МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської підготовки

Кафедра Інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Дослідження методів та інструментальних засобів інтелектуального аналізу геоданих»

Виконав студент 2 курсу групи МІС-19
спеціальності 122 Комп'ютерні науки

Киріяк Дмитро Євгенович

_____

Керівник к.геог.н., доц.
Кузніченко Світлана Дмитрівна

Рецензент к.т.н., доц.
Гнатовська Ганна Арнольдівна

Одеса 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет Магістерської підготовки

Кафедра інформаційних технологій

Рівень вищої освіти магістр

напрямок 122 Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри _____

_____

" 26 " жовтня 2020 р.

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Кириляку Дмитру Євгеновичу

(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів та інструментальних засобів інтелектуального аналізу геоданих»

керівник роботи к.геогр.н., доцент Кузніченко Світлана Дмитрівна

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу від "16" жовтня 2020 р. №194-С

2. Строк подання студентом проекту 7 грудня 2020 р.

3. Вихідні дані до роботи бібліотеки ГІС-аналізу та просторового моделювання пакету ESRI ArcGIS, векторні карти м. Одеса, імпортовані з картографічного веб-сервісу OpenStreetMap, мова програмування Python.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Графова модель транспортної мережі

1.1 Зв'язок теорії графів з геометрією транспортної мережі

1.2 Алгоритми пошуку характеристик графів

2. Методи та інструментальні засоби ГІС-аналізу транспортних мереж

2.1 Огляд інструментальні засоби ГІС для аналізу транспортних мереж

2.3 Огляд бібліотек Python для обробки просторової інформації

3. Створення програмних інструментів геообробки для розрахунку характеристик транспортних мереж

3.1 Побудова графа дорожньої мережі в ГІС

3.2 Побудова та аналіз зон доступності

3.3 Розробка скриптів Python для автоматизованої побудови маршрутів

5. Перелік графічного матеріалу

1 Векторна карта графу дорожньої мережі м. Одеса

2 Блок-схеми створених скриптів Python для автоматизованої побудови графу та маршрутів

3 Векторні карти доступності соціально-значущих об’єктів міста

6. Консультанти розділів проекту

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

7. Дата видачі завдання “ 26 ” жовтня 2020 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту	Термін виконання етапів проекту	Оцінка виконання етапу	
			у %	за 4-х бальною шкалою
1	Дослідження літературних джерел за темою кваліфікаційної роботи	26.10	80	добре
2	Дослідження графової моделі дорожньої мережі	02.11	80	добре
3	Аналіз алгоритми знаходження характеристик графа дорожньої мережі	7.11	80	добре
4	Аналіз функціональних можливостей вбудованих засобів ГІС для моделювання дорожньої мережі міста	10.11	80	добре
5	Обґрунтування вибору програмних засобів розробки інструментів геообробки	15.11	80	добре
6	Рубіжна атестація	19.11	80	добре
7	Створення інструментів Python для автоматизованої побудови маршрутів та графа дорожньої мережі	20.11	80	добре
8	Виконання ГІС-аналізу та просторового моделювання дорожньої мережі (на прикладі міста Одеса)	25.11	80	добре
9	Аналіз результатів дослідження. Формування висновків	01.12	80	добре
	Подання роботи на кафедру	07.12		
	Перевірка на плагіат	08.12		
	Рецензування	16.12		
	Інтегральна оцінка виконання етапів календарного плану (як середня по етапам)		**80**	**добре**

Студент _____ Кіріяк Д.Є.
(підпис) (прізвище та ініціали)

Керівник проекту _____ Кузніченко С.Д.
(підпис) (прізвище а ініціали)

АНОТАЦІЯ

на магістерську роботу «Дослідження методів та інструментальних засобів інтелектуального аналізу геоданих»,
студента Киріяка Дмитра Євгеновича

Актуальність роботи полягає в необхідності розробки алгоритмів та програмних засобів ГІС для проведення інтелектуального аналізу даних для вирішення управлінських завдань організації транспортної системи міста.

Мета роботи – дослідження методів та інструментальних засобів інтелектуального аналізу геоданих на прикладі вирішення завдань організації транспортної системи міста.

Задачі дослідження: виконати дослідження графової моделі дорожньої мережі та алгоритмів знаходження характеристик графа; виконати аналіз функціональних можливостей вбудованих засобів ГІС для моделювання дорожньої мережі міста; розробити власні інструменти автоматизованої побудови маршрутів та графа дорожньої мережі; провести ГІС-аналізу та просторове моделювання дорожньої мережі (на прикладі міста Одеса).

Об'єкт дослідження – граф транспортної мережі міста .

Предмет дослідження – методи та алгоритми ГІС-аналізу графа транспортної мережі міста.

Методи дослідження: теорія графів, ГІС-аналіз та просторове моделювання, методи збору та обробки геоданих, об'єктно-орієнтоване програмування.

Результати, їх новизна, теоретичне та практичне значення: розроблено програмне забезпечення для автоматизованої побудови графа дорожньої мережі та маршрутів з урахуванням бар'єрів.

Структура магістерської роботи складається з вступу, трьох розділів, висновків, переліку посилань на 20 найменувань, додатків. Повний обсяг роботи становить 80 сторінок, містить 39 рисунків, 1 таблиця і 12 формул.

КЛЮЧОВІ СЛОВА: геоінформаційна система, інтелектуальний аналіз, геодані, скрипт Python, граф, транспортна мережа, транспортна доступність.

ABSTRACT

for the master's thesis " Research into Methods and Tools of Intelligent Analysis of Geodata",

Kiriak Dmitro

The relevance of the study lies in the need to develop GIS methods and software for intelligent data analysis to solve management problems of organizing the city's transport system.

The purpose of the research is to study the methods and tools for the intellectual analysis of geodata using the example of solving the problems of organizing the city's transport system.

Research objectives: to carry out research of the graph model of the road network and algorithms for finding the characteristics of the graph; analyze the functionality of the built-in GIS tools for modeling the city road network; develop your own automated tools for building routes and a road network graph; carry out GIS analysis and spatial modeling of the road network (for example, we-hundred Odessa).

The object of research is the graph of the city's transport network.

The subject of the research is methods and algorithms for GIS analysis of the city transport network graph.

Research methods: graph theory, GIS analysis and spatial modeling, methods of collecting and processing geodata, object-oriented programming.

Results, their novelty, theoretical and practical significance: developed software for the automated construction of a graph of the road network and routes taking into account barriers.

The structure of the master's work consists of an introduction, three chapters, a conclusion, a list of references to 20 titles, applications. The total volume of work is 80 pages, contains 39 figures, 1 table and 12 formulas.

KEYWORDS: geographic information system, mining, geodata, Python script, graph, transport network, transport accessibility.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ВДМ	– Вулично-дорожня мережа
ГІС	– Географічні інформаційні системи
ПЗ	– Програмне забезпечення
BI	– Business Intelligence
ESRI	– Environmental Systems Research Institute
GRASS	– Geographic Resources Analysis Support System
GIS	– Geographic Information System
OGD	– Open Government Data
OSM	– Open Street Map
VGI	– Volunteered Geographic Information

ВСТУП

Інтелектуальний аналіз геоданих – це область знань, що швидко розвивається в результаті стрімкого зростання збору просторової інформації. Таке зростання стало можливим завдяки різним застосуванням, таким як: дистанційне зондування Землі, геоінформаційні системи, стандарти обміну просторовими даними через Інтернет і геолокаційні сервіси. Цінні знання можна отримати за допомогою сучасних методів інтелектуального аналізу геоданих. Отримані знання використовуються для підтримки прийняття рішень на основі просторової інформації. Оскільки прийняття рішень на основі даних стає все більш і більш важливим, а велика частина даних включає просторові компоненти, використання алгоритмів обробки геоданих стає важливою частиною сучасного інтелектуального аналізу даних.

В наш час одним з найбільш затребуваним напрямком використання інтелектуального аналізу геоданих є вирішення завдань організації транспортних мереж. Функціонування транспортної мережі, особливо на території великих населених пунктів потребує оптимізації певних параметрів таких як, наприклад, маршрути перевезення, розташування зупинок, зони транспортної доступності, обслуговування та ін. Безумовно, кращим інструментом для аналізу та моделювання просторової інформації про характеристики транспортних мереж є геоінформаційні системи (ГІС).

Метою даної магістерської роботи є дослідження методів та інструментальних засобів інтелектуального аналізу геоданих на прикладі вирішення завдань організації транспортної системи міста.

Для досягнення поставленої мети дослідження, треба вирішити наступні *завдання*:

– виконати дослідження графової моделі дорожньої мережі;
– описати алгоритми знаходження характеристик графа дорожньої мережі;

- виконати аналіз функціональних можливостей вбудованих засобів ГІС для моделювання дорожньої мережі міста;
- виконати вибір програмних засобів розробки;
- провести ГІС-аналізу та просторове моделювання дорожньої мережі (на прикладі міста Одеса) вбудованими засобами ГІС;
- розробити власні інструменти автоматизованої побудови маршрутів та графа дорожньої мережі;
- здійснити тестування створеного програмного інструменту в ГІС для вирішення різних завдань побудови оптимальних маршрутів.

1 ГРАФОВА МОДЕЛЬ ТРАНСПОРТНОЙ МЕРЕЖІ

1.1 Поняття транспортної мережі

Система – це сукупність елементів, які знаходяться у взаємозв'язку і взаємовідношенні між собою, яка утворює певну єдність, цілісність [1][1)].

Під транспортною системою в найбільш загальному випадку розуміють сукупність елементів транспортної інфраструктури та інфраструктури суб'єктів перевезень, включаючи систему управління, спрямовану на ефективне переміщення вантажів і пасажирів [2, 3][2)].

Під транспортною інфраструктурою розуміють – фізичні компоненти транспортної системи, які займають фіксоване положення в просторі і утворюють транспортну мережу, яка включає зв'язки і вузли [4][3)].

Основними завданнями транспортної системи є задоволення потреб економіки в перевезеннях вантажів і забезпечення мобільності населення. У зв'язку з цим ефективність транспортної системи завжди буде визначатися балансом між суперечливими вимогами економіки і суспільства. Прикладом є бажання пасажира, щоб транспорт під'їхав до зупинки, як тільки пасажир підійшов до неї, і бажання перевізника встановити такий інтервал руху, щоб транспортні засоби завжди були заповнені повністю і приносили максимальний дохід [5][4)].

---

1) [1] Горев, А. Э. Основы теории транспортных систем: учеб. пособие. СПбГАСУ. СПб., 2010. 214 с.

2) [2] Доля К. В., Доля О.Є. Геоінформаційні системи на транспорті: навч. посібник; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. Харків: ХНУМГ ім. О. М. Бекетова, 2018. 230 с.

[3] Якубович А.Н., Куфтинова Н.Г., Рогова О.Б. Информационные технологии на автотранспорте: учебное пособие. М.: МАДИ, 2017. 252 с.

3) [4] Трофименко, Ю.В. Якимов М. Р. Транспортное планирование: формирование эффективных транспортных систем крупных городов: монография. М.: Логос, 2013. 447 с.

4) [5] Ваксман, С.А. Информационные технологии в управлении городским общественным пассажирским транспортом (задачи, опыт, проблемы). Екатеринбург: Изд-во АМБ, 2012. 250 с.

1.2 Зв'язок теорії графів з геометрією транспортної мережі

Основні властивості транспортних мереж визначаються їх морфологічними характеристиками (характеристики форми і будови мережі). Для визначення морфологічних характеристик транспортна мережа представляється у вигляді графа. Прості конфігураційні частини мережі називаються структурними елементами (включають замкнуті контури – цикли і лінійні елементи – гілки), а складні – структурними компонентами.

Граф – це фігура, яка складається з точок (вершин) і відрізків (ланок), що з'єднують їх. Вершина графа – це точки на мережі, найбільш важливі для визначення відстаней або маршрутів руху автомобілів. Ланки графа – це відрізки транспортної мережі, які характеризують наявність дорожньої зв'язку між сусідніми вершинами [6, 7][1)].

Наведемо теоретико- множинне визначення графа. Нехай V – непорожня множина, наприклад $\{v_1, v_2, v_3, v_4, v_5\}$. $V^{(2)}$ – множина всіх його двоелементних підмножин. Пара (V, E), де E – довільна підмножина множини $V^{(2)}$, називається графом (неорієнтованим графом). Елементи множини V називаються вершинами графа, а елементи множини E – ребрами. Отже, граф – це кінцева множина V вершин і множина E ребер, $E \subset V^{(2)}$.

В більш компактній формі це визначення звичайно формулюється так: пара $\langle V, E \rangle$ називається неорієнтованим графом, якщо V – непуста множина елементів, що називається вершинами, а E – множина неупорядкованих пар різних елементів з V, що називається ребрами.

При записі різних співвідношень в теорії графів використовують позначення VG або $V(G)$ для множини вершин та EG або $E(G)$ для множини ребер графа G.

[1)] [6] Кузьменко І.М. Теорія графів. Навч. посіб. для здобувачів ступеня бакалавра за спеціальністю 122 «Комп'ютерні науки». Київ: КПІ ім. Ігоря Сікорського, 2020. 71 с.

[7] О. Оре. Теория графов. М.: Наука, 1980, 336 с.

Наочним способом представлення графа є рисунок (діаграма), на якому вершини зображуються точками, а ребра – лініями, що з'єднують зображення вершин реберної пари. На рис.1.1 наведені приклади представлення графа.

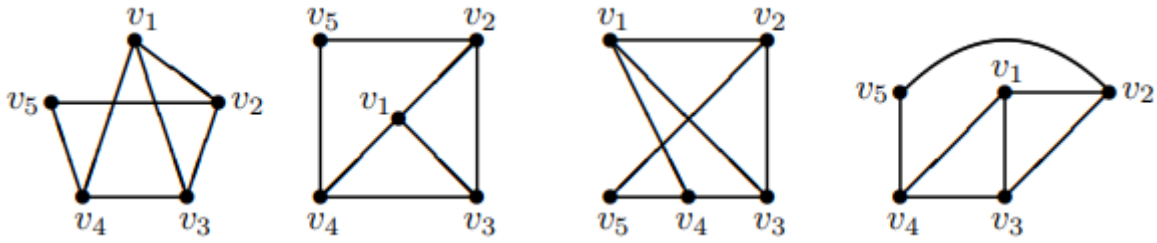


Рисунок 1.1 – Приклади представлення графа

В свою чергу графи діляться на три види. Орієнтований граф – граф, ребрам якого присвоєний напрямок. Спрямовані ребра йменуються також дугами, а в деяких джерелах ребрами. Неорієнтований граф – граф, ребра якого не мають напрямку. Змішаний граф – якщо у частини ланок графа є напрямок [6][1)].

Дві вершини, що утворюють ребро $\{v_i, v_j\}$, називають його кінцями, а про ребро говорять, що воно з'єднує v_i і v_j. Говорять також, що вершини v_i і v_j суміжні. Суміжними називають і ребра з загальною вершиною. Про ребро і вершину, яка є його кінцем, говорять що вони інцидентні. Так, наприклад, в графі на рис.1.1 вершини v_1 і v_2 суміжні, а вершини v_3 і v_5 не суміжні. Ребра $\{v_1, v_2\}$ і $\{v_1, v_3\}$, суміжні, а ребра $\{v_4, v_5\}$ і $\{v_2, v_3\}$ не суміжні. Вершина v_3 і ребро $\{v_2, v_3\}$ інцидентні.

Число ребер, інцидентних вершині v, визначає степінь вершини, яка позначається deg v. Так в графі на рис.1.1 deg v_1= deg v_2= deg v_3= deg v_4=3, а deg v_5=2. Вершина v називається ізольованою, якщо deg v=0, і кінцевою, якщо deg v=1. Ребро, що інцидент не кінцевій вершині, також називають кінцевим. Список степенів всіх вершин називають степеневою послідовністю графа.

[1)] [6] Кузьменко І.М. Теорія графів. Навч. посіб. для здобувачів ступеня бакалавра за спеціальністю 122 «Комп'ютерні науки». Київ: КПІ ім. Ігоря Сікорського, 2020. 71 с.

Множина вершин, суміжних з вершиною *v*, позначають як adj *v*. Наприклад, в графі на рис.1.1 adj $v_1=\{v_2, v_3, v_4\}$.

Важливими кількісними характеристиками графа є: число вершин $n=|V|$, яке визначає порядок графа, та число ребер $m=|E|$. Граф с *n* вершинами і *m* ребрами називається *(n,m)* – графом. Сума степенів вершин *(n,m)* – графа дорівнює подвоєному числу його ребер: $\sum_{i=1}^{n} \deg v_i = 2|E| = 2m.$

Якщо пара вершин з'єднується кількома ребрами або дугами одного напрямку, то ребра (дуги) називають кратними (паралельними). Дуга або ребро, яке з'єднує вершину саму з собою називається петлею. Граф без кратних дуг і петель називається простим. Вершини, що з'єднані ребром або дугою називають суміжними, також називають суміжними ребра, які мають загальну вершину.

Граф G називається повним, якщо будь-які дві його вершини суміжні.

Граф називається виродженим (порожнім), якщо будь-які дві його вершини не суміжні (тобто у нього немає ребер) [6][1)].

Транспортна мережа за допомогою графа відображається наступним чином: в якості вершин графа беруться населені пункти (або ж вузли – перетину лінійних зв'язків), в якості ребер – зв'язки між цими населеними пунктами (вузлами) (наприклад, дороги).

1.3 Чисельні характеристики графів

Розглянемо деякі чисельні характеристики графів, які є важливими при аналізі транспортних мереж.

Центром графа називається така вершина в графі, сума найкоротших відстаней від якої до всіх інших вершин найменша.

[1)] [6] Кузьменко І.М. Теорія графів. Навч. посіб. для здобувачів ступеня бакалавра за спеціальністю 122 «Комп'ютерні науки». Київ: КПІ ім. Ігоря Сікорського, 2020. 71 с.

Діаметром графа називається найбільше найкоротша відстань між усіма парами вершин.

Числом Кеніга вершини А називається максимальне з найкоротших відстаней між верішной А і всіма іншими вершинами графа.

Розглянемо приклад знаходження радіуса, діаметра і центру для графу на рис.1.2.

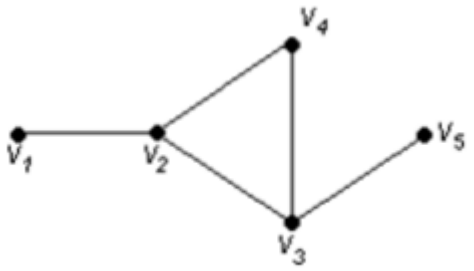


Рисунок 1.2 – Граф *G*

Щоб визначити центри, радіус, діаметр графа *G*, зображеного на рис.1.2, знайдемо матрицю відстаней між вершинами графа, елементами d_{ij} якої будуть відстані між вершинами v_i і v_j. Для цього необхідно скористатися графічним представленням графа. Зауважимо, що матриця $D(G)$ симетрична щодо головної діагоналі:

$$D(G) = \begin{bmatrix} 0 & 1 & 2 & 2 & 3 \\ 1 & 0 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 2 \\ 3 & 2 & 1 & 2 & 0 \end{bmatrix} \quad (1)$$

За допомогою отриманої матриці для кожної вершини графа *G* визначимо найбільше видалення з виразу:

$$r(v_i) = \max_j d(v_i, v_j) \quad (2)$$

для $i, j = 1, 2, ..., 5$. В результаті отримуємо: $r(v_1) = 3$, $r(v_2) = 2$, $r(v_3) = 2$, $r(v_4) = 2$, $r(v_5) = 3$. Мінімальне з отриманих чисел є радіусом графа G, максимальне – діаметром графа G. Значить, $R(G) = 2$, $D(G) = 3$, центром є вершини v_2, v_3, v_4.

Граф називається зваженим або навантаженим, якщо кожному ребру поставлено відповідно деяке число w (вага) [8][1)].

Матрицею ваг називається квадратна матриця розмірності $p'p$ (p – кількість вершин), елемент якої, що стоїть в i-ому рядку і j-ому стовпці визначається за правилом:

$$a_{ij} = \begin{cases} 0, & \text{якщо вершини } v_i \text{ і } v_j \text{ співпадають}, \\ \infty, & \text{якщо вершини } v_i \text{ і } v_j \text{ не суміжні}, \\ w, & \text{якщо вершини } v_i \text{ і } v_j \text{ не суміжні, і } w - \text{вага ребра } (v_i, v_j) \end{cases}$$

1.4 Алгоритми пошуку характеристик графів

Розглянемо алгоритм прискореного пошуку центру і радіусу: нехай дана матриця найкоротших відстаней $M = (m_{ij})$ зваженого неорієнтованого зв'язаного графа з невід'ємними вагами ребер.

Крок 1. Підготовка. Шукається пара вершин (x, y), кожна з яких є найвіддаленішої для іншої:

$$m_{x,y} = \max_{i=1,n} m_{iy} = \max_{i=1,n} m_{ix}. \qquad (3)$$

Для цього вибирається будь-яка вершина графа p_1, далі шукається вершина p_2, для якої $m_{p_1 p_2} = \max_{i=1,n} m_{iy} = \max_{i=1,n} m_{ix}$.

[1)] [8] Ураков А.Р., Тимеряев Т.В. Использование особенностей взвешенных графов для более быстрого определения их характеристик. Прикладная Дискретная Математика. 2012, №2 (16), стр.95-100.

Пошук вершин p_j, j = 2, 3, ..., триває до тих пір, поки не буде виконано рівність $p_{j-1} = p_{j+1}$, тоді $x = p_j$, $y = p_{j+1}$.

Крок 2. Пошук. Розглядаються всі вершини, крім x, y. Претендент на центр c шукається як вершина, максимум видалення якої від пари (x, y) мінімальний:

$$\max(m_{cx}, m_{cy}) = \min_{i=1,n}\left(\max\left(m_{ix}, m_{iy}\right)\right). \quad (4)$$

Після знаходження c вершина $r = max\ (m_{cx}, m_{cy})$ є першим наближенням радіуса графа.

Крок 3. Перевірка. Шукається периферійна вершина z, така, що

$$m_{zc} = \max_{i=1,n} m_{ic}. \quad (5)$$

Якщо $m_{zc} = r$, то, згідно з твердженням 1, r – радіус, c – один з центрів графа. Якщо $m_{zc} > r$, радіус R графа лежить в межах $r \leqslant R \leqslant m_{zc}$, переходимо до кроку 4.

Крок 4. Спроба заміни вершин x і y. Проводиться спроба прискорення пошуку. Шукається вершина t, для якої $m_{zt} > m_{xy}$. Якщо така вершина знайдена, то вважаємо $x = z$, $y = t$ і переходимо до кроку 2. Інакше переходимо до кроку 5.

Крок 5. Пошук нового претендента на центр. Серед нерозглянутих претендентів на центр знаходиться вершина d, для якої:

$$\max\left(m_{dx}, m_{dy}, m_{dz}\right) = \min_{i=1,n}\left(\max\left(m_{ix}, m_{iy}, m_{iz}\right)\right). \quad (6)$$

Якщо максимальна відстань від цієї вершини до інших менше, ніж поточна верхня межа радіусу $\left(\max\limits_{i=1,n} m_{di} < m_{zc}\right)$, то d – новий претендент на центр; вважаємо $c = d$, переходимо до кроку 3. Якщо $max\ m_{di} = m_{zc}$, то c – центр, r – радіус. Якщо $\max\limits_{i=1,n} m_{di} = m_{zc}$, то c – центр, r – радіус. Якщо $\max\limits_{i=1,n} m_{di} > m_{zc}$, то помічаємо вершину d як розглянуту і знову виконуємо крок 5.

Твердження 1. Нехай дано зважений неорієнтовний зв'язний граф з n вершинами та невід'ємними вагами ребер, x і y – дві його довільні вершини. Вершина c визначається по (4) та вершина z визначається по (5). Тогда, якщо $m_{zc} = r = \max(m_{cx}, m_{cy})$, то r – радіус, а c – один з центрів графа.

Розглянемо алгоритм швидкого пошуку діаметра. Кроки 1 і 2 алгоритму збігаються з відповідними кроками алгоритму пошуку центру і радіусу.

Вхідні дані алгоритму: матриця найкоротших відстаней $M = (m_{ij})$ зваженого неорієнтованого зв'язаного графа з невід'ємними вагами ребер.

Крок 1. Підготовка. Шукається пара вершин (x, y), що задовольняє (3). Величина $d = m_{xy}$ при цьому є поточним претендентом на діаметр.

Крок 2. Пошук опорної вершини. Шукається вершина c, яка задовольняє (4).

Крок 3. Пошук діаметра. Виконується пошук діаметра в стовбцях матриці, відповідних тільки тим вершинам i, для яких виконано нерівність $m_{ci} > d/2$.

Твердження 2. Нехай дано зважений неорієнтовний зв'язний граф з невід'ємними вагами ребер, вершини x, y визначаються по (3), вершина c визначається по (4) і $d = m_{xy}$. Тоді діаметр графа дорівнює d або знаходиться в стовбцях матриці найкоротших відстаней тих вершин i, для яких виконано нерівність $m_{ci} > d/2$.

Приклад. Нехай дано зважений граф G, який представлено на рис.1.3.

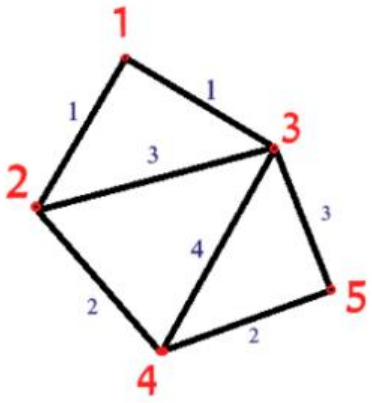


Рисунок 1.3 – Граф *G*

Складаємо матрицю найкоротших відстаней:

m_{ij}	1	2	3	4	5
1	0	1	1	3	4
2	1	0	3	2	4
3	1	3	0	4	3
4	3	2	4	0	2
5	4	4	3	2	0

Крок 1. Знаходимо пару вершин (*x*, *y*), кожна з яких є найбільш віддаленої для іншої:

$x = 1, y = 5$

Крок 2. Пошук. Розглянемо всі вершини, крім *x* і *y*: 2, 3, 4.

Претендент на центр *c*:

$$\left.\begin{array}{l} i = 2\ max\ (1,4) = 4 \\ i = 3\ max\ (1,3) = 3 \\ i = 4\ max\ (3,2) = 3 \end{array}\right| \Rightarrow \min = 3;$$

c: $\max(m_{cx}, c_{cy}) = 3$. Нехай $c = 3$.

Наближений радіус графа:

$r = \max(m_{cx}, m_{cy}) = 3$.

Крок 3. Перевірка. Найдемо периферійну вершина z

$$m_{zc} = \max_{i=1,n} m_{ic}. \quad (7)$$

max (1, 3, 0, 4, 3) = 4;

$z = m_{zc} > r = 3$, звідси, радіус R графа лежить в межах $3 \leqslant R \leqslant 4$.

Крок 4. Спроба заміни вершин x і y. Знайдемо t: $m_{zt} > m_{xy} = 4$, такого t немає, переходимо до кроку 5.

Крок 5. Пошук нового претендента на центр. Серед нерозглянутих претендентів на центр знайдемо таку вершину d, для якої

$$\max\left(m_{dx}, m_{dy}, m_{dz}\right) = \min_{i=1,n}\left(\max\left(m_{ix}, m_{iy}, m_{iz}\right)\right).$$

x = 1, y = 5, z = 4

$$\left.\begin{array}{l} i = 1\ max\ (0,\ 3,\ 4) = 4 \\ i = 2\ max\ (1,\ 2,\ 4) = 4 \\ i = 3\ max\ (1,\ 4,\ 3) = 3 \\ i = 4\ max\ (3,\ 0,\ 2) = 3 \\ i = 5\ max\ (4,\ 2,\ 0) = 4 \end{array}\right| \Rightarrow \min = 3;$$

Знайдемо d: max (m_{dx}, m_{dy}, m_{dz}) = max (3, 2, 0) = 3, при $d = 4$, $m_{zc} = 4$, переходимо до кроку 3.

Крок 3. maxd_i = max (3, 2, 4, 0, 2) = 4, отже, c – центр, що дорівнює 4. R – радіус дорівнює 3, а d – найбільша відстань, що дорівнює 4.

1.4.1 Зовнішній і внутрішній центри графа

Розглянемо кілька завдань на графах:

1) У невеликому місті будується пожежна частина. Яким чином вибрати місце для будівництва, що б час, за яке пожежники доїдуть в найдальший від частини район міста, було мінімальним?

2) Як створити мережу ресторанів, щоб з будь-якої точки міста можна було приїхати в один з таких ресторанів менш ніж за 10 хвилин? (Відповіддю на це завдання буде мінімальне число ресторанів та місця їх розташування.)

3) Як найбільш оптимально розмістити 5 служб доставки піци, щоб максимальний час шляху кур'єра до будь-якого будинку було мінімально.

4) Де встановити мережевий хаб, щоб довжина найдовшого дрота була мінімальною?

Всі ці завдання можна звести до пошуку певної точки (або точок) в графі, де вершини відповідають районам міста (або комп'ютерів в мережі), а ребра – дороги (або дроти). У всіх цих завданнях потрібно оптимізувати "найгірший варіант", тобто, наприклад, знайти таке місце для хаба, щоб довжина найдовшого дроти була мінімальна. Такі завдання часто зустрічаються при розміщенні аварійних служб, бомбосховищ, станцій метро, автомайстерень тощо. Мета таких завдань – розмістити деякі об'єкти так, щоб мінімізувати максимальне з відстаней (або час проїзду) від цього об'єкта до якоїсь будівлі (району, комп'ютера і т. п.). Саме тому цей клас задач прийнято називати мінімаксними завданнями розміщення. Якщо ж в задачі потрібно мінімізувати максимальну суму відстаней (або сумарне час) від об'єкта і назад, то це завдання відноситься до іншого класу задач - мінісумним завданням розміщення.

Будемо завжди розглядати орієнтовані сильно зв'язані графи, якщо не обумовлено зворотного. Граф G задається двома множинами: множиною вершин V і множиною ребер E. [9][1)]

Кожному ребру відповідає деяке невід'ємне число, яке має сенс довжини цього ребра (або часу проїзду по ньому). Відстань від вершини x_i до вершини x_j позначається $d(x_i, x_j) = d_{ij}$ і дорівнює мінімальній з сум довжин ребер всіх відстаней між цими вершинами. Будемо вважати, що довжина ребра во-

[1)] [9] Кристофидес Н. Теория графов. Алгоритмический подход. М.:Мир, 1978, 432 с.

лодіє адитивністю. Це означає, що якщо поділити ребро деякою точкою, то сума довжин одержаних відрізків дорівнює довжині ребра.

Кожній вершині x_i зіставимо деяке додатне число v_i, яке має сенс важливості або пріоритету. Тоді зваженою відстанню між двома вершинами x_i і x_j будемо називати число $l_{ij} = l(x_i, x_j) := v_j d(x_i, x_j) = v_j d_{ij}$.

Зауважимо, що зважена відстань не симетрично навіть в разі неорієнтованого графа, тобто $l_{ij} \neq l_{ji}$. Точно так же визначається зважений шлях між деякою точкою e_{ij}, що лежить на ребрі (x_i, x_j) і вершиною x_k: $l(x_i, x_j) := v_k d(e_{ij}, x_k)$. Будемо говорити, що точка y_2 досяжна з точки y_1 не більш ніж за λ, якщо $l(y_1, y_2) \leq \lambda$ [9][1)] Надалі завжди будемо говорити про зважену відстань, додатково це не обумовлюючи.

Розглянемо деякий граф $G = (V, E)$. Введемо дві множини:

$$R_\lambda^0(x_i) := \{x_j \mid l_{ij} \leq \lambda, x_j \in V\},$$
$$R_\lambda^1(x_i) := \{x_j \mid l_{ji} \leq \lambda, x_j \in V\}.$$

$R_\lambda^0(x_i)$ – це множина всіх вершин, відстань від x_i до яких не більше λ, а $R_\lambda^1(x_i)$ – множина всіх вершин, відстань від яких до x_i не більше λ. Очевидно, що існують такі λ, що ці множини не порожні (звичайно, якщо в графі більше однієї вершини).

Для кожної вершини визначимо два числа:

$$s_0(x_i) := \max(l_{ij}), x_j \in V,$$
$$s_t(x_i) := \max(l_{ji}), x_j \in V.$$

$s_0(x_i)$ і $s_t(x_i)$ називають відповідно зовнішнім і внутрішнім поділом для вершини x_i.

[1)] [9] Кристофидес Н. Теория графов. Алгоритмический подход. М.: Мир, 1978, 432 с.

Нехай λ_0 – найменша λ така, що для деякої x_i $R_\lambda^0(x_i) = V$, тобто довжина шляху до будь-якої вершини графа не перевищує λ_0. Тоді $s_0(x_i) = \lambda_0$ (безпосередньо випливає з визначень $R_\lambda^0(x_i)$ і $s_0(x_i)$).

Аналогічно, λ_t – найменша λ така, що для деякої x_i $R_\lambda^1(x_i) = V$, тобто довжина шляху від будь-якої вершини графа до x_i не перевищує λ_t. Тоді $s_t(x_i) = \lambda_t$.

Варто зауважити, що якби граф не був сильно пов'язаний, то хоча б одне з чисел λ_0, λ_t дорівнювало ∞.

Вершина x^*_0 така, що $s_0(x^*_0) = \min[s_0(x_i)]$, $x_i \in V$ називається зовнішнім центром графа G. Аналогічно, вершина x^*_t така, що $s_t(x^*_t) = \min[s_t(x_i)]$, $x_i \in V$ називається внутрішнім центром.

Значення $s_0(x^*_0)$ і $s_t(x^*_t)$ називаються відповідно зовнішнім і внутрішнім радіусами графа G. Позначаються ρ_0 і ρ_t. Можна так само визначити зовнішньо-внутрішній поділ: $s_{ot}(x_i) = \max\{v_j [d(x_i, x_j) + d(x_j, x_i)]\}$, $x_j \in V$, та зовнішньо-внутрішній центр, як вершину, на якій досягається мінімум зовнішньо-внутрішнього поділу [10][1)].

Можна інтерпретувати зовнішній центр як пожежний ділянку (мінімальний час на дорогу від ділянки до місця пожежі), внутрішній як бомбосховище (людині потрібний мінімальний час на дорогу туди), зовнішньо-внутрішній як служба швидкої допомоги (мінімально час на шлях туди і назад).

Наведемо алгоритм пошуку зовнішнього центру. Для початку побудуємо матрицю $A_{n\times n}$ (n – потужність множини V), де $a_{ij} = l_{ij}$, тобто її матрицю найкоротших відстаней. Для її побудови можна скористатися алгоритмом Флойда-Уоршалла або Дейкстри. Алгоритм Флойда-Уоршалла застосовується для знаходження найкоротших відстаней між усіма вершинами зваженого орієнтованого графа: нехай вершини графа $G = (V, E)$, $|V| = n$ пронумеровані

[1)] [10] Берж К. Теория графов и ее применения. Москва: Иностранная литература, 1962. 75 с.

від 1 до *n* і введено позначення d_{ij}^{k} для довжини найкоротшого шляху від *i* до *j*, який крім самих вершин *i*,*j* проходить тільки через вершини 1...*k*. Очевидно, що d_{ij}^{0} – довжина (вага) ребра (*i*,*j*), якщо таке існує (в іншому випадку його довжина може бути позначена як ∞).

Існує два варіанти значення $d_{ij}^{k}, k \in (1,...,n)$:

1. Найкоротший шлях між *i*,*j* не проходить через вершину *k*, тоді $d_{ij}^{k} = d_{ij}^{k-1}$

2. Існує більш короткий шлях між *i*,*j*, що проходить через *k*, тоді він спочатку йде від *i* до *k*, а потім від *k* до *j*. В цьому випадку, очевидно, $d_{ij}^{k} = d_{ik}^{k-1} + d_{kj}^{k-1}$

Таким чином, для знаходження значення функції досить вибрати мінімум з двох позначених значень.

Тоді рекурентна формула для d_{ij}^{k} має вигляд:

d_{ij}^{0} – довжина ребра (*i*, *j*);

$d_{ij}^{k} = \min\left(d_{ij}^{k-1}, d_{ik}^{k-1} + d_{kj}^{k-1}\right).$

Алгоритм Флойда-Уоршелла послідовно обчислює всі значення для $d_{ij}^{k}, \forall i, j$ для *k* від 1 до *n*. Отримані значення d_{ij}^{n} є довжинами найкоротших шляхів між вершинами *i*,*j*.

Далі помножимо кожен *j*-ий стовпець матриці *A* на v_j. Тоді зовнішній поділ для вершини з номером *i* – це максимальне значення в *i*-ому рядку матриці *A*. Порахуємо максимум в кожному рядку. Таким чином, отримаємо масив довжини *n*, де *i*-ий елемент – зовнішній поділ *i*-ої вершини.

Знайдемо найменший елемент в цьому масиві. Це буде найменше з зовнішніх розділень. Вершина з таким поділом і є зовнішній центр. В тому випадку, якщо вершин з таким поділом кілька, то всі вони – зовнішні центри.

Алгоритм пошуку внутрішнього центру відрізняється лише на першому кроці: кожний j-й рядок матриці A множиться на v_j [11][1)].

1.4.2 Абсолютні центри графа

Якщо припустити, що пожежна ділянка (бомбосховище, метро) може перебувати не тільки в якомусь районі, тобто вершині, а взагалі в будь-який точці графа, то потрібно трохи модифікувати дані вище визначення.

Для початку поширимо поняття поділів на точки ребер графа. При цьому в самих визначеннях фактично нічого не зміниться. Наприклад, так буде виглядати визначення зовнішнього поділу для довільної точки графа:

$$S_0(y) := \max [v_i\, d(y, x_i)],\ x_i \in V,\ y \in G$$

Використовуючи це розширене визначення розділень, визначимо абсолютний зовнішній і внутрішній центри. Абсолютний зовнішній центр – це така точка $y^*_0 \in G$, у якій зовнішній поділ мінімальний.

Аналогічно, абсолютний внутрішній центр – це така точка $y^*_t \in G$, у якої внутрішній поділ мінімальний.

Абсолютні радіуси визначаються як значення поділів відповідних абсолютних центрів. Позначаються r_0 і r_t. Розглянемо приклад на рис.1.4. Ваги ребер і вершин дорівнюють одиниці, центром є кожна вершина. У неорієнтованому графі зовнішні і внутрішні центри збігаються. Матриця відстаней має вигляд:

0	1	2	1	1
1	0	1	2	2
2	1	0	1	1
1	2	1	0	2
1	2	1	2	0

[1)] [11] В.А. Емеличев, О.И.Мельников. Лекции по теории графов. М.: Наука, 1990. 383с.

При додаванні червоною точки в середину ребра 1-5, вона стає абсолютним центром (рис.1.4). Для неї матриця відстаней виглядає так:

0.5 1.5 1.5 1.5 0

Таким чином, точка 6 більш «центральна», ніж будь-яка з вершин цього графа.

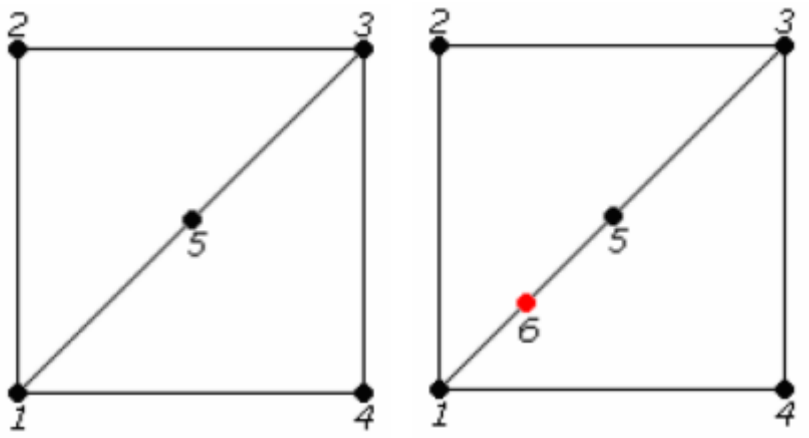


Рисунок 1.4 – Додавання абсолютного центру до графу *G*

Відзначимо, що наведений вище алгоритм пошуку центру тут використовувати неможливо, тому що кількість "потенційних абсолютних центрів" нескінченно, а отже неможливо порахувати поділ кожної такої точки.

В орієнтованому графі абсолютні центри не можуть лежати на орієнтованому ребрі. Тому будемо розглядати неорієнтовний граф (в разі орієнтованого графа можна викинути з розгляду всі спрямовані ребра, і тоді завдання зведеться до пошуку абсолютного центру в неорієнтованому). В цьому випадку абсолютні зовнішній і внутрішній центри збігаються. Тому немає сенсу уточнювати який саме центр ми шукаємо.

Має сенс розглянути два принципово різних алгоритму: метод Хакимі і ітераційний алгоритм.

Метод Хакимі – це алгоритм, що дозволяє знайти точне розташування абсолютного центру і значення його поділу (радіус). Він складається всього з двох кроків (рис.1.5):

1) для кожного ребра знайти точку з найменшим поділом.

2) з усіх цих точок вибрати точку з найменшим поділом – вона і буде абсолютним центром.

Другий крок алгоритму очевидний. Перший же на порядок складніше. Він здійснюється наступним чином.

Візьмемо ребро e_{ij}. Для будь-якої точки $y \in e_{ij}$ вірно наступне

$$s(y) = \max[v_k\, d(y, x_k)] = \max[v_k \min\{d(y, x_i) + d(x_i, x_k), d(y, x_j) + d(x_j, x_k)\}]$$

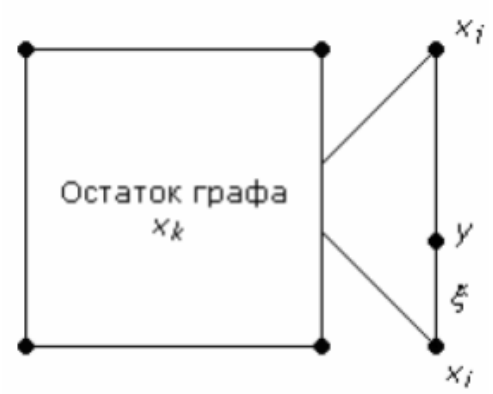


Рисунок 1.6 – Приклад розрахунку абсолютного центру графа G

Побудуємо на одному графіку дві залежності $v_k\{d(y, x_i) + d(x_i, x_k)\}$ і $v_k\{d(y, x_j) + d(x_j, x_k)\}$ від положення точки y на ребрі. Нижня огинаюча цих графіків – це графік залежності $v_k d(y, x_k)$ від положення точки y. Якщо побудувати на цьому ж графіку такі залежності для всіх інших $x_k \in V$, то верхня огинаюча цих графіків буде графіком залежності $s(y)$ від положення точки y. У загальному випадку це кусочно-лінійна функція. Її абсолютний мінімум (або мінімуми) – точка (и) с найменшим поділом на даному ребрі.

Алгоритм Хакимі можна значно оптимізувати. Ідея полягає в тому, щоб заздалегідь оцінити значення абсолютного радіусу і не розглядати ті ребра, на яких, виходячи з цих оцінок, абсолютного центру бути не може.

Припустимо, що абсолютний центр графа лежить на ребрі e_{ij}, тоді значення абсолютного радіусу не менше, аніж

$$p_{ij} = \max[v_k \min\{d(x_i, x_k), d(x_j, x_k)\}],\ x_k \in V.$$

отже, $P = \min(p_{ij})$, $e_{ij} \in E$ – обґрунтована нижня оцінка для абсолютного радіусу.

Нехай тепер абсолютний центр лежить в середині ребра e_{ij}, тоді абсолютний радіус

$r = p_{ij} + v^*_k d(x_i, x_j)/2$, де v^*_k – вага тієї вершини, в якій досягається $\max[v_k \min\{d(x_i, x_k), d(x_j, x_k)\}]$ (тобто тієї, яка визначає значення p_{ij}). Виходячи з цього,

$H = \min[p_{ij} + v^*_k d(x_i, x_j)/2]$ – обґрунтована верхня оцінка для абсолютного радіусу.

Таким чином, будь-яке ребро $e_{ij} \in E$, для якого $p_{ij} \geq H$, можна на розглядати [12][1)].

Ітераційний алгоритм. Цей алгоритм не дає точне розташування абсолютного центру і значення абсолютного радіусу. Результат його роботи – деяка область, в якої лежить абсолютний центр і, відповідно, зразкове значення абсолютного радіусу.

Його суть полягає в наступному:

1) Вибираємо деякий $\delta > 0$. Нехай λ спочатку дорівнює δ.

2) Знайдемо перетин всіх таких областей (рис.1.7)

$Q_\lambda(x_i) = \{y \in G: l(y, x_i) \leq \lambda\}$.

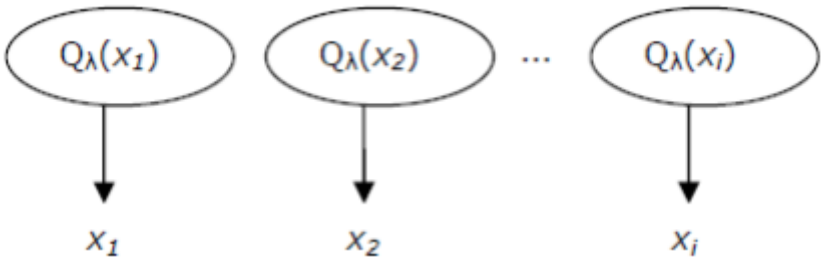


Рисунок 1.7 – Приклад ітераційного алгоритму

3) Якщо цей перетин непорожній, то центр знаходиться в ньому.

Якщо ж він порожній, то збільшимо λ на δ і повернемося до кроку 2.

[1)] [12] С.Ю.Белецкая. Комбинаторика. Графы. Алгоритмы. Учеб.пособие. Воронеж: ВГТУ, 2003.

Зауважимо, що розмір отриманої області залежить від δ. Іншими словами, чим менше ми візьмемо δ, тим буде більше точність отриманого результату. Якщо все ж потрібно точне значення, то можна поєднати цей алгоритм з методом Хакимі: спочатку знайти область, в якій лежить центр, а потім розглянути тільки ті ребра, частини яких лежать в цій області [12][1)].

[1)] [12] С.Ю.Белецкая. Комбинаторика. Графы. Алгоритмы. Учеб.пособие. Воронеж: ВГТУ, 2003.

2 МЕТОДИ ТА ІНСТРУМЕНТАЛЬНІ ЗАСОБИ ГІС-АНАЛІЗУ ТРАНСПОРТНИХ МЕРЕЖ

Аналіз даних – це процес вилучення інформації для підтримки прийняття рішення.

В останні роки доступність геопросторових даних значно збільшилась. З ростом обсягів добровільної географічної інформації (Volunteered Geographic Information, VGI), відкритих урядових даних (Open Government Data, OGD) та повсякденного використання шляхів пошуку за допомогою GPS, щодня створюється величезна кількість геопросторових даних. На сьогоднішній день також є достатньо можливостей для зберігання цих даних, використання всебічних методів аналізу та дуже детальної візуалізації результатів на льоту. Це може призвести до створення фундаменту бази знань, що може принести велику користь у процесі прийняття рішень. Останнім часом у цьому контексті термін Business Intelligence (BI) стає дедалі популярнішим, що означає «методи та технології, що збирають, зберігають, звітують та аналізують бізнес-дані, щоб допомогти людям прийняти бізнес рішення. Бізнес, державні та наукові організації все більше рухаються до процесів прийняття рішень, які базуються на інформації. Паралельно зростає і обсяг даних, що представляють діяльність організацій, які зберігаються в базах даних.

Використання методів обробки геопросторових даних вводить нові способи вилучення знань. Багато даних включають просторові компоненти (оцінки коливаються від 50% до 85%), що робить ще більш важливим використання методів, які орієнтовані на обробку просторової інформації. Аналіз даних руху – цікаве завдання у контексті інтелектуального аналізу геопросторових даних. Шляхом аналізу особливостей траєкторій та місць, які людина відвідала, може бути отримана цінна інформація про поведінку особистості.

Територіальна розподіленість транспортних систем робить їх ідеальним об'єктом автоматизації засобами геоінформаційних систем. Взагалі кажучи, ГІС є оптимальною платформою для комплексних рішень в сфері транспорту. Просторова складова є природною основою інтеграції завдань управління транспортною інфраструктурою, рішення розрахункових задач, задач оперативного управління, навігації та ін.

Так для вирішення завдань побудови та оптимізації маршрутів на існуючій дорожній мережі геоінформаційні системи пропонують цілий ряд інструментів. Наприклад, засоби просторового аналізу, наявні в модулі ArcGIS Spatial Analyst, дозволяють визначити транспортну потребу районів міста на основі аналізу різних чинників – щільності населення, рівня автомобілізації, розміщення центрів тяжіння (вокзалів, ринків, великих торгових центрів, розважальних комплексів) і т.д. Природно, виконувати такий аналіз зручно на основі цифрової карти і районування, також підготовлених в ГІС. Наявні в модулі ArcGIS Network Analyst засоби аналізу мереж дозволяють будувати оптимальні маршрути на реальній вулично-дорожньої мережі з усіма її можливостями і обмеженнями (дозволені напрямки руху, повороти, пропускна спроможність вулиць ін.).

2.1 Джерела та характеристики просторових даних

За своєю організацією географічні дані діляться на векторні, растрові і атрибутивні. Нижче розглянуті характеристики геоданих.

Векторні дані використовуються в ГІС для відображення об'єктів реального світу. У свою чергу, об'єктами реального світу є все, що ми бачимо навколо – річки, моря, гори, будівлі, автомобілі і т.п. Кожен з перерахованих елементів може бути представлений у вигляді об'єкта в ГІС-застосунку. Векторні об'єкти характеризуються атрибутами, які складаються з текстової та числової інформації, яка описує об'єкт. Зовнішній вигляд векторних об'єктів в застосунках ГІС завжди визначається їх геометрією і різниться за принци-

пом: точковий об'єкт (геометрія векторного об'єкта складається з одного вузла); лінійний об'єкт (геометрія векторного об'єкта містить два і більше вузла); полігональний об'єкт (геометрія векторного об'єкта утворена трьома і більше вузлами), геометрія векторного об'єкта представлена на рис.2.1 [13, 14][1)].

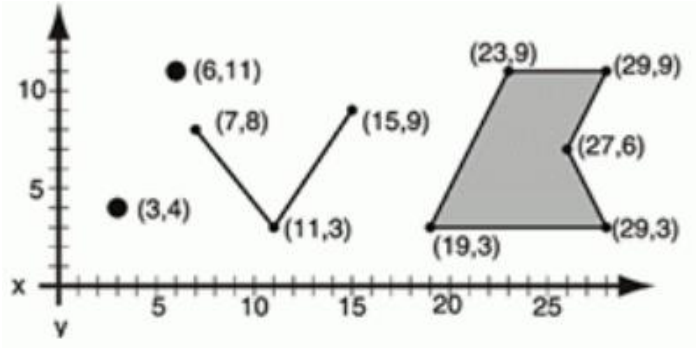


Рисунок 2.1 – Геометрія векторних об’єктів у графічному представленні

Растрові дані можуть бути використані в основному тільки з растровими моделями даних. У той час, коли векторні дані використовують геометрію, растрове моделювання використовує підхід побудови зображення за допомогою сітки пікселів (комірок), кожен з яких містить значення, що описує стан поверхні, яка охоплюється цією коміркою. Растрові дані використовуються в ГІС в разі, коли потрібно відобразити безперервне по площі явище (об'єкт), який неподільний на векторні об'єкти. Супутниковий знімок – це приклад растрових даних. Багато розробники ГІС-застосунків використовують растрові рішення в якості підкладки під векторні шари, тим самим покращуючи сприйняття інформації, що міститься в них, створюючи гібридні ГІС, структура растрових даних представлений на рис.2.2.

Атрибутивні дані відображають численні характеристики об'єкта. Ці показники можуть бути кількісними або якісними, але так або інакше повин-

[1)] [13] Світличний О.О., Плотницький С.В. Основи геоінформатики: Навчальний посібник. Суми: ВТД «Університетська книга», 2006. 295 с.

[14] Шипулін В. Д. Основи ГІС-аналізу: навч. Посібник. Харк. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. Х.: ХНУМГ, 2014. 330 с.

ні описувати об'єкт. Тому в ГІС кожен географічний об'єкт має один або більше атрибутів, які вказують, що з себе представляє об'єкт, описуючи його або містить будь-які величини, що відносяться до нього.

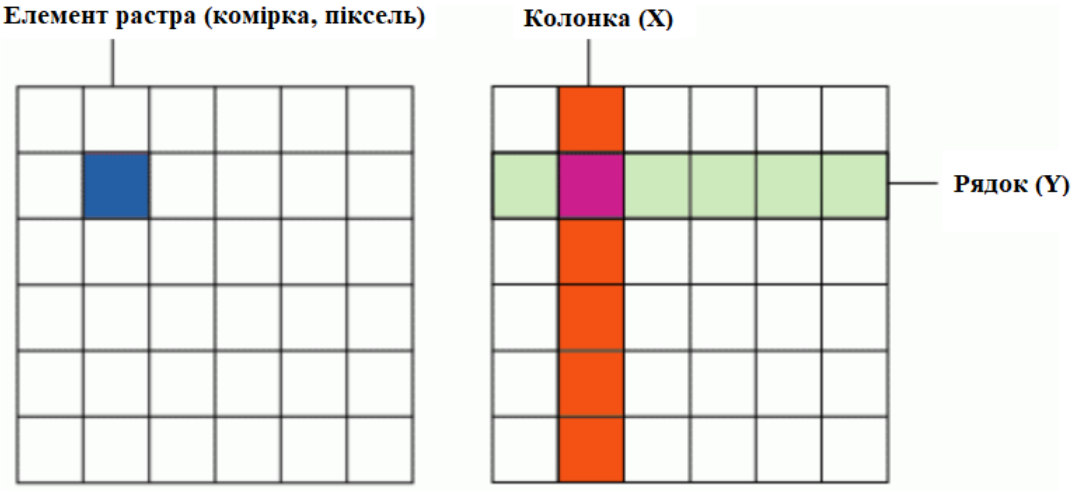


Рисунок 2.2 – Структура растрових даних

Об'єкти, які мають схожі атрибути можуть бути об'єднані в групу об'єктів за конкретною ознакою. Наприклад, класифікація доріг за типом дорожнього полотна (асфальтні, грунтові та ін.), за чисельністю населення, категорія землекористування та ін., в залежності від типу об'єкта і атрибутів [14][1)].

Джерела просторових даних для ГІС – це фундаментальна основа інформаційного забезпечення всіх геоінформаційних систем.

У цьому процесі джерелами просторових даних є аналогові або цифрові дані, які служать фундаментом для формування моделей просторових даних. Основні типи джерел геоданих:

– відкриті топографічні дані;
– картографічні дані;
– космічна зйомка;
– бази географічних даних;
– географічні карти;

[1)] [14] Шипулін В. Д. Основи ГІС-аналізу: навч. Посібник. Харк. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. Х.: ХНУМГ, 2014. 330 с.

– дані дистанційного обстеження території (об'єктів).

Нижче наведені відомі в даний час картографічні сервіси: Open Street Map, Scribble Maps, Build-A-Map, Яндекс.Карти, Google Maps, Animaps, Virtual Earth (Bing Maps).

Open Street Map – це веб-картографічний ресурс, який створюється своїми ж учасниками – користувачами Інтернету [15][1)]. Має детальну вільну і безкоштовну географічну карту, приклад представлений на рис.2.3.

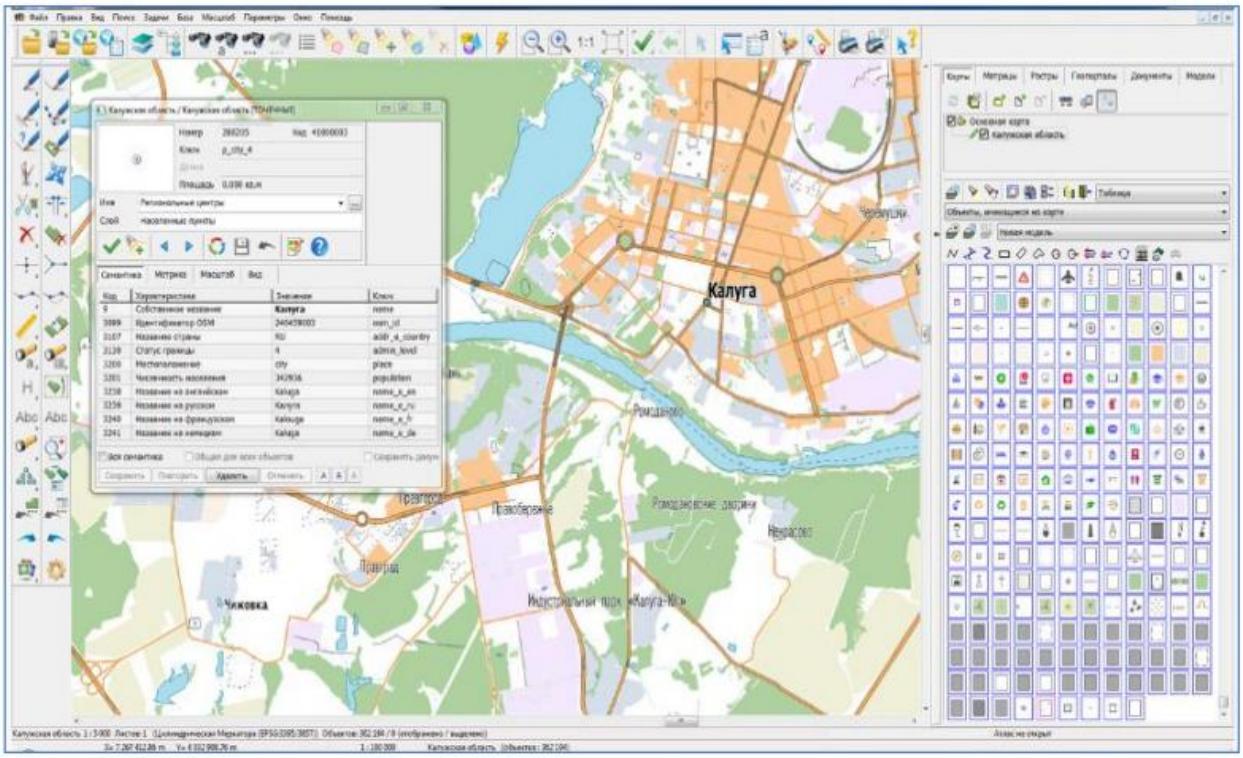


Рисунок 2.3 – Приклад карти на основі даних Open Street Map

Для того щоб створити карти використовують дані з особистих GPS-трекерів, відеозапис, супутникові знімки, фотографії, зроблені з великої висоти, і знання професіонала, що створює карту.

[1)] [15] Картографічні дані OpenStreetMap. URL: https://www.openstreetmap. org (Дата звернення 10.11.2020)

Будь-який користувач, який зареєстрований в сервісі може внести будь-які зміни на карту. Використовувати інформацію із закритих сервісів, таких як Google Maps, без схвалення правовласника заборонено.

Scribble Maps – картографічний онлайн сервіс, який підійде для використання широкого кола користувачів з різним рівнем комп'ютерної грамотності. Даний ресурс має простий інтерфейс і легко керованими різними можливостями, що робить його одним з популярних сервісів в сфері ГІС.

Scribble Maps має широкий спектр функцій, наприклад розмітка карти, вимір відстані між об'єктами, імпорт даних в формат KML і табличні дані, є можливість нанесення користувальницьких міток, необхідного тексту, зображень тощо.

Наступний картографічний сервіс – Build-AMap, який включає такі функціональні опції, як додавання геометок і треків, текстових написів, полігональних об'єктів, маршрутів та ін. Крім перерахованого, користувач сервісу має можливість сфокусувати в карту дані про погоду, геоінформаційні точки Google Places, знімки фотохостингу Panoramio.

Сформовану карту можна помістити в хмарне сховище і закріпити посиланням або кодом для вбудовування в веб-сторінку. Первинним ресурсом для вилучення картографічної інформації в цьому випадку стає ресурс Google Maps, на результати якого користувач поверх може накладати власні шари і вставляти власні дані.

Оптимальним для багатьох користувачів в даний час можна назвати набір застосунків на базі безкоштовного картографічного сервісу – Google Maps. Даний ресурс формує картографічні зображення на основі отриманих знімків в масштабі всієї земної кулі. Можливості надання інформації для Google Maps дуже високі, сервіс надає безлічі користувачів отримати зображення з дуже високою роздільною здатністю, максимально деталізовані. В ресурс Google Maps вбудована карта автомобільних доріг з пошуком маршрутів, що охоплює безліч територій.

Наступним розглянутим сервісом стане один з сервісів в структурі програмних продуктів Microsoft – сервіс Virtual Earth (Bing Maps). Даний сервіс функціонує вже понад 15 років (з 2005 року) і базується на технологіях Microsoft MapPoint і TerraServer. Практично з дати заснування даного продукту, він має можливості 3D моделювання, в зв'язку з чим, він згодом отримав назву «Live Search Maps» і був інтегрований в портал Live Search. В даний час ресурс носить ім'я Bing Maps, а платформа Virtual Earth.

Таким чином, підсумовує все викладене вище, оптимальним варіантом для виконання ГІС-аналізу та просторового моделюваня є транспортна мережа, яка представлена лінійними об’єктами в векторній моделі даних. Дані для побудови графа транспортной мережі в ГІС можуть бути отримані з вільного картографічного сервісу Open Street Map [15][1)].

2.2 Огляд алгоритмів та інструментальні засоби ГІС для аналізу транспортних мереж

Транспортна система одна з головних елементів розвитку, як цілої країни, так і окремого міста. Транспортна мережа має великий вплив на форму всієї територіальної суспільної системи. Завдяки ефективній міської транспортної системи визначаються форми і межі районів і мікрорайонів. Безсумнівно, великий вплив транспорту на географічний і урбаністичний характер зростання міських агломерацій і прилеглих територій. Встановлена пряма залежність транспортної доступності і транспортно-географічного розташування міських районів з розташованими на їх території соціально-значимими, житловими, виробничими, торговими і комунальними об'єктами від конфігурації транспортних комунікацій.

[1)] [15] Картографічні дані OpenStreetMap. URL: https://www.openstreetmap. org (Дата звернення 10.11.2020)

Громадський транспорт є одним з головних факторів, так як саме він значно визначає мобільність населення міст. Ще визначає форми і межі міських зон і впливає на географічні напрямки зростання міст.

Від виду маршрутної мережі громадського транспорту залежить транспортна доступність населення і географічне розташування районів.

Доступність об'єктів ілюструє початковий рівень транспортної доступності. Найбільш це принципово для великих населених пунктів, так як час пересування людей до робочого місця, місця проживання та обслуговування визначає зазвичай їх територіальну забудову. Ці показники відображають відсутність або наявність доступності даних послуг.

Доступність громадського транспорту розраховується від різних елементів. При проведенні аналізу необхідно визначати віддаленість від початкової точки до транспортних послуг (зупинок, павільйонів і т.д.) і аналізувати не за прямим вектору, а з урахуванням конкретних маршрутів, тобто за існуючою дорожньої мережі, то як йде реальний пасажир [16][1)].

Рівень транспортної доступності означає ступінь забезпеченості транспортними послугами. Маршрути транспорту накривають місто густою мережею, виходячи з цього транспортна доступність пасажирського транспорту виступає обов'язковою умовою для забезпечення якісних транспортних послуг.

У галузі транспорту ГІС давно вже показали свою ефективність завдяки можливості побудови оптимальних маршрутів, як для окремих перевезень, так і для цілих транспортних систем, в масштабі окремого міста чи цілої країни. При цьому можливість використання найбільш актуальної інформації про стан дорожньої мережі та пропускної здатності дозволяє будувати дійсно оптимальні маршрути.

Транспортні мережі в ГІС будуються за такою схемою:

[1)] [16] Особенности моделирования транспортной доступности. URL: http://smartloc.ru. (Дата звернення 10.11.2020)

– пошук і підготовка даних для побудови мережі серед відкритих інтернет-джерел;

– організація та підготовка даних (формування бази географічних даних, перевірка топології);

– створення набору мережевих дані в базі геоданих;

– побудова дорожньої мережі.

Технологія побудови мережі громадського транспорту починається зі створення графа дорожньої мережі.

Граф доріг – це цифрова векторна карта, що складається з топологічно пов'язаних дуг і вузлів, розташування і властивості яких із заданою точністю і повнотою передають маршрути і організацію руху наземного транспорту. Граф доріг створюється по виділеним об'єктам дорожньої мережі і являє собою призначену для користувача карту з дугами і вузлами. На етапі побудови в семантичні характеристики дуг і вузлів записується інформація про зв'язок мережі та атрибути для рішення пошукових завдань.

Засоби редагування графа доріг призначені для уточнення графа в місцях багаторівневих розв'язок і формування заборон поворотів. Користувач має можливість вручну видалити, додати вузли мережі, замінити дугу з двостороннім рухом на дугу з одностороннім рухом, провести розпаралелювання доріг, створити дуги і розвороти, сформувати на перехрестях заборони поворотів.

Пошук мінімального шляху між точками (населеними пунктами) здійснюється з урахуванням будь-яких характеристик, записаних в дуги мережі (тип доріг, швидкість руху, кількість проїзних частин). Найкоротший маршрут можна знайти або по мінімальній довжині шляху, або за мінімальним часом проходження маршруту. При знаходженні мінімального шляху є можливість виключення деяких дуг, наприклад, аварійних ділянок, з пошуку. Результати пошуку відображаються на карті у вигляді об'єкта – маршруту.

Для побудови мережі громадського транспорту в пакеті ArcGIS необхідно, щоб був включений додатковий модуль ArcGIS Network Analyst. Мо-

дуль ArcGIS Network Analyst призначений для проведення мережевого просторового аналізу [17][1)]. З його допомогою можна створити рішення, що дозволяють будувати маршрути, створювати шляхові листи, шукати найближчі пункти обслуговування, створювати зони обслуговування, розраховувати матриці витрат на переміщення і проводити аналіз придатності місця розташування. Використовуючи комплексну мережеву модель, можливо, легко створити граф з ГІС-даних, в тому числі і граф в тривимірному просторі. ArcGIS Network Analyst дозволяє динамічно моделювати реальні параметри мережі для вирішення задач оптимізації переміщення транспорту, включаючи заборонені повороти і розвороти, швидкісний режим, обмеження по висоті транспорту, завантаженість мережі протягом діб.

Щоб включити даний модуль необхідно на панелі інструментів перейти у вкладку «Налаштування» і вибрати «Додаткові модулі», поставити галочку навпроти «Network Analyst» (рис.2.4).

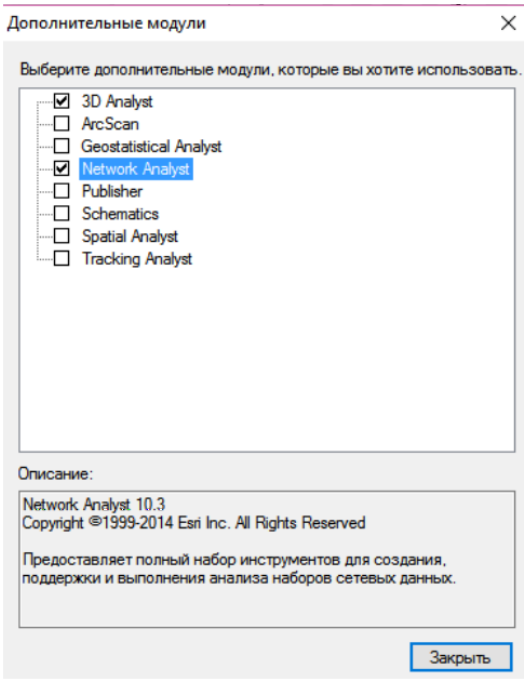


Рисунок 2.4 – Підключення модуля ArcGIS Network Analyst

[1)] [17] Довідка по модулю ArcGIS Network Analyst. URL: http://desktop.arcgis.com (Дата звернення 10.11.2020)

Панель інструментів Network Analyst в ArcMap (рис. 2.5) містить загальну інформацію та деякі функціональні можливості. Наприклад, з її допомогою можна дізнатися, який набір мережевих даних (якщо такий є) є активним. Крім того, за допомогою інструменту Ідентифікація мережі (Network Identify) можна переглянути атрибути мережевих елементів на карті; він також дозволяє вибирати мережевий аналіз і створює відповідний шар мережевого аналізу. Інші корисні кнопки на панелі інструментів: кнопка Шляховий лист (Directions), яка відкриває інструкції по навігації на маршруті з зазначенням поворотів; кнопка Вікно Network Analyst (Network Analyst Window), яка відкриває або закриває вікно Network Analyst; кнопка Розрахунок (Solve), яка формує результати мережевого аналізу [17][1)].

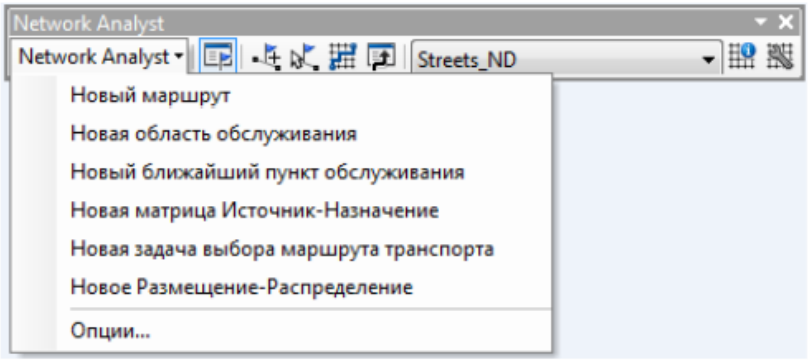


Рисунок 2.5 – Панель інструментів Network Analyst

2.2.1 Побудова тестових маршрутів

Побудова найкоротших маршрутів між обраними об'єктами виконується у вкладці Network Analyst – «Новий маршрут». Після цього в Таблиці змісту відобразиться шар мережевого аналізу (рис. 2.6).

[1)] [17] Довідка по модулю ArcGIS Network Analyst. URL: http://desktop.arcgis.com (Дата звернення 10.11.2020)

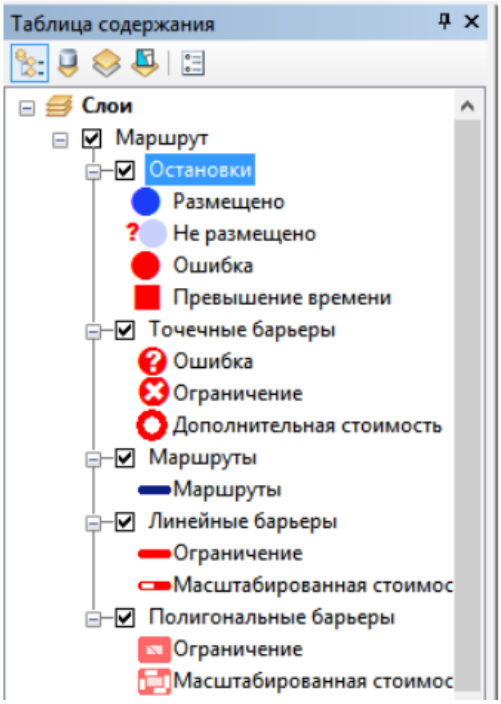


Рисунок 2.6 – Шар мережевого аналізу

Шар мережевого аналізу надає інформацію про мережеві завдання, а після їх рішення надає також і варіант рішення. Після створення шар мережевого аналізу є просто універсальною структурою для постановки мережевий завдання, наприклад, побудова маршруту, області обслуговування або завдання розміщення. Щоб конкретизувати загальну проблему, необхідно визначити властивості і заповнити даними шар аналізу. Коли завдання повністю визначена, можна розпочати процедуру її рішення. Результати зберігаються в шарі аналізу.

За допомогою вікна Network Analyst в ArcMap можна швидко і легко управляти вхідними та вихідними даними шарів мережевого аналізу (рис. 2.7).

На панелі інструментів Network Analyst вибираємо кнопку «Створити мережеві стани» і на карті вибираємо будь-яке розташування і ставимо точку. Потім за допомогою кнопки «Розрахунок» запускаємо поточний аналіз. На

карті автоматично вибудовується найкоротший маршрут між обраними точками (рис. 2.8).

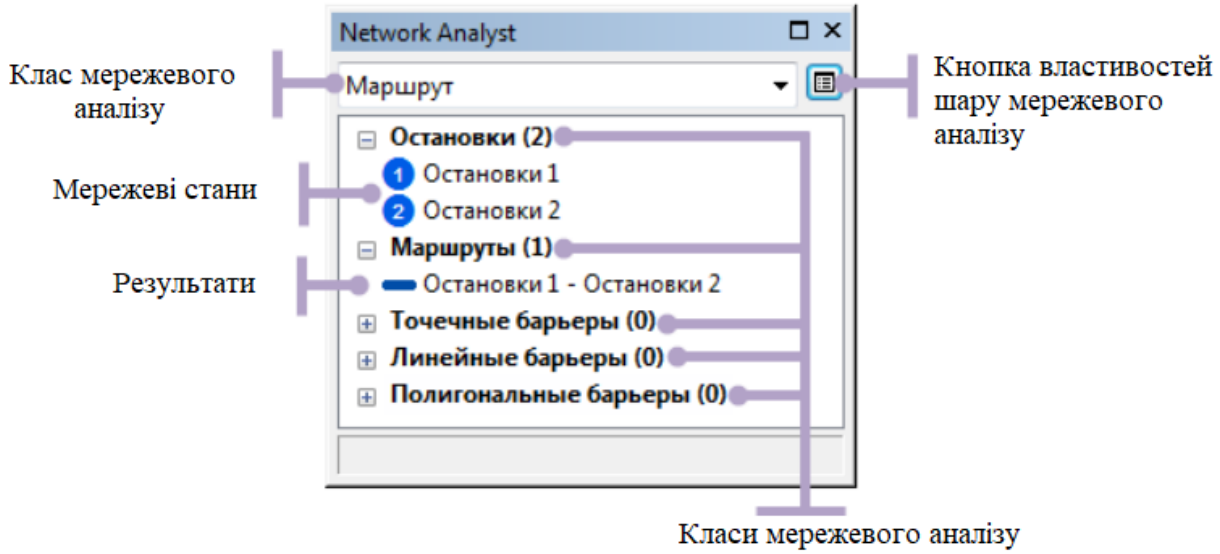


Рисунок 2.7 – Вікно Network Analyst

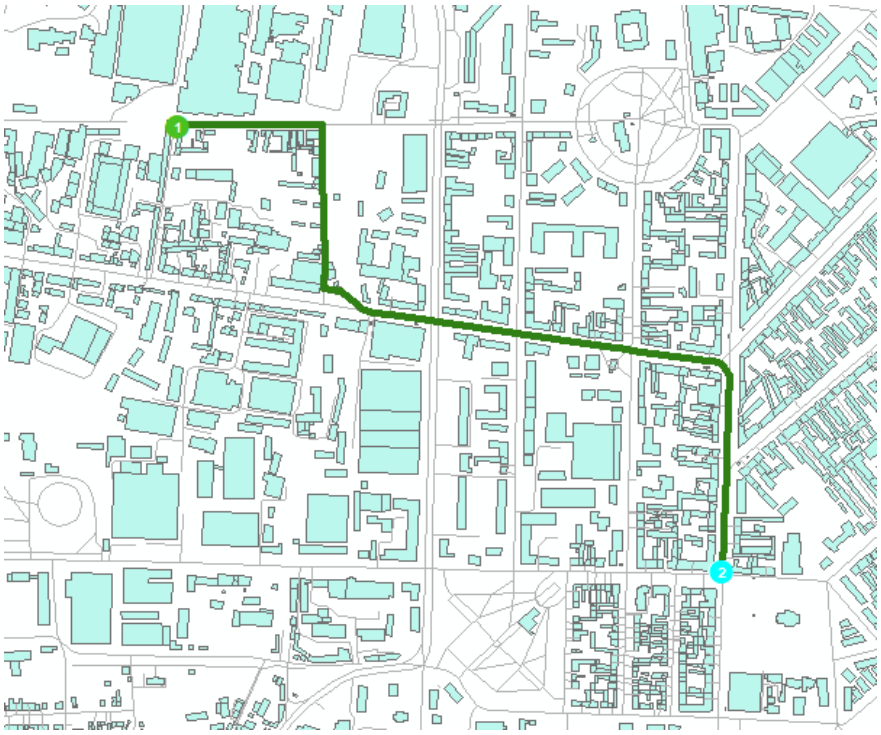


Рисунок 2.8 – Розрахований маршрут

2.2.2 Побудова областей обслуговування

За допомогою Network Analyst можна знайти області обслуговування для будь-якого місця розташування в мережі. Область обслуговування мережі – це регіон, який охоплює всі доступні вулиці, а саме вулиці, що знаходяться в межах зазначеного імпедансу.

Наприклад, область обслуговування 10-хвилинної доступності для пункту обслуговування, включає всі вулиці, які знаходяться в межах 10-хвилинної досяжності від цього пункту.

Області обслуговування, створювані Network Analyst, також допомагають обчислювати доступність. Концентричні області обслуговування показують, як змінюється доступність залежно від імпедансу. Після створення областей обслуговування їх можна використовувати для визначення того, скільки землі, людей або інших ресурсів знаходиться в околиці або регіоні.

Доступність – це те, яким чином можна найшвидше дістатися до потрібного місця. В Network Analyst, доступність може бути виміряна в термінах часу в дорозі, відстані і будь-якого іншого імпедансу в мережі. Оцінка доступності допомагає відповісти на основні питання, такі як: «Скільки людей проживає в 10 хвилинах їзди від кінотеатру? » або «Скільки покупців живе в межах півкілометра ходьби від продуктового магазину? » Аналіз доступності може допомогти визначити, наскільки придатним є місце для відкриття нового підприємства. Також дані про доступності можуть стати в нагоді для визначення місця розташування організацій поряд з існуючим підприємством, які можуть бути корисні при прийнятті маркетингових рішень.

Оцінити доступність можна за допомогою буферної відстані навколо точки. Наприклад, визначити кількість потенційних клієнтів, які живуть в радіусі 5 кілометрів від пункту обслуговування, використовуючи звичайний коло. Але, беручи до увагу, що люди переміщаються по дорогах, цей спосіб не відобразить актуальних даних про доступність місця. Мережа обслуговування, що розраховується Network Analyst, здатна подолати це обмеження

шляхом визначення вулиць в радіусі п'яти кілометрів від обраного місця, доступних через мережу доріг. Потім мережеві служби можуть визначити, які об'єкти знаходяться на доступних вулицях, наприклад, знайти конкуруюче підприємство в 5 хвилинах їзди від вашого місця розташування.

Кілька концентричних областей обслуговування допомагають зрозуміти, яким чином змінюється доступність при збільшенні імпедансу. Дані про доступність можна використовувати, наприклад, для визначення кількості лікарень, які перебувають в межах 5, 10 і 15 хвилин їзди від шкіл.

Шар аналізу області обслуговування містить всі вхідні дані, параметри і результати аналізу області обслуговування. Створити шар аналізу області обслуговування можна з використанням панелі інструментів Network Analyst, клацнувши Network Analyst, потім – Нова область обслуговування (New Service Area) (рис.2.9).

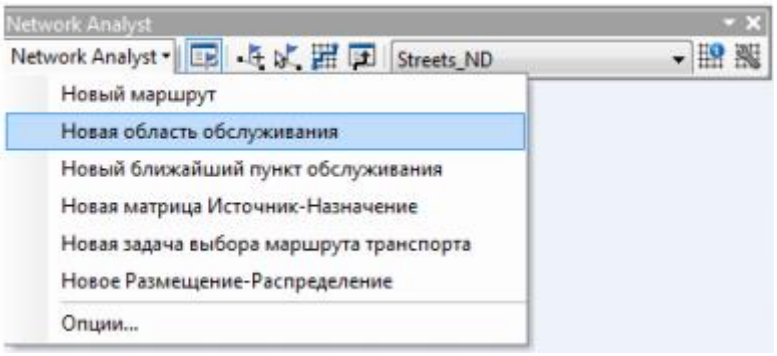


Рисунок 2.9 – Панель інструментів Network Analyst

При створенні шару аналізу області обслуговування цей шар відображається в вікні Network Analyst разом з шістьма класами мережевого аналізу: Пункти обслуговування, Лінії, Полігони, Точкові бар'єри, Лінійні бар'єри і Полігональні бар'єри (рис.2.10).

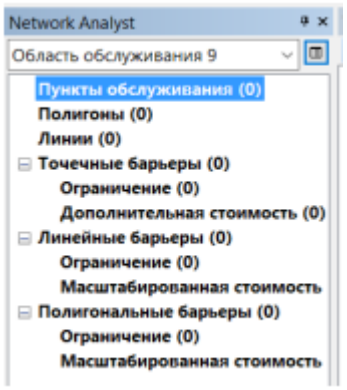


Рисунок 2.10 – Вікно Network Analyst

Шар аналізу області обслуговування також відображається в таблиці змісту як складовий шар з ім'ям «Область обслуговування» (Область обслуговування 1 і т. д., якщо область обслуговування з таким ім'ям вже існує на карті) (рис.2.11). Існує шість шарів просторових об'єктів, умовні позначення яких за замовчуванням можна змінити у відповідних діалогових вікнах Властивості шару.

В цьому класі мережевого аналізу зберігаються мережеві стани, які використовуються як пункти обслуговування в аналізі області обслуговування. Шар просторових об'єктів Пункти обслуговування відображається з використанням трьох символів за замовчуванням: Розміщений, Нерозміщення і Помилка. Умовні позначення шару «Пункти обслуговування» можна змінити в діалоговому вікні Властивості шару.

При створенні нового шару аналізу області обслуговування клас «Пункти обслуговування» створюється порожнім. Він заповнюється тільки після додавання в нього мережевих станів. Для створення області обслуговування необхідний як мінімум один пункт обслуговування.

У класі мережевого аналізу полігонів зберігаються результуючі полігони області обслуговування. Як і для інших шарів просторових об'єктів, до його умовних позначень можна отримати доступ в діалоговому вікні Властивості шару просторових об'єктів.

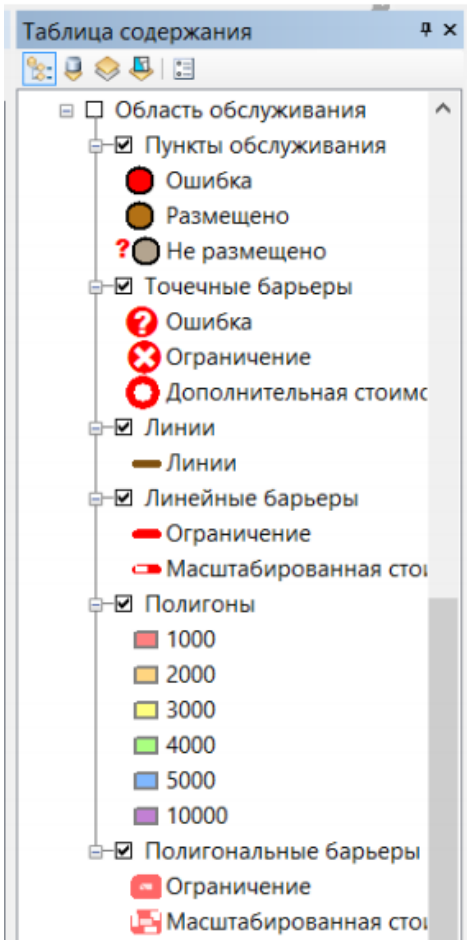


Рисунок 2.11 – Таблиця змісту інструменту Network Analyst

При створенні нового шару аналізу області обслуговування клас «Полігони» створюється порожнім. Він заповнюється тільки після розрахунку шару аналізу. Після обчислення області обслуговування полігони зберігаються в шарі просторових об'єктів і перераховуються у вікні Network Analyst.

У класі «Лінії» зберігаються результуючі області обслуговування в вигляді лінійних просторових об'єктів і представляються ребра мережі, які досягаються в межах заданого імпедансу. Лінії є більш достовірного представлення області обслуговування, ніж полігони, оскільки область обслуговування ґрунтується на вимірах уздовж мережі.

Лінії області обслуговування не створюються за замовчуванням в процесі аналізу області обслуговування; їх можна створити за вибором, вказав-

ши параметр «Створювати лінії на закладці "Властивості шару аналізу області обслуговування.

Шар просторових об'єктів ліній області обслуговування умовно позначається так само, як і інші шари лінійних просторових об'єктів.

Бар'єри використовуються для часового обмеження, додавання імпедансу і його масштабування на частинах мережі. При створенні нового шару мережевого аналізу класи бар'єрів створюються порожніми. Вони заповнюються тільки при додаванні в них об'єктів, при цьому додавання бар'єрів не потрібно.

2.3 Огляд бібліотек Python для обробки просторової інформації

При аналізі просторової інформації для автоматизації часто повторюваних завдань геообробки може бути використаний програмний код у вигляді окремих сценаріїв. Сценарії можуть збільшити продуктивність обчислень. Деякі завдання для сценаріїв включають загальне управління даними, такі як переформатування даних, копіювання файлів для резервних копій та пошук вмісту бази даних. Сценарії також можуть використовувати вже існуючі в ГІС набори інструментів геообробки. В ГІС загального призначення ArcGIS, яка була використана в магістерській роботі, використовується мова сценаріїв Python та геообробка за допомогою програмного забезпечення.

Мова програмування Python є ідеальною мовою програмування для користувачів ГІС з кількох причин [18][1)]:

– Python лаконічна, мова, яку легко інтерпретувати за допомогою чіткого візуального макета.

– Python є об'єктно-орієнтованою мовою. Об’єктно-орієнтоване програмування (ООП) організоване навколо об’єктів, які мають властивості та

[1)] [18] Laura Tateosian. Python for ArcGIS. Springer International Publishing Switzerland. 2015. p. 538

функції, що впливають на цей об’єкт. Мови ООП мають спільні умови, що полегшує роботу тих, хто їх має досвід роботи з іншими мовами.

– Python має достатню кількість довідкових ресурсів. Python – мова програмування з відкритим кодом. У дусі програмного забезпечення з відкритим кодом, спільнота програмістів Python розміщує безліч безкоштовної інформації в Інтернеті.

– ГІС підтримують мову програмування Python. Через багато причин, перерахованих вище, спільнота ГІС прийняла мову програмування Python. Програмне забезпечення ArcGIS, зокрема, інтегрувало Python і розширює функціональність Python з кожним новим випуском. Скрипти Python можна використовувати для запуску інструментів геообробки ArcGIS тощо. Програмне забезпечення ArcGIS Desktop навіть забезпечує вбудований командний рядок Python для запуску операторів коду Python. Деякі програми з відкритим кодом ГІС також забезпечують інтерфейси програмування на Python. Наприклад, GRASS GIS включає вбудований командний рядок Python для запуску інструментів геообробки GRASS через Python. Команди QGIS та PostGreSQL / PostGIS також можна запускати з Python.

– Python постачається з ArcGIS. Python встановлюється автоматично при встановленні ArcGIS.

Мова Python дозволяє використовувати різні формати даних ГІС, включаючи складені формати даних, такі як растри GRID, бази геоданих та файли форм.

Растр GRID визначає географічний простір із масивом комірок однакового розміру, розташованих у стовпцях і рядках. Формат файлу складається з двох каталогів, кожен з яких містить кілька файлів. Один каталог має назву растру і містить файли «.adf», які зберігають інформацію про ступінь, роздільну здатність комірок тощо.

*Shapefile з*берігає географічні об’єкти та їх негеографічні атрибути. Це популярний формат для зберігання векторних даних ГІС. Векторні дані зберігають функції у вигляді наборів точок та ліній, на відміну від растрів, які

зберігають дані у комірках сітки. Shapefile складається з трьох або більше файлів, кожен з різними розширеннями файлів. Файли ‘.shp’ (shapefile), ‘.shx’ (заголовок) та ‘.dbf’ (пов’язана база даних) є обов’язковими. Ви також можете мати додаткові файли, такі як файли ".prj" (проекція) та ".lyr" (шар).

Файли dBASE. Один із обов’язкових типів файлів шейп-файлів (‘.dbf’) також може існувати як окремий файл для зберігання табличних даних бази даних.

Layer Files. Файл ‘.lyr’ можна використовувати разом із shapefile для зберігання метаданих візуалізації. Зазвичай, коли shapefile переглядається в ArcGIS, символіка (елементи візуального представлення елементів, таких як колір, контур тощо) призначається довільно. Символіку можна редагувати в ArcMap, і тоді файл ‘.lyr’ може зберігати ці налаштування. Файл ‘.lyr’ містить специфікації для подання географічного набору даних (набору форм або растрового набору даних) і повинен зберігатися в тому ж каталозі, що і географічні дані.

Geodatabase. Esri має три формати баз геоданих: файловий, персональний та ArcSDE. База геоданих зберігає колекцію наборів даних ГІС. Кілька форматів даних (растрові, векторні, табличні тощо) можуть зберігатися разом у базі геоданих. На рис. 2.12 представлена база геоданих файлу region.gdb у провіднику ArcCatalog. Всередині каталогу знаходиться п’ять наборів даних (чотири векторні та один растровий).

Рисунок 2.12 – Файл geodatabase у проводніку ArcCatalog

ArcGIS пропонує великий набір інструментів для обробки даних. На панелі ArcToolbox в ArcCatalog (та ArcMap) перелічені набори інструментів - 3D Analyst, Analysis, Cartography тощо (рис. 2.13). Інструменти згруповані в набори інструментів за типом дій, які вони виконують, і кожен набір інструментів містить набори інструментів, які додатково групують інструменти за їх функціональністю.

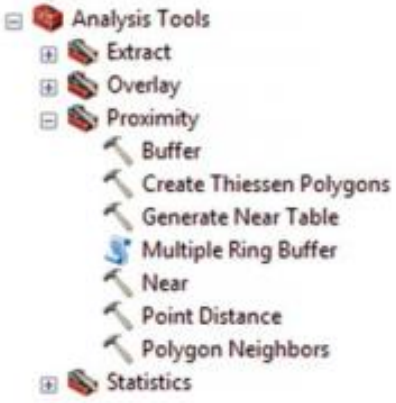


Рисунок 2.13 – Інструменти ArcToolbox

Python може викликати майже всі інструменти в ArcToolbox, і таким чином можуть бути автоматизовані повторювані процеси.

Інструменти ArcGIS Toolbox також можна запускати через ModelBuilder. ModelBuilder – це програма, вбудована в ArcCatalog (та ArcMap), що дозволяє користувачам створювати робочу візуалізацію, яка називається моделлю (рис.2.14). Інструменти ArcToolbox можна перетягнути на панель моделі та підключити, щоб створити робочий процес. Коли модель працює, вона виконує інструменти та базові оператори коду, які відповідають частинам моделі. Базовий код можна експортувати в скрипт Python [19][1)].

Щоб використовувати функціональність ArcGIS у Python, сценарію потрібно імпортувати arcpy. Пакет arcpy – це програмний засіб Python для доступу до ArcGIS з Python. Основні функціональні компоненти arcpy наведені

[1)] [19] Офіційний сайт Python. URL: http://www.python.org (Дата звернення 10.11.2020)

на рис.2.15 [18, 20][1)]. Функції arcpy (верхне ліве вікно на рис. 2.15) забезпечують підтримку робочих процесів геообробки. Наприклад, функції можна використовувати для переліку наборів даних, отримання властивостей набору даних, перевірки наявності даних, перевірки імені таблиці перед додаванням її до бази геоданих або виконання багатьох інших корисних завдань сценаріїв. Синтаксис виклику функції arcpy має наступний вигляд:

```
arcpy.functionName(argument1, argument2, argument3,...)
```

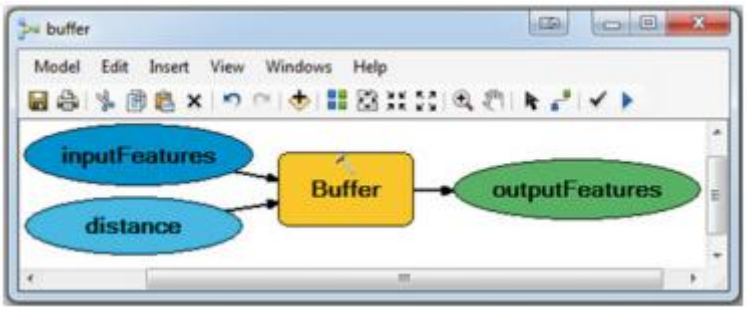


Рисунок 2.14 – Вікно ModelBuilder

Кожен інструмент ArcToolbox має налаштування, які він використовує для виконання операції, наприклад, допуск або розташування виводу. Налаштування середовища (environment settings) – це умови, які можна застосувати до всіх інструментів програми. ArcGIS має значення за замовчуванням для цих параметрів. Встановлення інших значень цих параметрів можна здійснювати за допомогою команд Python. Пакет arcpy має клас env, спеціальну структуру для зберігання пов'язаних властивостей. Властивості env керують налаштуваннями середовища. Формат для встановлення цих властивостей має наступний вигляд:

```
arcpy.env.property = value
```

Формат отримання цих властивостей:

[1)] [18] Laura Tateosian. Python for ArcGIS. Springer International Publishing Switzerland. 2015. p. 538

[20] Joel LawheadLearning. Geospatial Analysis with Python. Third Edition Packt Publishing. 2019 p.433

```
variable = arcpy.env.property
```

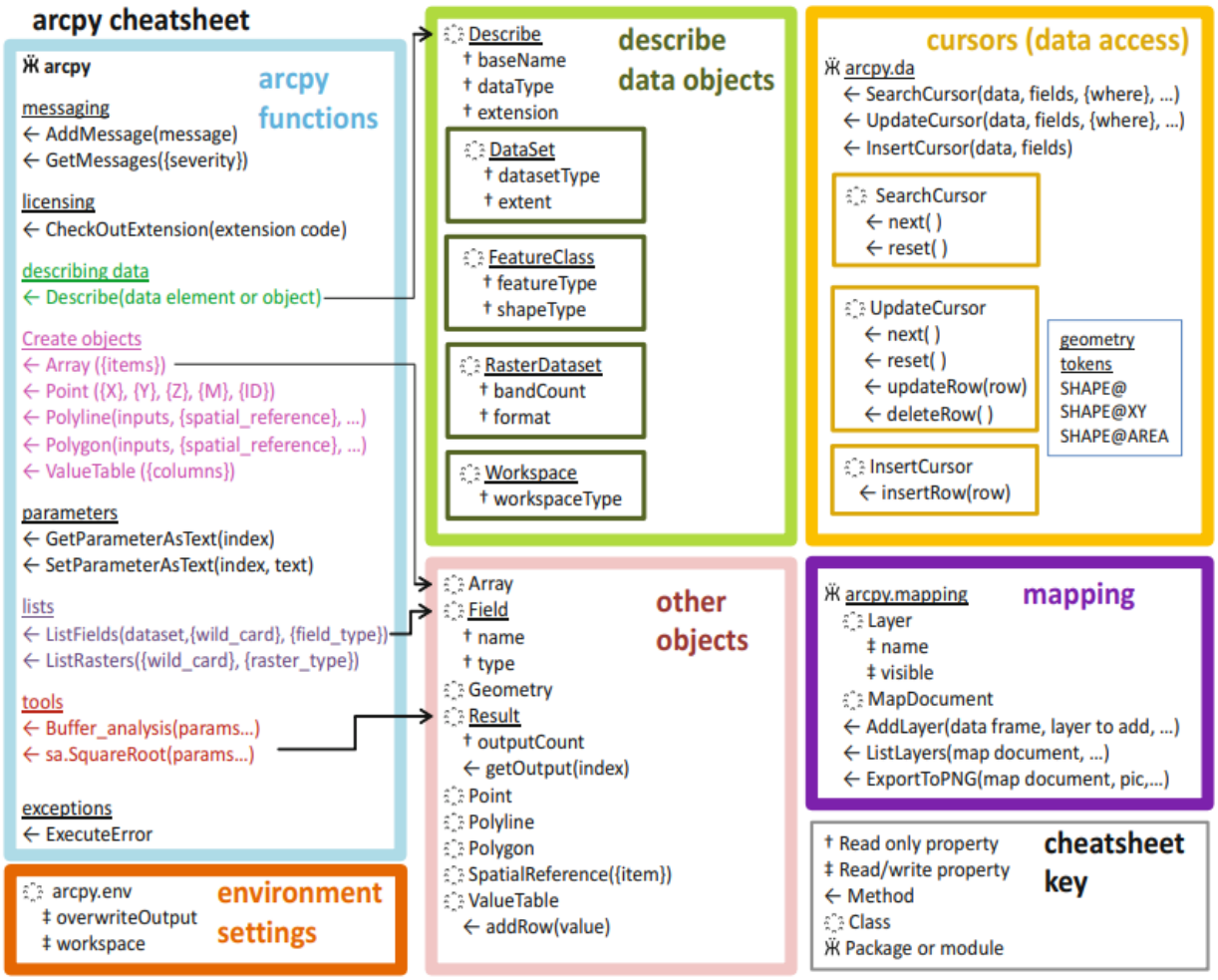


Рисунок 2.15 – Основні функціональні компоненти arcpy

3 СТВОРЕННЯ ПРОГРАМНИХ ІНСТРУМЕНТІВ ГЕООБРОБКИ ДЛЯ РОЗРАХУНКУ ХАРАКТЕРИСТИК ТРАНСПОРТНИХ МЕРЕЖ

3.1 Побудова графа дорожньої мережі в ГІС

В роботі був побудований граф дорожньої мережі для м.Одеса. В якості вихідних даних був використаний шар дорожньої мережі з твердим покриттям отриманий з картографічного веб-сервісу OpenStreetMap.

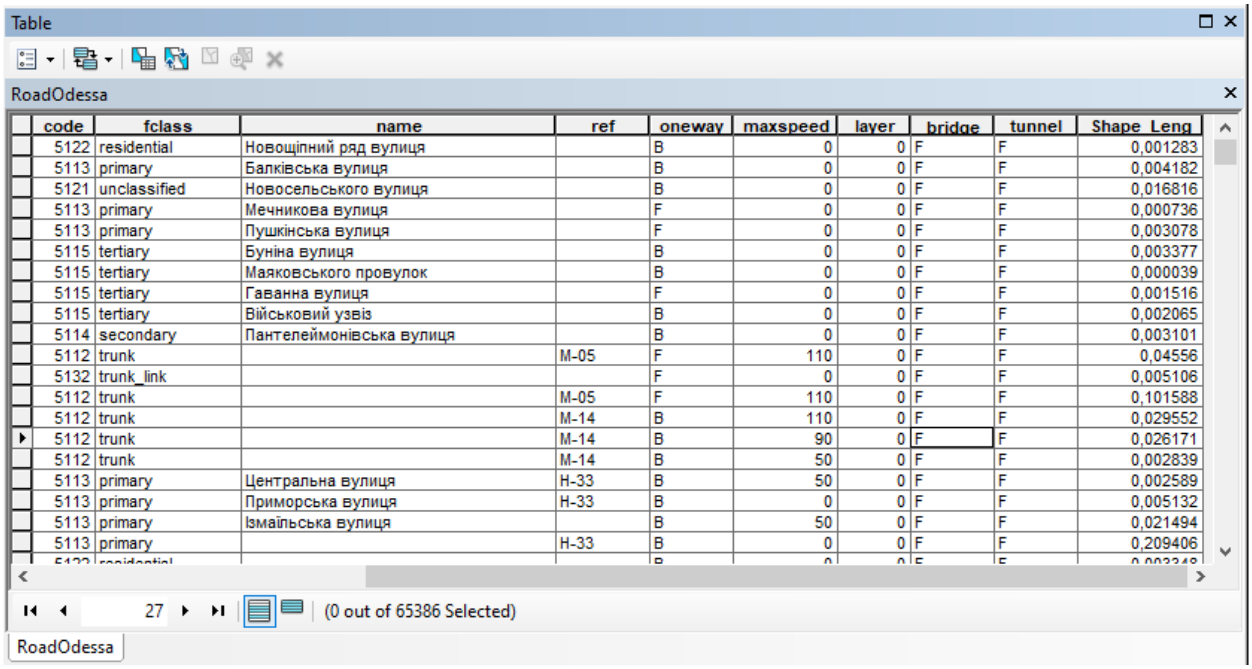
З метою проведення дослідження з пакета даних були використані наступні шари: вулично-дорожня мережа (ВДМ), об'єкти інтересу і зупинки громадського транспорту.

ВДМ – це комплект всіх доріг і вулиць населеного пункту, в даній роботі м. Одеси. Даний шар має атрибути: назва вулиці, тип дорожньої мережі, напрямок (односторонній рух чи ні), максимальна швидкість руху (за наявністю), режим швидкості (населений пункт або автомагістраль), кількість смуг, вид покриття дороги і OSM id. Фрагмент атрибутивної таблиці даного шару наведено на рис.3.1.

За типом дорожньої мережі виділені:

- secondary – основні міські магістралі;
- residential – дороги, що проходять всередині житлових зон;
- tertiary – важливі дороги місцевого значення, які з'єднують районні центри з селами, і села між собою;
- tertiary link – з'їзди з доріг;
- service – службові проїзди: всередині кварталів, в'їзні, і т.д;
- bridleway – доріжки для верхової їзди;
- construction – будується або ремонтується дорога;
- footway – пішохідні доріжки, тротуари;
- primary – автомобільні дороги регіонального призначення, вони з'єднують міста і / або обласні центри
- trunk – головні дороги, які не є автомагістралями;

– unclassified – інші автомобільні дороги місцевого значення, що входять в мережу доріг.



Table

RoadOdessa

code	fclass	name	ref	oneway	maxspeed	layer	bridge	tunnel	Shape_Leng
5122	residential	Новощіпний ряд вулиця		B	0	0	F	F	0,001283
5113	primary	Балківська вулиця		B	0	0	F	F	0,004182
5121	unclassified	Новосельського вулиця		B	0	0	F	F	0,016816
5113	primary	Мечникова вулиця		F	0	0	F	F	0,000736
5113	primary	Пушкінська вулиця		F	0	0	F	F	0,003078
5115	tertiary	Буніна вулиця		B	0	0	F	F	0,003377
5115	tertiary	Маяковського провулок		B	0	0	F	F	0,000039
5115	tertiary	Гаванна вулиця		F	0	0	F	F	0,001516
5115	tertiary	Військовий узвіз		B	0	0	F	F	0,002065
5114	secondary	Пантелеймонівська вулиця		B	0	0	F	F	0,003101
5112	trunk		M-05	F	110	0	F	F	0,04556
5132	trunk_link			F	0	0	F	F	0,005106
5112	trunk		M-05	F	110	0	F	F	0,101588
5112	trunk		M-14	B	110	0	F	F	0,029552
5112	trunk		M-14	B	90	0	F	F	0,026171
5112	trunk		M-14	B	50	0	F	F	0,002839
5113	primary	Центральна вулиця	H-33	B	50	0	F	F	0,002589
5113	primary	Приморська вулиця	H-33	B	0	0	F	F	0,005132
5113	primary	Ізмаїльська вулиця		B	50	0	F	F	0,021494
5113	primary		H-33	B	0	0	F	F	0,209406

27 (0 out of 65386 Selected)

RoadOdessa

Рисунок 3.1 – Фрагмент атрибутивної таблиці шару вулично-дорожньої мережі м. Одеса

Далі розглянуті атрибути шару зупинки громадського транспорту. Він містить: позначення виду зупинки (автобусна, тролейбусна або трамвайна), назва зупинки, OSM id. Фрагмент атрибутивної таблиці даного шару наведено на рис.3.2.

Наступні дані, які використовувалися це точки інтересів. Цей шар включається в себе таку інформацію як: назва об'єкта, вид точки інтересів (магазин, аптека і т.д.), OSM id. На рис.3.3 зображені атрибути шару точок інтересів.

Розглянемо докладніше алгоритм створення графу. Дорожня мережа представляє собою сукупність лінійних об’єктів, тому для створення графу треба перетворити їх в один лінійний об’єкт. Для цього в атрибутивній таблиці необхідно створити окреме поле типу double та виконати для нього операцію Dissolve.

Table

Transport

FID	Shape *	OBJECTID	osm_id	code	fclass	name
72	Point	73	1979440924	5621	bus_stop	
73	Point	74	1979465944	5601	railway_station	Житомирська
74	Point	75	1990091759	5621	bus_stop	вул. Степова (на вимогу)
75	Point	76	2059516126	5621	bus_stop	Вул. Космонавтів
76	Point	77	2078043543	5621	bus_stop	Вул. Філатова
77	Point	78	2078043550	5621	bus_stop	Вул. Ак. Заболотного
78	Point	79	2123016296	5601	railway_station	Одеса-Порт
79	Point	80	2136969522	5621	bus_stop	Куликове поле
80	Point	81	2141501012	5621	bus_stop	Вул. Сеченова
81	Point	82	2212765700	5621	bus_stop	Свято-Іверський монастир
82	Point	83	2225435146	5621	bus_stop	Санаторій "Якір"
83	Point	84	2225435150	5621	bus_stop	Вул. Довга
84	Point	85	2321040866	5621	bus_stop	Вул. Акордна
85	Point	86	2321040882	5621	bus_stop	Вул. Акордна
86	Point	87	2323060404	5621	bus_stop	Вул. Бугаївська
87	Point	88	2323284214	5621	bus_stop	Ф-ка технічних тканин
88	Point	89	2324928111	5621	bus_stop	Ф-ка технічних тканин
89	Point	90	2324928112	5621	bus_stop	Вул. Млинова
90	Point	91	2379818505	5621	bus_stop	Аеропорт
91	Point	92	2388597832	5603	tram_stop	Лузановка

1 (0 out of 975 Selected)

TransportP | Transport | Pois

Рисунок 3.2 – Фрагмент атрибутивної таблиці шару зупинок транспорту

Table

Pois

FID	Shape *	OBJECTID	osm_id	code	fclass	name
171	Point	172	2108483776	2005	post_office	Відділення зв'язку № 13
172	Point	173	2112539230	2401	hotel	Royal Street
173	Point	174	2113386255	2005	post_office	Відділення зв'язку № 3
174	Point	175	2113415462	2724	memorial	Гармата фрегата "Тігр"
175	Point	176	2113415467	2724	memorial	Пам'ятник-фонтан О. С. Пушкіну
176	Point	177	2113415990	2742	viewpoint	
177	Point	178	2113597245	2401	hotel	Особняк
178	Point	179	2113660584	2401	hotel	Корона
179	Point	180	2115470032	2084	college	Комп'ютерна Академія Шаг
180	Point	181	2115533119	2082	school	ЛСІ - LSE London School of English
181	Point	182	2115557494	2724	memorial	Пам'ятник-стул І. А. Ільфу та Є. П. Петрову
182	Point	183	2115557499	2724	memorial	Пам'ятник Л. О. Утьосову
183	Point	184	2117689144	2401	hotel	Адмірал
184	Point	185	2131904906	2724	memorial	Крила Перемоги
185	Point	186	2131940335	2005	post_office	Відділення зв'язку № 26
186	Point	187	2131989759	2005	post_office	Відділення зв'язку № 114
187	Point	188	2134712032	2601	bank	Восток
188	Point	189	2135914055	2504	mall	ТРЦ FONTAN SKY
189	Point	190	2135914057	2203	cinema	Cinema City
190	Point	191	2135936254	2511	convenience	

1 (0 out of 3780 Selected)

TransportP | Transport | Pois

Рисунок 3.3 – Фрагмент атрибутивної таблиці шару точок інтересів

Далі для отримання графу, що має ребра треба використати інструмент геообробки Feature To Line з бібліотеки Data Management Tools. Після виконання цього інструменті отримаємо граф, що має певну кількість ребер, кожне ребро є окремим лінійним об'єктом. Щоб визначити для кожного ребра його довжину, треба в атрибутивній таблиці створити окреме поле Length типу float и виконати для нього розрахунок Geometry calculate. Отриманий лінійний об'єкт ще не є повноцінним графом дорожньої мережі відповідного формату, що розпізнає пакет ArcMap. Перетворити його в формат New Network Dataset можна в ArcCatalog викликавши для файлу контекстне меню. Таким чином, було створено граф дорожньої мережи м. Одеси, що має 32940 ребер та 24695 вершин.

Для автоматизації процесу побудови графі дорожньої мережі в пакеті ArcMap був створений автономний скрипт Python. Блок-схема його програмного коду наведена на рис.3.1, а сам програмний код представлений у додатку А. У скрипті були використані функції бібліотеки arcpy та os. Скрипт приймає в якості вихідних даних лінійний шар дорожньої мережі міста у форматі shapefile та шлях до робочої директорії, в якій будуть зберігатися всі проміжні вихідні файли. Інтерфейс створеного скрипта наведено на рис.3.2. Інструмент є автономним скриптом, що запускається з ArcCatalog. Крім того, на рис.3.2 можна побачити зміст отриманої атрибутивної таблиці, що містить вершини графа.

Отриманий граф дорожньої мережі м. Одеса представлено на рис.3.3.

Крім того окремо був побудований граф для залізничної мережі м.Одеса, для якого були розраховані діаметр та топологічний центр. Для цього згідно алгоритму, наведеному в п.1.4 необхідно скласти матрицю найкоротших відстаней графу. Це завдання вирішувалося за допомогою функції New Cost Matrix інструменту Network Analyst. Для знаходження діаметру графу було знайдено найбільше значення в матриці найкоротших відстаней (значення в полі Total_lenght). Фрагмент отриманої матриці найкоротших відстаней з упорядкованими за спаданням відстанями наведено на рис.3.4.

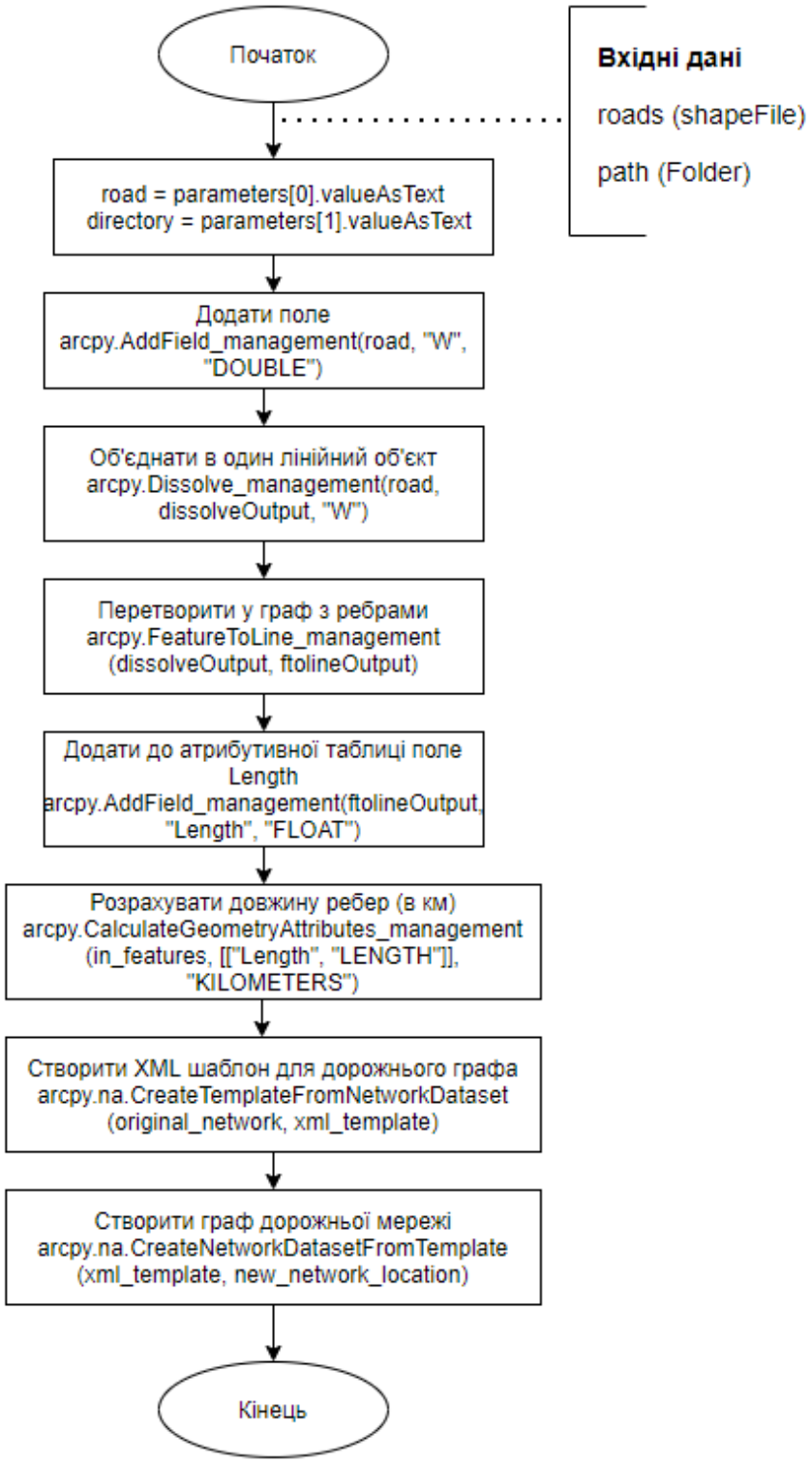


Рисунок 3.1 – Блок-схема алгоритму побудови дорожнього графа

Максимальна відстань з’єднує 1 і 1557 вершини графу залізничої мережі і складає 34.026 км. За допомогою функції New Route інструменту Network Analyst був побудований маршрут між цими вершинами, який представлений на рис.3.5.

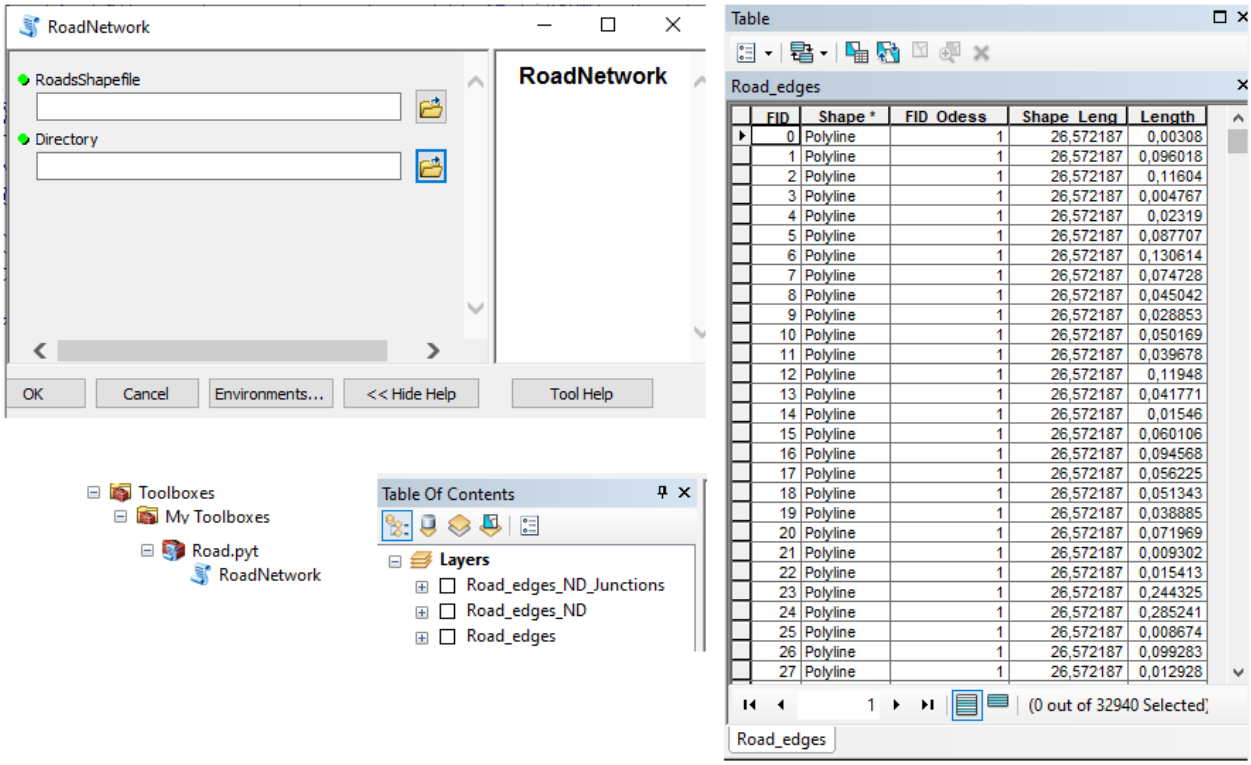


Рисунок 3.2 – Атрибутивна таблиця графа після запуску скрипта

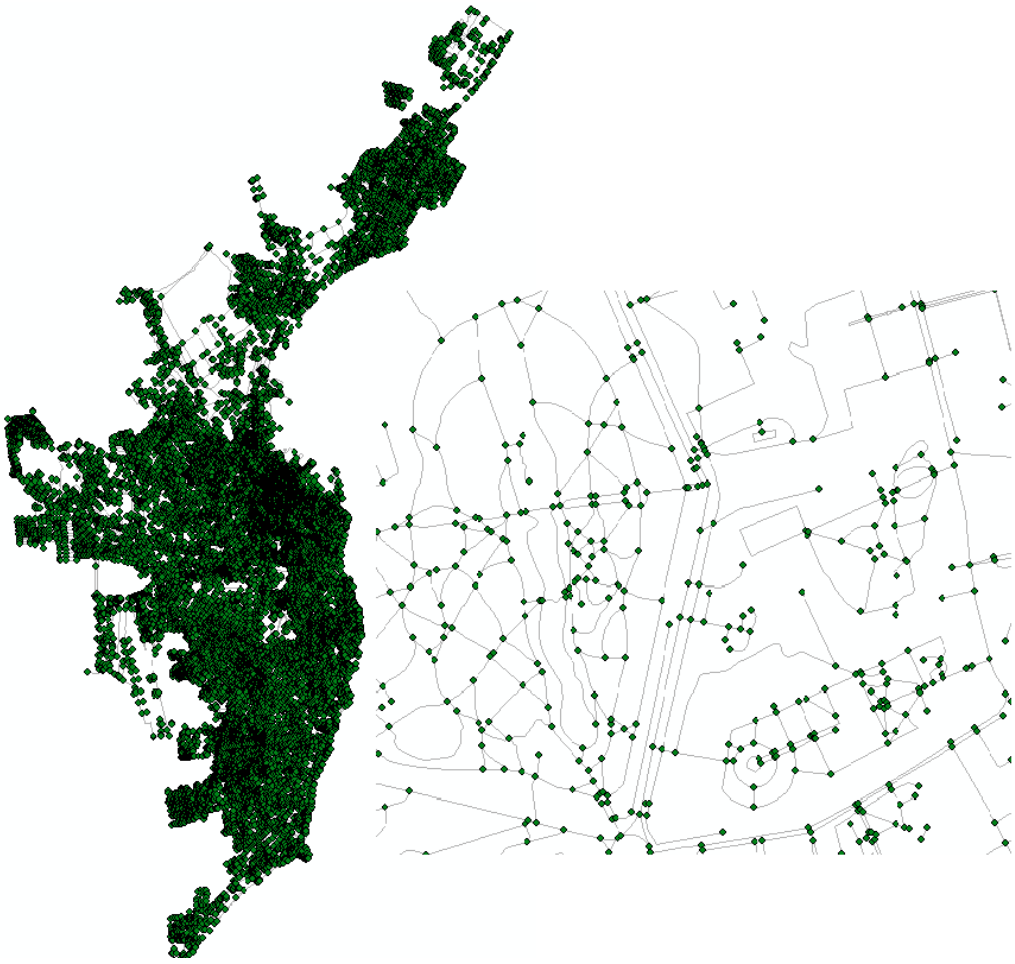
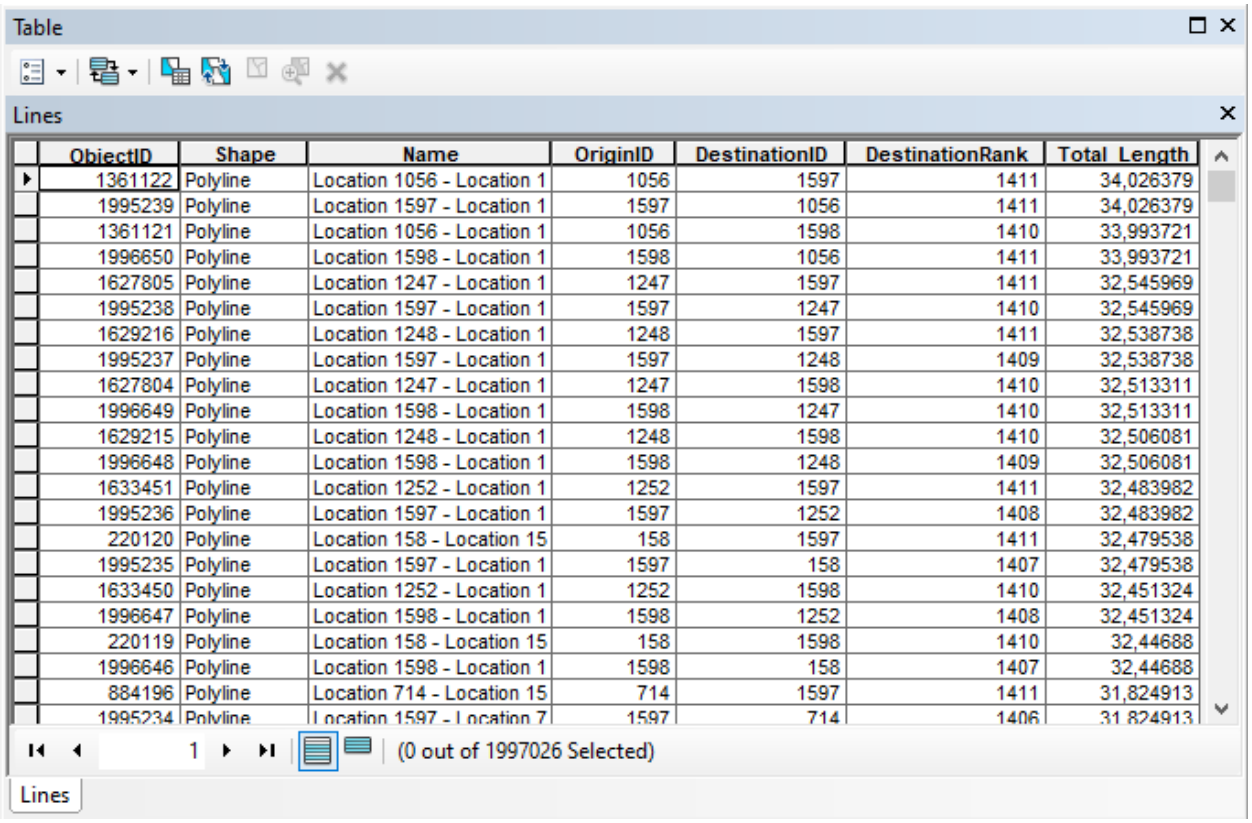


Рисунок 3.3 – Граф дорожньої мережі м. Одеси



ObjectID	Shape	Name	OriginID	DestinationID	DestinationRank	Total_Length
1361122	Polyline	Location 1056 - Location 1	1056	1597	1411	34,026379
1995239	Polyline	Location 1597 - Location 1	1597	1056	1411	34,026379
1361121	Polyline	Location 1056 - Location 1	1056	1598	1410	33,993721
1996650	Polyline	Location 1598 - Location 1	1598	1056	1411	33,993721
1627805	Polyline	Location 1247 - Location 1	1247	1597	1411	32,545969
1995238	Polyline	Location 1597 - Location 1	1597	1247	1410	32,545969
1629216	Polyline	Location 1248 - Location 1	1248	1597	1411	32,538738
1995237	Polyline	Location 1597 - Location 1	1597	1248	1409	32,538738
1627804	Polyline	Location 1247 - Location 1	1247	1598	1410	32,513311
1996649	Polyline	Location 1598 - Location 1	1598	1247	1410	32,513311
1629215	Polyline	Location 1248 - Location 1	1248	1598	1410	32,506081
1996648	Polyline	Location 1598 - Location 1	1598	1248	1409	32,506081
1633451	Polyline	Location 1252 - Location 1	1252	1597	1411	32,483982
1995236	Polyline	Location 1597 - Location 1	1597	1252	1408	32,483982
220120	Polyline	Location 158 - Location 15	158	1597	1411	32,479538
1995235	Polyline	Location 1597 - Location 1	1597	158	1407	32,479538
1633450	Polyline	Location 1252 - Location 1	1252	1598	1410	32,451324
1996647	Polyline	Location 1598 - Location 1	1598	1252	1408	32,451324
220119	Polyline	Location 158 - Location 15	158	1598	1410	32,44688
1996646	Polyline	Location 1598 - Location 1	1598	158	1407	32,44688
884196	Polyline	Location 714 - Location 15	714	1597	1411	31,824913
1995234	Polyline	Location 1597 - Location 7	1597	714	1406	31,824913

Рисунок 3.4 – Матриця найкоротших відстаней для графу залізничної мережі м.Одеса

Для знаходження топологічного центру графа відповідно до алгоритмів, що були наведені у розділі 1 магістерської роботи, треба знайти таку вершину графа, сума найкоротших відстаней от якої до інших вершини є мінімальною. Для цього треба відсортувати за зростанням таблицю по полю OriginID та DestinationID. Далі виконуємо операцію Desolve по полю OriginID з розрахунком сум за полем DestinationID. В отриманій таблиці знаходимо номер вершини з мінімальним значенням суми. Ця вершина і є топологічним центром графа. Таким чином, в роботі були розраховані основні характеристики графа залізничної мережі за допомогою вбудованого набору інструментів Network Analyst для мережевого аналізу. Треба відмити необхідність ретельної підготовки вихідних даних. Обчислення характеристик графу, що має велику кількість вершин та ребер, вимагає великих обчислювальних потужностей комп’ютера.

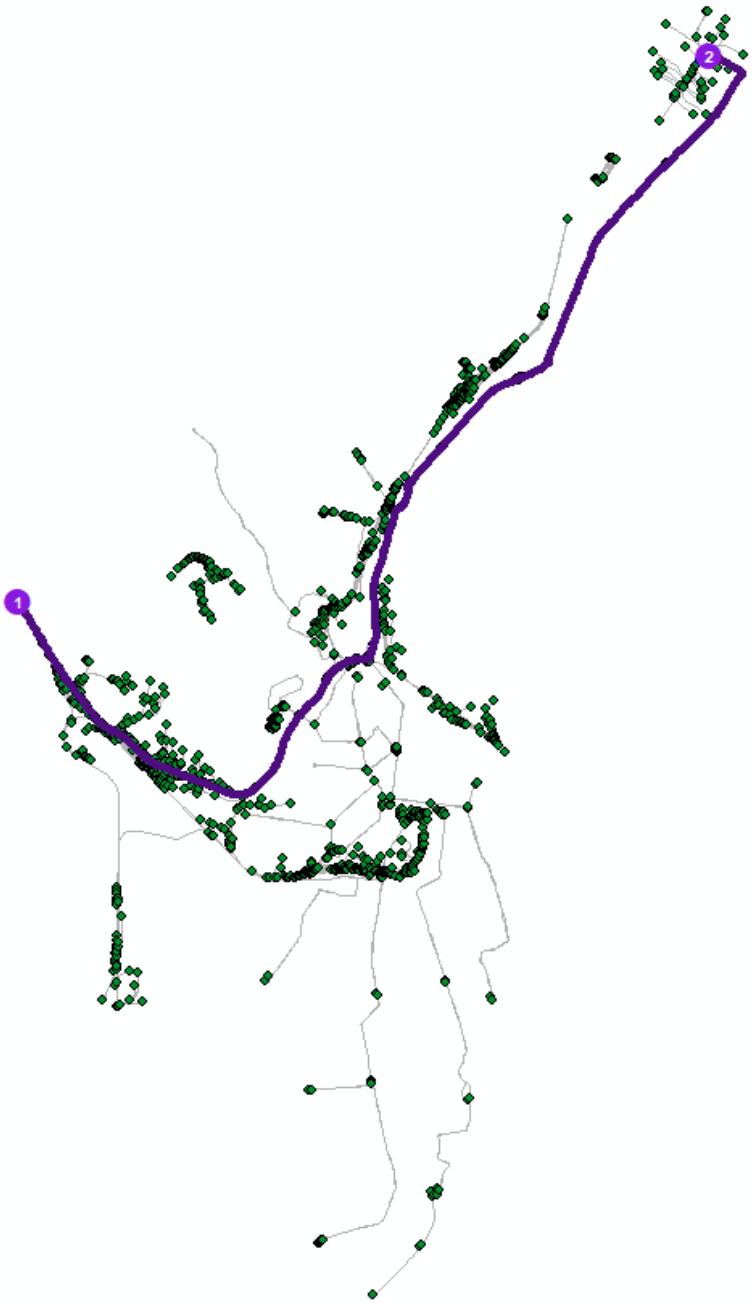


Рисунок 3.5 – Діаметр графу залізничної мережі м.Одеса

3.2 Побудова та аналіз зон доступності

Відстань об'єктів, які є значущими для населення з різних кутів зору від транспортних об'єктів можуть бути проаналізовані з урахуванням різних факторів і будуть залежати від об'єкта, цілей і масштабу дослідження.

При масштабному аналізі не варто забувати, що потрібно не тільки зображувати віддаленість від точки безпосереднього доступу до транспортних об'єктів (зупинки і т.д.), але і розраховувати це з різних ракурсів, як по прямій лінії (прямий маршрут), так і ламані лінії (реальні маршрути, маршрути з пересадками і т.п.), а також враховувати впливають фактори прохідності території (наприклад, занедбані промислові території, незабудовані території і т.д.).

В окремому випадку дослідження оцінюється доступність зупинок МПТ як об'єктів – точок доступу населення до транспортних послуг, а також віддаленість за критерієм часу.

Критерієм ефективності функціонування транспортної системи та локального рівня забезпеченості населення транспортними послугами обраний критерій доступності зупинок МПТ.

При цьому логічно, що маршрути МПТ пронизують міську територію як щільна транспортна мережа, внаслідок чого, доступність зупинок МПТ є не тільки критеріальним показником ефективності транспортних комунікацій, а й малою необхідною умовою використання транспортних послуг.

У масштабі дослідження доступність зупинок в м. Одеса була розрахована на основі критерію віддаленості зупинок, тобто розрахований шлях жителя до зупинки по існуючих вулицях, а не абстрактно на основі тільки радіусу, що задається.

Розрахунок доступності виконувався для точкових об’єктів, що є зупинками міського транспорту. Карта розташування зупинок на території міста наведена на рис. 3.6.

Результат розрахунку представлений на рис.3.7, жовтим кольором позначені об'єкти з доступністю: 0-300 метрів, коричневим кольором – 300-500 метрів, червоним – максимально недоступні з відстанню 800 метрів, сірим кольором виділені кордони м. Одеса.

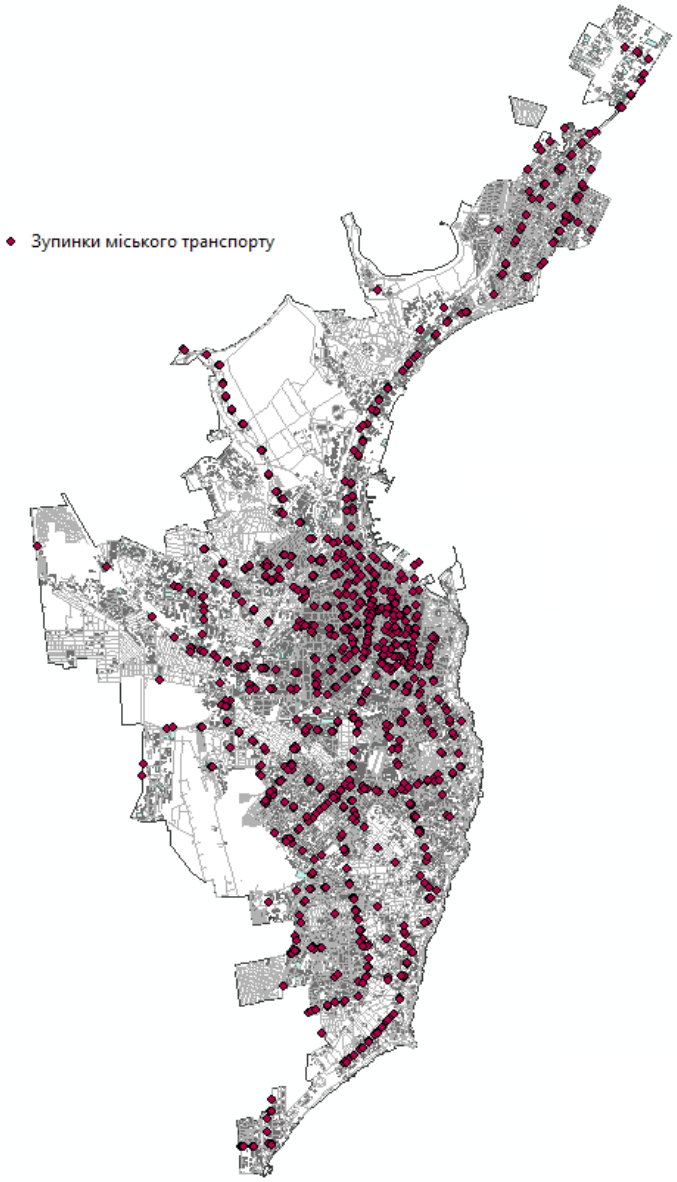


Рисунок 3.6 – Карта розташування зупинок на території м. Одеса

Нижче представлений алгоритм розрахунку доступності зупинок:

1) Завантаження зупинок громадського транспорту у вигляді точкового шару;

2) Завантаження графа транспортної мережі (лінійний шар) зваженого за довжиною ребер;

3) За допомогою інструменту Network Analyst створюється нова область обслуговування і завантажується положення (в нашому випадку зупинки);

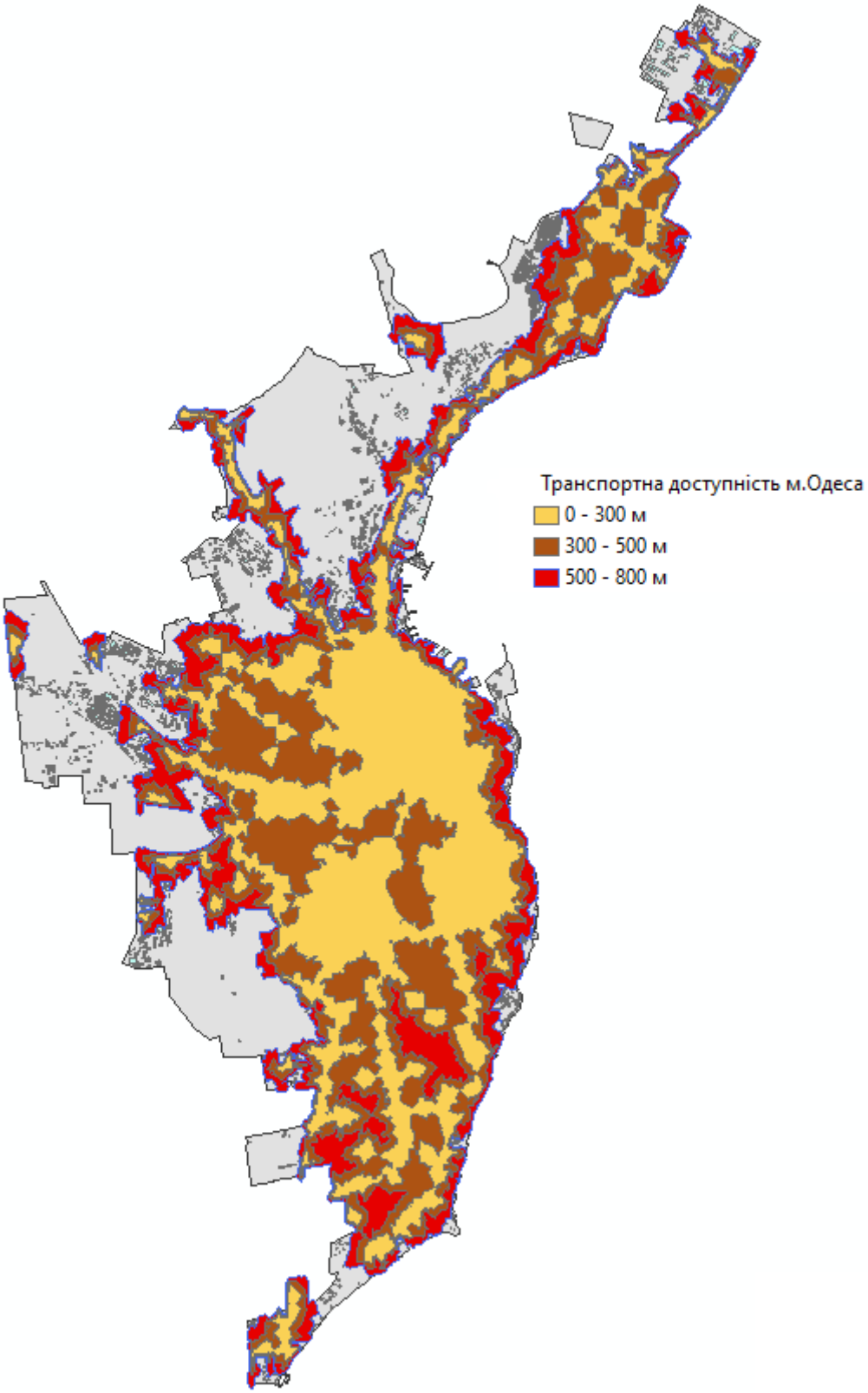


Рисунок 3.6 – Доступність зупинок міського транспорту м.Одеса

4) У властивостях області обслуговування задаємо необхідні параметри: кордону за замовчуванням "300", "500", "800", "об'єднувати по граничним значенням", "Кільця";

5) Далі виконується "Розрахунок" області обслуговування;

6) Наприкінці налаштовується відображення отриманої інформації (колір полігонів, товщина контуру і ін.).

Дані по площі зон доступності наведені в табл.3.1.

Таблиця 3.1 – Площа зон доступності зупинок міста Одеса

Зони доступності, м	Площа зон доступності, км^2	Площа зон доступності, %	Площа міста, км^2
0 – 300 м	48,2	29,6	162,7
300 – 500 м	35,4	21,8	
500 – 800 м	24,7	15,2	

Аналогічним способом можуть бути побудовані зони обслуговування для обраних точок інтересів. Наприклад, на рис.3.7 наведено розрахунок зон обслуговування для поштових відділень м. Одеса. Зони побудовані для 300 м, 500 м та 800 м. В Network Analyst можна задавати вартість чи вагу ребер графа дорожньої мережі по-різному. В роботі в якості ваги була використана відстань в км між вершинами, які з'єднує ребро графа, але можна визначати вартість переміщення в годинах руху. Крім того можна встановити різну вартість для руху вулицею в різних напрямках.

3.3 Розробка скриптів Python для автоматизованої побудови маршрутів

При побудові великої кількості маршрутів з використанням послідовності функцій інструменту Network Analyst може виникнути проблема обме-

ження обчислювальних ресурсів та завантаження процесора при перемалю-ванні проміжних результатів на карті.

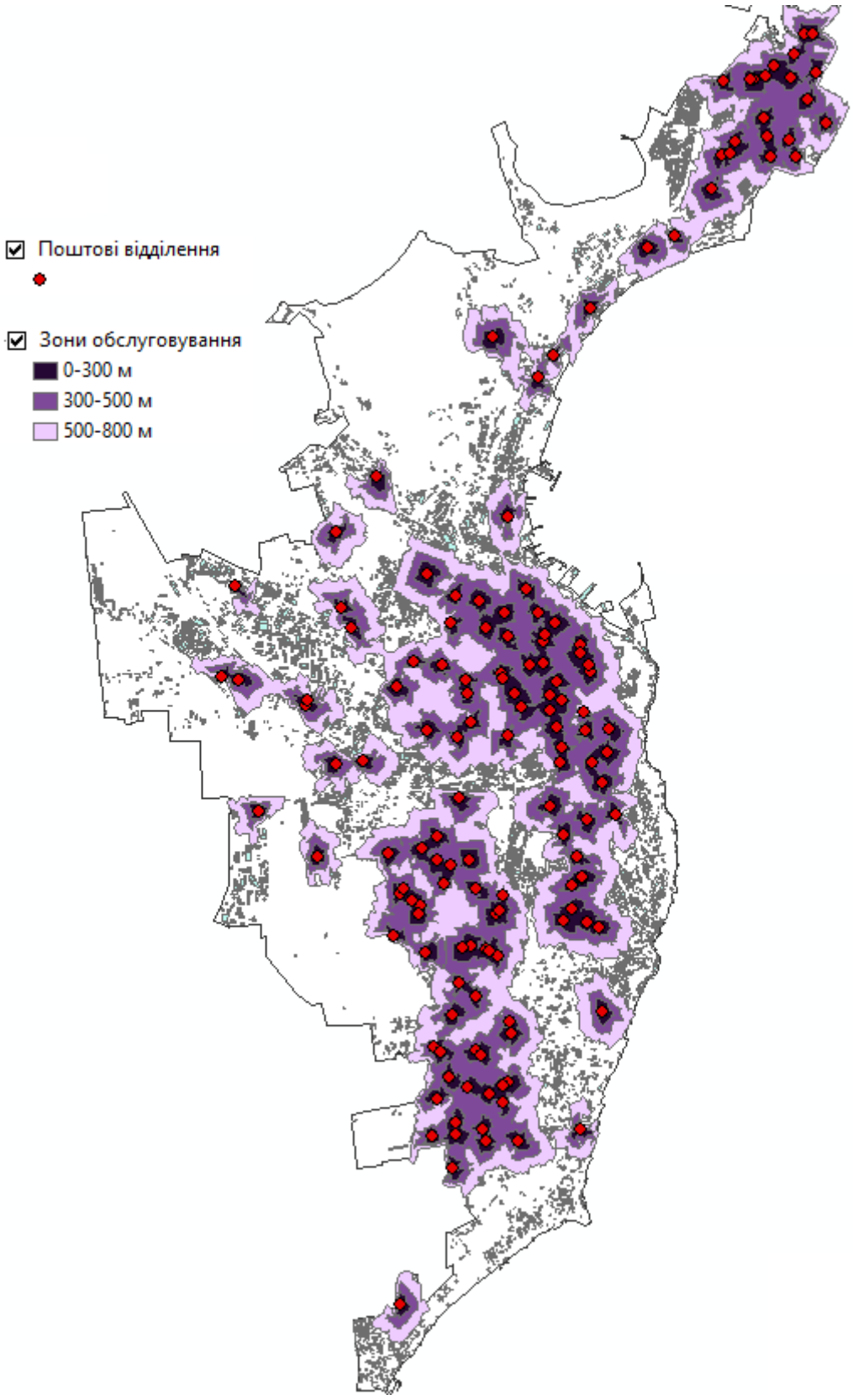


Рисунок 3.7 – Зони обслуговування поштових відділень м.Одеса

Щоб уникнути зазначеної проблеми корисним буде розробити власне програмне забезпечення у вигляді автономного скрипта Python з графічним інтерфейсом користувача для автоматизації процесу побудови маршрутів на графі дорожньої мережі. Відметемо, що найбільший ефект від використання подібного ПЗ буде відчутно саме при побудові великої кількості маршрутів. Ця задача зустрічається доволі часто, тим паче що як правило великий маршрут будується за великою кількістю точок, тому доводиться складати його з маленьких шляхів між сусідніми точками.

Розглянемо основні елементи програми і функції Python, що дозволяють програмно працювати з інструментом Network Analyst.

Для створення об'єкта маршруту треба використовувати функцію бібліотеки Arcpy з трьома параметрам: графом дорожньої мережі, назвою маршруту та імпеданс параметром.

```
route="testroute"
arcpy.na.MakeRouteLayer("Road_enges_ND", route, "Length")
```

Далі необхідно задати точки через які буде побудовано маршрут. Вони будуть зберігатися в окремому точковому шарі, яки треба створити у тимчасовій пам'яті комп'ютера з можливістю її перезапису:

```
arcpy.env.overwriteOutput = True
arcpy.CreateFeatureClass_managment("in_memory", "temp",
"POINT")
```

Далі в створений шар треба додати координати точок відправлення та призначення маршруту. В атрибутивній таблиці шару `temp` створимо текстове поле `Name` в якому буде зберігатися назви точок:

```
arcpy.AddField_management("in_memory/temp", "Name", "TEXT")
icurs = arcpy.da.InsertCursor("in_memory/temp",["SHAPE@XY",
"Name"])
icurs = insertRow([(32.6649982, 51.50755657),"Вул.
Ген.Петрова"])
...
icurs = insertRow([(32.5401146, 51.4713615),"ТЦ Мегадом"])
```

Далі треба позначити зміну `route` як шар просторових даних:

```
routeLayer = arcpy.mapping.Layer(route)
```

Отримуємо перелік об’єктів полів шару точок (станцій), вкажемо, що будемо працювати з зупинками (Stops):

```
fields =  arcpy.ListFields(“in_memory/temp”)
fieldMappings =  arcpy.na.NAClassFieldMappings(routeLayer, ‘Stops’, False, fields)
arcpy.na.AddLocations(route,”Stops”,“in_memory/temp”, fieldMappings)
```

Далі можна побудувати найкоротший маршрут між точками по існуючій дорожній мережі (рис.3.8):

```
arcpy.na.Solve(route)
```

Крім точок також можуть бути додані бар’єри, які можуть бути точковими, лінійними або полігональними, в залежності від задачі.

```
Fields1 =  arcpy.ListFields(“Restrictions”)
restrictionsMappings =  arcpy.na.NAClassFieldMappings (routeLayer, ‘Point Barriers’, False, fields)
arcpy.na.AddLocations(route,”Point Barriers”, “Restrictions”, restrictionsMappings)
```

Далі можна перебудувати маршрут з урахуванням доданих точок бар’єрів (рис.3.9):

```
arcpy.na.Solve(route)
```

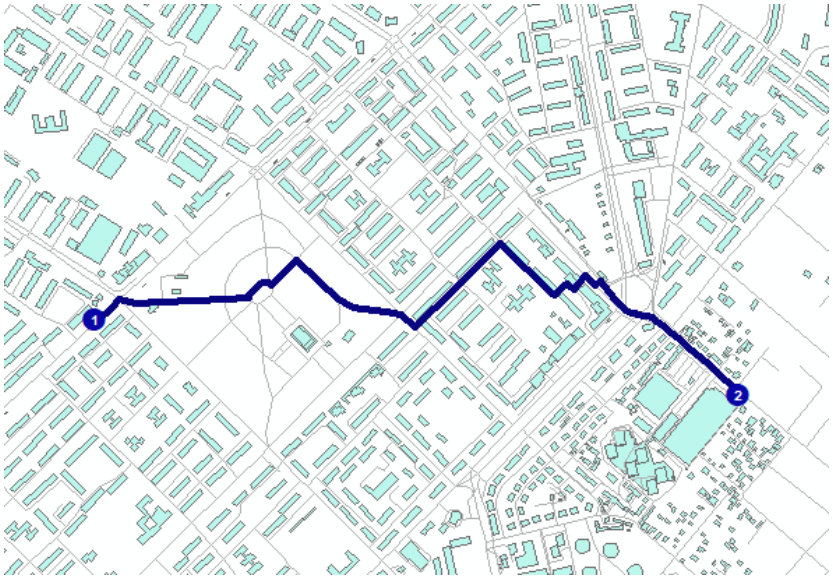


Рисунок 3.8 – Найкоротший маршрут між заданими точками

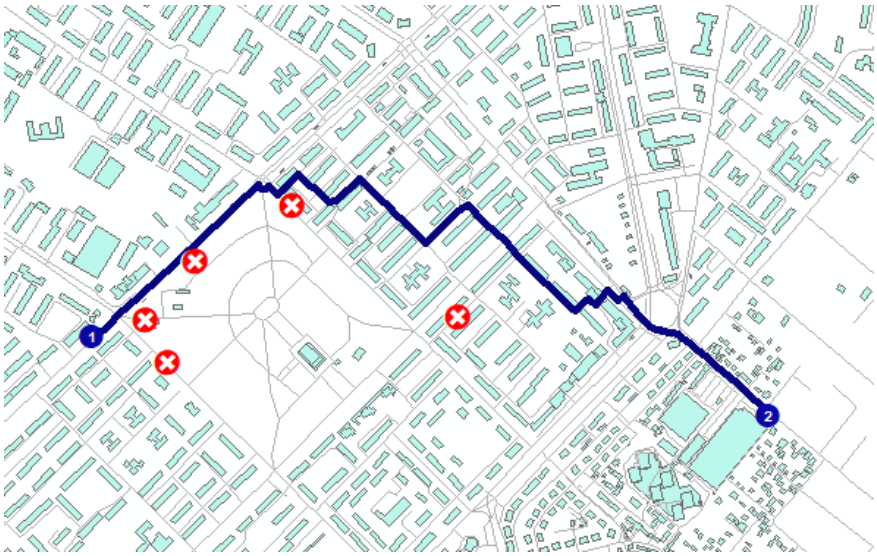


Рисунок 3.8 – Найкоротший маршрут між заданими точками з врахуванням бар’єрів

Наведений фрагмент програмного код можна обгорнути у функцію і додати словники точок, щоб за їх назвою завантажувати автоматично їх координати у функцію.

Ще одна можливість інструменту Network Analyst, яка не була розглянута раніше – це можливість находження найкоротшого маршруту до заданої точки інциденту від декількох відправних точок. Розглянемо приклад коли задано 6 точок, які можуть бути будь-яким пунктами швидкого реагування (швидкою допомогою, пожежними частинами, пунктами охорони тощо) і є точка інциденту, в якій, наприклад, сталася пожежа. Треба знайти одну найближчу відправну точку, яка знаходиться в заданому радіусу (наприклад, 1 км) і побудувати маршрут від неї то точки інциденту по заданому графу дорожньої мережі. Фрагмент карти з таким маршрутом наведена на рис. 3.9.

Якщо змінити умови задачі та побудувати всі маршрути від відправних точок, що лежать в радіусі не більше 1 км, то отримаємо карту, що наведена на рис.3.10. Побудова подібних рішень можлива завдяки використанню функції New Closest Facility інструменту Network Analyst. В подібні задачі можна додавати також точки бар’єри, як це було показано раніше.

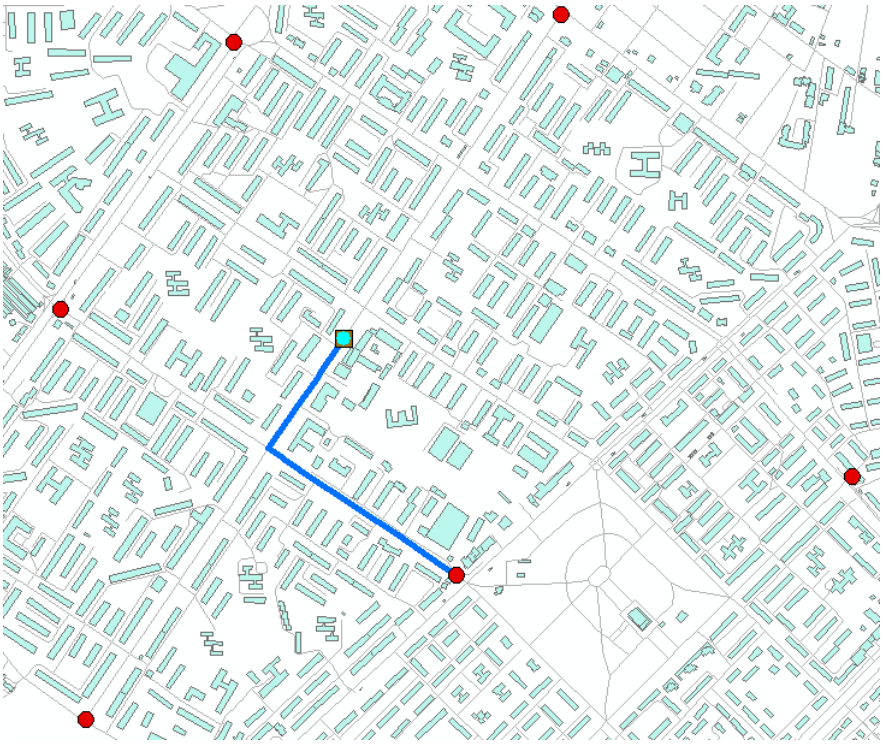


Рисунок 3.9 – Найкоротший маршрут до точки інциденту

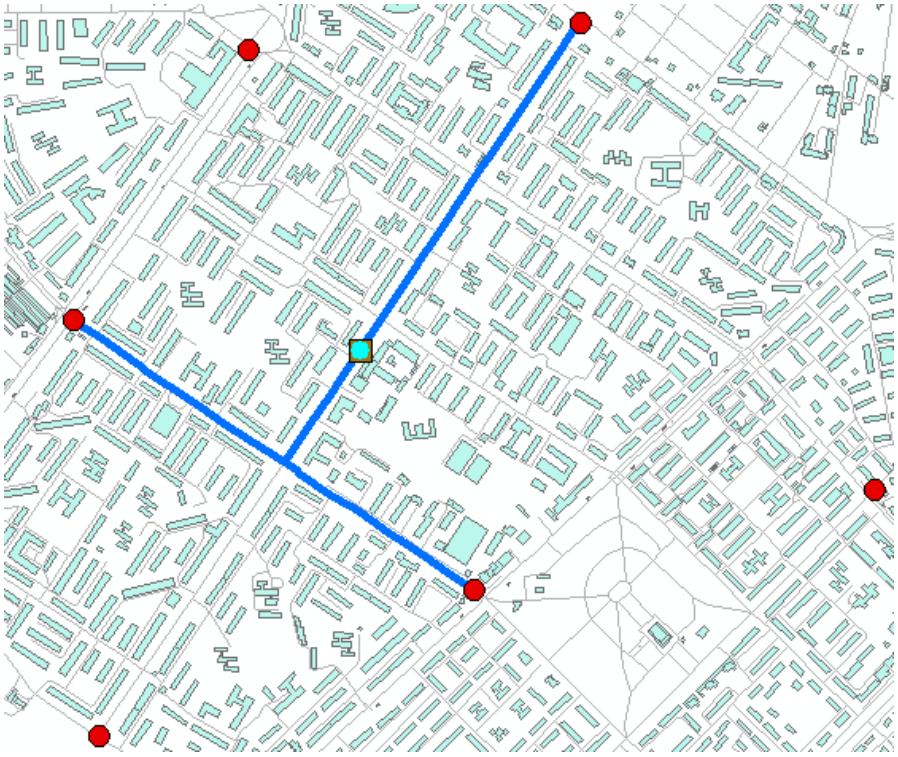


Рисунок 3.10 – Найкоротші маршрути до точки інциденту від відправних точок, що лежать в радіусі 1 км

Алгоритм роботи створеного на мові програмування Toolbox представлено на рис.3.11. Програмний код скрипта представлено у додатку В.

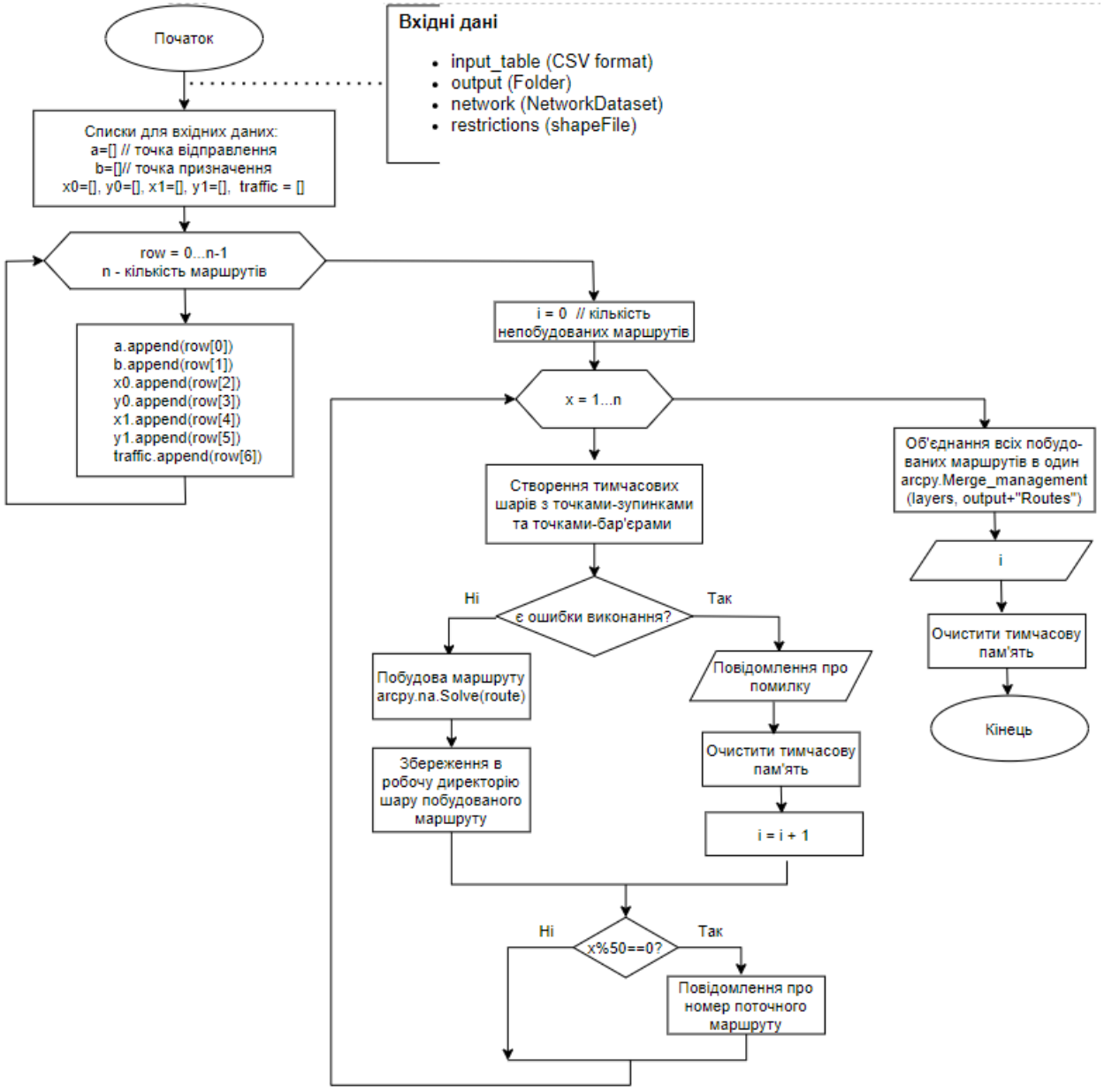


Рисунок 3.11 – Блок-схема виконання скрипта автоматизованої побудови маршрутів

В якості вхідних даних скрипт може завантажувати словники зупинок. В наведеній реалізації скрипт отримує дані з файлу excel, який має наступну послідовність колонок: точка відправлення, точка призначення, атрибут, що

характеризує маршрут (наприклад, пасажиропотік), і координати x1, y1, x2, y2 точок.

Інтерфейс інструменту представлено на рис.3.12 і має поля для задавання наступних параметрів: вхідна CSV-таблиця, директорія для збереження результатів роботи скрипта, граф дорожньої мережі, shape файл шар обмежень.

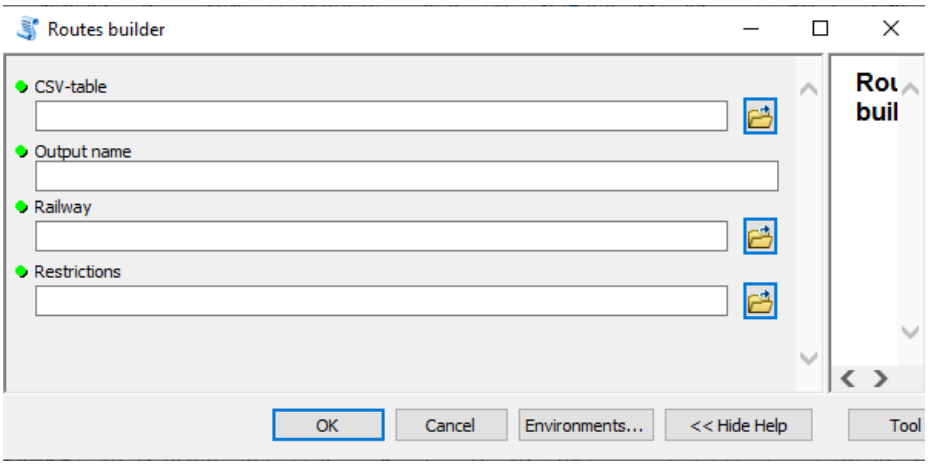


Рисунок 3.12 – Інтерфейс скрипта

ВИСНОВКИ

Інтелектуальний аналіз даних – це напрямок інформаційних технологій, що охоплює всю область проблем, пов'язаних з отриманням знань з масивів даних. В наш час накопичується велика кількість саме геопросторової інформації, в результаті аналізу якої можуть бути отримані знання для забезпечення підтримки прийняття рішень в різних галузях. ГІС є потужним інструментів для проведення інтелектуального аналізу геоданих. В даній роботі методи та засоби ГІС-аналізу були досліджені та використані для вирішення різних завдань організації транспортної системи міста.

Транспортна доступність є одним з ключових чинників при оцінці якості міського середовища. Завдання планування потоку в транспортній мережі відносяться до одного з можливих класів задач, що вирішуються за допомогою графів. Граф дозволяє наочно відобразити взаємозв'язки об'єктів реальної системи.

В роботі було виконано математичний опис та дослідження графової моделі дорожньої мережі, а також розглянуті алгоритми знаходження характеристик графа: радіуса, діаметру та центру. Наведені відповідні приклади. Розглянуті та вивчені види джерел геоінформаційних даних для побудови графу дорожньої мережі і моделювання транспортної доступності. Виконано аналіз функціональних можливостей вбудованих засобів ГІС для моделювання дорожньої мережі міста. Досліджені функції Python бібліотеки arcpy для аналізу дорожніх мереж.

Розроблені власні автономні скрипти Python для автоматизованої побудови маршрутів та графа дорожньої мережі. Виконано тестування створеного програмного інструменту в ГІС для вирішення різних транспортних завдань (на прикладі дорожньої мережі міста Одеса): знаходження найкоротших маршрутів до точки інциденту з врахування бар’єрів, аналіз транспортної доступності зупинок, аналіз зон обслуговування поштових відділень.

ПЕРЕЛІК ПОСИЛАНЬ

1) Горев, А. Э. Основы теории транспортных систем: учеб. пособие. СПбГАСУ. СПб., 2010. 214 с.

2) Доля К. В., Доля О.Є. Геоінформаційні системи на транспорті: навч. посібник; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. Харків: ХНУМГ ім. О. М. Бекетова, 2018. 230 с.

3) Якубович А.Н., Куфтинова Н.Г., Рогова О.Б. Информационные технологии на автотранспорте: учебное пособие. М.: МАДИ, 2017. 252 с.

4) Трофименко, Ю.В. Якимов М. Р. Транспортное планирование: формирование эффективных транспортных систем крупных городов: монография. М.: Логос, 2013. 447 с. 19.

5) Ваксман, С.А. Информационные технологии в управлении городским общественным пассажирским транспортом (задачи, опыт, проблемы). Екатеринбург: Изд-во АМБ, 2012. 250 с.

6) Кузьменко І.М. Теорія графів. Навч. посіб. для здобувачів ступеня бакалавра за спеціальністю 122 «Комп'ютерні науки». Київ: КПІ ім. Ігоря Сікорського, 2020. 71 с.

7) О. Оре. Теория графов. М.: Наука, 1980, 336 с.

8) Ураков А.Р., Тимеряев Т.В. Использование особенностей взвешенных графов для более быстрого определения их характеристик. Прикладная Дискретная Математика. 2012, №2 (16), стр.95-100.

9) Кристофидес Н. Теория графов. Алгоритмический подход. М.: Мир, 1978, 432 с.

10) Берж К. Теория графов и ее применения. Москва: Иностранная литература, 1962. 75 с.

11) В.А. Емеличев, О.И.Мельников. Лекции по теории графов. М.: Наука, 1990. 383с.

12) С.Ю.Белецкая. Комбинаторика. Графы. Алгоритмы. Учеб.пособие. Воронеж: ВГТУ, 2003.

13) Світличний О.О., Плотницький С.В. Основи геоінформатики: Навчальний посібник. Суми: ВТД «Університетська книга», 2006. 295 с.

14) Шипулін В. Д. Основи ГІС-аналізу: навч. Посібник. Харк. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. Х.: ХНУМГ, 2014. 330 с.

15) Картографічні дані OpenStreetMap. URL: https://www.openstreetmap.org (Дата звернення 10.11.2020)

16) Особенности моделирования транспортной доступности. URL: http://smartloc.ru. (Дата звернення 10.11.2020)

17) Довідка по модулю ArcGIS Network Analyst. URL: http://desktop.arcgis.com (Дата звернення 10.11.2020)

18) Laura Tateosian. Python for ArcGIS. Springer International Publishing Switzerland. 2015. p. 538

19) Офіційний сайт Python. URL: http://www.python.org (Дата звернення 10.11.2020)

20) Joel LawheadLearning. Geospatial Analysis with Python. Third Edition Packt Publishing. 2019 p.433

ДОДАТОК А
ПРОГРАМНИЙ КОД СКРИПТА ПОБУДОВИ ГРАФА ДОРОЖНЬОЇ МЕРЕЖІ

```
import arcpy
import os
class Toolbox(object):
    def __init__(self):
        """Define the toolbox (the name of the toolbox is the name of the
        .pyt file)."""
        self.label = "Road"
        self.alias = ""
        # List of tool classes associated with this toolbox
        self.tools = [RoadNetwork]
class RoadNetwork(object):
    def __init__(self):
        """Define the tool (tool name is the name of the class)."""
        self.label = "RoadNetwork"
        self.description = ""
        self.canRunInBackground = True
    def getParameterInfo(self):
        param0 = arcpy.Parameter(
          displayName ='AreaShapefile',
          name ='in_area',
          datatype ="DEShapeFile",
          parameterType ='Required',
          direction ='Input')
        param1 = arcpy.Parameter(
          displayName ='Directory',
          name ='out_dir',
          datatype ="DEFolder",
          parameterType ='Required',
          direction ='Input')
        return [param0, param1]
    def isLicensed(self):
        """Set whether tool is licensed to execute."""
        return True
```

```
    def updateParameters(self, parameters):
        """Modify the values and properties of parameters before internal
        validation is performed.  This method is called whenever a parameter
        has been changed."""
        return
    def updateMessages(self, parameters):
        """Modify the messages created by internal validation for each tool
        parameter.  This method is called after internal validation."""
        return
    def execute(self, parameters, messages):
        road = parameters[0].valueAsText
        arcpy.AddField_management(road, "W", "DOUBLE")
        directory = parameters[1].valueAsText
        dissolveOutput = directory + '/RoadDiss.shp'
        arcpy.Dissolve_management(road, dissolveOutput, "W")
        ftolineOutput = parameters[1].valueAsText + '/ftoline.shp'
        arcpy.FeatureToLine_management(dissolveOutput, ftolineOutput)
        arcpy.Delete_management(dissolveOutput)
        arcpy.AddField_management(ftolineOutput, "Length", "FLOAT")
        arcpy.env.outputCoordinateSystem =
arcpy.Describe(ftolineOutput).spatialReference
        # Set local variables
        in_features = ftolineOutput
        arcpy.CalculateGeometryAttributes_management(in_features,
[["Length", "LENGTH"]], "KILOMETERS")
        try:
          #Check out Network Analyst license if available. Fail if the Network
Analyst license is not available.
          if arcpy.CheckExtension("network") == "Available":
             arcpy.CheckOutExtension("network")
          else:
             raise arcpy.ExecuteError("Network Analyst Extension license is
not available.")
          #Set local variables
          original_network = directory + 'Road_ND'
          new_network_location = in_features
          xml_template = directory + 'NDTemplate.xml'
          #Create an XML template from the original network dataset
```

```
        arcpy.na.CreateTemplateFromNetworkDataset(original_network,
xml_template)
        #Create the new network dataset in the output location using the
template.
        #The output location must already contain feature classes and ta-
bles with
        #the same names and schema as the original network.
        arcpy.na.CreateNetworkDatasetFromTemplate(xml_template,
new_network_location)
        #Build the new network dataset
        arcpy.na.BuildNetwork(os.path.join(new_network_location,
"Road_ND")
      except Exception as e:
        # If an error occurred, print line number and error message
        import traceback, sys
        tb = sys.exc_info()[2]
        print(("An error occurred on line %i" % tb.tb_lineno))
        print((str(e)))
      return
```

ДОДАТОК В
ПРОГРАМНИЙ КОД СКРИПТА АВТОМАТИЗОВАНОЇ ПОБУДОВИ МАРШРУТІВ

```
Import arcpy
arcpy.env.overwriteOutput = True
class Toolbox(object):
    def __init__(self):
        """Define the toolbox (the name of the toolbox is the name of the
        .pyt file)."""
        self.label = "Routes builder"
        self.alias = ""
        # List of tool classes associated with this toolbox
        self.tools = [Tool]
class Tool(object):
    def __init__(self):
        """Define the tool (tool name is the name of the class)."""
        self.label = "Routes builder"
        self.description = ""
        self.canRunInBackground = False
    def getParameterInfo(self):
        """Define parameter definitions"""
        param0=arcpy.Parameter(
         displayName="CSV-table",
         name='input_table',
         datatype='DETable',
         parameterType='Required',
         direction='Input',
         multiValue=False)
        param1=arcpy.Parameter(
         displayName="Output name",
         name='output',
         datatype='GPString',
         parameterType='Required',
         direction='Input',
         multiValue=False)
        param2=arcpy.Parameter(
         displayName="Railway",
         name='railway',
```

```
        datatype='DENetworkDataset',
        parameterType='Required',
        direction='Input',
        multiValue=False)
       param3=arcpy.Parameter(
        displayName="Restrictions",
        name='restrictions',
        datatype='DEShapeFile',
        parameterType='Required',
        direction='Input',
        multiValue=False)
       params = [param0, param1, param2, param3]
       return params
   def isLicensed(self):
       """Set whether tool is licensed to execute."""
       return True
   def updateParameters(self, parameters):
       """Modify the values and properties of parameters before internal
       validation is performed.  This method is called whenever a parame-
ter
       has been changed."""
       return
   def updateMessages(self, parameters):
       """Modify the messages created by internal validation for each tool
       parameter.  This method is called after internal validation."""
       return
   def execute(self, parameters, messages):
       """The source code of the tool."""
       input_table=parameters[0].value
       output=parameters[1].valueAsText
       network=parameters[2].value
       restrictions=parameters[3].value
       a=[]
       b=[]
       x0=[]
       y0=[]
       x1=[]
       y1=[]
       traffic = []
```

```
        for row in arcpy.da.SearchCursor(input_table, ["from", "to", "X1",
"Y1", "X2", "Y2","traffic"]):
         a.append(row[0].replace("-","_").replace(".","_").replace("
","_").replace("","").replace(")",""))
         b.append(row[1].replace("-","_").replace(".","_").replace("
","_").replace("","").replace(")",""))
         x0.append(row[2])
         y0.append(row[3])
         x1.append(row[4])
         y1.append(row[5])
         traffic.append(row[6])
        i = 0
        for x in range(len(a)):
         route = a[x]+"__"+b[x]
         arcpy.CreateFeatureclass_management("in_memory","temp",
"POINT")
         arcpy.AddField_management("in_memory/temp",'Name',"TEXT")
         icurs=arcpy.da.InsertCursor("in_memory/temp",["SHAPE@XY",
'Name'])
         icurs.insertRow([(y0[x],x0[x]), a[x]])
         icurs.insertRow([(y1[x],x1[x]), b[x]])
         arcpy.na.MakeRouteLayer(network,route,"Length")
         routeLayer = arcpy.mapping.Layer(route)
         fields = arcpy.ListFields("in_memory/temp")
         fields1 = arcpy.ListFields(restrictions)
         facilitiesSubLayerName =
arcpy.na.GetNAClassNames(routeLayer)['Routes']
         fieldMappings = arcpy.na.NAClassFieldMappings(routeLayer,
'Stops',False, fields)
         restrictionsMappings = arcpy.na.NAClassFieldMappings(routeLayer,
'Point Barriers',False, fields1)

         arcpy.na.AddLocations(route,"Stops","in_memory/temp",fieldMappi
ngs)
         arcpy.na.AddLocations(route,"Point Barri-
ers",restrictions,restrictionsMappings)
         try:
                                 arcpy.na.Solve(route)
```

```
        arcpy.CopyFeatures_management(route+"/Routes",output+route)

        arcpy.AddField_management(output+route+".shp",'Traffic',"DOUBL
E")

        ucurs=arcpy.da.UpdateCursor(output+route+".shp",["Traffic"])
                        for row in ucurs:
                                row[0] = traffic[x]
                                ucurs.updateRow(row)
        except:
                arcpy.AddMessage('no solution found for route '+route)
                with file(output + "unsolved.csv", 'a') as out:
                        out.write(a[x].encode("windows-1251")+";")
                        out.write(b[x].encode("windows-1251")+";")
                        out.write(str(x0[x])+";")
                        out.write(str(y0[x])+";")
                        out.write(str(x1[x])+";")
                        out.write(str(y1[x])+";")
                        out.write(str(traffic[x])+"\n")
                i = i + 1
        arcpy.Delete_management(route)
        if x%50==0:
                arcpy.AddMessage(str(x)+" Done")
       arcpy.AddMessage(str(i)+" unsolved in total")
       arcpy.env.workspace = output
       layers = arcpy.ListFeatureClasses()
       arcpy.Merge_management(layers, output+"Routes")
       for lyr in layers:
          arcpy.Delete_management(lyr)
```