УДК 519.161

К.т.н., доцент Зорін Ю.М., студент Савкін В.В.

**Національний технічний університету України
«Київський політехнічний інститут»**

МОДИФІКАЦІЯ "МУРАШИНОГО" АЛГОРИТМУ ДЛЯ РОЗВ'ЯЗКУ ЗАДАЧІ КОМІВОЯЖЕРА

Abstract

Yuri Zorin, Assoc. Prof., PhD; Vitaliy Savkin, student
Modification of Ant Colony Optimization metaheuristic

The paper describes a modification of Ant Colony Optimization metaheuristic for Travelling Salesman Problem. The modification idea consists of graph partitioning, search of optimal subroutes with further combining of them. The ways of graph partitioning have been studied and compared. The description of proposed algorithm, its advantages, disadvantages and comparison with existing algorithms for TSP are provided. The prospects for further research and modification are proposed.

Вступ

Задача комівояжера полягає у знаходженні найвигіднішого маршруту, що проходить через вказані міста хоча б по одному разу. В умовах завдання вказуються критерій вигідності маршруту (найкоротший, найдешевший, сукупний критерій тощо) і відповідні матриці відстаней, вартості тощо. Звичайно задано, що маршрут повинен проходити через кожне місто тільки один раз, в такому випадку розв'язок знаходиться серед гамільтонових циклів.

Ця задача відноситься до класу *NP* - повних.

В роботі розглядається симетрична задача комівояжера, тобто матриця відстаней симетрична відносно головної діагоналі.

Постановка задачі

Метою роботи є модифікація алгоритму "мурашиної" колонії для зменшення часу отримання розв'язку, дослідження її особливостей та порівняння отриманого алгоритму з існуючими.

Ідея модифікації

Припустимо, що в графі G розмірністю N існує оптимальний чи близький до оптимального шлях S. Пронумеруємо вершини графу в порядку їх проходження шляхом S, та розділимо їх на K груп по N / K вершин (або, якщо K не є дільником N, на групи по N / K та $N / K + 1$ вершин). Легко бачити, що шлях S проходить крізь вершини i-ої групи по найкоротшому шляху в графі, що складається з вершин цієї групи, починається в першій її вершині й закінчується останньою. Тобто, можна запропонувати такий метод пошуку оптимізованого шляху: розбити граф на підграфи, знайти в них оптимальні шляхи і об’єднати їх в один.

Розбиття на підграфи

Очевидно, що розбивати граф необхідно таким чином, щоб це розбиття якомога більше збігалося з тим, що було описано в попередньому пункті. Але оскільки оптимальний розв’язок невідомий, то як початковий шлях S будемо брати деякий оптимізований шлях, що можна швидко отримати. Розглянуто чотири методи визначення початкового шляху:

а) алгоритм найближчого сусіда [1];

б) алгоритм найближчого сусіда + 2-opt-евристика [2];

в) MMAS (6 ітерацій) [3];

г) MMAS (6 ітерацій) + 2-opt-евристика.

Опис алгоритму

Параметрами алгоритму є граф G, спосіб побудови початкового шляху та кількість підграфів K, яку необхідно отримати.

Алгоритм має такі кроки:

1. Побудова початкового шляху.

2. Розбиття отриманого шляху на K груп (позначатимемо їх, як і відповідні їм шляхи, S_i) способом, описаним в пункті Ідея модифікації.

3. В кожній групі S_i відкидаємо першу $v_{i,1}$ та останню $v_{i,k}$ ($v_{i,k+1}$) вершини, а з тих, що залишились, формуємо граф.

4. Для кожного отриманого підграфу шукаємо за допомогою MMAS оптимізований шлях. Потім до шляху додаємо відкинуті вершини, для чого в ньому знаходимо дві послідовні вершини v_j та v_{j+1} для яких значення $min(W_1, W_2)$ є найменшим. W_1 та W_2 обраховуються наступним чином:

$$W_1 = weight((v_{i,1}, v_j)) + weight((v_{i,k}, v_{j+1})),$$

$$W_2 = weight((v_{i,k}, v_j)) + weight((v_{i,l}, v_{j+1})).$$

Далі, видаляємо зі шляху ребро (v_j , v_{j+1}) і якщо $W_1 < W_2$ то шляху додаємо ребра ($v_{i,1}$, v_j) та ($v_{i,k}$, v_{j+1}), інакше ($v_{i,k}$, v_j) та ($v_{i,l}$, v_{j+1}). Отриманий шлях позначимо P_i. Цей крок надає алгоритму значний потенціал розпаралелювання.

5. Для кожної пари (S_i, P_i) вибираємо шлях, довжина якого менша.

6. Послідовно поєднуємо вибрані на п'ятому кроці шляхи.

7. До отриманого шляху застосовуємо 2-opt-евристику [2].

Складність алгоритму становить $O(N^3)$, але в порівнянні зі звичайним MMAS, зменшується в

$$\frac{O(N^3)}{O\left(K \cdot \left(\frac{N}{K}\right)^3\right)} = O(K^2) \qquad (1)$$

раз.

Дослідження ефективності й порівняння

Для тестів використовувались бенчмарки з TSPLIB[4]: kroA200, lin318, d493. Графи поділялись на 12, 8, 6, 4, 3 частин, для кожного випадку проводилось 25 випробувань, підраховувалося середнє значення довжини шляху та його середньоквадратичне відхилення.

У результаті тестування було встановлено що:

а) при використанні методів, що завершуються процедурою 2-opt, результат не залежить від кількості частин, на яку поділяється граф, оскільки покращується лише 3-5% шляхів у підграфах. Це пояснюється тим, що початковий шлях є локальним оптимумом чи дуже близький до нього;

б) з чотирьох запропонованих методів визначення початкового шляху найкращим за співвідношенням якість / час є алгоритм найближчого сусіда. Найближчий до нього - MMAS + 2-opt дає шлях, на 0,5 – 1% менший за набагато більший час (від 4 разів, причому було помічено, що це відношення зростає з розміром задачі) ;

в) при розбитті на меншу кількість частин результат звичайно кращий, отже потрібно балансувати між швидкістю та якістю, обираючи кількість частин в діапазоні від 8 до 3.

Оскільки модифікація, що розглядається, є спробою оптимізації за часом, його необхідно порівняти з аналогічним, наприклад з алгоритмом найближчого сусіда + 2-opt. Якщо цей алгоритм виконується для однієї вершини, він дає результат на приблизно на 0,5% гірший, ніж запропонований, хоча і виконується значно швидше. Якщо він виконується для всіх вершин, то дає результат якісніше розробленого, але його обчислювальна складність стає $O(N^3)$, і він виконується навіть довше, ніж дана модифікація. Це наслідок зменшення в $O(K^2)$ раз обчислювальної складності (1).

Висновки

1. Розглянута модифікація зменшує обчислювальну складність порівняно зі звичайним MMAS та надає широкі можливості ефективного розпаралелювання, однак якість результату залежить від початкового шляху.
2. Розглянуту модифікацію можна застосовувати у випадках, коли потрібен швидкий розв’язок задачі, але швидкість не настільки критична, щоб використовувати 2-opt-евристику для однієї вершини. При цьому слід відзначити, що 2-opt-евристика завжди дає менш якісний результат.
3. В подальшому планується розробка та дослідження алгоритмів, які можуть забезпечувати краще початкове розбиття графа.

Література

1. http://en.wikipedia.org/wiki/Nearest_neighbour_algorithm
2. *G. A. CROES*. A method for solving traveling salesman problems. // Operations Res. 6. – 1958. – pp. 791 — 812.
3. *T. Stützle et H.H. Hoos*, MAX MIN Ant System. // Future Generation Computer Systems, volume 16. – 2000. – pp. 889 — 914.
4. http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95