ISSN 2311-4738 (print), ISSN 2413-3000 (online)
УДК 519.1 DOI: 10.20998/2413-3000.2023.7.8

В. П. ПРОКОПЕНКОВ

АНАЛІЗ МЕТОДУ ВІДКЛАДЕНИХ РІШЕНЬ ДЛЯ ПОШУКУ ГАМІЛЬТОНОВОГО ЦИКЛУ НА ГРАФІ

Предмет досліджень є рішення задачі пошуку гамільтонова циклу на графі, яка відноситься до NP класу складності. Мета роботи – розробка ефективного поліноміального алгоритму її оптимального вирішення. Дана робота є продовженням попередньої роботи автора, де запропоновано метод відкладених рішень, який використовує перебірну схему допустимих рішень задачі, що пояснюється неможливістю сформулювати умови для прямого знаходження оптимального рішення. Для графа з n вершин розмір простору перебору становить $(n-1)!$. Для великих значень n витрати часу неприпустимо великі і потрібно їх скорочення. У процесі роботи методу відкладених рішень, поки формоване рішення задачі не стане гамільтоновим циклом, воно називається частковим рішенням. Схема, покладена в основу методу відкладених рішень, забезпечує: відмову від повної побудови усіх рішень; одночасне формування множини часткових рішень; відкидання неперспективних рішень; можливість повернення до відкладених часткових рішень при необхідності; виключення втрати оптимального рішення при відкиданні часткових рішень, але, як показано, тільки при виборі правильної оцінки часткових рішень. Спочатку у якості оцінки була використана реальна довжина шляху часткового рішення. Для неповного графа з 20 вершин оптимальне рішення було знайдено за 0,005 хвилини, але на повному графі з 20 вершин час пошуку було порівняно з перебором усіх допустимих рішень задачі. У статті виконано аналіз і показано, що реальна довжина шляху як оцінка логічно обґрунтована і гарантує знаходження оптимального рішення, але не завжди гарантує мінімальні витрати часу на його пошук, оскільки при переборі простору допустимих рішень відпрацьовується схема пошуку в ширину, що тягне побудову практично всіх допустимих рішень. Цим пояснюються різні витрати часу на пошук оптимального рішення для тестових неповного і повного графів – множини допустимих рішень істотно відрізняються. У роботі як альтернатива запропонована інша оцінка – довжина шляху часткового рішення, що вимірюється в дугах графа. Показано, що використання цієї оцінки призводить до перебору рішень в глибину. Ця оцінка скорочує час на пошук рішення, але не гарантує оптимальний результат. Для успішного застосування методу потрібна розробка нової оцінки часткових рішень, яка б поєднувала якості розглянутих оцінок.

Ключові слова: граф, гамільтонів цикл, складність, NP-повнота, простір перебору, перебірний алгоритм, скорочення перебору, метод відкладених рішень, часткове рішення, оцінка часткового рішення.

V. PROKOPENKOV

ANALYSIS OF THE DEFERRED SOLUTIONS METHOD FOR FINDING OF HAMILTONIAN CYCLE ON A GRAPH

The subject of research is the solving of the problem of finding a Hamiltonian cycle on a graph, which belongs to the NP complexity class. The purpose of the work is to develop an effective polynomial algorithm for its optimal solving. This work is a continuation of the author's previous work, where the method of deferred solutions is proposed, which uses a iterating over acceptable solutions scheme, which is explaining by the inability to formulate conditions for directly finding the optimal solution. For a graph of n vertices, the size of the iteration space is $(n-1)!$. For large values n, the time costs are unacceptably large and their reduction is required. During the operation of the deferred solutions method, until the generated solution of the problem becomes a Hamiltonian cycle, it has name – a partial solution. The scheme underlying the deferred solutions method provides: the rejection of the complete construction of all solutions, the simultaneous formation of a set of partial solutions, the discarding of unpromising solutions, the possibility of returning to deferred partial solutions if necessary, the exclusion of the loss of the optimal solution when discarding partial solutions. However, as shown, only when choosing the correct estimate of partial solutions. At first, the real length of the partial solution path was using as an estimate. For an incomplete graph of 20 vertices, the optimal solution found in 0.005 minutes, but on a complete graph of 20 vertices, the search time was commensurate with the search of all possible solutions to the problem. The article analyzes and shows that the real length of the path as an estimate is logically justified and guarantees finding the optimal solution. However, does not always guarantee the minimum time spent searching for it, since when searching through the space of acceptable solutions, a search in width scheme worked out, which entails the construction of almost all acceptable solutions. This explains the different time spent on finding the optimal solution for tests incomplete and complete graphs – the sets of acceptable solutions differ significantly. In this work, as an alternative, another estimate is proposing – the path length of the partial solution, measured in the arcs of the graph. As shown that the use of this estimate leads to a search of solutions in depth. This estimate reduces the time to find a solution, but does not guarantee an optimal result. For the successful application of the method, it is necessary to develop a new estimate of partial solutions that would combine the qualities of the estimates considered.

Keywords: graph, Hamiltonian cycle, complexity, NP-completeness, enumeration space, enumeration algorithm, enumeration reduction, deferred solutions method, partial solution, estimate of a partial solution.

Вступ. Задача пошуку гамільтонова циклу на графі, відома як гра «Навколосвітня подорож» по додекаедру, запропонована В. Гамільтоном [1]. Замкнутий шлях в графі мінімальної довжини, який проходить через усі вершини графа по одному разу, називається в його честь гамільтоновим циклом [2] і є рішенням цієї задачі. Розробка ефективного алгоритму вирішення цієї задачі як і раніше актуальна для практичних і наукових завдань, наприклад [3], а основна проблема його розробки – неможливість сформулювати умови прямого визначення оптимального рішення. Для пошуку рішення задачі існують як евристичні, так і перебірні методи. Перші – задовільні за витратами часу, але не гарантують знаходження оптимального рішення, другі, для його забезпечення реалізують перебір множини допустимих рішень. Повний перебір цієї множини вимагає неполіноміальних витрат часу, так як для графа з n вершин розмір простору можливих рішень задачі оцінюється як $(n-1)!$, тому, розглянута задача відноситься до NP класу складності [4]. Щоб зменшити витрати часу на пошук необхідно скорочувати простір перебору, наприклад, використовуючи метод гілок і границь [5] або [6], але так, щоб не допустити втрати оптимального рішення задачі. В [7] для вирішення проблеми був

© В. П. Прокопенков, 2023

запропонований метод відкладених рішень, покликаний значно знизити витрати часу на пошук, але, як зазначено в [8], не виправдав очікувань.

Аналіз останніх публікацій можна знайти в роботах [7,9-10]. Дослідження цієї задачі тривають, розроблені нові рішення, наприклад, для кубічних графів в [11] було запропоновано рішення складності 1.26^n, а в [12] складність рішення знижена до 1.251^n. Відзначимо, що методи динамічного програмування, які як і перебірні відносяться до точних методів, що гарантують оптимальне рішення, мають складність 2^n [13-14], а всі перебірні – експоненціальну складність. Допустимі рішення в перебірних алгоритмах можна отримувати комбінаторними методами, в яких гамільтонів цикл розглядається як перестановка вершин графа або побудовою, використовуючи алгоритми обходу графа, наприклад, алгоритм Робертса і Флореса [15].

Скорочення витрат на перебір досягається відкиданням наперед неоптимальних рішень, наприклад, використанням методу гілок і границь [5], що має складність $n*log_2 n$, Такі методи здійснюють спрямований перебір допустимих рішень, при якому розглядаються тільки перспективні рішення і відкидаються інші. Для відкидання неперспективних рішень важливо вміти оцінювати рішення, тобто визначити функцію оцінки рішень. Функція оцінки повинна залежати від певних параметрів рішення, а її обчислювальна складність повинна бути мінімальною. Така функція повинна бути точною і надійною, щоб виключити ризик втрати оптимального рішення при відкиданні.

Мета роботи – дослідження, аналіз і доопрацювання методу відкладених рішень для вирішення задачі пошуку гамільтонова циклу на графі з поліноміальними витратами часу.

Постановка задачі. Нехай заданий граф $G = \langle V,E \rangle$, де $V = \left\{ v_i \mid i = \overline{1,n} \right\}$ – це множина вершин, а $E = \left\{ e_{ij} \mid i, j = \overline{1,n}, i \neq j \right\}$ – множина дуг графа. Дуга e_{ij} визначає наявність з'єднання між вершинами v_i та v_j, характеризується відстанню d_{ij}. Нехай задана початкова вершина $v_s \in V$.

Необхідно знайти гамільтонів цикл мінімальної довжини з вершини v_s, тобто кортеж $GC =< v_1, v_2, ..., v_k, ...v_{n-1}, v_n, v_{n+1} >$ з вершин графа *G*, для якого виконуються умови:

1) $v_1 = v_s$;

2) $v_{n+1} = v_s$;

3) для будь-якої пари вершин $v_i, v_j \mid i, j \in \{\overline{1,n}\}$ справедливо: якщо $i \neq j$, то $v_i \neq v_j$;

4) для $\forall v_k \mid k \in \{\overline{2,n}\}$ у графі *G* існують дуги: $e_{k-1,k}$ – з вершини v_{k-1} до вершини v_k та $e_{k,k+1}$ – з вершини v_k до вершини v_{k+1}.

Алгоритм вирішення задачі. Представлений в [1] алгоритм, що реалізує метод відкладених рішень для можливості його аналізу був незначно модифікований. Модифікація дозволила управляти кількістю рішень, що генеруються в процесі обчислень.

Структура даних алгоритму включає:

$\{Suc[i] \mid i = \overline{1,n}\}$ – опис графа *G*, де $Suc[i] = \{ j \mid \exists e_{i,j} \in E \}$ – список послідовників вершини v_i (множина номерів $j \mid j \leq n$, що визначають такі вершини v_j графа *G*, в які можна перейти з вершини v_i);

s – номер початкової вершини;

$PS =< pc, d_{pc} >$ – описує часткове рішення, де: pc – шлях у графі *G*, що визначає часткове рішення *PS* і $d_{pc} =| pc |$ – довжина цього шляху.

PQ – черга відкладених рішень, використовується для зберігання часткових рішень у порядку убування їх оцінки. Підтримує операції: $PQ.Add(PS)$ – додавання часткового рішення *PS* у чергу з урахуванням його оцінки; $PS = PQ.Remove\ (\ )$ – отримання та видалення першого в черзі часткового рішення з черги,

CurPS – поточне оброблюване часткове рішення;

how – кількість рішень, які треба знайти (якщо $how = 0$, необхідно знайти всі рішення);

result_list – список знайдених рішень, спочатку порожній.

Алгоритм.

1. Для кожної вершини j, що належить до $Suc[s]$, списку послідовників початкової вершини s виконати:

1.1 Сформувати початкове часткове рішення $PS_{sj} =< pc, \mid pc \mid >=<< s >, 0 >$.

1.2 Продовжити шлях у частковому рішенні PS_{sj} вершиною v_j: $pc = < s, j >$.

1.3 Скорегувати довжину часткового рішення PS_j: $\mid pc \mid = \mid pc \mid + d_{s,j}$.

1.4 Додати часткове рішення PS_{sj} в чергу відкладених рішень: $PQ.Add(PS_{sj})$.

2. Виконувати цикл поки черга відкладених рішень PQ не пуста:

2.1 Визначити поточне часткове рішення для обробки: $CurPS = PQ.Remove\ (\ )$.

2.2 Визначити номер K останньої вершини шляху часткового рішення $CurPS =< pc, \mid pc \mid >$.

2.3 Для кожної вершини j, що належить до $Suc[K]$ виконати:

2.3.1 Якщо $j \notin pc$ або $j = s$, сформувати часткове рішення $PS_{Kj} =< pc, |\, pc \,|>$:

$pc = pc + j$ – продовжити шлях у частковому рішенні PS_{Kj} вершиною v_j;

$|\, pc \,| = |\, pc \,| + d_{K,j}$ – скорегувати довжину часткового рішення PS_{Kj}.

2.3.2 Якщо $j = s$ і $K + 1 = n$, покласти $GC = PS_{Kj}$, рішення знайдено і додати рішення GC в список *result_list*. Якщо $how \,!= 0$ і кількість знайдених рішень у списку *result_list* рівна *how*, перейти до п. 3.

2.3.3 Додати часткове рішення PS_{Kj} в чергу відкладених рішень: $PQ.Add(PS_{Kj})$.

3. Зупинитися.

У процесі формування поки рішення не стало гамільтоновим циклом воно є частковим рішенням PS.

Пошук починається з часткового рішення, яке включає стартову вершину $PS =< pc, |\, pc \,|>=<< s >, 0 >$. Кожне часткове рішення будується крок за кроком. На кожному кроці з поточного часткового рішення $CurPS$ формуються нові часткові рішення. Для отримання нового часткового рішення до поточного рішення додається вершина, в яку можна перейти з його останньої вершини. При обробці поточного рішення з нього одночасно будується стільки нових часткових рішень, скільки в графі існує варіантів переходу з його останньої вершини в інші вершини графа. Усі нові побудовані часткові рішення, якщо вони можуть бути добудовані до повного рішення, зберігаються в черзі PQ і очікують обробки. Відпрацьоване поточне часткове рішення $CurPS$ видаляється з пам'яті.

Відкидання неперспективного часткового рішення розуміється як відкладання рішення з можливим поверненням до нього знову – як відмова на поточному кроці продовжувати добудовувати його до повного рішення. Це реалізується збереженням його в черзі PQ.

Для можливості реалізувати вибір перспективного часткового рішення для побудови повного рішення, яке стане поточним, кожне часткове рішення характеризується своєю оцінкою. Для виконання наступного кроку алгоритму вибирається те побудоване часткове рішення, яке має найкращу оцінку, тобто перше в черзі PQ.

Виконання схеми зупиняється після побудови необхідної кількості гамільтонових циклів або при відсутності часткових рішень, які можна добудовувати.

Первинне тестування з використанням у якості оцінки реальної довжини часткового рішення показало результативність розробленого методу. Однак, витрати часу на пошук оптимального рішення значно відрізнялися – для неповного графа з 20 вершин склали 0,005 хвилини, а для повного графа з 20 вершин були порівнянні з перебором усіх допустимих рішень задачі.

Можна, звичайно, заперечити – очікування отримати оптимальне рішення в повному графі з такими ж витратами часу як для неповного графа нічим не виправдані, і аргументувати це оцінкою розміру простору можливих рішень для цих графів. Так, дійсно, для використаного неповного графа "додекаедра" ця оцінка дорівнює $3*2^{18}*1 = 786432$, а для повного графа – $19! = 121.645.100.408.832.000$ теоретично можливих рішень. Проте, в це не хотілося вірити, так як розроблений метод був націлений на швидкий пошук рішення задачі і для цього поєднує в собі такі важливі якості, як:

- відмову від повної побудови всіх рішень, що дає значну економію часу;
- одночасне формування всіх рішень похідних від поточного часткового рішення $CurPS$, можна казати, перетворює послідовну обробку в "паралельну" і вносить свій внесок в економію часу;
- відкидання неперспективних рішень, яке також скорочує час пошуку і реалізується відкладанням рішень, забезпечуючи наступну якість;
- можливість повернення до відкладених рішень при необхідності, що в свою чергу забезпечує іншу якість;
- виключення втрати оптимального рішення.

Для перевірки і пояснення результатів первинного тестування в даній роботі проводиться аналіз досліджуваного методу.

Аналіз алгоритму. Розроблений алгоритм не викликає нарікань, цілком обґрунтований і підтвердив свою результативність. Якщо так, то, можливо, у всьому винна обрана оцінка часткових рішень, заснована на реальній довжині часткового рішення.

Для аналізу методу в якості тестового був обраний повний граф на площині розміру $n = 4$, опис якого представлено в табл. 1-2 і рис. 1.

Таблиця 1 – Вершини графа G_4

Вершина		
№	Координати	
	x	*y*
0	300	150
1	149	299
2	0	149
3	150	0

Таблиця. 2 – Дуги графа G_4

ij	0	1	2	3
0	∞	212	300	212
1	212	∞	211	299
2	300	211	∞	211
3	212	299	211	∞

$$\{Suc[0],\ Suc[1],\ Suc[2],\ Suc[3]\}$$
$$Suc[0]=\{1,2,3\}$$
$$Suc[1]=\{0,2,3\}$$
$$Suc[2]=\{0,1,3\}$$
$$Suc[3]=\{0,1,2\}$$

Рис. 1. Опис графа G_4 списками послідовників

Використання реальної довжини як оцінки часткового рішення. Ця оцінка забезпечує розміщення часткових рішень у черзі PQ у порядку зростання довжини шляху часткових рішень, тобто найбільш перспективним вважається те рішення, яке має меншу довжину шляху тобто перше в черзі.

Для графа G_4 був виконаний прогін алгоритму, для якого фіксувалися дії пов'язані з формуванням черги часткових рішень в процесі обчислень (див. табл. 3).

Таблиця 3 – Стан черги PQ для прогону алгоритму для графа G_4 (оцінка – реальна довжина)

Шаг s	Черга відкладених рішень PQ			Шаг s	Черга відкладених рішень PQ		
	індекс j	часткове рішення PS_j	Оцінка		індекс j	часткове рішення PS_j	Оцінка
0		п у с т о		8	0	< 0, 2, 1 >	511
1	0	< 0, 1 >	212		1	< 0, 2, 3 >	511
	1	< 0, 3 >	212		2	< 0, 1, 2, 3 >	634
	2	< 0, 2 >	300		3	< 0, 3, 2, 1 >	634
2	0	< 0, 3 >	212		4	< 0, 1, 3, 2 >	722
	1	< 0, 2 >	300		5	< 0, 3, 1, 2 >	722
	2	< 0, 1, 2 >	423	9	0	< 0, 2, 3 >	511
	3	< 0, 1, 3 >	511		1	< 0, 1, 2, 3 >	634
3	0	< 0, 2 >	300		2	< 0, 3, 2, 1 >	634
	1	< 0, 1, 2 >	423		3	< 0, 1, 3, 2 >	722
	2	< 0, 3, 2 >	423		4	< 0, 3, 1, 2 >	722
	3	< 0, 1, 3 >	511		5	< 0, 2, 1, 3 >	810
	4	< 0, 3, 1 >	511	10	0	< 0, 1, 2, 3 >	634
4	0	< 0, 1, 2 >	423		1	< 0, 3, 2, 1 >	634
	1	< 0, 3, 2 >	423		2	< 0, 1, 3, 2 >	722
	2	< 0, 1, 3 >	511		3	< 0, 3, 1, 2 >	722
	3	< 0, 3, 1 >	511		4	< 0, 2, 1, 3 >	810
	4	< 0, 2, 1 >	511		5	< 0, 2, 3, 1 >	810
	5	< 0, 2, 3 >	511	11	0	< 0, 3, 2, 1 >	634
5	0	< 0, 3, 2 >	423		1	< 0, 1, 3, 2 >	722
	1	< 0, 1, 3 >	511		2	< 0, 3, 1, 2 >	722
	2	< 0, 3, 1 >	511		3	< 0, 2, 1, 3 >	810
	3	< 0, 2, 1 >	511		4	< 0, 2, 3, 1 >	810
	4	< 0, 2, 3 >	511	12	0	< 0, 1, 3, 2 >	722
	5	< 0, 1, 2, 3 >	634		1	< 0, 3, 1, 2 >	722
6	0	< 0, 1, 3 >	511		2	< 0, 2, 1, 3 >	810
	1	< 0, 3, 1 >	511		3	< 0, 2, 3, 1 >	810
	2	< 0, 2, 1 >	511	13	0	< 0, 3, 1, 2 >	722
	3	< 0, 2, 3 >	511		1	< 0, 2, 1, 3 >	810
	4	< 0, 1, 2, 3 >	634		2	< 0, 2, 3, 1 >	810
	5	< 0, 3, 2, 1 >	634	14	0	< 0, 2, 1, 3 >	810
7	0	< 0, 3, 1 >	511		1	< 0, 2, 3, 1 >	810
	1	< 0, 2, 1 >	511	15	0	< 0, 2, 3, 1 >	810
	2	< 0, 2, 3 >	511				
	3	< 0, 1, 2, 3 >	634	16	-1	п у с т о	
	4	< 0, 3, 2, 1 >	634				
	5	< 0, 1, 3, 2 >	722				

На рис. 2 представлено повне дерево рішень для прогону алгоритму для графа G_4 за станом табл. 3.

Корінь дерева відповідає початковій вершині $s=0$. Вузли дерева – це номери вершин, дуги визначають перехід від вершини до вершини при побудові рішень. Листя це крайні вершини в частковому рішенні. Шлях від кореня дерева до листа визначає одне часткове рішення. Якщо вершина, яка є листом, має номер, що збігається з початковою вершиною $s=0$, то шлях від кореня дерева до цього листа описує повне рішення (гамільтонів цикл).

Для дерева числами шрифту меншого розміру вказано номер кроку при виконанні алгоритму, на якому досягається побудова відповідного часткового рішення. Завдяки цим номерам можна простежити

послідовність формування часткових рішень і повних рішень в просторі рішень задачі.

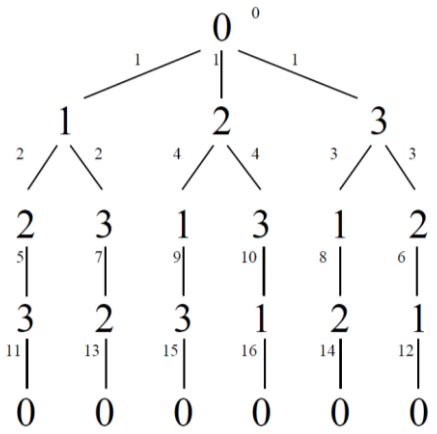

Рис. 2. Дерево рішень для таблиці 3

Для пояснення розглянемо стан на кроці 8, представлений на рис. 3.

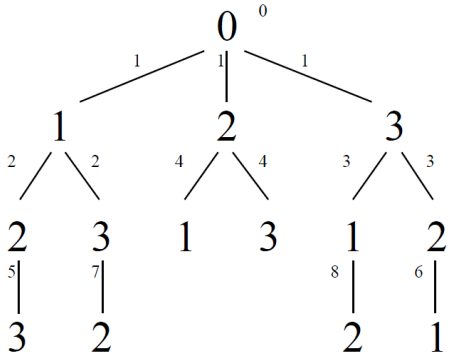

Рис. 3. Дерево рішень для таблиці 3 на кроці 8

З дерева і таблиці 3 видно, що на цьому кроці не побудовано жодного повного рішення. У черзі є відкладені часткові рішення для можливого продовження процесу побудови рішень:

<0,2,1>

<0,2,3>

<0,1,2,3>

<0,3,2,1>

<0,1,3,2>

<0,3,1,2> .

Зауважимо також, що повні рішення в черзі не зберігаються. Наприклад, на кроці 15 у черзі знаходиться часткове рішення <0,2,3,1> , з якого формується повне рішення <0,3,1,2,0> , але воно включається в список знайдених рішень *result_list* .

Дослідження дерева на рис. 2 показує:

1. При використанні у якості оцінки реальної довжини шляху часткового рішення, дерево простору рішень будується в ширину.

2. Порядок формування вузлів дерева (часткових рішень) залежить від довжини часткового рішення.

3. Витрати часу на пошук навіть одного рішення при такій оцінці завжди будуть співмірні з витратами на пошук всіх рішень, оскільки при використанні цієї оцінки результати будуть викидатися після повного формування дерева рішень.

4. Дана оцінка уповільнює процес формування рішень, збільшує витрати часу і пам'яті в процесі обчислень, але забезпечує отримання оптимального рішення.

5. Реальна довжина шляху часткового рішення як оцінка для обробки графа великої розмірності не прийнятна для використання, оскільки, в загальному випадку, дозволяє знайти рішення тільки після повного перебору (побудови) усіх рішень з простору можливих рішень.

Аналіз роботи алгоритму при використанні реальної довжини часткового рішення в якості оцінки показав її основний недолік – вона сприяє формуванню дерева рішень в ширину. При великому розмірі простору можливих рішень задачі це тягне порожні витрати на формування непотрібних рішень. Це також спричиняє додаткові та невиправдані витрати пам'яті на зберігання цих марних рішень у черзі часткових рішень. Таким чином, використання цієї оцінки, по суті, перекреслює всі позитивні якості схеми, покладеної в основу методу відкладених рішень.

Використання довжини в дугах як оцінки часткового рішення. Зрозуміло, що необхідно використовувати іншу оцінку часткових рішень таку, щоб вона на противагу розглянутій оцінці прискорювала процес формування повного рішення. Експериментуючи з можливими оцінками, було обрано оцінку, пов'язану з довжиною часткового рішення в дугах.

Довжину кожного часткового рішення можна вимірювати кількістю дуг графа, якими воно досягається. Наприклад, часткове рішення <0,2,3> реалізується двома дугами графа (0,2) та (2,3), тому довжина цього часткового рішення в дугах дорівнює 2. А відповідна якісна оцінка часткового рішення може бути побудована так: оцінка часткового рішення тим вище, чим більше довжина часткового рішення в дугах.

Таким чином, оцінка, заснована на довжині часткового рішення в дугах, забезпечує, що в черзі відкладених рішень вони розміщуються в порядку убування довжини в дугах, тобто перспективним вважається рішення, яке має велику довжину в дугах.

Для графа G_4 з використанням нової оцінки був виконаний прогін алгоритму, для якого фіксувалися дії пов'язані з формуванням черги часткових рішень в процесі обчислень (див. табл. 4).

Таблиця 4. Стан черги PQ для прогона алгоритму для графа G_4 (оцінка – довжина в дугах)

Шаг s	Черга відкладених рішень PQ			Шаг s	Черга відкладених рішень PQ		
	індекс j	часткове рішення PS_j	Оцінка		індекс j	часткове рішення PS_j	Оцінка
0		п у с т о		7	0	< 0, 2, 1 >	2
1	0	< 0, 1 >	1		1	< 0, 2, 3 >	2
	1	< 0, 2 >	1		2	< 0, 3 >	1
	2	< 0, 3 >	1	8	0	< 0, 2, 1, 3 >	3
2	0	< 0, 1, 2 >	2		1	< 0, 2, 3 >	2
	1	< 0, 1, 3 >	2		2	< 0, 3 >	1
	2	< 0, 2 >	1	9	0	< 0, 2, 3 >	2
	3	< 0, 3 >	1		1	< 0, 3 >	1
3	0	< 0, 1, 2, 3 >	3	10	0	< 0, 2, 3, 1 >	3
	1	< 0, 1, 3 >	2		1	< 0, 3 >	1
	2	< 0, 2 >	1	11	0	< 0, 3 >	1
	3	< 0, 3 >	1	12	0	< 0, 3, 1 >	2
4	0	< 0, 1, 3 >	2		1	< 0, 3, 2 >	2
	1	< 0, 2 >	1	13	0	< 0, 3, 1, 2 >	3
	2	< 0, 3 >	1		1	< 0, 3, 2 >	2
5	0	< 0, 1, 3, 2 >	3	14	0	< 0, 3, 2 >	2
	1	< 0, 2 >	1	15	0	< 0, 3, 2, 1 >	3
	2	< 0, 3 >	1	16	-1	п у с т о	
6	0	< 0, 2 >	1				
	1	< 0, 3 >	1				

На рис. 4 представлено повне дерево рішень для прогону алгоритму для графа G_4 за станом табл. 4.

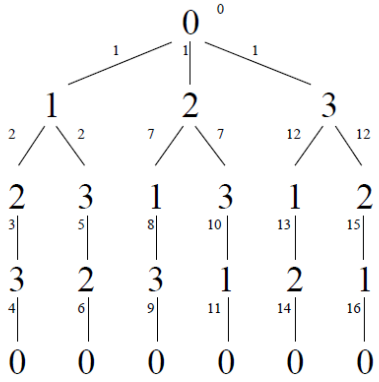

Рис. 4. Дерево рішень для табиці 4

Для пояснення розглянемо стан обчислень на кроці 8, представлений на рис. 5.

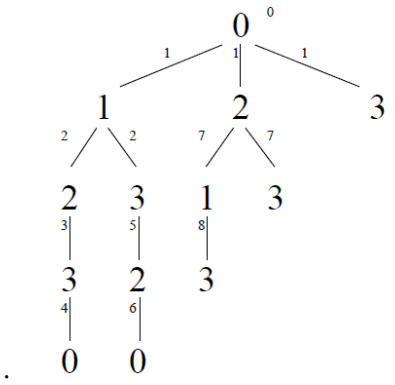

Рис. 5. Дерево рішень для таблиці 4 на кроці 8

З дерева видно, що до цього кроку вже побудовані повні рішення, завершені кроками 4 і 6, і в черзі є відкладені часткові рішення для продовження процесу побудови рішень:

<0,2,1,3>

<0,2,3>

<0,3> .

Вивчення отриманих результатів показує, що при використанні розглянутої оцінки, повне дерево простору рішень будується в глибину.

Якщо порівняти табл.3-4 (стану черги відкладених рішень PQ) в прогонах алгоритму з різними оцінками, однозначно видно, що обчислювальні витрати (часу і пам'яті) при використанні в якості оцінки реальної довжини часткових рішень значно більше ніж при використанні оцінки довжини в дугах.

Вивчення дерева на рис. 4 показує:

1. При використанні в якості оцінки довжини шляху часткового рішення в дугах, дерево простору рішень будується глибину.

2. Порядок формування вузлів дерева (часткових рішень) не залежить від довжини часткового рішення.

3. Витрати часу на пошук одного рішення при такій оцінці визначаються часом необхідним для побудови одного першого рішення.

4. Дана оцінка прискорює процес формування рішень, скорочує витрати часу і пам'яті в процесі обчислень, але не впливає на якість одержуваного рішення.

5. Використання довжини шляху часткового рішення в дугах як оцінки прийнятна для обробки графа великої розмірності, але не гарантує отримання

оптимального рішення, так як знаходиться перше рішення, яке можна побудувати.

Усі знайдені рішення для графа G_4 наведені у табл.5.

Таблиця 5. Усі рішення задачі для графа G_4

№	Повне рішення *GC*	Довжина циклу
0	< 0, 1, 2, 3, 0 >	846
1	< 0, 1, 3, 2, 0 >	1022
2	< 0, 2, 1, 3, 0 >	1022
3	< 0, 2, 3, 1, 0 >	1022
4	< 0, 3, 1, 2, 0 >	1022
5	< 0, 3, 2, 1, 0 >	846

Висновок. У статті виконано аналіз методу відкладених рішень при використанні двох різних оцінок часткових рішень.

Як видно з виконаного аналізу, метод відкладених рішень, по суті, реалізує перебір простору можливих рішень. При пошуку одного рішення (параметр $how = 1$) в залежності від використаної оцінки не всі рішення будуються повністю, а будуються лише ті, які в кожен розглянутий момент вважаються перспективним, тобто неперспективні рішення відкладаються з можливістю в подальшому продовження побудови цих рішень.

При використанні реальної довжини часткових рішень як оцінки, витрати часу на отримання оптимального рішення максимальні і сумірні з перебором всіх допустимих рішень задачі. Реальна довжина шляху як оцінка обґрунтована і гарантує знаходження оптимального рішення, але не завжди за малий час, оскільки при переборі простору допустимих рішень відпрацьовується схема пошуку в ширину, що тягне побудову практично всіх допустимих рішень.

При використанні цієї оцінки істотно різні витрати часу на пошук оптимального рішення для тестових неповного і повного графів пояснюються саме реалізованим пошуком в ширину і різним розміром просторів можливих рішень для цих графів.

Використання довжини часткових рішень в дугах як оцінки призводить до перебору рішень в глибину. При використанні цієї оцінки обчислювальні витрати часу на отримання рішення найменші і рівні часу побудови першого допустимого рішення, оптимальність якого в загальному випадку на жаль не гарантується. Зроблені висновки цілком пояснюють отримані результати первинного тестування в роботі [1].

Для успішного застосування методу потрібна розробка нової оцінки часткових рішень, яка б поєднувала якості розглянутих оцінок – швидкість і гарантію знаходження оптимального рішення.

Список літератури

1. Акимов О. Е. *Дискретная математика. Логика, группы, графы, фракталы*. М., 2005. 656 с.
2. Емеличев В. А., Ковалев М. М., Кравцов М. К. *Многогранники, графы, оптимизация*. М. 1981. 341 с.
3. Гери М., Джонсон Д. *Вычислительные машины и труднорешаемые задачи*. М., 1982. 416 с.
4. Little J. D. C. An Algorithm for the Traveling Salesman Problem. *Operations Research*. 1963. Vol.11. №6. p. 972–989. doi.org/10.1287/opre.11.6.972
5. Дойбер В.А., Косточка А. В., Закс Х. Более короткое доказательство теоремы Дирака о числе ребер в хроматически критических графах. *Дискретный анализ и исследование операций.* Новосибирский гос.ун-т, 1996. с. 28–34.
6. Оре О. *Теория графов*. М., 1980. 336 с.
7. *Гамильтонов граф*: сайт. – URL : https://ru.wikipedia.org/wiki/Гамильтонов_граф. (дата обращения : 4.10.2019).
8. Павленко В. Б. Теоретические аспекты построения гамильтонова цикла. *Теорія оптимальних рішень*. 2011, №10. с. 150–155. сайт . URL : http://dspace.nbuv.gov.ua/bitstream/handle/123456789/46787/22-Pavlenko.pdf?sequence=1 (дата обращения : 4.10.2019).
9. Прокопенков В. Ф., Кожин Ю. Н., Малых О. Н. Определение оптимального кольцевого маршрута, проходящего через заданное множество пунктов на карте. *Innovative technologies and scientific solutions for industries*. 2019. No.1 № 7. С. 102–112. doi.org/10.30837/2522-9818.2019.7.102
10. Харари Ф. *Теория графов*. М., 1973. 300 с.
11. Стивенс Р. *Алгоритмы. Теория и практическое применение*. М., 2016. 544 с.
12. *Муравьиный алгоритм* : сайт. URL : https://ru.wikipedia.org/wiki/Муравьиный_алгоритм. (дата обращения : 4.10.2019).
13. Pol R., Langdon W. B., McPhee N. F. A Field Guide to Genetic Programming. *Genetic Programming and Evolvable Machines*. 2009. Vol. 10 №2. p. 229 – 230. doi.org/10.1007/s10710-008-9073-y.
14. Прокопенков В. Ф. Модификация генетического алгоритма поиска гамильтонова цикла на графе. *Международная научная конференция MicroCAD 2016 : Секция №1 : Информационные и управляющие системы.* 2016. С. 32.
15. Прокопенков В. Ф. О возможности нахождения оптимального решения генетическим алгоритмом. *Международная научная конференция MicroCAD 2017 : Секция №1 : Информационные и управляющие системы.* 2017. С. 37.
16. Мартынов А. В., Курейчик В. М. Гибридный алгоритм решения задачи коммивояжера. *Известия ЮФУ. Технические науки*. – 2015. С.36–44.
17. Eppstein D. The travelling salesman problem for cubic graphs. *Lecture Notes in Computer Science*. 2003. P.307–318. doi.org/10.1007/978-3-540-45078-8_27
18. Iwama K., Nakashima T. An Improved exact algorithm for cubic graph TSP. *Lecture Notes in Computer Science, 2007.* p. 108 – 117. doi.org/10.1007/978-3-540-73545-8_13.
19. Bellman R. Dynamic Programming Treatment of the Travelling Salesman Problem. *Journal of the ACM*. 1962. Vol.9 № 1, p. 61–63. doi.org/10.1145/321105.321111
20. Held M. The travelling-salesman problem an minimum spanning trees. *Operations Research*. 1970. Vol. 18 № 6, p. 1138–1162. doi.org/10.1287/opre.18.6.1138
21. Roberts S. M., Flores B. An engineering approach to the travelling salesman problem. *Management Science*. 1967. Vol. 13 № 3, p. 269–288. doi.org/10.1287/mnsc.13.3.269
22. Прокопенков В. Ф. Новый метод поиска гамильтонова цикла на графе. *Вісник Національного технічного університету «ХПІ». Серія: Стратегічне управління, управління портфелями, програмами та проектами. 2020. № 2, С.43-49.* doi.org/10.20998/2413-3000.2020.2.6
23. Прокопенков В. Ф. Полиномиальный алгоритм поиска гамильтонова цикла на графе. *Вісник Національного технічного університету «ХПІ». Серія: Стратегічне управління, управління портфелями, програмами та проектами. 2021. № 1(3), С.55-65.* doi.org/10.20998/2413-3000.2021.3.8
24. Prokopenkov V.P. Kozhyn Y.N. Deferred solutions scheme for the problem of finding a Hamiltonian cycle solving. *Международная научная конференция MicroCAD 2021 : Секция №1 : Информационные и управляющие системы.* 2021. С. 16.

25. Прокопенков В. Ф. Параллельный алгоритм поиска гамильтонова цикла на графе. *Международная научная конференция MicroCAD 2015 : Секция №1 : Информационные и управляющие системы.* 2015. С. 25.

References (transliterated)

1. Akimov, O. E. (2005) *Diskretnaja matematika. Logika, gruppy, grafy, fraktaly* [Discrete mathematics. Logic, groups, graphs, fractals], M., 656 p.
2. Emelichev, V. A., Kovalev, M. M., Kravcov, M. K. (1981), *Mnogogranniki, grafy, opitimizacija* [Polyhedra, graphs, opitimization], M., 341 p.
3. Hery, M., Dzhonson, D. (1982), *Vychyslytel'nye mashyny y trudnoreshaemye zadachy* [Computational machines and difficult tasks], M., *419 p.*
4. Little, J. D. C. (1963) "An Algorithm for the Traveling Salesman Problem", *Operations Research*, Vol. 11, № 6, p. 972–989. doi.org/10.1287/opre.11.6.972
5. Doiber, V. A., Kostochka, A. V., Sachs, H. Bolee korotkoe dokazatel'stvo teoremy Diraka o chisle reber hromaticheski kriticheskih grafah [A shorter proof of Dirac's theorem on the number of edges of chromatically critical graphs]. *Diskretnyj analiz i issledovanie operacij* [Discrete analysis and operations research] Novosibirsk state University, 1996. - pp. 28–34.
6. Ore, O. (2009), *Teoryya hrafov* [The theory of graphs], M., 354 p.
7. *Gamil'tonov_graf* [Hamilton's graf], Available at : https://ru.wikipedia.org/wiki/Гамильтонов_граф. (last accessed: 04.10.2019)
8. Pavlenko, V. B. (2011) Teoreticheskie aspekty postroenija gamil'tonova cikla [Theoretical aspects of construction of the Hamiltonian cycle], *Teorija optimal'nih rishen'* [Theory of optimal solutions], №10, pp. 150–155. Available at : http://dspace.nbuv.gov.ua/bitstream/handle/123456789/46787/22-Pavlenko.pdf?sequence=1 (last accessed 04.10.2019).
9. Prokopenkov, V. F., Kozhin, Ju., N., Malyh, *O. N.* (2019) Opredelenie optimal'nogo kol'cevogo marshruta, prohodjashhego cherez zadannoe mnozhestvo punktov na karte [Determination of the optimal circular route passing through a given set of points on the map], *Innovative technologies and scientific solutions for industries*, No.1 № 7. pp. 102-112. doi.org/10.30837/2522-9818.2019.7.102
10. Harari, F. (1973), *Teorija grafov* [*Graph theory*], M., 300 p.
11. Stivens, R. (2016), *Algoritmy. Teorija i prakticheskoe primenenie* [Algorithms. Theory and practical application], M. , 544 p.
12. *Murav'inyj algoritm* [Ant Algorithm], Available at : https://ru.wikipedia.org/wiki/Муравьиный_алгоритм. (last accessed: 04.10.2019)
13. Pol, R., Langdon, W. B., McPhee, N. F. (2009), "A Field Guide to Genetic Programming", *Genetic Programming and Evolvable Machines*, Vol. 10, № 2, p. 229 – 230. doi.org/10.1007/s10710-008-9073-y.
14. Prokopenkov, V. F. (2016), Modifikacija geneticheskogo algoritma poiska gamil'tonova cikla na grafe [Modification of a genetic algorithm for finding a Hamiltonian cycle on a graph], *International Scientific Conference MicroCAD 2016: Section No. 1 – Information and Management Systems, p. 32.*
15. Prokopenkov, V. F. (2017), O vozmozhnosti nahozhdenija optimal'nogo reshenija geneticheskim algoritmom [On the possibility of finding the optimal solution by genetic algorithm], *International Scientific Conference MicroCAD 2017: Section No. 1 – Information and Management Systems, p. 37.*
16. Martynov, A. V., Kurejchik, V. M. (2015) Gibridnyj algoritm reshenija zadachi kommivojazhera [Hybrid algorithm for solving the traveling salesman problem], *Izvestija JuFU. Tehnicheskie nauki* [SFU news. Technical science]. p.36–44.
17. Eppstein, D. (2003), "The travelling salesman problem for cubic graphs", *Lecture Notes in Computer Science*, p. 307–318. doi.org/10.1007/978-3-540-45078-8_27
18. Iwama, K., Nakashima, T. (2007), An Improved exact algorithm for cubic graph TSP, *Lecture Notes in Computer Science*, p. 108 – 117. doi.org/10.1007/978-3-540-73545-8_13.
19. Bellman, R. (1962), Dynamic Programming Treatment of the Travelling Salesman Problem, *Journal of the ACM*, Vol.9 № 1, p. 61–63. doi.org/10.1145/321105.321111
20. Held, M. (1970), The travelling-salesman problem an minimum spanning trees, *Operations Research*, Vol. 18 № 6, p. 1138 – 1162. doi.org/10.1287/opre.18.6.1138
21. Roberts, S. M., Flores, B. (1967), An engineering approach to the travelling salesman problem, *Managment Science*, Vol. 13 № 3, p. 269–288. doi.org/10.1287/mnsc.13.3.269
22. Prokopenkov, V. F. (2020), Novyy metod poiska gamil'tonova tsikla na grafe [A new method for searching a Hamilton cycle on a graf], *Visnyk Natsionalnoho tekhnichnoho universytetu «KhPI». Seriia: Stratehichne upravlinnia, upravlinnia portfeliamy, prohramamy ta proektamy.* [Bulletin of the National Technical University "KHPI". Series: Strategic Management, portfolio management, programs and projects], № 2, pp.43-49. doi.org/10.20998/2413-3000.2020.2.6
23. Prokopenkov, V. F. (2021), Polinomial'nyy algoritm poiska gamil'tonova tsikla na grafe [Polynomial algorithm for finding a Hamiltonian cycle on a graph] , *Visnyk Natsionalnoho tekhnichnoho universytetu «KhPI». Seriia: Stratehichne upravlinnia, upravlinnia portfeliamy, prohramamy ta proektamy.* [Bulletin of the National Technical University "KHPI". Series: Strategic Management, portfolio management, programs and projects], № 1(3), pp.55-65. doi.org/10.20998/2413-3000.2021.3.8
24. Prokopenkov V.P. Kozhyn Y.N. (2021), Deferred solutions scheme for the problem of finding a Hamiltonian cycle solving. *International Scientific Conference MicroCAD 2021 : Section No. 1 – Information and Management Systems, p.* 16.
25. Prokopenkov, V. F. (2015), Parallelniy algoritm poiska gamiltonova cikla na grafe [A parallel algorithm for finding a Hamiltonian cycle on a graph], *International Scientific Conference MicroCAD 2015: Section No. 1 – Information and Management Systems, p. 25.*

Надійшла (received) 20.01.2022

Відомості про авторів / Сведения об авторах / About the Authors

Прокопенков Володимир Пилипович (Prokopenkov Vladymyr) – Національний технічний університет "Харківський політехнічний інститут", старший викладач кафедри системний аналіз та інформаційно-аналітичні технології, Харків, Україна; e-mail: prokopenkov.vf@gmail.com; ORCID ID: https://orcid.org/0000-0003-0084-9832.