Міністерство освіти і науки України
Національний університет «Києво-Могилянська академія»
Факультет інформатики
Кафедра математики

Курсова робота

освітній ступінь – бакалавр

на тему: **«ГРАЦІОЗНА РОЗМІТКА ГРАФІВ»**

Виконав: студент 3-го року навчання
освітньої програми «Прикладна математика»,
спеціальності 113 Прикладна математика

Аверкін Олександр Сергійович

Керівник: Тимошкевич Л. М.
кандидат фіз.-мат. наук, ст. викладач

Курсова робота захищена
з оцінкою _____

Секретар ЕК_____
(підпис)

«_____» _____ 20___р.

Київ - 2023

Графік підготовки курсової роботи до захисту

Графік узгоджено «_____» _____ 2023р.

№ з/п	Перелік робіт	Термін виконання етапу	Підпис наукового керівника	Дата ознайомлення наукового керівника	Примітка
1.	Отримання теми кваліфікаційної роботи.	26.10.2022			
2.	Ознайомлення з темою кваліфікаційної роботи.	30.10.2023			
3.	Розробка плану та структури роботи.	12.01.2023			
4.	Робота з науковою літературою, опис основних означень.	16.01.2023			
5.	Дослідження результатів отриманих в літературі.	13.03.2023			
6.	Робота над текстовим оформленням результатів.	05.04.2023			
7.	Попередній аналіз кваліфікаційної роботи. Виправлення помилок.	10.05.2023			
8.	Попередній захист кваліфікаційної роботи.	12.05.2023			
9.	Захист кваліфікаційної роботи.	23.05.2023			

Науковий керівник _____
(ПІБ)

Виконавець курсової роботи _____
(ПІБ)

Зміст

1 Вступ

1.1 Актуальність

Графи використовуються в різних областях, таких як транспорт, соціальні мережі, біологія та багато інших. Працюючи з графами, однією з основних завдань є їх розмітка, тобто присвоєння вершинам і/або ребрам деяких характеристик, як-от колір, вага, мітки тощо.

У статті 1967 року про розмітки графів [1] Александер Роса був першим, хто виділив новий клас розміток і ввів його під назвою β-розмітки. Соломон Голомб у 1972 році, незалежно, пропонує поняття граціозної розмітки [2], яке тотожнє до β-розмітки, і зараз цей термін використовується більш широко.

Граціозна розмітка графів є однією із важливих задач при роботі з графами, так як вона дозволяє представляти графи у вигляді послідовності чисел, яка може бути використана в різних застосуваннях. Особливістю граціозної розмітки є те, що вона дозволяє надавати кожній вершині унікальне натуральне число, а кожному ребру – унікальне число із заданого діапазону.

Застосування граціозної розмітки можна знайти в різних областях, таких як:

- Теорія графів: у теорії графів граціозні розмітки використовуються для створення різних структур даних, а також для вирішення завдань щодо пошуку певних шляхів та ланцюгів у графах.
- Комп'ютерна графіка: у комп'ютерній графіці граціозні розмітки можуть використовуватися для опису різних об'єктів, таких як зображення та 3D-моделі.
- Біологія: у біології граціозні розмітки можуть бути використані для аналізу біологічних мереж, таких як мережі білкових взаємодій.

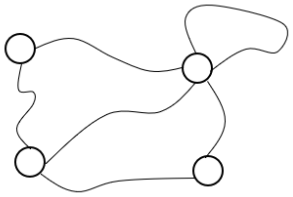
1.2 Мета, завдання дослідження

Дослідити і структурувати означення, твердження і теореми з граціозної розмітки графів. Представити огляд граціозних алгоритмів маркування для різних класів графів. Реалізувати програмний модуль мовою Python для побудови граціозної розмітки графів.

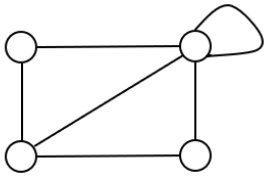
2 Граціозна розмітка графів

2.1 Основні означення теорії графів

Граф – це математичний об'єкт, який використовується для моделювання зв'язків між об'єктами. У графі об'єкти представляються вершинами (вузлами), а зв'язки між ними - ребрами (лініями). Прикладом графа може бути схема доріг у місті (рисунок 1а). В цьому випадку вершинами будуть перехрестя вулиць, а ребрами – дороги, що з'єднують ці перехрестя (рисунок 1б).



а) Схема доріг



б) Граф

Рис. 1

Існують *орієнтовані графи* (кожне ребро має напрямок та вказує на те, яка з двох вершин є початком, а яка - кінцем, тобто може бути пройдене тільки у напрямку від початкової вершини до кінцевої) та *неорієнтовані графи* (кожне ребро, що зв'язує дві вершини, може бути пройдене в обох напрямках). В подальшому будемо працювати з неорієнтованими графами. Вони є важливими у багатьох областях математики та науки про дані, так як вони можуть моделювати взаємодії між об'єктами, які не мають чіткої орієнтації. Наприклад, вони можуть використовуватися для моделювання соціальних мереж, дорожніх мереж, молекулярних структур та багато іншого. Введемо основні означення [3].

Означення 1. *Граф* G — це пара множин (V, E), де V – непорожня скінченна множина, а E – довільна підмножина множини всіх невпорядкованих пар різних елементів множини V. Позначається граф як $G = (V, E)$.

Елементи множини V називають *вершинами* графа G, а елементи множини E – його *ребрами*. Відповідно V називають *множиною вершин*, а E – *множиною ребер* графа G.

Ребро e, яке з'єднує дві вершини v і w записують за допомогою круглих дужок (v, w) (іноді просто vw). Вершини v і w називаються *кінцями ребра*

e. Для того щоб уточнити до якого саме графа відносяться дані вершини та ребра, їх позначають $V(G)$ та $E(G)$ відповідно.

Означення 2. *Суміжними вершинами* графа $G = (V, E)$ називаються такі вершини v і w, що $(v, w) \in E$. Два ребра називають *суміжними*, якщо вони мають спільну вершину.

Означення 3. *Інцидентними* вершину v і ребро e називають тоді, коли v – кінець ребра e.

Означення 4. *Степенем $\delta(v)$ вершини v* називають кількість інцидентних їй ребер.

Означення 5. *Маршрутом (шляхом)* у графі $G = (V, E)$ називають послідовність $v_1, e_1, v_2, e_2, ..., e_k, v_{k+1}$ таку, що кожні два сусідні ребра в ній мають спільну вершину. Отже, $e_i = (v_i, v_{i+1}), i = 1, 2, \ldots, k$. Вершину v_1 називають *початком*, а вершину v_{k+1} – *кінцем шляху.*

Означення 6. *Відстанню $d(v, w)$* між двома вершинами графа називають довжину найкоротшого маршруту, що з'єднує вершини v та w.

Означення 7. Маршрут, у якому всі ребра попарно різні, називають *ланцюгом*, а маршрут у якому всі вершини попарно різні – *простим ланцюгом*

Означення 8. Граф називається *зв'язним*, якщо існує шлях між будь-якою парою його вершин.

Означення 9. *Циклом* називають замкнений ланцюг ($v_1 = v_{k+1}$), а *простим циклом* називають відповідно замкнений простий ланцюг.

Означення 10. Цикл, що містить усі ребра графа, називають *ейлеровим.* Зв'язний граф, що має ейлерів цикл, називають *ейлеровим графом.*

Означення 11. Графи $G_1 = (V_1, E_1)$ і $G_2 = (V_2, E_2)$ називають *ізоморфними*, якщо існує таке взаємно однозначне відображення ϕ множини вершин V_1 на множину вершин V_2, що ребро $(v, w) \in E_1$ тоді й тільки тоді, коли ребро $(\phi(v), \phi(w)) \in E_2$.

Відображення ϕ називають *ізоморфним відображенням*, або *ізоморфізмом*, графа G_1 на граф G_2.

Означення 12. Граф називають *пласким*, якщо його діаграму можна зобразити на площині таким чином, щоб лінії, які відповідають ребрам графа, не перетиналися.

Означення 13. Граф називають *планарним*, якщо він ізоморфний деякому пласкому графу.

2.1.1 Класи графів

Для більш зручної і точної класифікації в теорії графів було введено різні класи графів, засновані на різних властивостях графів. Кожен клас графів має свої унікальні властивості та особливості, які можуть бути використані для аналізу їхньої структури та властивостей. Для позначення кожного класу графів були розроблені спеціальні позначення, які спрощують їх опис та дозволяють ідентифікувати їх серед багатьох інших графів. Перейдемо до введення означень основних класів графів.

Означення 14. *Граф-ланцюг* P_n є зв'язним графом із n вершинами, дві з яких є кінцевиви (мають степінь 1), а всі інші вершини, якщо вони є, мають степінь 2 (рисунок 2а).

Означення 15. *Цикл* C_n є таким зв'язним графом з n вершинами, що кожна вершина має степінь 2 (рисунок 2б).

Означення 16. *Деревом* T називають зв'язний граф без циклів (рисунок 2в).

Означення 17. *Гусеничним деревом* називається дерево, в якому існує такий ланцюг, що будь-яка вершина знаходиться на відстані не більше 1 від нього (рисунок 2г). Гусениця може містити петлі, але не може містити кратних ребер (два і більше ребра, інцидентних одним і тим же двом вершин).

Означення 18. Граф $G = (V, E)$ називають *повним*, якщо будь-які дві його вершини суміжні. Повний граф із n вершинами позначають K_n (рисунок 2ґ).

Означення 19. Граф $G = (V, E)$ називають *двочастковим*, якщо існує таке розбиття V_1, V_2 множини його вершин V на дві підмножини (частки), що для довільного ребра $(v, w) \in E$, або $v \in V_1$ і $w \in V_2$, або $v \in V_2$ і $w \in V_1$ (рисунок 2д).

Означення 20. Двочастковий граф $G = (V, E)$ називають *повним двочастковим графом*, якщо для будь-якої пари вершин його часток $v \in V_1$ і $w \in V_2$ маємо $(v, w) \in E$. Якщо $|V_1| = m$ і $|V_2| = n$, то повний двочастковий граф G позначають $K_{m,n}$ (рисунок 2е).

Означення 21. В теорії графів *колесом* W_n називається граф з n вершинами, утворений з'єднанням єдиної вершини зі всіма вершинами циклічного графа (рисунок 2є).

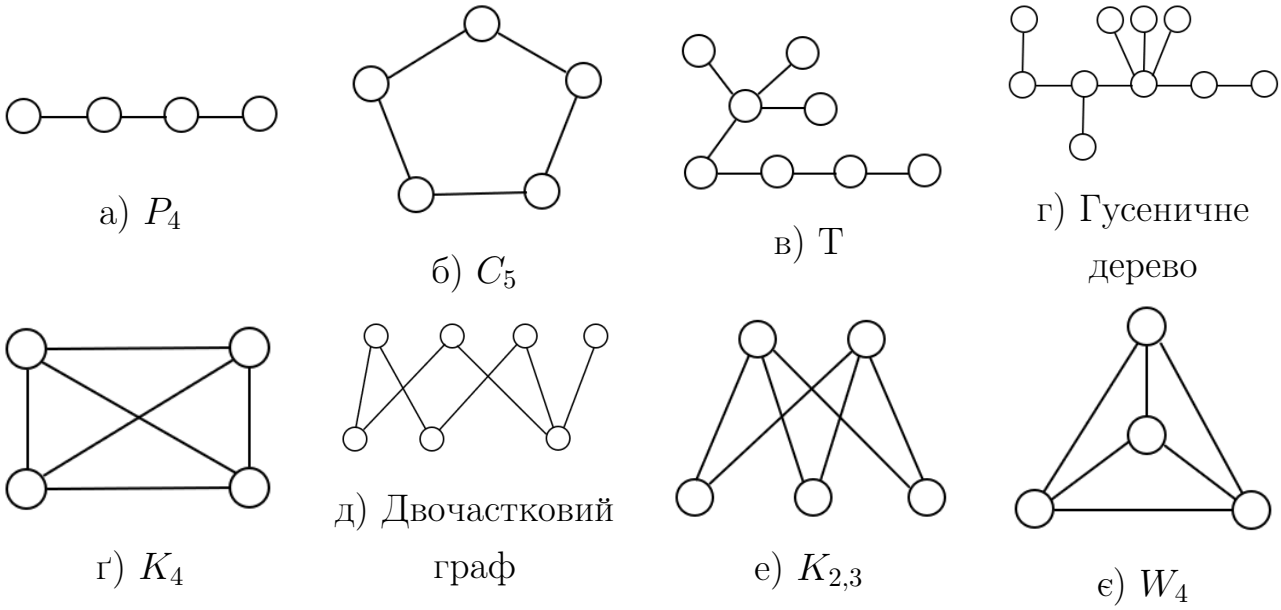


Рис. 2: Приклади розглянутих класів графів.

2.1.2 Розмітка графів

Означення 22. *Вершинною розміткою* графа $G = (V, E)$ є функція з множини вершин V у множину міток $N(f : V \to N)$. Граф з такою функцією називають графом з розміткою вершин. *Реберною розміткою* в свою ж чергу є функція з множини ребер E у множину міток $N(g : E \to N)$.

Означення 23. *Граціозною розміткою* графа G з n ребрами називають таку ін'єктивну функцію вершинної розмітки $f : V \to [0, n]$, та ін'єктивну функцію реберної розмітки $f\gamma : E \to [1, n]$, що кожне ребро позначено абсолютним значенням різниці між мітками двох його кінців ($f\gamma(uv) = |f(u) - f(v)|$). Якщо граф G допускає граціозну розмітку, то він називається *граціозним графом* (рисунок 3).

Рис. 3: Граціозні графи $K_{1,3}$ та K_3

2.1.3 Умова парності та реберно-послідовна граціозність графа

Теорема 1. Нехай G — ейлеровий граф. Якщо $m \equiv 1, 2 \pmod 4$, то G не є граціозним.

Доведення. Припустимо, що $G = (V, E)$ – граціозний ейлеровий граф. Нехай $f : V \to [0, m]$ — граціозна розмітка графа G, а $C = (u_0, u_1, ..., u_{m-1}, u_m = u_0)$ – ейлеровий цикл графа G. Взявши суму міток ребер циклу C за модулем 2 маємо:

$$\sum_{i=1}^{m} f_\gamma(u_{i-1}u_i) = \sum_{i=1}^{m} |f(u_{i-1}) - f(u_i)| \equiv \sum_{i=1}^{m} f(u_{i-1}) - f(u_i) \equiv 0 \pmod 2$$

А оскільки C є ейлеровим циклом, тобто цикл C проходить через кожне ребро рівно один раз, а f є граціозною розміткою графа G, то маємо:

$$\sum_{e \in E} f_\gamma(e) = \sum_{k=1}^{m} k = \frac{m(m+1)}{2} \equiv 0 \pmod 2$$

Таким чином, ми повинні мати $m \equiv 0, 3 \pmod 4$, щоб задовольнити дане рівняння. □

Умова парності забезпечує простий спосіб перевірити, чи може ейлеровий граф бути граціозним. Це викликає цікаве питання: чи існує клас графів, для якого умова парності також є достатньою умовою?

У теорії графів природно розглядати підструктури, що не задовольняють певні властивості графа, у нашому випадку - граціозність. Ці підструктури можуть бути підграфами, індукованими підграфами або іншими. Таким чином, можна розглядати пошук таких підструктур для класу граціозних графів. Однак Арумугам і Багга [5] довели, що кожен граф є індукованим підграфом граціозного графа.

Теорема 2. Кожен граф є індукованим підграфом граціозного графа.

Доведення. Дано граф $G = (V, E)$, побудуємо граф H з G так, що H є граціозним, а G є індукованим підграфом H. Розглянемо розмітку вершини $f : V \to [0, k]$, ін'єктивну для деякого $k \geq m$, що розмітка ребер $f_\gamma : E \to N$ також є ін'єктивною, і існують $u, v \in V$ з $f(u) = 0$ і $f(v) = k$. Нехай $\{x_1, x_2, ..., x_r\}$ – множина відсутніх міток ребер. Для загальності зауважимо,

що $x_1, x_2, ..., x_s$ не є мітками вершин, а $x_{s+1}, ..., x_r$, в свою чергу, є мітками вершин. Для кожного x_i, $1 \leq i \leq s$, додамо вершину w_i з міткою x_i і додамо ребро, що сполучає w_i з u так, щоб $f_\gamma(uw_i) = x_i$. Для кожного x_i, $s+1 \leq i \leq r$, додамо вершину w_i з міткою $k + x_i$ і з'єднаємо w_i з u і v так, щоб $f_\gamma(uw_i) = k + x_i$ і $f_\gamma(vw_i) = x_i$. Зауважимо, що на останньому кроці могли бути введені нові мітки відсутніх ребер шляхом створення міток вершин зі значеннями, більшими за k. Однак ці нові мітки відсутніх ребер не є мітками вершин. Отже, для кожної нової мітки відсутнього ребра y додамо нову вершину z_y з міткою y та з'єднаємо z_y з u так, щоб $f_\gamma(uz_y) = y$. Отриманий граф H є граціозним і містить G як індукований підграф. □

Структурувавши дане доведення, ми отримаємо наступний *алгоритм*: нехай задані розмітки $f : V \to [0, k]$ (ін'єктивна для деякого $k \geq m$) та $f_\gamma : E \to N$ (також ін'єктивна), і існують $u, v \in V$ з $f(u) = 0$ і $f(v) = k$. Нехай $\{y_1, ..., y_k\}$ – множина відсутніх міток ребер графа, $\{x_1, ..., x_r\}$ – множина міток вершин графа. Додамо вершину

$$w_i : \begin{cases} f(w_i) = y_i; f_\gamma(uw_i) = y_i, \text{ якщо } y_i \notin \{x_1, ..., x_r\} \\ f(w_i) = k + y_i; f_\gamma(uw_i) = k + y_i; f_\gamma(vw_i) = y_i, \text{ якщо } y_i \in \{x_1, ..., x_r\} \end{cases}$$

Приклад. Застосуємо цей алгоритм для K_5 та C_9.

Рисунок 4.

Крок 1: з графа K_5 бачимо, що ребра мають розмітку від 1 до 12 включно, проте маємо відсутні значення міток 5 та 7. Так як ці два значення не зустрічаються серед міток вершин графа K_5, ми просто додаємо дві вершини з мітками 5 та 7 і з'єднуємо кожну з цих вершин із верштною графа K_5, яка має мітку 0, щоб отримані ребра отримали мітки 5 та 7 відповідно. Отриманий граф є граціозним і містить K_5 як індукований підграф.

Рисунок 5.

Крок 1: відсутні значення міток графа C_9 — 1 та 5. Зауважимо, що мітка 1 зустрічаєтья серед міток вершин графа C_9, а мітка 5 – ні. Тому додамо вершину з міткою 5 та з'єднаємо її з вершиною, що має мітку 0, щоб утворилося ребро з міткою 5; а також додамо вершину з міткою $(k+1) = (13+1) = 14$ і з'єднаємо її з вершинами, що мають мітки 0 та 13, щоб мітки цих ребер мали значення $(13+1) = 14$ та 1 відповідно.

Крок 2: відсутні значення міток утвореного графа — 12 та 13. Зауважимо, що мітка 13 зустрічаєтья серед міток вершин утвореного графа, а мітка 12 – ні. Тому додамо вершину з міткою 12 та з'єднаємо її з вершиною, що має мітку 0, щоб утворилося ребро з міткою 12; а також додамо вершину з міткою $(k + 13) = (14 + 13) = 27$ і з'єднаємо її з вершинами, що мають мітки 0 та 14, щоб мітки цих ребер мали значення $(14 + 13) = 27$ та 13 відповідно.

Крок 3: відсутні значення міток новоутвореного графа — 15, 16, 17, ..., 26. Зауважимо, що жодна з цих міток не зустрічаєтья серед міток вершин новоутвореного графа. Тому додамо вершини з відповідними мітками та з'єднаємо їх з вершиною, що має мітку 0, щоб утворилися ребра з мітками 15, 16, 17, ..., 26. Отриманий граф є граціозним і містить C_9 як індукований підграф.

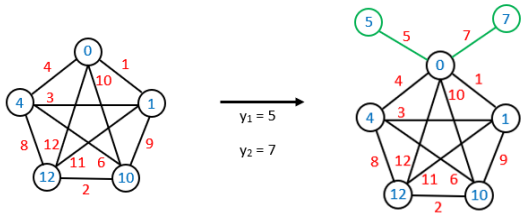


Рис. 4: Конструювання граціозного графа з K_5

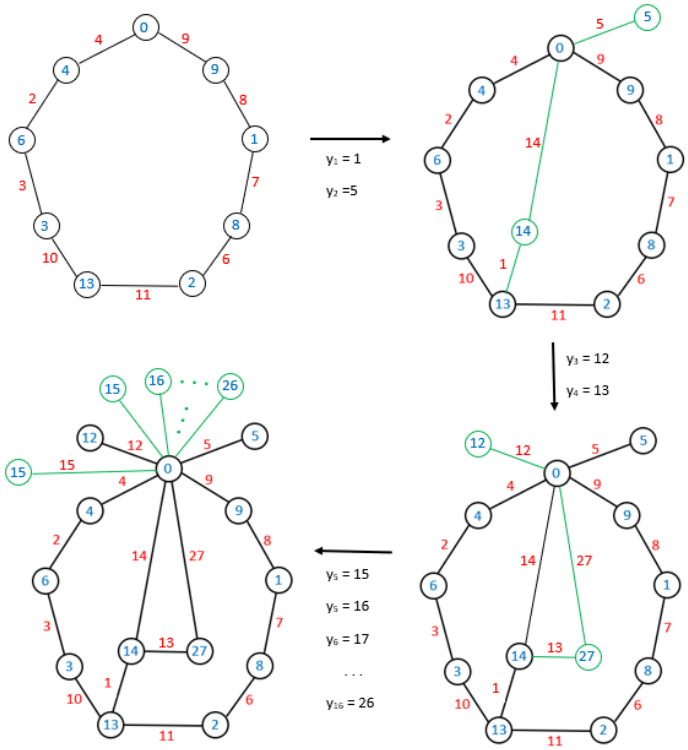


Рис. 5: Конструювання граціозного графа з C_9

2.2 Граціозна розмітка певних класів графів

У цьому розділі ми розглянемо граціозну розмітку для різних класів графів, включаючи шляхові графи, повні двочасткові графи, цикличні графи, колеса, дерева та гусениці. Представимо алгоритми побудови граціозної розмітки для кожного з цих класів [4] та роздивимось на прикладах їх застосування.

2.2.1 Граціозна розмітка графів-ланцюгів P_n

Твердження 1. Граф-ланцюг P_n є граціозним для всіх $n \geq 1$.

Доведення. Розглянемо граф-ланцюг P_n. Нехай задана така множина вершин $V(P_n) = u_0, u_1, ..., u_{n-1}$, що $u_{k-1}u_k \in E(P_n)$ для $0 < k < n$. Оскільки P_n має $m = n - 1$ ребро, ми маємо позначити вершини числами від 0 до $n - 1$ таким чином, щоб кожне число з проміжку $[1, n - 1]$ було міткою ребра. В такому разі починаємо з мітки ребра $n - 1$, тому що є лише один спосіб отримати абсолютну різницю, що дорівнює $n-1$, а саме тоді і лише тоді, коли вершина з міткою 0 суміжна з вершиною, що має мітку $n-1$. Таким чином, позначимо u_0 міткою 0, а u_1 міткою $n - 1$. Наступним кроком отримаємо мітку ребра із значенням $n-2$. Існує лише два можливі способи отримати $n-2$ як абсолютну різницю: $n - 2 = |(n - 2) - 0| = |(n - 1) - 1|$. Оскільки вершина u_0 більше не має немаркованих суміжних вершин, ми можемо отримати мітку ребра $n-2$, лише позначивши u_2 міткою 1. Продовжуючи розмітку за таким алгоритмом, наша кінцева розмітка набуде вигляду:

$$f(u_k) = \begin{cases} \frac{k}{2} & \text{якщо } k \text{ парне} \\ n - \frac{(k+1)}{2} & \text{якщо } k \text{ непарне} \end{cases}$$

Тепер для того щоб показати, що f дійсно є граціозною розміткою P_n, достатньо показати, що з'являється мітка ребра 1, яка, як очікується, з'явиться на останньому ребрі $u_{n-2}u_{n-1}$. Якщо n парне, то $f(u_{n-1}) = \frac{n}{2}$ і $f(u_{n-2}) = \frac{n-2}{2}$. Отже, $f_\gamma(u_{n-1}u_{n-2}) = \frac{n}{2} - \frac{n-2}{2} = 1$. Аналогічно встановлюємо мітку ребра 1, якщо n є непарним. □

```python
import networkx as nx

#перевірка чи є граф ланцюгом
def is_pathgraph(G):
    degrees = dict(G.degree())
    ends = [v for v, d in degrees.items() if d == 1]
    if len(ends) != 2:
        return False
    for v, d in degrees.items():
        if d != 2 and v not in ends:
            return False
    return True

#Функція, що граціозно маркує граф-ланцюг
def pathgraph_labeling(G):
  if not is_pathgraph(G):
        print("Граф не є ланцюгом")
        return
  n = len(G)
  labels = {}
  for node in sorted(G.nodes()):
    if(node % 2 == 0):
      labels[node] = node//2
    else:
      labels[node] = n - (node+1)//2
  return labels

# Приклад використання
G = nx.Graph([(0, 1), (1, 2), (2, 3)])
labels = pathgraph_labeling(G)

print("Граціозна розмітка: ", labels)
```

```
Граціозна розмітка:  {0: 0, 1: 3, 2: 1, 3: 2}
```

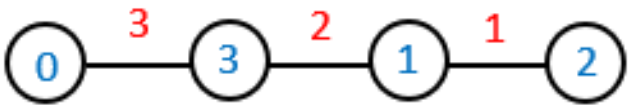


Рис. 6: Реалізація на Python побудови граціозної розмітки для P_4

2.2.2 Граціозна розмітка гусеничних дерев

Твердження 2. Всі гусеничні дерева є граціозними.

Доведення. Представимо гусеничне дерево як планарний двочастковий граф та надамо вершинам мітки так, як показано на рисунку 7. Легко перевірити, що така схема розмітки можлива завжди.

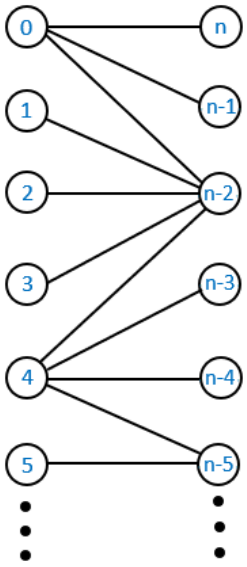
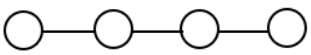


Рис. 7: Граціозна розмітка гусеничного дерева

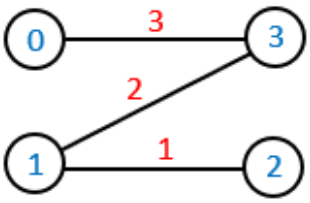
□

Зауважимо, що граф-ланцюг P_n також є гусеничним деревом, і дана схема маркування, коли вона застосована до графів-ланцюгів, дає таку саму розмітку.

Приклад. Представимо граф P_4 (рисунок 8а) як планарний та застосуємо схему граціозної розмітки як для гусеничного дерева (рисунок 8б).



а)



б)

Рис. 8

2.2.3 Граціозна розмітка повних графів K_n

Означення 24. *Комплементарністю* граціозного графа називається така властивість, що при заміні кожної мітки вершини k на $m - k$, отримана розмітка також буде граціозною, оскільки мітки ребер не зміняться: кінцеві вершини ребра з мітками a і b стануть $m - a$ і $m - b$ відповідно, і $|a - b| = |(m - a) - (m - b)|$.

Твердження 3. Повний граф K_n є граціозним тоді і тільки тоді, коли $n \leq 4$.

Доведення. Кожна вершина є суміжною з кожною іншою вершиною, таким чином, ми можемо позначити будь-яку вершину 0, а будь-яку іншу — m. Щоб отримати мітку ребра $m-1$, у нас є два варіанти: $m-1 = |(m-1)-0| = |m-1|$. Однак властивість комплементарності дозволяє нам вибрати будь-який з них. Вибравши позначення вершини міткою 1, ми отримаємо мітки ребер 1 і $m-1$. Тепер нам потрібно отримати мітку ребра $m - 2 = |(m - 2) - 0| = |(m - 1) - 1| = |m - 2|$. Ми не можемо позначити вершину мітками $m - 1$ або 2, оскільки це створить повторювану мітку ребра. Отже, нашим єдиним виходом є позначити вершину міткою $m - 2$, отримуючи мітки ребер 2, $m - 3$ та $m - 2$. Оскільки мітка $m - 3$ вже з'явилася на ребрі, наступна мітка ребра, яку ми повинні отримати, це $m - 4 = |(m - 4) - 0| = |(m - 3) - 1| = |(m - 2) - 2| = |(m - 1) - 3| = |m - 4|$. Знову ж таки, ми маємо лише один варіант без створення повторів міток ребер, який полягає в тому, щоб позначити вершину міткою 4, отримуючи реберні мітки 3, 4, $m - 6$ і $m - 4$. На цьому етапі ми позначили чотири вершини. Однак для K_5 ми мали б $m - 6 = 4$ як повторювану мітку ребра. Для $n \geq 6$ наступною міткою ребра, яку потрібно отримати, є $m - 5$. Але всі п'ять можливих способів отримати $m-5$ призводять до повтору мітки ребра. Отже, мітку $m-5$ на ребрі отримати вже неможливо. □

```python
import networkx as nx

# перевірка чи є граф повним (кількість ребер у графі має
# дорівнювати n(n-1)/2, де n - кількість вершин у графі)
def isComplete(graph):
  n = len(G.nodes())
  if len(G.edges()) == n*(n-1)/2:
    return True
  else:
    return False

#Функція, що граціозно маркує повний граф
def completeGraph_labeling(graph):
  n = len(G.nodes())
  m = len(G.edges())
  if isComplete(G) and n <= 4:
    n = len(graph)
    labeling = {}
    if n == 1:
        labeling[0] = 0
    elif n == 2:
        labeling[0] = 0
        labeling[1] = m
    elif n == 3:
        labeling[0] = 0
        labeling[1] = m
        labeling[2] = 1
    elif n == 4:
        labeling[0] = 0
        labeling[1] = m
        labeling[2] = 1
        labeling[3] = 4
    else:
        return None
    return labeling
  else:
    print("Граф не є повним або має більше 4 вершин.")

# Приклад використання
G = nx.Graph([(0, 1), (0, 2), (0, 3), (1,2), (1,3), (2,3)])
labeling = completeGraph_labeling(G)
print("Граціозна розмітка: ", labeling)
```

```
Граціозна розмітка:  {0: 0, 1: 6, 2: 1, 3: 4}
```

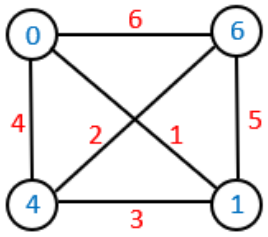


Рис. 9: Реалізація на Python побудови граціозної розмітки для K_4

2.2.4 Граціозна розмітка повних двочасткових графів $K_{m,n}$

Твердження 4. Повний двочастковий граф $K_{m,n}$ є граціозним для всіх $m, n \geq 1$.

Доведення. Нехай $G = (V_1, V_2, E)$ — двочастковий граф з $m = |V_1|$ і $n = |V_2|$. Поставимо у відповідність вершинам V_1 номери $0, 1, ..., m-1$, а вершинам з V_2 поставимо у відповідність числа $m, 2m, ..., mn$. Тоді отримана розмітка буде граціозною. □

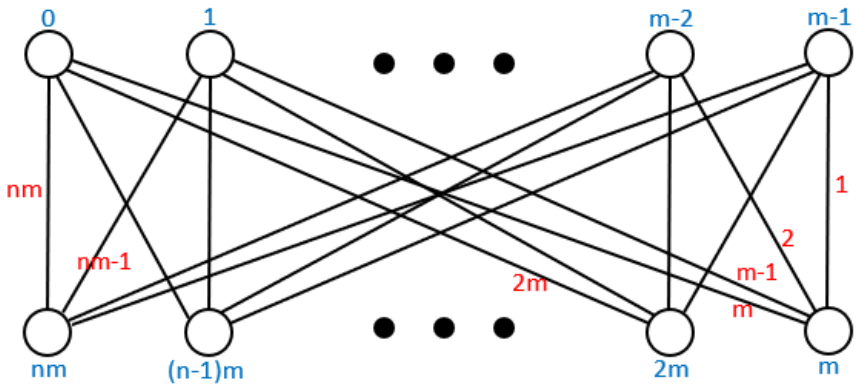


Рис. 10: Граціозна розмітка $K_{m,n}$

```python
import networkx as nx

#Функція, що граціозно маркує повний двочастковий граф
def completeBipartiteGraph_labeling(m, n):
    graph = nx.complete_bipartite_graph(m, n)
    labeling = {}
    for i in range(m):
        labeling[i] = i
    for j in range(n):
        labeling[m+j] = (j+1)*m
    return labeling

# Приклад використання
m, n = 3, 4
labeling = completeBipartiteGraph_labeling(m, n)
print("Граціозна розмітка: ", labeling)
```

```
Граціозна розмітка:  {0: 0, 1: 1, 2: 2, 3: 3, 4: 6, 5: 9, 6: 12}
```

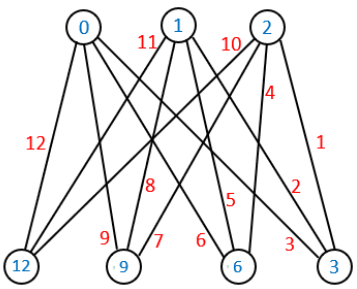


Рис. 11: Реалізація на Python побудови граціозної розмітки для $K_{3,4}$

2.2.5 Граціозна розмітка циклів C_n

Твердження 5. Цикл C_n є граціозним тоді і тільки тоді, коли $n \equiv 0, 3 \pmod 4$.

Доведення. Цикли є ейлеровими графами, тому, за умовою парності, якщо $m \equiv 1, 2 \pmod 4$, то C_n не є граціозним. Інакше назвемо $V(C_n) = \{u_0, u_1, ..., u_{n-1}\}$ так, що $u_k u_{k+1} \in E(C_n)$ для $0 \leq k \leq n-1$ та $u_n = u_0$.

Якщо $n \equiv 0 \pmod 4$, то вершини маркуються за наступною формулою:

$$f(u_i) = \begin{cases} \frac{i}{2} & \text{якщо } i = 0, 2, 4, ..., n-2 \\ n - \frac{(i-1)}{2} & \text{якщо } i = 1, 3, 5, ..., \frac{n}{2} - 1 \\ n - \frac{(i-1)}{2} - 1 & \text{якщо } i = \frac{n}{2} + 1, \frac{n}{2} + 3, ..., n-1 \end{cases}$$

Якщо $n \equiv 3 \pmod 4$, то вершини маркуються за наступною формулою:

$$f(u_i) = \begin{cases} \frac{i}{2} & \text{якщо } i = 0, 2, 4, ..., n-1 \\ n - \frac{(i-1)}{2} & \text{якщо } i = 1, 3, 5, ..., \frac{n+1}{2} - 1 \\ n - \frac{(i-1)}{2} - 1 & \text{якщо } i = \frac{n+1}{2} + 1, \frac{n+1}{2} + 3, ..., n-2 \end{cases}$$

□

```python
import networkx as nx

#перевірка чи є граф циклом
def is_cycle(G):
    if not nx.is_connected(G):
        return False
    for node in G.nodes:
        if G.degree(node) != 2:
            return False
    return True

#Функція, що граціозно маркує цикл
def cycle_labeling(G):
  if not is_cycle(G):
    print("Граф не є циклом")
    return
  n = G.number_of_nodes()
  labels = {}
  if(n % 4 == 0):
    for i in range(n):
      if i in range(n//2 + 1, n, 2):
        labels[i] = n - (i-1)//2 - 1
      elif (i % 2 == 0 and i < n-1):
        labels[i] = i // 2
      elif i in range(1, (n//2), 2):
        labels[i] = n - (i-1)//2
    return labels
  elif (n % 4 == 3):
    for i in range(n):
      if i in range((n+1)//2 + 1, n-1, 2):
        labels[i] = n - (i-1)//2 - 1
      elif(i % 2 == 0):
        labels[i] = i // 2
      elif i in range(1, ((n+1)//2), 2):
        labels[i] = n - (i-1)//2
    return labels
  else:
      return("Для даної к-ті вершин циклу неможливо задати граціозну розмітку")

# Приклад використання
C4 = nx.Graph([(0, 1), (1, 2), (2, 3), (3, 0)])
labeling_C4 = cycle_labeling(C4)
print("Граціозна розмітка C4: ", labeling_C4)

C7 = nx.Graph([(0, 1), (1, 2), (2, 3), (3, 4), (4, 5), (5, 6), (6, 0)])
labeling_C7 = cycle_labeling(C7)
print("Граціозна розмітка C7: ", labeling_C7)
```

```
Граціозна розмітка C4:  {0: 0, 1: 4, 2: 1, 3: 2}
Граціозна розмітка C7:  {0: 0, 1: 7, 2: 1, 3: 6, 4: 2, 5: 4, 6: 3}
```

Рис. 12: Реалізація на Python побудови граціозної розмітки для C_4 та C_7

2.2.6 Граціозна розмітка графів-колес W_p

Роберт Фрухт [6] показав, що всі колеса є граціозними графами.

Твердження 6. Колесо W_p є граціозним для всіх $p \geq 3$.

Доведення. Нехай $V(W_p) = \{u_0, u_1, ..., u_{p-1}, v\}$ — множина вершин, де v — вершина, з'єднана з циклом. Розглянемо два випадки:

1. Якщо $p \equiv 0 \pmod 2$, то вершини маркуються за наступною формулою:

$$f(v) = 0$$

$$f(u_i) = \begin{cases} 2p & \text{якщо } i = 0 \\ 2 & \text{якщо } i = p-1 \\ i & \text{якщо } i = 1, 3, 5, ..., p-3 \\ 2p-i-1 & \text{якщо } i = 2, 4, 6, ..., p-2 \end{cases}$$

2. Якщо $p \equiv 1 \pmod 2$, то вершини маркуються за наступною формулою:

$$f(v) = 0$$

$$f(u_i) = \begin{cases} 2p & \text{якщо } i = 0 \\ 2 & \text{якщо } i = 1 \\ p+i & \text{якщо } i = 2, 4, 6, ..., p-1 \\ p+1-i & \text{якщо } i = 3, 5, 7, ..., p-2 \end{cases}$$

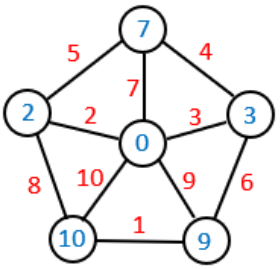
□

```python
#Функція, що граціозно маркує колесо
def wheel_labeling(graph):
    p = len(graph) - 1
    labeling = {}
    if(p % 2 == 0):
      for i, vertex in enumerate(graph):
        if vertex == "v":
          labeling[vertex] = 0
        elif i == 0:
            labeling[vertex] = 2*p
        elif i == p-1:
            labeling[vertex] = 2
        elif i % 2 == 1:
            labeling[vertex] = i
        else:
            labeling[vertex] = 2*p - i - 1
    if(p % 2 == 1):
      for i, vertex in enumerate(graph):
        if vertex == "v":
          labeling[vertex] = 0
        elif i == 0:
          labeling[vertex] = 2*p
        elif i == 1:
          labeling[vertex] = 2
        elif i % 2 == 0:
          labeling[vertex] = p + i
        else:
          labeling[vertex] = p + 1 - i
    return labeling

# Приклад використання
W5 = ["u0", "u1", "u2", "u3", "u4", "v"]
labeling_W5 = wheel_labeling(W5)
print(labeling_W5)

W4 = ["u0", "u1", "u2", "u3", "v"]
labeling_W4 = wheel_labeling(W4)
print(labeling_W4)
```

```
{'u0': 10, 'u1': 2, 'u2': 7, 'u3': 3, 'u4': 9, 'v': 0}
{'u0': 8, 'u1': 1, 'u2': 5, 'u3': 2, 'v': 0}
```



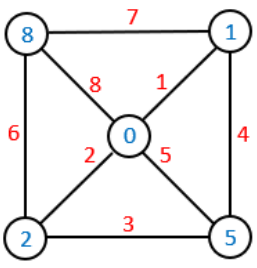


Рис. 13: Реалізація на Python побудови граціозної розмітки для W_5 та W_4

2.3 Граціозна розмітка дерев

Граціозність дерев є предметом тривалих досліджень теорії графів. Цікаво відзначити, що існує гіпотеза про те, що всі дерева є граціозними, що робить їх ще більш унікальними та привабливими для вивчення. Гіпотеза була вперше сформульована в 1976 р. [1], і з того часу не припиняються намагання та спроби її довести або спростувати.

2.3.1 Дерева з обмеженим діаметром

Означення 24. Ексцентриситетом $e(v)$ довільної вершини v зв'язного графа $G = (V, E)$ називають найбільшу з відстаней між вершиною v і всіма іншими вершинами графа G, тобто $e(v) = max\{d(v, w)\}$, де $w \in V$.

Означення 25. Діаметром зв'язного графа G (позначають $D(G)$) називають максимальний зі всіх ексцентриситетів вершин графа G.

Твердження 7. Дерева діаметру 1 є граціозними.

Доведення. Дане твердження випливає з того, що дерева з діаметром 1 є повними графами K_2, для яких вже була доведена граціозність. □

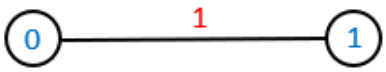


Рис. 14: Граціозна розмітка дерева діаметру 1

Твердження 8. Дерева діаметру 2 є граціозними.

Доведення. Дане твердження випливає з того, що дерева з діаметром 2 є зірками (дерева, які мають не більше однієї вершини, степінь якої перевищує одиницю, а усі інші вершини мають степінь одиниця). Зокрема, зірка є повним двочастковим графом $K_{1,q}$, а для графів даного класу вже була доведена граціозність. □

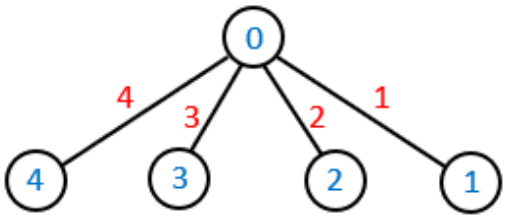
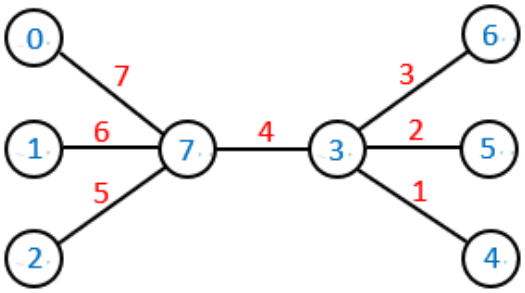


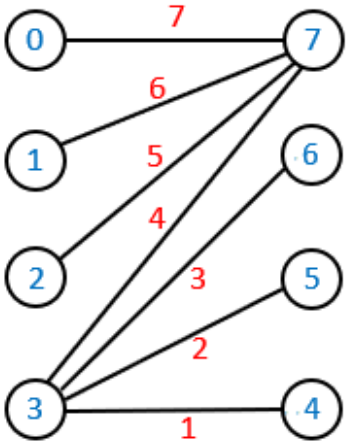
Рис. 15: Граціозна розмітка дерева діаметру 2

Твердження 9. Дерева діаметру 3 є граціозними.

Доведення. Дане твердження випливає з того, що дерева діаметру 3 є підкласом гусеничних дерев. □



(а)



(б)

Рис. 16: Граціозна розмітка дерева діаметру 3 (а) та його представлення у вигляді планарного двочасткового графа (б)

Для того щоб довести, що дерева з діаметром більшим за 4 також є граціозними, введемо деякі додаткові поняття.

Лема 1. Нехай T – дерево з граціозною розміткою f і $u \in V(T)$ – вершина з $f(u) = 0$. Якщо T' – це дерево, отримане з T шляхом додавання нової вершини v, яка є суміжною тільки з u, тоді T' також є граціозним.

Наслідок 1.1 Якщо вершина $w \in V(T)$ має мітку m, то після додавання нової вершини, суміжної тільки з w, також отримуємо граціозне дерево.

Наслідок 1.2 Якщо $u \in V(T)$ має мітку 0 (або m), а H є гусеничним деревом, то після додавання ребра між вершиною u і вершиною H з максимальним ексцентриситетом також отримуємо граціозне дерево.

Лема 1 дозволяє отримувати нові граціозні графи з менших шляхом додавання вершини. Тоді доцільно запитати, чи можна це використати, щоб довести гіпотезу граціозності дерев, тобто якось показати, що для будь-якого дерева існує кінцева послідовність граціозних дерев, яка починається з дерева з однією вершиною, кожне наступне дерево у цій послідовності є попереднім деревом до якого додали нову вершину, а останнє дерево послідовності є самим цільовим граціозним деревом.

Однією з достатніх умов існування такої послідовності є те, що кожне дерево допускає граціозну розмітку, в якій мітка 0 може бути присвоєна будь-якій вершині. У загальному контексті такі графи називають *0-поворотним* граціозними графами. Однак слід зауважити, що не кожне дерево є граціозним 0-поворотним [7].

Нехай T – дерево і $uv \in E(T)$. Позначимо через $T_{u,v}$ піддерево T, що містить v після видалення ребра uv. Точніше, якщо $S = \{w \in V(T) : v$ знаходиться на шляху з u у $w\}$, тоді $T_{u,v} = T[S]$.

Лема 2. Нехай T – дерево з граціозною розміткою f і $u \in V(T)$ – суміжна вершина з u_1 і u_2. Розглянемо $T' = T - (V(T_{u,u_1}) \cup V(T_{u,u_2}))$ і нехай $v \in V(T'), v \neq u$.

(а) Якщо $u_1 \neq u_2$ і $f(u_1)+f(u_2) = f(u)+f(v)$, то дерево, отримане диз'юнктним об'єднанням T', T_{u,u_1} і T_{u,u_2}, і з'єднанням v з u_1 і u_2 буде граціозним з тією ж самою граціозною розміткою f.

(б) Якщо $u_1 = u_2$ і $2f(u_1) = f(u) + f(v)$, то дерево, отримане диз'юнктним об'єднанням T' і T_{u,u_1}, і з'єднанням v з u_1 буде граціозним з тією ж самою граціозною розміткою f.

Доведення. Досить показати, що мітки ребер uu_1 і uu_2 такі самі, як і мітки vu_1 і vu_2

(а) $|f(u_1) - f(u)| = |f(u) + f(v) - f(u_2) - f(u)| = |f(v) - f(u_2)|$
$|f(u_2) - f(u)| = |f(u) + f(v) - f(u_1) - f(u)| = |f(v) - f(u_1)|$

(б) $|f(u_1) - f(u)| = |\frac{f(u)+f(v)}{2} - f(u)| = |\frac{f(v)-f(u)}{2}|$
$|f(u_1) - f(v)| = |\frac{f(u)+f(v)}{2} - f(v)| = |\frac{f(u)-f(v)}{2}|$

□

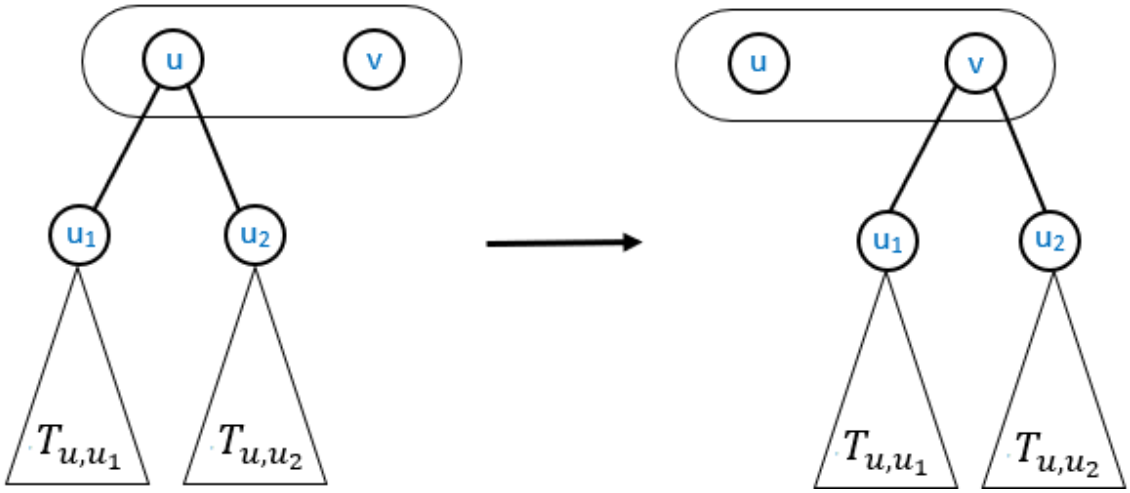


Рис. 17: Перенесення піддерева з u в v

Описана операція називається *перенесенням*. Здебільшого виконується саме перенесення листків дерева з однієї вершини на іншу. Послідовність перенесення суміжних листків з вершиною u до вершини v позначається як $u \to v$. Надалі для граціозних дерев утотожнюватимемо мітки вершин із самими вершинами, оскільки в дереві кожне число з $[0, n-1]$ має з'явитися як мітка вершин.

Твердження 10. Дерева діаметру 4 є граціозними.

Доведення. Розглянемо наступні види перенесень.

Перенесення $u \to v$ має тип 1, якщо перенесеними листками є $k, k+1, ..., k+s$. Цей вид перенесення можна реалізувати, якщо $u+v = k+(k+s)$. Ми використовуємо цей тип перенесення, коли хочемо залишити непарну кількість вершин, з'єднаних з u.

Перенесення $u \to v$ має тип 2, якщо перенесеними листками є $k, k+1, ..., k+s$ і $l, l+1, ..., l+s$ з $k+s < l$. Цей вид перенесення можна реалізувати, якщо $u+v = k+(l+s)$. Ми використовуємо цей тип перенесення, коли хочемо залишити парну кількість вершин, з'єднаних з u.

Відповідно до леми 1 достатньо показати, що кожне дерево T з діаметром 4 із центральною вершиною (яка єдина в T) непарного ступеня має граціозну розмітку, причому центральна вершина має максимальну мітку. Це справджується, тому що в дереві діаметром 4 будь-яке піддерево, яке корениться в одному з дочірніх елементів центральної вершини, є гусеничним деревом.

Нехай w – центральна вершина T, x – кількість вершин, суміжних з w, з

парним степенем, і y – кількість вершин, суміжних з w, з непарним степенем, більшим за 1. Нехай $d(w) = 2k + 1$ і розглянемо дерево на рисунку 18. Ми можемо отримати T з цього дерева за такою послідовністю перенесення: $0 \to m-1 \to 1 \to m-2 \to 2 \to m-3 \to ...$, де перші x переносів (або $x-1$, якщо $y = 0$) мають тип 1, а наступні $y - 1$ переносів (якщо $y > 1$) мають тип 2.

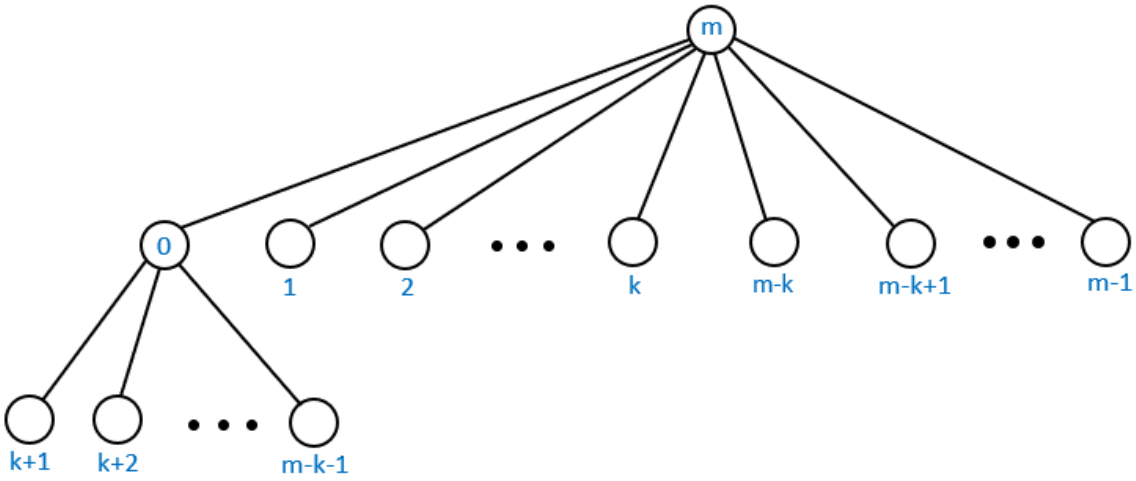


Рис. 18

Щоб переконатися, що ця послідовність працює, проаналізуємо перше перенесення. Нехай $\{u_1, ..., u_x\}$ – множина вершин, суміжних з w парного степеня. Починаючи з дерева на рисунку 18, центральна вершина w має мітку m. Перший перехід $0 \to m - 1$. Тоді u_1 є вершиною 0, і ми хочемо залишити $d(u_1) - 1$ вершин, приєднаних до неї. Спочатку ми маємо вершини $k + 1, k + 2, ..., m - k - 2, m - k - 1$, суміжні з 0. Оскільки $0 + (m - 1) = (k + 1) + (m - k - 2)$, то можна залишити $d(u_1) - 1$ вершин, виконавши перенесення типу 1 безперервної послідовності вершин до $m - 1$. Продовжуючи аналогічний аналіз, можна побачити, що ця послідовність працює. □

2.3.2 Всі дерева, які мають до 35 вершин, є граціозними

Для розв'язання питання про істинність гіпотези граціозності дерев існують два основних підходи. Перший полягає в доведенні граціозності та отриманні алгоритмів граціозної нумерації для окремих класів дерев. Для того, щоб довести гіпотезу, необхідно продемонструвати граціозність всіх можливих класів дерев. На сьогодні граціозність було доведено для таких класів як зірки, ланцюги, гусениці, оливкові дерева, деякі підкласи омарів (феєрверки, (2,2)-

гусениці, омари з досконалими парополученнями), бананові та узагальнені бананові дерева, а також для багатьох інших класів дерев. Крім того, було знайдено значну кількість методів поєднання кількох граціозних дерев для отримання більшого граціозного дерева.

Існує ще один спосіб відповісти на питання про те, чи є граціозними всі дерева - це використання комп'ютера. Так як гіпотеза про граціозність дерев була відкритою протягом тривалого часу, можна поставити під сумнів її істинність. Якщо знайдеться хоча б одне дерево, яке не допускає граціозної розмітки, то гіпотеза буде спростована. Але щоб довести, що дерево не може мати такої розмітки, потрібно перевірити експоненціальну кількість можливих способів його маркування. Тому використання комп'ютера є більш ефективним для розв'язання цього завдання. На даний момент не було знайдено жодного дерева, яке не задовольняло б умови граціозності.

У 2010 році Фанг [8] підтвердив граціозність всіх дерев з не більше ніж 35 вершинами, використовуючи детермінований алгоритм з поверненням. Цей результат замінює попередні результати в цьому напрямку: Олдред та Мак-Кей [9] встановили граціозність всіх дерев з не більше ніж 27 вершинами у 1998 році, а Хортон [10] підтвердив цей факт для всіх дерев з не більше ніж 29 вершинами у 2003 році.

Хоча існує вагомий аргумент на користь гіпотези про граціозність дерев – знаходження граціозної розмітки для всіх дерев з не більше ніж 35 вершинами, але можливість існування неграціозних дерев з більшою кількістю вершин не може бути виключена. Однак, більшість дослідників підтримують гіпотезу про граціозність дерев на основі граціозності багатьох класів дерев, результатів комп'ютерного пошуку та відсутності контрприкладів. Один з таких дослідників зазначив, що "віра в справедливість гіпотези про граціозність дерев настільки сильна, що, навіть якщо було б знайдено дерево, яке не допускає граціозної нумерації, його, швидше за все, не визнали б деревом"[11].

3 Висновки

Зважаючи на результати дослідження, можна зробити висновок, що граціозна розмітка графів є важливою та актуальною темою в теорії графів. В даній роботі розглянуті основні означення теорії графів, класи графів та граціозна розмітка графів. Зокрема, розглянуті умова парності та реберно-послідовна граціозність графа, а також завдяки її структуризації був виведений алгоритм побудови граціозного графа з неграціозного, який міститься в ньому як індукований підграф.

Також були досліджені граціозні розмітки певних класів графів, такі як графи-ланцюги, гусеничні дерева, повні графи, повні двочасткові графи, цикли та графи-колеса. Для кожного з цих класів графів була розроблена програма на Python, яка повертає його граціозну розмітку.

Таким чином, було продемонстровано, що граціозна розмітка може бути застосована до різних класів графів, що відкриває широкі можливості для подальших досліджень у цій галузі. Результати дослідження можуть бути корисні для розробки алгоритмів розподілу ресурсів у мережах зв'язку, побудови маршрутів в мережах передачі даних, аналізу білкових взаємодій та ін.

Отже, можна зробити висновок, що граціозна розмітка графів є актуальною та важливою темою у теорії графів, яка має практичне застосування.

4 Список літератури

[1]. Theory of Graphs International Symposium Rome 1966. Theory of graphs, international symposium: theorie des graphes, journees internationales d'etude; [papers]. New York : Gordon and Breach, 1967. P. 349-355.

[2]. Golomb S. W. How to number a graph. Graph theory and computing. Academic Press. New York. 1972. P.2.

[3]. Трохимчук Р. М., Нікітченко М. С. Теорія графів. Дискретна математика у прикладах і задачах : навч. посіб. С. 161–186. URL: http://csc.knu.ua/media/filer_public/89/10/89101127-5400-4d61-9840-7eab32caddab/discrete_mathematics.pdf.

[4]. Rodrigo Ming Zhou. Graceful labeling of graphs. Rio de Janeiro, 2016. 40 p. URL: https://www.cos.ufrj.br/uploadfile/publicacao/2668.pdf.

[5] Arumugam S., Bagga J. Graceful labeling algorithms and complexity – a survey. Journal of the indonesian mathematical society. 2012. URL: https://doi.org/10.22342/jims.0.0.14.1-9.

[6]. Frucht R. Graceful numbering of wheels and related graphs. Annals of the new york academy of sciences. 1979. Vol. 319, no. 1 Second Intern. P. 219–229. URL: https://doi.org/10.1111/j.1749-6632.1979.tb32792.x.

[7] Van Bussel F. 0-Centred and 0-ubiquitously graceful trees. Discrete mathematics. 2004. Vol. 277, no. 1-3. P. 193–218. URL: https://doi.org/10.1016/s0012-365x(03)00152-3.

[8] A new approach of the division optimization of attribute spaces for the decision tree construction / D. Zhang et al. 2010 third international workshop on advanced computational intelligence (IWACI), Suzhou, China, 25–27 August 2010. 2010. URL: https://doi.org/10.1109/iwaci.2010.5585144.

[9] Aldred R. E. L., McKay B. D. Graceful and harmonious labellings of trees.

Bulletin of the Institute of Combinatorics and its Applications 23 (1998). P. 69–72.

[10] Horton M. Graceful trees: statistics and algorithms : thesis. 2003. URL: https://eprints.utas.edu.au/19/1/GracefulTreesStatisticsAndAlgorithms.pdf.

[11] Alfalayleh, M., Brankovic, L., Giggins, H., Islam, M. Z. Towards the graceful tree conjecture: a survey. Proceedings of AWOCA2004. Ballina, Australia. 2004.