УДК 519.854
DOI https://doi.org/10.24144/2616-7700.2026.48(1).243-253

С. В. Чупов[1], А. В. Федорішко[2]

[1] ДВНЗ «Ужгородський національний університет»,
доцент кафедри системного аналізу та теорії оптимізації,
кандидат фізико-математичних наук, доцент
serhii.chupov@uzhnu.edu.ua
ORCID: https://orcid.org/0000-0002-7715-3924

[2] ДВНЗ «Ужгородський національний університет»,
аспірант кафедри системного аналізу та теорії оптимізації
anton.fedorishko@uzhnu.edu.ua
ORCID: https://orcid.org/0009-0002-8921-0882

ПОКРАЩЕНИЙ ГІБРИДНИЙ АЛГОРИТМ ДЛЯ КВАДРАТИЧНОЇ ЗАДАЧІ ПРО ПРИЗНАЧЕННЯ

Квадратична задача про призначення (QAP) є однією з найбільш вивчених комбінаторних задач оптимізації з різними практичними застосуваннями. У цій статті ми представляємо покращений гібридний алгоритм (IHBMA) для розв'язання QAP. IHBMA досліджує простір пошуку шляхом використання модифікованого алгоритму 2-OPT, та схеми алгоритму BMA на окремих етапах розв'язання задачі. Експериментальні оцінки на множині тестових задач типу "***Taie***" розмірності 125 та 175 показують, що запропонований підхід здатний досягти та перевершувати найвідоміші на даний момент результати для всіх екземплярів із середнім часом обчислення менше 2 години. Також надаються порівняння, щоб показати конкурентоспроможність запропонованого підходу відносно алгоритму HH-QAP для QAP.

Ключові слова: Квадратична задача про призначення, гібридний алгоритм наближеного пошуку, локальний алгоритм наближеного пошуку, випадкове збурення компонент розв'язку, ефективність алгоритмів.

1. Вступ. Відомо, що багато задач дискретної оптимізації, зокрема задача QAP, є NP-складними [13]. Основною проблемою при їх розв'язанні є експоненціальний ріст обчислювальних витрат при збільшенні розмірності задачі. Це обмежує можливості розв'язання таких задач у реальному масштабі часу. Крім того, обчислювальна складність задачі QAP викликана значним об'ємом обчислень (порядку $O(n^2)$), необхідних для отримання значення цільової функції та дослідження околу поточного розв'язку. Нагадаємо, що в алгоритмах для задачі QAP окіл $N(\pi)$ розв'язку π складається з усіх перестановок, які отримані шляхом обміну позиціями між усіма парами об'єктів. Таким чином, окіл містить $C_n^2 + 1$ перестановку, а обчислення значень цільової функцій для усіх перестановок з $N(\pi)$ потребує об'єму обчислень порядку $O(n^4)$. Якщо $n \geq 100$, то виконання хоча б n^2 ітерацій локального пошуку вже стає важкою обчислювальною задачею.

Практична важливість та обчислювальна складність задачі QAP підтверджується хронологією її дослідження. Сформульована вона була Koopmans та Beckmann [9] у 1957 р. У 1976 р. Sahni та Gonzalez довели [13], що ця задача належить до класу NP-складних задач. Burkard [4] у 1984 р. відзначив, що не можна побудувати точних методів для ефективного розв'язання QAP розмірністю, більшою за 20. І тільки з кінця 80-х років минулого сторіччя розпочався

бурхливий розвиток наближених методів відшукання якісних розв'язків квадратичної задачі про призначення.

Сьогодні більшість ефективних наближених алгоритмів для QAP базується на використанні різноманітних схем методу табу пошуку (TS), запропонованого F. Glover [7, 8]. Варто згадати такі версії методу, як надійний табу пошук [15] (ROTS), реактивний табу пошук [1] (RETS). В останні роки найбільш якісні результати отримано такими наближеними алгоритмами як Breakout Local Search, BLS [2], Memetic Algorithm for QAP, BMA [3] Improved Hybrid Genetic Algorithm, IHGA [11], ITS for QAP [12] та багатьма іншими.

Як правило схеми сучасних наближених методів розв'язання задач оптимізації включають дві фази: інтенсифікація пошуку та диверсифікація пошуку. На першій фазі здійснюється пошук локального екстремуму задачі, а на другій — робиться спроба вийти з його околу та перейти у іншу, ще не досліджену, область множини допустимих розв'язків задачі. Розглянемо більш детально деякі з ефективних алгоритмів розв'язання QAP.

Ітеративний заборонений пошук (ITS) [12] дотримується загальної схеми мета евристичного ітеративного локального пошуку. Він використовує традиційний заборонений пошук для досягнення локальних оптимумів та запускає фазу збурень (реконструкції) щоб уникнути досягнутого локального оптимуму. «Зруйнований» розв'язок стає новою відправною точкою для базової процедури TS. ITS використовує механізм збурень, який адаптивно змінює кількість випадкових рухів збурень в деякому інтервалі. ITS отримує чудові результати на неструктурованих екземплярах та реальних екземплярах.

У [18, 19] представлено ефективний алгоритм для розв'язання QAP, названий RITSR (повторюваний ітеративний алгоритм табу). Фаза інтенсифікації, яка названа дослідницьким пошуком реалізує схему ітеративного локального табу пошуку (ITSL). Цей пошук призначений для виявлення кращих розв'язків, що містяться на незначній відстані від межі поточного околу локального мінімуму. Якщо отримано кращий розв'язок, він запам'ятовується та починається детальний пошук нового, кращого за знайдений, розв'язку в околі знайденої перестановки. Після цього цикл повторень у режимі дослідницького пошуку розпочинається заново. Друга фаза пошуку — фаза диверсифікації — виконується кожен раз перед початком нового дослідницького пошуку у циклі повторень. У цій фазі збурюється поточний локальний мінімум так, щоб наступний дослідницький пошук відбувався на певній відстані від околу поточного розв'язку.

У [20] розроблено гібридно-евристичну модель з використанням структури високого рівня для алгоритмів розв'язання задачі QAP, яка поєднує ключові компоненти трьох відомих метаевристик: "Scatter Search", "Critical Event Tabu Search", and "Genetic Algorithm Random Key", які застосовуються до задачі квадратичного призначення (алгоритм PP-QAP).

У даній статті представлено покращений гібридний алгоритм розв'язання QAP названий IHBMA, що використовує ідею алгоритму HBMA [22]. У своїй роботі він використовує комбінацію модифікованого алгоритму 2-OPT, та схему алгоритму BMA. Наведена формальна схема алгоритму та результати числових експериментів. У висновках коротко оцінюється ефективність запропонованого алгоритму. Пропонуються подальші напрямки досліджень.

2. Постановка задачі. Дано n об'єктів, які потрібно розташувати у n

різних локаціях (місць призначень). Відомі величини потоків ресурсів a_{ij} між об'єктами i та j, $i, j = 1, \ldots, n$, і відстані b_{rs} між локаціями (пунктами) r та $s, r, s = 1, \ldots, n$. Потрібно знайти таке розподілення об'єктів по локаціях, щоб сума відстаней, помножена на відповідні потоки, була б мінімальною.

Математична постановка задачі QAP формулюється наступним чином: знайти:

$$\min_{\pi \in \Pi^n} f(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} b_{\pi_i \pi_j},$$

де $A = [a_{ij}]$ та $B = [b_{rs}]$ — квадратні матриці порядку n, Π^n — множина усіх перестановок розмірності n і π_i задає номер локації об'єкту i. При цьому, якщо матриці A та B є симетричними, тоді маємо симетричну задачу QAP для якої значення та прирости цільової функції обчислюються більш ефективно.

Можна подати математичну модель задачі QAP у бінарній формі. Нехай P — квадратна бінарна матриця, така що для будь якої перестановки π, $p_{i,\pi_i} = 1$, $i = 1, \ldots, n$. Побудуємо матрицю $\overline{B} = B * P$. Тоді постановка задачі буде такою:

$$\min_{\pi \in \Pi^n} f(\pi) = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} \bar{b}_{ij},$$

Квадратичні задачі про призначення можливо точно розв'язати тільки для не великих размірностей. У зв'язку з цим актуальними є розробка, дослідження нових і вдосконалення існуючих наближених алгоритмів розв'язання задачі QAP.

3. Оператор універсального кросоверу з випадковим доповненням. За принципом роботи це стандартний оператор рівномірного кросовера [21]. Деякі локації вибираються випадковим чином з першого розв'язку за розподілом Бернуллі з параметром 1/2. Місця, зайняті вибраними об'єктами з першої перестановки, копіюються у результуючий розв'язок. Місця, зайняті рештою об'єктів у другій перестановці, також копіюються, якщо вони ще не були включені з першого розв'язку. Результат доповнюється шляхом випадкового призначення відсутніх місць. Алгоритм універсального кросоверу з випадковим доповненням представлено на лістингу 2.

Лістинг 2: Універсальний кросовер з випадковим доповненням.

```
function UniCrossiver(left, right: Vector):Vector;
begin
order := Vector(1, {\dots}, n);
offspring := Vector{−1, . . . , −1};
                    n
assigned_facilities := List()
for i:=1 to n do
begin
  if bernoulli(0.5) then
    offspring[i] := left[i];
    assigned_facilities.Add(left[i]);
  end;
end;
shuffle(order);
for i:=1 to n do
```

```
begin
  if offspring[order[i]] = -1 then
      for j:=1 to right.Count do
      begin
          if not right}[j] in assigned_facilities then
          begin
              offspring[j] := right[j];
              assigned_facilities.Add(right[j]);
              break;
          end;
      end;
  end;
return offspring;
end;
```

У описанні алгоритму використовуються такі позначення та команди: *order* — визначає порядок перегляду елементів перестановки; команда *shuffle(order)* — випадковим чином перемішує порядок перегляду *order*; список *assigned_facilities* — містить перелік призначених локацій; команда *bernoulli*(0.5) — повертає випадкове значення за розподілом Бернуллі; перестановка *offspring* — результат роботи кросовера.

У даній роботі використовується саме універсальний кросовер з випадковим доповненням. На відміну від стандартного універсального кросовера місця, що залишились після призначення з перестановки *left*, заповнюються у порядку *order* з перестановки *right*.

4. Алгоритм 2-OPT. Серед простих алгоритмів локального пошуку найвідомішим є 2-OPT [17]. Алгоритм 2-OPT вперше був запропонований Кроесом у 1958 році для задачі комівояжера. Цей алгоритм використовує попарну транспозицію об'єктів. Якщо кількість місць розташування об'єктів позначена як n, кількість транспозицій у кожній ітерації буде $n(n-1)/2$. Спочатку алгоритм розглядає транспозицію об'єктів 1 та 2. Якщо значення цільової функції отриманого розв'язку менше, ніж значення цільової функції початкового розв'язку, то воно зберігається як кандидат для подальшого розгляду. В іншому випадку воно відкидається, і алгоритм розглядає транспозицію засобів 1 та 3. Якщо цей обмін генерує краще рішення, то воно зберігається як кандидат для подальшого розгляду; в іншому випадку, воно відкидається і так далі. Таким чином, щоразу, коли знайдено краще рішення, алгоритм відкидає попереднє найкраще рішення. Ця процедура продовжується, доки не будуть розглянуті всі попарні обміни. У роботі використовується модифікований алгоритм схема роботи якого представлена на лістингу 3 та 4.

Лістинг 3: Схема алгоритму локального пошуку.

```
function LocalSearch(p: Vector; f, maxi: Integer):(Vector, Integer);
begin
pc := p; pbest := p;
fc := f; fbest := f; /* current and best values */
D := ComputeGains(pc); /* compute vector of gains */
i := 0;
while true do
begin
  (pc, fc, D) := LocalSearchOne (pc, D, fc) /* See Alg. 3 */
  i := i + 1;
```

```
11    if fc < fbest then
12    begin
13        pbest := pc; fbest := fc;
14        i := 0;
15    end;
16    if i > maxi then break;
17  end;
18  return (pbest, fbest);
19  end;
```

Лістинг 4: Схема одного кроку алгоритму 2-OPT.

```
1   function LocalSearchOne(p, D: Vector; f: Integer):(Vector, Integer,
        Vector);
2   begin
3   fc := f; pc := p;
4   nmoves := n(n-1)/2;
5   ming := MAXINT64; minm := -1;
6   for i:=0 to nmoves do
7   begin
8     if D[j] < ming then
9     begin
10        ming := D[j]; minm := j;
11    end;
12  end;
13  if ming < 0 then
14  begin
15    fc := fc + ming;
16    swap(pc, minm);
17    D := UpdateGains(D, pc, minm);
18  end
19  else begin
20    (pc, fc) := MutatePermutation(pc, strength);
21  end;
22  return (pc, D, fc);
23  end;
```

У алгоритмі 3 використовується функція *ComputeGains* для побудови вектору приростів цільової функції для усіх можливих ходів. Під одним ходом розуміється перестановка двох позицій у заданій перестановці. У алгоритмі 4 це робиться за допомогою команди *swap*. Також у алгоритмі 3 функція *UpdateGains* використовується для перебудови вектору приростів після одного кроку. Функція *MutatePermutation* здійснює перехід від даної перестановки до іншої, де параметр *strength* визначає ступінь збурення.

5. Покращений гібридний алгоритм (IHBMA). Даний алгоритм на основі двох існуючих алгоритмів 2-OPT та Breakout Memetic Algorithm (BMA) [3] для розв'язання задачі QAP, використовує модифікований алгоритм 2-OPT та загальну схему алгоритму BMA.

Термін «меметичний алгоритм» (MA) використовується для позначення загального евристичного підходу, який зазвичай поєднує локальну оптимізацію з популяційною парадигмою. Мета такої комбінації полягає в тому, щоб скористатися перевагами як перехресного аналізу, який виявляє недосліджені, перспективні області пошуку, так і локальної оптимізації, яка знаходить гарні розв'язки, концентруючи пошук навколо цих областей. Зрештою, як і для будь-

якого методу, що базується на популяції, необхідно підтримувати здорову різноманітність популяції, щоб уникнути передчасної конвергенції.

Початкова популяція будується на основі модифікованого алгоритму 2-OPT для виявлення перспективних областей пошуку. Враховуючи таку популяцію, яка складається з локально оптимальних розв'язків, меметичний підхід генерує нові рішення, застосовуючи кросовер або мутацію, а потім фазу локального пошуку для покращення кожного розв'язку з поточної популяції. Правильний вибір оператора кросовера залежить від структури конкретної задачі. Більше того, успіх меметичного підходу обумовлений ефективністю процедури локального пошуку. Головна роль кросовера полягає у виявленні недосліджених перспективних областей простору пошуку, локальний пошук в основному спрямований на пошук хороших рішень шляхом концентрації пошуку навколо цих областей.

Покращений гібридний алгоритм для QAP (IHBMA) використовує оператор універсального кросоверу з випадковим доповненням, модифіковану процедуру локального пошуку 2-OPT, стратегію заміщення популяції на основі придатності та адаптивний механізм мутації. Кожний розв'язок для популяції, згенерований за допомогою універсального кросовера, покращується за допомогою модифікованої процедури локального пошуку 2-OPT. Даний алгоритм потім застосовує стратегію оновлення популяції, щоб можливо замінити найгірший об'єкт з популяції покращеною перестановкою. Щоб уникнути передчасної конвергенції, IHBMA запускає адаптивний механізм мутації для всієї популяції, якщо найкраще рішення, знайдене під час пошуку, не було покращено протягом фіксованої кількості ітерацій. Це зміщує пошук у віддалені регіони щоразу, коли виявляється застій пошуку. На ліст. 5 Представлена формальна схема алгоритму.

Лістинг 5: Формальна схема алгоритму IHBMA.

```
1  procedure IHBMA;
2  begin
3  P := GenerateRandomPopulation(psize);
4  P := BuildPopulation (P, maxi);
5  while not stopping condition do
6  begin
7    (left, right) := SelectParents(P);
8    pc := UniCrossover(left, right);
9    (pc, fc) := LocalSearch(pc, f(pc), maxi);
10   if fc < fbest then
11   begin
12        pbest := pc; fbest := fc;
13   end;
14   if (pbest not improved after ν iterations) then
15   begin
16        strength:=GenerateStrength(P);
17        P:=MutatePopulation(P, strength);
18        P:=BuildPopulation(P, maxi);
19   end
20   else
21   begin
22        P := UpdatePopulation(P, pc);
23   end;
```

```
24 end;
25 end;
```

Процедура *GenerateRandomPopulation* випадковим чином генерує множину перестановок, яка у термінах генетичних алгоритмів називається популяцією. Процедура *BuildPopulation* представлена нижче алгоритмом 6. Процедура *UniCrossover* використовує універсальний кросовер з випадковим доповненням (алгоритм 2) для побудови нової перестановки або нащадка. Процедура *LocalSearch* здійснює локальний пошук в області нащадка за алгоритмом 3. Команда *GenerateStrength* визначає ступінь збурення перестановок популяції, що здійснюється процедурою *MutatePopulation*. Після отримання чергової перестановки алгоритмом локального пошуку здійснюється оновлення популяції процедурою *UpdatePopulation*.

Алгоритм на лістингу 6 намагається покращити кожен член популяції за алгоритмом локального пошуку 3.

Лістинг 6: Алгоритм побудови або перебудови популяції.

```
1  procedure BuidPopulation(P: Population; maxi: Integer);
2  begin
3  for i:=1 to psize do
4  begin
5    (P[i], fi) := LocalSearch(P[i], f(P[i]), maxi);
6    if fp < fbest then
7    begin
8        pbest := P[i]; fbest := fi;
9    end;
10 end;
11 end;
```

6. Результати числових експериментів. Найбільш використовуваними штучними тестовими задачами, є дві серії тестів Taillard (*Taia* та *Taib*) [5]. В однорідних екземплярах (*Taia uniform distributed problems*) матриця відстаней є евклідовою матрицею відстаней між випадковими точками на колі, а матриця потоку є випадковою матрицею з цілим числом, випадково вибраним між двома межами. Реально-подібні екземпляри (*Taib real-like problems*) створені на основі певних реальних проблем та імітують деякі їхні властивості. Матриця відстаней також є евклідовою матрицею, але де точки кластеризовані (розбиваються на групи), а значення матриці потоку розподілені експоненціально. Зараз використовуються інші тестові екземпляри. Серія екземплярів *Taie* та *Dre* була спеціально розроблена для складних метаевристик [6]. Крім того, створено екземпляри, які систематично змінюють деякі параметри тестової задачі, що пов'язані з домінуванням та розрідженістю значень потоків [14]. Також було запропоновано окремий випадок QAP, який є поліноміально розв'язуваним [10], для тестування алгоритмів "чорної скриньки".

Числові експерименти проводилися на тестових задачах типу TaiXXXeYY, де XXX — визначає розмірність задачі, а YY — номер тестової задачі. У наступних таблицях використані такі позначення: *Instances* — назва задачі; *BKS* — відомий рекорд; f^* — знайдене рекордне значення; $\Delta_{\%} = 100 * (f^* - BKS)/BKS$ — середнє відхилення цільової функції від відомого рекорду у процентах. Результати розв'язання таких задач для алгоритму HH-QAP було взято з [20].

У таблицях 1, 2 наведені результати розв'язання тестових задач типу ***tai-XXXeYY***, алгоритмами HH-QAP та IHBMA.

Таблиця 1.

Задачі "***Tai125eyy***". Час розв'язання *3600* сек.

Instances	**BKS**	**HH-QAP**		**IHBMA**	
		f^*	$\Delta_{\%}$	f^*	$\Delta_{\%}$
Tai125e01	35426	35426	0	35426	0
Tai125e02	36202	36178	-0.07	36178	-0.07
Tai125e03	30498	30498	0	30498	0
Tai125e04	33084	33084	0	33084	0
Tai125e05	38432	37210	-3.18	37210	-3.18
Tai125e06	35546	34624	-2.59	34624	-2.59
Tai125e07	32712	31466	-3.81	**31020**	**-5.17**
Tai125e08	36354	34424	-5.31	34424	-5.31
Tai125e09	35008	34244	-2.18	34244	-2.18
Tai125e10	34898	34898	0	34898	0
Tai125e11	33082	32186	-2.71	**32132**	**-2.87**
Tai125e12	32326	32326	0	32326	0
Tai125e13	35380	34364	-2.87	**34280**	**-3.11**
Tai125e14	30460	30460	0	30460	0
Tai125e15	34328	32614	-4.99	32614	-4.99
Tai125e16	32674	31058	-4.95	31058	-4.95
Tai125e17	35512	35274	-0.67	**35074**	**-1.23**
Tai125e18	38702	36888	-4.69	**36712**	**-5.14**
Tai125e19	33034	32966	-0.21	32966	-0.21
Tai125e20	31988	30896	-3.41	30896	-3.41

Таблиця 2.

Задачі "***Tai175eyy***". Час розв'язання *7200* сек.

Instances	**BKS**	**HH-QAP**		**IHBMA**	
		f^*	$\Delta_{\%}$	f^*	$\Delta_{\%}$
Tai175e01	57540	57540	0	57540	0
Tai175e02	51036	50110	-1.81	**50002**	**-2.03**
Tai175e03	53900	53900	0	53900	0
Tai175e04	63182	60416	-4.38	**60352**	**-4.48**
Tai175e05	51278	50004	-2.48	50004	-2.48
Tai175e06	54752	54752	0	54752	0
Tai175e07	52502	52178	-0.62	**52140**	**-0.69**
Tai175e08	57304	55024	-3.98	**54984**	**-4.05**
Tai175e09	53238	50020	-6.04	**49664**	**-6.71**
Tai175e10	52010	51022	-1.9	**50998**	**-1.95**
Tai175e11	54892	54384	-0.93	54384	-0.93
Tai175e12	59564	57492	-3.48	**57400**	**-3.63**
Tai175e13	59840	58952	-1.48	**58828**	**-1.69**
Tai175e14	55520	52430	-5.57	52430	-5.57
Tai175e15	49668	47092	-5.19	**46612**	**-6.15**
Tai175e16	55968	55764	-0.36	55764	-0.36
Tai175e17	58572	57690	-1.51	**57688**	**-1.51**
Tai175e18	51574	47972	-6.98	47972	-6.98
Tai175e19	52298	49974	-4.44	**49878**	**-4.63**
Tai175e20	56616	55342	-1.95	55342	-1.95

Слід зазначити, що для усіх тестових задач рекордні значення цільової функції, отримані за алгоритмом HH-QAP, було досягнуто або перевершено. Зокрема, для задач розмірності 125 у п'яти задачах з 20 було знайдено кращі розв'язки, а у задачах розмірності 175 у одинадцяти задачах з 20 було отримано покращення.

7. Висновки та перспективи подальших досліджень. У цій статті пропонується ефективна реалізація гібридного алгоритму IHBMA для розв'язання задачі QAP. Метою цього алгоритму є покращення продуктивності пошуку шляхом спроби мінімізувати негативний вплив явища стагнації, що є однією з головних перешкод ітераційного пошуку, особливо у випадках, коли необхідно виконати значну кількість кроків на кожному етапі пошуку. Кожен з етапів пошуку алгоритму IHBMA базується на парадигмі інтенсифікації та диверсифікації, де інтенсифікація спрямована на пошук кращих розв'язків поблизу поточного, тоді як диверсифікація відповідає за вихід з поточного локального оптимуму та переміщення до нових невивчених областей простору пошуку. На кожному етапі пошуку використовується різні алгоритми. На етапі дослідницького пошуку для виявлення перспективних областей використовується модифікований алгоритм 2-OPT на етапі детального пошуку застосовується алгоритм, побудований за схемою BMA, але усі процедури у ньому модифіковані. Наприклад, для визначення наступної перестановки для пошуку за локальним алгоритмом використовується оператор універсального кросоверу з випадковим доповненням. Кожен окремий алгоритм задає свою траєкторію пошуку, тобто процесу переходу від одного розв'язку до іншого. Тому, якщо використовується тільки один конкретний алгоритм при розв'язанні задачі, то дуже часто такий пошук призводить до переходу у режим стагнації (застою), особливо для "важких" задач, у яких при наймі розмірність є більшою за 100. Комбінація застосування алгоритмів дозволяє більш ефективно здійснювати процедуру диверсифікації, що значно підвищує ефективність всього алгоритму. Про це яскраво свідчать результати числових експериментів. Алгоритм IHBMA показує кращі результати як за кількістю розв'язаних задач та і за середнім часом розв'язання у порівнянні з використанням алгоритму HH-QAP.

У подальших дослідженнях планується:

– Застосувати алгоритм IHBMA до задачі про максимальний розріз графу.
– Застосувати алгоритм IHBMA до багатовимірної булевої задачі про ранець.
– Використати комбінації інших алгоритмів, таких як RITSR, ITS, ReTS, IHGA разом із 2-OPT та BMA.
– Дослідити вплив різних операторів кросоверу з метою визначення більш придатних.
– Розробити механізм динамічної адаптації параметрів алгоритму, наприклад, таких як тип кросоверу, початкова ступінь мутації популяції, умова переходу до знаходження інших областей пошуку в залежності від ходу процесу розв'язання.

Конфлікт інтересів

Автори заявляють, що не мають конфлікту інтересів щодо даного дослідження, включаючи фінансовий, особистий, авторський або будь-який інший, який міг би вплинути на дослідження, а також на результати, представлені в даній статті.

Фінансування

Дослідження здійснено в рамках кафедральної науково-дослідної роботи «Моделі і методи системного аналізу в міждисциплінарних дослідженнях» (державний обліковий номер 0125U003246).

Доступність даних

Усі дані доступні в цифровій або графічній формі в основному тексті рукопису.

Використання штучного інтелекту

Автори підтверджують, що при створенні даної роботи вони не використовували технології штучного інтелекту.

Внесок авторів

С. В. Чупов: концептуалізація, формальний аналіз, розробка алгоритмів, програмування, написання — оригінальний проект. А. В. Федорішко: програмування, написання, проведення числових експериментів — оригінальний проект.

Список використаної літератури

1. Battiti, R., & Tecchiolli, G. (1994). The Reactive Tabu Search. *ORSA J. on Computing,* 6, 126–140.
2. Benlic, U., & Hao, J. K. (2013). Breakout local search for the quadratic assignment problem. *Applied Mathematics and Computation,* 219(9), 4800–4815.
3. Benlic, U., & Hao, J. K. (2015). Memetic search for the quadratic assignment problem. *Expert Systems with Applications,* 42(1), 584–595.
4. Burkard, R. E. (1984). Quadratic assignment problems. *European J. of Oper. Res.* 15, 283–289.
5. Burkard, R. E., Karisch, S. E., & Rendl, F. (1997). Qaplib – a quadratic assignment problem library. *Journal of Global optimization,* 10, 391–403.
6. Drezner, Z., Hahn, P. M., & Taillard, E. D. (2005). Recent advances for the quadratic assignment problem with special emphasis on instances that are difficult for metaheuristic methods. *Annals of Operations research,* 139, 65–94.
7. Glover, F. (1989). Tabu Search – Part I, *ORSA J. on Computing,* 1(3), 190–206.
8. Glover, F. (1990). Tabu Search – Part II, *ORSA J. on Computing,* 2(1), 4–32.
9. Koopmans, T. C., & Beckmann, M. J. (1957). Assignment problems and the location of economic activities. *Econometrica,* 25, 53–76.

10. Laurent, M., & Seminaroti, M. (2015). The quadratic assignment problem is easy for robinsonian matrices with toeplitz structure. *Operations Research Letters,* 43(1), 103–109.
11. Misevicius, A. (2004). An improved hybrid genetic algorithm: New results for the quadratic assignment problem. *Knowledge Based Systems,* 17(2–4), 65–73.
12. Misevicius, A. (2012). An implementation of the iterated tabu search algorithm for the quadratic assignment problem. *OR Spectrum,* 34(3), 665–690.
13. Sahni, S., & Gonzalez, T. (1976). P-complete approximation problems. *J. of the ACM,* 23, 555–565.
14. Stutzle, T., & Fernandes, S. (2004). New benchmark instances for the qap and the experimental analysis of algorithms. In *European Conference on Evolutionary Computation in Combinatorial Optimization.* Springer, 199–209.
15. Taillard, E. (1991). Robust taboo search for the quadratic assignment problem. *Parallel Computing,* 17, 443–455.
16. Taillard, E. D. (1995). Comparison of iterative searches for the quadratic assignment problem. *Location science,* 3(2), 87–105.
17. Zhang, Q., Sun, J., Tsang, E., & Ford, J. (2006). Estimation of Distribution Algorithm with 2-opt Local Search for the Quadratic Assignment Problem. In: Lozano, J. A., Larrañaga, P., Inza, I., Bengoetxea, E. (eds). *Towards a New Evolutionary Computation.* Studies in Fuzziness and Soft Computing, (Vol. 192). Springer: Berlin, Heidelberg. https://doi.org/10.1007/3-540-32494-1_12
18. Sergienko, I. V., Shylo, V. P., Chupov, S. V., & Shylo, P. V. (2020). On solving the quadratic assignment problem. *Cybernetics and Systems Analysis,* (1), 64–69 [in Ukrainian].
19. Shylo, V. P., Chupov, S. V., Boyarchuk, D. O., & Shylo, P. V. (June 3–5, 2018) New approaches to solving the quadratic assignment problem. In the book: *VII International Scientific and Practical Conference "Mathematics. Information Technologies. Education."* : Abstracts of reports. Lutsk – Svityaz. Lutsk: Lesya Ukrainka Eastern European National University, 121–122 [in Ukrainian].
20. Wang, H., & Alidaee, B. (2023). A New Hybrid-heuristic for Large-scale Combinatorial Optimization: A Case of Quadratic Assignment Problem, *Computers & Industrial Engineering,* 179(1), 109220.
21. Misevičius, A., & Kilda, B. (2005). Comparison of crossover operators for the quadratic assignment problem, *Information Technology And Control,* 34(2), 109–119.
22. Chupov, S. V., & Fedorishko, A. V. (2025). Hybrid algorithm for solving the quadratic assignment problem. *Scientific Bulletin of Uzhgorod University. Series "Mathematics and Computer Science".* 47(2), 303–311. https://doi.org/10.24144/2616-7700.2025.47(2).303-311

Chupov S. V., Fedorishko A. V. Improved hybrid algorithm for the quadratic assignment problem.

The quadratic assignment problem (QAP) is one of the most studied combinatorial optimization problems with various practical applications. In this paper, we present an improved hybrid algorithm (IHBMA) for solving QAP. IHBMA explores the search space by using a modified 2-OPT algorithm, and the BMA algorithm scheme at separate stages of solving the problem. Experimental evaluations on a set of "***Taie***" type test problems of dimensions 125 and 175 show that the proposed approach is able to achieve and surpass the best known results for all instances with an average computing time of less than 2 hours. Comparisons are also provided to show the competitiveness of the proposed approach with respect to the HH-QAP algorithm for QAP.

Keywords: Quadratic assignment problem, hybrid approximate search algorithm, local approximate search algorithm, random perturbation of solution components, algorithm efficiency.

Отримано: 25.11.2025 Прийнято: 20.12.2025 Опубліковано: 29.01.2026