УДК 004.023; 004.94

О. Муляревич, В. Голембо
Національний університет “Львівська політехніка”,
кафедра електронних обчислювальних машин

ІМПЛЕМЕНТАЦІЯ МЕТОДІВ ЛОКАЛЬНОЇ ОПТИМІЗАЦІЇ У КОМП’ЮТЕРНІЙ СИСТЕМІ ДЛЯ РОЗВ’ЯЗАННЯ ДИНАМІЧНОЇ ЗАДАЧІ КОМІВОЯЖЕРА З ВИКОРИСТАННЯМ МОДЕЛІ РОЙОВОЇ ПОВЕДІНКИ АГЕНТІВ

Розглянуто імплементацію методів локальної оптимізації у комп’ютерній системі для розв’язання динамічної задачі комівояжера, що ґрунтується на використанні моделі ройової поведінки агентів.

Ключові слова: агенти, динамічна задача комівояжера, метод мурашиної колонії, методи локальної оптимізації, ройова поведінка.

This paper is devoted to the solving one of the combinatorial optimization task – the Dynamic Travelling Salesman Problem (DTSP) by using computer system based on swarm behavior model of collective agents and benefits of local optimization methods usage.

Key words: agents, DTSP, ant colony method, local optimization methods, swarm behavior.

Вступ

Сьогодні все більшої популярності набувають дослідження з розширення сфер застосування колективів автономних агентів. Особливої уваги заслуговують задачі комбінаторної оптимізації, до яких належать популярна задача комівояжера (ЗК) [1], яка полягає у одноразовому послідовному обходженні усіх точок графа агентом-комівояжером та поверненням до точки-старту з найменшими витратами, будь-то час, відстань, ціна тощо. Розроблена комп’ютерна система [2,3] з використанням моделі ройової поведінки агентів здатна розв’язувати як статичну (без змін вхідних даних під час обчислення результату), так і динамічну (в умовах динамічних змін вхідних даних) ЗК за прийнятний на практиці час обчислення. Проте отримувані результати для ЗК на кількість пунктів більше 100 можуть різнитись до 10 % від довжини оптимального маршруту. З метою покращення результату без втрати швидкодії та здатності розв’язувати ЗК в умовах динамічних змін вхідних даних було вирішено розглянути можливість використання методів локальної оптимізації як додаткового програмного модуля системи, на який подається вже сформований квазіоптимальний маршрут ЗК з метою його покращення.

Підходи до розв’язання задач комбінаторної оптимізації

За допомогою розробленої класифікації методів розв’язання ЗК [4] під час аналізу існуючих методів було обрано найперспективніші для розв’язання динамічної ЗК методи з використанням колективу агентів. Динамічна ЗК є найбільш наближена до реальності, особливо під час застосування для роз’язання задач логістики в транспортних та інформаційних мережах.

До методів з використанням колективу агентів належать такі методи ройового інтелекту: алгоритми “соціальних комах” (алгоритм “бджолиного рою”, алгоритм “мурашиної колонії” [5,6]), метод ройових часток, алгоритм бактеріального рою, алгоритм зграї риб, алгоритм зграї птахів [7] та інші [5,8]. Алгоритм “мурашиної колонії” продемонстрував одні з кращих результатів розв’язання динамічних ЗК, істотно також, що цей метод здатний розв’язувати ЗК без потреби перезапуску обчислення завдяки конструктивному характеру та в умовах частково невідомих вхідних даних.

Окреме місце займають методи локального пошуку [9], або як їх ще називають – методи локальної оптимізації. Ці методи дозволяють покращити результат – зменшити довжину

отриманого результуючого маршруту ЗК. Методи локальної оптимізації розрізняються згідно з коефіцієнтом складності k, де k – це кількість ребер, які беруть участь у перестановках. Найпоширенішими є методи низької складності (алгоритми 2-opt, 3-opt, проміжна форма *2.5*-opt) та метод локальної оптимізації з динамічно змінюваним коефіцієнтом складності k [10]. (рис. 1).

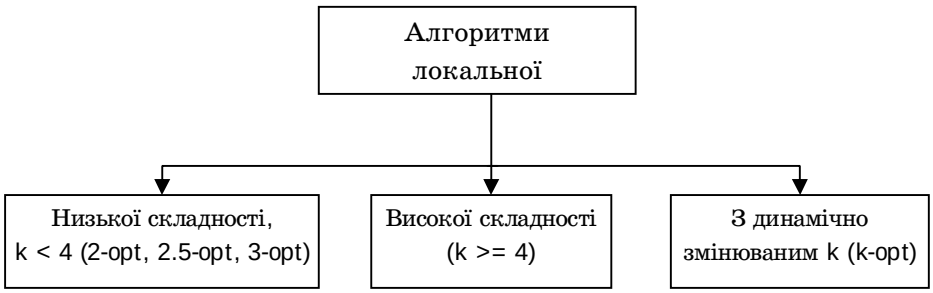


Рис. 1. Класифікація алгоритмів локальної оптимізації розв'язку ЗК

У класичній реалізації алгоритм 2-opt вилучає 2 ребра, що входять до маршруту і додає два нові так, щоб новий маршрут став коротшим. Причому існує лише один варіант заміни двох ребер на нові два, при якому новий маршрут залишиться маршрутом, а не утворяться 2 цикли.

Аналогічно використовуються й інші алгоритми локальної оптимізації з різницею лише у тому, що замість двох ребер використовують три ребра для оптимізації у 3-opt алгоритмі, або ж **k** ребер у найбільш гнучкому і ефективному k-opt алгоритмі з динамічно змінюваним коефіцієнтом k, який був розроблений як модифікація алгоритмів локальної оптимізації та відомий під назвою - метод Ліна-Кернігана [9]. У цьому методі змінна k динамічна величина, що змінюється під час обчислення остаточного результату. Для розв'язання ЗК з використанням методу Ліна-Кернігана та методів декомпозиції К.Хельсгаун розробив програмну аплікацію LKH [9].

Особливості застосування методів локальної оптимізації в розробленій системі

Методи локальної оптимізації [9] є метаевристичними методами розв'язання складних у обчисленні оптимізаційних задач. Методи локальної оптимізації можуть використовуватись для задач, що формулюються як знаходження розв'язку, максимального за певним критерієм, серед існуючих можливих рішень. Алгоритми локальної оптимізації перебирають можливі розв'язки методом виконання локальних змін поки результат не зведеться до оптимального або не буде вичерпано певний ліміт часу чи кількість спроб. Для розв'язання ЗК найпопулярнішими є k-opt алгоритми [9,11].

Більшість задач можна сформулювати в термінах "простору" та "цілі" декількома різними шляхами. Так, наприклад, для ЗК розв'язок може бути гамільтоновим циклом, а критерієм максимізації буде комбінація певної кількості пунктів та довжина циклу. Але розв'язком може також бути шлях, а утворити цикл буде ще одною частиною завдання для локального пошуку. Цей підхід використовується в програмній аплікації LKH при "склеюванні" окремих маршрутів. Алгоритми локальної оптимізації починають аналіз від вхідного певного розв'язку або його частини та послідовно переходять до сусіднього. Це є можливим тільки тоді, коли відношення сусідства між розв'язками, чи його частинами, визначено в просторі пошуку. Як приклад, якщо в сформованому шляху – розв'язку ЗК переставити лише один пункт, то отримаємо новий шлях. Початковий та отриманий новий шляхи є сусідніми. Отже, для результату ЗК кількість сусідніх розв'язків залежить від кількості пунктів. Відповідно до складності методу локальної оптимізації, збільшується кількість перестановок – k, що визначає кількість ребер, яку буде переставлено, та відповдіно кількість сусідніх розв'язків для аналізу.

Умова припинення виконання алгоритму зазвичай визначається як обмеження по часу або ж у кількості спроб проведення локальної оптимізації. Алгоритми локальної оптимізації можуть бути зупинені у будь-який час виконання, проте здатні працювати лише за відсутності динамічних змін в

процесі їх виконання. Крім того, алгоритми локальної оптимізації по завершенню свого виконання, тобто коли можливих локальних покращень не виявлено, не гарантують найоптимальнішого результату. Це зазвичай відбувається тоді, коли складності методу локальної оптимізації не достатньо для виявлення кращого сусіднього розв'язку.

Алгоритм Ліна-Кернігана на практиці є доволі ефективним, проте в деяких випадках приймає експоненційні складність та часові витрати на обчислення результату. Особливістю алгоритму Ліна-Кернігана є те, що він працює зі змінною k, значення якої перевизначається в кожній ітерації локального пошуку.

При встановленні списку сусідів для вузла метод локальної оптимізації зменшує складність з $O(n^2)$ до O(m*n), однак, якщо m взяти недостатньо великим, то ефективність алгоритму локальної оптимізації зменшується. Найчастіше застосовуються алгоритми низької складності: 2-opt, 2.5-opt, 3-opt алгоритми. Згідно з твердженням С. Ліна [9], 3-opt алгоритм продемонстрував найкращі результати за обчислювальним часом.

Отже, методи локальної оптимізації з $k < 4$ можуть істотно покращити точність результатів, що отримуються розробленою комп'ютерною системою для розв'язання ЗК динамічного характеру на базі методу мурашиної колонії з врахуванням розроблених модифікацій, без значного збільшення часу обчислення. Аналіз на можливість оптимізації ділянки маршруту не анулює можливість обчислення ЗК динамічного характеру, якщо час витрачений на аналіз є значно меншим за період динамічних змін.

Програмна реалізація методів локальної оптимізації

Використання методів локальної оптимізації збільшує точність отриманого результату за рахунок перестановок ребер в знайденому машруті. Для імплементації методів локальної оптимізації необхідно розробити програмну реалізацію обраних через невелику складність та час обчислення трьох алгоритмів: 2-opt, 2.5-opt, 3-opt. Реалізація цих методів для розв'язання ЗК в доступних джерелах інформації відсутня. Через це було вирішено викласти хід виконання перестановок перелічених алгоритмів локальної оптимізації, які були використані для реалізації додаткового програмного модуля та його імплементацію в розроблену комп'ютерну систему.

Розглянемо довільно обрану ділянку графа, з вже прокладеними з'єднаннями між точками. Вузли, що беруть участь у перевірці на доцільність перестановки, позначимо як s1 та s2 (station). Попередній до s1 вузол згідно з маршрутом – p_s1 (previous to s1), наступний – s_s1 (sequent after s1). Аналогічно і до вузла s2: p_s2 та s_s2. Якщо поточний пункт s1 ще не був оптимізований, проводимо пошук пункту s2 так, щоб відстань між s1 та s2 була менша за відстань між s1 та s_s1, яка позначена на схемі як R – радіус на рис. 2.

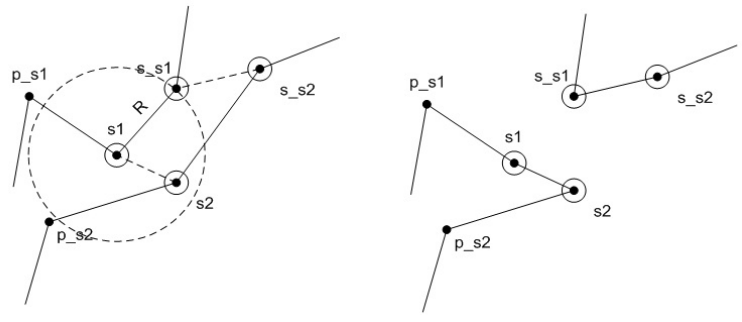


Рис. 2. Виконання в 2-opt алгоритмі перестановки першого типу на довільно обраній ділянці графа

Після чого виконується перевірка доцільності перестановки: вилучення ребер s1-s_s1 та s2-s_s2, додавання ребер s1-s2 та s_s1-s_s2. В подальшому будемо називати цю перестановку як 2-opt перестановку 1-го типу: перестановка з задіяними такими пунктами за маршрутом відносно поточного s1 та знайденого s2 пунктів. Виконання цієї перестановки для певної ділянки маршруту

зображено на рис. 2. Тут і надалі зліва зображено початковий стан, справа – стан маршруту після перестановки. Колами обведено пункти, що беруть участь у перестановці.

Якщо ця перестановка (рис. 2.) зменшить поточний маршрут за вартістю, то відбувається перехід до етапу виконання перестановки, тобто відповідної корекції маршруту, поточний пункт відмічається як оптимізований, та відбувається перехід до наступного. Якщо ж ця перестановка не доцільна, то відбувається пошук пункту s2 (рис. 3) так, щоб тепер відстань між p_s1 та p_s2 була менша за відстань між p_s1 та s1, яка позначена на схемі як R – радіус на рис.3. Після чого виконується вилучення ребер p_s1-s1 та p_s2-s2, додавання ребер p_s1-p_s2 та s1-s2. Будемо називати її 2-opt перестановку другого типу, в якій задіяні попередні за маршрутом пункти до поточного s1 та деякого s2. Такий тип перестановки для деякої ділянки маршруту зображено на рис. 3.

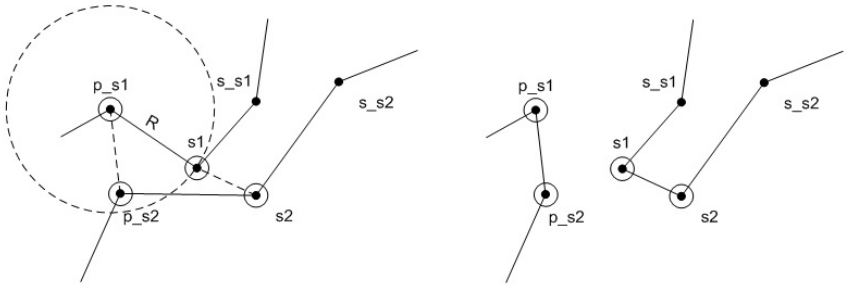


Рис. 3. Виконання в 2-opt алгоритмі перестановки другого типу на довільно обраній ділянці графа

Якщо 2-opt перестановка другого типу зменшить поточний маршрут за вартістю аналогічно як і до першого типу перестановки, то виконується перехід до корекції шляху та продовження циклу локальної оптимізації.

Фактично цей алгоритм за потреби можна зупинити на будь-якому етапі. Аналогічно до розглянутого алгоритму 2-opt реалізовано метод 2.5-opt, який включає в себе певні етапи 2-opt алгоритму та елементи 3-opt алгоритму, оскільки при вилученні пункту з одного місця маршруту та вставки його в іншу ділянку також задіяні три ребра.

Спочатку виконуються спроби оптимізації 2-opt перестановкою першого і другого типу, аналогічно до 2-opt алгоритму. Після чого відбувається аналіз доцільності вставки міста s2, з пулу сусідніх пунктів відносно поточного, між пунктами s1 та s_s1 – будемо називати цю вставку 2.5-opt вставкою першого типу. Цю вставку для певної ділянки маршруту ЗК зображено на рис. 4.

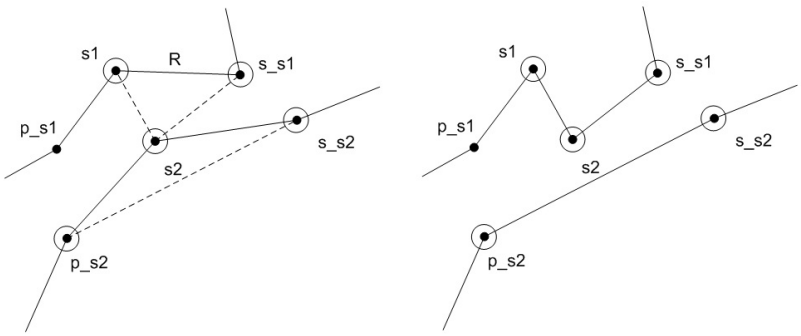


Рис. 4. Виконання в 2.5-opt алгоритмі вставки першого типу на довільно обраній ділянці графа

Якщо цей тип вставки не зменшує маршрут за вартістю, то аналізуємо на доцільність вставки в маршрут деякого пункту s2 між пунктами p_s1 та s1. Цей тип вставки називатимемо 2.5-opt вставкою другого типу, яка зображена на рис. 5. Метод локальної оптимізації 3-opt функціонує потрібно до двох попередньо розглянутих. Етап оптимізації є набагато складнішим відмінно від 2-opt та 2.5-opt, де було, відповідно, лише 2 та 4 перевірки.

Для 3-opt виконуються спочатку дві 2-opt перевірки (рис.6), при чому якщо вони доцільні, є можливість закінчити оптимізацію, чи зберегти результат та продовжити аналіз наступних можливих перестановок (рис. 7).

На рис. 6 зображено 2-opt перестановку першого типу відносно пункту s1 (рис. 6, *а*) та 2-opt перестановку другого типу відносно пункту s_s1 (рис.6, б). Як і на попередніх рисунках, пункти, що беруть участь у перестановці ребер, обведено колами. Пунктиром наведено ребра, що утворюються після перестановки.

В першому випадку вилучається ребро s2-s_s2, в другому – p_s2-s2, ребро s1-s_s1 вилучається в обох випадках.

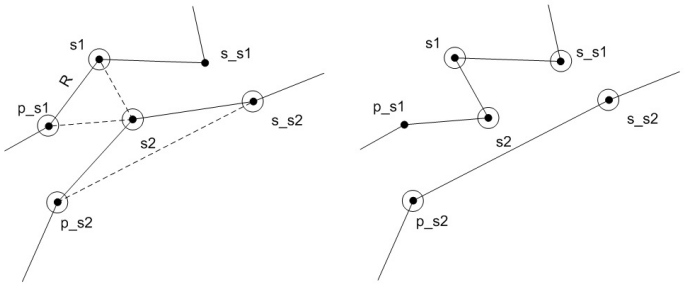
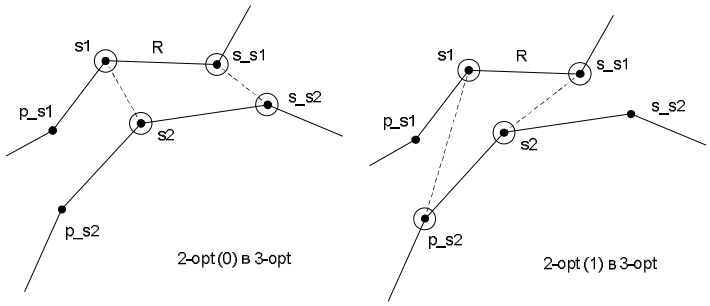


Рис. 5. Виконання в 2.5-opt алгоритмі вставки другого типу на довільно обраній ділянці графа



а *б*

Рис. 6. Виконання 2-opt перестановок в 3-opt алгоритмі на довільно обраній ділянці графа

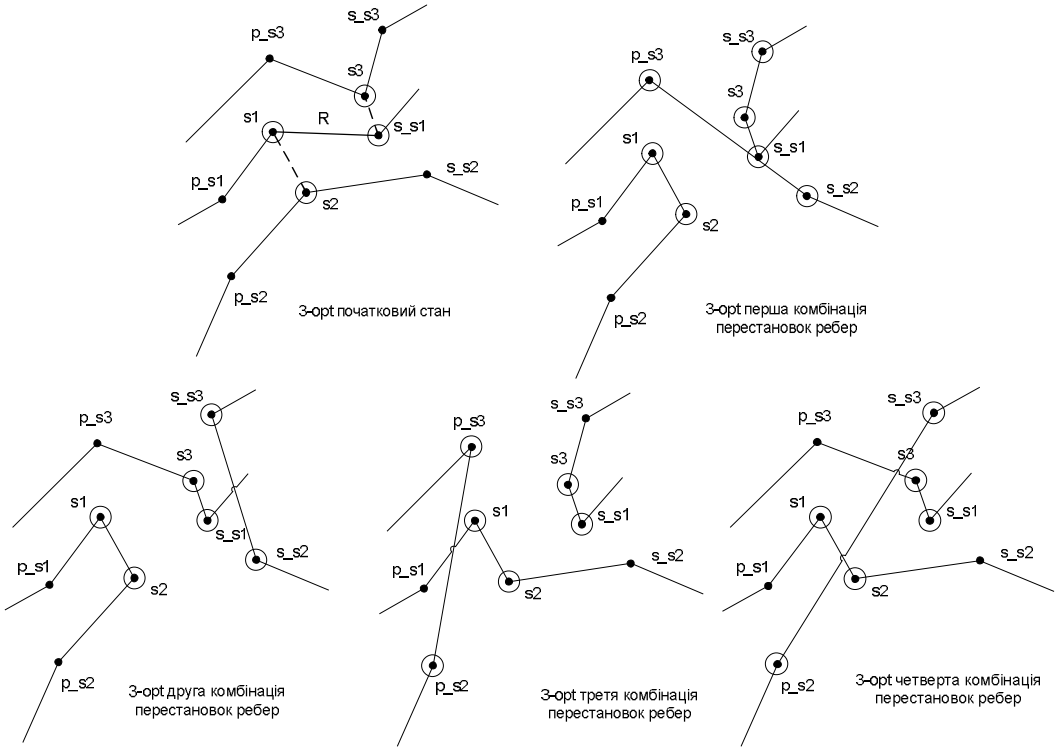


Рис. 7. Комбінації перестановок в 3-opt алгоритмі на довільно обраній ділянці графа

Після чого відбувається подальша перевірка на доцільність перестановок 2-х ребер із знайденим ще при 2-opt перевірці пунктом s2 та поточно знайденим пунктом s3, що є сусіднім для пункту s_s1. Причому пункти s2 та s3 обрані так, щоб ребра s3-s_s1 та s1-s2 були менші за ребро s-s_s1, яке позначено як R – радіус.

Якщо перестановки з вилученням ребра s1-s_s1 та додаванням ребер s1-s2 та s_s1-s3 є доцільними, то проводиться пошук комбінації перестановки, яка дозволяє досягнути найбільшого зменшення довжини маршруту. У розробленій реалізації 3-opt алгоритму аналізують чотири варіанти комбінацій перестановок ребер, зображені на рис.7. Ребро s1-s_s1 вилучається у кожній з наведених комбінацій.

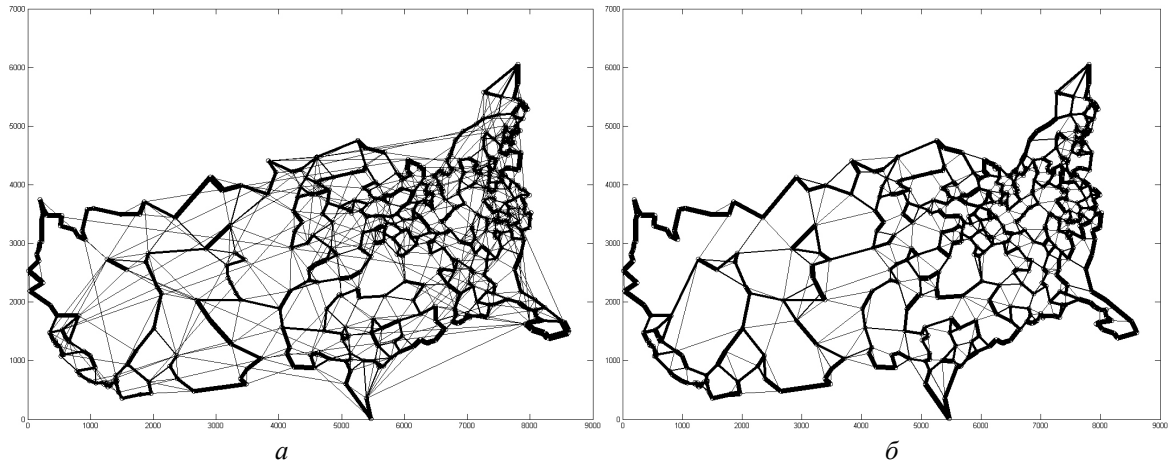
Після перевірки кожної з комбінацій зберігається можлива довжина покращеного маршруту для підсумкового вибору найоптимальнішої перестановки та її виконання.

Якщо після проведення корекції покращити результуючий маршрут не вдалось, то поточний пункт s1 відмічається в списку оптимізованих та береться наступний за випадковою послідовністю, ще неоптимізований, пункт маршруту ЗК. Згідно з проведеними досідженнями, було вирішено використовувати методи локальної оптимізації для покращення усіх знайдених агентами маршрутів.

Дослідження та оцінка результатів імплементації додаткового програмного модуля на базі методів локальної оптимізації в розроблену комп'ютерну систему

Для проведення дослідження та оцінки результатів імплементації додаткового програмного модуля на базі методів локальної оптимізації в розроблену комп'ютерну систему було використано бібліотечний файл att532.tsp [12], який включає вхідні дані для ЗК на 532 точки (в даному випадку координати міст в США). Під час розв'язання ЗК методом мурашиної колонії відбувається створення так званої "мурашиної пам'яті" - мережі феромонів, тобто відкладених міток, на які мурахи орієнтуються при пошуку маршруту. Саме за допомогою її існування агенти здатні легко пристосовуватись до динамічних змін початкових вхідних умов в процесі розв'язання ЗК. На рис. 8 наведено вигляд такої "пам'яті мурашника" для ЗК на 532 точки. Товщина ребра графа тим більша, чим більша величина цифрової мітки (у мурах – феромони), відкладеної на ньому.

На рис. 8 показано стан міток після виконання 10 ітерацій запуску циклу пошуку шляхів. На рис. 8, *а* показано стан міток без використання методів локальної оптимізації, на рис. 8, *б* – з використанням 3-opt алгоритму локальної оптимізації. Як бачимо, застосування методів локальної оптимізації пришвидшує процес зникнення додаткових ребер, що усуваються як непридатні, через періодичне зменшення значення міток на них та відсутність оновлення – проходження через них агентів.



а *б*

Рис. 8. Вигляд пам'яті колективу агентів ("пам'ять мурашника") – матриці міток під час розв'язання ЗК (532 точки): а) без застосування методів локальної оптимізації; б) з застосуванням 3-opt алгоритму локальної оптимізації

У таблиці наведено хід обчислення результату ЗК протягом перших трьох ітерацій: 1) без застосування методів локальної оптимізації; 2) з застосуванням після кожної ітерації запуску циклу пошуку шляхів агентами додатково одного з імплементованих методів локальної оптимізації: 2-opt, 2,5-opt, 3-opt. В таблиці також наведено середній час обчислення результату ЗК в розглянутих режимах функціонування розробленої комп'ютерної системи.

Хід обчислення результату ЗК (532 точки).

	Довжина маршруту				Кількість покращень \ перестановок			Середній час обчислення, с			
I	без застосування методів локальної оптимізації	з застосуванням методу локальної оптимізації:			2-opt	2,5-opt	3-opt	без застосування методів локальної оптимізації	з застосуванням методу локальної оптимізації:		
		2-opt	2,5-opt	3opt					2-opt	2,5-opt	3opt
1	49787	30105	29387	28780	428	1239	614	0.9	4.4	4.6	4.95
2	39067	29806	29327	28675	202	1012	625				
3	38939	29665	29202	28375	224	797	625				

Оптимальний маршрут для цієї ЗК має довжину 27686 (згідно з даними з бібліотечного файла). Маршрут такої довжини було отримано розробленою системою моделювання в режимі з використанням методу 3-opt локальної оптимізації. Як бачимо застосування кожного методу локальної оптимізації відповідно до його складності збільшує час обчислення маршруту, проте значно зменшує кількість ітерацій циклу пошуку шляху агентами та збільшує точність отримуваного результату. На рис. 9 наведено графічно розв'язки ЗК на 532 точки в чотирьох режимах функціонування системи після 30 циклів "запуску мурах": без використання методів локальної оптимізації (рис. 9, *а)*; з застосуванням 2-opt методу локальної оптимізації (рис. 9, *б*); з застосуванням 2,5-opt методу локальної оптимізації (рис. 9, *в*); з застосуванням 3-opt методу локальної оптимізації (рис. 9, *г*).

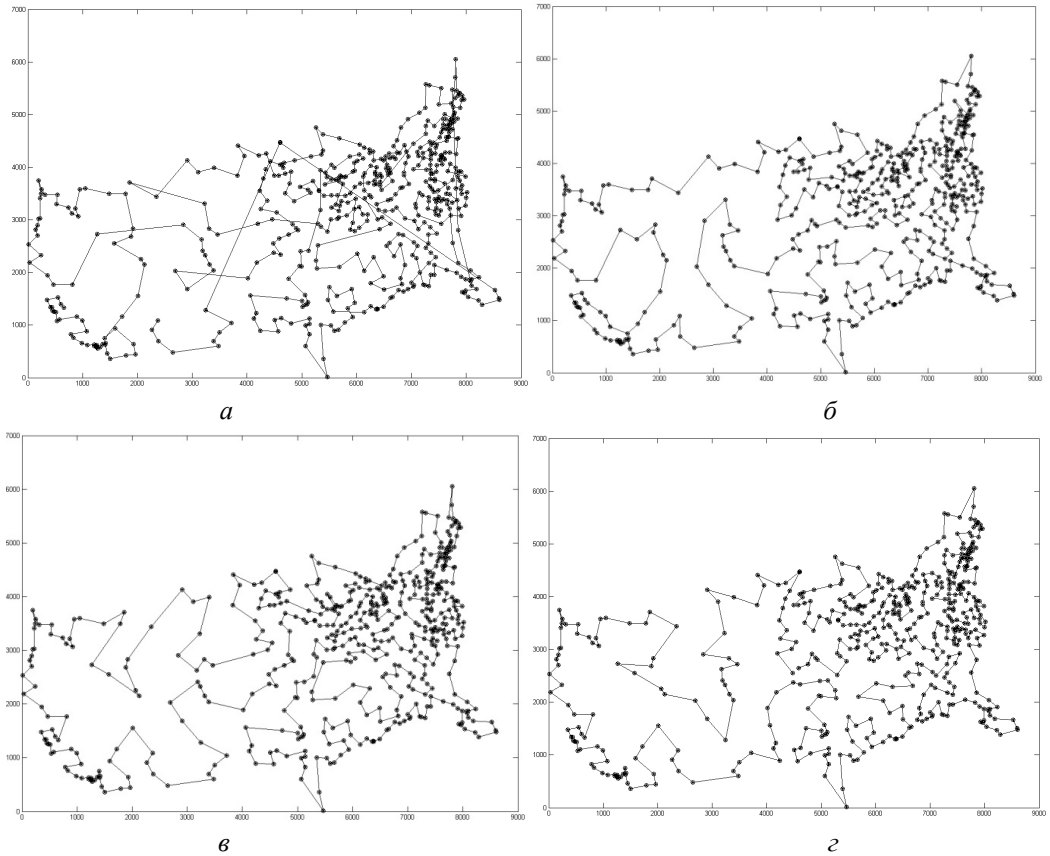


Рис. 9. Результуючі маршрути ЗК (532 точки)

Довжина результуючого маршруту, як видно з таблиці та рисунків, зменшується відповідно до збільшення складності застосовуваного методу локального пошуку. Розроблена комп'ютерна система на базі моделі ройової поведінки агентів, навіть, з використанням методів локальної оптимізації, здатна розв'язати ЗК на 532 точки за менший час, ніж програмна аплікація LKH, якій необхідно в середньому 7 секунд для розв'язання ЗК.

Розроблена система здатна видати квазі-оптимальний маршрут з різницею від оптимального маршруту, що становить менше ніж 6 %, вже за 1 с. Конструктивний характер використовуваного методу "мурашиної колонії" робить можливим знаходження маршруту вже після першого ітерації циклу пошуку шляхів, тобто за декілька мілісекунд, після чого цей результат обробляється в додатковому програмному модулі на базі методів локальної оптимізації з метою зменшення довжини отриманого маршруту.

Висновки

Імплементація методів локальної оптимізації у розробленій комп'ютерній системі для розв'язання ЗК з використанням моделі ройової поведінки агентів дозволяє зменшити різницю між квазі-оптимальним та оптимальним маршрутами та знизити кількість ітерацій циклу пошуку маршруту агентами. Так, при розв'язанні ЗК на 532 точки система моделювання в умовах динамічних змін вхідних даних вже за 1 с здатна видавати маршрут з різницею від оптимального менше ніж 6%. А за відсутності динамічних змін вхідних даних через 5 с видати оптимальний результат з використанням 3-opt методу локальної оптимізації.

1. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы. Построение и анализ, – 2-ое изд. – Вильямс, 2005. 2. Голембо В.А., Муляревич О.В. Модифікація методу мурашиної колонії для розв'язання задачі комівояжера колективом автономних агентів // Вісник Нац. ун-ту "Львівська політехніка". – 2011. – №717: Комп'ютерні системи та мережі. – 24 – 30 с. 3. Голембо В.А., Бочкарьов О.Ю., Муляревич О.В. Нові підходи до розв'язку задач комбінаторної оптимізації колективом автономних агентів // Матер. 5-ї Міжнар. наук.-техн. конф. ACSN-2011 "Сучасні комп'ютерні системи та мережі: розробка та використання". – Львів. – 2011. – 227 – 230 с. 4. Муляревич О. Переваги застосування колективної поведінки агентів для розв'язку задачі комівояжера динамічного характеру // Тези доп. студентської секції "Кібер фізичні системи в метрології" IX Міжнар. наук.-техн. конференції "Методи і засоби вимірювань фізичних величин" – "Температура-2012", 25–28 вересня 2012р. – Львів: ПП Сорока Т.Б., 2012. – 165 – 168 с. 5. Bonabeau E., Dorigo M., Theraulaz G. Swarm intelligence: From Natural to Artificial Systems. – Oxford University Press, 1999. 6. Stützle T., López-Ibáñez M., Pellegrini P., Maur M., Oca M., Birattari M., Michael Maur, Dorigo M. "Parameter Adaptation in Ant Colony Optimization" // Technical Report, IRIDIA, Université Libre de Bruxelles, 2010. 7. Vahit Tongur and Erkan Ulker Migrating Birds Optimization for Traveling Salesman Problem // Матеріали 6-ї Міжнародної науково-технічної конференції ACSN-2013 "Сучасні комп'ютерні системи та мережі: розробка та використання". – Львів. – 2013. – 219 – 223 с. 8. Субботін С.О., Олійник А.О., Олійник О.О. Неітеративні, еволюційні та мультиагентні методи синтезу нечіткологічних і нейромережних моделей: Монографія / Під заг. ред. С.О. Субботіна. – Запоріжжя: ЗНТУ, 2009. 9. Helsgaun K. General k-opt submoves for the Lin-Kernighan TSP heuristic // Mathematical Programming Computation , 2009. – 119 – 163 p. 10. Базилевич Р., Кутельмах Р. Дослідження ефективності існуючих алгоритмів для розв'язання задачі комівояжера. – Львів: Нац. ун-ту "Львівська політехніка", Вісник № 638, 2009. – 235 – 244 с. 11. Hoos H.H., Stützle T. Stochastic Local Search: Foundations and Applications. – San Francisco: Morgan Kaufmann Publ., 2005. 12. TSPLIB official homepage. Exist from 1995, © Copyright Universität Heidelberg. URL: http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/