МАТЕМАТИЧНЕ МОДЕЛЮВАННЯ ТА ОБЧИСЛЮВАЛЬНІ МЕТОДИ

УДК 004.023

Ю. М. ЗОРІН, С. В. ПОДОЛЬСКИЙ

Національний технічний університет України “Київський політехнічний інститут”, м. Київ

РОЗВ’ЯЗАННЯ КВАДРАТИЧНОЇ ЗАДАЧІ ПРО ПРИЗНАЧЕННЯ МЕТОДОМ ЛОКАЛЬНИХ ОПТИМУМІВ

Анотація.У роботі запропоновано евристичний алгоритм розв’язання квадратичної задачі про призначення. На відміну від існуючих евристичних алгоритмів, які потребують визначення декількох вхідних параметрів, запропонований алгоритм використовує лише один параметр і базується на ідеї використання комбінації локальних оптимумів в околі перестановок двох елементів. При цьому він дозволяє отримати більш якісні розв’язки за дещо менший час в порівнянні з відомими алгоритмами.

Ключові слова: **квадратична задача про призначення, евристичний алгоритм, комбінаторна оптимізація, локальний мінімум.**

Аннотация. В работе предложен эвристический алгоритм решения квадратичной задачи о назначениях. В отличие от известных эвристических алгоритмов, требующих задания нескольких входных параметров, предлагаемый алгоритм использует только один параметр, и базируется на идее использования комбинации различных локальных оптимумов в окрестности перестановок двух элементов. При этом он позволяет получить болем качественные решения за несколько меньшее время по сравнению с известными алгоритмами.

Ключевые слова: квадратичная задача о назначениях, эвристический алгоритм, комбинаторная оптимизация, локальный минимум.

Abstract. The paper presents a new heuristic algorithm for the quadratic assignment problem. Unlike existing heuristic algorithms that require a few input parameters setting, the proposed algorithm uses only one parameter and is based on the idea of exploiting a combination of different local optima in the neighborhood of the two elements permutations. At the same time it allows to obtain solutions of better quality in slightly lesser time compared to the known algorithms.

Keywords: quadratic assignment problem, heuristic algorithm, combinatorial optimization, local minimum.

Вступ

Квадратична задача про призначення [1] по праву вважається однією з найскладніших задач комбінаторної оптимізації. Проблема формулюється таким чином. Є N об'єктів і N їх місць призначення, задані відстані між місцями призначення та інтенсивності потоків ресурсів між об'єктами. Задача полягає в розміщенні об'єктів у місцях призначення таким чином, щоб мінімізувати суму добутків відстаней на інтенсивність відповідних потоків. Інакше кажучи, задана цільова функція

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} d_{ij} f_{\pi(i)\pi(j)},$$

де $\pi(x)$ - номер об'єкта , призначеного місцю з номером x, d_{ij} - відстань між місцями i і j , $f_{\pi(i)\pi(j)}$ - інтенсивність потоків між об'єктами $\pi(i)$ і $\pi(j)$. Оскільки задача відноситься до класу NP - hard, не існує " точних " алгоритмів, що дозволяють розв’язати її за поліноміальний час . До речі, задача про комівояжера може розглядатися як спеціальний випадок квадратичної задачі про призначення, в якому всі об'єкти об'єднані за допомогою інтенсивностей потоків з константним значенням в кільце, а всі інші потоки дорівнюють нулю. Таким чином , квадратична задача про призначення є більш складною ніж задача про комівояжера .

Окрім значного теоретичного інтересу, який викликає квадратична задача про призначення, існує широкий спектр практичних її застосувань, включаючи такі області як планування, паралельні та розподілені обчислення, статистичний аналіз даних, балансування запуску турбін і виробництво комп’ютерів [2]. Серед найбільш відомих евристичних алгоритмів розв’язання квадратичної задачі про призначення можна згадати Fast Ant System [3] і Simulated Annealing [4].

Мета дослідження

Метою роботи є розробка евристичного алгоритму розв’язання квадратичної задачі про призначення, який не поступається відомим в швидкодії й якості розв’язків, але не потребує налаштування численних параметрів, яке зазвичай здійснюється експериментальним шляхом і вимагає значних зусиль.

Алгоритм розв’язання квадратичної задачі про призначення.

В основі роботи алгоритму лежить використання локальних оптимумів, що не повторюються. Пошук локального оптимуму ґрунтується на тому, що за поліноміальний час можна перевірити доцільність

всіх *N(N*-3*)/*2 можливих перестановок пар елементів в деякому векторі, що є кандидатом на розв'язок задачі розміром ***N*** [5].

Кожна перестановка пари елементів i і j розв'язку π призводить до зміни значення цільової функції на деяке значення $\Delta(\pi, i, j)$, яке може бути обчислено за формулою

$$\Delta(\pi,i,j) = (d_{ii} - d_{jj}))(w(\pi_j,\pi_j) - w(\pi_i,\pi_i)) + (d_{ij} - d_{ji})(w(\pi_j,\pi_j) - w(\pi_i,\pi_i)) +$$

$$+ \sum_{k \neq i,j}((d_{ki} - d_{kj})(w(\pi_k,\pi_j) - w(\pi_k,\pi_i)) + (d_{ik} - d_{jk})(w(\pi_j,\pi_k) - w(\pi_i,\pi_k)))$$

Обчислювальна складність такої операції становить *O(n)*.

Здається, що при перевірці, або здійсненні, деякої перестановки розглядати її повторно недоцільно. Однак в подальшому, ця перестановка може виявитися корисною, якщо після її застосування хоча б одна інша перестановка виявиться ефективною і буде виконана. При цьому можна зробити евристичне припущення, що перестановки, які були здійснені раніше, є оптимізуючими з більшою ймовірністю, ніж ті, що були здійснені нещодавно. Це припущення базується на тому факті, що чим більше перестановок було здійснено після деякої конкретної перестановки, тим більше добутків відстаней на відповідні потоки пари об'єктів цієї перестановки могло потенційно зменшитись за абсолютним значенням. Виходячи з цього розроблено такий алгоритм знаходження локального оптимуму.

1. Створити список допустимих перестановок та заповнити його випадковою послідовністю всіх можливих перестановок пар індексів масиву розміру *N*.
2. Створити порожній список заборонених перестановок.
3. Якщо список допустимих перестановок порожній, закінчити пошук.
4. З ймовірністю, пропорційною номеру перестановки, нумеруючи з кінця списку допустимих, вибрати з нього випадкову перестановку пар індексів.
5. Якщо вибрана перестановка є оптимізуючою, виконати її й додати список заборонених перестановок в кінець списку допустимих. Список заборонених перестановок після цього встановлюється порожнім.
6. Додати вибрану на 4-му кроці перестановку в кінець списку заборонених.

У пункті 4 при ймовірнісному виборі індексу наступної перестановки замість використання колеса рулетки, алгоритмічна складність якого становить ***O(N)***, пропнується вибір індексу перестановки за допомогою відображення $R \to i$, де R – генератор псевдовипадкових чисел з рівномірним розподілом, i – індекс наступної перестановки у списку перестановок, яка буде використана. Зазначене відображення виконується таким чином. Нехай кількість доступних перестановок дорівнює ***S***. Необхідно обрати перестановку з ймовірністю, пропорційною індексу ***i*** цієї перестановки в списку. Розглянемо суми виду

$$1 + 2 + 3 + ... + i = \frac{i(i+1)}{2}$$

$$1 + 2 + 3 + ... + i + (i+1) = \frac{i(i+1)(i+2)}{2}$$

Тоді при попаданні величини R в діапазон від $\frac{i(i+1)}{2}$ до $\frac{i(i+1)(i+2)}{2} - 1$ включно можна знайти індекс ***i***, що задовольняє зазначеним вище умовам. Для натурального R, розв'язавши квадратне рівняння

$$R = \frac{i(i+1)}{2}, R \in \left[1, \frac{S(S+1)}{2}\right],$$

отримаємо індекс

$$i = \frac{-1 + \sqrt{1 + 8R}}{2}$$

Складність такої операції становить *O(1)*. Експериментально встановлено [5], що запропонована техніка вибору індексу перестановки дає збільшення швидкодії приблизно в 1,5 рази у порівнянні з підходом, коли всі перестановки обираються з однаковою ймовірністю.

В запропонованому алгоритмі введені поняття «інтенсивність» та матриця інтенсивностей M, елементи m_{ij} якої відповідають бажаності призначення j-го об’єкта i-му місцеположенню. Використовуючи запропоновану матрицю інтенсивностей, на кожній ітерації алгоритму виконується конструювання розв’язків, які потім і приводяться до локальних оптимумів за зазначеним вище алгоритмом. У випадку, якщо внаслідок конструювання отримуємо локальний оптимум, очевидним є факт передчасної збіжності алгоритму, що потребує виконання диверсифікації.

Конструювання нового розв’язку виконується за таким алгоритмом.

1. Із списку місцеположень, яким ще не призначені об’єкти, обрати випадковим чином з рівномірною ймовірністю довільне місцеположення i.
2. За допомогою колеса рулетки серед ще непризначених об’єктів вибрати для даного місцеположення об’єкт j з ймовірністю, що є пропорційною відповідному значенню елемента m_{ij} з матриці інтенсивностей M.
3. Якщо залишились непризначені об’єкти (або місцеположення), перейти до кроку 1, інакше – завершити конструювання розв’язку.

Якщо зробити припущення, що кількість можливих локальних оптимумів як мінімум на порядок менша, ніж кількість всіх можливих розв’язків $N!$, то для запобігання передчасній збіжності алгоритму слід уникати повторного впливу одного й того ж локального оптимуму на матрицю інтенсивностей. Тобто доцільною є результуюча матриця, елементи якої формуються як комбінація або перетин різних локальних оптимумів. У зв’язку з цим необхідно вести список локальних оптимумів і виконувати в ньому пошук при спробі додавання нового. З метою підвищення швидкодії доцільним є зберігання не самих векторів, а лише їх хеш-кодів.

Остаточно узагальнений алгоритм передбачає таку послідовність дій.

1. Ініціалізувати матицю інтенсивностей довільним додатним числом та створити порожній список хеш-кодів потенційних розв’язків задачі, а також ініціалізувати значення сумарної вартості розв’язків $C_H = 0.$
2. Сконструювати новий розв’язок задачі за допомогою колеса рулетки.
3. Якщо хеш-код сконструйованого розв’язку міститься в списку хеш-кодів, виконати диверсифікацію, скинувши в початкові відповідні значення матриці інтенсивностей для даного розв’язку, і перейти до кроку 2.
4. Оптимізувати сконструйований розв’язок до локального оптимуму.
5. Якщо хеш-код отриманого на 4-му кроці локального оптимуму вже міститься в списку хеш-кодів, перейти до кроку 2.
6. Додати хеш-код отриманого локального оптимуму до списку хеш-кодів та додати до сумарної вартості розв’язків значення цільової функції поточного розв’язку
$$C_H = C_H + C.$$
7. Помножити відповідні значення елементів m_{ij} матриці інтенсивностей M для призначень $i \to j$ отриманого оптимуму на значення коефіцієнту
$$k = \left(\frac{C_H}{C \times H} \right)^{\alpha},$$

де C_H – сумарна вартість розв'язків, C – значення цільової функції поточного розв'язку, H – довжина списку хеш-кодів, α – коефіцієнт інтенсифікації.

8. Якщо значення цільової функції даного локального оптимуму є меншою за найкращу на даний момент, запам'ятати поточний розв'язок.
9. Якщо виконується умова критерію зупинки, завершити алгоритм і повернути відомий оптимальний розв'язок, інакше – перейти до кроку 2.

В запропонованому алгоритмі значення інтенсивностей m_{ij} матриці M, що відповідають бажаності призначень, фактично змінюються в залежності від того, на скільки кращим або гіршим від середньостатистичного є отриманий розв'язок. Вартість середньостатистичного розв'язку отримується як середня вартість всіх відомих розв'язків C_H / H. Тоді значення $C_H /(C \times H)$ фактично означає, у скільки разів значення поточного розв'язку менше, ніж значення середньостатистичного розв'язку. Оскільки зазвичай після деякої кількості ітерацій вартість кожного наступного розв'язку стає досить близькою до вартості оптимального, то дане відношення буде близьким до одиниці. Для того, щоб підвищити «контрастність» між вартістю поточного розв'язку та середньостатистичного, дане відношення підноситься до степеня $\alpha > 1$. Таким чином, ми маємо автоматичну корекцію бажаності призначень розв'язків як кращих, так і гірших, ніж середньостатистичні.

З викладеного випливає, що алгоритм має лише один вхідний параметр - коефіцієнт інтенсифікації α. За замовчанням можна використовувати значення $\alpha = 1$. Очевидно, що початкові значення елементів матриці інтенсивностей при такому підході не мають значення, а лише мають бути однаковими.

Результати тестування

Показники швидкодії й якості отриманого розв'язку розробленого алгоритму наведені в порівнянні з результатами авторських реалізацій алгоритмів Fast Ant System (FANT) і Simulated Annealing (SA). В якості вхідних даних бралися матриці розмірністю 20, заповнені випадковими числами у діапазоні від 1 до 99 включно. Кількість ітерацій для кожного з алгоритмів була підібрана таким чином, щоб забезпечити приблизно однаковий середній час виконання всіх алгоритмів. У табл. 1 наведені усереднені за 1000 запусків результати роботи алгоритмів.

Таблиця 1 – Порівняння результатів роботи FANT, SA та запропонованого алгоритму

	Час виконання	Кількість ітерацій	Значення цільової функції
SA	585мс	1350	864112
FANT	581 мс	2071	863995
Запропонований алгоритм	568 мс	1000	863226

В результаті проведених експериментів для згенерованих вхідних наборів було визначено оптимальне значення коефіцієнта інтенсифікації $\alpha = 2$.

Висновки

Результати числових експериментів доводять, що запропонований алгоритм у порівнянні з алгоритмами FANT та SA дозволяє отримати більш якісні розв'язки за дещо менший час при довільних вхідних даних. Перспективним є впровадження схеми, за якою всі значення приросту вартості розв'язку внаслідок парних перестановок обчислюються заздалегідь перед знаходженням локального оптимуму, а потім коректуються після кожної перестановки пари елементів [6].

Запропонований алгоритм потребує визначення лише одного вхідного параметру α, який може бути заданий апріорі.

Список літератури

1. Assignment problems and the location of economic activities [Text] / T. C. Koopmans, M. J. Beckman // Econometrica. — 1957. — № 25. — P. 53-76.
2. The quadratic assignment problem [Text] / E. L. Lawler // Management Science. — 1963. — № 9. — P. 586-599.
3. FANT : Fast Ant System [Technical Report] / E. D. Taillard // Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale. — 1998.
4. An improved annealing scheme for the QAP / D. T. Connolly // European Journal of Operational Research. — 1990. — № 46. — P. 93–100.
5. Зорін Ю.М., Подольський С.В. Алгоритм перетину локальних оптимумів для оптимізації задачі про квадратичні призначення // III наук.-техн. конф. «Прикладна математика та комп'ютинг». Тези доповідей. — К.: НТУУ «КПІ», 13-15 квітня 2011. — С. 353 – 356.

6. Зорін Ю.М., Подольський С.В. Табу-пошук для квадратичної задачі про призначення // IV наук.-техн. конф. «Прикладна математика та комп’ютинг». Тези доповідей. — К.: НТУУ «КПІ», 12 квітня 2012. — С. 273 – 276.

Відомості про авторів

Зорін Юрій Михайлович – доцент каф. СПіСКС, факультет прикладної математики, Національний технічний університет Україні “КПІ”.
Подольський Сергій Валентинович – EPAM Systems, Software Engineer.