КІБЕРНЕТИКА
та КОМП'ЮТЕРНІ
ТЕХНОЛОГІЇ

УДК 519.85 DOI:10.34229/2707-451X.21.3.2

П.І. СТЕЦЮК, Д.І. СОЛОМОН, М.Ю. ГРИГОРАК

ЗАДАЧІ ПРО НАЙКОРОТШІ k-ВЕРШИННІ ЦИКЛИ ТА ШЛЯХИ

Робота присвячена побудові математичних моделей для задач про найкоротші цикли та шляхи, які проходять через задану кількість вершин орієнтованого графа. Для пошуку найкоротшого циклу сформульовано дві задачі – задача змішаного булевого і лінійного програмування та задача дискретного програмування. Зв'язність циклу в першій задачі забезпечується завдяки моделюванню задачі про потік, а в другій задачі забезпечується завдяки використанню обмежень А.Таккера для задачі комівояжера. Досліджено ефективність розв'язання задач за допомогою сучасних версій gurobi та cplex. Для пошуку найкоротшого шляху побудовано задачу змішаного булевого і лінійного програмування. За її допомогою знайдено оптимальні маршрути для відвідування пунктів виноробства Малопольського винного шляху у напрямку Львів – Вроцлав – Львів.

Ключові слова: *орграф, найкоротший шлях, булева змінна, лінійне програмування, гамільтонів цикл, гамільтонів шлях, задача комівояжера, AMPL, gurobi, cplex.*

© П.І. Стецюк, Д.І. Соломон, М.Ю. Григорак, 2021

Вступ. При вирішенні конкретних транспортних та логістичних проблем в умовах, що постійно змінюються, часто виникає необхідність пошуку найкоротшого шляху. «Задачі про найкоротший шлях, в яких на шляхи накладаються деякі обмеження, часто настільки складні, що відповідні алгоритми дозволяють знаходити оптимальні розв'язку лише таких задач, розміри яких (наприклад, число вершин) на кілька порядків менше, ніж у аналогічних задач без обмежень» [1, с. 177]. Задачі на побудову в графі шляхів, які відповідають різним обмеженням розглядаються в [2, 3], а також в [4], де викладено теорію і методи розв'язання задач про оптимальний розподіл однорідних і неоднорідних потоків у мережах.

Спеціальні задачі пошуку найкоротшого шляху при малій кількості лінійних обмежень загального вигляду розглядаються в [5]. Для них алгоритми розв'язання релаксованих задач використовують метод еліпсоїдів у поєднанні з алгоритмом Дейкстри для знаходження найкоротших шляхів у орієнтованому графі. До задач про найкоротші шляхи з обмеженнями відносяться *NP*-важкі задачі – задача комівояжера (найкоротший гамільтонів цикл) і задача пошуку найкоротшого гамільтонового шляху.

У статті розглянуто оптимізаційні задачі знаходження найкоротших циклів і шляхів, що проходять через задану кількість вершин орієнтованого графа. У розділі 1 сформульовано дві задачі математичного програмування для знаходження найкоротшого циклу. Щоб забезпечити зв'язність циклу використано ідею моделювання задачі про потік [6] та використано обмеження, аналогічні побудованим для задачі комівояжера [7]. У розділі 2 встановлено зв'язок формулювань обох задач з розділу 1 із задачею комівояжера і досліджено ефективність їх розв'язання за допомогою сучасних версій gurobi і cplex. У розділі 3 наведено формулювання задачі про найкоротший k-вершинний шлях. З її допомогою знайдено оптимальні маршрути для відвідування пунктів виноробства Малопольського винного шляху в напрямку Львів – Вроцлав – Львів (розділ 4).

1. Два формулювання задачі про найкоротший k-вершинний цикл. Нехай $D_{n,n}$ – повний граф, де n – кількість вершин, а $d_{ij} > 0$ – довжина дуги між вершинами i і j, $i \neq j$. Зафіксуємо вершину s в графі $D_{n,n}$. Цикл, який починається і закінчується у вершині s і проходить через k вершин, де $1 \leq k \leq n-1$ (вершина s у розрахунок не береться), будемо називати k-вершинним циклом в графі $D_{n,n}$. Якщо $k = n-1$, то цей цикл збігається з гамільтоновим циклом, який проходить через всі вершини графа $D_{n,n}$. k-вершинний цикл, якому відповідає найменша сумарна довжина $(k+1)$ дуг, що входять до нього, будемо називати найкоротшим k-вершинним циклом, а його довжину позначимо d_k^*.

Розглянемо два формулювання задач цілочисельного лінійного програмування для знаходження найкоротшого k-вершинного циклу. Щоб забезпечити зв'язність циклу в першому формулюванні використовується ідея моделювання задачі про потік [6], а в другому використовуються обмеження, аналогічні тим, які побудовані С. Міллером, А. Таккером та Р. Земліним для задачі комівояжера [7]. З обох формулювань випливають відомі постановки задач комівояжера [8, 9].

Перше формулювання. Нехай x_{ij} – булева змінна, що дорівнює одиниці, якщо в цикл входить дуга, що з'єднує вершини i та j, і нулю в іншому випадку. Так як $i \neq j$, то кількість таких змінних дорівнює $n(n-1)$. Нехай булева змінна y_i дорівнює одиниці, якщо цикл проходить через вершину i, і нулю в іншому випадку. Кількість таких змінних дорівнює $(n-1)$, тому що вершина s зафіксована. Позначимо z_{ij} невід'ємну змінну, яка задає величину потоку певного продукту від вершини i до вершини j. Цих змінних рівно стільки ж, як і булевих змінних x_{ij}.

Знаходженню найкоротшого k-вершинного циклу відповідає задача змішаного булевого і лінійного програмування [10, 11]:

знайти:

$$d_k^* = \min_{x_{ij}} \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} d_{ij} x_{ij} \tag{1}$$

при обмеженнях

$$\sum_{j=1, j\neq s}^{n} x_{sj} = 1, \quad \sum_{j=1, j\neq i}^{n} x_{ij} = y_i, \quad i = 1,\ldots,n, \quad i \neq s, \tag{2}$$

$$\sum_{j=1, j\neq s}^{n} x_{js} = 1, \quad \sum_{j=1, j\neq i}^{n} x_{ji} = y_i, \quad i = 1,\ldots,n, \quad i \neq s, \tag{3}$$

$$\sum_{i=1, i\neq s}^{n} y_i = k, \tag{4}$$

$$z_{ij} - k x_{ij} \leq 0, \quad i, j = 1,\ldots,n, \quad i \neq j, \tag{5}$$

$$\sum_{j=1, j\neq s}^{n} z_{sj} = k, \quad \sum_{j=1, j\neq s}^{n} z_{js} = 0, \tag{6}$$

$$\sum_{j=1, j\neq i}^{n} z_{ij} - \sum_{j=1, j\neq i}^{n} z_{ji} = -y_i, \quad i = 1,\ldots,n, \quad i \neq s, \tag{7}$$

$$x_{ij} = 0 \vee 1, \quad i, j = 1,...,n, \quad i \neq j, \tag{8}$$

$$y_i = 0 \vee 1, \quad i = 1,\ldots,n, \quad i \neq s, \tag{9}$$

$$z_{ij} \geq 0, \quad i, j = 1,...,n, \quad i \neq j. \tag{10}$$

Теорема 1 [10]. Якщо k – ціле число, яке задовольняє нерівності $1 \leq k \leq n-1$, то обмеження (2) – (10) описують всі можливі k-вершинні цикли, які починаються і закінчуються у вершині s графа $D_{n,n}$.

Мінімізація цільової лінійної функції в (1) відповідає знаходженню k-вершинного циклу мінімальної довжини d_k^*. Обмеження (8) визначають булеві змінні x_{ij}, обмеження (9) – булеві змінні y_i, а обмеження (10) – невід'ємні змінні z_{ij}. Умови (2) – (7) мають такий зміст.

Обмеження (2) описують одноразовий вхід у вершину s і в ті k вершин, для яких $y_i = 1$, а обмеження (3) описують одноразовий вихід з вершини s і тих k вершин, для яких $y_i = 1$. Обмеження (4) задає умову, що рівно для k вершин змінні $y_i = 1$, і визначає набір вершин, через які проходить цикл, що починається у вершині s.

Сімейства обмежень (5), (6) і (7) гарантують зв'язність k-вершинного циклу. Обмеження (5) забезпечують перевезення продукту між вершинами i і j тільки в тому випадку, якщо $x_{ij} = 1$. Обмеження (6), (7) означають, що з вершини s необхідно вивезти k одиниць продукту, залишаючи в кожній з вершин циклу лише одну одиницю продукту. Це дозволяє уникнути підциклів у графі $D_{n,n}$ і забезпечує зв'язність k-вершинного циклу мінімальної довжини d_k^*.

Задача (1) – (10) містить $N_1 = 2n^2 - n - 1$ змінних, з яких $n^2 - 1$ є булеві, а $n(n-1)$ – невід'ємні, і $M_1 = n^2 + 2n + 2$ обмежень, у тому числі $3n + 2$ – лінійні рівності, а $n(n-1)$ – лінійні нерівності.

Зазначимо, що формулювання задачі (1) – (10) може бути застосоване і для неповного графа, якщо його доповнити відсутніми дугами і значення довжин для них встановити рівними сумі довжин всіх дуг неповного графа.

Друге формулювання. Тут булеві змінні x_{ij} і y_i залишаються такими ж, як і в першому формулюванні. Зв'язність k-вершинного циклу в графі $D_{n,n}$ забезпечуватимуть нові цілочисельні змінні u_i, значення яких відповідають номеру кроку, на якому відвідується вершина i. Оскільки вершину s відвідувати не потрібно, то кількість змінних u_i дорівнюватиме $(n-1)$.

Знаходженню найкоротшого k-вершинного циклу відповідає наступна задача цілочисельного лінійного програмування [11]:

знайти:

$$d_k^* = \min_{x_{ij}} \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} d_{ij} x_{ij} \tag{11}$$

при обмеженнях

$$\sum_{j=1, j \neq s}^{n} x_{sj} = 1, \quad \sum_{j=1, j \neq i}^{n} x_{ij} = y_i, \quad i = 1,\ldots,n, \quad i \neq s, \tag{12}$$

$$\sum_{j=1, j \neq s}^{n} x_{js} = 1, \quad \sum_{j=1, j \neq i}^{n} x_{ji} = y_i, \quad i = 1,\ldots,n, \quad i \neq s, \tag{13}$$

$$\sum_{i=1, i\neq s}^{n} y_i = k, \tag{14}$$

$$u_i - u_j + kx_{ij} \leq k-1, \quad i, j = 1,...,n, \quad i \neq s, j \neq s, i \neq j, \tag{15}$$

$$x_{ij} = 0 \vee 1, \quad i, j = 1,...,n, \quad i \neq j, \tag{16}$$

$$y_i = 0 \vee 1, \quad i = 1,\ldots,n, \quad i \neq s, \tag{17}$$

$$u_i - \text{цілі числа}, \quad 1 \leq u_i \leq k, \quad i = 1,\ldots,n, i \neq s. \tag{18}$$

Теорема 2 [11]. Якщо k – ціле число, яке задовольняє нерівностям $1 \leq k \leq n-1$, то обмеження (12) – (18) описують всі можливі k-вершинні цикли, які починаються і закінчуються у вершині s графа $D_{n,n}$.

Мінімізація лінійної функції в (11) відповідає знаходженню k-вершинного циклу мінімальної довжини d_k^*, а обмеження (12) – (14) аналогічні обмеженням (2) – (4) і визначають ті k вершин в графі $D_{n,n}$, через які проходить цикл, що починається у вершині s. Обмеження (15) забезпечують зв'язність k-вершинного циклу та визначають порядок відвідуваності вершин циклу.

Припустимо, що є два цикли. Один з них не проходить через вершину s. Позначимо його $(i_1,\ldots,i_p,i_1)$. З обмежень (15) випливає, що для кожної пари вершин, що йдуть по порядку в цьому циклі, справедливі наступні нерівності:

$$\begin{gathered} u_{i_1} - u_{i_2} + k \leq k-1, \\ u_{i_2} - u_{i_3} + k \leq k-1, \\ \vdots \\ u_{i_p} - u_{i_1} + k \leq k-1. \end{gathered}$$

Склавши ці нерівності, отримуємо $pk \leq p(k-1)$, що неможливо при $p \neq 0$. Отже, для будь-якого підцикла, що не проходить через вершину s, обмеження (15) не виконуються.

Переконаємося, що цикл, що проходить через k вершин, для яких $y_i = 1$, задовольняє обмеженням (15), тобто можна підібрати відповідні значення змінних u_i. Нехай $u_i = p$, якщо вершина i, для якої $y_i = 1$, відвідується на кроці p. Тоді нерівність $u_i - u_j \leq k-1$ виконується при $x_{ij} = 0$, так як $p \leq k$, а змінна $u_j \geq 1$. Якщо $x_{ij} = 1$, то $u_i = p$, а $u_j = p+1$. В цьому випадку $p-(p+1)+k \leq k-1$, звідси, $k-1 \leq k-1$ і обмеження виду (15) виконується як строга рівність.

Задача (11) – (18) містить $N_2 = n^2 + n - 2$ змінних, з яких $n^2 - 1$ є булеві, а $(n-1)$ – цілочисельні, і $M_2 = n^2 - n + 3$ обмежень, у тому числі $2n+1$ – лінійні рівності, а $(n-1)(n-2)$ – лінійні нерівності. Задача (11) – (18) може бути застосована для неповного графа, якщо незв'язані між собою вершини графа з'єднати дугами, приписавши їм значення довжин, що дорівнюють сумі довжин всіх дуг неповного графа.

2. Задача комівояжера і обчислювальні експерименти. Якщо $k = n-1$, то з обмеження (4) усі булеві змінні y_i дорівнюють одиниці і $(n-1)$-вершинний цикл збігається з гамільтоновим циклом. Знаходженню найкоротшого гамільтонового циклу відповідає задача змішаного булевого і лінійного програмування:

знайти:

$$d_k^* = \min_{x_{ij}} \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} d_{ij} x_{ij} \tag{19}$$

при обмеженнях

$$\sum_{j=1, j\neq i}^{n} x_{ij} = 1, \quad \sum_{j=1, j\neq i}^{n} x_{ji} = 1, \quad i = 1,...,n, \tag{20}$$

$$z_{ij} - (n-1)x_{ij} \leq 0, \quad i, j = 1,...,n, \quad i \neq j, \tag{21}$$

$$\sum_{j=1, j\neq s}^{n} z_{sj} = n-1, \quad \sum_{j=1, j\neq s}^{n} z_{js} = 0, \tag{22}$$

$$\sum_{j=1, j\neq i}^{n} z_{ij} - \sum_{j=1, j\neq i}^{n} z_{ji} = -1, \quad i = 1,\ldots,n, \quad i \neq s, \tag{23}$$

$$x_{ij} = 0 \vee 1, \quad z_{ij} \geq 0, \quad i, j = 1,...,n, \quad i \neq j. \tag{24}$$

Задача (19) – (24) збігається з формулюванням задачі комівояжера, наведеної в [8, стор. 46]. Обчислювальні аспекти щодо її розв’язання за допомогою сучасних програм gurobi і cplex можна знайти в [12].

Якщо $k = n-1$, то для знаходження найкоротшого гамільтонового циклу з другого формулювання (11) – (18) випливає задача цілочисельного лінійного програмування:

знайти:

$$d_k^* = \min_{x_{ij}} \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} d_{ij} x_{ij} \tag{25}$$

при обмеженнях

$$\sum_{j=1, j\neq i}^{n} x_{ij} = 1, \quad \sum_{j=1, j\neq i}^{n} x_{ji} = 1, \quad i = 1,...,n, \tag{26}$$

$$u_i - u_j + (n-1)x_{ij} \leq n-2, \quad i, j = 1,...,n, \quad i \neq s, j \neq s, i \neq j, \tag{27}$$

$$x_{ij} = 0 \vee 1, \quad i, j = 1,...,n, \quad i \neq j. \tag{28}$$

$$u_i - \text{цілі числа}, 1 \leq u_i \leq (n-1), \quad i = 1,\ldots,n, \quad i \neq s. \tag{29}$$

Задача (25) – (29) відрізняється від формулювань задачі комівояжера, наведених в [8, стор. 45] і [9, стор. 65], де зв'язність маршруту, що проходить через n міст (вершин), забезпечується на основі умов:

$$u_i - u_j + nx_{ij} \leq n-1, \quad i, j = 1,...,n, \quad i \neq j. \tag{30}$$

Обмеження (30) використовують на одну вершину більше, так як вони не вимагають, щоб цикл проходив через вершину s.

Використання як задачі (1) – (10), так і задачі (11) – (18) розширює можливості застосувань у порівнянні з задачами комівояжера (19) – (24) і (25) – (29). Так, наприклад, задачі (1) – (10) і (11) – (18) можуть бути легко адаптовані для визначення кільцевих маршрутів при плануванні пасажирських і вантажних перевезень [13]. І якщо за допомогою обмежень вигляду (5) – (7) легко врахувати транспортування заданих обсягів вантажу, то за допомогою обмежень вигляду (15) легко врахувати порядок відвідування деяких пунктів призначення.

Для знаходження оптимальних розв’язків задач (1) – (10) і (11) – (18) можна використовувати сучасне програмне забезпечення для вирішення завдань змішаного цілочисельного і лінійного про-

грамування. Найбільш популярними з таких програм є програми **gurobi** [14] і **cplex** [15], їх сучасні версії Gurobi 9.1.1 і CPLEX 20.1.0.0 доступні на NEOS-сервері [16].

Щоб дослідити в якому співвідношенні знаходяться «час роботи зазначених програм» і «розміри задач (графів)» на мові моделювання AMPL [17] було розроблено опис математичних моделей для задач (1) – (10) і (11) – (18) і адаптовано для графів формату * .tsp, де вершини NODES ототожнюються з координатами точок на площині – xcoord {NODES} та ycoord {NODES}, а відстані між кожною парою вершин визначається як округлена до цілого числа евклідова відстань між відповідними цій парі точками площини. AMPL-реалізація задач (1) – (10) і (11) – (18) для графа st70.tsp (має 70 вершин) з бібліотеки TSPLIB наведена у додатку. В додатку також наведено і протоколи розв'язання обох задач для знаходження найкоротших 10-вершинних циклів у графі st70.tsp за допомогою програм Gurobi 9.1.1 і CPLEX 20.1.0.0 з NEOS-сервера. AMPL-реалізацію з додатку можна використовувати для знаходження k -вершинних циклів для довільного графа з бібліотеки TSPLIB, для чого достатньо дані для графа st70.tsp замінити на дані необхідного tsp-графа.

За допомогою програми Gurobi 9.1.1 з NEOS-сервера можна успішно (за кілька хвилин) розв'язувати задачі (1) – (10) і (11) – (18) для 100-вершинних графів. Це підтверджують результати обчислювальних експериментів, наведені в табл. 1, для тестових прикладів задачі комівояжера з бібліотеки TSPLIB з кількістю відвідуваних вершин від 70 до 101. Назви задач (графів) наведені в першій колонці табл. 1, а в другій – вказано кількість вершин відповідного повного графа. В підколонках «N_1», «M_1», «N_2» і «M_2» наведено кількість змінних і кількість обмежень для розв'язуваних задач (1) – (10) і (11) – (18). З даної таблиці видно, що час, витрачений програмою Gurobi 9.1.1 на пошук оптимального маршруту комівояжера, істотно залежить від того, яка форма задачі використовується. Менші з часів виділено жирним шрифтом і вони не перевищують двох хвилин для обраних тестових прикладів.

ТАБЛИЦЯ 1. Результати програми **gurobi** для розв'язання задач (1) – (10) та (11) – (18), де $k = n - 1$

Назва графа (TSPLIB)	n	Задача (1) – (10)			Задача (11) – (18)		
		N_1	M_1	t_{gurobi}	N_2	M_2	t_{gurobi}
st70.tsp	70	9729	5042	**15**	4968	4833	928
eil76.tsp	76	11475	5930	16	5850	5703	**4**
kro100A.tsp	100	19899	10202	**61**	10098	9903	390
kro100E.tsp	100	19899	10202	**116**	10098	9903	870
eil101.tsp	101	20300	10405	38	10300	10103	**27**

Задача знаходження найкоротшого k -вершинного циклу складніша, ніж задача знаходження найкоротшого гамільтонового циклу. Це пояснюється тим, що в ній потрібно визначити підмножину k вершин, для якої буде знайдений гамільтонів підцикл. Це підтверджують наведені в табл. 2 результати роботи програм Gurobi 9.1.1 і CPLEX 20.1.0.0 при розв'язанні задачі (1) – (10) для п'яти відомих графів kro100A ÷ kro100E з бібліотеки TSPLIB (тут $n = 100$, $k = 99$ та $k = 49$), назви тестових задач наведені в першій колонці даної таблиці. Розрахунки проводилися на NEOS-сервері.

З табл. 2 видно, що знаходження 49-вершинних циклів вимагає більших витрат за часом, ніж задача комівояжера (99-вершинний цикл). Так, наприклад, для графа kroC100.tsp вони більше в тридцять разів. Тут витрати часу (в секундах) визначалися для розв'язання задачі (1) – (10) і обчислювалися за допомогою функції "_solve_time" мови AMPL.

ТАБЛИЦЯ 2. Програми **gurobi** та **cplex** для розв'язання задачі (1) – (10), $s=1$, $k=99$ та $k=49$

Задача	d^*_{99}	t_{gurobi}	t_{cplex}	d^*_{49}	t_{gurobi}	t_{cplex}
kroA100.tsp	21282	48	350	9184	184	178
kroB100.tsp	22141	114	441	9096	241	883
kroC100.tsp	20749	50	901	9307	1622	8343
kroD100.tsp	21294	83	368	8929	396	605
kroE100.tsp	22068	106	506	9312	263	762

3. Задача про найкоротший k-вершинний шлях. Нехай $D_{n,n}$ – повний граф, де n – кількість вершин, а $d_{ij}>0$ – довжина дуги між вершинами i і j, $i \neq j$. Будемо розглядати орієнтований граф, як на рис. 1, що включає граф $D_{n,n}$ і дві додаткові вершини a і b, які не співпадають з вершинами графа $D_{n,n}$. Позначимо $d_{ai} \geq 0$ – довжину дуги, що зв'язує вершину a і вершину i графа $D_{n,n}$, а $d_{ib} \geq 0$ – довжину дуги, що зв'язує вершину i графа з вершиною b.

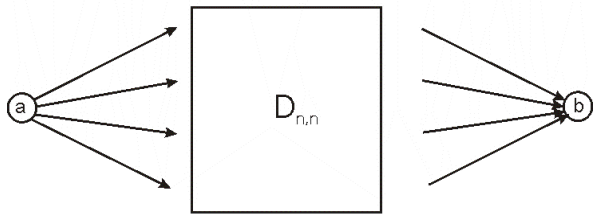

РИС. 1. Орграф, що включає граф $D_{n,n}$ і вершини a, b

Зв'язний шлях з вершини a до вершини b, який проходить через k вершин графа $D_{n,n}$, де $1 \leq k \leq n$, будемо називати k-вершинним шляхом з a до b. Якщо $k=n$, то цей шлях проходить через всі вершини графа $D_{n,n}$. Позначимо $(a, i_1, \ldots, i_k, b)$ – послідовність вершин в k-вершинному шляху, де $i_1, \ldots, i_k$ – вершини графа $D_{n,n}$, що йдуть по порядку. Він містить $(k+1)$ дуг, де одна дуга з'єднує вершину a з вершиною i_1, $(k-1)$ дуг послідовно з'єднують вершини $i_1, \ldots, i_k$, і одна дуга з'єднує вершину i_k з вершиною b. Зв'язний k-вершинний шлях, якому відповідає найменша сумарна довжина $(k+1)$ дуг, що входять до нього, будемо називати найкоротшим k-вершинним шляхом, а його довжину позначимо d^*_{abk}.

Розглянемо формулювання задачі змішаного булевого та лінійного програмування для знаходження найкоротшого k-вершинного шляху. Щоб забезпечити зв'язність шляху, тобто уникнути підциклів у графі $D_{n,n}$, використовується ідея моделювання задачі про потік, аналогічно тому, як це зроблено для задачі (1) – (10).

Нехай x_{ij} – булева змінна, яка дорівнює одиниці, якщо в шлях входить дуга, яка починається у вершині i й закінчується у вершині j, і дорівнює нулю у протилежному випадку. Позначимо x_{ai} і x_{ib} – булеві змінні такого ж типу як x_{ij}, але для дуг з вершини a до вершини i й з вершини i до вершини b. Кількість змінних x_{ai}, x_{ib} і x_{ij} дорівнює $n+n+n(n-1)=n(n+1)$. Нехай y_i – булева змінна, яка дорівнює одиниці, якщо шлях проходить через вершину i, та дорівнює нулю у протилежному випадку. Кількість таких змінних дорівнює n. Нехай невід'ємна змінна z_{ij} задає величину потоку певного продукту від вершини i до вершини j, а невід'ємна змінна z_{ai} задає величину потоку від вершини a до вершини i. Кількість цих змінних дорівнює $n+n(n-1)=n^2$.

Знаходженню найкоротшого k-вершинного шляху з вершини a до вершини b відповідає наступна задача:

знайти

$$d^*_{abk} = \min_{y_i, x_{ij}, z_{ij}} \left\{ \sum_{i=1}^{n} d_{ai} x_{ai} + \sum_{i=1}^{n} d_{ib} x_{ib} + \sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} d_{ij} x_{ij} \right\}, \tag{31}$$

при обмеженнях

$$\sum_{i=1}^{n} x_{ai} = 1, \quad \sum_{i=1}^{n} x_{ib} = 1, \tag{32}$$

$$x_{ai} + \sum_{j=1, j \neq i}^{n} x_{ji} = y_i, \quad \sum_{j=1, j \neq i}^{n} x_{ij} + x_{ib} = y_i, \quad i = 1, \ldots, n, \tag{33}$$

$$\sum_{i=1}^{n} y_i = k, \tag{34}$$

$$z_{ai} - k x_{ai} \leq 0, \quad i = 1, ..., n, \tag{35}$$

$$\sum_{i=1}^{n} z_{ai} = k, \tag{36}$$

$$z_{ij} - (k-1) x_{ij} \leq 0, \quad i, j = 1, ..., n, \quad i \neq j, \tag{37}$$

$$z_{ai} + \sum_{j=1, j \neq i}^{n} z_{ji} - \sum_{j=1, j \neq i}^{n} z_{ij} = y_i, \quad i = 1, \ldots, n, \tag{38}$$

$$x_{ai} = 0 \vee 1, \quad x_{ib} = 0 \vee 1, \quad i = 1, \ldots, n, \quad x_{ij} = 0 \vee 1, \quad i, j = 1, ..., n, \quad i \neq j, \tag{39}$$

$$y_i = 0 \vee 1, \quad i = 1, \ldots, n, \tag{40}$$

$$z_{ai} \geq 0, \quad i = 1, ..., n, \quad z_{ij} \geq 0, \quad i, j = 1, ..., n, \quad i \neq j. \tag{41}$$

Задача (31) – (41) – задача змішаного булевого лінійного програмування. Вона містить $2n(n+1)$ змінних, з яких $n^2 + 2n$ є булеві, а n^2 – невід'ємні, та $n^2 + 3n + 4$ обмежень, у тому числі $3n + 4$ – лінійні рівності, а n^2 – лінійні нерівності.

Теорема 3 [18]. Якщо k – ціле число, яке задовольняє нерівностям $1 \leq k \leq n$, то обмеження (30) – (39) описують всі можливі k-вершинні шляхи з вершини a до вершини b.

Мінімізація цільової функції в (31) відповідає знаходженню найкоротшого (мінімального за довжиною) шляху з вершини a до вершини b, який проходить через k вершин графа $D_{n,n}$. При цьому d^*_{abk} відповідає довжині найкоротшого шляху. Але сам найкоротший шлях може бути неєдиним, і якщо це так, то розв'язок задачі (31) – (41) забезпечує тільки один з можливих найкоротших k-вершинних шляхів з вершини a до вершини b.

4. Оптимальні k-маршрути для Малопольського винного шляху [19]. Обчислювальний експеримент проводився на задачах відвідування заданої кількості пунктів виноробства з 20 найбільш відвідуваних для напрямку Львів – Вроцлав – Львів (Малопольський винний шлях). Їх розташування показано на рис. 2 і характеризується тим, що відстані від Львова і Вроцлава до пунктів виноробства Малопольського винного шляху істотно більші, ніж відстані між пунктами виноробства.

РИС. 2. Пункти виноробства в напрямку Львів – Вроцлав

У табл. 3 наведені назви пунктів виноробства, d_{ai} – відстані до кожного з них від Львова і Вроцлава, а також d_{ib} – відстані від пунктів виноробства до Вроцлава і Львова. Відстані задані в кілометрах і обчислені за допомогою веб-сервісу Google Maps (https://maps.google.com/).

ТАБЛИЦЯ 3. Відстані Львів – пункти виноробства – Вроцлав

i	Назва пунктів виноробства	Львів		Вроцлав	
		d_{ai}	d_{ib}	d_{ai}	d_{ib}
1	Szawapier	386	383	287	286
2	Nad Dobrą Wodą	361	341	267	270
3	Piwnice Antoniego	253	253	389	386
4	Amonit	369	367	274	270
5	Rodziny Steców	272	261	369	369
6	Comte	392	393	244	242
7	Kresy	371	369	277	273
8	Krokoszówka Górska	367	367	273	271
9	Hybridium	365	365	264	263
10	Nad Dworskim Potokiem	289	291	319	318
11	Srebrna Góra	351	347	270	263
12	Zadora	276	275	358	354
13	Zawisza	284	285	367	364
14	Kuźnia	247	265	379	375
15	Demeter	266	268	380	377
16	Uroczysko	284	284	366	363
17	Smykań	338	340	315	315
18	Chodorowa	317	319	387	394
19	Dosłońce	341	343	299	295
20	Gaj	340	339	279	275

У табл. 4 наведені d_{ij} – відстані між пунктами виноробства (тут індекс $i = 1, 2, \ldots, n$ змінюється по горизонталі, а індекс $j = 1, 2, \ldots, n$ змінюється по вертикалі). Відстані d_{ij} задані в кілометрах і також обчислені за допомогою веб-сервісу Google Maps.

ТАБЛИЦЯ 4. Відстані між пунктами виноробства Малопольського винного шляху

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
1	0	83.1	172	89.3	152	86.5	93.3	88.8	32.7	101	71.7	140	136	149	149	148	68.4	127	106	46.7
2	80.3	0	145	12.4	108	40.7	15.7	10.2	49.8	51	12.3	96.3	109	105	106	121	55.6	142	44.1	34.3
3	171	126	0	141	12	177	143	151	149	70.9	136	31	40	29	30.7	33	93	56.6	119	125
4	89.4	12.4	145	0	134	32.9	2.2	6.8	58.9	83.1	23.8	122	118	130	132	130	73.6	151	41.8	43.4
5	151	126	12	132	0	157	136	131	129	51.3	116	16	22.8	0.35	13.5	15.8	80	39.4	102	105
6	86.9	41	177	33	157	0	31	39.8	64.1	107	44	145	142	154	156	153	106	174	54.1	67
7	91.4	14.4	156	2	135	31	0	8.8	60.8	85	25.7	124	120	132	134	132	83.9	153	39.9	45.3
8	81	10.2	151	6.8	131	39.8	9	0	56.6	80.8	21.5	120	116	128	130	128	79.7	149	44.9	41.1
9	32.5	51.1	150	59.1	130	63.4	61.3	56.9	0	79.8	41.5	118	115	127	129	127	68.4	148	84.7	31.6
10	100	56.8	65	83	51.7	106	71.4	68	81.9	0	55.1	35.2	35.4	52	49.1	43.2	46.2	68	59.1	54.4
11	71.6	12.2	136	23	116	47	25.9	21.5	41	65.3	0	104	100	113	114	112	64.2	136	54.4	25.6
12	139	114	21	120	16	145	124	119	117	35.1	104	0	19.8	16.4	21.6	10.2	74	45	85.7	93
13	136	110	40	117	22	141	120	116	114	36.2	90.2	19.8	0	23	14.4	14.4	59	33.3	92	89.5
14	148	122	23	129	10.6	154	133	128	126	48.6	103	18.7	13	0	1.3	15.6	70.2	29.6	104	102
15	149	106	29.4	130	12.2	155	134	130	128	50	104	20.3	14.4	1.6	0	12	71.7	28.7	106	103
16	147	122	33.2	130	15.9	153	132	127	125	49.9	112	10.2	14.4	15.6	18.5	0	71.7	44.4	93.9	101
17	68.3	70.3	92	78.4	80	102	80.5	76.1	69.7	46.2	60.7	91.8	58	70.2	71.7	71.7	0	70.6	81.6	36.5
18	127	155	56.6	151	39.4	174	153	148	146	68.9	133	45	33.2	29.6	28.7	47	70.7	0	125	122
19	106	43.9	117	41.9	101	54	40.2	44.9	94	59	56.2	84.7	90.8	103	105	92.7	80.7	123	0	59.7
20	42	35.1	126	41.5	106	66.4	45.3	40.9	31.7	55.1	25.5	93.8	103	102	104	102	36.6	123	78.3	0

У табл. 5 наведені довжини всіх можливих найкоротших k-вершинних маршрутів для трьох задач відвідування k пунктів виноробства для напрямку Львів – Вроцлав (позначені d^*_{abk}) і для напрямку Вроцлав – Львів (позначені d^*_{bak}). У першій задачі в маршрут могли включатися будь-які з пунктів виноробства, у другій задачі в маршрут обов'язково включався дев'ятий пункт виноробства "Hybridium", а в третій задачі – в маршрут включалися як дев'ятий пункт виноробства, так і дванадцятий пункт виноробства "Zadora".

ТАБЛИЦЯ 5. Довжини найкоротших маршрутів для $k = 1 \div 20$

k	$y_i = 0 \vee 1, i = 1,\ldots,20$		$y_9 = 1$		$y_9 = y_{12} = 1$	
	d^*_{abk}	d^*_{bak}	d^*_{abk}	d^*_{bak}	d^*_{abk}	d^*_{bak}
1	607	608	628	629	–	–
2	601	615.1	633.9	634.6	656	657
3	613.2	628.2	637	641.7	645.7	654
4	614.3	630.4	639.7	653	645.7	653
5	617	635	643.8	659.9	648.6	659.9
6	628.9	642.1	648	664.6	652.8	664.6
7	639.7	655	659.9	671.5	659.9	671.5
8	643.3	660.3	670.7	681.6	670.7	681.6
9	649	663	683.4	695.4	683.4	695.4
10	658.4	671.9	695	703	695	703
11	670.3	682	716.5	728.3	716.5	728.3
12	681.1	695.8	723	738.4	723	738.4
13	693.8	707	740	752.2	740	752.2
14	705.4	728.2	751.6	764.1	751.6	764.1
15	730.2	743.6	766.3	778.7	766.3	778.7
16	758.5	771.3	779	790.6	779	790.6
17	806	818.3	806.2	818.3	806.2	818.3
18	853.7	865.9	853.7	865.9	853.7	865.9
19	901.9	913.5	901.9	913.5	901.9	913.5
20	964.8	976.8	964.8	976.8	964.8	976.8

Розрахунки проводилися за допомогою програми gurobi на NEOS-сервері (http://www.neos-server.org/neos/solvers/). Знаходження найкоротшого маршруту, що проходить через всі 20 пунктів виноробства вимагало кілька секунд роботи програми gurobi. Отже, результати обчислювальних експериментів показали можливість використання запропонованої моделі для вибору оптимальних маршрутів у режимі реального часу.

Висновки. У розділі розроблені нові математичні моделі задач знаходження оптимальних гамільтонових шляхів і циклів для підграфів заданої потужності в орієнтованому графі. Окремим їх випадком є задача пошуку найкоротшого гамільтонового циклу, що рівносильно відомій задачі комівояжера. Задачі (1) – (10) і (11) – (18) можуть бути легко адаптовані для визначення кільцевих маршрутів при плануванні пасажирських і вантажних перевезень. І якщо за допомогою обмежень вигляду (5) – (7) легко врахувати транспортування заданих обсягів вантажу, то за допомогою обмежень вигляду (15) легко врахувати порядок відвідування деяких пунктів призначення.

Задача (31) – (41) дозволяє знаходити як найкоротший гамільтонів шлях у повному графі, так і найкоротші гамільтонові шляхи в підграфах, які отримуються вирізанням з повного графа з n вершинами таких повних підграфів, які містять k вершин, де $1 < k \le n$. Для цього досить у фор-

мулюванні задачі (31) – (41) або покласти всі відстані d_{ai} і d_{ib} рівними нулю, або прибрати внесок у цільову функцію (31) від тих дуг, які зв'язують вершини a і b з вершинами повного графа.

Розроблені математичні моделі задач знаходження найкоротших k-вершинних шляхів і циклів можуть бути використані при проектуванні і компонуванні технічних об'єктів, оптимізації транспортування товарів, аналізі та прогнозуванні економічних процесів, визначенні оптимальних маршрутів при плануванні пасажирських і вантажних перевезень. Розроблене програмне забезпечення на мові моделювання AMPL може бути використано для розв'язання інших класів прикладних задач оптимізації, наприклад, оптимальної організації процесу управління множиною транзакцій та запитів при їх реалізації у мережевих базах даних [20].

Додаток. AMPL-реалізація задач (1) – (10) та (11) – (18) для графів з бібліотеки TSPLIB

AMPL-код задач (1–10) та (11) – (18) для графа st70.tsp

```
param n>=5; #кількість вершин в графі
param s>=1<=n; #фіксована вершина (початок циклу)
param k>=1<=(n-1) default n-1; #кількість вершин циклу

set NODES := 1..n; #вершини та дуги повного графа
set ARCS within (NODES cross NODES) := {i in NODES, j in NODES: i!=j};
param xcoord{NODES}; param ycoord{NODES}; #координати вершин
param d{ARCS} >= 0; #довжини дуг (евклідова відстань, округлена)

#Невідомі (змінні)
var x{ARCS} binary; var y{NODES} binary; #булеві
var z{ARCS} >=0; #неперервні
var u{NODES} >= 1 <= k integer; #цілочисельні

#Мінімізувати довжину циклу:
minimize dk_min: sum{(i,j) in ARCS} d[i,j] * x[i,j];
subject to #за обмежень
con2 {i in NODES}:  #одноразовий вхід в вершину i
sum{j in NODES: (i,j) in ARCS } x[i,j] = (if i == s then 1 else y[i]);
con3 {i in NODES}:  #одноразовий вихід з вершини i
sum{j in NODES: (j,i) in ARCS } x[j,i] = (if i == s then 1 else y[i]);
con4: sum{i in NODES: i!=s} y[i] = k; #цикл містить k вершин
con5 {(i,j) in ARCS}:  z[i,j] - k*x[i,j] <= 0 ;     #обмеження (5)-(7)
con6_1: sum{j in NODES: (s,j) in ARCS } z[s,j] = k; #відповідають за
con6_2: sum{j in NODES: (j,s) in ARCS } z[j,s] = 0; #зв'язність циклу
con7 {i in NODES: i!=s}:                            #у задачі (1)-(10)
          sum{j in NODES: (i,j) in ARCS } z[i,j]
        - sum{j in NODES: (j,i) in ARCS } z[j,i] = - y[i];
con15{(i,j) in ARCS: i!=s and j!=s}: # зв'язність циклу у задачі (11)-(18)
           u[i] - u[j] + k*x[i,j] <= k-1;

problem problem1: dk_min,x,y,z,con2,con3,con4,        #задача (1)-(10)
                  con5, con6_1, con6_2, con7;
problem problem2: dk_min,x,y,u,con2,con3,con4,con15;  #задача (11)-(18)

printf "st70.tsp data:\n" ;  #дані для графа st70.tsp

data;

param n:= 70;
```

```
param s:= 1;
param k:= 10;

param: xcoord :=
 1 64   2 80   3 69   4 72   5 48   6 58   7 81   8 79   9 30  10 42
11  7  12 29  13 78  14 64  15 95  16 57  17 40  18 68  19 92  20 62
21 28  22 76  23 67  24 93  25  6  26 87  27 30  28 77  29 78  30 55
31 82  32 73  33 20  34 27  35 95  36 67  37 48  38 75  39  8  40 20
41 54  42 63  43 44  44 52  45 12  46 25  47 58  48  5  49 90  50 41
51 25  52 37  53 56  54 10  55 98  56 16  57 89  58 48  59 81  60 29
61 17  62  5  63 79  64  9  65 17  66 74  67 10  68 48  69 83  70 84;
param: ycoord :=
 1 96   2 39   3 23   4  42   5 67   6 43   7 34   8 17   9 23  10 67
11 76  12 51  13 92  14   8  15 57  16 91  17 35  18 40  19 34  20  1
21 43  22 73  23 88  24  54  25  8  26 18  27  9  28 13  29 94  30  3
31 88  32 28  33 55  34  43  35 86  36 99  37 83  38 81  39 19  40 18
41 38  42 36  43 33  44  18  45 13  46  5  47 85  48 67  49  9  50 76
51 76  52 64  53 63  54  55  55  7  56 74  57 60  58 82  59 76  60 60
61 22  62 45  63 70  64 100  65 82  66 67  67 68  68 19  69 86  70 94;

for{(i,j) in ARCS} { #обчислюємо довжини дуг
  let d[i,j] := round(sqrt((xcoord[i]-xcoord[j])*(xcoord[i]-xcoord[j])
             +(ycoord[i]-ycoord[j])*(ycoord[i]-ycoord[j])));
}

#display d;
display n,k,s;

#option solver gurobi;
printf "problem1:\n" ;
solve problem1;
display _solve_time;

display dk_min;
printf "Display x:\n" ;
for{i in NODES} {
  for{j in NODES: (i,j) in ARCS and x[i,j] >= 0.02} {
     printf "%d -> %d [%0.2f]\n", i, j, x[i,j];
  }
}
printf "problem2:\n" ;
solve problem2;
display _solve_time;

display dk_min;
printf "Display x:\n" ;
for{i in NODES} {
  for{j in NODES: (i,j) in ARCS and x[i,j] >= 0.02} {
     printf "%d -> %d [%0.2f]\n", i, j, x[i,j];
  }
}
printf "Display u:\n" ;
for{i in NODES}{
   if y[i] > 0 then printf "%d\t %d\n" ,i,u[i];

}
```

Протокол роботи програми Gurobi 9.1.1

```
************************************************************

   NEOS Server Version 6.0
   Job#     : 10733941
   Password : kdInwMVG
   Solver   : milp:Gurobi:AMPL
   Start    : 2021-07-17 03:19:38
   End      : 2021-07-17 03:19:43
   Host     : prod-sub-1.neos-server.org

   Disclaimer:

   This information is provided without any express or
   implied warranty. In particular, there is no warranty
   of any kind concerning the fitness of this
   information for any particular purpose.

   Announcements:

   We will require an email address  with all job submissions
   on the NEOS Server starting on January 8, 2021.
   Please see:
     https://neos-guide.org/content/FAQ#email

************************************************************
You are using the solver gurobi_ampl.

Checking ampl.mod for gurobi_options...
Executing AMPL.
processing data.
processing commands.
Executing on prod-exec-6.neos-server.org
st70.tsp data:
n = 70
k = 10
s = 1

problem1:

Presolve eliminates 70 constraints and 70 variables.
Adjusted problem:
9660 variables:
        4899 binary variables
        4761 linear variables
4972 constraints, all linear; 28980 nonzeros
        211 equality constraints
        4761 inequality constraints
1 linear objective; 4830 nonzeros.

Gurobi 9.1.1: threads=4
Gurobi 9.1.1: optimal solution; objective 74
8076 simplex iterations
1 branch-and-cut nodes
plus 1006 simplex iterations for intbasis
_solve_time = 3.30517
```

```
dk_min = 74

Display x:
1 -> 23 [1.00]
13 -> 29 [1.00]
22 -> 63 [1.00]
23 -> 38 [1.00]
29 -> 36 [1.00]
31 -> 13 [1.00]
36 -> 1 [1.00]
38 -> 22 [1.00]
59 -> 69 [1.00]
63 -> 59 [1.00]
69 -> 31 [1.00]

problem2:

Presolve eliminates 0 constraints and 4832 variables.
Adjusted problem:
4968 variables:
        4899 binary variables
        69 integer variables
4833 constraints, all linear; 23943 nonzeros
        141 equality constraints
        4692 inequality constraints
1 linear objective; 4830 nonzeros.

Gurobi 9.1.1: threads=4
Gurobi 9.1.1: optimal solution; objective 74
58947 simplex iterations
5808 branch-and-cut nodes
_solve_time = 8.50752

dk_min = 74

Display x:
1 -> 23 [1.00]
13 -> 29 [1.00]
22 -> 63 [1.00]
23 -> 38 [1.00]
29 -> 36 [1.00]
31 -> 13 [1.00]
36 -> 1 [1.00]
38 -> 22 [1.00]
59 -> 69 [1.00]
63 -> 59 [1.00]
69 -> 31 [1.00]

Display u:
13       8
22       3
23       1
29       9
31       7
36       10
38       2
59       5
63       4
69       6
```

Протокол роботи програми CPLEX 20.1.0.0

```
************************************************************

   NEOS Server Version 6.0
   Job#     : 10733945
   Password : KTmXpltz
   Solver   : milp:CPLEX:AMPL
   Start    : 2021-07-17 03:22:35
   End      : 2021-07-17 03:22:43
   Host     : prod-sub-1.neos-server.org

************************************************************
You are using the solver cplexamp.

Checking ampl.mod for cplex_options...
Executing AMPL.
processing data.
processing commands.
Executing on prod-exec-6.neos-server.org
st70.tsp data:
n = 70
k = 10
s = 1

problem1:

Presolve eliminates 70 constraints and 70 variables.
Adjusted problem:
9660 variables:
        4899 binary variables
        4761 linear variables
4972 constraints, all linear; 28980 nonzeros
        211 equality constraints
        4761 inequality constraints
1 linear objective; 4830 nonzeros.

CPLEX 20.1.0.0: threads=4
CPLEX 20.1.0.0: optimal integer solution; objective 74
2397 MIP simplex iterations
0 branch-and-bound nodes
_solve_time = 5.91247

dk_min = 74

Display x:
1 -> 36 [1.00]
13 -> 31 [1.00]
22 -> 38 [1.00]
23 -> 1 [1.00]
29 -> 13 [1.00]
31 -> 69 [1.00]
36 -> 29 [1.00]
38 -> 23 [1.00]
59 -> 63 [1.00]
63 -> 22 [1.00]
69 -> 59 [1.00]
```

```
problem2:

Presolve eliminates 0 constraints and 4832 variables.
Adjusted problem:
4968 variables:
        4899 binary variables
        69 integer variables
4833 constraints, all linear; 23943 nonzeros
        141 equality constraints
        4692 inequality constraints
1 linear objective; 4830 nonzeros.

CPLEX 20.1.0.0: threads=4
CPLEX 20.1.0.0: optimal integer solution; objective 74
62937 MIP simplex iterations
5215 branch-and-bound nodes
_solve_time = 14.4143

dk_min = 74

Display x:
1 -> 36 [1.00]
13 -> 31 [1.00]
22 -> 38 [1.00]
23 -> 1 [1.00]
29 -> 13 [1.00]
31 -> 69 [1.00]
36 -> 29 [1.00]
38 -> 23 [1.00]
59 -> 63 [1.00]
63 -> 22 [1.00]
69 -> 59 [1.00]

Display u:
13       3
22       8
23       10
29       2
31       4
36       1
38       9
59       6
63       7
69       5
```

Список літератури

1. Кристофидес Н. Теория графов. Алгоритмический подход. М: Мир, 1978. 432 с.
2. Ермольев Ю.М. Кратчайшие допустимые пути. *Кибернетика*. 1966. № 3. С. 88–95.
3. Ермольев Ю.М., Мельник И.М. Кратчайшие допустимые пути. II. *Кибернетика*. 1967. № 1. С. 63–71.
4. Ермольев Ю.М., Мельник И.М. Экстремальные задачи на графах. Киев: Наукова думка, 1968. 176 с.
5. Стецюк П.И., Петрухин В.А., Бугров Н.В., Хрипко К.Ю. Метод эллипсоидов и условно-оптимальный маршрут. Міжнародна наукова конференція *"Сучасна інформатика: проблеми, досягнення та перспективи розвитку"*, присвячена 90-річчю від дня народження академіка В.М. Глушкова (12–13 вересня 2013 року). Київ: Інститут кібернетики імені В.М. Глушкова НАН України. 2013. С. 111–113.

6. Gavish B., Graves S.C. The travelling salesman problem and related problems. Working Paper OR-078-78, 1978. Operations Research Center. MIT. Cambridge. MA.
7. Miller C.E., Tucker A.W., Zemlin R.A. Integer programming formulation of travelling salesman problem. *J. ACM*. 1960. 3. P. 326–329.
8. Алексеева Е.В. Построение математических моделей целочисленного линейного программирования. Примеры и задачи: Учеб. пособие / Новосиб. гос. ун-т. Новосибирск, 2012. 131 с.
9. Гамецкий А.Ф., Соломон Д.И. Исследование операций. Том II. Кишинэу, Эврика, 2008. 592 с.
10. Стецюк П.И. Формулировки задач для кратчайшего *k*-вершинного пути и кратчайшего *k*-вершинного цикла в полном графе. *Кибернетика и системный анализ*. 2016. № 1. С. 78–82.
11. Стецюк П.И. Две задачи о кратчайшем *k*-вершинном цикле. Матеріали XIII Міжнародної науково-практичної конференції *«Математичне та програмне забезпечення інтелектуальних систем (MPZIS–2015)»* (18–20 листопада 2015. Дніпропетровськ). С. 215–221.
12. Стецюк П.И., Соломон Д.И. Вычислительные аспекты задачи коммивояжера. Тези доповідей VIII Міжнародної науково-технічної конференції *«Інформаційно-комп'ютерні технології 2016»* (22–23 квітня 2016 року). Житомир: ЖДТУ, 2016. С. 91–92.
13. Solomon Dumitru. Transporturi rutiere de mărfuri si pasageri. Cartea I. *Organizarea transporturilor rutiere de mărfuri*. (Proiect de an) Ch.: Evrica, 2014. 170 p.
14. Gurobi Optimization, Inc. Gurobi Optimizer Reference Manual. 2014. http://www.gurobi.com/
15. CPLEX Optimizer. High-performance mathematical programming solver for linear programming, mixed-integer programming and quadratic programming. https://www.ibm.com/analytics/cplex-optimizer
16. NEOS Solver. https://neos-server.org/neos/solvers/
17. Fourer R., Gay D., Kernighan B. AMPL, A Modeling Language for Mathematical Programming. Belmont: Duxburry Press, 2003. 517 p.
18. Стецюк П.И., Лефтеров А.В., Федосеев А.И. Кратчайший *k*-вершинный путь. *Компьютерная математика*. 2015. № 2. С. 3–11.
19. Стецюк П.И., Лефтеров А.В., Лиховид А.П., Федосеев А.И. Оптимизационный сервис для выбора винных маршрутов. *Математическое моделирование, оптимизация и информационные технологии:* материалы V Международной научной конференции (Кишинэу, 22–25 марта 2016 г.). Кишинэу: Эврика, 2016. Т. II. С. 337–344.
20. Андрианова Е.Г., Раев В.К., Фильгус Д.И. Определение кратчайших гамильтоновых путей в произвольных графах распределенных баз данных. *Российский технологический журнал*. 2019. Т. 7, № 4. С. 7–20. https://doi.org/10.32362/2500-316X-2019-7-4-7-20

Одержано 23.07.2021

Стецюк Петро Іванович,
доктор фізико-математичних наук,
завідувач відділу методів негладкої оптимізації
Інституту кібернетики імені В.М. Глушкова НАН України, Київ,
stetsyukp@gmail.com
https://orcid.org/0000-0003-4036-2543

Соломон Дмитро Ілліч,
доктор технічних наук, головний науковий співробітник
лабораторії математичного моделювання
Інституту математики та інформатики Академії наук Молдови, Кишинеу,
dumitru.solomon@math.md

Григорак Марія Юріївна,
доктор економічних наук, завідувачка кафедри логістики Національного авіаційного університету, Київ.
m_grigorak@ukr.net
https://orcid.org/0000-0002-5023-8602

MSC 90C15, 49M27

Petro Stetsyuk [1 *], Dumitru Solomon [2], Maria Grygorak [3]

Problems on Shortest *k*-Node Cycles and Paths

[1] *V.M. Glushkov Institute of Cybernetics of the NAS of Ukraine, Kyiv*
[2] *Institute of Mathematics and Informatics of the Academy of Sciences of Moldova, Chisinau*
[3] *National Aviation University, Kyiv*
[*] *Correspondence: stetsyukp@gmail.com*

The paper is devoted to the construction of mathematical models for problems on the shortest cycles and paths, that pass through a given number of nodes of a directed graph. Such cycles and paths are called *k*-node, where $1<k<n$, n is the number of nodes in the graph.

Section 1 formulates two problems for finding the shortest *k*-node cycle – a mixed Boolean and linear programming problem and a discrete programming problem. Both problems include constraints from the classical assignment problem, describing a one-time entry into a node and a one-time exit from a node for those nodes through which the cycle passes. The cycle connectivity in the first problem is ensured by modeling the flow problem, and in the second problem, it is ensured by using the A. Tucker constraints for the travelling salesman problem.

Section 2 establishes a connection between the formulations of both problems from Section 1 and the travelling salesman problem and investigates the efficiency of their solution using modern versions of gurobi and cplex programs and the AMPL modeling language.

Section 3 contains the formulation of the shortest *k*-node path problem, which is represented by a mixed Boolean and linear programming problem. With its help, the optimal routes were found for visiting the wine-making points of the Malopolskie Wine Route in the direction Lviv-Wroclaw-Lviv (Section 4). Here a map for the 20 most visited wine-making points of the Malopolskie Wine Route and a table of the distances between them and the distances from them to Lviv and Wroclaw, calculated using the Google Maps web service, are presented.

The developed mathematical models of the problems of finding the shortest *k*-node paths and cycles and the developed software in the AMPL modeling language can be used for the design and arrangement of technical objects, optimization of the transportation of products, analysis and forecasting of economic processes, determination of optimal routes when planning passenger and freight traffic, optimal organization of the process of managing a set of transactions and queries during their implementation in network databases and other classes of applied optimization problems.

Keywords: digraph, shortest path, Boolean variable, linear programming, Hamiltonian cycle, Hamiltonian path, travelling salesman problem, AMPL, gurobi, cplex.