УДК 004

РОЗРОБКА ТА АНАЛІЗ ІНФОРМАЦІЙНОЇ СИСТЕМИ СУПРОВОДЖЕННЯ РОБОТИ МАРШРУТНО-КВАЛІФІКАЦІЙНИХ КОМІСІЙ З ПЛАНУВАННЯ ГІРСЬКИХ ТУРИСТИЧНИХ МАРШРУТІВ

Кравченко В. Д., Решетнік В. М.

Харківський національний університет радіоелектроніки, Харків

Популярність гірського туризму щороку зростає, але питання забезпечення безпеки походу часто залишається поза увагою туристів. Одним із ключових елементів забезпечення безпеки походу є його реєстрація, що здійснюється через маршрутно-кваліфікаційні комісії (МКК), які здійснюють перевірку маршрутів, реєстрацію походів та видачу маршрутних книжок. Через відсутність зручних механізмів оформлення багато туристів уникають реєстрації. Реєстрація здійснюється особисто в МКК або через електрону пошту, що є тривалим й незручним процесом. Також визначення маршруту, який проходить через обрані точки інтересу з урахуванням максимального часу походу є складною задачею для туристів незалежно від їх рівня підготовки. Розв’язанням проблем може стати спеціалізована веб-платформа, що забезпечить оперативну взаємодію між туристами та МКК і надасть інструменти для планування маршрутів на основі вибраних точок інтересу й обмеження за часом походу.

Головним завданням системи є забезпечення побудови маршрутів за критеріями користувача та підвищення зручності процесу реєстрації та безпеки туристів. Система надає функціонал для подання заявок на реєстрацію походів, отримання консультацій стосовно маршрутів, формування звіту з проведеного походу та додавання нових точок і ділянок шляху для подальших планувань маршрутів.

Побудова маршруту здійснюється на основі місць, які хоче відвідати користувач, місця старту й фінішу маршруту та часу для його проходження.

Кожне місце (точку) маршруту користувач самостійно оцінює певним ваговим коефіцієнтом (балом), що відповідає зацікавленості у його відвідуванні. Маршрут, який проходить через певну кількість точок з ваговими коефіцієнтами, отримує оцінку, що дорівнює сумі балів вагових коефіцієнтів та відповідає зацікавленості користувача. Система визначає оптимальний маршрут, що буде проходити через усі або частину заданих користувачем точок, тривалість якого не перевищує встановлений час та максимізує його оцінку. Маршрут будується з використанням мапи, що зберігається у вигляді графу, де вершини – це перехрестя доріг, ребра – дороги між ними. Для кожного ребра зберігається інформація про час проходження та довжину. Для визначення маршрутів між вказаними користувачем точками можна використовувати алгоритми пошуку шляху на графі, але для їх використання користувач має самостійно визначити точки, через які буде проходити маршрут, а також послідовність їх відвідування. Визначення точок, які будуть у маршруті, та послідовність їх обходу з використанням обмеження за часом виконується шляхом розв'язання задачі орієнтування, але її розв'язання виконується на повних графах, в якому усі вершини – точки, які хоче відвідати користувач. Для пошуку оптимального маршруту з урахуванням встановлених обмежень використовується двоетапний підхід. На першому етапі визначаються маршрути з мінімальним часом проходження між кожною парою визначених користувачем точок. На другому етапі визначається порядок відвідування точок шляхом розв'язання задачі орієнтування на основі цих даних. Оптимальним маршрутом вважається той, який забезпечить найбільшу оцінку [1]:

$$\mathrm{R}_{best} = \max \sum_{i=1}^{n} \sum_{j=1}^{n} S_i x_{ij}$$

За умови:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} t_{ij} x_{ij} \leq T_{max}$$

Де R_{best} – оцінка маршруту, S_i – ваговий коефіцієнт точки, n – кількість вершин у графі, x_{ij} – бінарна змінна, яка позначає наявність ребра ij в маршруті, t_{ij} – час переходу між вершинами ij, який визначається на першому етапі, T_{max} – максимальна тривалість маршруту. Задача орієнтування розв'язується на повному графі, в якому вершини – точки, які хоче відвідати користувач, вага ребер дорівнює t_{ij}. Остаточний маршрут визначається шляхом обходу точок в послідовності, знайденій під час розв'язку задачі орієнтування, з використанням маршрутів, знайдених на першому етапі.

Для пошуку найкоротшого шляху у графі розглядалися 3 алгоритми: алгоритм Дейкстри, двоспрямований алгоритм Дейкстри та алгоритм A*, які порівнювалися за часом виконання. Ці алгоритми було обрано, оскільки їх реалізація є в стандартних бібліотеках мови програмування Python, яка використовувалась для розробки системи. Найгіршим алгоритмом виявився алгоритм Дейкстри, який працював в середньому на 20% довше. Різниця у часі виконання двох інших алгоритмів є незначною.

Розв'язок задачі орієнтування можна виконувати з використанням точних та наближених методів. Задача орієнтування є NP-складною, тому точні алгоритми можуть витрачати велику кількість часу на пошук розв'язку. Для отримання наближеного розв'язку задачі використовують генетичні алгоритми, алгоритми локального пошуку та алгоритми колективного інтелекту [2]. В роботі розглядаються 1 точний та 2 наближені алгоритми з різною стратегією пошуку рішення. У якості точного алгоритму обрано алгоритм пошуку в глибину, оскільки він дозволяє просто врахувати обмеження за часом в порівнянні з іншими. У якості наближених алгоритмів обрано мурашиний алгоритм та алгоритм імітації відпалу. Мурашиний алгоритм є алгоритмом на основі колективного інтелекту, який імітує поведінку мурах при пошуку найкращого шляху. Алгоритм обрано, оскільки він дозволяє досліджувати великі простори рішень, виконує пошук маршруту шляхом пересування агента графом, і може бути модифікований в залежності від задачі [1]. Алгоритм імітації відпалу є алгоритмом локального пошуку.

Алгоритм обрано, оскільки він уникає локальних максимумів за рахунок можливості прийняття гіршого рішення, ніж поточне найкраще рішення [3]. Порівняння алгоритмів виконувалося за часом їх роботи та нагородою для фінального рішення. Алгоритми порівнювалися на графах, що містять від 5 до 40 вершин (з кроком у 5 вершин), не враховуючи стартову та фінішну вершину. Дослідження проводилося для трьох сценаріїв, які відрізняються кількістю вершин в найкращому рішенні: 5–25% вершин, 40–60% вершин та 75–95% вершин. Великий проміжок між нижньою та верхньою межею кількості вершин забезпечує можливість дослідження усіх сценаріїв на графах з малою (меншою за 20) кількістю вершин та спрощення підбору параметрів запуску для великих графів. Результати для сценарію 40-60% вершин подано на рисунку 1. Алгоритми зупинялись примусово, якщо час виконання перевищував 10 секунд.

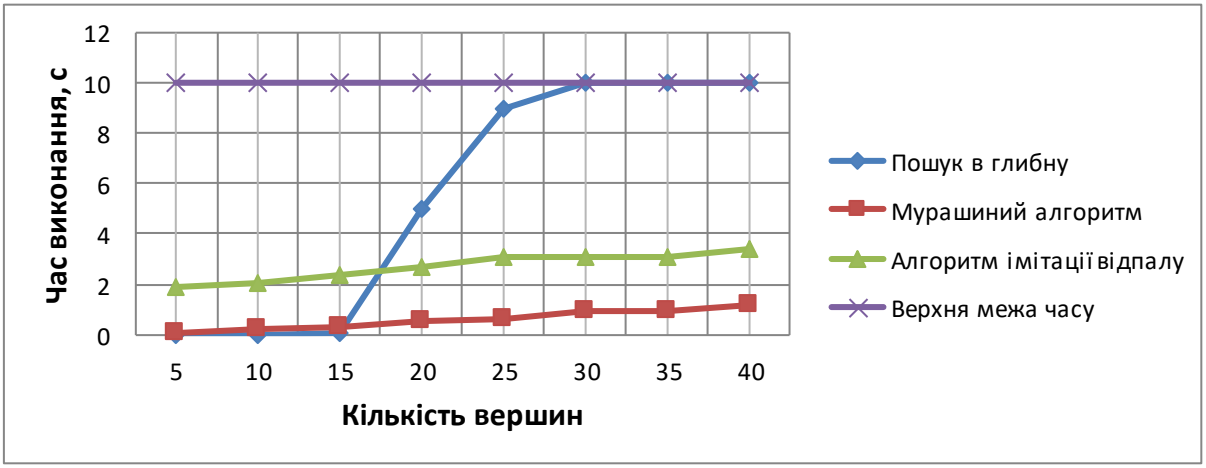

Рисунок 1 – Залежність часу виконання від кількості вершин

Час виконання пошуку в глибину різко зростав для графів з більше ніж 15 вершинами, час виконання мурашиного алгоритму та алгоритму імітації відпалу зростав повільно при збільшенні кількості вершин, алгоритм імітованого відпалу в середньому працював на 2 секунди довше. Оцінка найкращого знайденого маршруту R_{best} для кожного алгоритму на кожному графі подано у таблиці 1. Для інших двох сценаріїв отримано схожі

результати: форма кривих залишалася подібною, змінювалися лише різниця часу виконання між двома останніми алгоритмами та кількість вершин у графі, для якої час виконання пошуку в глибину перевищував 10 секунд.

Таблиця 1 – Залежність R_{best} від кількості вершин у графі

	5	**10**	**15**	**20**	**25**	**30**	**35**	**40**
Пошук в глибину	**214**	**271**	**526**	**748**	**832**	**-**	**-**	**-**
Мурашиний алгоритм	**214**	**271**	**526**	**735**	**808**	**988**	**876**	**1139**
Алгоритм імітації відпалу	**214**	**271**	**526**	**733**	**773**	**913**	**867**	**1130**

На основі проведеного аналізу для побудови маршруту з мінімальним часом проходження між заданими користувачем точками рекомендується використовувати двоспрямований алгоритм Дейкстри, оскільки він не потребує використання евристичних функцій для виконання пошуку. Для визначення точок, які будуть присутні у маршруті та порядку їх обходу, пропонується використовувати алгоритм пошуку в глибину, якщо заданих точок менше 11, оскільки далі час виконання цього алгоритму суттєво збільшиться. Якщо заданих точок більше 11, доцільно використовувати мурашиний алгоритм, оскільки час його роботи менший за час роботи алгоритму імітаційного відпалу, а оцінка маршруту є вищою.

Література:

1. Liang Y.-C., Smith A. E. AN ANT COLONY APPROACH TO THE ORIENTEERING PROBLEM. Journal of the Chinese Institute of Industrial Engineers. 2006. Vol. 23, no. 5. P. 403–414.
2. Karbowska-Chilinska J., Zabielski P. A Genetic Algorithm vs. Local Search Methods for Solving the Orienteering Problem in Large Networks. Lecture Notes In Computer Science. 2012. No. 7828.

3. M.; ASKERZADE B. Simulated annealing approach for solving at Time Dependent Orienteering Problem. Communications Faculty Of Science University of Ankara. 2016. Vol. 58, no. 1. P. 17–28.