Теорія оптимальних рішень

УДК 519.8

Л.Ф. ГУЛЯНИЦЬКИЙ

ДИВЕРСИФІКАЦІЯ ПОШУКУ В АЛГОРИТМАХ ОПТИМІЗАЦІЇ МУРАШИНИМИ КОЛОНІЯМИ

Запропоновано підхід до розробки нових алгоритмів оптимізації мурашиними колоніями. Він сприяє створенню алгоритмів розв'язування задач комбінаторної оптимізації з підвищеною точністю. Наведено результати проведеного обчислювального експерименту із розв'язування серії задач комівояжера з відомої бібліотеки, які підтвердили можливість успішної модифікації одного із найефективніших мурашиних алгоритмів.

Вступ. Ідеї багатьох прикладних алгоритмів оптимізації навіяні природою; значну частину з них відносять до ройового інтелекту (swarm intelligence) [1]. Серед відомих алгоритмів ройового інтелекту в комбінаторній оптимізації найбільший інтерес становлять алгоритми оптимізації мурашиною колонією (ОМК; ant colony optimization – ACO), запропоновані Марком Доріго (1992) [2, 3]. Мурашині алгоритми – це багатоагентні системи, де поведінка кожного агента, який називатимемо штучною мурахою, або надалі просто мурахою, заснована на моделюванні поведінки справжніх мурашок. Вони застосовуються для розв’язання багатьох типів задач комбінаторної оптимізації (ЗКО), починаючи з класичної задачі комівояжера (ЗК). Деякі дослідники під ОМК спочатку розуміли певний підклас мурашиних алгоритмів, однак останнім часом зазначений термін у більшості публікацій використовується стосовно всіх алгоритмів цього типу – так будемо вживати його і надалі.

Алгоритми ОМК застосовуються до задач оптимізації, які можуть бути охарактеризовані загалом таким чином: їх розв’язок складається з компонентів (складових), з яких можна покроково будувати фрагменти розв’язків, а на завершальному етапі роботи алгоритму – і повний розв’язок. Наприклад, у ЗК, що стала однією з перших задач, до якої був застосований алгоритм ОМК, такими фрагментами є дуги, що з’єднують міста з яких і формується маршрут – гамільтонів контур.

В роботі пропонується диверсифікація пошуку в ОМК шляхом розгляду варіантів продовження побудови розв'язку, що враховують не одну, а декілька вершин, які можуть бути включені в цей маршрут [4]. Для аналізу ефективності пропонованого підходу здійснено обчислювальний експеримент із розв'язування відомих ЗК у випадку, коли ця кількість вершин дорівнює двом.

1. Загальна схема алгоритмів ОМК.

В алгоритмах ОМК формується спеціальна модель задачі, тому вони належать до класу моделе-орієнтованих методів. Така модель задачі подається у вигляді повного зваженого графа $G(V,E)$, де $v_i \in V, i=1,...,n,$ – вершини, що відповідають компонентам розв'язку, а $e_{ij} \in E, e_{ij}=(v_i,v_j), v_i,v_j \in V,$ – ребра, які відповідають можливим з'єднанням (переходам) між відповідними вершинами. Для кожного ребра може бути визначена функція вартості з'єднання, можливо, залежна від деякого параметра часу *t*.

Умови ЗКО, що розв'язується, можуть визначати набір обмежень $\Pi = \Pi(V,E,t)$ для елементів *V* та *E*, які визначають припустимість зв'язків між компонентами та з'єднаннями, а в підсумку – і побудованого з них розв'язку.

Розв'язки задачі оптимізації можуть бути подані як припустимі шляхи на графі *G*. Алгоритми ОМК можуть використовуватися для знаходження припустимих шляхів мінімальної вартості, що задовольняють обмеженням задачі. Вартість *f*(*x*), що відповідає розв'язку *x*, є функцією всіх вартостей з'єднань, які належать цьому розв'язку.

Мурахою в таких алгоритмах вважають параметричний рандомізований жадібний алгоритм, який покроково будує з множини компонент (вершин чи ребер графа задачі) припустимий розв'язок ЗКО. У своїй роботі він використовує евристичну інформацію та феромонний слід, які характеризують ребра (рідше – вершини) графа задачі.

Евристична інформація (зазвичай позначається η_{ij}) – це числове значення, що не залежить від знайдених на попередніх кроках розв'язків і відображає ступінь бажаності включення в побудований фрагмент розв'язку того чи іншого нового ребра графа моделі $e_{ij} \in E$. Евристичні значення η_{ij} базуються на апріорній інформації, що відображає умови конкретної задачі та надається джерелом, відмінним від мурах; вони можуть залежати від часу (ітерації) *t*.

Рівень феромону (феромонний слід) – τ_{ij}, що відповідає ребру $e_{ij} \in E$, – це додатне число, яке показує, наскільки часто мурахами використовувалося це ребро на попередніх кроках чи при формуванні повного розв'язку. Феромонні сліди є для мурах довготривалою пам'яттю щодо всього процесу пошуку. Залежно від вибраного способу подання задачі, феромонні сліди можуть відповідати всім дугам задачі або тільки деяким з них.

У мурашиних алгоритмах популяція агентів (або мурашок) спільно розв'язує сформульовану задачу оптимізації, використовуючи вищезазначене подання на графі моделі задачі.

Отже, основні компоненти обчислювальної схеми мурашиних алгоритмів такі: модель задачі, що подається спеціальним графом; феромонні значення; евристична інформація; пам'ять (локальна та глобальна).

Як зазначалося, в алгоритмах ОМК кожна штучна мураха – це послідовний жадібний алгоритм, що при побудові шляху в графі задачі, який описується упорядкованою послідовністю вершин, використовує імовірнісне правило для вибору чергового компонента розв'язку.

Нехай m – кількість мурах в одному поколінні; кожна мураха k, $k = 1,...,m$, має пам'ять M^k, яку використовує для зберігання інформації про пройдений шлях. Мурахи починають із початкового фрагмента (вершини графа) і рухаються в припустимі сусідні вершини, поступово формуючи розв'язок. Процедура пошуку завершується, якщо для мурахи k виконано принаймні одну з її завершальних умов.

Стани задачі визначаються в термінах скінченних послідовностей $y = (v_{s_1}, v_{s_2}, ...), v_{s_r} \in V$, елементів V (або, що рівносильно, E), які на всіх проміжних кроках мурахи є фрагментами розв'язку задачі оптимізації. Якщо Y – множина всіх можливих послідовностей, то множина $\overset{\circ}{Y}$ усіх (під)послідовностей, які задовольняють обмеженням задачі $\Pi = \Pi(V, E, t)$, є підмножиною Y: $\overset{\circ}{Y} \subseteq Y$, а її елементи визначають припустимі стани задачі.

Нехай на певному кроці мураха k побудувала фрагмент розв'язку y, останньою компонентою якого є вершина $i \in V$, тобто вона перебуває в цій вершині: $y = (\ldots, i)$. Тоді мураха може переміститися до будь-якої вершини j із множини припустимих сусідніх вершин N_i^k, визначуваних як $N_i^k = \{ j : j \in N_i \wedge (y, j) \in \overset{\circ}{Y} \}$, де N_i – множина всіх сусідніх до i вершин графа задачі. Перехід здійснюється на основі ймовірнісного правила рішення.

Імовірнісне правило рішення для мурах – це функція:

1) значень, що зберігаються в локальній для вершини структурі даних – матриці мурашиних маршрутів $A_i = (a_{ij})$, значення елемента якої отримують функціональною композицією локально доступних для вершини значень феромонних слідів і евристичних значень;

2) особистої пам'яті мурахи, що зберігає передісторію її дій;

3) обмежень задачі.

При переміщенні з вершини i в сусідню вершину j мураха може змінити феромонний слід τ_{ij} на дузі (i, j). Це називається онлайновим покроковим оновленням феромону. Завершивши формування розв'язку, мураха може пройти той самий шлях назад і поновити феромонні сліди на пройдених дугах. Це називається онлайновим відстроченим оновленням феромону.

Після того, як мураха, знайшовши розв'язок, пройшла шлях назад до початкової вершини, вона завершує діяльність, звільняючи всі зайняті ресурси.

Крім діяльності мурашок, ці алгоритми можуть включати ще дві процедури.

1. Випаровування феромону – це процес, за допомогою якого інтенсивність феромонного сліду на ребрах графа задачі автоматично зменшується з часом; використовується коефіцієнт $\rho, \rho \in (0,1)$. Випаровування феромону необхідне для того, щоб уникнути дуже швидкої збіжності алгоритму до субоптимальної області. Воно здійснює корисну форму забування, сприяючи дослідженню нових областей у просторі пошуку.

2. Дії Демона, що можуть використовуватися для здійснення централізованих дій, які не можуть бути виконані окремими мурахами. Прикладом такої діяльності є активація й виконання процедури локальної оптимізації, у якій як початкове наближення застосовується один чи кілька знайдених мурахами розв'язків, або ж збирання глобальної інформації, що може бути потрібна для прийняття рішення: буде або не буде корисним відкладання додаткового феромону для відхилення процесу пошуку від зони знайденого локального розв'язку. Важливо наголосити, що включення процедури дій Демона в алгоритм необов'язкове.

Наприклад, Демон може проглядати шляхи, знайдені кожною мурахою в колонії, і відкладати додатковий феромон на дугах, використаних однією чи кількома мурахами, що знайшли найкоротші шляхи.

Оновлення феромону, виконані Демоном, називаються офлайновими оновленнями феромону.

Загальну обчислювальну схему алгоритмів ОМК можна подати так (рис. 1) [5]. На етапі ініціалізації здійснюються початкові налаштування: вибір значень параметрів алгоритму (зокрема, кількості мурах m у колонії), задання значень феромонного сліду. Після утворення нового покоління мурах їх розміщують довільно у вершинах V графа G. Зазначені вершини є початковими фрагментами маршрутів.

Як зазначалося вище, на кожному кроці конкретна мураха досліджує найближчих сусідів тієї вершини графа задачі, якою закінчується побудований нею на даний момент фрагмент маршруту. Отже, для кожної мурахи k із популяції здійснюється:

a) формування підмножини припустимих вершин $N_i^k \subseteq N_i$ із множини вершин, сусідніх для тієї вершини $i \in V$, яка є останньою в поточному фрагменті маршруту;

b) обчислення ймовірності p_{ij}^k переходу від вершини i до довільної припустимої вершини $j \in N_i^k$ як функції від значень феромону τ_{ij} на ребрі $(i, j) \in E$ та евристичної інформації η_{ij};

c) імовірнісний вибір чергової припустимої вершини $s \in N_i^k$, включення її в кінець фрагмента маршруту, що будується;

d) модифікація значень τ_{ij}, якщо вибрано онлайнове оновлення.

```
procedure ACO (x)
 ініціалізація_алоритму;
 while критерій_завершення_не_задоволений do
 формування_популяції_мурах;    {поточне покоління}
 foreach мураха_з_популяції do  {життєвий цикл мурахи}
 ініціалізація_мурахи;
 M = оновлення_пам'яті_мурахи;
 while поточний_стан ≠ повний_розв'язок do
      A=локальна_ матриця_мурашиних_маршрутів;
      сформувати_множину припустимих_вершин;
      p = обчислити_ймовірність_переходів(A, M, П);
      наступний_стан = правило_прийняття_рішення(p, П);
      перейти_в_наступний_стан(наступний_стан);
      if онлайнове_покрокове_оновлення_феромону then
         відкласти_феромон_на_відвіданій_дузі;
         поновити_ матрицю_мурашиних_маршрутів_A;
      endif
      M = оновити_внутрішній_стан;
    endwhile
    if онлайнове_відстрочене_оновлення_феромону then
      foreach відвіданої_дуги_побудованого_розв'язку do
         відкласти_феромон_на_відвіданій_дузі;
         оновити_матрицю_мурашиних_маршрутів_A;
      endforeach
 endif
 завершити діяльність;
 endforeach
 випаровування_феромону;
 оновлення_рекорду (x);
 дії_Демона;          {необов'язково}
 endwhile
end
```

РИС. 1. Псевдокод алгоритмів ОМК

Коли всі мурахи завершують роботу, може здійснюватися модифікація феромонних значень для певних дуг графа G (офлайнове оновлення), а також випаровування феромону (зниження значень τ_{ij}) для всіх його ребер, якщо такі дії не поєднані з оновленням феромону.

Оскільки алгоритм не має релаксаційного характеру, то необхідне запам'ятовування найкращого маршруту.

Найчастіше використовуються такі критерії завершення обчислень у мурашиних алгоритмах:

1. Відсутність (чи невелика зміна) поліпшення наявного рекорду чи середньої пристосованості популяції мурашок на кількох останніх ітераціях.

2. Перевищення заданої кількості $H_{\max}$ ітерацій (змін поколінь/популяцій), де $H_{\max}$ – параметр алгоритму.

3. Вичерпання заданого часу на обчислення.

Найважливішими аспектами мурашиних алгоритмів є: спосіб оновлення феромону, імовірності переходів до наступної вершини та правила завершення роботи алгоритму; більш детально з ними можна ознайомитися в [2, 3, 5].

2. Пропонована модифікація пошуку в алгоритмах ОМК.

В усіх відомих модифікаціях ОМК на кожному кроці для вершини, що є останньою в поточному фрагменті розв'язку (маршруту в графі задачі), формується множина припустимих сусідніх вершин і обчислюється імовірність переходу до кожної з цих вершин. На основі цих ймовірностей і вибирається чергова вершина для продовження наявного фрагменту маршруту [2, 3, 6].

Імовірнісне правило переходу мурахи k від поточної вершини $i \in V$ до нової вершини $j \in N_i^k$ у типових мурашиних алгоритмах базується на визначенні імовірності $p_{ij}^{\,k}$:

$$p_{ij}^{\,k} = \frac{a_{ij}(t)}{\sum\limits_{r \in N_i^k} a_{ir}(t)}, \tag{1}$$

де $a_{ij}(t)$ – елемент матриці мурашиних маршрутів, який залежить від значень феромонного сліду та евристичної інформації в момент часу t.

Найпоширенішими варіантами обчислення імовірностей при переході є

$$p_{ij}^{\,k} = \frac{\tau_{ij}^{\,\alpha}(t)\eta_{ij}^{\,\beta}(t)}{\sum\limits_{r \in N_i^k} \tau_{ir}^{\,\alpha}(t)\eta_{ir}^{\,\beta}(t)} \tag{2}$$

чи

$$p_{ij}^{\,k} = \frac{\tau_{ij}^{\,\alpha}(t) + \eta_{ij}^{\,\beta}(t)}{\sum\limits_{r \in N_i^k} [\tau_{ir}^{\,\alpha}(t) + \eta_{ir}^{\,\beta}(t)]}, \tag{3}$$

де α, β – параметри алгоритму, які відображають ступінь значущості феромонного сліду та евристичної інформації (α, $\beta > 0$).

Якщо кожне ребро $(i, j) \in E$ характеризується числовим значенням q_{ij}, що відображає ступінь його непривабливості (наприклад, довжина в ЗК), то використовують нормовані евристичні значення:

$$\eta_{ij} = 1 - q_{ij} \,/ \sum_{s \in N_i} q_{is}. \tag{4}$$

Тоді значення елементів матриці мурашиних маршрутів можна розраховувати як комбінацію феромонних та евристичних значень:

$$a_{ij}(t)=\frac{w\tau_{ij}(t)+(1-w)\eta_{ij}}{w+\frac{1-w}{\|N_i\|-1}}, \tag{5}$$

де $w\in[0,1]$.

У деяких алгоритмах застосовується псевдовипадкове пропорційне правило, засноване на використанні додаткового параметра $p_0\in[0,1]$: з імовірністю p_0 мураха переходить з вершини $i\in V$ до нової вершини $j\in N_i^k$, для якої максимізується добуток кількості феромону на величину евристичної інформації, піднесених до відповідних степенів, тобто

$$j=\arg\max\{\tau_{ir}^{\alpha}(t)\eta_{ir}^{\beta}(t):r\in N_i^k\},$$

водночас як з імовірністю $1-p_0$ обирається вершина за загальним правилом, тобто за формулою (1) з використанням (2) – (5).

Зрозуміло, що у випадку покладення $p_0=0$ отримуємо принцип вибору, який застосовується у звичайних алгоритмах ОМК; якщо ж покласти $p_0=1$, то працює лише псевдовипадкове пропорційне правило.

Таким чином, при формуванні маршруту мураха керується тільки інформацією, що знаходиться від неї на відстані одного переходу (кроку). Іншими словами, одне застосування імовірнісно-пропорційного правила приводить до одного кроку мурахи.

Далі пропонується новий підхід для формування маршруту, при якому мурахи на кожному кроці використовують інформацію не тільки з одного ребра, що може бути включене у фрагмент розв'язку, а також і інформацію з більшої кількості можливих ребер. Отже, на кожній ітерації мураха може додати до шляху одразу декілька вершин або, іншими словами, може зробити декілька «кроків» [4].

Так, для двокрокової версії алгоритму ОМК для кожної мурахи k за наявності уже побудованого фрагменту розв'язку $y=(\ldots,i)$ формується множина ще невідвіданих ребер (i,s), (s,j), $s\in N_i^k$, $j\in N_{(s)}{}^k$, де $N_{(s)}{}^k$ – множина припустимих для мурахи k вершин графа задачі за умови, що до існуючого фрагмента розв'язку додана вершина s, $y^+=(\ldots,i,s)$, та обчислюється імовірність переходу від вершини i до вершини j через вершину s з урахуванням евристичної інформації та поточних значень феромону. Для двокрокового алгоритму ОМК перехід k-ої мурахи з вершини i в j через вершину s на поточній ітерації t здійснюється з імовірністю, що може розраховуватися на основі (2) за наступною формулою:

$$p_{ij}{}^k=\frac{\tau_{is}{}^{\alpha}(t)\eta_{is}{}^{\beta}(t)}{\sum\limits_{r\in N_i^k}\tau_{ir}{}^{\alpha}(t)\eta_{ir}{}^{\beta}(t)}\frac{\tau_{sj}{}^{\alpha}(t)\eta_{sj}{}^{\beta}(t)}{\sum\limits_{r\in N_{(s)}{}^k}\tau_{sr}{}^{\alpha}(t)\eta_{sr}{}^{\beta}(t)}.$$

Зрозуміло, що аналогічним чином можна використати формули (3) – (5).

Можливість перегляду інформації на декілька кроків вперед дозволяє уникати субоптимальних розв'язків для деяких задач, а отже, отримувати більш якісний результуючий розв'язок. Ця стратегія може бути використана у будь-якій модифікації мурашиного алгоритму, що робить її універсальним засобом покращення ефективності алгоритмів ОМК.

3. Макс-мінний алгоритм мурашиних колоній.

Дослідження модифікацій алгоритмів ОМК показали, що підвищення продуктивності може бути отримано за допомогою більш інтенсивного використання кращих розв'язків знайдених у ході пошуку. Використання більш жадібних стратегій пошуку потенційно посилює проблему передчасної стагнації. Таким чином, ключем до найкращої роботи алгоритмів ОМК є комбінування підходу сильнішого використання кращих розв'язків із ефективними механізмами запобігання передчасній стагнації. Максі-мінний алгоритм мурашиних систем (MAX-MIN Ant System – MMAS) був розроблений спеціально для задоволення зазначених вище вимог [7, 8].

MAX-MIN модифікація алгоритму ОМК відрізняється від класичного алгоритму мурашиних систем (Ant System) трьома ключовими аспектами:

– для уникнення стагнації розв'язку вводиться нижня та верхня межі для можливих значень феромону на ребрі; тобто, використовується інтервал значень феромону, обмежений $\tau_{\min} \le \tau_{ij}(t) \le \tau_{\max}$, $\forall t, i, j$;

– початкове значення феромону встановлюється на рівні $\tau_{\min}$;

– для більш активного використання кращих розв'язків після кожної ітерації тільки один агент залишає слід феромону.

Останній аспект робить MMAS алгоритм схожим на іншу модифікацію – ACS [3, 5]. Різниця полягає у тому, що існує декілька схем вибору агента для оновлення феромону: завжди використовувати поточний найкращий розв'язок (рекорд); використовувати кращий розв'язок для даної ітерації; почергове використання кращого глобального розв'язку та кращого розв'язку для даної ітерації (наприклад, використовувати різні підходи в залежності від парності номера ітерації) [8].

Значення для верхньої і нижньої меж значень феромону, зазвичай, розраховуються за наступними формулами:

$$\tau_{\max}(t) = \frac{1}{1-\rho} \cdot \frac{1}{f^{gb}(t)},$$

$$\tau_{\min}(t) = \frac{\tau_{\max}(t)(1-\sqrt[n]{p_b})}{(\frac{n}{2}-1)\sqrt[n]{p_b}},$$

де $f^{gb}(t)$ – найкраще відоме на даний момент значення цільової функції, а p_b – імовірність отримати кращий розв'язок; значення цього параметра підбирається експериментальним шляхом і зазвичай наближається до 0.05.

При оновленні феромону здійснюється «обрізання» випадаючих значень феромону:

$$\tau_{ij}(t) = \begin{cases} \tau_{\max}(t), & \text{якщо } \tau_{ij}(t) > \tau_{\max}(t), \\ \tau_{\min}(t), & \text{якщо } \tau_{ij}(t) < \tau_{\min}(t). \end{cases}$$

3. Дослідження ефективності.

Для порівняння ефективності запропонованої модифікації алгоритму ОМК із бібліотеки TSPLIB вибрано ряд задач комівояжера з відомими оптимальними розв'язками [9]. Кожна задача розв'язана стандартним та модифікованим алгоритмом MMAS 20 разів, порівнювалися середні значення відносної похибки (відхилення довжини знайденого маршруту від довжини оптимального). Ця похибка розраховувалася за формулою

$$\varepsilon = \frac{f - f^{opt}}{f^{opt}} \times 100(\%),$$

де f – довжина шляху в знайденому відповідним алгоритмом розв'язку, f^{opt} – довжина оптимального розв'язку із бібліотеки TSPLIB.

При розв'язуванні задач застосовані наступні параметри:

- коефіцієнт впливу феромон $\alpha = 1$;
- коефіцієнт впливу відстані $\beta = 2$;
- коефіцієнт вивітрювання феромону $\rho = 0.5$;
- умова завершення роботи алгоритму: 2000 ітерацій;
- кількість мурах була еквівалентна кількості міст у задачі.

Всі алгоритми реалізовані на одній і тій же програмній базі на мові C#, пошук розв'язків здійснювався на персональному комп'ютері з 8 гБ оперативної пам'яті та чотирьохядерним процесором з тактовою частотою 3.4 ГГц, що дозволяє порівнювати отримані результати між собою[1].

Результати обчислювального експерименту для чотирьох ЗК показано на рис. 2, де показано середню відносну похибку стандартного («звичайного») і модифікованого («двокрокового») алгоритмів MMAS та розмірність задач, які розв'язувалися.

Відповідно до результатів обчислювального експерименту, використання двокрокової версії дозволило збільшити точність результатів на 0.02 % – 0.05 % у залежності від задачі. Таким чином, експериментальні дані підтверджують доцільність застосування запропонованої модифікації алгоритмів ОМК у випадках, коли точність отриманих результатів важливіша за швидкість знаходження розв'язку і навіть найменше її підвищення дає суттєвий ефект.

[1] Програмна реалізація здійснена О.С. Чегренцем в рамках виконання магістерської роботи під керівництвом автора

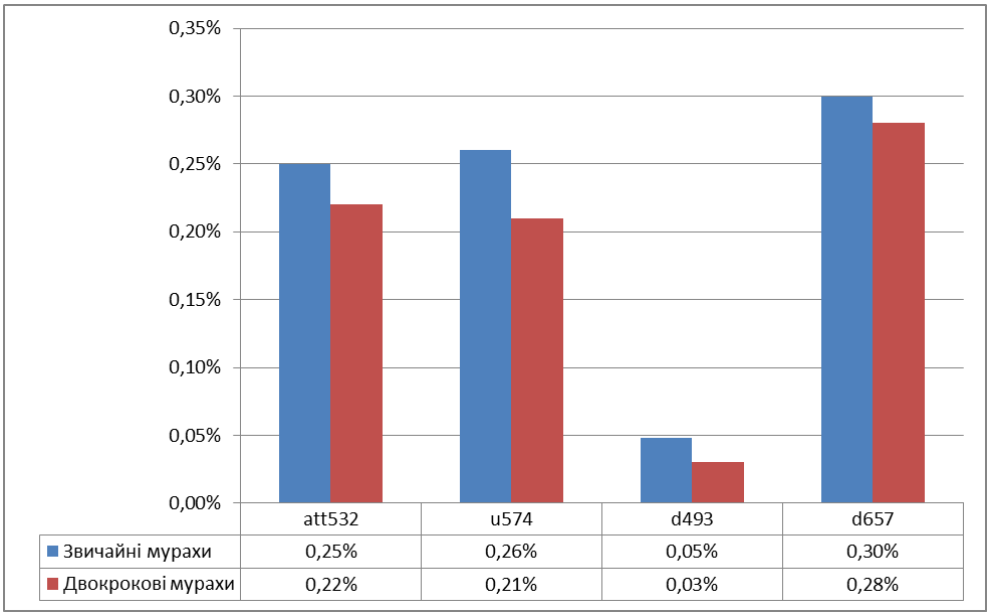

РИС. 2. Порівняння відносної похибки стандартного та модифікованого алгоритму MMAS

Висновки. У багатьох прикладних ЗКО навіть незначне покращення точності має дуже важливе значення, що і обумовлює потребу в розробці алгоритмів, які дозволяють досягти цього. Пропонований підхід дозволяє розширити сферу пошуку в алгоритмах ОМК, створюючи умови для пошуку покращених розв'язків шляхом уникнення передчасної збіжності – однієї з найгостріших проблем для більшості прикладних алгоритмів комбінаторної оптимізації.

Напрямами подальших досліджень можуть стати питання порівняльного аналізу практичної ефективності алгоритмів, розроблених на основі пропонованого підходу, та інших відомих прикладних алгоритмів комбінаторної оптимізації. Такий аналіз доречно провести і при розв'язуванні задач комбінаторної оптимізації з різних класів. Цікавим також є питання про кількість вершин графа задачі, які ефективно проглядати при добудові фрагмента розв'язку.

Перспективним також є використання модифікованих алгоритмів у гібридних метаевристиках [10], зокрема, у кооперативних метаевристиках [11].

Л.Ф. Гуляницкий

ДИВЕРСИФИКАЦИЯ ПОИСКА В АЛГОРИТМАХ ОПТИМИЗАЦИИ МУРАВЬИНЫМИ КОЛОНИЯМИ

Предложен подход к разработке новых алгоритмов оптимизации муравьиными колониями. Он ориентирован на создание алгоритмов решения задач комбинаторной оптимизации с повышенной точностью. Приведены результаты вычислительного эксперимента по решению серии задач коммивояжера из известной библиотеки, которые подтвердили возможность успешной модификации одного из самых эффективных муравьиных алгоритмов.

L.F. Hulianytskyi

SEARCH DIVERSIFICATION IN ANT COLONY OPTIMIZATION ALGORITHMS

The approach to new ant colony optimization algorithms development is proposed. It aims on creating the algorithms to solve optimization problems with higher precision. The results of computational experiments on solving a set of travelling salesman problems from benchmark library are presented and they confirm the possibility of successful modification of one of the most effective ant colony optimization algorithms.

1. *Pintea C.* Advances in Bio-inspired Computing for Combinatorial Optimization Problems. Springer, 2014. 188 p.
2. *Dorigo M., Stützle T.* Ant colony optimization. – Cambridge (MA): MIT Press, 2004. 348 p.
3. *Dorigo M., Blum C.* Ant colony optimization theory: A survey. *Theoretical Computer Science*. 2005. 344. 2 – 3. P. 243 – 278.
4. *Гуляницький Л.Ф.* Диверсифікація пошуку в алгоритмах ОМК. Abstracts of Int. Conf. "*Problems of Decision Making under Uncertainties (PDMU–2011)*": (September 19 – 23, 2011, Yalta, Ukraine). Київ, 2011. P. 66 – 67.
5. *Гуляницький Л.Ф., Мулеса О.Ю.* Прикладні методи комбінаторної оптимізації. К.: Видавничо-поліграфічний центр "Київський університет", 2016. 142 с.
6. *Штовба С.Д.* Муравьиные алгоритмы: теория и применение. *Программирование*. 2005. № 4. С. 1 – 16.
7. *Stützle T., Hoos H.H.* MAX-MIN ant system. *Future Gen. Comput. Systems*. 2000. 16 (8). P. 889 – 914.
8. *Stützle T., Hoos H.H.* The MAX-MIN Ant System and local search for the traveling salesman problem. IEEE International Conference on Evolutionary Computation (ICEC'97), 1996. P. 309 – 314.
9. *TSPLIB* – library of sample instances for the TSP [Електронний ресурс]. – Режим доступу: http://www.iwr.uni-heidelberg.de/groups/comopt/software/ TSPLIB95.
10. *Blum C., Puchinger J., Raid G.R., Roli A.* Hybrid metaheuristics in combinatorial optimization: A survey. *Applied Soft Computing*. 2011. **11**, 6. P. 4135 – 4151.
11. *Hulianytskyi L.F., Sirenko S.I.* Cooperative model-based metaheuristics. *Electronic Notes in Discrete Mathematics*. 2010. **36**. P. 33 – 40.

Одержано 25.01.2017