МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний аерокосмічний університет ім. М. Є. Жуковського
«Харківський авіаційний інститут»

Факультет радіоелектроніки, комп'ютерних систем та інфокомунікацій

Кафедра комп'ютерних систем, мереж і кібербезпеки

Розрахункова робота
з дисципліни «Методи моделювання та оптимізації безпечних комп'ютерних систем»
(назва дисципліни)
на тему: «Вирішення задачі комівояжера»

Виконав: студент 5 курсу групи № 555ім
напряму підготовки (спеціальності)
125 Кібербезпека та захист інформації
(шифр і назва напряму підготовки (спеціальності))
Орлов Станіслав Валерійович
(прізвище й ініціали студента)
Прийняв: д.т.н., професор
Морозова Ольга Ігорівна
(посада, науковий ступінь, прізвище й ініціали)

Національна шкала: ___________
Кількість балів: _____
Оцінка: ECTS _____

Харків – 2023

Постановка задачі:

Номер варіанта	Значення елементів матриці відстаней																			
	r_{12}	r_{13}	r_{14}	r_{15}	r_{21}	r_{23}	r_{24}	r_{25}	r_{31}	r_{32}	r_{34}	r_{35}	r_{41}	r_{42}	r_{43}	r_{45}	r_{51}	r_{52}	r_{53}	r_{54}
22	7	9	7	8	6	6	7	8	9	5	8	7	6	8	7	9	8	8	7	6

Вирішити задачу комівояжера:

1) Методом динамічного програмування
2) Методом гілок і границь
3) Жадібним методом

Вирішення задачі

1) Метод динамічного програмування

$$W_{n-1}(G^*,i) = \min_{j \in G^*} [\, r(i,j) + W_{n-2}(G^* \setminus j,\, j)\,]. \qquad (4)$$

$W_0(G^*,i) = r(i,1)$, $\qquad W_1(G^*,i) = W(j,i) = r(i,j) + r(j,1)$,

Матриця відстаней

i	j				
	1	2	3	4	5
1	-	7	9	7	8
2	6	-	6	7	8
3	9	5	-	8	7
4	6	8	7	-	9
5	8	8	7	6	-

На першому кроці оптимізації визначемо відстані через будь-які дві вершини в початкову:

W({2}, 3) = r(3,2) + r(2,1) = 5 + 6 = 11
W({2}, 4) = r(4,2) + r(2,1) = 8 + 6 = 14
W({2}, 5) = r(5,2) + r(2,1) = 8 + 6 = 14

W({3}, 2) = r(2,3) + r(3,1) = 6 + 9 = 15
W({3}, 4) = r(4,3) + r(3,1) = 7 + 9 = 16
W({3}, 5) = r(5,3) + r(3,1) = 7 + 9 = 16

W({4}, 2) = r(2,4) + r(4,1) = 7 + 6 = 13
W({4}, 3) = r(3,4) + r(4,1) = 8 + 6 = 14
W({4}, 5) = r(5,4) + r(4,1) = 6 + 6 = 12

W({5}, 2) = r(2,5) + r(5,1) = 8 + 8 = 16
W({5}, 3) = r(3,5) + r(5,1) = 7 + 8 = 15
W({5}, 4) = r(4,5) + r(5,1) = 9 + 8 = 17

Отримані значення будуть використовуватись на другому кроці оптимізації

(i = 2)

W({3,4},2) = min[r(2,3) + W({4},3); r(2,4) + W({3},4)] = min[**6+14**; 7+16] = 20
W({3,5},2) = min[r(2,3) + W({5},3); r(2,5) + W({3},5)] = min[**6+15**; 8+16] = 21
W({4,5},2) = min[r(2,4) + W({5},4); r(2,5) + W({4},5)] = min[7+17; **8+12**] = 20

W({2,4},3) = min[r(3,2) + W({4},2); r(3,4) + W({3},4)] = min[**5+13**; 8+16] = 18
W({2,5},3) = min[r(3,2) + W({5},2); r(3,5) + W({2},5)] = min[**5+16**; 7+14] = 21
W({4,5},3) = min[r(3,4) + W({5},4); r(3,5) + W({4},5)] = min[8+17; **7+12**] = 19

W({2,3},4) = min[r(4,2) + W({3},2); r(4,3) + W({2},3)] = min[8+15; **7+11**] = 18
W({2,5},4) = min[r(4,2) + W({5},2); r(4,5) + W({2},5)] = min[8+16; **9+14**] = 23
W({3,5},4) = min[r(4,3) + W({5},3); r(4,5) + W({3},5)] = min[7+15; 9+16] = 22

W({2,3},5) = min[r(5,2) + W({3},2); r(5,3) + W({2},3)] = min[8+15; **7+11**] = 18
W({2,4},5) = min[r(5,2) + W({4},2); r(5,4) + W({2},4)] = min[8+13; **6+14**] = 20
W({3,4},5) = min[r(5,3) + W({4},3); r(5,4) + W({3},4)] = min[**7+14**; 6+16] = 21

Отримані значення будуть використовуватися на третьому (i = 3) – передостанньому – кроці. Комівояжер знаходиться у пункті з номером 2. Треба побувати у вершинах 3,4,5.

W({3,4,5},2) = min[r(2,3)+W({4,5},3); r(2,4)+W({3,5},4); r(2,5)+W({3,4},5)] =
= min[**6+19**; 7 + 22; 8 + 21] = min[**25**;29;29] = 25

Таким чином, встановлено, що при знаходженні комівояжера на третьому кроці в пункті 2 йому слід рухатися в пункт 3.

Далі розрахуємо інші 3 значення, які стосуються того, в якому напрямку потрібно рухатися комівояжеру, якщо він буде знаходитись відповідно в пунктах 3,4,5:

W({2,4,5},3) = min[r(3,2)+W({4,5},2); r(3,4)+W({2,5},4); r(3,5)+W({2,4},5)] =
= min[**5+20**; 8+23; 7+20] = min[25; 31; 27] = 25
W({2,3,5},4)=min[r(4,2)+W({3,5},2); r(4,3)+W({2,5},3); r(4,5)+W({2,3},5)] =
=min[8+21; 7+21; **9+18**] = min[29; 28; 27] = 27

W({2,3,4},5) =min[r(5,2)+W({3,4},2); r(5,3)+W({2,4},3); r(5,4)+W({2,3},4)] =
min[18+20; 7+18; **6+18**] = min[28; 25; 24] = 24

Наприцінці, на останньому кроці, враховуючи результати отримані на попередньому кроці, отримаємо:

W({2,3,4,5},1)= min[**r(1,2) + W({3,4,5},2);** r(1,3) + W({2,4,5},3); r(1,4) +

W({2,3,5},4); **r(1,5) + W({2,3,4},5)]** = min[**7 + 25**; 9 + 25; 7 + 27; **8 + 24**] =
min[**32**; 34; 35; **32**] = 32

Таким чином, мінімальна довжина напрямку дорівнює 32.

2) Метод гілок і границь

i	j				
	1	2	3	4	5
1	-	7	9	7	8
2	6	-	6	7	8
3	9	5	-	8	7
4	6	8	7	-	9
5	8	8	7	6	-

Запишемо мінімальні елементи відповідних рядків у стовпець U_i та віднімемо елементи U_i з відповідних елементів рядка

i	j					U_i
	1	2	3	4	5	
1	-	0	2	0	1	7
2	0	-	0	1	2	6
3	4	0	-	3	2	5
4	0	2	1	-	3	6
5	2	2	1	0	-	6

Внизу отриманої матриці приєднуємо рядок V_j, в якому записуємо мінімальні елементи стовпців. Віднімаємо елемнти V_j із відповідних стовпців матриці і отримаємо:

i	j					U_i
	1	2	3	4	5	
1	-	0	2	0	1	7
2	0	-	0	1	2	6
3	4	0	-	3	2	5
4	0	2	1	-	3	6
5	2	2	1	0	-	6
V_i	0	0	0	0	1	

В результаті обчислень отримаємо матрицю, наведену по рядкам і стовпцям

i	j				
	1	2	3	4	5
1	-	0	2	0	1
2	0	-	0	1	2
3	4	0	-	3	2
4	0	2	1	-	3
5	2	2	1	0	-

Знаходимо константу наведення:

$\gamma = \sum_{i=1}^{5} Ui + \sum_{j=1}^{5} Vj = 30 + 1 = 31$

Знаходимо ступені нульової повністю наведеної матриці.

i	j				
	1	2	3	4	5
1	-	0^{0}	2	0^{0}	0^{1}
2	0^{0}	-	0^{1}	1	1
3	4	0^{1}	-	3	1
4	0^{1}	2	1	-	2
5	2	2	1	0^{1}	-

Ітерація 1

Шукаємо значення мінімальних елементів у кожному рядку та поелементно їх віднімаємо.

i	j				
	1	2	3	4	5
1	-	0^{0}	2^{2}	0^{0}	-
2	0^{0}	-	0^{0}	1^{1}	1^{0}
3	4^{4}	0^{0}	-	3^{3}	1^{0}
4	0^{0}	2^{2}	1^{1}	-	2^{1}
5	2^{2}	2^{2}	1^{1}	0^{0}	-

Внизу отриманої матриці приєднуємо рядок V_j, в якому записуємо мінімальні елементи стовпців. Віднімаємо елемнти V_j із відповідних стовпців матриці і отримаємо:

I	j				U_i
	1	2	3	4	
2	0^0	-	0^1	1	0
3	4^1	0^5	-	3	0
4	0^1	2	1	-	0
5	-	2	1	0^2	0
V_i	0	0	0	0	

Знаходимо суму найменших значень по строкам та стовбцям та таким чином включаємо у гамільтонов контур дугу (1;5). Значення (1;5) = 0

Поточне найменше значення константи наведення = 31 + 0 = 31

Ітерація 2

Розбиваємо множину усіх гамільтонових контурів (0) на дві підмножини, які включають і не включають дугу (3;2). Отримана матриця:

I	j				U_i
	1	2	3	4	
2	0^0	-	0^1	1	0
3	4^1	-	-	3^0	3
4	0^1	2^0	1^1	-	0
5	-	2^0	1^1	0^0	0
V_i	0	2	0	0	

Знаходимо суму найменших значень по 3 рядку та 2 стовбцю:

(!3;2) = 5

Нижня межа підмножини (!3;2) дорівнює 31+5=36

Розбиваємо множину всіх гамільтонових контурів на 2 підмножини, яку включають та не включають дугу (3;2). Запишемо мінімальні значення по кожному рядку та стовпцю у відповідній колонці та рядку.

I	j			U_i
	1	3	4	
2	0^1	-	1	0
4	0^1	1	-	0
5	-	1	0^2	0
V_i	0	1	0	

Знаходимо суму найменших значень по 3 рядку та 2 стовбцю:

(3;2)=1

Нижня межа підмножини (3;2) дорівнює 31+1=32

Порівнюючи нижні межі підмножин бачимо, що 31 < 36, тому включаємо ребро (3;2) до значення константи = 31 + 0 + 1

Ітерація 3

Враховуючи поточний гальмітоновий маршрут, виключаємо дугу (5;4).

Розбиваємо множину всіх гамільтонових контурів на дві підмножини, які включають і не включають дугу (5;4): (54) та (!54)

Шукаємо значення мінімальних елементів у кожному рядку та поелементно їх віднімаємо. Та знаходимо мінімальні значення кожного рядку та стовпця відповідно ($U_{i,}$ V_i).

I	j			U_i
	1	3	4	
2	0^0	-	1^0	0
4	0^0	0^0	-	0
5	-	0^0	-	0
V_i	0	0	1	

Знаходимо суму найменших значень по 5 рядку та 4 стовбцю:

(!54) = 1

Виключаємо дугу (5;4)

	1	3	U_i
2	0^0	-	0
4	0^0	0^0	0
V_i	0	0	

Знаходимо суму найменших значень по строкам та стовпцям та визначаємо поточну дугу маршруту:

(5;4) = 0

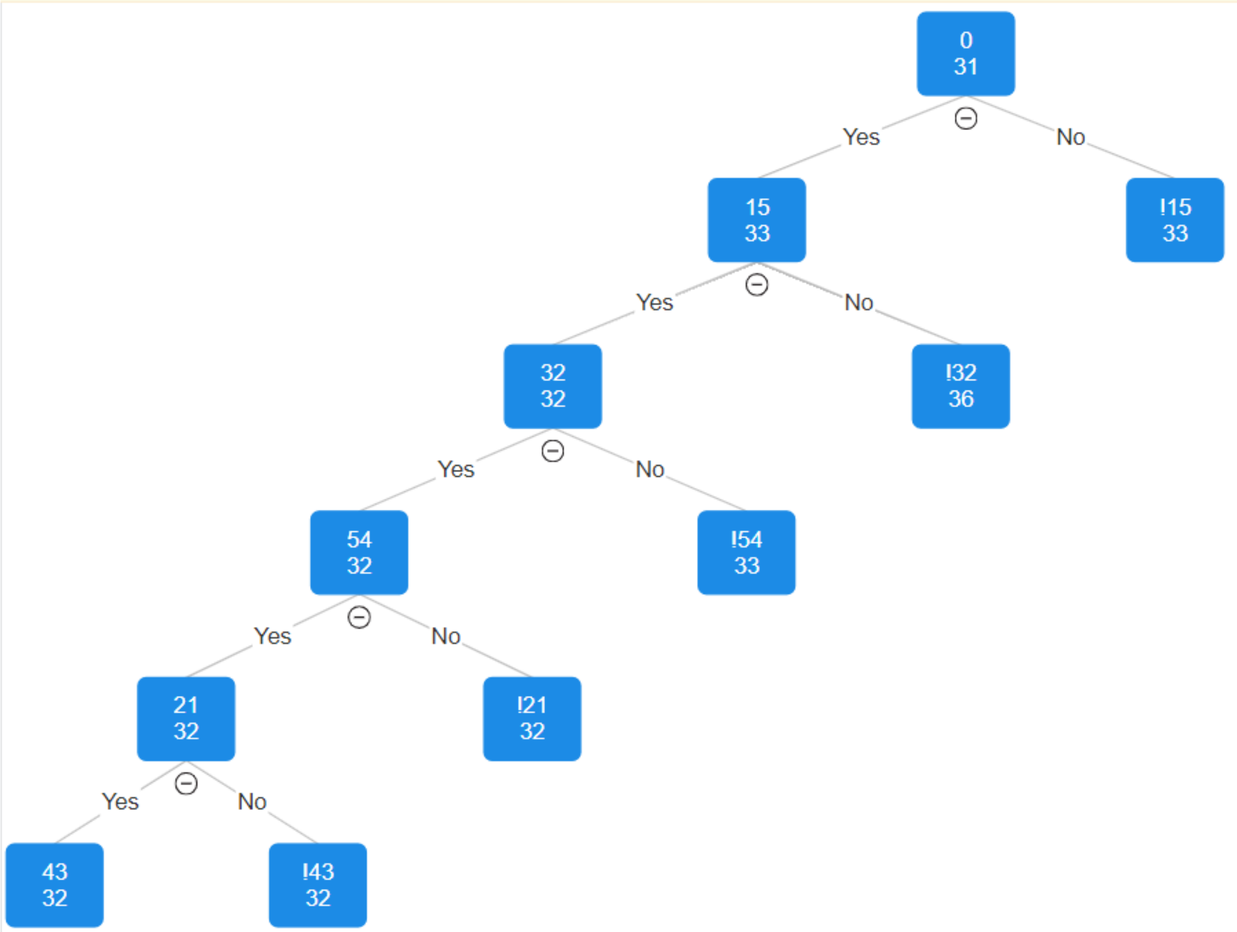
(5;4) < (!54): 31 + 0 < 31 + 1

Включаємо поточну дугу у загальний маршрут

Цикл:

(1,5), (3,2), (5,4), (2,1), (4,3)

Довжина марштрута = 31 + 0 + 1 + 0 + 0 + 0 = 32


0
31
Yes
No
15
33
!15
33
Yes
No
32
32
!32
36
Yes
No
54
32
!54
33
Yes
No
21
32
!21
32
Yes
No
43
32
!43
32

3) Жадібний метод

i	j				
	1	2	3	4	5
1	-	7	9	7	8
2	6	-	6	7	8
3	9	5	-	8	7
4	6	8	7	-	9
5	8	8	7	6	-

Крок 1. В рядку з номером 1 є 2 елементи з мінімальним значенням рівним 7: 2-ий та 4-ий стовпець. Але 4 вершина має також мінімальне значення на 5 кроці, тому шлях комівояжер почне з 2 вершини:

2 ->

Крок 2. Мінімальним елементом 2 рядка є елементи із значенням 6: 1-й та 3-1 стовпець. Але 1 вершина має мінімальна значення на 4 кроці, тож комівояжер продовжить свій шлях через 3 вершину:

2 -> 3 ->

Крок 3. Мінімальним елементом 3 рядка є 2-а вершина зі значенням 5. Через те ща з цієї вершини вояжер почав свій шлях, виберемо наступну оптимальну 5 вершину зі значенням 7.

2 -> 3 -> 5 ->

Крок 4. Мінімальним елементом 3 рядка є 1-а вершина зі значенням 6. Тож комівояжер продовжить свій шлях через 1 вершину:

2 -> 3 -> 5 -> 1 ->

Крок 5. Оскільки маршрут комівояжера пролягає через усі вершини, крім вершини з номером 4, то, очевидно, з вершини з номером 1 цей маршрут пройде у вершину з номером 4, з якої потім у початкову вершину.

2 -> 3 -> 5 -> 1 -> 4 -> 2

Довжина цього маршруту дорівнює 36, що не збігається з оптимальним маршрутом 32. Очевидно, що на першому кроці був не вірно зроблений вибір на користь другого елементу, що не дозволяє використати отпимальне значення на 5 кроці. Тож оптимальний маршрут матиме значення:

Почнему маршрут з 4 вершини та виконуючи попередні кроки 1-5 отримаємо наступний маршрут:

4 -> 1 -> 2 -> 3 -> 5 -> 4

Довжина цього маршруту дорівнює 32 (6+7+6+7+6), що і збігається з оптимальною довжиною маршруту

Висновок.

В результаті можна зробити висновок, що метод динамічного програмування виявився легшим за метод гілок і границь, і однаково точним з ним. Жадібний метод виявився найлегшим, але результат наближений. В усіх методах одинакові довжини напрямків = 32.