КОМПЬЮТЕРНАЯ МАТЕМАТИКА

УДК 519.8

Л.Ф. ГУЛЯНИЦЬКИЙ, В.В. СТОРЧЕВИЙ

# ОПТИМІЗАЦІЯ МАРШРУТІВ ГРУПИ БПЛА МОДИФІКОВАНИМ АЛГОРИТМОМ МУРАШИНИХ СИСТЕМ

*Розглянуто проблему маршрутизації групи безпілотних літальних апаратів (БПЛА) при обстеженні заданих об'єктів. Показано, що модель руху транспортних засобів з декількома депо можна використати для планування польоту групи БПЛА. Розроблено спеціальний макс-мін алгоритм мурашиних систем для розв'язування однієї із таких задач маршрутизації з декількома депо. Проведено порівняльний аналіз трьох алгоритмів розв'язування.*



**Вступ.** Безпілотні літальні апарати (БПЛА) набувають все більш широкого використання в різних сферах, зокрема, у військовій для ведення повітряної розвідки – як тактичної, так і стратегічної. Ефективним варіантом їх застосування є вирішення завдань у складі групи [1 – 4]. У роботі досліджується задача математичного моделювання маршрутів групи БПЛА, для розв'язування якої пропонується спеціальний алгоритм мурашиних колоній, оскільки ефективність мурашиних алгоритмів підтверджується досвідом розв'язування різних задач маршрутизації транспорту (Vehicle Routing Problems, VRP) [5 – 8]. Для дослідження ефективності розробленого алгоритму проведено обчислювальний експеримент, у якому порівнювалися результати його застосування із алгоритмами спеціального локального пошуку і повного перебору.

**1. Постановка задачі пошуку об'єкта**

Розглядається проблема пошуку оптимальних маршрутів для групи спеціальних літальних апаратів, у першу чергу, БПЛА. Вона полягає у тому, що перед заданою групою БПЛА, які можуть стартувати з різних точок пуску та мають можливість закінчувати маршрут у різних місцях (депо), стоїть завдання облетіти низку заданих об'єктів (точок на місцевості) з мінімізацією сумарної довжини маршрутів чи тривалості польотів за умов, що кожен об'єкт відвідується одним і лише одним БПЛА і всі об'єкти мають бути відвіданими. При цьому часто слід враховувати ще й ряд додаткових обмежуючих умов (дальність польоту без підзаряджання чи дозаправлення, вантажопідйомність, погодні умови тощо). Отже, приземлення БПЛА може здійснюватися в декількох заданих територіально зонах. Припускається також, що характеристики кожної зони вибираються так, що у конкретній зоні можливе приймання заданої кількості БПЛА: у такій зоні можуть завершувати свій маршрут всі чи частина БПЛА. Кожну зону будемо ідентифікувати її центроїдом. Отже, маршрут кожного із БПЛА повинен закінчуватися в одній із таких зон, але конкретна зона для БПЛА не вказується. Цю проблему назвемо задачею маршрутизації з альтернативними депо. Кількість місць старту БПЛА може перевищувати кількість самих БПЛА – це створює додаткові можливості для вибору місць (чи моментів) старту. Також відмітимо, що в процесі розв'язування задачі маршрутизації може оптимізуватися також і кількість задіяних БПЛА.

*Задача*. За умови наявності декількох можливих зон приймання БПЛА і місць старту БПЛА, а також завершення маршрутів БПЛА лише в одній із виділених зон слід визначити як маршрути БПЛА з оптимізацією сумарної довжини (тривалості польотів) та їх задіяної кількості, так і місця (зони) їх приймання. Зауважимо, що в одній зоні може завершуватися декілька маршрутів БПЛА.

Змістовна постановка задачі маршрутизації груп БПЛА для пошуку об'єктів може бути подана так. Задано координати місць старту, кількість БПЛА, координати точок, які необхідно відвідати, координати зон приймання, польотний ресурс кожного БПЛА, ємність зон приймання.

Критерієм виступатиме мінімум сумарної довжини маршрутів польоту для виконання завдання, що і визначатиме час виконання завдання; мінімум БПЛА.

**2. Математична модель**

Вважаємо заданими:

$m$ – загальна кількість БПЛА;

$b$ – кількість місць старту ($b \geq m$);

$e$ – кількість зон приймання БПЛА ($e \geq 1$);

$n$ – кількість об'єктів, які слід відвідати;

$R^k$ – польотний ресурс *k*-го БПЛА, $k = 1, ..., m$.

Підкреслимо, що кількість місць старту БПЛА не може бути меншою $m$, а в разі, коли $m > n$, частина із них не буде задіяною.

Розглянемо зважений граф $G = (Y, U)$, де вершини $Y$ відповідають об'єктам, місцям старту та зонам приймання, тобто $\|Y\| = n + b + e$, задавши таку їх нумерацію: від 1 до $n$ – об'єкти, від $n+1$ до $n+b$ – місця старту, а від $n+b+1$ до $n+b+e$ – зони (їх центроїди).

Позначимо $V = \{1, ..., n\}$ множину вершин, які відповідають об'єктам, $B = \{n+1, ..., n+b\}$ – множину вершин, які відповідають місцям старту, $E = \{n+b+1, ..., n+b+e\}$ – вершини, що відповідають зонам приймання. Зрозуміло, що для вершин із множини $B$ матимемо лише вихідні дуги, а для вершин із $E$ – лише вхідні.

Для подання розв’язку введемо матрицю $X=(x_{ij})$ розмірністю $m\times(n+b+e)$, де $x_{ij}\in\{0,1\}$, яка складається із трьох блоків. Перший блок включає стовпчики від 1 до $n$, другий – від $n+1$ до $n+b$, третій – від $n+b+1$ до $n+b+e$, а рядки будуть відповідати номерам БПЛА. Ця матриця буде описувати фактично етап розбиття, на якому кожному БПЛА приписуються ті об’єкти, які він має відвідати, разом із одним із місць старту та місцем завершення маршруту.

Позначимо ¢$_j$ максимальну кількість БПЛА, які можуть бути прийняті в зоні $j$, $j=1,...,e$.

Задача розбиття (як окремий етап чи складова загального процесу розв’язування) може бути подана так.

Обмежуючі умови:

$$\sum_{j=n+1}^{n+b} x_{ij}\le 1,\quad i=1,...,m; \tag{1}$$

$$\sum_{j=n+b+1}^{n+b+e} x_{ij}\le 1,\quad i=1,...,m; \tag{2}$$

$$\sum_{i=1}^{m} x_{ij}=1,\quad i=1,...,n; \tag{3}$$

$$\sum_{i=1}^{m} x_{ij}\le 1,\quad j=n+1,...,n+b; \tag{4}$$

$$\sum_{j=n+1}^{n+b}\sum_{i=1}^{m} x_{ij}=m; \tag{5}$$

$$\sum_{i=1}^{m} x_{ij}\le \delta_{j-n-b},\quad j=n+b+1,...,n+b+e; \tag{6}$$

$$L^i(\overline{x_i})\le R^i,\quad i=1,...,m. \tag{7}$$

Задача полягає у пошуку такого розв’язку *X*, для якого

$$F(X)\equiv\sum_{i=1}^{m} L^i(\overline{x_i})\to\min, \tag{8}$$

за умов (1) – (7), де $\overline{x_i}$ – це $i$-й вектор-рядок матриці $X$, а $L^i(\overline{x_i})$ – довжина маршруту $i$-го задіяного БПЛА як розв’язку задачі комівояжера, який визначається вершинами, що відповідають ненульовим компонентам вектора $\overline{x_i}$, починаючи з певного стартового місця $s\in B: x_{is}=1$ і закінчуючи зоною $f\in E: x_{if}=1$.

Зрозуміло, що для незадіяних БПЛА відповідний вектор-рядок матиме всі нульові координати.

Тобто матриця $X$ , як елемент простору розв'язків задачі (1) – (8), у компактному вигляді може бути подана так:

$$X = \begin{pmatrix} \overline{x_1} \\ \vdots \\ \overline{x_m} \end{pmatrix},$$

де $\overline{x_i} = (x_{i1}, x_{i2}, ..., x_{i,n+b+e})$ .

Умова (1) визначає, що кожен і-й БПЛА стартує лише з одного із можливих місць старту (чи в момент часу, який і визначає таке місце), якщо ж деякий БПЛА не задіяний, то відповідний рядок стає нульовим. Аналогічно, формула (2) описує ситуацію, що до конкретного центроїда від об'єктів або веде одне і лише одне ребро графа *G*, або жодного. Умова (3) визначає, що кожен об'єкт $j \in V$ відвідується лише одним БПЛА. Умова (4) визначає, що з кожного місця або стартує один БПЛА, або ж це місце не задіяне, а разом із умовою (5) – що кількість задіяних місць співпадає із кількістю БПЛА. Можливість приймати в кожній зоні $j \in E$ (в іншому трактуванні – у момент часу) не більше, ніж $\delta_j$ БПЛА, відображена у формулі (6). Обмеження на польотний ресурс кожного літального апарату визначено умовою (7).

Зауважимо, що не виключаються випадки, коли в одній зоні (чи в момент часу) може прийматися лише один БПЛА, тоді $\delta_j = 1$ для відповідної зони, чи всі БПЛА можуть бути прийняті в одній зоні – в цьому разі $e = 1$, а $\delta_1 = m$.

Таким чином, ця задача подібна до задачі маршрутизації транспортних засобів із декількома депо (Multiple Depot Vehicle Routing Problem, MDVRP) з додатковими обмеженнями [1, 5].

MDVRP є більш складною задачею, ніж класична задача VRP. До того ж, MDVRP є NP-складною задачею, це означає, що не існує ефективного алгоритму для отримання оптимального її розв'язування. Точні методи, такі як метод гілок і меж, метод гілок і відтинання, є неефективними для розв'язування MDVRP через надмірну трудомісткість. Для того, щоб ефективно розв'язувати подібні задачі, використовуються наближені алгоритми [9].

**3. Алгоритми розв'язування задачі**

*Прямий перебір*. Найпростіший точний метод, який можна застосувати – це прямий перебір всіх можливих варіантів. Алгоритм повного перебору всіх можливих розв'язків задачі складається з чотирьох етапів. Перший етап – формування всіх можливих перестановок початкових депо. Для визначення кількості перестановок з $b$ елементів маємо формулу $P_b = b!$. Другий етап – розбиття множини з $n$ об'єктів, які необхідно відвідати. Загальна кількість поділів довільної *n*-елементної множини дорівнює числу Белла, яке можна обчислити як суму чисел Стірлінга другого роду, $B_n = \sum_{m=0}^{n} S(n,m)$ [10]. Третій етап – формування всіх можливих розміщень для $e$ зон приймання. Загальна кількість розміщень визначається за наступною формулою: $A_e^b = \frac{b!}{(b-e)!}$ . Останній етап – композиція всіх маршрутів отриманих на перших трьох кроках, загальна кількість варіантів $k = b!\sum_{m=0}^{n} S(n,m)\frac{b!}{(b-e)!}$. Через таку надзвичайно велику кількість варіантів можна зробити висновок, що методом повного перебору неможливо знайти оптимальний розв'язок за прийнятний час. Його застосування виправдане лише для задач невеликої розмірності для отримання точного розв'язку.

*Локальний пошук.* Алгоритми локального пошуку використовують поняття околу в просторі розв'язків задачі $X$. Варіант розв'язку складається з множини маршрутів – послідовності номерів клієнтів, які відвідуються кожним БПЛА. Кожен маршрут починається з стартового депо та закінчується зоною приймання. Наприклад, маємо два БПЛА. Перший стартує з місця старту № 1, відвідує клієнта № 4, потім клієнта № 3 та летить у зону приймання № 6. Другий починає рух з місця старту № 2, відвідує клієнтів № 5, № 7, № 8 та летить у зону приймання № 9, що відобразимо так:

$$\left\{ \begin{array}{c} \{1,4,3,6\} \\ \{2,5,7,8,9\} \end{array} \right\}.$$

Для генерації околів $O(x)$ довільної точки $x \in X$ застосовується оператор переміщення, який є одним з найбільш ефективних способів модифікації маршруту [11].

Для розв'язування сформульованої задачі пропонується два наближених алгоритми: табу пошуку та мурашиний алгоритм. *Табуйований пошук*, який довів свою ефективність при розв'язуванні ряду задач класу MDVRP [12], пропонується гібридизувати з конструктивним алгоритмом, який реалізує принцип жадібного вибору.

Псевдокод алгоритму подамо так.

**procedure TABU_SEARCH_WITH_GREEDY** (*x*);
  ініціалізація_алгоритму;
  $x :=$ припустимий варіант розв'язку, сформований жадібним алгоритмом;
  $T := \varnothing$;
  $x_{rec} := x$
  **while** не_виконується_критерій_завершення **do**
    пошук_прийнятного_варіанта $y \in O(x) \setminus T$ ;

$x := y$;
$T := T \cup x$;
**if** $f(x) < f(x_{rec})$ **then** $x_{rec} := x$ **;**
**if** $|T| >$ максимальний_розмір_табу_списку **then**
видалення_з_множини_T_найстарішого_елемента;
**endwhile**
**end**

*Макс-мін алгоритм мурашиних систем.* Більш інтенсивне використання кращих розв'язків у ході пошуку призводить до підвищення продуктивності алгоритмів оптимізації мурашиною колонією [13], але при цьому підвищується ризик передчасної збіжності. Тому доречно комбінувати використання кращих розв'язків та процедури запобігання передчасній збіжності. Для задоволення цих двох потреб розроблений спеціальний макс-мін алгоритм мурашиних систем як покращення алгоритму мурашиних систем [14]. Для уникнення стагнації розв'язку вводиться інтервал значень для феромону, тобто вводиться поняття нижньої та верхньої межі. На початку роботи алгоритму матриця феромонів ініціалізується значеннями нижньої межі феромону. Після кожної ітерації тільки один агент залишає слід феромону, зазвичай це агент, що знайшов кращий маршрут на поточній ітерації. Для розв'язування сформульованої задачі використовується модифікований макс-мін алгоритм мурашиних систем. Розв'язок будується покроково: поточна мураха вибирає наступну вершину графа задачі, після переходу в яку ця вершина стає недоступною для відвідання мурахами, які роблять свій крок пізніше.

Псевдокод алгоритму можна подати так.

**procedure MMAS** (*x*)
ініціалізація_алоритму;
**while** кількість_ітерацій_без_покращення_менша_заданої **do**
формування_популяції_мурах;
**while** не_всі_вершини_відвідані **do**
**for** чергова_мураха_з_популяції **do**
ініціалізація_поточного_кроку_мурахи;
*M* = оновлення_пам'яті_мурахи;
*A* = локальна_матриця_мурашиних_маршрутів;
сформувати_множину_припустимих_вершин_зурахуванням
_наявного_ресурсу;
*p* = обчислити_ймовірність_переходів(*A*, *M*, П);
наступний_стан = правило_прийняття_рішення(*p*, П);
вибравши вершину, перейти_в_наступний_стан;
*M* = оновити_внутрішній_стан;
вилучити вибрану вершину із списку припустимих;

```
        endfor
       endwhile
    завершити_діяльність;
    відкласти_феромон_на_найкращому_маршруті_на_поточній_ітерації;
    випаровування_феромону;
    оновлення_рекорду (x);
    endwhile
end
```

Тут П – предикат, що описує обмеження задачі (більш детальніший опис обчислювальної схеми алгоритмів оптимізації мурашиними колоніями див. в [15]).

**4. Обчислювальний експеримент**

Оскільки у відкритому доступі немає даних для даної задачі, то був згенерований набір задач з використанням системи Google Maps API.  Згенеровані задачі мають розмірність від 10 до 500 об’єктів, які необхідно відвідати, у середньому 4 – 7 місць старту та 2 – 5 зон приймання.

Додаткові позначення:

$n -$ кількість об’єктів, що необхідно відвідати; *MMAS* – модифікований мах-мінний алгоритм мурашиних систем; *TabuSearch* – локальний пошук (табу пошук) з використанням оператора переміщення; *Bruteforce* – алгоритм прямого перебору; $f_{поч}$ – початковий розв’язок; $f_{роз}$ – розв’язок задачі, отриманий алгоритмом; $q = \frac{f_{поч} - f_{роз}}{f_{поч}} \cdot 100\,\%$ – відносне покращення початкового розв’язку (у відсотках); $t -$ час виконання алгоритму (у мілісекундах).

Всі алгоритми реалізовані на одній і тій же програмній базі на мові Kotlin, пошук розв’язків здійснювався на персональному комп’ютері з 16 ГБ оперативної пам’яті та восьмиядерним процесором з тактовою частотою 3.6 ГГц. Це дозволяє порівнювати час роботи алгоритмів.

Жирним шрифтом виділені найкращі розв’язки між всіма алгоритмами (таблиця).

*Параметри*. Для локального пошуку застосовані наступні параметри: $l = 5$ – розмір списку заборон, $iter = 500$ – кількість можливих ітерацій, на яких не відбувається покращення розв’язку. Для макс-мін алгоритму мурашиних систем: $noa = m$ (кількість БПЛА) – кількість мурах, $noi$ = 1000 – кількість можливих ітерацій, на яких не відбувається покращення розв’язку, $\alpha$ = 1 – ступінь значущості феромонного сліду, $\beta$ = 1 – ступінь значущості евристичної інформації, $e$ = 0.1 – стала випаровування.

Параметри підібрані шляхом вибору найкращих з проведеної серії попередніх випробувань.

ТАБЛИЦЯ

| Задача | *n* | MMAS | | | | TabuSearch | | | | Bruteforce | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | $f_{поч}$ | $f_{роз}$ | *q* | *t* | $f_{поч}$ | $f_{роз}$ | *q* | *t* | $f_{роз}$ | *t* |
| 1 | 10 | 2757.1 | 2193.9 | 20.4 | 61 | 2465.5 | 2157.3 | 12.5 | 6 | **1690.1** | 62 |
| 2 | 16 | 3987.4 | 3299.2 | 17.2 | 69 | 4510.5 | 3669.4 | 18.6 | 7 | **3151.2** | 23621 |
| 3 | 30 | 7696.4 | **4369.5** | 43.2 | 158 | 6941.6 | 4478.9 | 35.4 | 30 | – | – |
| 4 | 50 | 7947.9 | **5738.7** | 27.7 | 294 | 7784 | 6443 | 17.2 | 149 | – | – |
| 5 | 100 | 13619.2 | 9693.4 | 28.8 | 978 | 16430 | **9665.8** | 41.1 | 212 | – | – |
| 6 | 500 | 28725.4 | **19698.2** | 31.4 | 44712 | 29906 | 24869.6 | 16.8 | 35840 | – | – |

Як випливає з результатів експерименту, розроблений модифікований макс-мінний алгоритм мурашиних систем продемонстрував підвищені показники точності в порівнянні з алгоритмами локального пошуку та прямого перебору.

**Висновки.** Запропоновано змістовну постановку виділеної задачі маршрутизації для виконання обльоту об'єктів групою БПЛА з умовою завершення маршруту в зонах приймання та обмеженнях на ресурси БПЛА як спеціальної задачі комбінаторної оптимізації.

Показано, що математична модель руху групи БПЛА може бути зведена до моделі маршрутизації транспортних засобів із декількома депо. Таку оптимізаційну задачу можна розв'язувати методом повного перебору лише при малій розмірності задачі. Для розв'язування сформульованої задачі маршрутизації групи БПЛА розроблено макс-мін алгоритм мурашиних систем. Особливістю алгоритму є покрокова взаємодія мурах при формуванні розв'язків.

Напрямком подальших досліджень може бути покращення мурашиного алгоритму за рахунок використання «далекозорих» мурах [16] та паралельної реалізації острівної моделі [17].

*Л.Ф. Гуляницкий, В.В. Сторчевый*

ОПТИМИЗАЦИЯ МАРШРУТОВ ГРУППЫ БПЛА МОДИФИЦИРОВАНЫМ АЛГОРИТМОМ МУРАВЬИНЫХ СИСТЕМ

Рассмотрена проблема маршрутизации группы беспилотных летательных аппаратов (БПЛА). Показано, что модель движения транспортных средств с несколькими депо можно применить для планирования полета группы БПЛА. Разработан макси-минный алгоритм муравьиных колоний для решения одной из задач маршрутизации. Проведен сравнительный анализ алгоритмов для решения задачи маршрутизации группы БПЛА.

*L.F. Hulianytskyi, V.V. Storchevyi*

ROUTES OPTIMIZATION OF UAV GROUPS BY MODIFIED ANT SYSTEM ALGORITHM

The problem of routing a group of unmanned aerial vehicles (UAVs) is considered. It is shown that the model of movement of vehicles with several depots can be used to plan the flight of UAV groups. Max-min ant system algorithm for solving a multi-depot routing problem is developed. A comparative analysis of algorithms for solving the problem of routing a group of UAVs is presented.

**Список літератури**


1. Golden B., Raghavan S., Wasil E. The Vehicle Routing Problem: Latest Advances and New Challenges. *Springer Science+Business Media*. 2008.
2. Гуляницький Л.Ф., Рибальченко О.В. Формалізація та розв'язування одного типу задач маршрутизації БПЛА. *Теорія оптимальних рішень*. 2018. 17.– С. 107 – 114.
3. Adbelhafiz M., Mostafa A., Girard A. Vehicle Routing Problem Instances: Application to Multi-UAV Mission Planning. *AIAA Guidance, Navigation, and Control Conference, Ontario*. 2010.
4. *Chiang W.*, *Li Y.*, *J. Shang*, *Urban T.L.* Impact of drone delivery on sustainability and cost: Realizing the UAV potential through vehicle routing optimization, Applied Energy. 2019. 242. P. 1164 – 1175.
5. Гуляницкий Л.Ф. Проблема оптимизации маршрутов транспортных средств с временными окнами. *Компьютерная математика*. 2007. № 1. С. 122 – 132.
6. Liu D., Li S. Research on efficient online planning of emergency logistics path based on doublelayer ant colony optimization algorithm. *International Journal of Computers and Applications.* 2018. 33. P. 1 – 7.
7. Franco C., Aguilar H., Ochoa-Zezzatti A., Gallegos P. Comparison between instances to solve the CVRP. *International Journal of Combinatorial Optimization Problems and Informatics*. 2018. 9(2). P. 41 – 54.
8. Yu M., Li S., Kong M., Song J., Yang J., Ren G. Comparison of advantages and disadvantages among various algorithms in logistics path design – Taking H-group as an example. *Cognitive Systems Research*. 2018. 52. P. 843 – 852.
9. Pereira F. B., Tavares J. Bio-inspired Algorithms for the Vehicle Routing Problem. *Studies in Computational Intelligence 161, Springer*. 2009. P. 79.
10. Липский В. Комбинаторика для программистов: Пер. с польск. М.: Мир. 1988. 213 с.
11. Kohl N. Exact methods for Time Constrained Routing and Related Scheduling Problems. *PhD thesis, Department of Mathematical Modelling, Technical University of Denmark.* 1995.
12. Soto M., Sevaux M., Rossi A., Reinholz A. Multiple neighborhood search, tabu search and ejection chains for the multi-depot open vehicle routing problem. *Computers & Industrial Engineering*. 2017. Vol. 107. P. 211 – 222.
13. Dorigo M., Stützle T. Ant Colony Optimization: Overview and Recent Advances. *Springer International Publishing AG*. 2019. P. 311 – 352.
14. Stützle T., Hoos H.H. MAX-MIN ant system. *Future Generation Computer Systems*. 2000. P. 889 – 914.
15. Гуляницький Л.Ф., Мулеса О.Ю. Прикладні методи комбінаторної оптимізації: навч. посіб. К.: Видавничо-поліграфічний центр "Київський університет". 2016. 142 с.
16. Гуляницький Л.Ф. Новий алгоритм оптимізації мурашиними колоніями. *Сучасна інформатика: проблеми, досягнення та перспективи розвитку*. Пр. Міжн. конф., присвяченої 60-річчю заснування ІК імені В.М. Глушкова НАН України (Київ, 13 – 15 грудня 2017 р.). К.: ІК ім. В.М. Глушкова НАН України. 2017. С. 41 – 43.
17. Mora A.M., García-Sánchez P., Merelo J.J. Pareto-based multi-colony multi-objective ant colony optimization algorithms: an island model proposal. *Soft Computing.* 2013. Vol. 17. P. 1175.


Одержано 20.04.2019

***Про авторів:***

*Гуляницький Леонід Федорович*,

доктор технічних наук, завідувач відділу
Інституту кібернетики імені В.М. Глушкова НАН України,
E-mail:LeonHul.ICyb@dmail:com

*Сторчевий Владислав Володимирович*,

магістрант Інституту кібернетики імені В.М. Глушкова НАН України.