Більшість різновидів мурах використовують непряму форму контакту, через феромонні сліди. Щоб збільшити ефективність алгоритму роботи, а саме процесу пошуку найкоротшого маршруту між локальними точками, ми використали феромонні сліди. Бажаним результатом є запам'ятовування оптимальних маршрутів та швидкий розрахунок шляхів із більшою концентрацією феромонів [4–6].

Другим алгоритмом у дослідженні та проектуванні модифікованого алгоритму вибрано генетичний алгоритм, що містив у собі адаптивні методи пошуку, які й сьогодні часто використовують для розв'язування транспортних задач функціональної оптимізації. Вони основані на генетичних процесах біологічних організмів [6–9].

Генетичний алгоритм працює із сукупністю "осіб" – популяції, кожна з яких являє собою вирішення певної проблеми. Кожна особа оцінюється мірою її "пристосування" та оцінки ефективності виконання її роботи. В роботі використовується функція схрещування агентів. Такий підхід дає можливість ефективніше обслуговувати туристів та безпосередньо автоматизує цей процес, так, що кожен агент міститиме в собі усі попередні результати виконаних маршрутів. Це дасть змогу не шукати розв'язування знову, а використати вже знайдене розв'язування. Це пришвидшить роботу створеної програми.

Завданням дослідження є створення гібридного алгоритму та системи, які значно покращать ефективність виконання транспортних завдань у сфері туризму.

Постановка цілі та задач

На основі системного аналізу предметної області та сучасних технологій, що використовуються для розв'язування транспортних задач, а також алгоритмів і методів реалізації таких задач була сформована мета статті.

Мета статті – підвищити ефективність перевезення між містами та раціональне розміщення пунктів збору туристів із урахування раціонального використання ресурсів, таких як: витрати пального, вибір типу транспорту, кількість місць для стояння та посадкових у транспорті, витрати часу на перевезення, а також розробити мобільний додаток із інтуїтивно простим і зрозумілим інтерфейсом.

Для досягнення цілі у статті поставлено такі завдання:

1. Проаналізувати сучасні технології, що використовуються для розв'язування транспортних задач, та сформулювати базову постановку задачі із перевезення туристів між населеними пунктами із урахуванням раціонального використання ресурсів.

2. Розробити алгоритми для розв'язування пошуку оптимального маршруту між містами, розміщення пунктів збору туристів та завантаженості транспорту туристами відносно кількості посадкових місць у транспорті.

3. Розробити мобільний додаток на основі розроблених алгоритмів розв'язування транспортної задачі у сфері туризму із урахуванням розміщення пунктів збору і туристів у транспорті.

4. Дослідити ефективність виконання алгоритму у вигляді числового експерименту, а також перевірити ефективність роботи мобільного додатка методом "спочатку тест".

Матеріали та методи дослідження алгоритмів

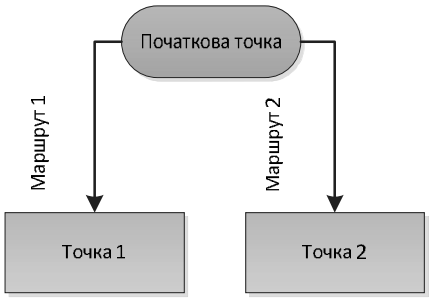

Рис. 1. Напрямки руху агентів

Першочерговим питанням дослідження стало вивчення основних функцій мурашиного алгоритму. Рух мурах являє собою повний неорієнтований граф. Кожне ребро має вагу, яка позначається як відстань між двома вершинами, з'єднаними між собою. Граф має два напрямки, тому мураха (агент) може рухатись по ребру в будь-якому напрямку (рис. 1).

Ймовірність залучення агента в маршрут пропорційна до кількості виділеного феромону на ребрі, а кількість виділеного феромону пропорційна до довжини маршруту. Тому можна зробити висновок, що чим коротший маршрут, тим більша кількість виділеного феромону p буде на ребрі та більша кількість мурах використає його, синтезуючи власний маршрут. Моделювання такого підходу, який використовує тільки додатний обернений зв'язок, приводить до передчасної збіжності – більшість

агентів x рухається по локально-оптимальному маршруту, тобто по ребрі w_{ij}, та виділяє феромони, які позначаються $\frac{Q}{L_x(t)}$, де змінна Q є константою, якій присвоюється число (концентрація) виділення феромону для відмічування пройденого маршруту мурахою; L_x – відстань, яку пройшла мураха; змінна t – час випаровування феромону. Якщо феромон швидко зникає, то це призводить до втрати пам'яті колонії та оптимальних рішень, але, з іншого боку, довготривале випаровування феромону дає можливість отримати стійке оптимальне розв'язування, яке матиме вигляд:

$$w_{ij}(t+1) = (1-p)w_{ij} + \sum_{x \in w_{ij}} \frac{Q}{L_x}. \quad (1)$$

Після того як шлях мурахи завершений, а ребра оновлені відповідно до довжини шляху і відбулося випаровування феромонів на всіх ребрах, тоді алгоритм запускається знову. Приклад наведено на рис. 2. Проте для розв'язання однієї із поставлених задач потрібно використати подвійне навантаження на граф. Додатковим навантаженням на локальні точки будуть туристи. Тому виникає залежність навантаженості транспорту (агента) від туристів. Якщо агент не має можливості задовольнити потреби всіх туристів, то програма викликає наступного агента доти, доки потреби туристів не будуть задоволені.

Роботу функцій мурашиного алгоритму можна описати так:

1. Здійснюється ініціалізація, тобто введення початкових даних (кількість транспорту, кількість туристів).
2. Вибір маршруту та розташування локальних точок на карті.
3. Якщо оптимальний маршрут вже існує, то програма повідомляє про це користувача й одразу видає результат, якщо ні – відбувається перехід до наступного кроку.
4. Вибирають першого агента.
5. Агент шукає шлях від однієї точки до іншої.
6. Рухаючись по карті, агент виділяє феромонні сліди.
7. Якщо маршрут задовільний, то розраховується нанесений феромон і видається результат.
8. Якщо маршрут незадовільний, то до попереднього агента приєднується ще один агент.

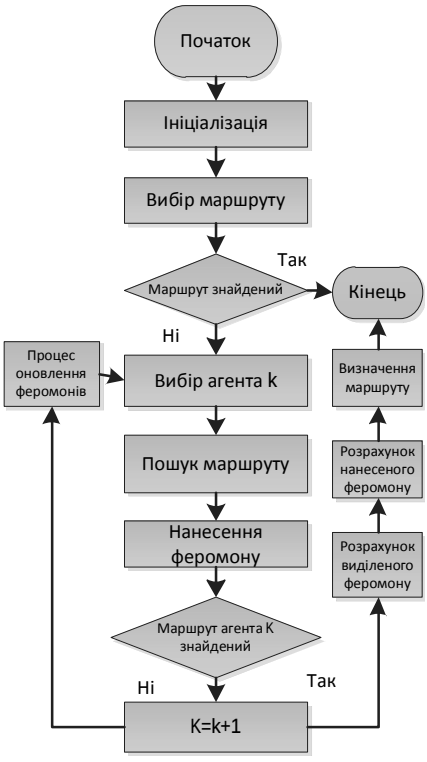

Рис. 2. Блок-схема мурашиного алгоритму

Класичний генетичний алгоритм запропонував Дж. Голланд як алгоритм, який імітує адаптацію популяції до заданої функції пристосування. Під час проектування гібридного алгоритму виникло завдання визначити найкращу структуру або значення параметрів об'єктів (маршрутів, агентів). Така задача називається оптимізаційною. Якщо оптимізація поєднана із розрахунком оптимальних значень параметрів за заданою структурою об'єкта, то її іменують параметричною оптимізацією. Задача вибору оптимальної структури є структурною оптимізацією. Основним елементом генетичного алгоритму вибрано евристичний алгоритм пошуку, який використовується для розв'язування задач оптимізації та моделювання за допомогою випадкового підбору, комбінування і варіації параметрів із застосуванням певних механізмів, які нагадують біологічну еволюцію.

Відмінна особливість генетичного алгоритму – акцент на використання оператора “схрещування”, який здійснює операцію рекомбінації рішень, у цьому випадку заданих об’єктів, роль яких аналогічна до ролі схрещування у живій природі [9–12].

Розроблення функцій генетичного алгоритму, використовуваних у гібридному алгоритмі, складається із декількох етапів:

1. Підготовча частина – формування початкової популяції (початковий набір рішень). Сам алгоритм для формування може бути різним, проте використовувався випадковий процес, щоб охопити більшу область для виконання пошуку рішень.

2. Відбір – важливий етап в алгоритмі, відповідає за вибір напрямів розвитку усіх можливих маршрутів (популяції), тобто відбір відкидає розв’язування із низьким значенням пристосування, що сприяє покращенню середньої пристосованості всієї популяції.

3. Схрещування – етап, на якому відбувається створення нових рішень в популяції, яка пройшла через відбір. Особливість його в тому, що у разі використання схрещування беруть два чи більше можливих розв’язувань із популяції, а з них – складові частини і з’єднують у нове розв’язування, яке залишається у популяції.

4. Оцінювання рішень та зупинка алгоритму. Для розв’язування задачі необхідно застосувати генетичний алгоритм, оснований на самих розв’язуваннях. Тому застосовується підхід із кількістю створених популяцій. Також зупинка може відбутися завчасно, якщо вже є готове розв’язування.

Розроблення та опис роботи операторів ініціалізації та схрещування

Основним завданням модифікованого оператора ініціалізації є генерувати маршрути так, щоб вони були коректні, зберігаючи функції мінімізації шляху, який має пройти транспорт.

Для того, щоб знайти оптимальне розв’язування задачі перевезення туристів із m пунктів початку маршруту $A_1, A_2, \mathbf{K}, A_m$ в n пунктів кінцевих зупинок маршруту $B_1, B_2, \mathbf{K} B_n$, мінімізуємо витрати ресурсів на перевезення туристів. Визначимо через $c_{i,j}$ – витрати ресурсів на перевезення із i-го пункту початку маршруту в j-й пункт кінцевої зупинки, через a_i – кількість туристів в i-му пункті початку руху транспорту. Також через b_j позначимо потреби туристів у транспорті в j-му пункті кінцевої зупинки, через x_{ij} – кількість туристів, яких потрібно перевезти з i-го пункту початку маршруту до j-го пункту кінцевої зупинки. Математична модель пошуку розв’язування за допомогою операторів схрещування та ініціалізації матиме такий вигляд:

$$F(x) = \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij} x_{ij}, \tag{2}$$

за таких трьох умов:

$$\sum_{j=1}^{n} x_{ij} = a_i, i = \overline{1,m} \tag{3}$$

$$\sum_{i=1}^{m} x_{ij} = b_j, j = \overline{1,n} \tag{4}$$

$$x_{ij} \geq 0, i = \overline{1,m}; \; j = \overline{1,n} \tag{5}$$

Модифікований оператор ініціалізації має пам’ять, тобто базу даних збережених маршрутів, локальних точок, а також розрахунків, що дає змогу системі не витрачати ресурси на пошук нового маршруту, що збільшує ефективність та швидкість роботи мобільного додатка.

Для забезпечення роботи використаємо алгоритм, зображений на рис. 3.

Цей алгоритм можна описати так:

1. Задають змінні, які містять в собі номери груп клієнтів, вектор пунктів на карті, вектор потреб кожного клієнта, сумісність кожного транспортного засобу та безпосередньо самі локальні змінні.

2. Відбувається кластеризація пунктів, заданих на основі географічного методу локалізації. Як наслідок, один кластер міститиме в собі точки, відстань між якими буде мінімальною.

3. Виконується ітерація для кожного кластера.

4. Створюється i-й кластер.

5. Розміщується випадковий елемент кластера, а попередній елемент видаляється.

6. Виконується перевірка сумарної потреби клієнтів, які входять у новий маршрут. Якщо перевірка сумарної потреби клієнтів більша, то її неможливо додати у поточний маршрут.

7. Якщо сумарна потреба у попередньому пункті більша або ж така сама, то цикл повторюється.

Оператор схрещування працює так, щоб отримувати не тільки нові маршрути, а й зменшити їх кількість, що дає змогу також зменшити кількість транспортних засобів. Для цього застосуємо алгоритм, зображений на рис. 4.

Обґрунтуємо опис роботи алгоритму модифікованого оператора схрещування:

1. Задають вхідні параметри локальних точок маршрутів, які містять в собі область маршруту та параметри результату схрещування.

2. Об'єднання маршрутів, які містяться в обох батьків. Як наслідок, створюється карта маршрутів, в якій точки (міста) відвідувались n кількість разів.

3. Цикл виконується доти, доки в об'єднаному маршруті є хоча б один маршрут.

4. Заміщення маршрутів та видалення попередніх маршрутів, які були у батьків.

5. Перевірка на наявність аналогічного маршруту. Якщо такий маршрут знайдено, його пропускають.

6. Перевірка на сумарну потребу групи туристів. Якщо потреба може бути виконаною, то параметри поміщаються у новий маршрут, а якщо не виконана, то цикл повторюється до створення нового маршруту.

7. Якщо умови сумарної потреби групи клієнтів виконані, то цикл завершується. Якщо ні, то цикл повторюється знову.

Рис. 3. Блок-схема модифікованого оператора ініціалізації оптимізації туристичних маршрутів

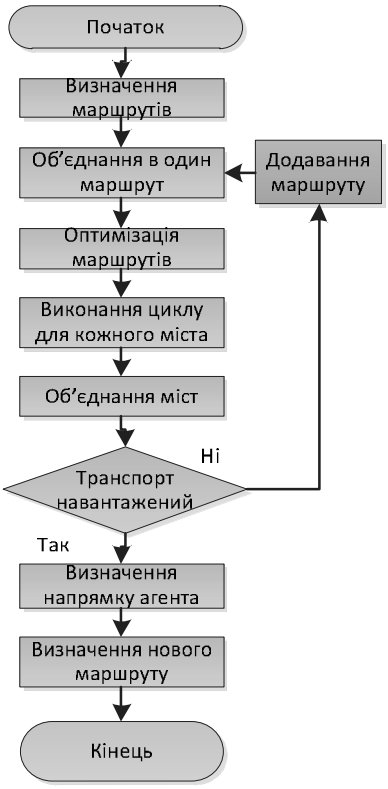

Рис. 4. Блок-схема роботи модифікованого оператора схрещування

Обґрунтування мобільного додатка під операційну систему IOS

Під час розроблення програмного забезпечення вибрано екстремальний метод програмування "спочатку тест". Отже, якщо програмний код проходить всі тести, то він є ефективним, що збільшує продуктивність роботи самого продукту і зменшує навантаження на пристрій, на якому запущений цей продукт. Під час розроблення виникали такі проблеми:

1. Великий обсяг пам'яті мобільного додатка – 400 Мб.

2. Залишок великої кількості інформаційного сміття після виконання циклів роботи мобільного додатка.

3. Велике навантаження на серверну частину.

4. Карта світу та її завантаження.

5. Складна ієрархічна система класів мобільного додатка.

6. Будується шлях по непрохідних областях (річках, озерах, океанах).

Для того щоб вирішити ці проблеми, команда розробників прийняла розв'язування, що частково покращили ефективність роботи мобільного додатка.

Великий обсяг пам'яті мобільного додатка, як для типу сервісу та обслуговування, спричиняє багато проблем, таких як: зменшення швидкості роботи додатка, поява помилок, завчасне припинення роботи додатка тощо. Більшу частину обсягу пам'яті додатка займала карта. Тому, скориставшись можливістю працювати із картою на сервері через з'єднання Інтернет, повністю було вирішено цю проблему.

Залишок великої кількості сміття у мобільному додатку після виконання циклів роботи зменшував швидкість роботи додатка і перешкоджав з'єднанню з мережею Інтернет. Для вирішення такої проблеми у програму був доданий код, мета якого – періодично чистити "сміття", а також кеш, щоб звільнити обсяг пам'яті, необхідний для нормальної роботи програмного забезпечення.

Карта світу та її завантаження пов'язані із проблемою великого навантаження на серверну частину. Для того, щоб мобільний додаток був легкий та швидкісний, необхідно було зменшити навантаження на саму систему. Але, з іншого боку, це навантаження ми переклали на серверну частину. Тому під час подальшого розвитку та розроблення програмного продукту намагатимемось вирішити проблему із навантаженням на сервер.

Найважливіший етап розроблення цього мобільного додатка – розроблення алгоритмів модифікованих операторів схрещування та ініціалізації, а також область їх застосування. Отже, записуючи алгоритм у програмний код, довелося створити ієрархічні системи класів. Такі системи важко протестувати і процес є довготривалим, тому що помилка в такому випадку може призводити до помилок у роботі додатка під час виконання другорядних дій.

Опис класів та тестування програмного забезпечення

Класи та тестування мобільного додатка можна описати так:

1. Клас TravellingSalesmanAlgorithm містить логічні структури, вбудовані у програму, які розмічують популяцію, схрещування батьків та нащадків f цього алгоритму. Входження даних у цей клас визначає список точок x, які визначені попередньо на стадії ініціалізації. Робота цього класу – це робота з класом Location, видає оптимальне розв'язування для цих точок. Цю фітнес-функцію F використовуємо для оптимізації роботи задачі

$$F(x_{i,j}) = \frac{1}{f(x)} \quad . \tag{6}$$

Якщо фітнес-функція набуває значення 1, тоді шлях від міста i до міста j є визначений, в іншому випадку функція набуває попереднього значення. Цей клас взаємодіє із класом Location, а саме передає результат отриманих даних на опрацювання оптимальності шляху.

Клас Location є моделлю даних, який містить в собі функцію для знаходження дистанції L між двома парами локальних точок m_1, m_2 та n_1, n_2, тобто використовує формулу (1):

$$L = \sqrt{(m_1 - m_2)^2 + (n_1 - n_2)^2} = \sqrt{(n_2 - n_1)^2 + (n_2 - n_1)^2} . \quad (7)$$

Клас вираховує повну дистанцію через всі точки, а також змінює індекси між точками для коректної роботи алгоритму. Тестування пошуку маршруту Чернівці–Київ показало, що цей клас будує шлях доти, доки пройдена дистанція і похибка не набудуть найменшого значення. Приклад тестування продемонстрований у таблиці.

Результати тестування пошуку оптимального маршруту

Чернівці-Київ	Пройдена дистанція, км	Похибка, км
Ітерація № 1	524.86125	–1.890507
Ітерація № 2	514.60852	–2.588139
Ітерація № 3	515.12161	–0.116215
Ітерація № 4	538.03895	–1.549931
Ітерація № 5	534.78239	–2.258549
Ітерація № 6	534.09532	–3.000126
Ітерація № 7	537.51959	–0.335941
Ітерація № 8	549.80766	–1.615849
Ітерація № 9	548.92406	–2.923222
Ітерація № 10	559.58426	–3.186893
Ітерація № 11	558.62982	–4.263554
Ітерація № 12	514.88224	–3.186893
Ітерація № 13	514.60852	–0.116215

Використання проектованої системи оптимізації маршрутів туризму проюлюстровано на рис. 5.

Клас ViewController забезпечує роботу системи відображення всіх отриманих даних візуально, тобто відповідає за візуалізацію програмного забезпечення загалом. Він працює із отриманим результатом роботи двох попередніх класів для відображення роботи алгоритму на карті. Всі навантаження на граф розраховують за пріоритетністю кожної складової цього навантаження, а саме: обчислюють місткість транспорту, що обслуговує туристів, щодо його можливостей. Також цей клас розраховує використання палива і шукає оптимальну дистанцію.

На сторінці головного меню системи можна вибрати локальні точки відвідування та кількість туристів у цих точках, тобто міста, між якими будуватиметься найкоротший маршрут. Натиснувши на ліву кнопку біля рядка введення міста, можна ввести кількість туристів, а якщо натиснути на кнопку “Обрати”, відкриється сторінка вибору міста, на якій ми зможемо підтвердити наш вибір. Після того як локальні точки введені, можемо побудувати маршрут, на якому будуть позначатись дороги від одного міста до іншого різними кольорами. Під картою будуть описані підказки в дорозі, відстань, час поїздки, кількість туристів, а також витрати пального.

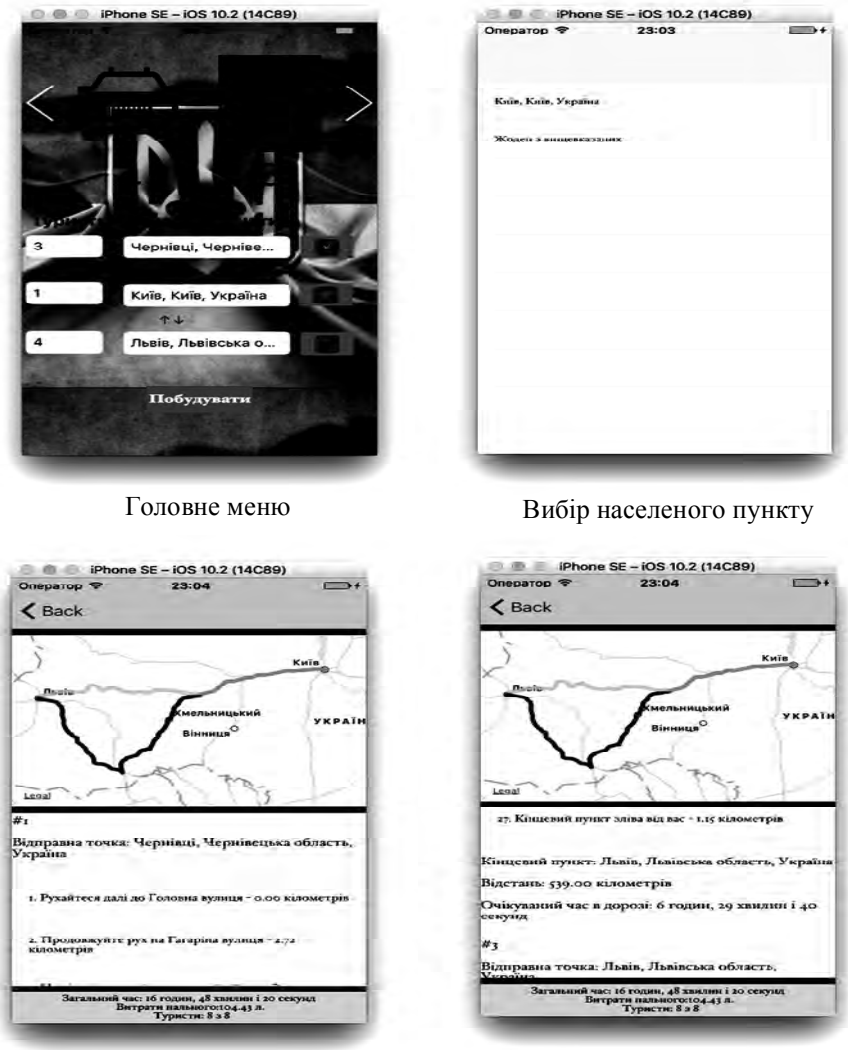

Рис. 5. Екранні зображення проектованої системи оптимізації маршрутів туризму

Висновок

Проаналізовано сучасні методи оптимізації маршрутів, що використовуються для розв'язування транспортних задач, за допомогою яких спроектували розв'язування з перевезення туристів між населеними пунктами із урахуванням раціонального використання ресурсів. До таких методів належать генетичний та мурашиний алгоритми, за допомогою яких виконуються пошук оптимального маршруту та розстановка пунктів збору.

На основі створених модифікацій операторів ініціалізації та схрещування побудовано систему, яка розв'язує транспортну задачу в сфері туризму із урахуванням розміщення пунктів збору, а також туристів у транспорті.

Під час дослідження ефективності виконання алгоритму у вигляді числового експерименту та перевірки ефективності роботи мобільного додатка методом "спочатку тест" було виявлено, що система будує маршрут від одного населеного пункту до іншого, у якого похибка та дистанція найменші.

1. Кажаров А. А. Мурашині алгоритми для вирішення транспортних задач / Кажаров А. А., Курейчик В. М. // Російська академія наук. Теорія і системи управління. – 2010. – С. 32–45.

2. Ємельянова Т.С. Розв'язування еталонних транспортних задач з кластерним розташуванням клієнтів із використанням генетичних алгоритмів / Т. С. Ємельянова // Нечіткі системи і обчислення (НСМВ-2008): наукова конф. з міжнар. участ. – 2008. – С. 195–199. 3. Гладков Л. А. Генетичні алгоритми: навч. посіб. / Л. А. Гладков, В. В. Курейчик, В. М. Курейчик. – М .: Фізмат, 2006. – С. 320. 4. Горячев Ю. В. Генетичні алгоритми багатокритеріальної конфліктної оптимізації./ Ю. В. Горячев. – М.: 2001. – С. 102. 5. Курейчик В. В. Застосування генетичного алгоритму розв'язання задачі тривимірної упаковки / В. В. Курейчик, Д. В. Заруба, Д. Ю. Запорожець // Новини ПФУ. Технічні науки. – 2012. – С. 8–14. 6. Бова В. В. Інтегрована підсистема гібридного і комбінованого пошуку в задачах проектування та управління / Бова В. В., Курейчик В. В. // ПФУ. Технічні науки. – 2010. – С. 37–42. 7. Курейчик В. М. Пошукова адаптація: теорія і практика / В. М. Курейчик, Б. К. Лебедєв, О. К. Лебедєв. – М.: Фізмат, 2006. – С. 272. 8. Розробка і аналіз генетичного та гібридного алгоритму для розв'язування задач дискретної оптимізації / А. В. Єрмеєв: автореф. дис. … канд. тех. наук. – Омск, 2000. – С. 22. 9. Гвоздєв С. Е. Математичне програмування / С. Е. Гвоздєв // Новосибірськ: НГАСУ – 2001. – С. 96. 10. Бобарикін В. А. Математичні методи розв'язування автотранспортних задач / В. А. Бобарикін // СЗПІ. – 1986. – С. 83. 11. Алєксєєв А. О., Транспортна задача по критерію часу при обмеженій кількості транспортних ресурсів / А. О. Алєксєєв // Математичні методи оптимізації і управління в складних системах. КГУ. – 1984. – С. 60–65. 12. Верховський Б. С. Задачі лінійного програмування типу транспортних / Б. С. Верховський // ДАН СССР. – 1963. – Т. 151. – № 3. – С. 515–518.