DOI 10.34185/1562-9945-6-155-2024-03
УДК 004.421.2

О.В. Глушков

ПОБУДОВА ДЕРЕВА ШТЕЙНЕРА ЗА ДОПОМОГОЮ МЕТОДА КЛАСТЕРИЗАЦІЇ

Анотація. В даній роботі розглядається метод побудови дерева Штейнера для оптимізації мережевих структур у розподілених комп'ютерних системах. Основна мета роботи полягає у дослідженні та впровадженні вдосконаленого алгоритму для знаходження точок Штейнера за допомогою методу кластеризації. Основна ідея методу полягає у використанні специфічного підходу до визначення точок Штейнера, що оптимізують під'єднання заданих точок у просторі. Метою цього підходу є зменшення обчислювальної складності, зберігаючи при цьому адекватну точність у побудові дерева Штейнера. Через спрощений підхід до кластеризації та визначення точок Штейнера, цей метод має потенціал значно оптимізувати процес вирішення поставленого завдання, особливо в сценаріях з великою кількістю точок. Для визначення його ефективності проведено дослідження на графах з чотирма, п'яти та шести вершинами розташованими на площині випадковим чином. Тестування проводилось за допомогою спеціального програмного забезпечення, написаного мовою Python. Загалом, дослідження показало, що метод кластеризації є ефективним інструментом для визначення точок Штейнера, що дозволяє знизити обчислювальну складність та забезпечити адекватну точність у побудові дерева Штейнера. Подальші дослідження в цьому напрямку можуть сприяти вдосконаленню методів оптимізації мережевих структур, що є важливим для широкого спектру практичних застосувань.
Ключові слова: дерево Штейнера, мінімальне остовне дерево, метод оптимізації, кластеризація графа.

Постановка проблеми. Задача Штейнера (або задача дерева Штейнера) - це комбінаторна оптимізаційна задача в графах, яка полягає в пошуку найкоротшого можливого дерева (графа) для з'єднання заданого набору вершин, які називаються терміналами. Відрізняючись від більш традиційних мінімальних остовних дерев, дерево Штейнера може включати додаткові точки, які не є частиною первісного набору, але які допомагають скоротити загальну довжину дерева. Додаткові точки відомі як точки Штейнера.

Ця задача є комбінаторною оптимізаційною задачею NP-складності і може бути дуже складною для розв'язання, особливо на великих графах. Тому існують різні алгоритми, які намагаються наблизити оптимальне рішення або знайти розв'язок в прийнятний час.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Метод Штейнера має безліч застосувань у різних галузях, включаючи телекомунікації (для з'єднання мереж з мінімальною довжиною кабелів), маршрутизацію мереж, виробництво печатних плат, дизайн логістичних систем і багато інших областей.

Найпростіший приклад задачі Штейнера, коли кількість точок дорівнює трьом. Рішення задачі Штейнера для трьох точок було знайдене ще в XVII столітті. П. Ферма, Є. Торричеллі та Б. Кавальєрі в свій час довели, що дерево Штейнера для трикутнику будується за допомогою однієї додаткової точки (при умові що всі кути трикутника менші 120°), котра називається точкою Ферма-Торричелли-Штейнера. Якщо один з кутів трикутника більше або дорівнює 120°, то дерево Штейнера складається зі сторін цього кута [1].

На Рис. 1 показаний процес побудови точки Штейнера S для трьох точок A, B, C. Для пошуку точки Штейнера треба на будь-якій стороні начального трикутника добудувати назовні рівносторонній трикутник A'BC. Точка S перетин кола, описаного навколо трикутника A'BC, та відрізка AA' буде являтися точкою Штейнера. Також цю точку можна знайти, якщо добудувати рівносторонні трикутники к будь-яким іншим сторонам [2].

Рішення задачі Штейнера для трьох точок є основою для побудови найкоротшої сітки Штейнера для великої кількості точок. Відомо, що для будь-якої кінцевої системи точок на площини існує кінцева множина сіток Штейнера. Для находження найкоротшого шляху серед множини точок треба знайти всі сітки Штейнера, а потім вибрати найкоротшу. Однак точне рішення задачі потребує розгляду великої кількості варіантів, так що навіть найкращі алгоритми, що виконуються на самих швидкодіючих комп'ютерах, не в змозі надати рішення для великої множини заданих точок за реально прийнятний час. Більше того, задача Штейнера належить до класу задач, для яких, на думку багатьох сучасних дослідників, ефективні алгоритми ніколи не будуть знайдені. З огляду на затребуваність задачі для практичних додатків, актуальною є побудова нових алгоритмів, у тому числі дають наближене рішення задачі Штейнера [3].

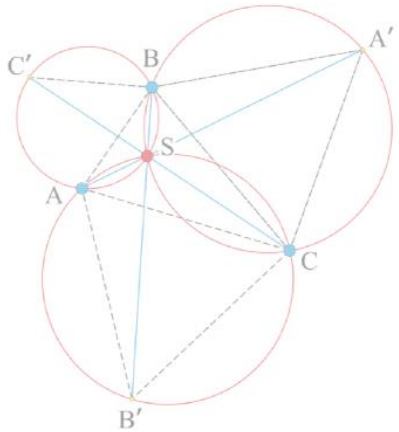

Рисунок 1 - Побудова точки Штейнера S для трикутника ABC геометричним методом[1]

Мета. В цій роботі передбачається дослідити вдосконалений алгоритм знаходження точок Штейнера для побудови мереж в розподілених комп'ютерних системах за

допомогою розробленого програмного забезпечення. Для програмної реалізації обрано мову Python.

Методика. Попри значну актуальність та важливість побудови дерева Штейнера, варто зазначити, що ця задача належить до надскладних проблем. Це означає, що не існує відомого алгоритму, який здатен знайти точне оптимальне рішення для довільного вхідного набору даних у поліноміальний час. В результаті, більшість існуючих алгоритмів зосереджуються на наближених або евристичних рішеннях, які можуть не гарантувати оптимальності, але забезпечують практично прийнятні результати у розумний час.

Однак, навіть ці методи часто стикаються з обмеженнями у вигляді високих обчислювальних витрат та недостатньої точності, особливо при роботі з великими або складними графами. Таким чином, розробка альтернативних алгоритмів, які здатні ефективніше вирішувати цю задачу, є критично важливою для подальшого прогресу в областях, де використовується дерево Штейнера.

Одним з таких методів може бути метод кластеризації точок на площині. Основна ідея методу полягає у використанні специфічного підходу до визначення точок Штейнера, що оптимізують під'єднання заданих точок у просторі. Метою цього підходу є зменшення обчислювальної складності, зберігаючи при цьому адекватну точність у побудові дерева Штейнера. Через спрощений підхід до кластеризації та визначення точок Штейнера, цей метод має потенціал значно оптимізувати процес вирішення поставленого завдання, особливо в сценаріях з великою кількістю точок. Для визначення його ефективності проведемо дослідження на графах з чотирма, п’яти та шести вершинами розташованими на площині випадковим чином.

Програма Steiner Point Calculator для знаходження точки Штейнера для трикутника. Експериментальна програма для обчислення та візуалізації точки Штейнера виконана мовою Python з використанням стандартних бібліотек: Tkinter для створення графічного інтерфейсу користувача (GUI), Matplotlib для малювання графіків та NumPy для математичних обчислень. Основна частина програми, функція обчислення точки Штейнера, здійснюється за допомогою методу оптимізації Nelder-Mead, який є частиною бібліотеки scipy.optimize. Цей метод дозволяє знайти точку всередині трикутника, яка мінімізує загальну відстань до трьох вершин.

Інтерфейс користувача програми складається з полів для вводу координат трьох точок, що формують трикутник, та вікно для графічного відображення. Результат представлений як текстове повідомлення з координатами точки та візуалізацією на графіку, де зображено вхідні точки, їхнє з'єднання та розташування точки Штейнера. Інтерфейс модуля Steiner Point Calculator представлений на рис. 2.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

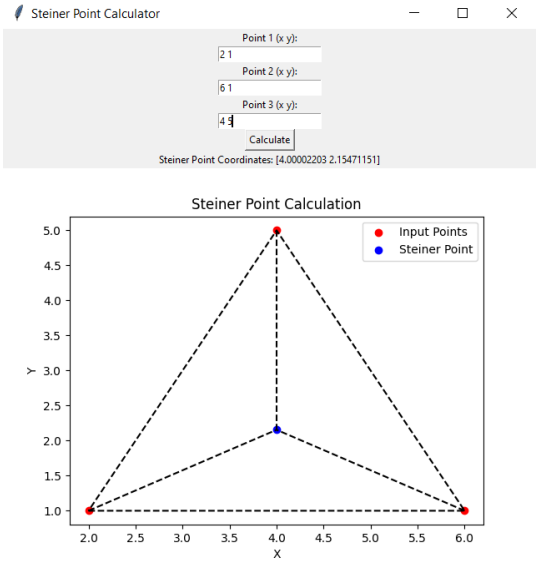

Рисунок 2 - Інтерфейс модуля Steiner Point Calculator

Виконання коду відбувається наступним чином:

Імпорт необхідних бібліотек.

Опис функцій:

- calculate_steiner_point(p1, p2, p3): Розраховує точку Штейнера для заданих точок p1, p2, p3. Використовує метод оптимізації Nelder-Mead з scipy.optimize для мінімізації суми відстаней від точки до вершин трикутника.

- distance_sum(point, p1, p2, p3): Допоміжна функція всередині calculate_steiner_point, яка обчислює суму відстаней від заданої точки до трьох точок трикутника.

- on_calculate(): Зчитує координати з текстових полів, обчислює точку Штейнера, викликає функцію plot_graph для візуалізації результатів та виводить координати точки Штейнера.

- plot_graph(p1, p2, p3, steiner_point): Малює графік з вхідними точками та точкою Штейнера.

- Програма виконує функцію on_calculate, яка перетворює введені дані в числовий формат, перевіряє їх валідність та викликає calculate_steiner_point.

- calculate_steiner_point використовує метод оптимізації для знаходження точки Штейнера.

- Після знаходження точки Штейнера, plot_graph візуалізує вхідні точки та точку Штейнера на графіку.

- Координати точки Штейнера відображаються на інтерфейсі.

Фрагмент коду програми представлено на Рис. 3.

```
def calculate_steiner_point(p1, p2, p3):
    def distance_sum(point, p1, p2, p3):
        return (
            np.linalg.norm(np.array(point) - np.array(p1)) +
            np.linalg.norm(np.array(point) - np.array(p2)) +
            np.linalg.norm(np.array(point) - np.array(p3))
        )
    initial_point = [(p1[0] + p2[0] + p3[0]) / 3, (p1[1] + p2[1] + p3[1]) / 3]
    result = minimize(distance_sum, initial_point, args=(p1, p2, p3), method='Nelder-Mead')
    if result.success:
        return result.x
    else:
        raise ValueError("Optimization failed to find the Steiner point")
```

Рисунок 3 - Фрагмент коду програми Steiner Point Calculator»

Для чотирьох точок

Вибір Точок: Візьмемо чотири точки, розміщених випадково на площині з такими координатами (Рис. 4):

A (2, 3); B(4, 9); C(13, 7); D(10, 1)

Довжина ребер рахується за формулою (1):

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{1}$$

AB=6,32; BC= 9.22; CD=6,71; DA=8,25.

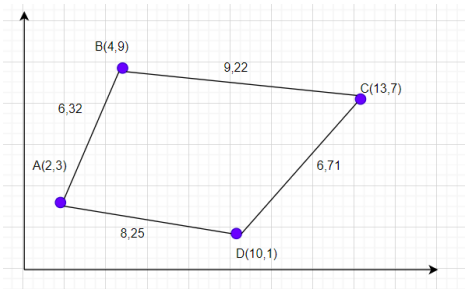

Рисунок 4 - Початкові точки графа

Кластеризація: Спочатку ми поділимо площу на 2 трикутника (Рис. 5). Далі знаходимо точки Штейнера (ТШ) для кожного з цих трикутників за допомогою програми Steiner Point Calculator.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

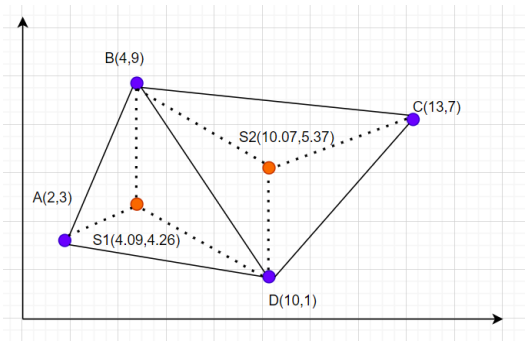

Рисунок 5 - Розділення графа на кластери

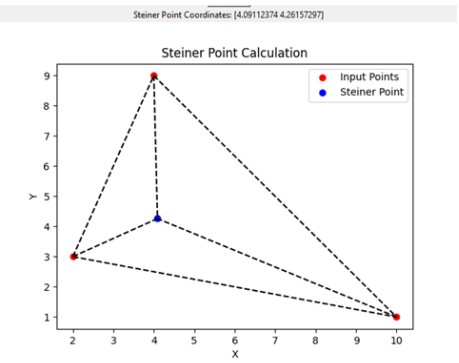

Рисунок 6 - Знаходження ТШ для трикутника

Координати нових точок: S1 (4.09, 4.26); S2(10.07, 5.37); Об'єднаємо точки в дерево Штейнера (ДШ): по дві найближчі до отриманих та з'єднаємо їх.

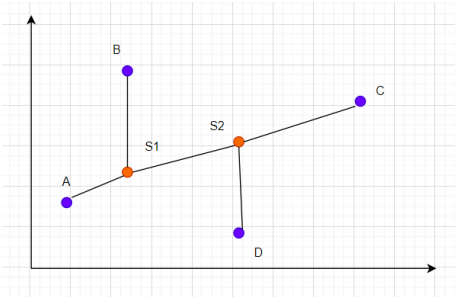

Рисунок 7 - ДШ для 4-х точок прямим з'єднанням

Для порівняння знайдемо суму всіх ребер в отриманому дереві Штейнера та мінімальним остовним деревом (МОД) для чотирьох початкових точок.

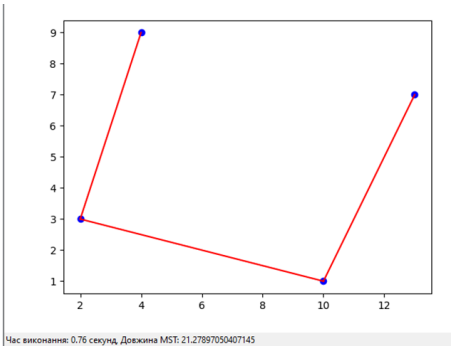

Рисунок 8 - МОД для 4 точок за допомогою EMST

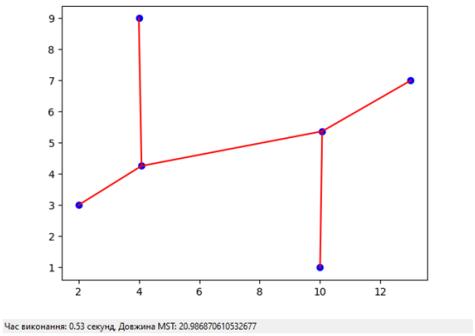

Рисунок 9 - ДШ для 4-х точок за допомогою EMST

Довжина МОД 21.28

Довжина ребер дерева Штейнера за формулою (1):

A–S1= 2,44; B–S1=4,74; C–S2=3,35; D–S2=4,37; S1–S2=6,08;

Загальна сума довжин ребер після з'єднання на графі: 20,98

Загальна сума довжин ребер ДШ за допомогою EMST: 20,98

Загальна сума довжин ребер МОД за допомогою EMST: 21,28

Для п’яти точок

Візьмемо п’ять точок, розміщених випадково на площині з такими координатами (Рис. 10):

A (1, 3); B(4, 8); C(9, 8); D(10, 3); E(6, 1)

Довжина ребер: AB=5,83; BC=5,00; CD=5,10; DE=4,47; EA=5,39.

Кластеризація: Спочатку ми поділимо площу на три трикутника (Рис. 11).

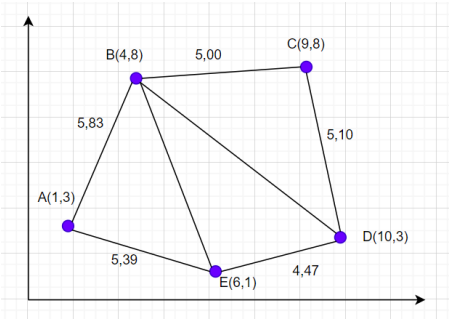

Рисунок 10 - Початкові точки графа з п’ятьох вершин

Знаходження Точок Штейнера для Кластерів: Для кожного з цих кластерів з трьох точок ми визначимо відповідну точку Штейнера за допомогою програми Steiner Point Calculator (Рис. 11).

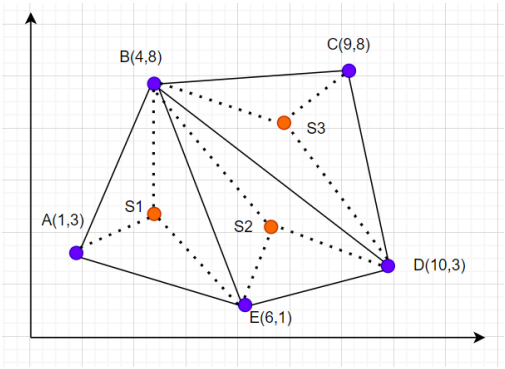

Рисунок 11 - Розділення графа з п’ятьма точками на кластери

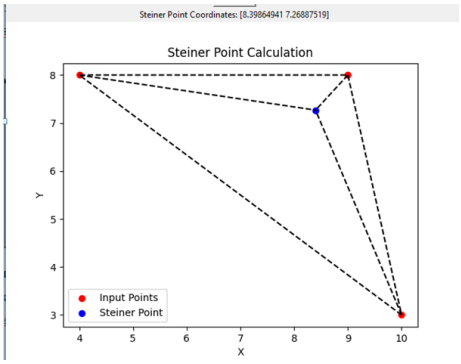

Рисунок 12 - Знаходження ТШ для трикутника

S1 (3.06, 3.66); S2(7.07, 2.93); S3(8.40, 7.27)

Об'єднання отриманих точок S1, S2, S3 в новий трикутник та знаходження точки Штейнера для нового трикутника S4(6.54, 3.78)

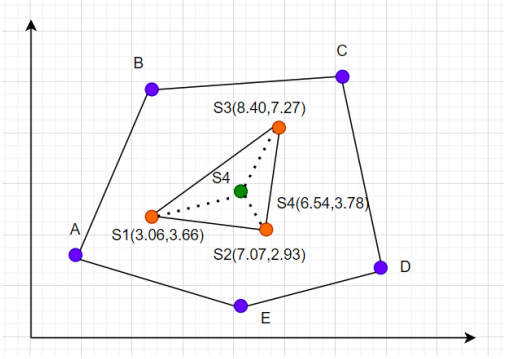

Рисунок 13 - Результуюча ТШ для графа з п’яти точок

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Нова точка в отриманому трикутнику і буде розрахунковою точкою Штейнера для початкових п’яти вершин (Рис. 14).

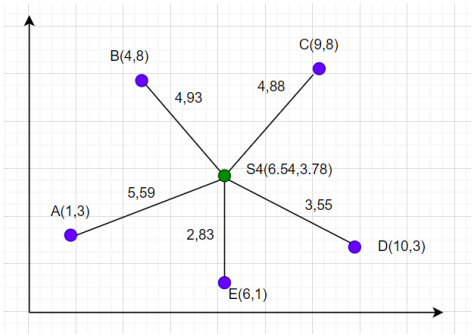

Рисунок 14 - ДШ для 5-и точок прямим з’єднанням

Для порівняння знайдемо суму всіх ребер в отриманому дереві Штейнера та мінімальним остовним деревом для п’ятьох початкових точок за допомогою програми EMST.

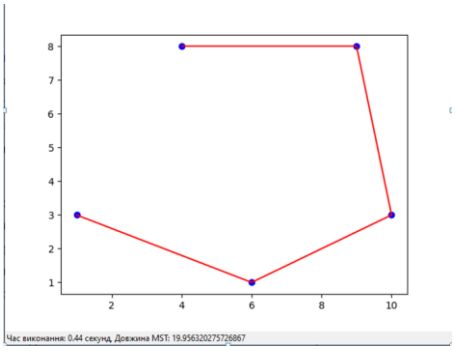

Рисунок 15 - МОД для п’яти точок за допомогою EMST

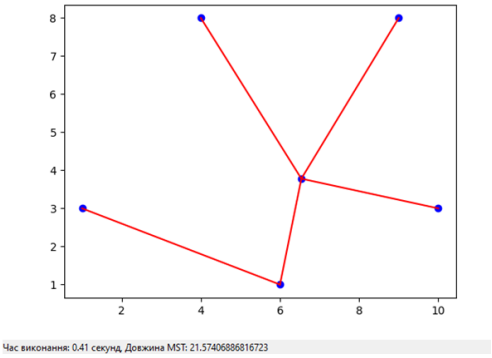

Рисунок 16 - ДШ для п’яти точок за допомогою EMST

Довжина ребер ДШ прямим з’єднанням(Рис. 14): 21.78.
(A–S4=5.59; B–S4=4.93; C–S4=4.88; D–S4=3.55; E–S4=2.83).
Довжина МОД за допомогою EMST(Рис 15): 19.9.
Довжина ДШ за допомогою програми EMST(Рис.16): 21,5.

Для шести точок

Вибір Точок: Візьмемо шість точок, розміщених випадково на площині з такими координатами:

A (1, 2); B(2, 7); C(6, 5); D(8, 3.5);E(5, 1);F(3, 3).

потім поділимо площу в середині графа на трикутники (Рис 17).

Вага ребер графа дорівнює відстані між його точками:

A-B=5.10; B-C=4.47; C-D=2.50; D-E=3.91; E-A=4.12;

A-F=2.24; B-F=4.12; C-F=3.61; D-F=5.03; E-F=2.83.

Тепер за допомогою програми Steiner Point Calculator знайдемо точки Штейнера для кожного трикутника (Рис. 18):

S1=(A; B; F) = (2.53; 3.19)

S2=(F; B; C) = (3.73; 4.72)

S3=(F; C; D) = (5.99; 4.69);

S4=(F; D; E) = (5.01; 1.81);

S5=(F; E; A) = (2.95; 2.71);

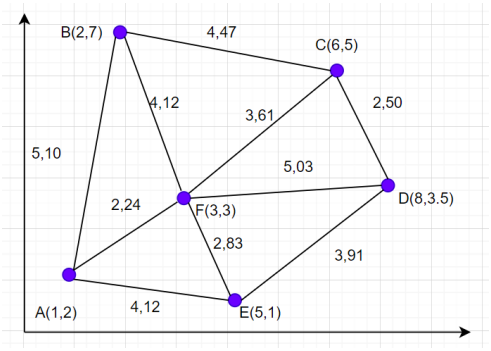

Рисунок 17 - Початкові точки графа з шести вершин

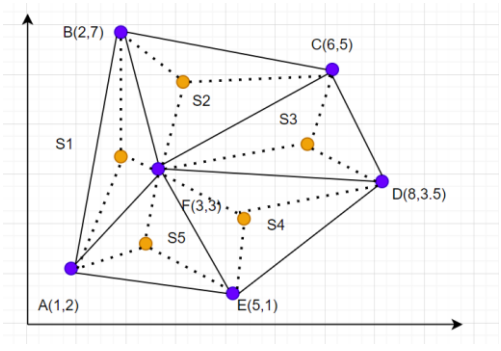

Рисунок 18 - Розділення графа з шести точками на кластери

Далі розділяємо площу графа з п’яти точок на три трикутника та знаходимо для кожного точку Штейнера H1, H2, H3 (Рис. 19).

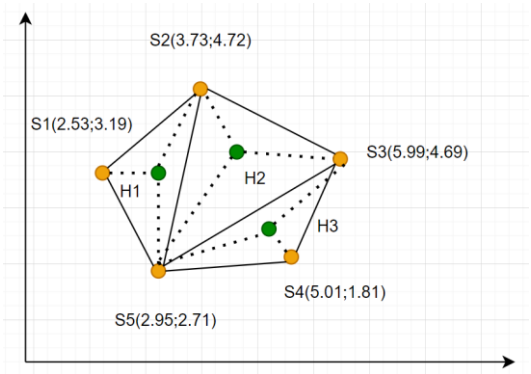

Рисунок 19 - ТШ для кластерів другої ітерації

H1(S1; S2; S5)=(2.71;3.18)

H2(S2; S3; S5)=(3.85; 4 .54)

H3(S3; S4; S5)=(4.61; 2.61)

Точка Штейнера (Рис. 21) для трикутника H1-H2-H3 буде передбачуваною точкою Штейнера (T) для шести початкових вершин графа T(3.63; 3.48).

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

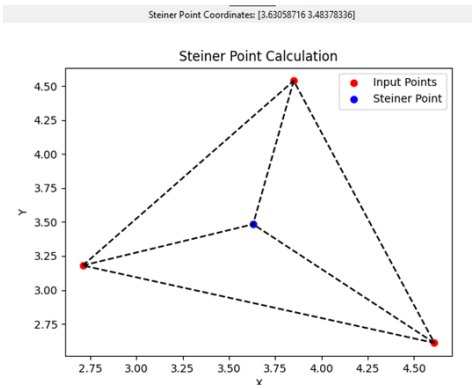

Рисунок 20 - Результуюча ТШ для графа з шести точок

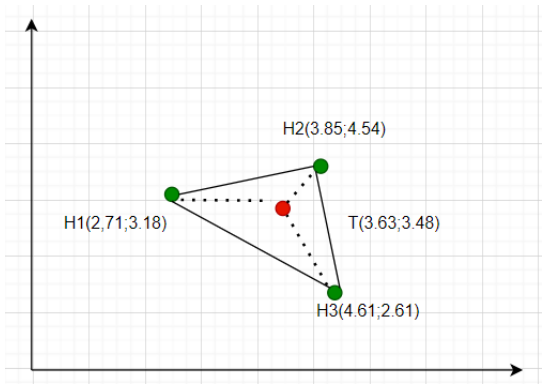

Рисунок 21 - Результуюча ТШ для графа з шести вершин

Для порівняння з МОД знайдемо суму всіх ребер отриманого дерева Штейнера за формулою:

A-T = 3.02; B-T = 3.88; C-T = 2.82; D-T = 4.37; E-T = 2.83; F-T = 0.79.

Сума дорівнює 17,71

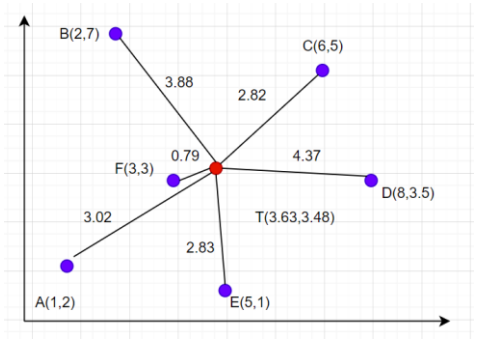

Рисунок 22 - ДШ для графа з шести точок прямим з’єднанням

Знайдемо МОД початкових шести точок за допомогою програми EMST

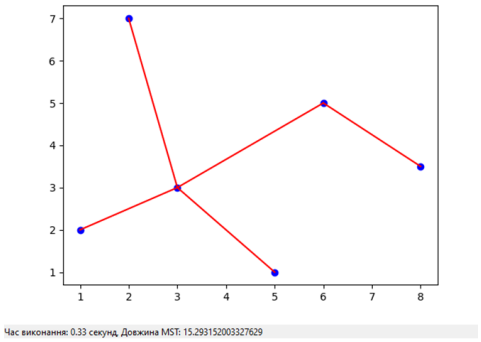

Рисунок 23 - МОД для графа з шести точок за допомогою EMS

Довжина МОД дорівнює 20,23.

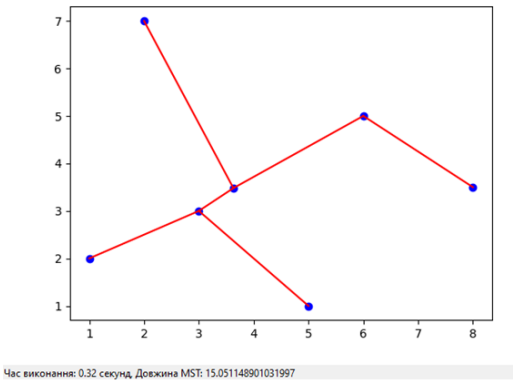

Рисунок 24 - ДШ для графа з шести точок за допомогою EMST

Довжина ребер ДШ прямим з'єднанням(Рис. 22): 17,71.

Довжина МОД за допомогою EMST(Рис. 23): 15,29.

Довжина ДШ за допомогою програми EMST(Рис. 24): 15,05.

Результати дослідження: При розробці вдосконаленого алгоритму знаходження точок Штейнера були проведені дослідження мінімальних остовних дерев та дерев Штейнера побудованих різними методами на графах з різною кількістю початкових вершин.

Чотири точки:

За методом прямого з'єднання та EMST отримано однакову довжину ребер (20,98), свідчачи про однакову ефективність обох методів у цьому конкретному випадку.

Метод МОД за допомогою EMST показав дещо вищу довжину ребер (21,28), що може вказувати на його меншу ефективність для невеликої кількості точок.

П'ять точок:

Пряме з'єднання дало найбільшу довжину ребер (21.78), що свідчить про його відносно меншу ефективність у цьому сценарії.

Метод EMST з деревом Штейнера показав кращі результати (21,5) порівняно з прямим з'єднанням.

Найкращі результати показав метод МОД за допомогою EMST (19.9), вказуючи на його вищу ефективність для цієї кількості точок.

Шість точок:

Пряме з'єднання знову показало найбільшу довжину ребер (17,71).

Метод EMST з деревом Штейнера виявився найефективнішим, демонструючи найменшу довжину ребер (15,05).

Метод МОД за допомогою EMST показав трохи вищу довжину ребер (15,29) порівняно з методом EMST з деревом Штейнера, але все ще кращі результати, ніж пряме з'єднання.

1. Результати вказують на необхідність адаптації методів залежно від конкретних сценаріїв та кількості точок для досягнення оптимальних результатів.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

2. У випадку з чотирма точками, методи прямого з'єднання ДШ та ДШ за допомогою програми EMST показали ідентичні результати, тоді як МОД за допомогою EMST був менш ефективним.

3. Для п'яти та шести точок метод МОД за допомогою EMST виявився більш ефективним, ніж інші методи, особливо порівняно з прямим з'єднанням.

Висновки:

1. Дослідження ефективності кластеризації в визначенні точок Штейнера показало, що використання цього методу може бути ефективним у зниженні обчислювальної складності, особливо у сценаріях з великою кількістю точок. Цей метод демонструє адекватну точність у побудові дерева Штейнера, що робить його корисним для оптимізації процесу розв'язання задач.

2. Детальне дослідження, виконане на графах з різною кількістю точок, вказує на те, що різні методи знаходження оптимальних рішень виявляються більш або менш ефективними залежно від конкретної кількості точок. Цей аналіз підкреслює необхідність адаптації алгоритму до специфіки задачі та підтверджує його універсальність і гнучкість.

3. Розроблений експериментальний зразок програми реалізації алгоритму дозволяє знаходити точку Штейнера для графа з трьох точок.

4. Порівняльний аналіз різних методів (пряме з'єднання, ДШ за допомогою програми EMST, МОД за допомогою EMST) показав, що ефективність кожного методу значно відрізняється залежно від кількості точок. Наприклад, виявилося, що для п’яти точок метод МОД за допомогою EMST є найбільш ефективним, тоді як для чотирьох точок методи прямого з'єднання та EMST показали ідентичні результати.

Базуючись на отриманих результатах, важливо провести подальші дослідження з використанням більшої кількості точок і різних конфігурацій графів. Це допоможе більш детально зрозуміти масштаби застосовності розробленого алгоритму та його ефективність у широкому спектрі задач.

Отримані результати можуть бути корисними в практичних застосуваннях, де необхідна оптимізація мережевих структур, наприклад, у телекомунікаціях або при плануванні транспортних мереж. Адаптація алгоритму до специфічних умов цих застосувань може значно підвищити їх ефективність.

ЛІТЕРАТУРА

1. Melzak, Z. A. Companion to Concrete Mathematics. John Wiley & Sons, Inc., 1973.
2. Winter, Pawel. Steiner Problem in Networks: A Survey. Networks, 17(2), 129–167.
3. The Shortesr-Network Problem
URL: https://mathweb.ucsd.edu/~ronspubs/89_01_shortest_network.pdf
4. Ольшевський А.І. Алгоритми побудови маршруту при груповому розсиланні мережевих пакетів даних дистанційного навчання на базі ДонНТУ. Штучний інтелект, № 4, 2007, 483-490.
5. Wirsansky, E. Hands-On Genetic Algorithms with Python. Packt Publishing, 2020, 346 p.
6. Литвиненко В.А., Хованскоѵ С.А., Максюта Д.Ю. Адаптивний алгоритм побудови дерева Штейнера, КПУ, 2016, 9 с.

7. Бібліотека алгоритмів обчислювальної геометрії
URL: https://www.cgal.org/.
8. Kruskal-based approximation algorithm for the multi-level Steiner tree problem. URL: https://arxiv.org/pdf/2002.06421v2.pdf

REFERENCES

1. Melzak, Z. A. Companion to Concrete Mathematics. John Wiley & Sons, Inc., 1973.
2. Winter, Pawel. Steiner Problem in Networks: A Survey. Networks, 17(2), 129–167.
3. The Shortest-Network Problem.
URL: https://mathweb.ucsd.edu/~ronspubs/89_01_shortest_network.pdf
4. Olshevskyi, A.I. Algorithms for Routing in Group Distribution of Network Data Packets for Distance Learning Based on DonNTU. Artificial Intelligence, No. 4, 2007, 483-490.
5. Wirsansky, E. Hands-On Genetic Algorithms with Python. Packt Publishing, 2020, 346 p.
6. Lytvynenko, V.A., Khovanskov, S.A., Maksyuta, D.Yu. Adaptive Algorithm for Building a Steiner Tree, KPU, 2016, 9 p.
7. Library of Computational Geometry Algorithms. URL: https://www.cgal.org/
8. Kruskal-based Approximation Algorithm for the Multi-Level Steiner Tree Problem. URL: https://arxiv.org/pdf/2002.06421v2.pdf

Received 01.08.2024.
Accepted 04.09.2024.

Construction of a Steiner Tree Using the Clustering Method

This paper examines the method of constructing a Steiner tree for optimizing network structures in distributed computer systems. The primary goal of the work is to investigate and implement an advanced algorithm for finding Steiner points using the clustering method. The main idea of the method is to use a specific approach to determining Steiner points that optimize the connection of given points in space. The objective of this approach is to reduce computational complexity while maintaining adequate accuracy in constructing the Steiner tree. Due to the simplified approach to clustering and determining Steiner points, this method has the potential to significantly optimize the problem-solving process, especially in scenarios with a large number of points. To determine its effectiveness, studies were conducted on graphs with four, five, and six vertices randomly located on a plane. Testing was carried out using special software written in Python. Overall, the research showed that the clustering method is an effective tool for determining Steiner points, allowing for reduced computational complexity and providing adequate accuracy in constructing the Steiner tree. Further research in this direction may contribute to the improvement of network structure optimization methods, which is important for a wide range of practical applications.
Keywords: Steiner tree, minimum spanning tree, optimization method, graph clustering.

Глушков Олег Володимирович - аспірант кафедри електронних обчислювальних машин Українського державного університету науки і технологій (м. Дніпро).

Hlushkov Oleh - Postgraduate Student, Department of Electronic Computing Machines, Ukrainian State University of Science and Technology, Dnipro, Ukraine.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)