МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Кваліфікаційна робота бакалавра

на тему: Аналіз алгоритмів для моделювання оптимального маршруту в ГІС

Виконала студентка групи К-19і
спеціальності 122 «Комп'ютерні науки»
Олійник Олександр Васильович

Керівник доктор філософії
Бучинська Ірина Вікторівна

Рецензент д.ф-м.н., професор
Ковальчук Володимир Володимирович

Одеса 2021

ЗМІСТ

СКОРОЧЕННЯ

ГІС – геоінформаційна система;

TSP − Travelling salesman problem;

NP − Nondeterministic Polynomial;

АКС – автоматизована картографічна система;

ПЗ – програмне забезпечення;

ГІС – географічна інформаційна система;

СУБД – система управління базами даних;

OGC – Open Geospatial Consortium;

VRP – Vehicle routing problem.

ВСТУП

Завдання побудови оптимального маршруту, вперше описане в 1832 році в книзі «Комівояжер − як він повинен вести себе і що повинен робити для того, щоб доставляти товар і мати успіх у своїх справах – поради старого кур'єра», є актуальною і на сьогоднішній день розробляються нові методи розв'язання задачі, реалізуються програми, які дозволяють працювати з кількістю вузлів близьким до мільйону за прийнятний час. Незгасний інтерес до цього завдання обумовлений різноманітністю застосувань її на практиці. Пошук оптимального шляху широко використовується у всіх завданнях транспортної логістики, на виробництві – у вигляді задач мінімізації часу переналагодження і при роботі діропробивні преси. [1]

Питання оптимізації маршруту піднімаються в працях таких відомих діячів математики як Вільям Роуен Гамільтон, Джордж Данциг, Річард Карп, Девід Аплгейт, Герхард Райнельт. [1]

Незважаючи на велику теоретичну базу, на жаль, представлено не так багато додатків, що дозволяють людям, далеким від математики і програмування, використовувати існуючі розробки для вирішення практичних завдань. Користувач, який хоче обійти n-ну кількість місць за мінімальний час, використовуючи такі популярні картографічні сервіси як «Яндекс.Карти» [2] і «Google Карти» [3], не може вирішити поставлене завдання, так як маршрут в додатку будується по заздалегідь заданому в певному порядку списку місць.

Мета даної роботи полягає в аналізі існуючих алгоритмів моделювання оптимального маршруту в ГІС та реалізації завдання комівояжера з використанням технології ArcGIS [4].

Для досягнення зазначеної мети поставлені наступні завдання:

– розглянути основні алгоритми пошуку оптимального маршруту;

– провести аналіз даних алгоритмів і вибрати один з них для практичної реалізації;

– на основі отриманих даних показати результати, які дозволяють користувачам отримати оптимальний шлях між заданими пунктами.

Актуальність даної роботи полягає в аналізі алгоритмів, оптимальних для поставленого завдання, і створенні зручного додатку з пошуку оптимального шляху для гідів, екскурсоводів і потребують швидкому доступі до різноманітних маршрутах туристів. Практична значимість роботи виражається в потенціалі для подальшого розвитку і комерціалізації додатку.

Структура кваліфікаційної роботи бакалавра складається з вступу, 4 розділів, висновків, переліку посилань на 35 найменувань. Повний обсяг проекту становить 67 сторінок.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ ПРОГРАМНИХ СИСТЕМ

Перший розділ складається з огляду предметної області, що включає в себе опис та історію завдання комівояжера, а також аналізу вже існуючих додатків. Ставиться завдання дипломної роботи, формулюються вимоги до її реалізації.

1.1 Основні характеристики завдання комівояжеру

Комбінаторика – це розділ математичної науки, що вивчає комбінації та їх кількість, складені з тих чи інших умов із заданих об'єктів. Вперше питання, які вивчає комбінаторика, були підняті в працях математиків Стародавньої Греції, Індії та Китаю. Інтерес до предмета збільшився в 19-му і 20-му століттях у зв'язку з розвитком теорії графів.

У комбінаторики є багато застосувань в інших областях математики, включаючи теорію графів, програмування, криптографію і теорію ймовірності. Серед провідних математиків цій галузі можна виділити Блеза Паскаля, Якоб Бернуллі, Леонхард Ейлера і Пала Ердеша. [5]

Завдання комівояжера (Travelling salesman problem, TSP) – класична задача комбінаторики. Щоб наочніше показати її значимість, нижче наведена коротка історія вирішення поставленого завдання.

Математичні проблеми, пов'язані із завданням комівояжера, були вивчені в 1800-их роках ірландським математиком сером Вільямом Роуеном Гамільтоном і британським математиком Томасом Пенінгтон Кіркменом. У 1857 Гамільтон створив гру Ікосіан, в правилах якої сказано, що учасники повинні з'єднати 20 точок додекаедру так, щоб кожна точка використовувалася не більше одного разу, також кінцева точка шляху повинна збігатися з вихідною. Ця гра лягла в основу появи гамільтонова графа. [6]

Завдання пошуку оптимально шляху була вперше розглянута з математичної точки зору в 1930-их роках математиком і економістом Карлом менджер у Відні. У 1940-их статистики Мехаланобіс, Джессен, Гош і Маркс намагалися знайти застосування задачі комівояжера в сільськогосподарському секторі, що призвело до її популяризації – починаючи з середини 1950-х роках методи розв'язання задачі стали публікуватися в наукових журналах.

Хоча завдання комівояжера проста для розуміння, її вкрай складно вирішити [7]. У 1972 Річард М. Короп показав, що завдання Гамільтона циклу належить до класу NP-повних задач (Nondeterministic Polynomial time), які має на увазі NP-складність завдання TSP [8]. Це відкриття дало наукове пояснення очевидною обчислювальної труднощі знаходження оптимальних маршрутів. Методи рішення TSP ставали більш складними, кількість міст зростала.

Нижче представлена коротка історія етапів розв'язання задачі комівояжера.

У 1954 Данциг, Фалкерсона і Джонсон видали опис методу для вирішення TSP і практично проілюстрували його, вирішивши завдання з 49 містами, з'єднавши таким чином міста кожного штату Америки. У 1971 році Гельд і Карпо вирішили задачу пошуку оптимального шляху між 64 містами. Пізніше в 1977 році, Мартін Гротчел побудував шлях між 120 німецькими містами. У 1973 Лін і Керниган знайшли застосування завдання TSP в інженерії – вони поєднали 318 пунктів, отриманих в результаті різання лазером.

У 1987 році Голландія і Гроешел була вирішена задача з 666 містами, пізніше в цьому ж році Падберг і Рінальді знайшли оптимальний тур, що зв'язує вже 2 392 міста (рис. 1). У 1994 році кількість міст збільшилася до 7397, через 10 років кількість міст досягло до 24978. У 2006 було отримано рішення задачі комівояжера з 85900 містами. [9]

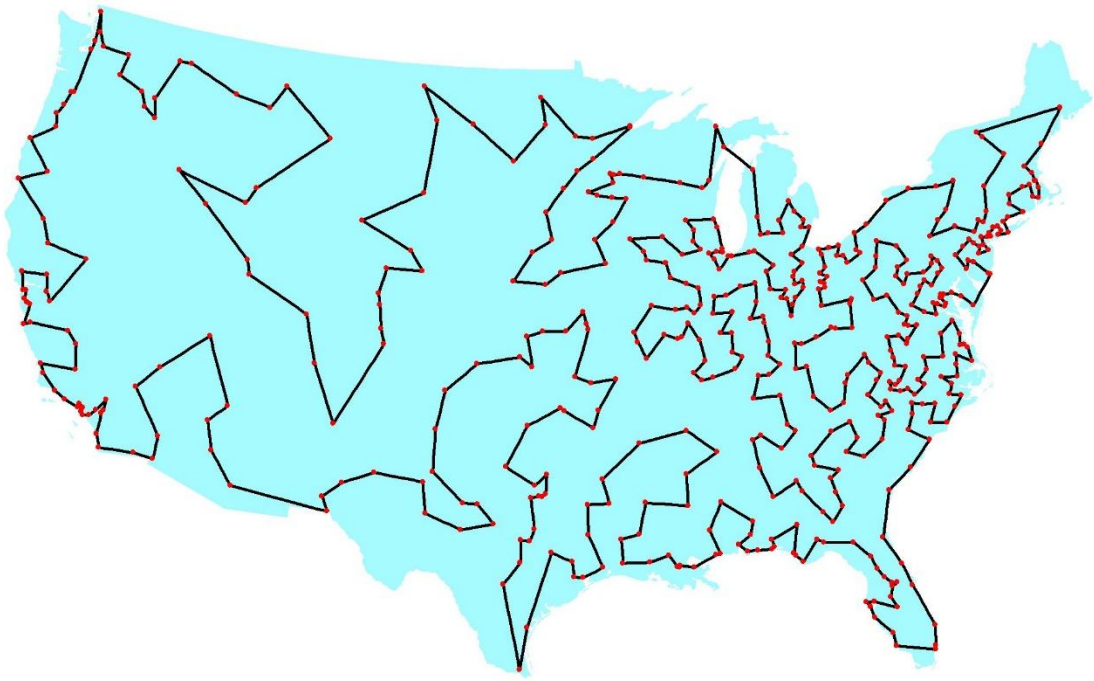

Рисунок 1 – Оптимальний тур Америкою Падберга і Рінальді (2392 міста)

1.2 Аналіз існуючих сервісів і їх недоліки

Популярні картографічні сервіси типу Яндекс.Карти і Google Maps не пропонують користувачам можливість пошуку оптимально шляху. При введенні декількох координат сервіс вибудовує маршрут в тому порядку, в якому дані були введені. Користувачі можуть вибирати засоби пересування (на машині, пішки, на наземному транспорті), але всі ці зміни впливають виключно на варіанти побудови маршруту між його фіксованими точками.

Аналіз, проведений шляхом порівнянням десятків як російськомовних, так і зарубіжних картографічних сервісів показує, що серед найпопулярніших варіантів не у всіх доступна функція побудови оптимально шляху, і далеко не завжди вона працює коректно. Нижче представлений звіт про чотири найбільш гідних уваги сервіси і їх недоліки.

Серед російськомовних сервісів в першу чергу можна виділити Meganavigator [10]. Він включає в себе конструктор карт, візуалізацію GPS

треків і побудова оптимальних маршрутів. Сервіс являє собою сайт з вбудованими Яндекс.Картами і полями для введення пунктів маршруту (рис 2).

Тестування функціоналу показало наявність істотних недоліків:

– незважаючи на те, що Meganavigator позиціонує себе як сервіс для побудови автомобільних, велосипедних і пішохідних маршрутів, на сайті відсутній вибір засобу пересування; режим, виставлений за умовчанням, відповідає пересуванню на автотранспортному засобі;
– у сервісу не передбачені підказки при введенні адреси, що ускладнює роботу з ним;
– при побудові маршруту перший пункт вибирається довільно із заданого списку адрес; немає можливості задавати адресу як перший за замовчуванням;
– при виборі пунктів на інтерактивній карті без введення адрес в текстові поля, маршрут не будувати.

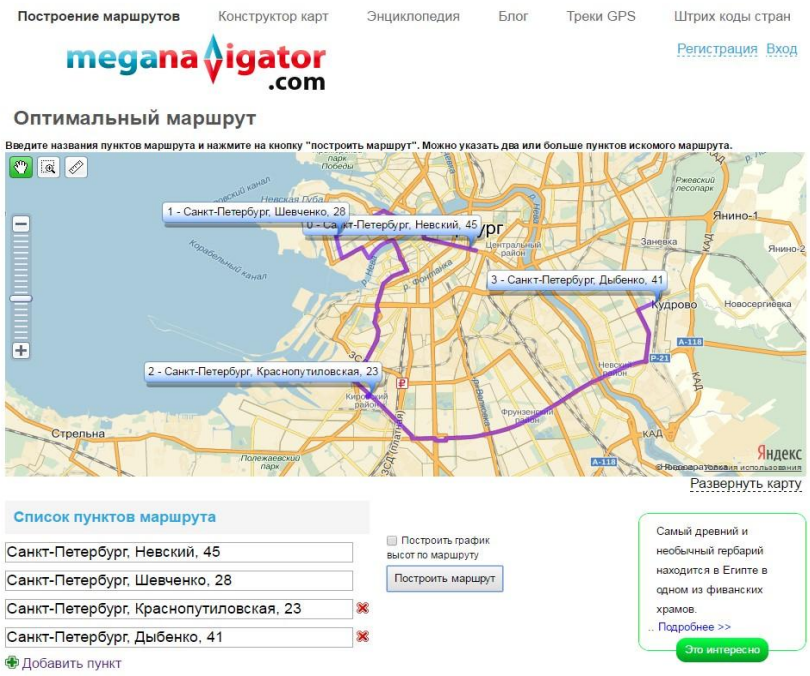

Рисунок 2 – Сервіс MegaNavigator

Наступний російськомовний сервіс Логіст [11] має меншу кількість недоліків, але всі вони істотні:

– серед засобу пересування можливий вибір між автомобілем (побудова маршруту по проїжджих дорогах) і вертольотом (прямий шлях між пунктами маршруту);
– сервіс в першу чергу розрахований для вирішення завдань логістики та побудови маршруту перевезень продукції між містами (рис. 3).

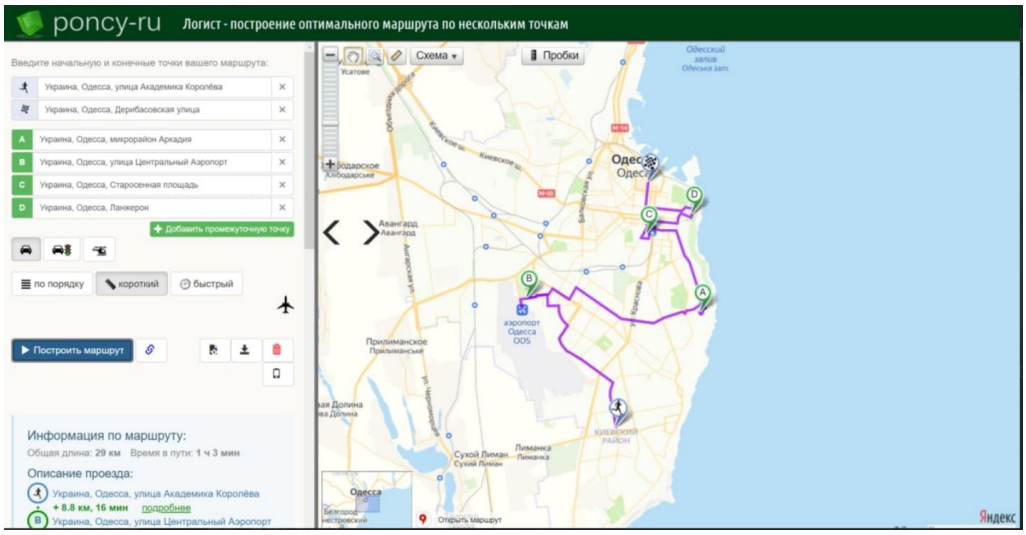

Рисунок 3 – Сервіс Логіст

Серед зарубіжних картографічних сервісів з функцією побудови оптимального маршруту найбільш популярний Speedy Route [12]. В описі продукту сказано, що сервіс розраховує найбільш ефективний за кількістю витрат палива маршрут між декількома локаціями, де вихідна і кінцева точка єдині (рис. 4). В першу чергу можна виділити такі недоліки як:

– Speedy Route розрахований виключно для автомобілістів;
– при введенні даних виникають труднощі перекладу російськомовних назв на англійську мову;
– у сервісу передбачено мінімальну кількість пунктів маршруту – 5; маршрути, що складаються з 4 пунктів побудувати неможливо;

– на сайті представлена урізана версія карти для ознайомлення, для доступу до повної версії потрібно оформити платну підписку.

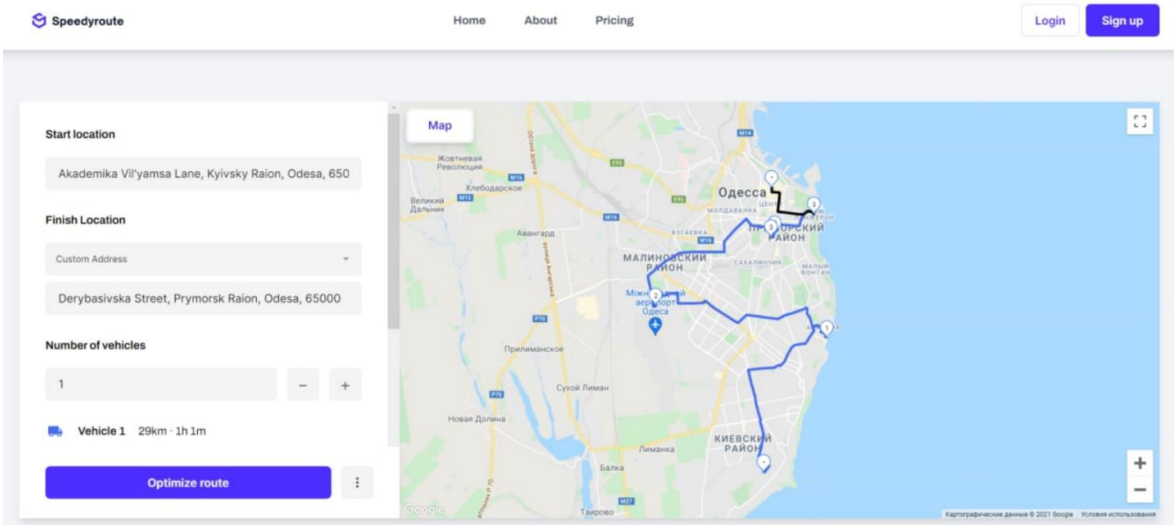

Рисунок 4 – Сервіс Speedy Route

Ще один сервіс, що дозволяє будувати оптимальні маршрути, - CargoApps від німецької компанії Impargo. Він дає можливість вносити необмежену кількість проміжних точок та оптимізувати маршрут з закріпленими початковою та фінішною або тільки початковою точками (рис. 5). Можна обирати принцип оптимізації – за довжиною шляху або вартістю проїзду (за наявності платних доріг).

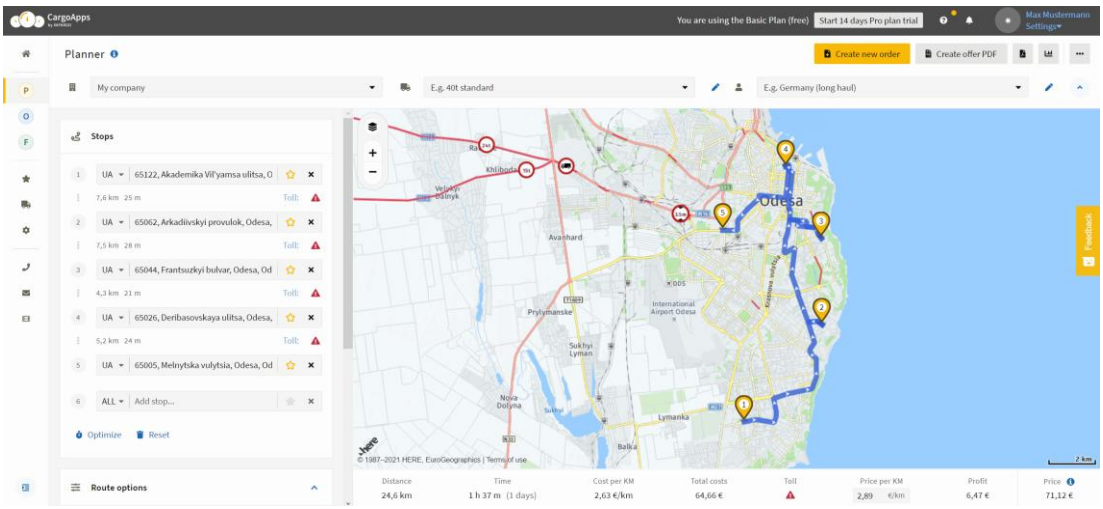

Рисунок 5 – Сервіс CargoApps

Недоліки сервісу CargoApps:

- безкоштовно користуватись можна тільки 14 днів;
- труднощі при пошуку російськомовних адрес;
- сервіс орієнтований на перевезення вантажів.

Підсумовуючи все вищезазначене можна сказати, що поки не існують сервіси, які б будували оптимальні пішохідні маршрути усередині міста і одночасно були зручними і зрозумілими для користувачів при взаємодії з ними.

1.3 Переваги та недоліки існуючих картографічних сервісів

Зазначені переваги та недоліки існуючих картографічних сервісів відображені у порівняльній табл. 1.

Таблиця 1 – Переваги та недоліки існуючих картографічних сервісів

Картографічні сервіси	Переваги	Недоліки
MegaNavigator	конструктор карт; візуалізація GPS треків	маршрут відповідає лише пересуванню на автотранспорті; не передбачені підказки при введенні адреси; при побудові маршруту немає можливості задавати першу адресу
Логіст	можна обирати принцип оптимізації – за порядком локацій, довжиною шляху або швидкістю проїзду	можливий вибір тільки між автомобілем і вертольотом; в першу чергу розрахований для побудови маршруту перевезень продукції між містами

Speedy Route	зрозумілий інтерфейс; можна розрахувати найбільш ефективний за кількістю витрат палива маршрут між декількома локаціями	сервіс виключно для автомобілістів; труднощі при пошуку російськомовних адрес; мінімальна кількість пунктів маршруту – 5; платна підписка
CargoApps	можна оптимізувати маршрут з закріпленими початковою та фінішною або тільки початковою точками; можна обирати принцип оптимізації – за довжиною шляху або вартістю проїзду	безкоштовно користуватись можна тільки 14 днів; труднощі при пошуку російськомовних адрес; сервіс орієнтований на перевезення вантажів

Таким чином, у цьому розділі був представлений огляд предметної області - представлені опис і коротка історія формування завдання пошуку оптимального маршруту, наведений аналіз недоліків існуючих додатків. За результатами огляду було поставлено завдання диплома з вимоги до її реалізації.

2 АЛГОРИТМИ МОДЕЛЮВАННЯ ОПТИМАЛЬНОГО ШЛЯХУ

У цьому розділі буде проведений аналіз існуючих алгоритмів TSP з описом їхніх достоїнств і недоліків, буде детально розібрано застосування генетичних алгоритмів до вирішення завдання пошуку оптимального маршруту і показано, чому саме генетичний алгоритм був обраний в якості інструменту розробки програми.

2.1 Аналіз алгоритмів пошуку оптимального шляху

Алгоритми для вирішення завдання комівояжера можна розділити на точні (exact algorithm) і неточні (non-exact algorithm). Точні алгоритми включають в себе перебір всіх можливих варіантів, в окремих випадках рішення можуть бути швидко знайдені, але в цілому здійснюється перебір $n!$ циклів. Другі в загальних випадках застосовуються для задач, які неможливо вирішити точно (обчислення певних інтегралів, рішення нелінійних рівнянь, витяг квадратного кореня ...), якщо існуючі точні рішення вимагають значних і невиправданих витрат часу при високій складності завдання, і як частина більш складного алгоритму, з допомогою якого завдання вирішується точно. [13]

2.1.1 Основні характеристики точного алгоритму

У свою чергу існує дві групи точних алгоритмів – одна з них використовує методи релаксації лінійного програмування TSP: алгоритм Гомори, метод внутрішньої точки, метод гілок і меж; друга, менша група, використовує методи динамічного програмування. Характерна особливість методів обох груп – гарантія знаходження оптимальних рішень при загальній трудомісткості процесу. [14]

2.1.1.1 Опис алгоритму повний перебір (Brute Force)

Один з найбільш очевидних методів вирішення задачі комівояжера – метод повного перебору або грубої сили. Його суть полягає в переборі всіх можливих варіантів шляхів, алгоритм вирішення можна записати як:

- визначити загальне число можливих гамільтонових контурів;
- визначити вагу кожного гамільтонова контуру, склавши вага всіх його ребер;
- вибрати гамільтонов контур з мінімальною вагою, який і буде оптимальним.

Метод повного перебору має низку переваг – він гарантує знаходження рішення задачі TSP, при цьому він прямолінійний і простий у виконанні. У той же час, алгоритм вважається неефективним при работе з великим об'ємом даних, так як для знаходження оптимального маршруту вимагає знайти вага $(n - 1)!$ гамільтонових контурів.

Табл. 2 демонструє кількість часу, потрібного для виконання завдання комівояжера методом повного перебору при комп'ютерної потужності, що дозволяє вважати 1 мільйон гамільтонових контурів в секунду.

Таблиця 2 – Розрахунковий час вирішення TSP методом повного перебору [15]

Розрахунковий час вирішення TSP методом повного перебору	
Кількість міст	Розрахунковий час
10	1/3 секунди
13	8 хвилин
15	1 рік
20	193 роки

2.1.1.2 Опис методу гілок і меж (Branch and Bound)

Метод гілок і меж часто використовується для знаходження оптимального рішення задач комбінаторної оптимізації. Його суть полягає в розбитті множини на підзадачі і виключення свідомо неоптимальні рішень.

Нехай граф V містить всі міста, П – безліч всіх перестановок міст, що покриває всі можливі рішення. Розглянемо перестановку $\pi \in$ П, в якій кожному місту призначається наступник – i для πi міста. Таким чином, тур можна записати як (1, π (1), π (π (1)), ..., 1). Якщо число міст в турі одно n, тоді перестановку називають циклічною. Задача про призначення ставить перед собою мету знайти циклічні перестановки, а завдання комівояжера переслідує ту ж мету, але з обмеженням, що у цих перестановок повинна бути мінімальна вартість. Метод гілок і меж в першу чергу знаходить рішення задачі про призначення, вартість якої для n міст досить велика і асимптотично дорівнює O (n1). [16]

Якщо був знайдений повний тур, то отримане значення також є вирішенням завдання комівояжера. В іншому випадку проблема поділяється на декілька підгалузей, кожна з яких виключає деякі дуги подтура, таким чином виключаючи сам подтур. Метод, за допомогою якого вираховується, яку дугу слід видалити, називають правилом розгалуження. Важливе зауваження – не повинно існувати дубльованих подзадач, їх загальна кількість має бути мінімізовано.

Будемо використовувати критерій, який гарантує незалежність подзадач – розглядається включений набір дуги і вибирається мінімальне число дуг, які не належать набору. Позначимо E як безліч виключених дуг і I як безліч включених. Розкладемо I. Виберемо t дуги подобластей x5x6 ... x7 які не належать I. Завдання розділена на t нащадків так, щоб у j :; нащадка були E <виключених дуги I <включених дуг. Запишемо у вигляді формули:

$$\left.\begin{array}{c} E_j = E \cup \{x_j\} \\ I_j - I \cup \{x_1, x_2, \ldots x_{j-1}\} \end{array}\right\} \text{k} = 1, 2, \ldots . \text{j}$$

Але x <- виключена дуга j :; подзадач і включена дуга в (j + 1) F: області. Це означає тур, отриманий рішенням (j + 1) F: завдання, може мати x <дугу, але тур, отриманий рішенням (j + 1) F :, не містить цю дугу. Це гарантує відсутність дублюються маршрутів.

Кількість можливих рішень одно (n - 1)! / 2, для n = 50 це приблизно 3 × 1062. Цей метод найбільш часто використовується при кількості вузлів від 40 до 60. [17]

Необхідність цілком вирішувати завдання лінійного програмування у всій області допустимих рішень можна вважати головним недоліком описаного вище методу. Для задач з великим об'ємом даних метод гілок і меж є невиправдано трудомістким, в той же час алгоритм є надійним методом вирішення цілочислових задач.

2.1.1.3 Опис алгоритму Гомори (The Cutting Plane)

У 1954 році була представлена робота Данцига, Фалкерсона і Джонсон, що описує новий метод розв'язання задачі комівояжера, який також може бути використаний для вирішення будь-якої проблеми

$$minimize\ c^T x\ subject\ to\ x \in S$$

де $c \neq 0$, S – кінцеве підмножина деякого RU, і таким чином це ми зможемо знайти точки S. Це ітераційний алгоритм – кожне повторення починається з лінійної програмної релаксації. Запишемо у вигляді формули:

$$minimize\ c^T x\ \ subject\ to\ Ax \leq b$$

де багатогранник P, певний як {x: Ax, містить S і обмежений. Так як P обмежений, ми можемо знайти оптимальне рішення x * як екстремальну

точку P. Якщо x * належить S, то оптимальне рішення знайдено (2.2); в іншому випадку деякий лінійне нерівність задовольняє всі точки S і порушує x *. Така нерівність називають алгоритмом Гоморі, який докладно описав його 1958 методом відтинають площин або просто відсікань. [18]

Даний метод використовується для побудови точних або наближених задач, особливо часто зустрічається в поєднанні з методом гілок і меж і тоді називається методом гілок і відсікань. Обидва методи засновані на вирішенні послідовності релаксувати подзадач лінійного програмування.

В алгоритмі Гоморі релаксовані підзадачі поступово покращують апроксимацію целочисленной завдання, зменшуючи околиця оптимального рішення. Якщо оптимальність не вдалося отримати, тоді шукається наближене рішення з похибкою.

У методу відсікань є перевага над методом гілок і меж – перші більш зручні для апаратного обчислення, так як для їх рішення не потрібен великий обсяг оперативної пам'яті для зберігання дерева рішень. [19]

2.1.1.4 Опис методу динамічного програмування (Dynamic Programming)

Розглянемо задачу з n містами і відстанями d \ <між будь-якими двома містами, шлях починається і закінчується в місті n]. TSP може бути вирішена за час O (n!) Методом повного перебору, але алгоритм динамічного програмування дозволяє скоротити час до O (n627).

Позначимо підзадачу: нехай S буде підмножиною міст, що містить 1 і як мінімум ще одне місто, j буде містом відмінним від 1, позначимо через C (S, j) найкоротший шлях, що починається в 1, що проходить всі міста S і закінчується в j.

Запишемо алгоритм у вигляді псевдо-коду.

$$for\ all\ j\ do\ C(\{1,j\},j) := d_{1j}$$
$$for\ s := 3\ to\ n\ do\ (the\ size\ of\ the\ subsets\ considered\ this\ round)$$
$$for\ all\ subsets\ S\ of\ \{1,\dots n\} of\ size\ n\ and\ containing\ 1\ do$$
$$for\ all\ j \in S, j \neq 1\ do$$
$$C(S;\ j) := min_{i\neq j, i\in S}[C(S-\{j\},i) + d_{ij})]$$
$$opt := min_{j\neq 1}[C(\{1,2,\dots n\},j) + d_{j1}]\ [20]$$

Даний метод використовується для підвищення ефективності обчислювальних повторень, зберігаючи проміжні результати і знову використовуючи їх при необхідності.

2.1.2. Основні характеристики неточних алгоритмиыв

В цілому алгоритми даної групи пропонують потенційно неоптимальні, але швидкі рішення. У свою чергу наближені алгоритми можна розділити на дві категорії: наближені (Approximation Algorithms) і евристичні (Heuristic Algorithms).

2.1.2.1 Опис алгоритму Крістофідеса (Christofides 'Algorithm)

Алогорітм Крістофідеса використовується для вирішення метричних TSP – з додатковою умовою, що для матриці відстаней виконано нерівність трикутника:

$$\forall i,j,k\ \ d_{ik} \geqslant d_{ij} + d_{jk}$$

Велика частина евристичних алгоритмів належать до 2-наближеному класу. Професор Нікос Крістофайдс в 1976 році допрацював один з існуючих алгоритмів (метод подвійного мінімального остовного дерева, $O\ (n6log6\ (n))$) так, що час вирішення завдання не перевищує оптимальний час більш ніж на 3/2. [21]

Рішення оригінального алгоритму можна записати так: знайти мінімальне дерево з безлічі всіх міст, продублювати всі ребра і побудувати

Ейлером граф, побудувати гамільтонів цикл, пройшовши кожен вузол тільки один раз і вибираючи найліпших шлях, що веде з кожного вузла.

Алгоритм Крістофідеса складається з послідовності наступних дій:

- визначити мінімальне дерево з безлічі всіх міст;
- визначити паросочетание з мінімальною вагою безлічі вершин непарного степеня і побудувати Ейлером граф;
- визначити ейлерів обхід і побудувати гамільтонів цикл, уникаючи відвідуваних вузлів. [21]

Основна відмінність – додаткове обчислення паросполучення з мінімальною вагою. Ця частина також найбільш трудомістка, тому час виконання алгоритму зростає до O (n1). Проведені тести показали, що алгоритм Крістофідеса на 10% вище нижньої межі Хелд-Карп. [22]

2.1.2.2 Опис алгоритму найближчого сусіда (NearestNeighbour)

Один з найпростіших евристичних методів рішення TSP. Головне правило алгоритму – завжди вибирати сусіднє місто (сусіда). Рішення завдання складається з наступних кроків:

- вибрати будь-яке місто;
- визначити сусіднє місто, не включений в маршрут, і перейти в нього;
- перевірити чи залишилися міста, не включені в маршрут, якщо відповідь позитивна – повторити другий крок.
- щоб завершити тур додати ребро між останнім визначеного міста і першим.

У загальному випадку трудомісткість рішення задачі дорівнює O (n6). Нижня межа вартості оптимального маршруту на 10% вище нижньої межі Хелд-Карп. [23]

2.1.2.3 Опис жадібного алгоритму (Greedy)

Щоб вирішити TSP використання жадібний алгоритм, ми досліджуємо всі ребра, що виходять з міста-вузла, і вибираємо n найкоротших дуг. Якщо

ті n найкоротших дуг формують гамильтонов цикл, тоді ми знайшли оптимальне рішення. [24]

Трудомісткість рішення задачі жадібним алгоритмом дорівнює O (n6). Нижня межа вартості оптимального маршруту вище нижньої межі Хелд-Карп на 15-20%. [22]

2.1.2.4 Опис алгоритму Керніган - Ліна (Lin-Kernighan)

Алгоритм Керніган - Ліна вважається одним із найбільш ефективних методів пошуку оптимальних або майже оптимальних рішень задачі комівояжера. Однак розробка і реалізація алгоритму лише проста, так як алгоритм складається з безлічі кроків, більшість з яких сильно впливає на роботу алгоритму. [25] Створення алгоритму Керніган було натхненне наглядом, що статичне K в оптимальний метод не дає найкраще рішення. З'явилася ідея використовувати різні стадії оптимальний методу у виконанні евристичного алгоритму. На практиці було показано, що практично неможливо заздалегідь передбачити яке K следуюет використовувати, щоб досягти кращого компромісу між трудомісткістю і якістю рішення. Лін і Керниган прибрали цей недолік, ввівши оптимальну змінну, таким чином значення K змінюється під час виконання алгоритму. [26] Трудомісткість при цьому дорівнює O (n6.6). [22]

2.1.2.5 Опис алгоритму пошуку з заборонами (TabuSearch)

Головна проблема алгоритму найближчого сусіда полягає в частому застряванні в точці локального оптимуму. Цього можна уникнути, застосувавши алгоритм пошуку із заборонами, в 1977 році запропонований Ф. Гловером. Даний метод дозволяє переходити від одного локального оптимуму до іншого в пошуку глобального оптимуму, після переходу ребро потрапляє в список заборон і повторно не використовується, крім випадків, коли воно може поліпшити побудований оптимальний шлях. На практичному рівні заборонений набір зберігається як комбінація раніше відвідуваних

кроків, який дозволяє побудувати подальший шлях щодо поточного рішення і сусідніх вузлів. [27]

Головним недоліком цього методу є його час виконання – трудомісткість алгоритму оцінюється як O (n1). [22]

2.1.2.6 Опис мурашиного алгоритму (Ant Colony Optimization)

Мурашиний алгоритм – ефективний поліноміальних алгоритм, натхненний натхненна поведінкою справжніх мурах. Вперше його принципи були описані в 1991 Марко Дориго. Мурашкам властиво співпрацювати в пошуках харчові ресурси, тому вони залишають слід хімічної речовини, феромонів, на їх шляху від гнізда до джерела їжі. [26] Цей тип невербальної комунікації називають стігмергія – стимуляція, заснована на досвіді попередніх мурах і спрямована на підвищення продуктивності. [28]

Для вирішення завдання комівояжера як правило використовують близько 20 мурах. Їх розміщують у випадкові міста і відправляють в інші міста. Їм не дозволяють двічі відвідувати один і той же місто, тільки якщо вони не завершують маршрут. Той мураха, який вибрав найкоротший тур, буде залишати слід феромонів обернено пропорційний довжині маршруту. Цей слід феромонів буде зчитуватися наступним мурахою при виборі міста, і з великою ймовірність він піде тим же шляхом, ще сильніше зміцнивши слід. Цей процес буде багаторазово повторений поки не буде знайдено маршрут, досить короткий, щоб бути оптимальним.

Серед недоліків алгоритму хочеться виділити, що перше отримане рішення може виявитися одним з найгірших в плані оптимізації, однак при повторному рішенні метод видає досить точний результат. [29]

2.1.2.7 Опис алгоритму нижня межа Хелд-Карпа (The Held-Karp Lower Bound)

Найпоширеніший спосіб виміряти ефективність евристичного алгоритму для вирішення TSP - це порівняти результати з нижньою гранню

Хелд-Карпа. Ця нижня межа є рішенням TSP, знайденим за поліноміальний час за допомогою симплекс-методу. Нижня межа Хелд-Карпа приблизно 0.8% нижче оптимальної тривалості туру. [30] У той же час вона гарантовано не перевищує оптимальний час більш ніж на 2/3. [22]

Станом на 2015 алгоритм Крістофідеса вважається самим ефективних методом для вирішення завдання комівояжера на загальних метричних просторах, хоча відомі кращі наближення для окремих випадків. Також добре в тестах себе показали алгоритм Керніган-Ліна і жадібний евристичний алгоритм. [22]

2.2 Опис генетичного алгоритму

Генетичні алгоритми належать до методів оптимізації, в основу яких лягли біологічні процеси, що протікають в природі. Чарльз Дарвін у своїй еволюційної теорії ввів визначення природного відбору, згідно з якою особи, більш присолені до умов навколишнього середовища, мають більше шансів на виживання і продовження роду, і навпаки – непристосовані осипи піддаються виборчому знищення. Основою відбору є мутації генів і їх комбінації, що формуються при розмноженні і передаються потомству. В ході природного відбору виживають екземпляри з найбільшою функцією пристосованості. Це чисельна характеристика, яка може змінюватися в залежності від умов конкретного завдання. Пристосовані особини схрещуються (кросовер) і дають потомство. Випадкові мутації також можуть впливати на розвиток популяції. На рис. 6 представлена загальна схема генетичного алгоритму: [19]

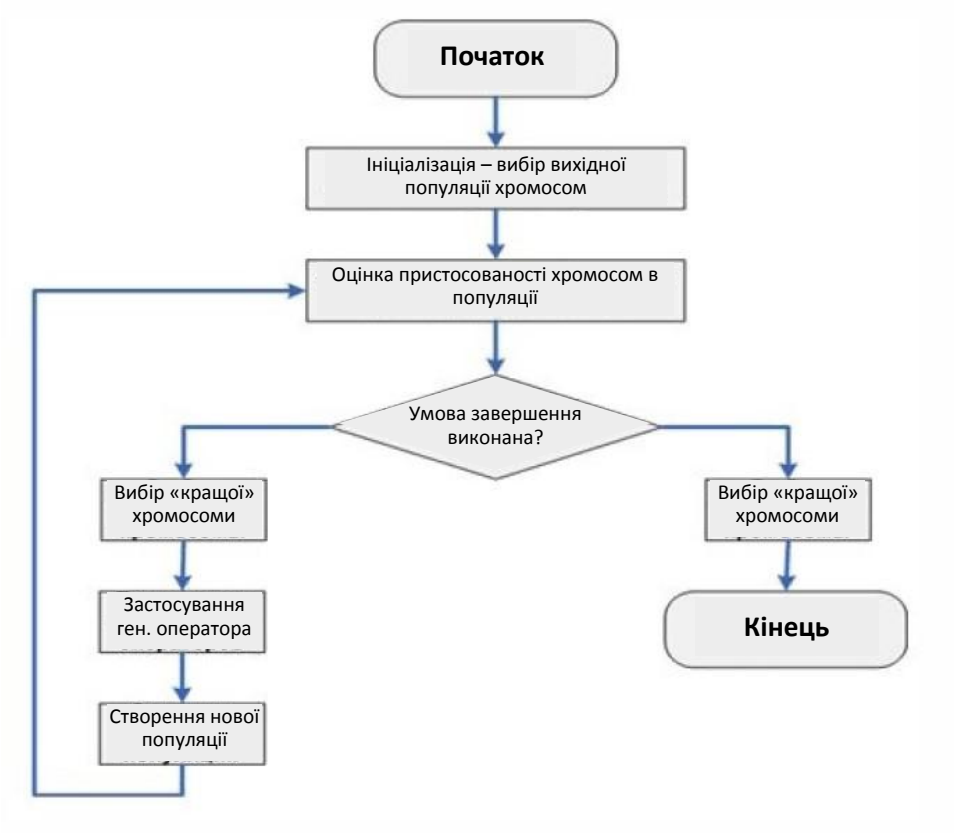

Рисунок 6 – Загальна схема генетичного алгоритму [19]

Ми можемо позначити завдання оптимізації як задачу знаходження функції f ($x5$, $x6$,..., $x7$), що носить назву функція пристосованості, яка дозволяє виділити найбільш пристосованих особин популяції для розмноження і найменш пристосованих, які викреслюються, підвищуючи тим самим пристосованість нового покоління. Потрібно, щоб на області визначення виконувалося нерівність f, область є обмеженою. Параметри функції записується як рядки, що складаються з бітів. Рядок, отримана конкатенацией рядків, називається особиною: [19]

$$1010\ 10110\ 101 \ldots 10101$$
$$|\ x1\ |\ x2\ |\ x3\ |\ \ldots\ |\ xn\ |$$

Генетичний алгоритм універсальний, так як тільки функція пристосованості та кодування рішень залежать від умов поставленого

завдання. При виконанні алгоритму враховуються наступні правила: початкова популяція вибирається випадково, кількість її особин залишається незмінним, кожна з них записується як рядок з певною довжиною кодування.

Кожен крок алгоритму можна розбити на три етапи:

- пропорційний залежно від пристосованості відбір особин поточного покоління, які мають право виробляти потомство, і формування з них проміжної популяції;
- проміжну популяцію ділять на пару і з якоюсь імовірністю схрещують, в результаті в нове покоління потрапляє сама пара або її нащадки при їх присутності. Нащадки формуються з відсічених частин батьківських рядків, розділеної певною точкою.
- відбувається мутація отриманого покоління, що не допускає передчасної збіжності. Кожен біт особини з імовірністю не більше 1% записується як протилежну початковій.

Як заздалегідь заданих критеріїв отримання оптимального рішення можуть бути певне число зміни поколінь або сходження популяції – умова виконується, якщо всі рядки є майже ідентичними і знаходяться в області екстремуму. Рішенням алгоритму буде особина з найбільшим значенням функції пристосованості. [19]

При використанні генетичного алгоритму для вирішення завдання пошуку оптимального шляху виконуються всі перераховані вище кроки, пристосованість особини фактично служить мірою довжини маршруту. Існують кілька різних реалізацій класичного генетичного алгоритму в залежності від підходу до етапів схрещування і мутації. Деякі з них дали хороші показники при тестуванні, сильно перевершивши алгоритм Керніган-Ліна. Але також, як і у алгоритму пошуку з заборонами трудомісткість процесу створює проблему при досить великій кількості вузлів. Дослідження, представлене в статті "Comparison of Algorithms for Solving Traveling Salesman Problem" [31] показує результати порівняння трьох

методів: алгоритму найближчого сусіда, генетичного і жодного алгоритмів. Як приклад можна користуватися карткою Сполучених Штатів Америки, областю 9.9 мільйонів км., представлена на рис. 7. Міста були обрані довільно.

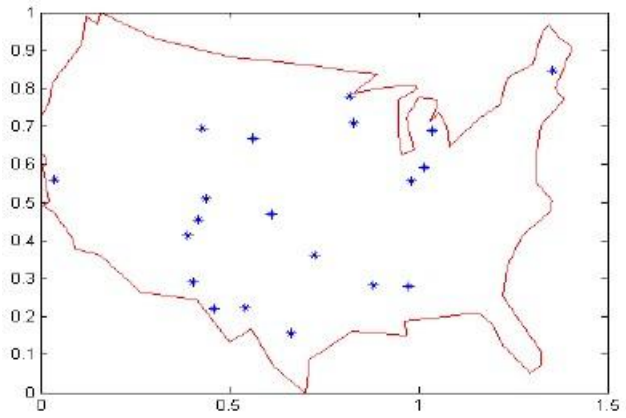

Рисунок 7 – Вихідний приклад з 20 міст

Оптимальне рішення показано на рис. 8 із загальною довжиною маршруту 4616 км. Це рішення має високу складність і складається з найбільшого числа ітерацій[31].

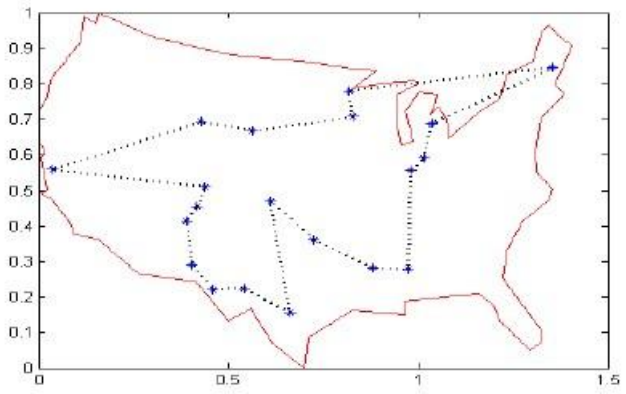

Рисунок 8 – Вихідний приклад з 20 міст [31]

Вирішуючи ту ж задачу методом найближчого сусіда, був отриманий маршрут, показаний на рис. 9. Його довжина становить 15800км.

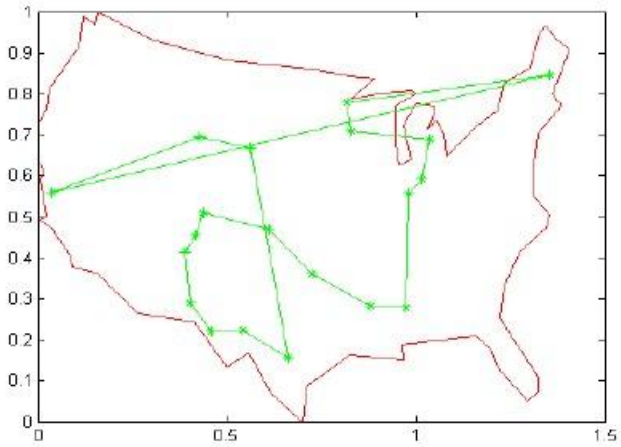

Рисунок 9 – Маршрут, отриманий з використанням алгоритму найближчого сусіда

Рішення TSP для 20 міст з використанням генетичного алгоритму показано на рис. 10. Хоча було виконано більше ітерацій, ніж в попередньому прикладі, даний алгоритм знайшов більш оптимальний маршрут рівний 11900км.

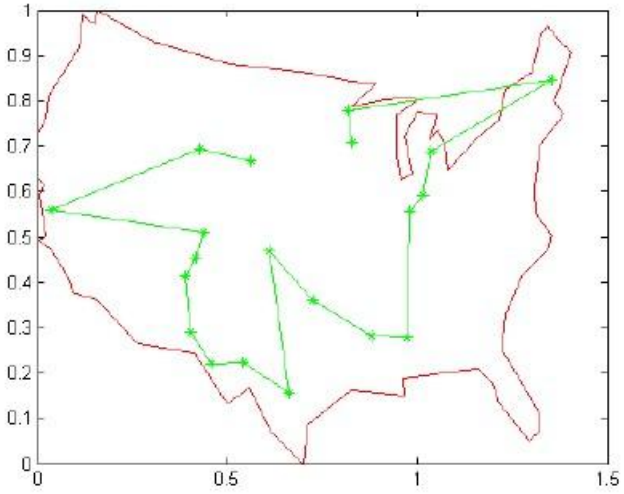

Рисунок 10 – Маршрут, знайдений генетичним алгоритмом

Той же самий приклад TSP було вирішено з використанням жодного алгоритму. Результат показаний на рисунку 2.6. і має довжину 12900км. Хоча

це рішення має незначно скорочення відстані в порівнянні з алгоритмом найближчого сусіда, у нього все ж більш висока складність і час виконання. Генетичний алгоритм показав себе як найефективніший метод вирішення TSP з невеликим обсягом даних, але в той же час як самий трудомісткий.

Подібні порівняння було наведено для задач великої розмірності. Результати дослідження, проведені на 2,2 GHz 16GB of 1600MHz DDR3L SDRAM Intel Core i7 MacBook і включають в себе порівняння таких параметрів як довжина оптимально шляху в кілометрах, витрачений час в секундах і кількість ітерацій, представлені на табл. 3.1. та 3.2 [31]

Таблиця 3 – Порівняння алгоритму найближчого сусіда, генетичного і жадібних алгоритмів при вирішенні задач з 100

Обраний алгоритм	Довжина оптимального маршруту (км)	Витрачений час (Сек)	Кількість ітерацій
Алгоритм найближчого сусіда	26664	2.5	100
Генетичний алгоритм	225479	45	10000
Жадібний алгоритм	23311	0.07	18

Таблиця 4 – Порівняння алгоритму найближчого сусіда, генетичного і жадібних алгоритмів при вирішенні задач з 1000 містами

Обраний алгоритм	Довжина оптимального маршруту (км)	Витрачений час (Сек)	Кількість ітерацій
Алгоритм найближчого сусіда	83938	95.5	1000
Генетичний алгоритм	282866	468	10000
Жадібний алгоритм	72801	127	151

Рис. 10 вище наочно показує, що при кількості міст в завданню комівояжера близькому до 100, генетичний алгоритм програє жадібному за

всіма параметрами. При аналізі TSP с 1 000 вихідними містами генетичний алгоритм в корені неефективний. [31]

Підсумовуючи сказане вище, можна сказати, що головним недоліком генетичного алгоритму є відсутність гарантії, що отримане рішення є оптимальний, як і саме його перебування за прийнятне оптимально час.

У той же час генетичні алгоритми показує високу ефективність в задачах з невеликою кількістю міст. У ситуаціях, коли в задачах великої розмірності відсутня упорядкованість вступних даних, генетичний алгоритм є єдиною альтернативою алгоритму повного перебору. Головна його перевага – його можна використовувати для вирішення складних неформалізованих проблем для яких не існує приватних методів вирішення, що дозволяє йому ефективно вирішувати нестандартні завдання. [19]

Таким чином, у розділі 2 було проведено аналіз методів вирішення TSP, подано короткий опис алгоритмів, перераховані їхні переваги і недоліки. Більш детально був описаний генетичний алгоритм, також підведені підсумки порівняльного дослідження алгоритму з рядом інших. За результатами даного порівняння було виявлено, що для задач з невеликою кількістю міст, до яких відноситься практичне завдання дипломної роботи, одним з найбільш ефективних є генетичний алгоритм, який і був обраний для подальшого роботи.

3. ОПИС КАРТОГРАФІЧНИХ СИСТЕМ

Картографічна система – це система технічних, програмних, інформаційних, лінгвістичних і організаційних засобів, призначена для створення карт і моделей в цифровий і (або) графічної формах.

У цифровій картографії карти будуються на екрані монітора комп'ютера. Для цього використовують автоматизовані картографічні системи (АКС), створені на базі спеціального класу програмного забезпечення (ПЗ). Наприклад, GeoMedia, Intergraph MGE, ESRI ArcGIS, EasyTrace, Панорама, Mapinfo і ін.

При цьому не слід плутати АКС і Географічні інформаційні системи (ГІС), так як їх завдання різні. Геоінформаційна система (географічна інформаційна система, ГІС) - система збору, зберігання, аналізу та графічної візуалізації просторових (географічних) даних і пов'язаної з ними інформації про необхідні об'єктах.

Поняття геоінформаційної системи також використовується в більш вузькому сенсі - як інструменту (програмного продукту), що дозволяє користувачам шукати, аналізувати і редагувати як цифрову карту місцевості, так і додаткову інформацію про об'єкти. Геоінформаційна система може включати до свого складу просторові бази даних (в тому числі під керуванням універсальних СУБД), редактори растрової і векторної графіки, різні засоби просторового аналізу даних

Однак на практиці один і той же набір ПЗ є інтегрованим пакетом, використовуваним для побудови і АКС, і ГІС (яскраві приклади - ArcGIS, QGIS, GeoMedia і MGE).

3.1 Система QGIS

QGIS (Quantum GIS) – вільна кроссплатформенная геоінформаційна система, що складається з настільної і серверної частини:

- QGIS Desktop - настільна ГІС для створення, редагування, візуалізації, аналізу і публікації геопросторової інформації.
- QGIS Server і QGIS Web Client - серверні додатки для публікації в мережі проектів, створених в QGIS Desktop, через сервіси, сумісні з OGC-стандартами (наприклад, WMS і WFS) [32].

QGIS працює в Windows і в більшості платформ Unix (включаючи Mac OS), підтримує безліч векторних і растрових форматів і баз даних, а також має багатий набір вбудованих інструментів (рис. 11).

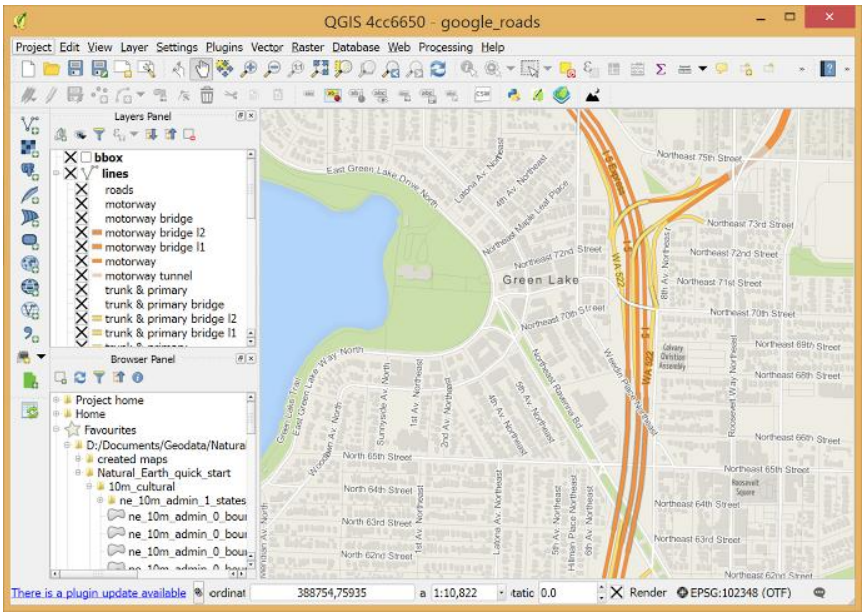

Рисунок 11 – Інтерфейс системи QGIS Desktop

Графічний інтерфейс включає в себе безліч корисних інструментів, наприклад:

- миттєве перепроектування;
- компонувальник карт;
- панель огляду;

- просторові закладки;
- визначення / вибірка об'єктів;
- редагування / перегляд / пошук атрибутів;
- підписування об'єктів;
- зміна символіки векторних і растрових шарів;
- додавання шару координатної сітки засобами розширення fTools;
- додавання до макету карти стрілки на північ, лінійки масштабу і знака авторського права;
- збереження і завантаження проектів.

У QGIS можна створювати і редагувати векторні дані, а також експортувати їх в різні формати. Щоб мати можливість редагувати і експортувати в інші формати растрові дані, необхідно спочатку імпортувати їх в GRASS. Можна аналізувати векторні просторові дані в PostgreSQL / PostGIS та інших форматах, використовуючи модуль Processing, написаний на мові програмування Python. В даний час QGIS надає можливість використовувати інструменти аналізу, вибірки, геопроцессінга, управління геометрією і базами даних. Також можна використовувати інтегровані інструменти GRASS, які включають в себе функціональність більш ніж 300 модулів GRASS.

За допомогою модуля QTiles можна генерувати тайли для роздачі карт по протоколу TMS. QGIS може використовуватися для експорту даних в map-файл і публікації його в мережі Інтернет, використовуючи встановлений веб-сервер Mapserver. QGIS може використовуватися як клієнт WMS / WFS і як сервер WMS.

3.2 Система GeoMedia

GeoMedia Professional від Hexagon Geospatial – це рішення для управління ГІС для створення карт та аналізу географічної інформації за

допомогою розумних інструментів, які збирають та редагують просторові дані (рис. 12) [33].

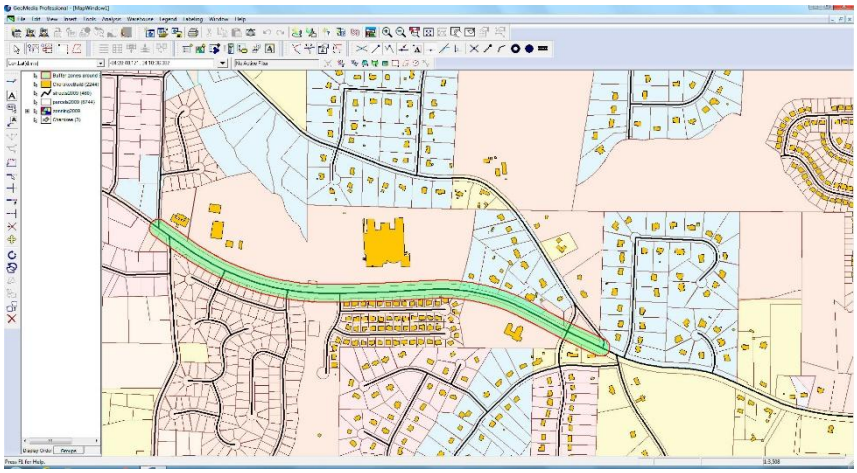

Рисунок 12 – Інтерфейс системи GeoMedia Professional

GeoMedia використовується для:

- створення географічних даних;
- управління базами геопросторових даних;
- об’єднання бізнес-даних, геоданих та географічних даних;
- створення друкованої та м’якої копій карт;
- проведення аналізу у реальному часі;
- базова платформа для декількох додатків;
- перевірки географічних даних;
- публікації геопросторової інформації;
- аналіз відображеної інформації.

Сімейство GeoMedia складається з 17 додатків, розроблених для регіонального та місцевого самоврядування, національного та федерального уряду, збройних сил та розвідки, комунальних послуг, зв'язку, фотограмметрії та транспортних ринків.

Система не покладається на власні дані, а скоріше отримує доступ і використовує джерела даних безпосередньо або дані, які відповідають

відкритим стандартам, таким як ті, що визначені Open Geospatial Consortium (OGC) та іншими. Це корпоративна система, що забезпечує організації можливість доступу, аналізу та розповсюдження інформації через організацію або через Інтернет.

3.3 Система MapInfo

MapInfo Pro - це програмний продукт для настільних ГІС, що випускається компанією Precisely (раніше Pitney Bowes Software та MapInfo Corporation) і використовується для картографування та аналізу місцезнаходження (рис. 13). MapInfo Pro дозволяє користувачам візуалізувати, аналізувати, редагувати, інтерпретувати, розуміти та виводити дані, щоб виявити взаємозв'язки, закономірності та тенденції. MapInfo Pro дозволяє користувачам досліджувати просторові дані в межах набору даних, символізувати об’єкти та створювати карти [34].

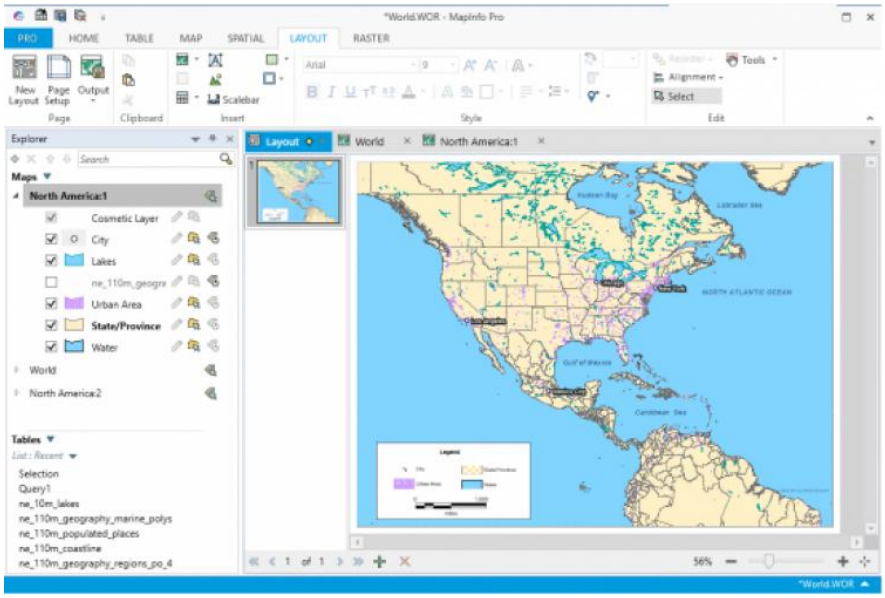

Рисунок 13 – Інтерфейс системи MapInfo Pro

MapInfo Pro підтримує всі поширені формати даних, включаючи офісні формати, такі як Microsoft Excel, Access, формати реляційних і просторових

баз даних (Oracle, Microsoft SQL Server, PostGIS, SQLite), формати графічних даних (AutoCAD DXF / DWG, SHP, DGN) і багато інших.

В роботі можна використовувати зображення практично будь-яких форматів (аерофотознімки, супутникові багатозональні знімки, скановані паперові карти і ін.). Крім того, MapInfo Pro має доступ до гібридних карт і знімкам Microsoft Bing. Інструментарій MapInfo Pro для створення і редагування графічних і табличних даних дозволяє швидко і зручно вносити зміни як на картах, так і в семантичні дані. А розвинені інструменти аналізу дозволяють робити просторові запити, оверлейні операції, будувати буферні зони, а також багато іншого.

Майстер створення карт в MapInfo Pro і зручні настройки дозволяють користувачам з будь-яким рівнем підготовки швидко створити наочні карти. В якості фону карти можна використовувати космічні знімки, а зверху накласти ваші дані у вигляді точок, ліній і полігонів. Стиль і зовнішній вигляд будь-якого набору даних легко змінити, використовуючи методи аналітичної обробки і настройки відображення. Можна використовувати статистичні та математичні функції, для відображення розрахованих значень на мапі символами або кольором. Наприклад, території продажів можуть бути розфарбовані відповідно до числа клієнтів на кожній території.

MapInfo Pro дозволяє друкувати або публікувати карти будь-якого розміру, супроводивши їх примітками, легендою і графіками. Збереження або експорт карт підтримується в будь-якому зручному для користувача форматі. А завдяки хмарним сервісам MapInfo Pro створення зон транспортної доступності і геокодування можливо без необхідності установки стороннього програмного забезпечення.

3.4 Система ArcGIS

ArcGIS – це геоінформаційна система (ГІС) для роботи з картами та географічною інформацією, що підтримується Інститутом досліджень екологічних систем (ESRI). Вона використовується для створення та використання карт, складання географічних даних, аналізу картографічної інформації, обміну та виявлення географічної інформації, використання карт та географічної інформації в ряді додатків та управління географічною інформацією в базі даних (рис. 14).

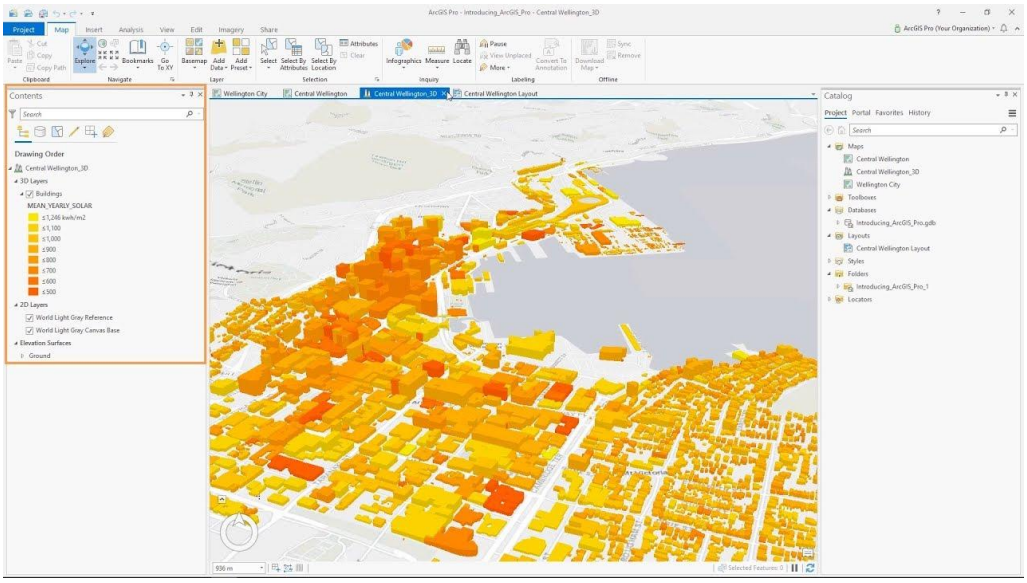

Рисунок 14 – Інтерфейс системи ArcGIS

Система забезпечує інфраструктуру для надання карт та географічної інформації, доступної для всієї організації, спільноти та всьому Інтернету.

ArcGIS складається з наступного програмного забезпечення для Windows:

– ArcReader, що дозволяє переглядати та запитувати карти, створені з іншими продуктами ArcGIS;
– ArcGIS Desktop (його часто називають "ArcMap", щоб відрізнити його від ArcGIS Pro), складається з чотирьох основних програм:

- ArcMap;
- ArcScene;
- ArcGlobe;
- ArcCatalog.

– ArcGIS Pro, новий інтегрований ГІС-додаток. ArcGIS Pro працює у 2D та 3D для картографії та візуалізації та включає штучний інтелект (ШІ).

У складі продукту ArcGIS Enterprise також є серверне програмне забезпечення ArcGIS, а також додатки ArcGIS для мобільних пристроїв, таких як телефони та планшети. Розширення можна придбати окремо для підвищення функціональності ArcGIS. Отримання сертифіката на програмне забезпечення ArcGIS також доступне для професіоналів від початківців до експертів за допомогою навчальних програм Esri.

ArcCatalog – це програма управління даними, яка використовується для перегляду наборів даних та файлів на своєму комп'ютері, базі даних чи інших джерелах. Окрім показу наявних даних, ArcCatalog також дозволяє користувачам переглядати дані на карті. ArcCatalog також надає можливість перегляду та управління метаданими для наборів просторових даних.

ArcMap – це програма, яка використовується для перегляду, редагування та запиту геопросторових даних та створення карт. Інтерфейс ArcMap складається з двох основних розділів, включаючи зміст зліва та рамки даних, що відображають карту. Елементи змісту відповідають шарам на карті.

ArcToolbox містить інструменти геообробки, перетворення даних та аналізу, а також значну частину функціональних можливостей ArcInfo. Також можна використовувати пакетну обробку за допомогою ArcToolbox для часто повторюваних завдань.

ArcScene – це програма, яка дозволяє користувачеві переглядати свої ГІС-дані у тривимірному форматі та доступна з ліцензією 3D Analyst. У

властивостях шару ArcScene є функція екструзії, яка дозволяє користувачеві перебільшувати функції, що мають три виміри.

ArcGlobe – ще одна програма ArcGIS для 3D-візуалізації, яка доступна з ліцензією 3D Analyst. ArcGlobe – програма для тривимірної візуалізації, яка дозволяє переглядати великі обсяги даних ГІС на поверхні земної кулі.

Додаток ArcGIS Pro було додано до ArcGIS Desktop у лютому 2015 року. Він мав поєднані можливості інших інтегрованих програм і був побудований як повністю 64-розрядна програма. Можливості ArcGIS Desktop [35]:

- просторовий аналіз – використання сотень інструментів для проведення просторового аналізу. Ці інструменти дозволяють вам перетворювати дані в корисну інформацію і автоматизувати велику частину ГІС-процесів, наприклад: вимірювання відстані і площі, проведення розширеного статистичного аналізу, створення складних моделей геообработки просторових даних.
- управління даними. За допомогою більш ніж 70 форматів даних можна легко інтегрувати дані для візуалізації та аналізу. Великий набір географічних і табличних інструментів, а також інструментів управління метаданими, дозволяє: отримувати географічну інформацію зі своїх даних, записувати, переглядати і управляти метаданими, визначати, експортувати і імпортувати моделі та набори даних, створювати і управляти схемами баз геоданих.
- картографування та візуалізація (Створення карт і візуалізація). Вироблення високоякісних карт за допомогою: великої бібліотеки символів і позначень, графіків, звітів і функцій анімації, готових шаблонів карт, вдосконалених інструментів для малювання, багатофункціональних картографічних інструментів, розширене редагування.

- автоматизація та редагування робочих процесів за допомогою: інструменти розширеного редагування і координаційної геометрії (COGO) для спрощення проектування і введення даних.
- можливість редагування бази геоданих декількома користувачами відразу, обмін даними між відомствами, організаціями і співробітниками на робочих місцях, доступ до даних з найбільш поширених програм CAD.
- картографічні проекції. З величезним вибором картографічних проекцій і географічних систем координат ArcGIS for Desktop дозволяє інтегрувати дані з розрізнених джерел в єдину систему: об'єднання даних з декількох джерел, конвертація просторових даних в інші системи Координат, проведення різних аналітичних операцій.
- робота з даними дистанційного зондування (ДЗЗ). ArcGIS for Desktop надає багато способів обробки растрових даних: використання растрових зображень в якості базової карти для аналізу інших верств даних, проведення різних видів аналізу зображень, поширення серед інших користувачів в організації, ефективне зберігання і обробка растрових даних
- поширення (публікація) даних. За допомогою ArcGIS for Desktop можна публікувати свої дані і карти в хмарному сервісі ArcGIS Online.
- налагодження та розробка доповнень. Зручна настройка користувальницького інтерфейсу. Додавання і видалення кнопок меню і панелей інструментів. Розробка власних доповнень до ArcGIS Engine який доступний через Esri Developer Network (EDN).

4. РЕАЛІЗАЦІЯ СИСТЕМИ ЗНАХОДЖЕННЯ ОПТИМАЛЬНОГО ШЛЯХУ

4.1 Вимоги до системи, що розробляється

Аналіз вже існуючих застосувань, представлений в розділі 1, допомагає сформулювати вимоги, які можна пред'явити до системи, що розробляється:

- функціонал програми полягає в побудові оптимального маршруту по заданих даними і виведення його на інтерактивній карті;
- сервіс повинен відповідати законам UX (User Experience) і бути зручним для використання;
- одне з ключових параметрів сервісу це його доступність – найбільш вдалим рішенням буде реалізація програми як веб-сайту;
- у зв'язку зі стрімкістю розвитку науки і постійним перебуванням нових методів і оптимізації старих, сервіс повинен бути спроектований таким чином, щоб обсяг робіт зі зміни оптимізаційного алгоритму був мінімальний.

4.2 Реалізація систем пошуку за допомогою алгоритму Кристофидеса

Ейлеровим шляхом в графі називається довільний шлях, що проходить через кожне ребро графа в точності один раз.

Замкнуте Ейлером шлях називається ейлеровим обходом або ейлеровим циклом.

Ейлеров граф - граф, в якому існує Ейлером обхід. Наведемо критерій ейлерову графа.

Завдання комівояжера називається метричної, якщо для матриці відстаней виконано нерівність трикутника: $\forall i, j, k \; dik \leq dij + djk$

```
def EulerCircuit(G):
    if not EulerCircuitExists(G):
        return None

    # Первая попавшаяся – в ЭЦ!
    EP = [ G.nodes()[0] ]
    while G.number_of_edges() > 0:
        for i, v in enumerate(EP):
            # куда бы добавить цикл?
            if G.degree(EP[i]) > 0:
                break
        while G.degree(v) > 0:
            #пока не в «тупике»
            w = G.neighbors(v)[0]
            G.remove_edge(v, w)
            i += 1
            EP.insert(i, w)
            v = w  # и повторяем все с w
    return EP
```

```
[0] + 0 : < 1 2 3 4 0 > -> [0, 1, 2, 3, 4, 0]
[0, 1, 2, 3, 4, 0] + 1 : < 4 1 > -> [0, 1, 4, 1, 2, 3, 4, 0]
[0, 1, 4, 1, 2, 3, 4, 0]
```

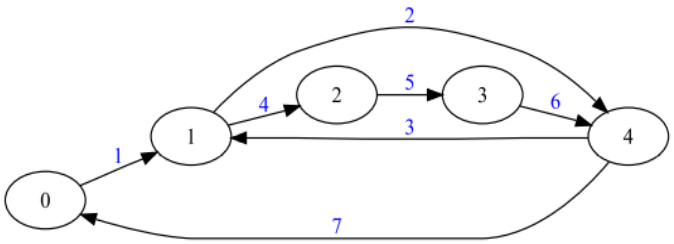

Рисунок 15 Алгоритм находження Ейлерового циклу

Зауважимо, що тут розглядається повний граф.

Необхідно відзначити, що метрична задача комівояжера N P-повна. Це легко довести, заме тив, що якщо ваги ребер повного графа приймають тільки два значення 1 і 2, то задача є метри ческий. У свою чергу, побудова мінімального обходу тут еквівалентно відповіді на питання: суще ствует чи в графі з вершинами вихідного графа і ребрами, що мають вагу 1, гамільтонів цикл?

Для цієї варіації завдання можна запропонувати наступний наближений алгоритм з Мультипла катівнях точністю 2

У розглянутому в алгоритмі прикладі Ейлером цикл складає наступний маршрут:

$$v0 \to v1 \to v5 \to v4 \to v5 \to v9 \to v8 \to v9 \to v13 \to$$
$$\to v12 \to v13 \to v9 \to v10 \to v14 \to v15 \to v14 \to v10 \to$$
$$\to v11 \to v10 \to v6 \to v7 \to v6 \to v2 \to v3 \to$$
$$\to v2 \to v6 \to v10 \to v9 \to v5 \to v1 \to v0$$

Відповідний йому вкладений тур (гамільтонів цикл) такий:

[v0, v1, v5, v4, v9, v8, v13, v12, v10, v14, v15, v11, v6, v7, v2, v3, v0].

4.3 Реалізація системи пошуку оптимальногго шляху в ArcGIS

Виконуємо аналіз завдання знаходження транспортного маршруту з урахуванням парних замовлень. Мета розробки – знайти найбільш оптимальні маршрути для автопарку, щоб перевозити людей, у яких немає іншого доступу до транспорту, з дому до лікарні для проходження лікування. Вправа буде виконано шляхом вирішення задачі знаходження транспортного маршруту (VRP) з урахуванням парних замовлень, яка пов'язує два послідовних замовлення (зупинки) так, що транспорт буде підбирати людей і доставляти їх в потрібну лікарню. Також будуть дотримані додаткові вимоги за допомогою інших характеристик шару аналізу завдання VRP. Наприклад, буде введено максимальне час у дорозі для парних замовлень так, щоб люди не проводили занадто багато часу в дорозі. Будуть використані тимчасові обмеження, так як люди не повинні спізнюватися в лікарню. Деяким людям потрібні крісла-коляски, тому для них потрібно буде відправити транспорт зі спеціальними підйомниками для крісел-колясок. Після визначення маршрутів будуть створені покрокові вказівки результуючих маршрутів, які можна відправити електронним способом або роздрукувати і передати водіям.

Для підготовки відображення запускаємо ArcMap. Відкриваємо діалогове вікно Відкрити документ ArcMap (Open ArcMap Document). Переходимо до папки C: \ ArcGIS \ ArcTutor \ ArcGIS Network Analyst \ Tutorial. Активуємо додатковий модуль ArcGIS Network Analyst. Для цього:

1. Обираємо Налаштування (Customize) > Додаткові модулі (Extensions). Відкриється діалогове вікно Додаткові модулі (Extensions).
2. Відзначаємо ArcGIS Network Analyst.
3. Обираємо на кнопку Закрити (Close).

Обираємо Налаштування > Панелі інструментів> Network Analyst. Панель інструментів Network Analyst буде додана в ArcMap (рис. 16).

Рисунок 16 – Панель інструментів Network Analyst

На панелі інструментів Network Analyst обираємо на кнопці Вікно Network Analyst. Відкриється вікно Network Analyst (рис. 17).

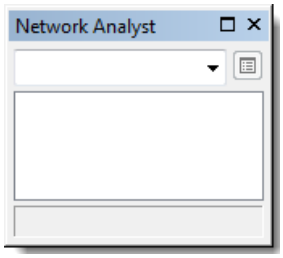

Рисунок 17 – Вікно Network Analyst

Створюємо шар аналізу для вибору маршруту транспорту. Обираємо на пункті Network Analyst на панелі інструментів Network Analyst і на Нова задача вибору маршруту транспорту (рис. 18).

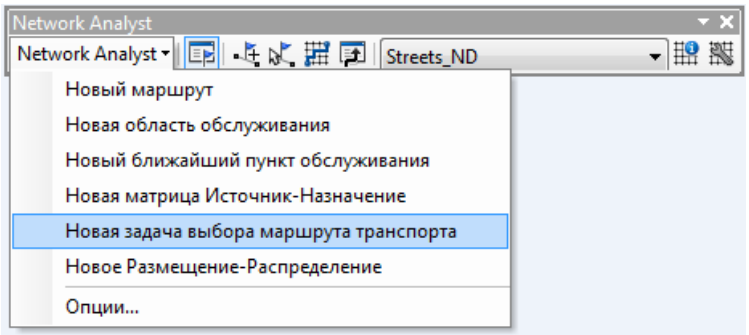

Рисунок 18 – Нова задача вибору маршруту транспорту

Шар аналізу завдання вибору маршруту транспорту доданий у вікно Network Analyst. Класи мережевого аналізу: Виклики (Orders), Гаражі (Depots), Маршрути (Routes), Повернення в гараж (Depot Visits), Межі

(Breaks), Зони маршрутів (Route Zones), Вихідні точки маршрутів (Route Seed Points), Оновлення маршрутів (Route Renewals), Спеціальні вимоги (Specialties), Пари замовлень (Order Pairs), Точкові бар'єри (Point Barriers), Лінійні бар'єри (Line Barriers) і Полігональні бар'єри (Polygon Barriers) – порожні (рис. 19).

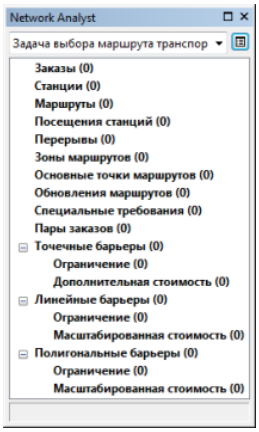

Рисунок 19 – Порожні класи мережевого аналізу

Також у вікно Таблиця змісту (Table Of Contents) доданий новий шар аналізу (рис.20).

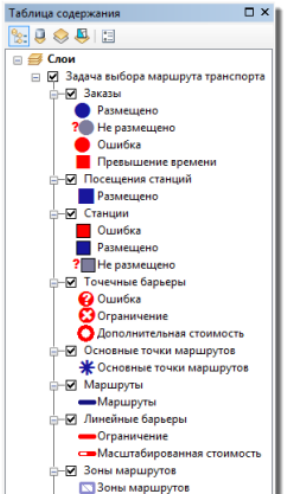

Рисунок 20 – Таблиця змісту

Додаємо спеціальні умови. Припустимо, що у логістичної компанії є три машини. Одна з машин, що працюють в центрі, обладнана для

навантаження крісел-колясок. Додаємо Інвалідне крісло в якості спеціального вимоги так, щоб замовлення, які мають потребу в цьому вимозі, були призначені маршруту з ліфтом для інвалідних крісел.

У вікні Network Analyst обираємо правою кнопкою миші на кнопці Спеціальні вимоги (0) і вибираємо команду Додати елемент (рис. 21).

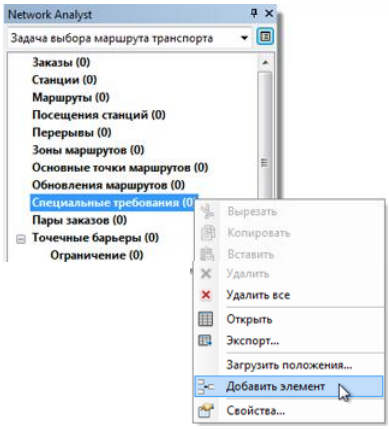

Рисунок 21 – Додати елемент

Нова спеціальна умова Елемент 1 з'явиться в розділі Спеціальні вимоги у вікні Network Analyst. Відкриється вікно Властивості для нової спеціальної умови.

У вікні Властивості вводимо Крісло-коляска в рядку Ім'я (рис. 22).

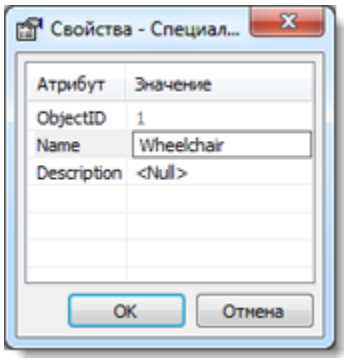

Рисунок 22 – Властивості спеціальної умови

Таблиця містить відомості про кожного пацієнта, включаючи імена і домашні адреси, назва і адреса лікарні, в яку вони повинні бути доставлені,

час, коли пацієнта потрібно підібрати тощо. У табл. 5 наведено опис полів електронної таблиці.

Таблиця 5 – Опис полів електронної таблиці

Атрибут	Опис
OrderName1	Ім'я пацієнта, якого потрібно перевезти
PatientAddress	Адреса, за якою потрібно забрати пацієнта
OrderName2	Унікальне ім'я адреси призначення
HospitalAddress	Адреса, за якою потрібно доставити пацієнта
PickFrom	Найперший час, коли пацієнта можна забрати за адресою PatientAddress
PickTo	Найпізніший час, коли пацієнта можна забрати за адресою PatientAddress
TotalPassengers	Загальна кількість пасажирів, яких потрібно забрати
MaxTransitTime	Максимальний час, який пацієнт може провести в машині
SpecialtyNames	Вказує особливі потреби пасажира, наприклад, крісло-коляску

В даному випадку пасажири і лікарні пов'язані, оскільки кожен пацієнт повинен потрапити в певну лікарню. Цю ситуацію можна моделювати за допомогою парних замовлень, завантаживши обох пацієнтів і адреси лікарень в класи аналізу мережі Замовлення і зв'язавши їх з новими об'єктами парних замовлень.

Виконаємо геокодування за адресами пацієнтів і лікарень і завантажимо результуючі розташування як замовлення. У вікні Каталогу в директорії Home знаходимо файл OrderPairs.xls і двічі обираємо. Файл OrderPairs.xls розгорнеться і ми побачимо таблицю Patients $. Обираємо правою кнопкою таблицю Patients $ і вибераємо Геокодувати адреси. З'явиться діалогове вікно Вибрати локатор адрес. Обираємо підготовлений

локатор адрес KyivLocator. Переходимо ОК. З'явиться діалогове вікно Геокодування адрес.

Обираємо стрілку спадного меню поруч із кнопкою огляд біля спадаючого списку таблиць адрес. Відкриється діалогове вікно Виберіть таблицю, яка містить адреси. Двічі обираємо на файл OrderPairs.xls. Двічі обираємо на елемент Patients $. Електронна таблиця Patients буде додана в список, що розкривається.

У списку Street or Intersection обираємо PatientAddress. Обираємо оглядову кнопку поруч з вікном Вихідний шейп-файл або клас об'єктів. Відкриється діалогове вікно Збереження даних. У списку Шукати в вибираємо Home - Tutorial.

У списку Зберегти як тип вибираємо Класи просторових об'єктів персональної і файлової баз геоданих. Список файлів і робочих областей оновиться. Двічі обираємо на файл Kyiv.gdb. Видаляємо ім'я в текстовому полі Ім'я і вводимо Patients. У базі геоданих з'явиться клас об'єктів з ім'ям Patients. Переходимо Зберегти (Save).

Текстове поле Вихідний шейп-файл або клас об'єктів в діалоговому вікні Геокодування адрес оновиться з урахуванням нового вихідного шляху (рис. 23).

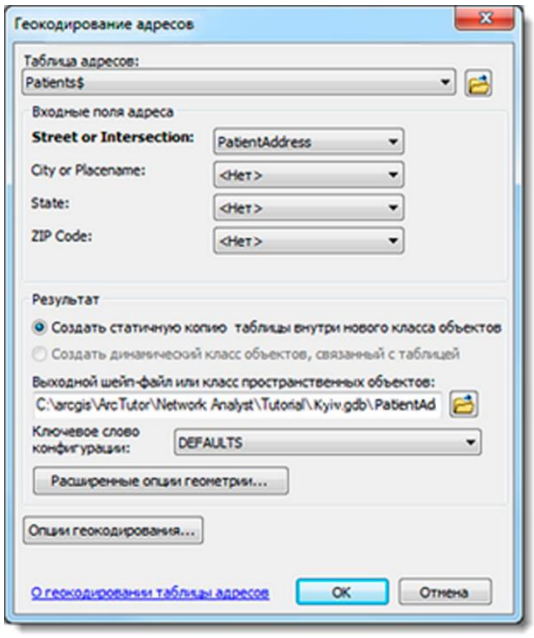

Рисунок 23 – Вікно Геокодування адрес (пацієнти)

Переходимо ОК. Відкриється діалогове вікно Геокодування адрес і покаже, що всі адреси були знайдені. Обираємо Закрити (Close). Геокодовані адреси будуть додані в документ карти в якості шару просторових об'єктів Результат геокодування: Patients.

Повторюємо ці кроки, щоб завантажити лікарні призначення пацієнтів, але вносимо наступні зміни (рис. 24):

1. Задаємо властивості Street or Intersection значення HospitalAddress.
2. Вводимо DestinationHospitals в текстовому полі Ім'я.

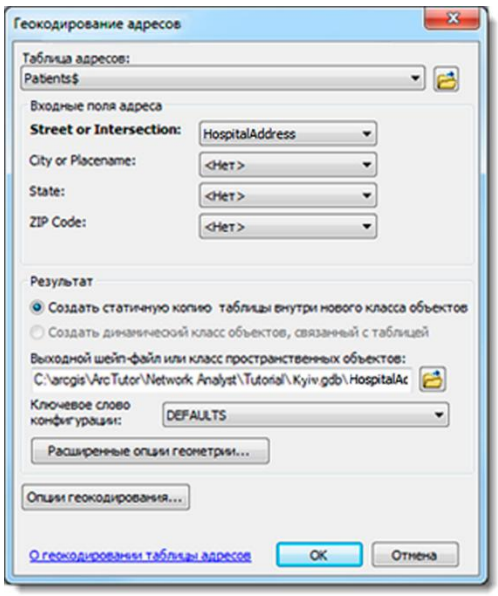

Рисунок 24 – Вікно Геокодування адрес (лікарні)

У таблиці змісту повинно бути два векторних шару Результати геокодування: DestinationHospitals і Результати геокодування: Patients.

У вікні Таблиця змісту вимикаємо два шари результатів геокодування, щоб вони стали невидимими на мапі.

У вікні Network Analyst обираємо правою кнопкою миши на кнопці Замовлення (0) і вибераємо команду Завантажити положення. Відкриється

діалогове вікно Завантажити положення. Вибираємо Результати геокодування: Patients в випадаючому списку Завантажити з.... Переглянути властивості аналізу положень діалогового вікна Завантажити положення можна, вказав, які атрибути шару Результати геокодування: Patients міститимуть значення, на які буде посилатися Network Analyst для вирішення завдання з знаходження транспортного маршруту.

Налаштовуємо властивості, перераховані в розділі Властивості аналізу положень так, щоб вони відповідали значенням полів шару Результати геокодування: Patients (рис. 25). Зіставляємо властивість Name полю OrderName1.

Порівнюємо властивість Description полю PatientAddress, TimeWindowStart1 полю PickFrom, TimeWindowEnd1 полю PickTo, PickupQuantities полю TotalPassengers. Переконуємося, що властивість SpecialtyNames автоматично відповідає полю SpecialtyNames.

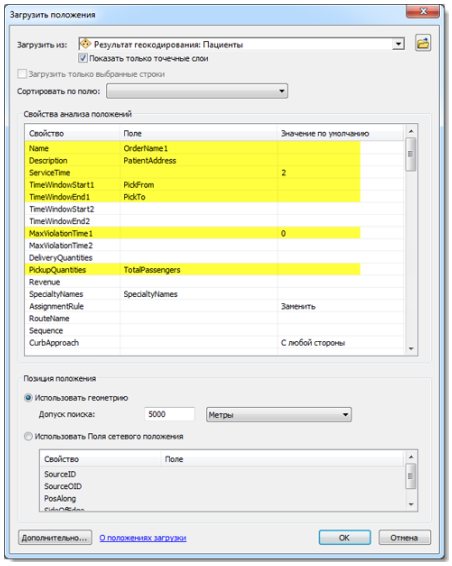

Рисунок 25 – Вікно Завантажити положення (пацієнти)

Вводимо значення 2 в стовпці Значення за замовчуванням для властивості ServiceTime. Всі завантажені адреси пацієнтів мають значення 2 для властивості ServiceTime, яке означає час (у хвилинах) посадки в машину для кожного пасажира.

Вводимо значення 0 в стовпці Значення за замовчуванням для властивості MaxViolationTime1. Встановивши для всіх властивостей MaxViolationTime1 значення 0, інструмент вирішення завдання VRP буде шукати тільки маршрути для замовлень в зазначений час.

Переходимо ОК. Завантажено 15 замовлень. Замовлення видно на карті і в вікні Network Analyst.

Завантажуємо лікарні призначення в якості замовлень.

У вікні Network Analyst обираємо правою кнопкою миші на кнопці Замовлення (15) і вибираємо команду Завантажити положення. Відкриється діалогове вікно Завантажити положення.

Вибираємо Результати геокодування: DestinationHospitals в списку Завантажити з.

Налаштовуємо властивості, перераховані в розділі Властивості аналізу положень так, щоб вони відповідали значенням полів шару Результати геокодування: DestinationHospitals (рис. 26).

Зіставляємо властивість Name полю OrderName2. Звертаємо увагу, що значення для атрибута Name має бути унікальним в класі аналізу мережі Замовлення. В даному випадку є кілька пацієнтів, яким потрібно відвідати одну і ту ж лікарню. Якби адреси лікарень використовувалися для виведення значення атрибута Name для Замовлень, інструмент вирішувач VRP повернув би повідомлення про помилку через повторюваних значень імен. Порівнюємо властивість Description полю Hospital Address.

Порівнюємо властивість DeliveryQuantities полю TotalPassengers. Переконуємося, що властивість SpecialtyNames автоматично відповідає полю SpecialtyNames.

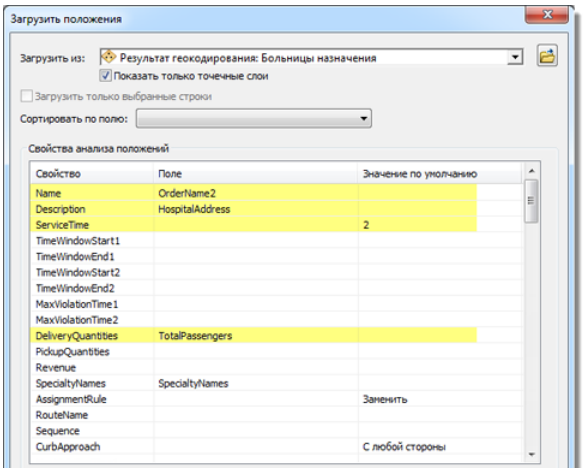

Рисунок 26 – Вікно Завантажити положення (лікарні)

Вводимо значення 2 в стовпці Значення за замовчуванням для властивості ServiceTime. Переходимо ОК. Замовлення перераховані у вікні Network Analyst в розділі аналізу мережі Замовлення і відображені в якості замовлень на мапі шару завдання по знаходженню транспортного маршруту.

Пасажирів потрібно доставляти в певну лікарню. Додаючи об'єкти в клас аналізу мережі Пари замовлень, можна вказати, в яку лікарню потрібно відвезти пацієнтів і максимальний час, який пацієнти можуть провести в машині протягом поїздки в одну сторону.

У вікні Network Analyst обираємо правою кнопкою миші на кнопці Пари замовлень (0) і вибираємо команду Завантажити положення. Відкриється діалогове вікно Завантажити положення.

Обираємо кнопку Огляд поруч зі стрілкою спадаючого списку Завантажити з. Двічі обираємо на файл OrderPairs.xls. Двічі обираємо на елемент Patients$. Електронна таблиця Patients додається в список Завантажити з в діалоговому вікні Завантажити положення. Налаштовуємо властивості, перераховані в розділі Властивості аналізу розташувань так, щоб вони відповідали значенням з таблиці Patients $.

1. Зіставляємо властивість FirstOderName полю OrderName1.
2. Порівнюємо властивість SecondOrderName полю OrderName2.

3. Переконаємося, що властивість MaxTransitTime автоматично відповідає полю MaxTransitTime.

   Переходимо на кнопку ОК. Пари замовлень перераховані у вікні Network Analyst в розділі Пари замовлень класу мережевого аналізу (рис. 27).

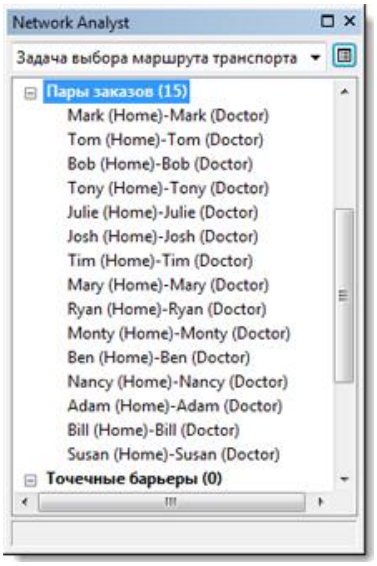

Рисунок 27 – Пари замовлень класу мережевого аналізу

Логістична компанія використовує машини з трьох розташувань, зазначених на шарі просторових об'єктів CentralDepots в ArcMap. Ці точкові характеристики потрібно додати в клас аналізу мережі Станції.

У вікні Network Analyst обираємо правою кнопкою миші кнопку Станції (0) і вибираємо команду Завантажити положення. Відкриється діалогове вікно Завантажити положення (рис. 28).

Вибираємо CentralDepots в списку Завантажити з. У розділі Властивості аналізу положень переконаємося, що властивість Name автоматично відповідає полю Name.

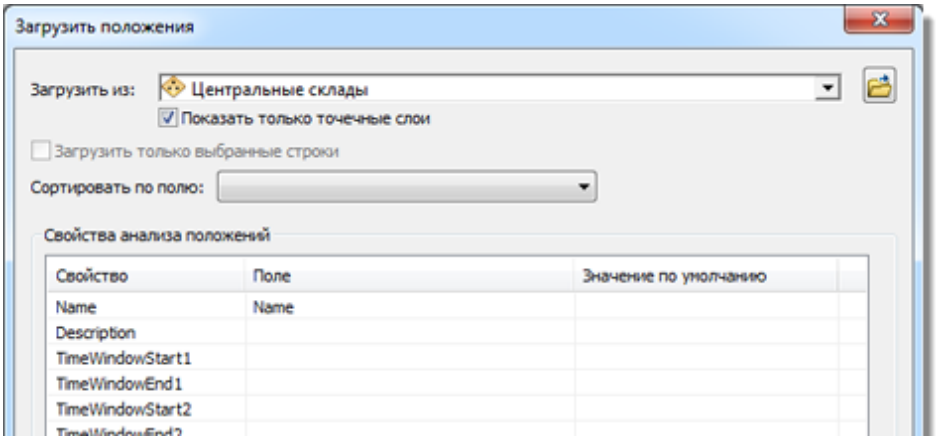

Рисунок 28 – Діалогове вікно Завантажити положення

Переходимо ОК. З станції перераховані у вікні Network Analyst на вкладці класу аналізу мережі Станції і відображені в якості замовлень на мапі шару завдання по знаходженню транспортного маршруту.

У логістичної компанії три машини, кожна з яких може перевозити не більше шести пасажирів. Машини їдуть з автобази і повертаються на неї після виконання маршруту. Одна з машин, що працюють в центрі, обладнана для навантаження крісел-колясок.

Потрібно додати три маршрути (по одному для кожної машини) і вказати, що машина, яка їздить по місту, була обладнана для навантаження крісел-колясок.

У вікні Network Analyst обираємо правою кнопкою миші на кнопку Маршрути (0) і вибираємо команду Додати елемент (рис. 29).

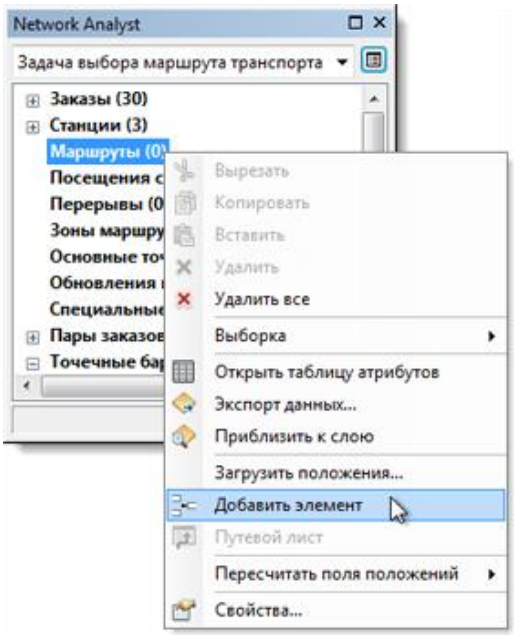

Рисунок 29 – Додати елемент

Новий маршрут Елемент1 додається на вкладці класу аналізу мережі Маршрути, після чого відкриється вікно Властивості (рис. 30). У вікні Властивості задаємо атрибути для маршрута, як показано в таблиці 6, не змінюючи значення атрибутів, задані за замовчуванням. У стовпці опису наводяться пояснення кожного значення.

Таблиця 6 – Атрибути маршрута

Атрибут	Значение	Описание
Ім’я канала	Центр міста	Назва маршрута.
StartDepotName	Автобаза у центрі міста	Машина виїжджає з автобази в центрі міста.
EndDepotName	Автобаза у центрі міста	Машина повертається на автобазу в центрі міста після виконання маршрута.
Ємність	6	Машина може вмістити не більше шести пасажирів одночасно.
SpecialtyNames	Крісло-коляска	Машина обладнана для завантаження крісел-колясок.

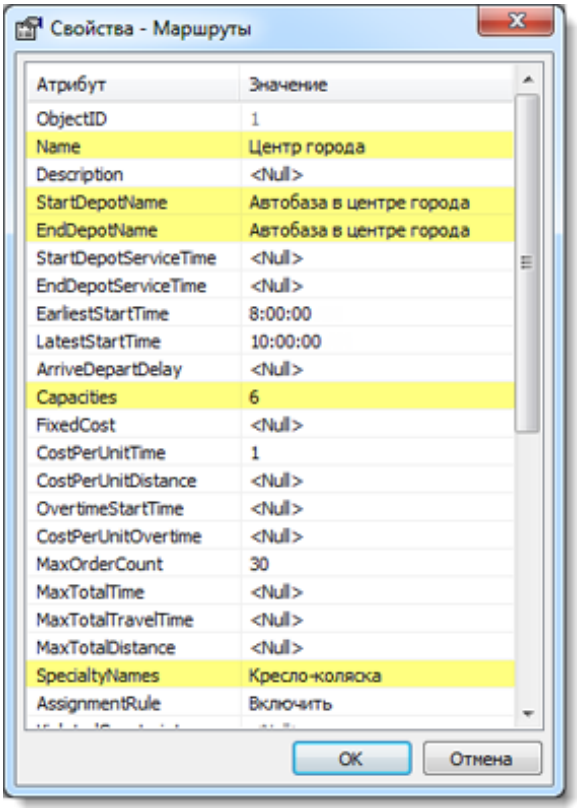

Рисунок 30 – Вікно Властивості маршрута

Переходимо ОК. Новий маршрут Центр міста вказаний тепер у вікні Network Analyst. Додаємо два інших маршрута для машин, що працюють в Шевченківському і Голосіївському районі. Використовуємо наступні значення для цих нових маршрутів у табл. 7.

Таблиця 7 – Атрибути маршрутів 2 і 3

Атрибут	Значение
Имя канала	Шевченківський район
StartDepotName	Автобаза Шевченківського района
EndDepotName	Автобаза Шевченківського района
Ємність	6
Имя канала	Голосіївський район
StartDepotName	Автобаза Голосіївського района
EndDepotName	Автобаза Голосіївського района
Ємність	6

Ці машини обладнані для навантаження крісел-колясок. У вікні Network Analyst показані три об'єкти маршрутів, які перераховані в класі аналізу мережі Маршрути (рис. 31).

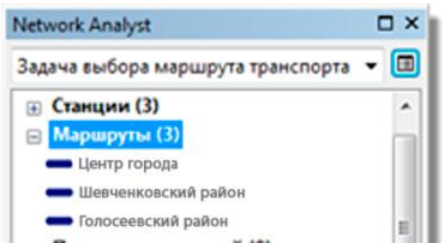

Рисунок 31 – Маршрути у вікні Network Analyst

Компанія використовує три машини, які мають ліцензії на обслуговування замовлень тільки в певній зоні. Потрібно додати зони маршрутів і зв'язати з з машинами або маршрутами.

У вікні Network Analyst вибираємо елемент Зони маршрутів (0). Вибираємо інструмент Створення мережевого положення (Create Network Location) на панелі інструментів Network Analyst. На карті малюємо багатокутник, який в загальних рисах охоплює центр міста (рис. 32).

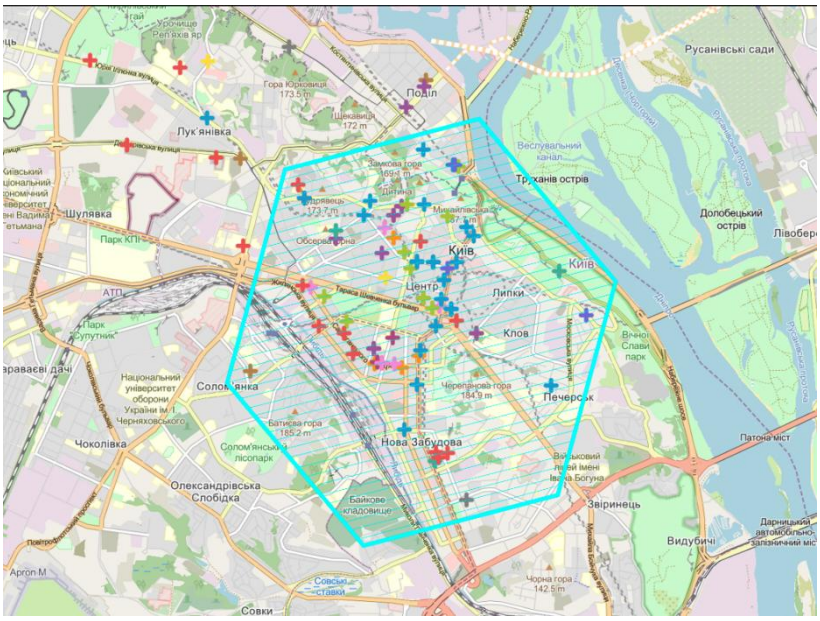

Рисунок 32 – Багатокутник Створення мережевого положення

Нова зона маршруту Графічний покажчик 1 буде додана в клас зон маршруту у вікні Network Analyst. У вікні Network Analyst вибираємо об'єкт нової зони маршруту Графічний покажчик 1. Відкриється вікно Властивості для зони маршруту (рис. 33). Задаємо властивості зони маршруту, як показано в таблиці 8.

Таблиця 8 – Властивості зон маршрута

Атрибут	Значение	Описание
RouteName	Центр міста	Назва маршрута, з яким пов’язана ця зона маршрута.
IsHardZone	Так	Машина не може обслуговувати замовлення, які не входять до зони маршруту. Якщо ця властивість має значення True, то машина буде отримувати тільки ті замовлення, які відносяться до зони маршруту

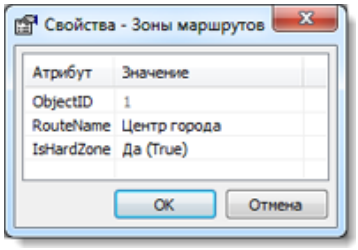

Рисунок 33 – Вікно властивостей зон маршрутів

Додаємо дві інші зони маршруту: одну для Шевченківського району та другу для Голосіївського району. На карті і в вікні Network Analyst має бути три об'єкти зони маршруту (рис. 34).

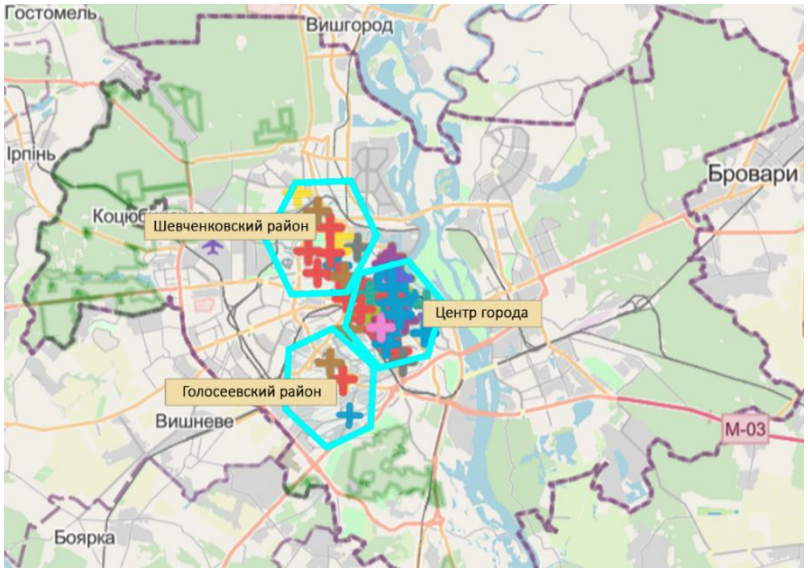

Рисунок 34 – Об’єкти зони маршрута на карті

Далі потрібно задати властивості для аналізу завдання по знаходженню транспортного маршруту. Переходимо кнопку Властивості шару аналізу (Analysis Layer Properties) у вікні Network Analyst (рис. 35).

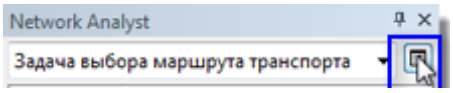

Рисунок 35 – Властивості шару аналізу

Відкриється діалогове вікно Властивості шару (Layer Properties). Переходимо на закладку Налаштування аналізу (Analysis Settings). У списку Атрибут часу вибираємо значення TravelTime (Хвилини). Інструмент вирішення задачі VRP використовує цей атрибут для розрахунку витрат між замовленнями і автобазами на основі часу. У списку Атрибут відстані нічого не повинно бути вибрано. Оскільки параметри витрат на основі відстані, такі як CostPerUnitDistance або MaxTotalDistance, не використовуються, атрибут відстані не потрібно.

Задаємо властивості Дата за замовчуванням значення День тижні. У списку День тижні вибираємо Понеділок.

Оскільки ємність машини вимірюється лише кількістю пасажирів, властивість Число характеристик ємності повинно мати значення 1. Якби місткість вимірювалася числом пасажирів і максимальним числом крісел-колясок, число характеристик ємності мало б значення 2. Значення інших властивостей залишаємо за замовчуванням (рис. 36).

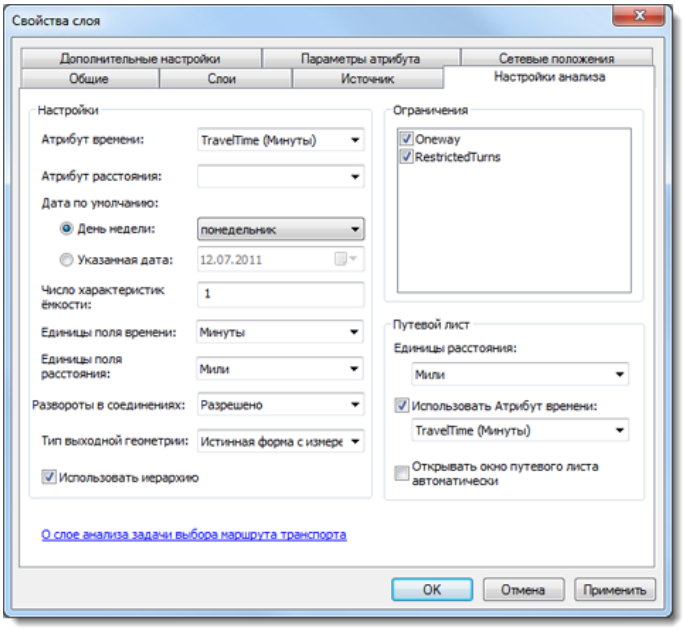

Рисунок 36 – Вікно Властивості шару

Запускаємо процес для знаходження рішення. Обираємо на кнопці Розрахунок (Solve) на панелі інструментів Network Analyst. Інструмент вирішення задачі VRP вирахує маршрути для кожної машини. Кожен маршрут починається на автобазі, підбирає одного або двох чоловік, якщо час, який вони можуть провести в машині, менше значення властивості MaxTransitTime, зазначеного в парному замовленні, відвозить їх в потрібну лікарню, продовжує підбирати і відвозити інших людей і в кінці повертається на автобазу . Маршрути обслуговують тільки замовлення в заданих зонах маршрутів (рис. 37).

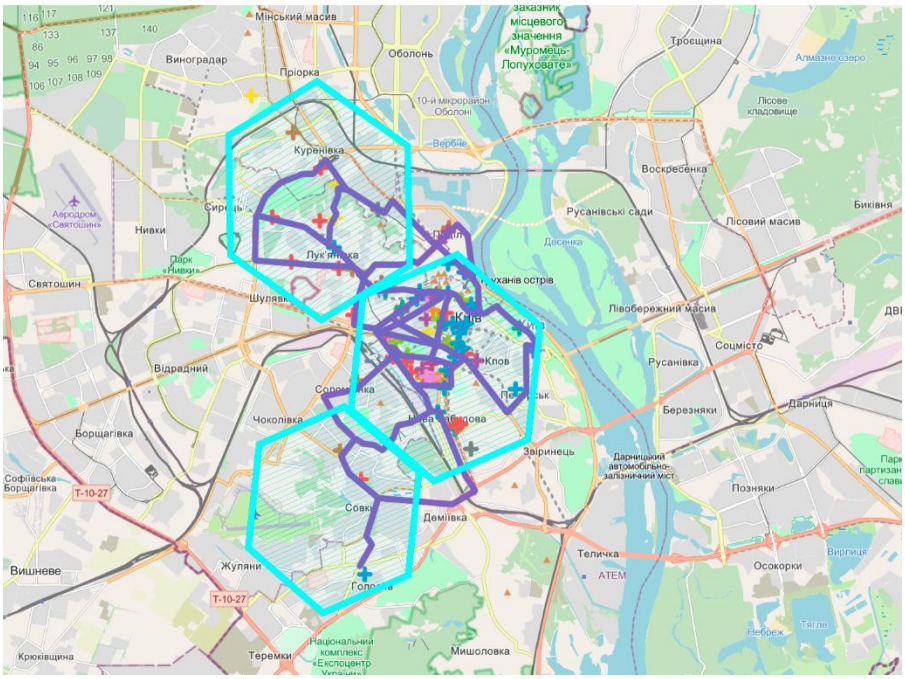

Рисунок 37 – Побудовані маршрути на карті

Для визначення покрокових подорожніх листів для маршрутів обираємо на кнопці Вікно напрямків (Directions Window) на панелі інструментів Network Analyst. Відкриється діалогове вікно Напрями (Directions).

При необхідності також можна експортувати завдання по знаходженню транспортного маршруту в якості файлу шару (<ім'я файлу> .lyr) на диск, щоб його можна було завантажити в інший документ карти.

ВИСНОВКИ

В даній роботі було розглянуто завдання побудови оптимального маршруту. Була проаналізована предметна область, що включає в себе опис та історію завдання комівояжера. Аналіз існуючих додатків дозволив побачити їхні недоліки та правильно поставити завдання дипломної роботи.

Було проведено аналіз методів вирішення задачі знаходження оптимального шляху. Алгоритми для вирішення завдання комівояжера можна розділити на точні і неточні. Точні алгоритми включають в себе перебір всіх можливих варіантів. Неточні алгоритми використовуються для задач, які неможливо вирішити точно (обчислення певних інтегралів, рішення нелінійних рівнянь, витяг квадратного кореня), якщо існуючі точні рішення вимагають значних і невиправданих витрат часу при високій складності завдання.

В роботі подано короткий опис алгоритмів цих груп, перераховані їхні переваги і недоліки. Більш детально був описаний генетичний алгоритм, також підведені підсумки порівняльного дослідження алгоритму з рядом інших. За результатами даного порівняння було виявлено, що для задач з невеликою кількістю місць, до яких відноситься практичне завдання дипломної роботи, одним з найбільш ефективних є генетичний алгоритм.

За допомогою сервісу ArcGIS було виконано аналіз завдання знаходження транспортного маршруту з урахуванням парних замовлень, мета якого – знайти найбільш оптимальні маршрути для автопарку, щоб перевозити людей з дому до лікарні для проходження лікування. Завдання було виконано шляхом вирішення задачі VRP з урахуванням парних замовлень. Після визначення маршрутів створюються покрокові вказівки результуючих маршрутів, які можна відправити електронним способом або роздрукувати і передати водіям.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Geunes J. Operations Planning: Mixed Integer Optimization Models
2. (Operations Research Series). CRC Press, 2014. P. 167–172.
3. Яндекс.Карты. https://yandex.ru/maps
4. Google Maps https://maps.google.com/
5. WebGL public wiki https://www.khronos.org/webgl/wiki/Main_Page
6. Cook W. J. In Pursuit of the Traveling Salesman: Mathematics at the Limits of Computation. Princeton University Press, 2012. P. 19–39.
7. Applegate D.L., Bixby R.E., Chvátal V., Cook W.J. The Traveling Salesman Problem. Princeton University Press, 2007. P. 44–52.
8. Левитин А. Алгоритмы. Введение в разработку и анализ. Вильямс, 2006. P. 160.
9. Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. Мир, 1982.
10. Okano H. Study of Practical Solutions for Combinatorial Optimization Problems. Tohoku University, 2009. P. 14–17.
11. Meganavigator. http://meganavigator.com/
12. Логист. http://logist.poncy.ru/
13. Speedy Route. https://www.speedyroute.com/
14. Левитин А. Алгоритмы. Введение в разработку и анализ. Вильямс, 2006. 35–36 с.
15. Kona H., Burde A., Dr. Zanwar D. R. A Review of Traveling Salesman Problem with Time Window Constraint // IJIRST – International Journal for Innovative Research in Science & Technology, 2015. Vol. 2, Issue 1. P. 253–254.
16. Tannenbaum P. Excursions in Mathematics. University of Kansas, 2011. P. 25.

17. Clausen J. Branch and Bound Algorithms – Principles and Examples. University of Copenhagen, 1999. P. 5–6.
18. Tollis I. G. Algorithms and Complexity. University of Crete, 2000. P. 140–146.
19. Applegate D., Bixby R., Chvatal V., Cook W. TSP cuts outside the template paradigm. Donet, 2000. P. 1–10.
20. Захарова Е.М., Минашина И.К. Обзор методов многомерной оптимизации // Информационные процессы, 2014. Том 14, № 3. стр. 265–266.
21. Papadimitriou C., Vazirani U. Efficient Algorithms and Intractable Problems. Berkeley EECS, 1999. Chapter 10.
22. Goodrich M., Roberto T. Algorithm Design and Applications, Wiley, 2015. P. 513–514.
23. Nilsson C. Heuristics for the Traveling Salesman Problem. Linkoping University, 2011. P. 1–6.
24. Alsalibi B.A., Jelodar M.B., Venkat I. A Comparative Study between the Nearest Neighbor and Genetic Algorithms: A revisit to the Traveling Salesman Problem // International Journal of Computer Science and Electronics Engineering (IJCSEE), 2013. Vol. 1, Issue 1.
25. Gutin G.,Yeo A. The Greedy Algorithm for the Symmetric TSP. University of London, 2002. P. 1–2.
26. Gutin G., Karapetyan D. Lin-Kernighan Heuristic Adaptations for the Generalized Traveling Salesman Problem. Royal Holloway London University, 2010. P. 1–5.
27. Gupta R., Chauhan C., Pathak K. Survey of Methods of Solving TSP along with its Implementation using Dynamic Programming Approach. // International Journal of Computer Applications, 2012. Vol. 52, No.4. P. 1–6.

28. Basu S. Tabu Search Implementation on Traveling Salesman Problem and Its Variations: A Literature Survey. Indian Institute of Management Calcutta, 2012. P. 1–8.
29. Dorigo M., Caro G. D. Ant Algorithms for Discrete Optimization // Artificial Life, 1999. Vol. 5, No 2. P. 139–140.
30. Dorigo M., Gambardella L. M. Ant colonies for the traveling salesman problem. Université Libre de Bruxelles, 1996. P. 1–4.
31. Johnson D. S., McGeoch L. A., Rothberg E. E. Asymptotic Experimental Analysis for the Held-Karp Traveling Salesman Bound. AT&T Bell Laboratories, 1999. P. 1–2.
32. Abdulkarim H.A., Alshammari I. F. Comparison of Algorithms for Solving Traveling Salesman Problem. // International Journal of Engineering and Advanced Technology, 2015. Vol.4, Issue 6. P. 76–79.
33. QGIS - лучшая настольная ГИС с открытым исходным кодом. https://www.qgis.org/ru/site/about/index.html
34. Leverage Your Geospatial Data with GeoMedia for GIS and Mapping. https://www.hexagongeospatial.com/products/power-portfolio/geomedia
35. Mapnfo Pro – Мировой лидер на рынке ГИС и картографических приложений. http://www.mapinfo.ru/node/211
36. ESRI Ukraine. http://www.esri.ua/sarticle.php?id=5