

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
промышленных технологий,
электроэнергетики и транспорта
.....
(наименование)

Кафедра
«Транспортные машины»
.....
(наименование)

Направление подготовки

23.03.01 «Технология транспортных
процессов»
.....
(код и наименование)

Профиль

«Организация и безопасность движения»
.....
(наименование)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
на тему

Оптимизация маршрута перевозок, сети магазинов «Магнит» в
Ленинском районе г. Пенза на основе теории графов.

Студент

.....
(подпись, дата)

Салов Вадим Олегович
.....
(ФИО полностью)

Руководитель работы:
к.т.н., доцент

.....
(подпись, дата)

Генералова А.А.
.....
(фамилия, инициалы)

Нормоконтролер:
к.т.н., доцент

.....
(подпись, дата)

Каледа В.Н.
.....
(фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от *10 июня 2020 г. № 9а*)

Заведующий кафедрой:
д.т.н., профессор

.....
(подпись, дата)

Салмин В.В.
.....
(фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ 2020 г. № _____)

Секретарь ГЭК:
к.т.н., доцент

.....
(подпись, дата)

Жук А.П.
.....
(фамилия, инициалы)

Пенза 2020

ФГБОУ ВО «ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ
КАФЕДРА «ТРАНСПОРТНЫЕ МАШИНЫ»

«Утверждаю»
Заведующий кафедрой ТМ

д.т.н., профессор. _____ В.В. Салмин
подпись

« 28 » ноября 2019 г.

ЗАДАНИЕ
на выпускную квалификационную работу

Студенту гр. 16МГ1 Политехнического института, очной формы обучения по направлению 23.03.01 «Технология транспортных процессов»

Салову Вадиму Олеговичу

Научный руководитель работы: к.т.н., доцент Генералова А.А.

Время выполнения ВКР: с 28.11. 2019 г. по 24.05.2020 г.

Место преддипломной практики: кафедра «Транспортные машины» ПГУ

Тема работы: «Оптимизация маршрута перевозок, сети магазинов «Магнит» в Ленинском районе г. Пенза на основе теории графов»

Тема утверждена приказом ректора ПГУ № 1273а/о от «28» ноября 2019 г.

Техническое задание на работу (назначение устройства, условия применения, внешние воздействия, специальные требования и т. п.): *разработать программу по оптимизации маршрута перевозок сети супермаркетов «Магнит» путем поиска кратчайшего пути на основе метода теории графов.*

Объем и содержание основной части работы

Пояснительная записка (перечень вопросов, подлежащих разработке, расчетов, обоснований, описаний):

- введение с обоснованием актуальности темы исследования, постановкой цели и задач исследования – 1 – 2 страницы;
- анализ УДС Ленинского района г. Пензы, теоретические основы транспортной логистики – 10 – 20 страниц;
- организационно-управленческий раздел – анализ и выбор алгоритма поиска оптимального пути; понятие генетического алгоритма; реализация поиска – 10 – 20 страниц;
- экономический раздел – экономическая оценка расхода топлива по маршруту различным видом транспорта – 10 – 20 страниц;
- заключение – выводы по результатам проведенных исследований – 1-2 страницы.

Графическая часть (перечень и содержание чертежей, плакатов)

1 . «Актуальность, цель и задачи ВКР»	2 л., А1
2 . «Географическое расположение супермаркетов «Магнит» Ленинского района г. Пензы»	1 л., А1
3 . «Граф супермаркетов «Магнит» Ленинского района г. Пензы»	1 л., А1
4 . «Матрица смежности супермаркетов «Магнит» Ленинского района г. Пензы.»	1 л., А1
5. Расчётное время решения задачи коммивояжёра методом полного перебора	1 л., А1
6 . «Схема работы генетического алгоритма»	1 л., А1
7 . «Оптимальный маршрут следования»	1 л., А1
8 . «Расчет нормы расхода топлива»	1 л., А1
9 . «Выводы по результатам ВКР»	1 л., А1

Рекомендуемая литература:

Рекомендуемая литература:

1. Ашарина, И.В. Основы программирования на языках C и C++: Курс лекций для высших учебных заведений / И.В. Ашарина. — М.: Гор. линия-Телеком, 2012. — 208 с.
2. Белоусова, С.Н. Основные принципы и концепции программирования на языке VBA в Excel: Учебное пособие / С.Н. Белоусова, И.А. Бессонова. — М.: БИНОМ. ЛЗ, 2010. — 200 с
3. Еремеева Л.Э. Транспортная логистика: учебное пособие / Л. Э. Еремеева; Сыкт. лесн. ин-т. — Сыктывкар: СЛИ, 2013. — 260 с.
4. Панченко, Т. В. Генетические алгоритмы: учебно-методическое пособие / под ред. Ю. Ю. Тарасевича. — Астрахань : Издательский дом «Астраханский университет», 2007 — 87 с.

Дата выдачи задания: « 28 » ноября 2019 г.

Руководитель дипломного проекта:

к.т.н., доцент _____ А.А. Генералова

Задание к исполнению принял студент

группы 16МГ1 _____ В.О. Салов

« 28 » ноября 2019 г.

ФГБОУ ВО «ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ
КАФЕДРА «ТРАНСПОРТНЫЕ МАШИНЫ»

«УТВЕРЖДАЮ»

Руководитель ВКР

к.т.н., доцент _____ А.А. Генералова
подпись

« 29 » ноября 2019 г.

КАЛЕНДАРНЫЙ ПЛАН

выполнения выпускной квалификационной работы
*«Оптимизация маршрута перевозок, сети магазинов Магнит в
Ленинском районе г. Пенза на основе теории графов»*
студенту учебной группы 16МГ1 **Салову Вадиму Олеговичу**
(направление 23.03.01 Технология транспортных процессов)

Наименование выполняемой работы	Срок исполнения		Отметка о выполнении
	начало	окончание	
1	2	3	4
1 Получение задания, его уяснение, беседа с руководителем. Подготовка к выполнению ВКР.	28.11.19	30.11.19	
2 Подбор и изучение рекомендованной литературы.	05.12.19	18.12.19	
3 Изучение объекта исследования, постановка цели и задачи работы. Написание введения.	19.12.19	18.01.20	
4 Анализ улично-дорожной сети Первомайского района г. Пензы, представление географического расположения супермаркетов «Магнит» Первомайского района г. Пензы. Написание первого раздела работы.	20.01.20	23.01.20	
5 Анализ существующих сервисов и их недостатки.	24.01.20	24.02.20	
6 Написание второй главы работы.	25.02.20	24.03.20	
7 Экономическое обоснование предлагаемых мероприятий. Написание третьей главы.	05.12.19	18.12.19	

1	2	3	4
8 Оформление пояснительной записки, схем и рисунков.	24.03.19	18.05.20	
9 Представление пояснительной записки и графической части руководителю на просмотр и подготовка к предварительной защите.	18.05.20	24.05.20	
10 Представление ВКР нормоконтролёру	25.05.20	30.05.20	
11. Проверка ВКР на плагиат	01.06.20	06.06.20	
12 Предварительная защита ВКР	04.06.20	05.06.20	
13 Полное завершение всех работ и представление зав. кафедрой на подпись пояснительной записки (совместно с графической частью и отзывом руководителя).	12.06.20	22.06.20	
14 Защита выпускной квалификационной работы	24.06.20	26.06.20	

Студент учебной группы 16МГ1 _____ В. О. Салов

« 29 » ноября 2019 г

РЕФЕРАТ

УДК 656.1/5: 517.977.58

ББК 39.38

Салов, В.О. Оптимизация маршрута перевозок, сети магазинов «Магнит» в Ленинском районе г. Пенза на основе теории графов / Салов Вадим Олегович. – Пенза: ПГУ, 2020. – 64 страницы, 3 части, 20 рисунков, 6 таблиц, 32 источника.

ТРАНСПОРТНАЯ ЛОГИСТИКА, УЛИЧНО-ДОРОЖНАЯ СЕТЬ, ОПТИМАЛЬНЫЙ МАРШРУТ, АЛГОРИТМ, ТРАНСПОРТИРОВКА, СЕРВИС, ТРАНСПОРТНЫЙ ГРАФ, МИКРОРАЙОНИРОВАНИЕ, МАТРИЦА СМЕЖНОСТИ, ГЕНЕТИЧЕСКИЙ АЛГОРИТМ.

Выпускная квалификационная работа состоит из введения, трёх глав, заключения, списка литературы и приложения

В первой главе проводится анализ улично-дорожной сети Ленинского района г. Пензы, представлено географическое расположение супермаркетов «Магнит» Ленинского района г. Пензы.

Во второй главе проводится анализ и выбор алгоритма поиска оптимального пути, произведена реализация поиска.

В третьей главе рассчитывается стоимость поездки по маршруту на автомобилях разных марок

Объектом исследования являются транспортные потоки перевозки грузов супермаркетов «Магнит» г. Пензы.

Предмет исследования – маршрут перевозок, сети магазинов «Магнит» Ленинского района г. Пензы.

Цель работы – построение оптимального маршрута сети магазинов «Магнит» Ленинского района города Пензы.

Для реализации поставленной цели необходимо решить следующий ряд задач:

1. Рассмотреть основные алгоритмы поиска оптимального маршрута;
2. Провести анализ данных алгоритмов и выбрать один из них для практической реализации;
3. Написать код выбранного алгоритма, используя программу Excel, на языке программирования VBA
4. Реализация алгоритма оптимизации в программе для работы с электронными таблицами Microsoft Excel на языке VBA.

ESSAY

UDC 656.1/5:517.977.58
BBK 39.38

Salov, V.O. Optimization of the transportation route, the Magnit store chain in the Leninsky district of Penza based on graph theory / Salov Vadim Olegovich. - Penza: PSU, 2020. - 64 pages, 3 parts, 20 figures, 6 tables, 32 sources.

TRANSPORT LOGISTICS, STREET ROAD NETWORK, OPTIMAL ROUTE, ALGORITHM, TRANSPORTATION, SERVICE, TRANSPORT GRAPH, MICRO-ZONING, MATRIX OF ADJACENCY, GENETIC ALGORITHM.

Final qualification work consists of an introduction, three chapters, conclusion, bibliography and applications

The first chapter analyzes the road network of the Leninsky district of Penza, presents the geographical location of the Magnit supermarkets of the Leninsky district of Penza.

The second chapter analyzes and selects the optimal path search algorithm, and searches are implemented.

The third chapter calculates the cost of a trip on a car of different brands

The object of the study is the traffic flows of the cargo transportation of the Magnit supermarkets in Penza.

The subject of the study is the transportation route, the Magnit chain of stores in the Leninsky district of Penza.

The purpose of the work is to build the optimal route for the Magnit store chain in the Leninsky district of the city of Penza.

To achieve this goal it is necessary to solve the following series of tasks:

1. Consider the basic algorithms for finding the optimal route;
2. To analyze these algorithms and choose one of them for practical implementation;
3. Write the code of the selected algorithm using Excel in the VBA programming language
4. Implementation of the optimization algorithm in a program for working with Microsoft Excel spreadsheets in VBA.

Содержание

Введение	9
ГЛАВА 1.Обзор задачи	11
1.1 Теоретические основы транспортной логистики	11
1.1.1 Понятие и сущность транспортной логистики	11
1.1.2 Характеристика автомобильного транспорта и его инфраструктуры	18
1.1.3Транспортные тарифы и документальное оформление перевозки грузов .	20
1.2 Анализ улично-дорожной сети Ленинского района г. Пензы	23
1.2.1 Географическое расположение супермаркетов «Магнит» Ленинского района г. Пензы	25
1.3 Анализ существующих сервисов и их недостатки	26
1.4 Основные положения теории графов	29
1.5 Выводы по первой главе	39
ГЛАВА 2. МАТЕМАТИЧЕСКАЯ ОСНОВА ПРОЕКТА	40
2.1 Анализ и выбор алгоритма поиска оптимального пути	40
2.2 Понятия генетического алгоритма	42
2.3 Реализация поиска	48
2.4 Вывод по второй главе	50
3. Экономический раздел	51
3.1 Экономия топлива и факторы влияющие на нее	51
3.2 Расчет нормы расхода топлива	52
3.3 Вывод по третьей главе	58
ЗАКЛЮЧЕНИЕ	59
СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ	60
Приложение А Поиск решения в Excel	63
Приложение Б Листинг программного кода.....	64

Введение

Одним из направлений организации транспортной логистики является оптимизация не только расходов по задействованию автотранспортных средств на предприятии, но также и оптимизации самих перевозок. При этом одна из самых важных и затратных областей – это логистика транспортных перевозок. Её компетенция распространяется от доставки материалов на производство до развоза готового товара конечным покупателям.

Первоочередной задачей любого предприятия является поиск резервов снижения затрат на осуществляемую деятельность и как следствие повышение собственной конкурентоспособности и рентабельности. В современных условиях поиск таких резервов строится на основе логистического подхода, что связано с расширением содержания логистики, превращающейся из вспомогательного элемента, обеспечивающего реализацию хозяйственных процессов, в важный инструмент организации и ведения хозяйственной деятельности.

Грамотно организованная система перевозок позволяет сократить расходы на содержание автопарка, а также минимизировать риски задержек и порчи груза. Именно для этого необходимо постоянно оптимизировать систему транспортной логистики, тем самым уменьшать затраты без потери качества перевозки. Это связано в первую очередь со структурой логистических затрат, значительную долю в которых (20-40% и более) составляют именно расходы на транспортную составляющую.

Целью данной работы является построение оптимального маршрута сети супермаркета «Магнит» Ленинского района города Пензы.

Для достижения указанной цели поставлены следующие задачи:

1. Рассмотреть основные алгоритмы поиска оптимального маршрута;
2. Провести анализ данных алгоритмов и выбрать один из них для практической реализации;

3. Написать код выбранного алгоритма, используя программу Excel, на языке программирования VBA.

4. Реализация алгоритма оптимизации в программе для работы с электронными таблицами Microsoft Excel на языке VBA.

Важность исследования обусловлена тем, что в условиях рыночной экономики только высокоорганизованные предприятия с качественным управлением могут добиться эффективной работы. Разрабатываемые предложения должны быть направлены на совершенствование транспортного процесса, повышение эффективности использования подвижного состава, сокращение его простоев, порожних пробегов и нерациональных перевозок.

В конечном счете реализация данных предложений способна обеспечить положительный эффект непосредственно на автотранспортном предприятии, улучшить основные показатели его работы и принести дополнительную прибыль.

1 ОБЗОР ЗАДАЧИ

1.1 Теоретические основы транспортной логистики

1.1.1 Понятие и сущность транспортной логистики

Транспортная логистика – это новая первостепенная дисциплина в средних и высших транспортных учебных заведениях на всех уровнях обучения, которая дополняет цикл «рыночных» фундаментальных дисциплин: рынок транспортных услуг и транспортной техники, маркетинг, менеджмент (на транспорте), управление персоналом (на транспорте) и так далее.

Транспортная логистика является видом логистики, которая управляет комплексом операций, осуществляющих физическое перемещение товарно-материальных ценностей (ТМЦ) между участниками логистической цепи с наименьшими затратами, то есть перемещение необходимого количества товара в требуемую точку, наилучшим маршрутом за нужное время и с минимальными издержками. Расходы на изготовление любого товара формируются из себестоимости его изготовления и затрат на выполнение всех операций в цепочке «производитель – конечный покупатель». На перемещение материального потока от источника сырья до конечного потребителя необходимы затраты, которые могут достигать 50 % от общей стоимости затрат на логистику.

Предмет транспортной логистики – это рациональная организация процессов движения грузов. К задачам, которые решает транспортная логистика, эксперты относят:

- 1) формирование транспортных систем, в том числе транспортных цепей и транспортных коридоров;
- 2) гарантия технологической целостности транспортно-складского процесса;
- 3) совместное составление плана транспортного процесса с производственным и складским;
- 4) определение оптимального маршрута доставки груза;

- 5) выбор вида и типа транспортного средства;
- 6) совместное составление плана транспортных процессов на разных видах транспорта [8].

Учитывая то, что логистика является отраслью знаний об управлении транспортными потоками, организация ее применения должна основываться как на фундаментальных пяти принципах теории управления, так и на своих специфичных принципах, углубляющих процессные процедуры логистики.

1. Системность, включающая в себя организацию потока, выделение изменяющихся и перемещающихся объектов в отдельную управляемую подсистему и применение в отношении нее системного подхода; связь расходов на некоторые операции по перевозкам и поставкам товаров со стратегическим планом предприятия; достижение взаимодействия логистики с производством и маркетингом; организация планирования, сбыта, производства, закупок, транспортировки и хранения как единого материального потока логистической системы.

2. Комплексность, которая включает в себя формирование всех видов обеспечения (развитой инфраструктуры) для реализации движения потоков в определенных условиях; координацию действий опосредованных и непосредственных участников движения продуктов и ресурсов; реализацию централизованного контроля по выполнению задач, стоящих перед логистическими структурами предприятия; стремление компаний к тесному и взаимовыгодному сотрудничеству с внешними партнерами по товарной цепочке и нахождению крепких связей между различными подразделениями предприятия в пределах внутренней деятельности.

3. Научность, то есть усиление расчетного начала на всех этапах управления потоком: начиная от планирования и заканчивая анализом, выполнение сходных расчетов всех параметров траектории перемещения материального потока; признание того, что квалифицированные кадры имеют статус важного ресурса предприятия.

4. Конкретность, то есть точное установление определенного результата в качестве цели движения потока согласно техническим, экономическим и другим требованиям; реализация перемещения с минимальными затратами всех видов ресурсов.

5. Конструктивность - это диспетчеризация потока, постоянный контроль изменения и движения каждого элемента потока и своевременная корректировка его перемещения; тщательное выявление деталей всех действий материально-технического обеспечения и транспортировка продукции.

6. Надежность, включающая обеспечение безопасности и безотказности перемещения, сохранение технических средств и коммуникаций с целью изменения траектории движения потока при необходимости; широкое применение современных технических средств движения и управления перемещением; повышение качества и скорости поступающей информации и совершенствование технологии ее обработки.

7. Вариантность, то есть возможность адекватного реагирования предприятия на изменяющийся спрос; целенаправленное формирование сохранённых мощностей, загрузка которых выполняется согласно заранее разработанным резервным планам предприятия.

8. Интегративность. Существующие интегративные свойства показывают, что качество системы хотя и зависит от характеристик ее компонентов, но не полностью ими определяется. Систему не сводят к простой совокупности элементов; разделяя ее на отдельные компоненты и изучая каждый из них по отдельности, невозможно оценить параметры системы в целом. Противоположная сторона показывает, что логистическая система считается эффективной, если она содействует всем элементам цепи: «снабжение – производство – сбыт – потребление» и вносит свой вклад в достижение общей цели – оптимизации общих логистических издержек по Парето.

9. Эффективность - это способность логистической системы. При данной степени развития производственных технологий, рыночных отношений, при

определенных субъектах этой системы дойти до возможного минимума логистических затрат.

10. Гибкость, то есть встроенность в логистическую систему механизмов, которая даёт возможность прогнозировать вероятные изменения состояния внешней экономической среды и разрабатывать соответствующие им действия.

11. Целостность, включающая сотрудничество доведению воздействий управляющих структур до всех структурных компонентов логистической системы, формированию информационного содействия между ними, которое направлено на достижение поставленных целей логистической системы. Предусмотрен анализ логистической системы как единой целой, состоящей из взаимодействующих элементов, часто разнородных и разнокачественных, но совместно ориентированных на конечные результаты логистической системы.

12. Превентивность. В противоположной девиантной концепции управления, которая нацелена не на предупреждение диспропорций, отклонений, а на вероятное устранение их негативных последствий, в логистике допускается только превентивная концепция управления, предупреждающая возникновение вышеуказанных проблем.

При создании логистических систем должны учитываться такие принципы, как:

- принцип логической последовательности проектирования системы, характеризующийся тем, что сначала проводится изучение макросреды и ее функционирования, а потом на микроуровне исследуется взаимодействие звеньев и модулей внутри системы;

- принцип постоянного, на всех элементах логистической системы, согласования информационных, ресурсных, энергетических и других параметров системы;

- принцип единства целей всей системы и ее отдельных подсистем (модулей).

В процесс организации транспортной логистики предприятия входит выполнение следующих операций:

- выбор вида транспортировки;
- выбор типа транспорта;
- предназначение транспортного средства;
- выбор логистических провайдеров, то есть подрядчиков по транспортировке, и перевозчика;
- оптимизация характеристик транспортного процесса.

В процесс разработки стратегии транспортно-логистического обеспечения предприятия входят следующие этапы:

- изучение вариантов;
- изучение цен;
- накопление груза, его сортировка на складе;
- изучение и анализ поставщиков;
- использование различных видов транспорта;
- тесные взаимоотношения с выбранными перевозчиками;
- исследование, анализ затрат;
- внешние поставщики транспортных услуг, договоры с внешними источниками;
- задачи безопасности.

Одной из ключевых задач транспортной логистики считается организация технологии перевозки.

Транспортно-технологическим процессом называется перемещение груза (товара) от места его изготовления до места потребления. Как любой производственный процесс, он включает в себя отдельные последовательно выполняемые операции.

Несмотря на последовательность технологических элементов, включающих в себя подачу автомобильного транспортного средств к пункту погрузки, размещение в кузове транспортного средства груза, доставку груза

грузополучателю, разгрузку и порожний пробег к пункту погрузки, кажется однозначной, вероятными являются разные варианты выполнения данных операций.

Как уже упоминалось выше, каждый транспортно-технологический процесс включает в себя:

- процесс перевозки грузов (товара) каким-либо видом транспорта;
- транспортно-экспедиционные операции, предшествующие перевозке и завершают ее.

К числу операций, которые являются обязательным компонентом процесса перевозки, относят:

- составление плана потребности определенного количества транспортных средств, составление и представление своевременных заявок на нужный транспорт;
- приведение груза в состояние, необходимое для транспортировки, – подбор тары и упаковка товара или груза, маркировка тарных зон, обеспечение бирками и тому подобное;
- доставка и передача грузов на станции отправления, приём и вывоз со станции поступивших грузов и движение их на склады или в магазины торговой компании;
- документальное оформление приемки и сдачи грузов на станциях отправления и назначения, заполнение перевозочных и иных документов, оплата провозных и иных платежей, получение информации о поступивших грузах и так далее;
- погрузочно-разгрузочные работы, которые выполняются при сдаче и приемке грузов на станциях, складах, в магазинах.

На рис. 1.1 показаны схемы процессов перевозки грузов. Они обладают циклическим характером. Это значит, что движение грузов происходит по повторяющемуся производственному циклу. Темп этого цикла обуславливается частотой, которая зависит от средней длительности цикла, который отличается

большой степенью динамизма, постоянной сменой состояния и изменением состава компонентов.



Рисунок 1.1 – Технологическая схема процесса перевозки грузов одним видом транспорта

Различаются такие этапы процесса перевозки, как:

- доставка подвижного состава на погрузку;
- подготовка груза к отправке;
- хранение груза в месте производства и на промежуточных пунктах;
- складирование;
- экспедиторские операции и тому подобное.

Транспортные компании на первый план выдвигают задачи оптимизации использования подвижного состава и уменьшения времени его оборота. Процесс перевозки – это совокупность операций, начиная от подготовки груза к отправлению и заканчивая его получением, сопряженных с передвижением груза в пространстве без изменения его размеров, геометрических форм и физико-химических свойств (этапы 1 – 2 – 3 – 4 – 5 на рис. 1.1).

Аспект логистики в процессе организации автомобильных перевозок состоит из нового методологического содержания, которое заключается в главной составляющей части перевозок, где должно быть проектирование рационального перевозочного процесса. Под этим подразумевается поиск наилучших технически возможных и организационных решений, которые

обеспечивают высокую эффективность перевозки грузов от места их изготовления до места потребления [8].

1.1.2 Характеристика автомобильного транспорта и его инфраструктуры

Автомобильный транспорт имеет ряд технико-экономических специфических черт, определяющих его достоинства и широкое применение в различных отраслях народного хозяйства, к ним относятся:

- большая подвижность и маневренность. Автомобили перевозят грузы из пункта производства непосредственно в пункт потребления без

- промежуточного складирования и перегрузки, то есть «от двери до двери»;

- большая скорость перемещения грузов. Автомобильный транспорт уступает по скорости передвижения только воздушному транспорту;

- в некоторых случаях – более короткий путь движения грузов. Целесообразно доставлять грузы автомобильным транспортом тогда, когда расстояние передвижения по автомобильным дорогам меньше, чем по железным дорогам.

Недостатки автомобильного транспорта:

- высокая себестоимость. Она существенно выше, чем на железнодорожном и водном транспорте. Высокая себестоимость образуется в результате небольшой грузоподъемности единицы подвижного состава и высокой сложности автомобильного подвижного состава;

- сравнительно высокая стоимость материально-технической базы обслуживания автомобилей, хотя в некоторых случаях эта база развита недостаточно;

- небольшая длина и плохое техническое состояние существующих автомобильных дорог [9].

Автомобильный подвижный состав можно разделить на грузовой, пассажирский и специальный.

К грузовому подвижному составу относят грузовые автомобили, автомобили-тягачи, прицепы, полуприцепы.

Пассажирский подвижный состав включает легковые автомобили, автобусы, пассажирские прицепы и полуприцепы, предназначенные для выполнения разных не транспортных работ.

Автотягач в сцепе с прицепом (полуприцепом) называется автопоездом. Пассажирский автомобиль вместимостью меньше 8 человек, включая водителя, относят к легковым, а вместимостью более 8 человек – к автобусам.

Есть ещё одно разделение автомобилей. Они делятся в зависимости от рода потребляемого топлива и вида двигателя, бывают: дизельные, карбюраторные, газогенераторные, газобаллонные, газотурбинные, электрические и паровые.

Все автомобили, в зависимости от числа колес и числа ведущих колес, условно обозначаются формулой, где первая цифра означает число колес, а вторая – число ведущих колес (4×2 и т.д.).

Автомобильный транспорт характеризуется установленным типажом транспортных средств и производственно-технической базой, необходимой для организации технического обслуживания, эксплуатации и ремонта транспортных средств. В инфраструктуре транспортной логистики выделяют две ключевые составляющие:

- транспортная сеть: автомобильные дороги; постоянные технические сооружения и устройства (депо, средство связи и управления и т.п.);
- подвижной состав автомобильного транспорта.

Для подвижного состава существуют следующие основные параметры:

- эксплуатационная и техническая скорость;
- габаритные размеры грузовых емкостей и самих транспортных средств;
- мощность двигателя;
- полная масса, нагрузка на оси;
- габаритные размеры прицепов и грузоподъемность и т.п.

Для путей сообщения основными параметрами являются:

- ширина проезжей части;
- пропускная способность;
- допустимая нагрузка на дорожное полотно.

Для терминалов характерны следующие показатели:

- число оборотов;
- полезная складская площадь;
- производительность складского и подъемно-транспортного оборудования и тому подобное.

1.1.3 Транспортные тарифы и документальное оформление перевозки грузов

Необходимо изучить потребности клиентуры для того, чтобы предложить пакет транспортных услуг. Очень часто на транспорте некоторых промышленно развитых государств изучением потребностей клиентов занимаются специальные логистические центры и другие структуры. Они проводят исследование грузопотоков и распределение их по сети.

Расчеты за услуги, которые оказывают транспортные предприятия, осуществляются при помощи транспортных тарифов.

Транспортные тарифы - это механизмы формирования оплаты предприятиям-перевозчикам за осуществляемые транспортные и им сопутствующие услуги.

Тарифы включают в себя:

- платы, которые взыскиваются за перевозку грузов;
- сборы за дополнительные действия и операции, связанные с перевозкой грузов;
- правила исчисления сборов и плат.

Для того, чтобы определить стоимость перевозки грузов на автомобильном транспорте используют следующие виды тарифов:

- сдельные тарифы на перевозку грузов;
- тарифы за повременное пользование грузовыми автомобилями;
- тарифы на перевозку грузов на условиях платных автотонно-часов;
- тарифы за перегон подвижного состава;
- тарифы из покилометрового расчета;
- договорные тарифы.

На величину тарифной платы влияют следующие факторы:

- 1) расстояние перевозки;
- 2) масса груза;
- 3) объемная масса груза, характеризующаяся возможностью автомобиля (по данному показателю все грузы, которые перевозятся автомобильным транспортом, подразделяются на четыре класса);

- общий пробег;
- грузоподъемность автомобиля;
- тип автомобиля;
- время использования автомобиля;
- район, в котором выполняется перевозка, а также некоторые другие факторы.

Каждый тариф на перевозку грузов при помощи автомобильного транспорта учитывает не весь комплекс факторов, а только некоторые из них, которые являются наиболее существенными в условиях определенной перевозки. Например, для определения стоимости перевозки по сдельному тарифу обращается внимание на расстояние перевозки, массу груза и его класс, характеризующий степень использования грузоподъемности данного автомобиля.

Если расчеты по тарифу, то за повременное использование грузовых автомобилей учитывается грузоподъемность автомобиля, общий пробег и

время использования. В большинстве случаев размер платы за использование автомобиля зависит от района, в котором выполняется перевозка. Это объясняется устойчивыми различиями уровней себестоимости перевозок грузов по различным районам. Изменения в тарифной стоимости вносятся при помощи поясных поправочных коэффициентов [4].

Документальное оформление перевозки грузов важно для организации транспортной логистики.

Важным является юридический документ, который регулирует взаимоотношения сторон, его называют - договор.

Договор перевозки грузов – это основной транспортный договор, согласно которому перевозчик обязан доставить груз, вверенный ему, в пункт назначения и передать его получателю, а отправитель обязан осуществить установленную плату за перевозку груза.

В систему транспортных договоров по перевозке товарных грузов входят основные и смежные виды договоров.

К основным договорам относят:

- договор перевозки груза;
- договор фрахтования.

К смежным договорам необходимо отнести:

- договор транспортной экспедиции;
- договор погрузки, выгрузки, хранения;
- договор об организации перевозок;
- договор на централизованный завоз (вывоз).

Так как договор перевозки груза имеет консенсуальный характер, то применяют два документа: один из них дает возможность оформить заключение договора перевозки груза, а другой – заверяет принятие груза к перевозке. Именно поэтому заключение договора перевозки должно быть обязательно подтверждено составлением следующих перевозочных

документов: транспортная накладная, дорожная ведомость, квитанция, которые предусмотрены транспортными кодексами и уставами.

При автомобильных перевозках в соответствии с п. 47 Устава автомобильного транспорта отправитель груза должен представить автотранспортной организации товарно-транспортную накладную, составляемую обычно в четырех экземплярах, она является главным перевозочным документом. Без этого документа грузы не принимаются к перевозке. В случае перевозки грузов нетоварного характера (песок, грунт) необходимо оформлять акты взвешивания или замера.

Прием груза от отправителя удостоверяется подписью водителя товарно-транспортной накладной во всех экземплярах, причём один из которых вручается грузоотправителю.

1.2 Анализ улично-дорожной сети Ленинского района г. Пензы

«Магнит» — одна из ведущих розничных сетей в стране. «Магнит» является лидером по количеству магазинов и территории их размещения. По состоянию на 31 декабря 2019 года розничная сеть насчитывала 20725 магазинов: 14622 магазина «у дома», 473 супермаркета «Магнит Семейный» и 5630 магазинов drogery, расположенных в 3742 населенных пунктах.

Розничная сеть ведет свою деятельность в мультиформатной модели, что позволяет «Магниту» удовлетворять разные потребности и желания покупателей.

Эффективный процесс доставки продукции в магазины возможен благодаря мощной логистической системе. Для более качественного хранения товаров и оптимизации их поставки в торговые точки, в компании создана дистрибьюторская сеть, включающая 38 распределительных центров и 35 автотранспортных предприятий. Своевременную доставку продуктов во все магазины розничной сети позволяет осуществить собственный автопарк, на 31 декабря 2019 года автопарк состоял из 5656 автомобилей. Кроме того,

розничная сеть «Магнит» является крупнейшим частным работодателем в России. Общая численность сотрудников компании составляет около 300000 человек. Компании неоднократно присуждалось звание «Привлекательный работодатель года».[20]

В ленинском районе г. Пенза насчитывается 17 универсамов магнит. Связность между магазинами развитая УДС, возможен проезд по дорогам с правилами движения, отсутствуют удалённые магазины, проезд к которым невозможен, Улично-дорожная сеть представляет собой множество переплетенных, скрещенных улиц и дорог.

Степень развития дорожной сети определяется ее протяженностью и плотностью, которая измеряется отношением протяженности дорог к площади территории. Обычно при определении плотности дорожной сети учитывают основные (магистральные) улицы и дороги, а второстепенные не принимают во внимание. Определение оптимальной плотности сети городских магистралей и автомобильных дорог представляет противоречивую задачу. Чем выше плотность дорожной сети, тем чаще пересечения дорог (если не предусматривать строительство развязок в разных уровнях), которые приводят к задержкам ТС и ДТП. Высокая плотность дорожной сети предопределяет снижение скоростей сообщения, что противоречит интересам населения и требованиям экономической эффективности автомобильных перевозок. Важной характеристикой, с помощью которого можно оценить эффективность УДС, является коэффициент непрямолинейности, характеризующий отношение фактического расстояния для проезда по УДС к минимально возможному расстоянию (определяемому по прямой линии). Параметры УДС (например, пропускная способность, загрузки транспортных потоков) весьма существенно влияют на скорость сообщения, которая может быть реализована на маршрутах пассажирских и грузовых перевозок. Эти параметры необходимо знать при планировании и управлении перевозками и разработке конкретных маршрутов.[22]

1.2.1 Географическое расположение супермаркетов «Магнит» Ленинского района г. Пензы.

На рис. 1.2 представлено расположение супермаркетов «Магнит» на карте Ленинского района.

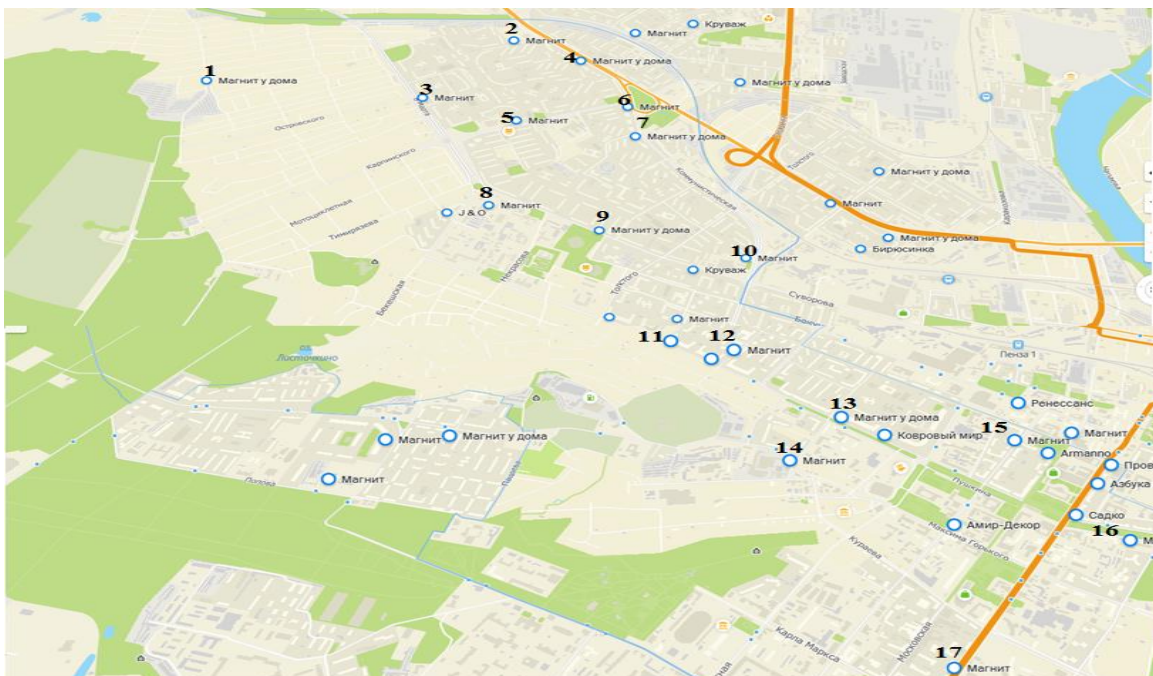


Рисунок 1.2 – Расположение супермаркетов Магнит Ленинского района

На данном рисунке представлено расположение супермаркетов магнит Ленинского района города Пензы. Число супермаркетов составляет 17 шт. Для определения координат положения магазинов используются координаты положения универсамов, полученных при помощи электронного справочника с картами 2GIS.[31] Полученные координаты сведены в таблицу 1.1.

Таблица 1.1 – Расположение универсамов Магнит Ленинского района г. Пензы

№	Адрес универсама	Долгота	Широта
1	Ул. Большая Бугровка 113	53°20.9713'	44°95.3484'
2	Ул. 8 марта 7б	53°21.0926'	44°97.1666'
3	Ул. Фурманова 7а	53°21.6702'	44°97.6427'
4	Ул. Проспект победы 16	53°21.5472'	44°98.3156'

Табл. 1.1 (продолжение)

№	Адрес универсама	Долгота	Широта
5	Ул. Карпинского 37	53°21.0376'	44°97.967'
6	Ул. Проспект победы 8	53°21.2539'	44°98.7602'
7	Ул. Коммунистическая 35	53°20.9988'	44°98.9552'
8	Ул. Суворова 225	53°20.3759'	44°97.9859'
9	Ул. Суворова 168	53°20.4314'	44°98.8689'
10	Ул. Ворошилова 2	53°20.384'	44°99.8463'
11	Ул. Пушкина 45	53°19.8683'	44°99.142'
12	Ул. Пушкина 43а	53°19.9201'	44°99.5094'
13	Ул. Пушкина 15	53°19.6871'	45°00.5578'
14	Ул. Ставского 4	53°19.3272'	45°00.2074'
15	Ул. Володарского 74а	53°19.8322'	45°01.4938'
16	Ул. Славы 10	53°19.4589'	45°02.3894'
17	Ул. Кирова 10а.	53°18.689'	45°01.6906'

1.3 Анализ существующих сервисов и их недостатки

Популярные картографические сервисы вроде Яндекс.Карты и Google Maps не предоставляют пользователям возможность поиска оптимально пути. При вводе нескольких координат сервис выстраивает маршрут в том порядке, в котором данные были введены. Пользователи могут выбирать средства передвижения (на машине, пешком, на наземном транспорте), но все эти настройки влияют исключительно на варианты построения маршрута между его фиксированными точками. Анализ, проведенный путем сравнения десятков как русскоязычных, так и зарубежных картографических сервисов показывает, что среди самых популярных вариантов только у одного доступна функция построения оптимально пути, и далеко не всегда она работает корректно. Ниже представлен подробный отчет о двух наиболее достойных внимания сервисах и их недостатках.[25,29]

Среди русскоязычных сервисов в первую очередь можно выделить Meganavigator [27]. Он включает в себя конструктор карт, визуализацию GPS треков и построение оптимальных маршрутов. Сервис представляет из себя сайт со встроенной Яндекс картой и полями для ввода пунктов маршрута (рис. 1.3).

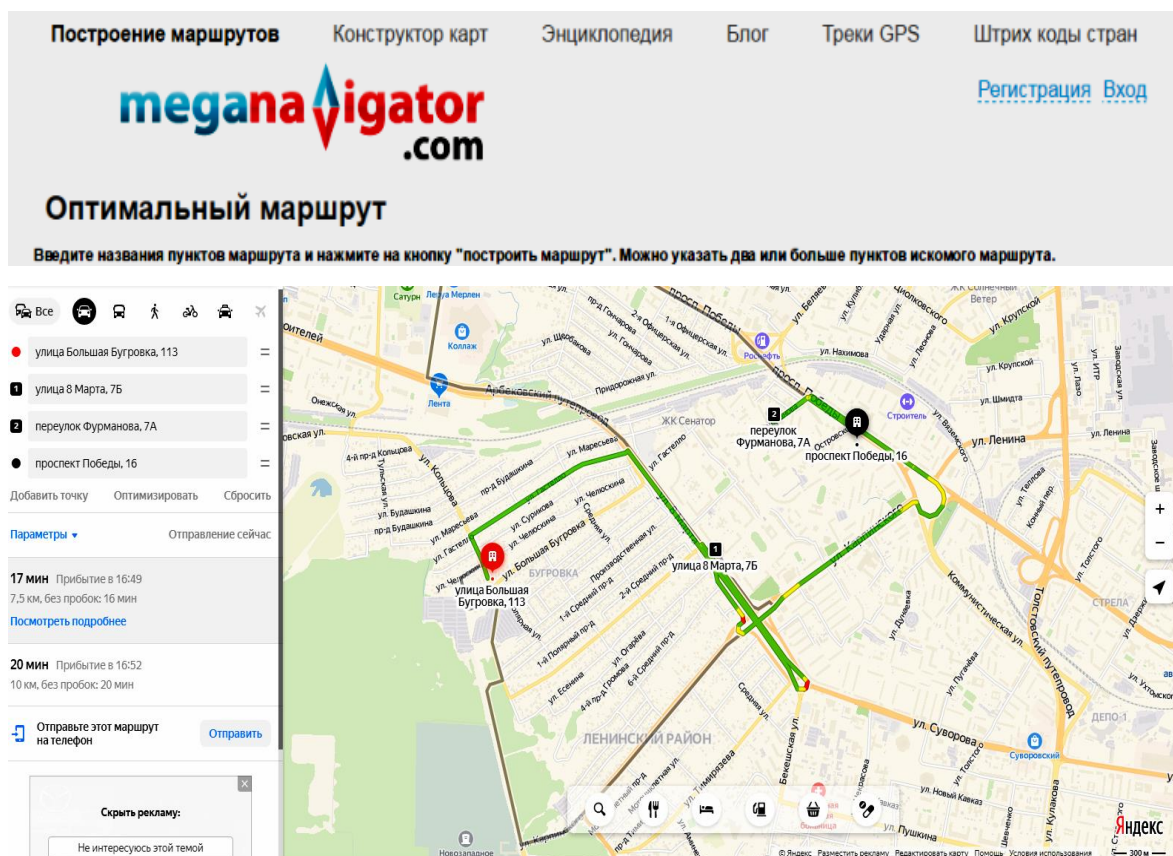


Рисунок 1.3 – Интерфейс приложения Meganavigator

Тестирование функционала показало наличие существенных недостатков:

- 1) несмотря на то, что Meganavigator позиционирует себя как сервис для построения автомобильных, велосипедных и пешеходных маршрутов, на сайте отсутствует выбор средства передвижения; режим, выставленный по умолчанию, соответствует передвижению на автотранспортном средстве;
- 2) у сервиса не предусмотрены подсказки при вводе адреса, что затрудняет работу с ним;

3) при построении маршрута первый пункт выбирается произвольно из заданного списка адресов; нет возможности задавать адрес как первый по умолчанию;

4) при выборе пунктов на интерактивной карте без ввода адресов в текстовые поля, маршрут не строиться.

Среди зарубежных картографических сервисов с функцией построения оптимального маршрута наиболее популярен Speedy Route [25]. В описании продукта сказано, что сервис рассчитывает наиболее эффективный по количеству затрат топлива маршрут между несколькими локациями, где исходная и конечная точка едины (рис. 1.4). В первую очередь можно выделить такие недостатки как:

- 1) Speedy Route рассчитан исключительно для автомобилистов;
- 2) при вводе данных возникают трудности перевода русскоязычных названий на английский язык;
- 3) у сервиса предусмотрено минимальное количество пунктов маршрута – 5; маршруты, состоящие из 4 пунктов построить невозможно;
- 4) на сайте представлена урезанная версия карты для ознакомления, для доступа к полной версии требуется оформить платную подписку.

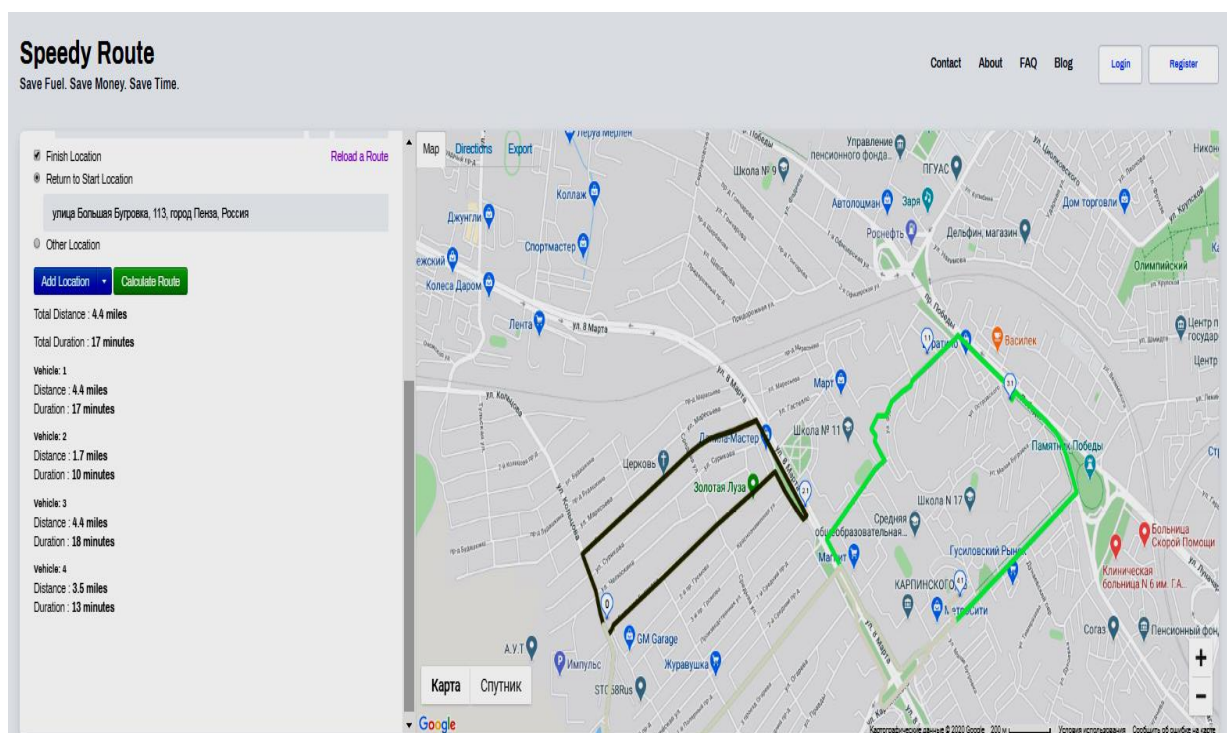


Рисунок 1.4 – Интерфейс приложения Speedy Route

Суммируя все вышеперечисленное можно сказать, что пока не существуют сервисы, которые бы строили оптимальные маршруты для перевозок внутри города и одновременно были удобными с точки зрения и понятными для пользователей при взаимодействии с ними.

1.4 Основные положения теории графов

Коммивояжёр (фр. *commis voyageur*) – бродячий торговец. Задача коммивояжёра – важная задача транспортной логистики, отрасли, занимающейся планированием транспортных перевозок. Коммивояжёру, чтобы распродать нужные и не очень нужные в хозяйстве товары, следует объехать n пунктов и в конце концов вернуться в исходный пункт. Требуется определить наиболее выгодный маршрут объезда. В качестве меры выгодности маршрута (точнее говоря, невыгодности) может служить суммарное время в пути, суммарная стоимость дороги, или, в простейшем случае, длина маршрута.[10]

Математические проблемы, связанные с задачей коммивояжера, были изучены в 1800-ых годах ирландским математиком сэром Уильямом Роуэном Гамильтоном и британским математиком Томасом Пеннингтоном Киркменом.

В 1857 Гамильтон создал игру Икосиан, в правилах которой сказано, что участники должны соединить 20 точек додекаэдра так, чтобы каждая точка использовалась не более одного раза, также конечная точка пути должна совпадать с исходной. Эта игра легла в основу появления гамильтонова графа.

Задача поиска оптимально пути была впервые рассмотрена с математической точки зрения в 1930-ых годах математиком и экономистом Карлом Менджером в Вене. В 1940-ых статистики Мэхаланобис, Джессен, Гош и Маркс пытались найти применение задаче коммивояжера в сельскохозяйственном секторе, что привело к ее популяризации – начиная с

середины 1950ых годах методы решения задачи стали публиковаться в научных журналах.

Хотя задача коммивояжера проста для понимания, ее крайне сложно решить. В 1972 Ричард М. Карп показал, что задача гамильтонова цикла принадлежит к классу NP-полных задач (Nondeterministic Polynomial time), которые подразумевает NP-сложность задачи TSP. [10] Это открытие дало научное объяснение очевидной вычислительной трудности нахождения оптимальных маршрутов.

Методы решения TSP становились более сложными, количество городов возрастало. Ниже представлена краткая история этапов решения задачи коммивояжера.

В 1954 Данциг, Фалкерсон и Джонсон издали описание метода для решения TSP и практически проиллюстрировали его, решив задачу с 49 городами, соединив таким образом города каждого штата Америки. В 1971 году Хельд и Карп решили задачу поиска оптимального пути между 64 городами. Позже в 1977 году, Мартин Гротчел построил путь между 120 немецкими городами. В 1973 Лин и Керниган нашли применение задачи TSP в инженерии – они соединили 318 пунктов, полученных в результате резки лазером. [10]

В 1987 году Голландом и Гроешел была решена задача с 666 городами, позже в этом же году Падберг и Ринальди нашли оптимальный тур, связывающий уже 2392 города. В 1994 году количество городов увеличилось до 7397, спустя 10 лет количество городов достигло до 24978. В 2006 было получено решение задачи коммивояжера с 85900 городами. [10]

Множество всех дорог города или района составляет дорожную сеть. Транспортная сеть — это совокупность дорог региона, пригодных для движения заданных транспортных средств. Транспортная сеть всегда является частным случаем дорожной сети и, как правило, строится для различных типов транспортных средств: легковые автомобили, грузовые полной массой до 3,5 т и т.д.

Модель транспортной сети может быть представлена в виде графа. Граф — это фигура, состоящая из точек (вершин) и соединяющих их отрезков (звеньев)[7]. Пример транспортного графа (на рис 1.5). Проблему коммивояжёра можно представить в виде модели на графе, то есть, используя вершины и ребра между ними. Таким образом, вершины графа соответствуют городам, а рёбра между вершинами и— пути сообщения между этими городами. Каждому ребру можно сопоставить критерий выгодности маршрута, который можно понимать, как, например, расстояние между городами, время или стоимость поездки.

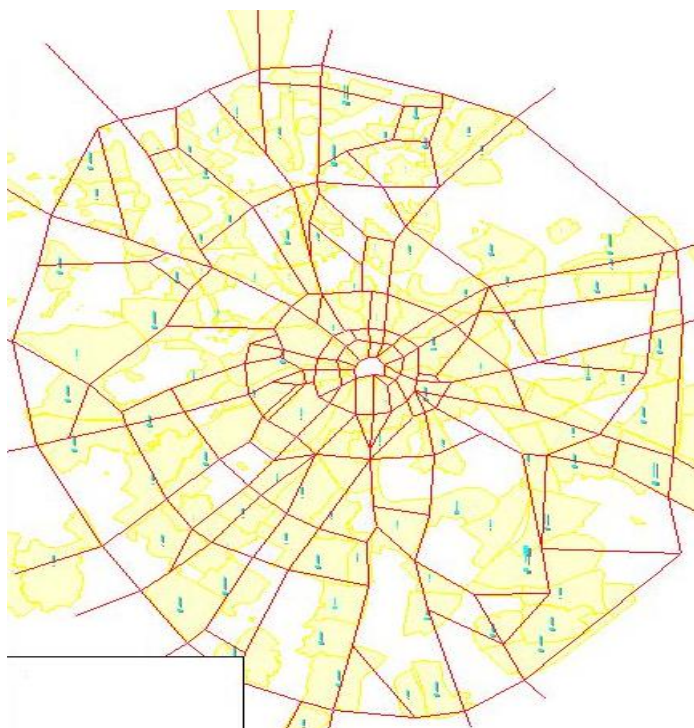


Рисунок 1.5 – Пример транспортного графа

Гамильтоновым циклом называется маршрут, включающий ровно по одному разу каждую вершину графа.[15]

В целях упрощения задачи и гарантии существования маршрута обычно считается, что модельный граф задачи является полностью связным, то есть, что между произвольной парой вершин существует ребро. В тех случаях, когда между отдельными городами не существует сообщения, этого можно достичь

путём ввода рёбер с максимальной длиной. Из-за большой длины такое ребро никогда не попадет к оптимальному маршруту, если он существует.

Таким образом, решение задачи коммивояжёра - это нахождение гамильтонова цикла минимального веса в полном взвешенном графе.

В зависимости от того, какой критерий выгодности маршрута сопоставляется величине ребер, различают различные варианты задачи, важнейшими из которых являются симметричная и метрическая задачи.

Вершины графа — это точки на сети, наиболее важные для определения расстояний или маршрутов движения.

Звенья графа — это отрезки транспортной сети, характеризующие наличие дорожной связи между соседними вершинами. Звенья графа характеризуются числами, которые могут иметь различный физический смысл. Чаще всего это расстояние, но может использоваться, например, и время движения. Ориентированные по направлению звенья графа называются дугами. Фактически всякое неориентированное звено графа включает в себя две равноценные, но противоположно направленные дуги. В зависимости от того, все или часть звеньев имеют направление, граф является ориентированным или смешанным.

Граф, каждая вершина которого может быть соединена некоторой последовательностью звеньев с любой другой его вершиной, называется связанным графом. Иначе говоря, каждая вершина связанного графа должна иметь как минимум одну входящую и одну выходящую дугу. Граф, моделирующий транспортную сеть, обязательно должен быть связанным, чтобы всегда был путь из любой вершины в любую другую вершину. Числа, характеризующие звенья такого графа, обычно выражают протяженность пути, время или стоимость проезда.[11]

Для моделирования транспортной сети необходимо иметь:

1) картографический материал, обычно это карты крупного масштаба, так как они позволяют с большой точностью делать замеры расстояний между пунктами;

2) сведения о размещении основных грузообразующих (ГОП) и грузопоглощающих организаций (ГПП);

3) дополнительные сведения из коммунальных и дорожных организаций в виде перечня улиц с характеристикой их проезжей части;

4) сведения по организации уличного движения, т. е. схемы организации движения на перекрестках, площадях и транспортных развязках, а также сведения о различных ограничениях движения, связанные с установленными дорожными знаками.

Имея эти данные, моделирование транспортной сети начинают с размещения вершин графа. За вершины графа принимают ГОП, ГПП, центры крупных жилых кварталов или небольших обособленных жилых пунктов и пересечения улиц. Каждой вершине присваивается порядковый номер или другое условное обозначение. После размещения вершин их связывают дугами или звеньями.

При построении модели транспортной сети особое внимание следует уделить максимально возможному уменьшению числа вершин. В противном случае транспортная сеть будет излишне сложна и определение кратчайших расстояний потребует длительного времени. Для снижения размерности и ускорения расчетов для транспортных сетей больших городов используется микро- и микрорайонирование.

Микрорайонирование транспортной сети заключается в использовании в качестве вершин не пересечений дорожной сети (перекрестков), а центров микрорайонов.

Макрорайонирование (агрегирование) транспортной сети заключается в разбиении ее на отдельные подсети, расчеты по которым могут выполняться отдельно, а затем объединяться для получения общего

результата. Этот способ особенно эффективен при пересчете расстояний из-за изменения дорожной обстановки, поскольку требуется пересчет только той подсети, в которой изменились транспортные связи.

Граф, как и большинство других математических объектов, может быть представлен на компьютере (сохранен в его памяти). Существуют несколько способов его интерпретации, вот наиболее известные из них[23]:

1. Матрица смежности;
2. Матрица инцидентности;
3. Список смежности;
4. Список ребер.

Матрица смежности графа — это квадратная матрица, в которой каждый элемент принимает одно из двух значений: 0 или 1. Число строк матрицы смежности равно числу столбцов и соответствует количеству вершин графа. На рис. 1.6 показана матрица смежности.

А) 0 – соответствует отсутствию ребра,

Б) 1 – соответствует наличию ребра.

	1	2	3	4
1	0	1	0	1
2	0	0	1	1
3	0	1	0	0
4	1	0	1	0

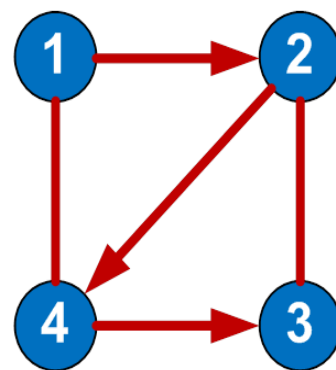


Рисунок 1.6 – Матрица смежности

Когда из одной вершины в другую проход свободен (имеется ребро), в ячейку заносится 1, иначе – 0. Все элементы на главной диагонали равны 0 если граф не имеет петель.

Матрица инцидентности (инциденции) графа — это матрица, количество строк в которой соответствует числу вершин, а количество столбцов – числу

рёбер. В ней указываются связи между инцидентными элементами графа (ребро(дуга) и вершина).

В неориентированном графе если вершина инцидентна ребру то соответствующий элемент равен 1, в противном случае элемент равен 0.

В ориентированном графе если ребро выходит из вершины, то соответствующий элемент равен 1, если ребро входит в вершину, то соответствующий элемент равен -1, если ребро отсутствует, то элемент равен 0.

Матрица инцидентности для своего представления требует нумерации рёбер, что не всегда удобно. На рис. 1.7 показана матрица инцидентности.

	1	2	3	4	5
1	1	0	0	1	0
2	-1	1	0	0	1
3	0	1	-1	0	0
4	0	0	1	1	-1

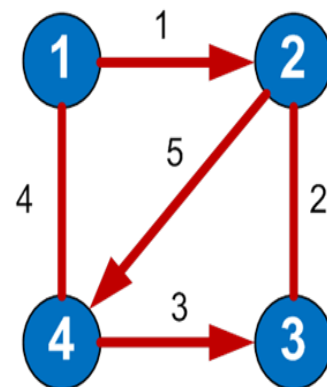


Рисунок 1.7 – Матрица инцидентности.

Список смежности (инцидентности)

Если количество ребер графа по сравнению с количеством вершин невелико, то значения большинства элементов матрицы смежности будут равны 0. При этом использование данного метода нецелесообразно. Для подобных графов имеются более оптимальные способы их представления. По отношению к памяти списки смежности менее требовательны, чем матрицы смежности. Такой список можно представить в виде таблицы, столбцов в которой – 2, а строк — не больше, чем вершин в графе. В каждой строке в первом столбце указана вершина выхода, а во втором столбце – список вершин, в которые входят ребра из текущей вершины. На рис. 1.8 показана матрица смежности

1	2, 4
2	3, 4
3	2
4	1, 3

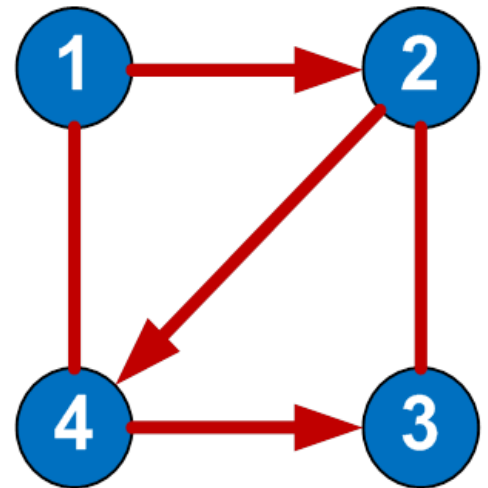


Рисунок 1.8 – Список смежности

Преимущества списка смежности:

1. Рациональное использование памяти.
2. Позволяет быстро перебирать соседей вершины.
3. Позволяет проверять наличие ребра и удалять его.

Недостатки списка смежности:

1. При работе с насыщенными графами (с большим количеством рёбер) скорости может не хватать.

2. Нет быстрого способа проверить, существует ли ребро между двумя вершинами.

Количество вершин графа должно быть известно заранее.

1) для взвешенных графов приходится хранить список, элементы которого должны содержать два значащих поля, что усложняет код:

- 2) номер вершины, с которой соединяется текущая;
- 3) вес ребра.

Список рёбер

В списке рёбер в каждой строке записываются две смежные вершины и вес соединяющего их ребра (для взвешенного графа). Количество строк в списке ребер всегда должно быть равно величине, получающейся в результате

сложения ориентированных рёбер с удвоенным количеством неориентированных рёбер. На рис. 1.9 показан список рёбер.

	Начало	Конец	Вес
1	1	2	
2	1	4	
3	2	3	
4	2	4	
5	3	2	
6	4	1	
7	4	3	

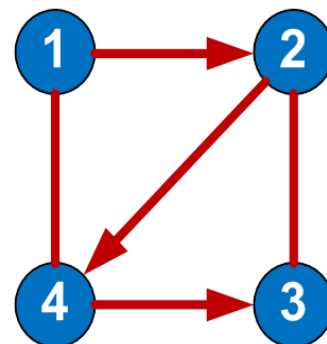


Рисунок 1.9 – Список рёбер

Какой способ представления графа лучше? Ответ зависит от отношения между числом вершин и числом рёбер. Число рёбер может быть довольно малым (такого же порядка, как и количество вершин) или довольно большим (если граф является полным). Графы с большим числом рёбер называют плотными, с малым — разреженными. Плотные графы удобнее хранить в виде матрицы смежности, разреженные — в виде списка смежности[7].

Таким образом, в данной работе УДС представляет собой ориентированный граф, в котором координаты магазинов являются вершинами графа, а расстояния между ними — рёбрами. Для формирования матрицы смежности рассматриваемой сети супермаркетов «Магнит» необходимо воспользоваться таблицей 1.1, в которой указаны координаты магазинов Ленинского района г. Пензы. Расстояние между двумя точками сети определяется по формуле 1.1:

$$AB = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \quad (1.1)$$

Определение расстояние между двумя точками показано на рис. 1.10

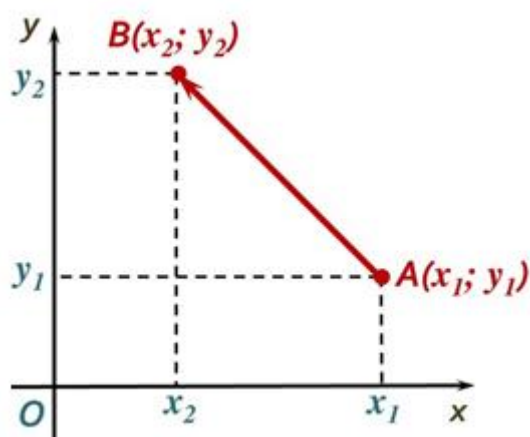


Рисунок 1.10 – Расстояние между двумя точками

Следует отметить, что в данной задаче принимается допущение, что возможен прямой проезд между магазинами. Используя формулу (1.1) была определена матрица расстояний, необходимая для вычисления расстояния между магазинами (таблица 1.2) [16]

Таблица 1.2 – Матрица смежности супермаркетов «Магнит» Ленинского района г. Пензы.

Номера магазинов	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0,0	2,0	2,7	3,4	2,9	3,8	4,0	3,0	4,0	5,0	4,4	4,8	6,0	5,7	7,0	8,0
2	2,0	0,0	0,8	1,4	0,9	1,8	2,0	1,2	2,0	3,1	2,6	2,9	4,1	3,9	5,0	6,1
3	2,7	0,8	0,0	0,8	0,8	1,3	1,6	1,5	1,9	2,8	2,6	2,8	3,9	3,9	4,7	5,8
4	3,4	1,4	0,8	0,0	0,7	0,6	0,9	1,4	1,4	2,1	2,1	2,2	3,2	3,2	4,0	5,1
5	2,9	0,9	0,8	0,7	0,0	0,9	1,1	0,7	1,2	2,2	1,8	2,1	3,3	3,1	4,1	5,2
6	3,8	1,8	1,3	0,6	0,9	0,0	0,4	1,3	0,9	1,5	1,6	1,7	2,7	2,7	3,4	4,5
7	4,0	2,0	1,6	0,9	1,1	0,4	0,0	1,3	0,6	1,2	1,3	1,3	2,3	2,3	3,1	4,2
8	3,0	1,2	1,5	1,4	0,7	1,3	1,3	0,0	1,0	2,1	1,4	1,8	3,0	2,7	4,0	5,0
9	4,0	2,0	1,9	1,4	1,2	0,9	0,6	1,0	0,0	1,1	0,7	0,9	2,1	1,9	3,0	4,1
10	5,0	3,1	2,8	2,1	2,2	1,5	1,2	2,1	1,1	0,0	1,0	0,6	1,1	1,2	1,9	3,0
11	4,4	2,6	2,6	2,1	1,8	1,6	1,3	1,4	0,7	1,0	0,0	0,4	1,6	1,3	2,6	3,6
12	4,8	2,9	2,8	2,2	2,1	1,7	1,3	1,8	0,9	0,6	0,4	0,0	1,2	1,0	2,2	3,2
13	6,0	4,1	3,9	3,2	3,3	2,7	2,3	3,0	2,1	1,1	1,6	1,2	0,0	0,6	1,1	2,1
14	5,7	3,9	3,9	3,2	3,1	2,7	2,3	2,7	1,9	1,2	1,3	1,0	0,6	0,0	1,5	2,4
15	7,0	5,0	4,7	4,0	4,1	3,4	3,1	4,0	3,0	1,9	2,6	2,2	1,1	1,5	0,0	1,1
16	7,5	5,7	5,6	4,9	4,9	4,3	4,0	4,5	3,7	2,8	3,1	2,8	1,7	1,8	1,3	1,2

1.5 Выводы по первой главе

В данной главе был представлен обзор предметной области – рассмотрены теоретические основы транспортной логистики, представлены описание и краткая история формирования задачи поиска оптимального маршрута, анализ УДС Ленинского района г. Пензы, в котором будет произведена оптимизация маршрута перевозок супермаркета «Магнит». По результатам обзора поставлены задачи и цель выпускной квалификационной работы.

2 МАТЕМАТИЧЕСКАЯ ОСНОВА ПРОЕКТА

2.1 Анализ и выбор алгоритма поиска оптимального пути

В предыдущей главе было рассмотрено, что улично-дорожную сеть Ленинского района г. Пензы с расположенными на ней необходимыми супермаркетами можно представить в виде графа, а путь решения поставленной задачи – решение задачи коммивояжёра.

Целью данной задачи является поиск кратчайшего маршрута между супермаркетами «Магнит» Ленинского района г. Пензы, откуда происходит перевозка товаров. В качестве начального может быть выбран любой супермаркет. Результатом работы будет являться посещение всех заданных магазинов и возвращение в стартовый. Главное условие – посещение каждого супермаркета «Магнит» должно осуществляться один раз, что не касается стартового магазина, который посещается перевозчиком в начале и в конце пути (рис. 2.1).

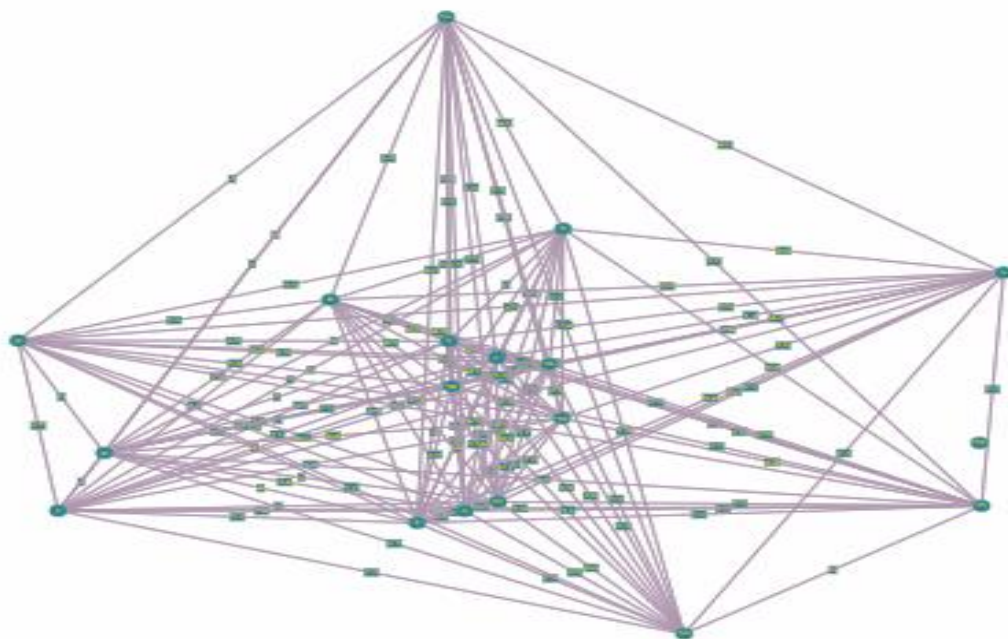


Рисунок 2.1 – Граф супермаркетов «Магнит» Ленинского района г. Пензы

Когда лучший маршрут найден, стартовым может быть любой его магазин.

Есть и иные трактовки задачи, например, найти маршрут с минимальной стоимостью.

Тривиальное решение – это перебор всех возможных вариантов, но этот метод не подходит при небольшом числе посещаемых городов из-за неприемлемых вычислительных затрат.

Самое простое решение – употребление *жадного алгоритма*, главной идеей которого является выбор супермаркета, который находится ближе всех к супермаркету нахождения перевозчика. В результате полученный исход далёк от оптимального. Его главная идея заключается в переборе всех возможных вариантов путей, алгоритм решения записывают так:

- 1) найти общее число возможных гамильтоновых контуров;
- 2) найти вес каждого гамильтонова контура - контур в ориентированном графе, где коммивояжёр проходит через все вершины графа, притом только по одному разу, сложив вес всех его ребер;
- 3) выбрать гамильтонов контур с минимальным весом, который и будет оптимальным.

Метод полного перебора обладает рядом преимуществ: во-первых, он гарантирует поиск решения задачи коммивояжёра, при этом он прямолинеен и прост в исполнении; во –вторых алгоритм считается нецелесообразным при работе с большим объемом данных.

В реализации перебора возможно применение 2 условий, сокращающих время и отсекающих неоптимальные пути:

1. Если длина нового пути превышает найденный локальный минимум – пропускать этот вариант и все, которые он порождает.
2. Если выбранное ребро пересекает одно из ранее построенных ребер – пропустить этот вариант и те, которые он порождает, т.е. это запрет циклов.

Табл. 2.1 демонстрирует количество времени, которое требуется для решения задачи коммивояжера методом полного перебора при компьютерной

мощности, позволяющей считать 1 миллион гамильтоновых контуров в секунду.

Таблица 2.1 – Расчётное время решения задачи коммивояжёра методом полного перебора [12]

Расчётное время решения задачи коммивояжёра методом полного перебора	
Количество городов	Расчётное время
10	1/3 с
13	8 минут
15	1 год
20	193 года

Во время поиска точного решения, перебирая все возможные варианты, особое значение имеет астрономический порядок. Именно поэтому для решения этой задачи возможен подход к практическому решению, используя при этом эвристический алгоритм. Поэтому, как известно, привычные обоснования применимости метода отсутствуют. "Правдоподобные рассуждения, наша интуиция, опыт и машинный эксперимент - вот пока те оправдания эвристики, которыми мы располагаем". Изучение эвристического алгоритма в работе позволяет найти локальный минимум в ЗК с помощью моделирования на компьютере законов генетики и естественного отбора Ч. Дарвина. Впервые идея этого алгоритма была предложена Дж. Холландом и названа им "генетический алгоритм (ГА)". Генетический алгоритм является алгоритмом прямого поиска или методом оптимизации на основе сопоставления значений функций в пробных точках. С помощью генетического алгоритма в настоящее время решается ряд практических задач.[12]

2.2 Понятия генетического алгоритма

Ген или сновная концепция классической генетики была введена Менделем с целью объяснения наблюдаемой статистики наследования.

Носители генов в клеточном ядре - нитевидные тела, которые называются хромосомами. Локус - место, занимаемое геном в хромосоме. Схематически хромосома представляется как прямолинейный отрезок, а локусы – как последовательные участки, на которые этот отрезок разбит. Действия генов проявляются в больших сообществах организмов, которые скрещиваются между собой. Эти сообщества называются биологами популяциями. Популяция существует в пространстве и во времени. репродукция путем скрещивания – это основа существования. Считается, что вся популяция состоит в пространстве и времени из дискретных поколений $F_0, F_1, F_2, \dots$. Поколение F_{i+1} – это совокупность особей, родители которых принадлежат поколению F_i , поколение F_0 фиксируется априори. Нами рассматривались такие популяции, у которых для любого состояния z в поколении F однозначно определено состояние, которое возникает в следующем поколении при помощи скрещивания и отбора. В данной работе понятия "особь, организм" мы идентифицируем с понятием "хромосома". Отображение V , определяемое уравнением, называют эволюционным оператором. Учитывая биологический смысл эволюционного оператора и непрерывность отображения V , обычно требуется обеспечение устойчивости при замене вероятностей близкими к ним частотами. В данном случае популяция считается динамической системой на компактном подмножестве называются конечномерного пространства. Равновесными состояниями популяции неподвижные точки эволюционного оператора V называют. Они существуют в силу известной топологической теоремы Брауэра. Генетический алгоритм позволяет находить эти состояния популяции: "Работу алгоритма прекращаем при достижении популяцией состояния адаптации, идентифицируемому по стягиванию ядра популяции сначала в плотное облачко, а затем в точку. Кроссинговер как механизм изменчивости теряет в таких условиях свою силу – при скрещивании идентичных родителей потомок ничем не будет отличаться ни от одного из них.". По сути дела, этот эвристический алгоритм опирается на используемую в динамике отбора

восходящую к R. A. Fisher фундаментальную теорему о возрастании средней приспособленности популяции в ходе естественного отбора. Определяя функционал для каждой популяции качества, неподвижные точки эволюционного оператора должны соответствовать точкам его локального экстремума. Однако, нельзя с уверенностью утверждать, что новое решение дает глобальный экстремум.

Основные понятия генетического алгоритма в случае ЗК имеют такой смысл: номер вершины графа городов обозначает номер города его называют геном, проходя через который строится маршрут коммивояжера; последовательность генов называется хромосомой, которая определяет замкнутый маршрут, проходящий через все заданные вершины графа, которые обозначают все заданные города; популяция – набор из нескольких возможных маршрутов. Состояние популяции - множество возможных маршрутов, которые удовлетворяют условия задачи. Функционал качества – длина построенного маршрута.

Способ задания первой популяции существенно влияет на эффективность метода, точнее систематическое перечисление вершин графа. Нами реализован «поиск в ширину», где вершины обходятся в следующем порядке: во-первых, выбирается случайный город в качестве точки старта; во-вторых, он рассматривается как текущий; в - третьих, сравниваются длины путей, проходящих из текущего города в ещё не посещенные города, а из них уже выбирается кратчайший путь. Полученный таким образом следующий город становится текущим. Данные действия повторяются до тех пор, пока все города не будут перебраны. При выборе различных точек старта можно получить до n различных маршрутов, то есть хромосом. Оставшиеся хромосомы, до заданного числа M , заполняем случайными маршрутами.

В теории ГА значимым является выбор кодировки чисел, то есть номеров городов. Для задачи коммивояжера нами была выбрана десятичная кодировка. Эффективность применения ГА к ЗК зависит от настройки операторов мутации

и кроссинговера. Мутация, точечное, незначительное, случайное изменение в хромосоме осуществляется несколькими способами. Самым распространенным является выбор двух случайных генов, а затем обмен их местами. Мы использовали другую схему: если в хромосоме случайным образом выбирается ген с номером i , то он меняется местом с геном с номером $n + 1 - i$. Целью оператора мутации является расширение пространства, поиск точечных изменений в хромосоме за счет случайных мутаций. В классическом варианте действие кроссинговера следующее: выбираются две определенные хромосомы, это должны быть случайные хромосомы, далее также случайным образом определяется точка кроссинговера, происходит обмен участками хромосом [26,28].

В работе рассматривается возможное употребление генетического алгоритма для поиска благоприятного решения.

Суть подхода состоит в том, что генерируется множество маршрутов. Данное множество называется популяцией, затем к маршрутам множества применяются операции скрещивания и мутации.

В результате обмена участками маршрута дочерний маршрут формируется при скрещивании двух выбранных маршрутов.

Внутри ранее созданного дочернего маршрута при мутации меняются местами участки.

Маршрут представлен в виде списка `tour` (рис. 2.2) номеров магазинов размера $\text{cnt} + 1$, где cnt – это число посещаемых коммивояжером магазинов. При этом первый магазин маршрута из списка добавляется и в начало, и в конец списка, что позволяет представить маршрут в виде цепочки магазинов (рис. 2.2).



Рисунок 2.2 - Представление маршрута в виде цепочки магазинов.

Первый и последний города совпадают: $\text{tour}[0] = \text{tour}[\text{cnt}]$

Лучший маршрут – это самая короткая цепочка.

Генетический алгоритм универсален, это связано с тем, что функция приспособленности и кодирование решений зависят от условий поставленной задачи. Выполняя алгоритм, учитываются следующие правила: начальная популяция, которая выбирается случайно, количество особей должно остаться неизменным, причём каждая из них записывается строкой с определенной длиной кодировки [16].

Каждый шаг алгоритма можно разбить на три этапа:

1) пропорциональный в зависимости от приспособленности отбор особей текущего поколения, который имеет право производить потомство, и формирование из них промежуточной популяции;

2) промежуточная популяция делится на пару и с какой-то вероятностью скрещивают, в результате получается новое поколение, где сама пара или ее потомки при их присутствии. Потомки формируются из отсеченных частей родительских строк, которые разделены некой точкой.

3) происходит мутация полученного поколения, не допускающая преждевременной сходимости. Каждый бит особи с вероятностью не более 1% записывают как противоположный исходному.

Схема работы генетического алгоритма показана на рисунке 2.3 [6]

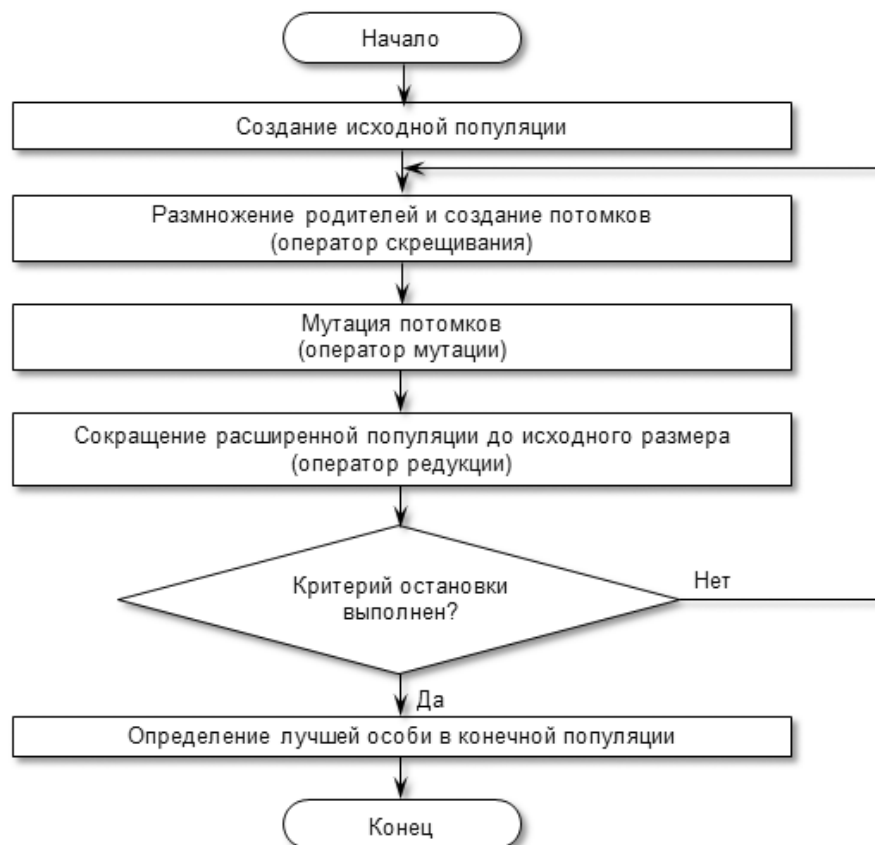


Рисунок 2.3 - Схема работы генетического алгоритма

Критерий остановки работы генетического алгоритма характеризуется следующими событиями:

- формированием заданного числа поколений;
- исчерпывание времени, отведенного на эволюцию;
- популяцией достижения заданного качества (значение критерия одной (нескольких, всех) особей превысило заданный порог);
- достижение некоторого уровня сходимости (особи популяции становятся настолько подобными, что дальнейшее их совершенствование происходит чрезвычайно медленно);
- и т.д.

Используя генетический алгоритм для решения задачи поиска оптимального пути, выполняются все ранее перечисленные шаги, где приспособленность особи фактически служит мерой длины маршрута [5].

2.3 Реализация поиска

Модель популяции. Популяцией называется список маршрутов. Маршрут в списке представляется числом из диапазона $[0, \text{population Size} - 1]$, где population Size – это размер популяции.

Размер списка стабилен, однако это и необязательно. Главным отличием является то, что в популяцию можно добавить дочерний маршрут, но не заменять его, как это делают в программе, он называется нецелесообразным, то есть слишком длинным маршрутом популяции. Существует процедура чистки популяции, она удаляет из популяции через определённые промежутки времени плохие маршруты и так далее.

Начиная создание начальной популяции, для каждого магазина определяется список close Cities магазинов-соседей, то есть супермаркетов, они ближе всех расположены к текущему магазину. Число магазинов-соседей у каждого супермаркета одинаково и равно $\text{number Of Close Cities}$.

Элемент списка – это число из диапазона $[0, \text{cnt} - 1]$, где cnt – число магазинов в маршруте. Супермаркеты упорядочены по списку в порядке возрастания расстояний магазинов до текущего [2].

Начальная популяция формируется следующим образом:

1. Получить случайное число из диапазона $[0, \text{cnt} - 1]$ и комментировать его как номер текущего магазина (cnt – число магазинов в маршруте).

2. Получить случайное число из диапазона $[0, 99]$, если оно меньше $\text{chance Use Close City}$, то получится следующий город маршрута по принципу жадного алгоритма, то есть взять его из списка close Cities магазинов-соседей, причём берется первый свободный город списка. Свободные, то есть в списке включённых маршрутов супермаркета соседей нет, то случайным образом выбирается свободный магазин из всего списка магазинов.

3. Если случайное число более или равно chance Use Close City, то в маршрут расширяется, в него добавляется свободный магазин, который случайно выбирается из общего списка магазинов.

4. Передвинуть, сохраняя последовательность, в каждом маршруте супермаркета так, чтобы ранее заданный магазин оказался на первом месте city Number (в программе city Number = 0).

5. Добавить в конец каждого маршрута его первый магазин (city Number).

По приведенной схеме сформирована модель популяции, которая позволяет довольно просто реализовывать операции скрещивания и мутации [3].

На рис. 2.4 представлен результат работы программы для 17 магазинов.

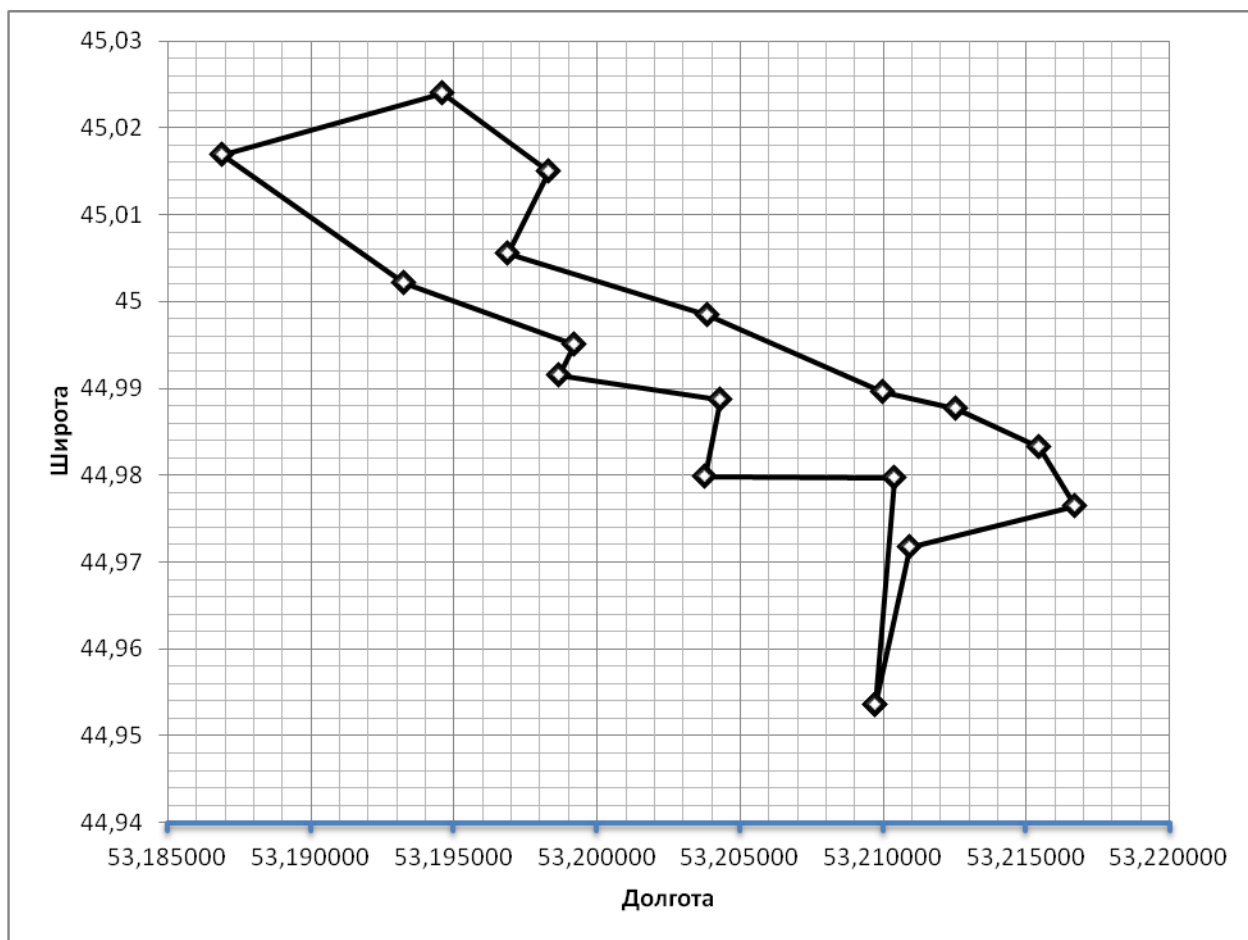


Рисунок 2.4 – Оптимальный маршрут следования между магазинами «Магнит» Ленинского района г. Пензы

На рисунке 2.5 представлен результат последовательности перевозок.

[illegible]

Рисунок 2.5 – Последовательность перевозок

2.4 Выводы по второй главе

В работе проведен расчет решения задачи коммивояжера, полученный методом случайного поиска. Суть заключалась в отыскании самого короткого маршрута, проходящего через заданные магазины по одному разу с последующим возвратом в исходный магазин. Длина всего маршрута составила 18,7 км. Глядя на то, как широко «разбросаны» магазины по Ленинскому району, можно сказать, что оптимизация маршрута прошла успешно, ведь при посещении 17 магазинов коммивояжер проезжал всего 18.7 км.

3 Экономический раздел

3.1 Экономия топлива и факторы влияющие на нее

Существует множество рычагов воздействия на затраты, и при рациональном их использовании возможно значительное снижение себестоимости автоперевозок, что означает повышение экономической составляющей и оптимизации автотранспортного предприятия.

Экономия топлива является очень важным фактором, влияющим на эффективность работы предприятия. Важно отметить, что на расход топлива существенно влияет режим движения автомобиля, а также выбор оптимальных скоростей. Важный резерв экономии топлива оптимизация режимов движения и скоростей в различных условиях эксплуатации с помощью специальных приборов и электро-вычислительных машин [1].

Норма расхода топлива осуществляется несколькими способами. Самым распространенным является применение норм расхода топливно-смазочных материалов на автомобильном транспорте, которые утверждаются Распоряжением Минтранса РФ от 14.03.2008 г. № АМ-23-р «О введении в действие методических рекомендаций «Нормы расхода топлив и смазочных материалов на автомобильном транспорте» [14]. Нормы расхода топлива и смазочных материалов на автомобильном транспорте предназначаются для расчетов нормативного значения расхода топлива по месту потребления, для ведения статистической и оперативной отчетности, определения себестоимости перевозок и других видов транспортных работ, планирования потребности предприятий в обеспечении нефтепродуктами, для расчетов по налогообложению предприятий, осуществления режима экономии и энергосбережения потребляемых нефтепродуктов, проведения расчетов с пользователями транспортными средствами, водителями и так далее.

На расход топлива конкретного автомобиля влияет техническая исправность транспортного средства, срок его эксплуатации, состояние движущихся частей, габаритов кузова, тяжесть перевозимого груза и другое. Ко

всему перечисленному можно добавить, что на расход топлива влияют погодные условия и профессионализм водителя. Постоянный контроль норм расхода топлива, позволяет экономить ресурсы и деньги предприятия, поэтому необходимо рассчитывать показатели нормы и делать их анализ [21].

3.2 Расчет нормы расхода топлива

Потребление топлива транспортным средством считается исходя из нескольких показателей: марка транспортного средства, его модель и условия эксплуатации. Базовая норма – показатель, который измеряется в литрах и считается на 100 километров пути (пробега). Для подсчета нужна информация относительно расхода топлива и пробега автомобиля по маршруту[19].

Сеть супермаркетов Магнит для перевозки груза использует седельные тягачи семейства MAN TGA (табл. 3.1)[29].

Таблица 3.1 – Характеристики моделей машин MAN TGA

Модель тягача	Грузоподъемность, тонн	Расход топлива на 100 км	Год выпуска
26350	15	33	2008
19350	19	35	2008
18360	10	29	2008

Также рассмотрим варианты доставки груза, различными видами грузовых автомобилей, для оценки экономической эффективности перевозки (табл. 3.2).

Возьмем следующие автомобили

Таблица 3.2 – Характеристика грузовых моделей машин

Марка автомобиля	Грузоподъемность, тонн	Расход топлива на 100 км	Год выпуска
Газель Next	2,6	20	2008

Isuzu giga	8	19	2008
------------	---	----	------

Маршрут оптимизации в данной работе состоит из 17 супермаркетов, километраж, которого составляет 18,7 км в городе с населением больше 500 тысяч человек. Исходя из этих данных, мы можем рассчитать норму расхода топлива для данного маршрута по формуле (3.1):

$$Q_H = 0,01 \cdot H_{san} \cdot S + H_w \cdot W \cdot (1 + 0,01 \cdot D), \quad (3.1)$$

где Q_H – нормативный расход топлива грузовика, л;

S – пробег, км;

H_{san} – норма расхода топлив на пробег грузовика в снаряженном состоянии без груза;

$$H_{san} = H_s + H_g \cdot G_{пр} \quad (3.2)$$

где H_s – базовая норма расхода топлив на пробег грузовика в снаряженном состоянии, л/100 км ($H_{san} = H_s$, л/100 км, для одиночного автомобиля;

H_g – норма расхода топлива, л/100 т.км;

$G_{пр}$ – собственная масса прицепа или полуприцепа, т;

H_w – норма расхода топлив на транспортную работу, л/100 т.км ;

W – объем транспортной работы, т.км:

$$W = G_{гр} \times S_{гр}$$

здесь $G_{гр}$ – масса груза, т;

$S_{гр}$ – пробег с грузом, км);

D – поправочный коэффициент (суммарная относительная надбавка или снижение) к норме, %.

Коэффициент D подбирается по следующим правилам:

- 1) Кондиционер или система климат-контроля – плюс 8%;
- 2) Эксплуатация в городской черте (от 250 тыс. жителей) – плюс 15%;
- 3) После 100 тыс. км пробега и 5 лет эксплуатации – плюс 5%;

При расчете будем считать, что поправочный коэффициент D будет равен 28% т.к. все автомобили имеют все заданные условия поправочного коэффициента.

Норма расхода топлива на перевозку полезного груза составляет $H_w = 1,3$ л/100 т·км [24].

1. Норма расхода топлива на пробег седельного тягача Man TGA 26350 (рис. 3.1) без груза составляет $H_{san} = 33$ л/100 км.

Нормативный расход топлива составляет:

$$Q_H = 0,01 (33 \cdot 18,7 + 1,3 \cdot 15 \cdot 18,7) \cdot (1 + 0,01 \cdot 28) = 12,6 \text{ л}$$



Рисунок 3.1 – автомобиль Man TGA 26350

2. Норма расхода топлива на пробег седельного тягача Man TGA 19350 (рис. 3.2) без груза составляет $H_{san} = 35$ л/100 км

Нормативный расход топлива составляет:

$$Q_H = 0,01 \cdot (35 \cdot 18,7 + 1,3 \cdot 19 \cdot 18,7) \cdot (1 + 0,01 \cdot 28) = 14,3 \text{ л}$$



Рисунок 3.2 – Автомобиль Man TGA 19350

3. Норма расхода топлива на пробег седельного тягача Man TGA 18360 (рис. 3.3) без груза составляет $H_{\text{san}} = 35$ л/100.

Нормативный расход топлива составляет:

$$Q_{\text{н}} = 0,01 \cdot (29 \cdot 18,7 + 1,3 \cdot 10 \cdot 18,7) \cdot (1 + 0,01 \cdot 28) = 10 \text{ л}$$



Рисунок 3.3 – Автомобиль Man TGA 18360

4. Норма расхода топлива на пробег Газель Next (рис. 3.4) без груза составляет $H_{\text{сан}} = 20$ л/100



Рисунок 3.4 – Автомобиль Газель Next

Нормативный расход топлива составляет:

$$Q_{\text{н}} = 0,01 \cdot (20 \cdot 18,7 + 1,3 \cdot 2,6 \cdot 18,7) \cdot (1 + 0,01 \cdot 28) = 5,59 \text{ л}$$

5. норма расхода топлива на пробег седельного тягача Isuzu giga (рис.3.5) без груза составляет: $H_{\text{сан}} = 19$ л/100



Рисунок 3.5 – Автомобиль Isuzu giga

Нормативный расход топлива составляет:

$$Q_{н\equiv} = 0,01 \cdot (19 \cdot 18,7 + 1,3 \cdot 8 \cdot 18,7) \cdot (1 + 0,01 \cdot 28) = 7 \text{ л}$$

Результаты сводим в таблицу 3.3

Таблица 3.3 – Результаты расчета расхода нормы топлива

Марка машины	Расход топлива , л/100км	Расход топлива , л/ по маршруту 18.7 км
MAN TGA 26350	33	12.6
MAN TGA 19350	35	14,3
MAN TGA 18360	28	10
Газель Next	20	5.59
Isuzu giga	19	7

Для непосредственного расчета экономической эффективности нам потребуется средняя стоимость одного литра дизельного топлива, которая равна 48 рублей на АЗС «Лукойл» по состоянию на 2020 год.[32] Для равных условий установим, что каждому автомобилю будет необходимо перевести 8 тонн груза, это средний вес груза перевозимый по России по данным сайта[32] Стоимость поездки будет определяться по формуле 3.3

$$c = \text{расход топлива груженного автомобиля} \cdot \text{цену 1 литра топлива} \quad (3.3)$$

Стоимость поездки по маршруту для каждой машины составит:

$$\text{MAN TGA 26350} = 12.6 \cdot 48 = 604,8 \text{ рубля}$$

$$\text{MAN TGA 19350} = 14.3 \cdot 48 = 686,4 \text{ рубля}$$

$$\text{MAN TGA 18360} = 10 \cdot 48 = 480 \text{ рублей}$$

$$\text{Газель Next} = 5.59 \cdot 48 = 268,32 \text{ рубля}$$

$$\text{Isuzu giga} = 7 \cdot 48 = 336 \text{ рублей}$$

В связи с тем, что в каждый магазин требуется доставить разное количество тонн груза, соответственно для оптимизации перевозок и

целесообразного использования транспорта необходимо перевозить груз на различных грузовых автомобилях, а именно:

- От 5 до 10 тонн по маршруту поедут модели Isuzu giga,
- От 10 тонн и выше Man Tga 26350, 18360, 19350
- От 1 до 5 тонн Газель Next.

3.3 Выводы по третьей главе

1. В результате проведенных расчетов была определена средняя величина затрат на топливо для транспортировки груза по маршруту в 18.7 км. В среднем она составляет от 270 до 700 рублей в зависимости от грузоподъемности транспортного средства.

2. На основании проведенных расчетов было установлено, что гораздо выгоднее использовать все выше перечисленные автомобили. Малотоннажному транспорту можно отдать транспортировку груза между 3-6 магазинами, а развоз груза между всеми магазинами достанется средне- и крупнотоннажным грузовикам. Это позволит сократить расходы на топливо для транспортировки груза

Заключение

1. В ходе анализа существующих сервисов для создания маршрута и их недостатков было выявлено, что существует необходимость разработки универсального алгоритма оптимизации движения по маршруту.

2. Разработан алгоритм оптимизации движения по маршруту на основе задачи коммивояжера в программе Excel на языке программирования VBA, позволяющий использовать универсальное количество пунктов для оптимизации движения.

3. Произведен расчет норм расхода топлива для различных типов машин по оптимизированному маршруту. В результате расчета выявлены необходимые типы машин для доставки груза с точки зрения рационального использования грузоподъемности и стоимости перевозки.

Список использованных источников

1. Автомобильный транспорт [Электронный ресурс] – Режим доступа: http://www.pppa.ru/additional/09autoroads/a_roads_529.php
2. Ашарина, И.В. Основы программирования на языках С и С++: Курс лекций для высших учебных заведений / И.В. Ашарина. – М.: Гор. линия-Телеком, 2012. – 208 с.
3. Белоусова, С.Н. Основные принципы и концепции программирования на языке VBA в Excel: Учебное пособие / С.Н. Белоусова, И.А. Бессонова. — М.: БИНОМ. ЛЗ, 2010. – 200 с
4. Гаджинский А.М. Логистика: учебник для высших учебных заведений по направлению подготовки «Экономика» / А. М. Гаджинский. – Москва: Дашков и К°, 2013. – 420 с.
5. Генетический алгоритм [Электронный ресурс] : Режим доступа: https://ru.wikipedia.org/wiki/Генетический_алгоритм
6. Гладков Л.А., Курейчик В.В., Курейчик В.М. Генетические алгоритмы / Под ред. В.М. Курейчика. – 2-е изд., исправл. и доп. – М.: ФИЗМАТЛИТ, 2010 – 368 с. – ISBN 978-5-9221-0510-1.
7. Граф [Электронный ресурс] – Режим доступа: <https://prog-cpp.ru/data-graph/>
8. Еремеева Л.Э. Транспортная логистика: учебное пособие / Л. Э. Еремеева; Сыкт. лесн. ин-т. – Сыктывкар: СЛИ, 2013. – 260 с.
9. Жигалова В.Н. Логистика: учебное пособие / В. Н. Жигалова – Томск: Эль Контент, 2013. – 166 с.
10. Задача коммивояжера [Электронный ресурс] : Режим доступа: https://ru.wikipedia.org/wiki/Задача_коммивояжера
11. Зыков А.А. Основы теории графов. – М.:Наука, 1987, 384 с.
12. Левитин А. Алгоритмы. Введение в разработку и анализ. Вильямс, 2006. Р. 160.

13. Минтранс России [Электронный ресурс] : Режим доступа: <https://mintrans.ru/>
14. О. Оре Графы и их применение. Пер. с англ. под ред. И.М. Яглома. - М., «Мир», 1965, 174 с.
15. Панченко, Т. В. Генетические алгоритмы [Текст] : учебно-методическое пособие / под ред. Ю. Ю. Тарасевича. – Астрахань : Издательский дом «Астраханский университет», 2007 –м 87 с.
16. Расстояние между двумя точками [Электронный ресурс] – Режим доступа: https://ru.onlinemschool.com/math/library/analytic_geometry/point_point_length/
17. Савин В.И. Перевозки грузов автомобильным транспортом: Справ. Пособие. – М. : Дело и сервис, 2002. – 544 с.
18. Сеть магазинов Магнит [Электронный ресурс] : Режим доступа: [https://ru.wikipedia.org/wiki/Магнит_\(сеть_магазинов\)](https://ru.wikipedia.org/wiki/Магнит_(сеть_магазинов))
19. Стуканов В.А. Автомобильные эксплуатационные материалы: Учеб.пособие. Лабораторный практикум. – М.: ФОРУМ:ИНФРА-М, 2003. – 208с.
20. Уилсон Р. Введение в теорию графов. Пер. с англ. 1977. 208 с.
21. Фокин В.В., Марков С.Б. Материаловедение на автомобильном транспорте: Учеб.пособие. – СПб: СПбГИЭУ, 2004. – 231 с
22. Экономическая эффективность моторного топлива [Электронный ресурс] : Режим доступа: https://studbooks.net/1588352/tehnika/otsenka_ekonomicheskoy_effektivnosti_primeneniya_komprimirovannogo_gaza_kachestve_motornogo_topliva
23. Яндекс.Карты. [Электронный ресурс] : Режим доступа: <https://yandex.ru/maps>
24. 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.

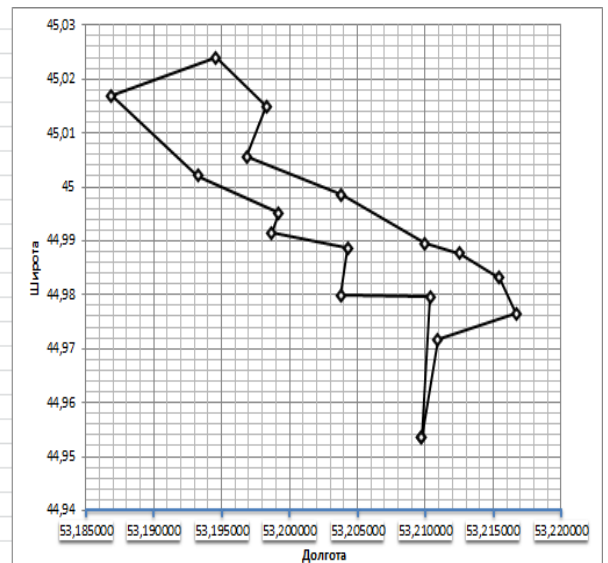
25. Speedy Route. [Электронный ресурс] : Режим доступа: <https://www.speedyroute.com>
26. Dorigo M., Caro G. D. Ant Algorithms for Discrete Optimization // Artificial Life, 1999. Vol. 5, No 2. P. 139–140
27. Meganavigator [Электронный ресурс] : Режим доступа: <http://ww1.meganavigator.com/?sub1=428ddb0c-5d4f-11ea-a397-f8536b6dc515>
28. Google Maps [Электронный ресурс] : Режим доступа: <https://maps.google.com/>
29. MAN и «Магнит». Эволюция и варианты грузовиков[Электронный ресурс] : Режим доступа: <http://Papadimitriou C., Vazirani U. Efficient Algorithms and Intractable Problems. Berkeley EECS, 1999. Chapter 10. s://mpark.pro/pulse-of-industry/305-man-and-magnit-history-of-partnership.html>
30. Papadimitriou C., Vazirani U. Efficient Algorithms and Intractable Problems. Berkeley EECS, 1999. Chapter 10.
31. 2GIS [Электронный ресурс] : Режим доступа: <https://2gis.ru/penza>
32. ООО «Лукойл» [Электронный ресурс] – Режим доступа: <https://lukoil.ru/>

ПРИЛОЖЕНИЕ А – Поиск решения в Excel

E29		fx																							
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	Координаты магазинов						Расстояния между магазинами																		
2	Магнит	№ Магнита	X	Y	X, км	Y, км		Магазины	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3	Магнит0	0	53,209713	44,953484	5913,454	5004,28343		0	0,0	2,0	2,7	3,4	2,9	3,8	4,0	3,0	4,0	5,0	4,4	4,8	6,0	5,7	7,0	8,0	7,5
4	Магнит1	1	53,210926	44,971666	5913,58881	5006,30747		1	2,0	0,0	0,8	1,4	0,9	1,8	2,0	1,2	2,0	3,1	2,6	2,9	4,1	3,9	5,0	6,1	5,7
5	Магнит2	2	53,216702	44,976427	5914,23073	5006,83747		2	2,7	0,8	0,0	0,8	0,8	1,3	1,6	1,5	1,9	2,8	2,6	2,8	3,9	3,9	4,7	5,8	5,6
6	Магнит3	3	53,215472	44,983156	5914,09403	5007,58655		3	3,4	1,4	0,8	0,0	0,7	0,6	0,9	1,4	1,4	2,1	2,1	2,2	3,2	3,2	4,0	5,1	4,9
7	Магнит4	4	53,210376	44,97967	5913,52769	5007,19849		4	2,9	0,9	0,8	0,7	0,0	0,9	1,1	0,7	1,2	2,2	1,8	2,1	3,3	3,1	4,1	5,2	4,9
8	Магнит5	5	53,212539	44,987602	5913,76807	5008,08149		5	3,8	1,8	1,3	0,6	0,9	0,0	0,4	1,3	0,9	1,5	1,6	1,7	2,7	2,7	3,4	4,5	4,3
9	Магнит6	6	53,209988	44,989552	5913,48457	5008,29856		6	4,0	2,0	1,6	0,9	1,1	0,4	0,0	1,3	0,6	1,2	1,3	1,3	2,3	2,3	3,1	4,2	4,0
10	Магнит7	7	53,203759	44,979859	5912,79231	5007,21953		7	3,0	1,2	1,5	1,4	0,7	1,3	1,3	0,0	1,0	2,1	1,4	1,8	3,0	2,7	4,0	5,0	4,5
11	Магнит8	8	53,204314	44,988689	5912,85399	5008,20249		8	4,0	2,0	1,9	1,4	1,2	0,9	0,6	1,0	0,0	1,1	0,7	0,9	2,1	1,9	3,0	4,1	3,7
12	Магнит9	9	53,20384	44,998463	5912,80131	5009,29055		9	5,0	3,1	2,8	2,1	2,2	1,5	1,2	2,1	1,1	0,0	1,0	0,6	1,1	1,2	1,9	3,0	2,8
13	Магнит10	10	53,198683	44,99142	5912,22819	5008,50651		10	4,4	2,6	2,6	2,1	1,8	1,6	1,3	1,4	0,7	1,0	0,0	0,4	1,6	1,3	2,6	3,6	3,1
14	Магнит11	11	53,199201	44,995094	5912,28576	5008,91551		11	4,8	2,9	2,8	2,2	2,1	1,7	1,3	1,8	0,9	0,6	0,4	0,0	1,2	1,0	2,2	3,2	2,8
15	Магнит12	12	53,196871	45,005578	5912,02681	5010,0826		12	6,0	4,1	3,9	3,2	3,3	2,7	2,3	3,0	2,1	1,1	1,6	1,2	0,0	0,6	1,1	2,1	1,7
16	Магнит13	13	53,193272	45,002074	5911,62684	5009,69253		13	5,7	3,9	3,9	3,2	3,1	2,7	2,3	2,7	1,9	1,2	1,3	1,0	0,6	0,0	1,5	2,4	1,8
17	Магнит14	14	53,198322	45,014938	5912,18807	5011,12457		14	7,0	5,0	4,7	4,0	4,1	3,4	3,1	4,0	3,0	1,9	2,6	2,2	1,1	1,5	0,0	1,1	1,3
18	Магнит15	15	53,194589	45,023894	5911,7732	5012,12156		15	8,0	6,1	5,8	5,1	5,2	4,5	4,2	5,0	4,1	3,0	3,6	3,2	2,1	2,4	1,1	0,0	1,2
19	Магнит16	16	53,18689	45,016906	5910,91757	5011,34365		16	7,5	5,7	5,6	4,9	4,9	4,3	4,0	4,5	3,7	2,8	3,1	2,8	1,7	1,8	1,3	1,2	0,0

Магазин	Последовательность	Расстояние
Магнит0	0	2,0
Магнит1	1	0,8
Магнит2	2	0,8
Магнит3	3	0,6
Магнит5	5	0,4
Магнит6	6	1,2
Магнит9	9	1,1
Магнит12	12	1,1
Магнит14	14	1,1
Магнит15	15	1,2
Магнит16	16	1,8
Магнит13	13	1,0
Магнит11	11	0,4
Магнит10	10	0,7
Магнит8	8	1,0
Магнит7	7	0,7
Магнит4	4	2,9
Магнит0	0	18,7
Число уникальных		17

Кратчайший путь	
53,209713	44,95348
53,210926	44,97167
53,216702	44,97643
53,215472	44,98316
53,212539	44,9876
53,209988	44,98955
53,203840	44,99846
53,196871	45,00558
53,198322	45,01494
53,194589	45,02389
53,186890	45,01691
53,193272	45,00207
53,199201	44,99509
53,198683	44,99142
53,204314	44,98869
53,203759	44,97986
53,210376	44,97967
53,209713	44,95348



ПРИЛОЖЕНИЕ Б– Листинг программного кода.

Программа реализована на VBA как Windows Form Application.

```
using System; // Random, InvalidCastException и пр.
```

```
using System.Windows.Forms; // PictureBox (tourDiagram), Application (см. класс  
class Program), MessageBox
```

```
using System.Collections.Generic; // Для List
```

```
using System.Data; // DataSet, DataRowCollection
```

```
using System.Drawing; // Image, Graphics, Point
```

```
using System.Threading; // ThreadPool
```

```
using System.IO; // FileNotFoundException
```

```
namespace Tsp
```

```
{
```

```
// Класс TspForm управляет формой приложения
```

```
public partial class TspForm : Form
```

```
{
```

```
    public int populationSize = 0; // Размер популяции
```

```
    public int maxGenerations = 0; // Предельное число поколений
```

```
    public int mutationChance = 0; // Вероятность мутации дочернего маршрута
```

```
    public int wrGoupSize = 0; // Размер рабочей группы
```

```
    public int numberOfCloseCities = 0; // Число магазинов-соседей любого  
магазина
```

```
    public int chanceUseCloseCity = 0; // Вероятность добавления в маршрут  
магазина из списка магазинов-соседей
```

```
    public int seed = 0; // Затравка датчика случайных чисел
```

```
    // Список магазинов, которые должен посетить коммивояжер
```

```
    Cities cityList = new Cities();
```

```
    // Tsp – класс, реализующий генетический алгоритм решения задачи  
коммивояжера (tsp-алгоритм)
```

```
    // Если tsp не null, то задача выполняется
```

```

Tsp tsp;
// Образ для отображения магазинов и маршрута
public Image cityImage;
// Графический объект для образа cityImage
public Graphics cityGraphics;
// Используется отдельная нить, чтобы обновлять карту магазинов во время
вычислений

public delegate void DrawEventHandler(Object sender, TspEventArgs e);
// Стандартный конструктор
public TspForm()
{
    InitializeComponent();
}
// tsp-алгоритм порождает событие foundNewBestTour при нахождении
текущего лучшего маршрута
// При этом оживляется GUI-нить, что позволяет обновить карту магазинов
private void tsp_foundNewBestTour(object sender, TspEventArgs e)
{
    if (this.InvokeRequired)
    {
        try
        {
            this.Invoke(new DrawEventHandler(DrawTour), new object[] { sender, e });
            return;
        }
        catch (Exception) { } // Неудача
    }
    // Отображаем маршрут
    DrawTour(sender, e);

```

```

}
// Отображает последний лучший маршрут
// и обновляет поля с числом итераций, длиной маршрута и числом улучшений
решения

public void DrawTour(object sender, TspEventArgs e)
{
    Tour bestTour = e.BestTour; // Лучший маршрут
    cityList = e.CityList;
    this.lastFitnessValue.Text = Math.Round(bestTour.Fitness, 2).ToString();
    this.lastIterationValue.Text = e.Generation.ToString();
    this.NumberOfImprv.Text = e.Imprv.ToString();
    int city = bestTour[0];
    int nextCity;
    cityGraphics.FillRectangle(Brushes.White, 0, 0, cityImage.Width,
cityImage.Height);
    Point pos;
    int cTour = bestTour.Count;
    for (int step = 1; step < cTour; step++)
    {
        pos = cityList[city].Location;
        nextCity = bestTour[step];
        city = nextCity;
        // Отображаем магазин в виде окружности
        // Начальный магазин маршрута заливаем красным цветом, второй – синим,
последний – желтым
        if (step == 1) // Окружность начального магазина маршрута заливаем красным
цветом
        {
            cityGraphics.FillEllipse(Brushes.Red, pos.X - 5, pos.Y - 5, 10, 10);

```

```

    }
    else if (step == 2) // Окружность второго магазина маршрута заливаем синим
цветом
    {
        cityGraphics.FillEllipse(Brushes.Blue, pos.X - 5, pos.Y - 5, 10, 10);
    }
    else if (step == cTour - 1) // Окружность предпоследнего магазина маршрута
заливаем желтым цветом
    {
        cityGraphics.FillEllipse(Brushes.Gold, pos.X - 5, pos.Y - 5, 10, 10);
    }
    else
    {
        cityGraphics.DrawEllipse(Pens.Black, pos.X - 3, pos.Y - 3, 6, 6);
    }
    // Соединяем 2 магазина линией
    cityGraphics.DrawLine(Pens.Black, pos, cityList[nextCity].Location);
}
tourDiagram.Image = cityImage;
if (e.Done)
{
    StartButton.Text = "Пуск";
    StatusLabel.Text = "Откройте XML или укажите магазин на карте мышкой";
    StatusLabel.ForeColor = Color.Black;
}
}
// Отображает магазины на растровой карте
private void DrawCityList(Cities cityList)
{

```

```

cityImage = new Bitmap(tourDiagram.Width, tourDiagram.Height);
cityGraphics = Graphics.FromImage(cityImage);
foreach (City city in cityList)
{
    // Магазин в виде окружности
    cityGraphics.DrawEllipse(Pens.Black, city.Location.X - 3, city.Location.Y - 3, 6,
6);
}
this.tourDiagram.Image = cityImage;
// Обновляем в форме текстовое поле с числом магазинов
this.NumberCitiesValue.Text = cityList.Count.ToString();
}
// Запускает или останавливает решение задачи коммивояжера
private void StartButton_Click(object sender, EventArgs e)
{
    // Останавливаем нить tsp, если задача решается
    if (tsp != null)
    {
        tsp.Halt = true;
        return;
    }
    try
    {
        populationSize = Convert.ToInt32(populationSizeTextBox.Text);
        maxGenerations = Convert.ToInt32(maxGenerationTextBox.Text);
        mutationChance = Convert.ToInt32(mutationTextBox.Text);
        wrGoupSize = Convert.ToInt32(groupSizeTextBox.Text);
        numberOfCloseCities = Convert.ToInt32(NumberCloseCitiesTextBox.Text);
        chanceUseCloseCity = Convert.ToInt32(CloseCityOddsTextBox.Text);
    }
}

```

```

    seed = Convert.ToInt32(randomSeedTextBox.Text);
}
catch (FormatException)
{
    MessageBox.Show("В полях ввода должны быть числа", "Ошибка",
    MessageBoxButtons.OK, MessageBoxIcon.Error);

    return;
}
string show = "";
bool ok = true;
if (cityList.Count == 0)
{
    show = "Список магазинов пуст"; ok = false;
}
if (ok && cityList.Count < 5)
{
    show = "Число магазинов не может быть меньше, чем 5"; ok = false;
}
if (ok && populationSize <= 0)
{
    show = "Укажите размер популяции"; ok = false;
}
if (ok && maxGenerations <= 0)
{
    show = "Задайте максимальное число поколений"; ok = false;
}
if (ok && ((mutationChance < 0) || (mutationChance > 100)))
{
    show = "Число мутаций должно быть между 0 и 100"; ok = false;
}

```

```

    }
    if (ok && ((wrGoupSize < 2) || (wrGoupSize > populationSize)))
    {
        show = "Размер рабочей группы задается между 2 и размером популяций"; ok
= false;
    }
    if (ok && ((numberOfCloseCities < 3) || (numberOfCloseCities >=
cityList.Count)))
    {
        show = "Число магазина-соседа для создания начальной популяции должно
быть между 3 и числом магазинов"; ok = false;
    }
    if (ok && ((chanceUseCloseCity < 0) || (chanceUseCloseCity > 100)))
    {
        show = "Вероятность использования магазина-соседа при создании начальной
популяции лежит между 0 и 100"; ok = false;
    }
    if (ok && seed < 0)
    {
        show = "Затравка датчика случайных чисел не может быть менее нуля"; ok =
false;
    }
    if (!ok)
    {
        MessageBox.Show(show, "Ошибка", MessageBoxButtons.OK,
MessageBoxIcon.Error);
        return;
    }
    this.StartButton.Text = "Стоп";

```

```

ThreadPool.QueueUserWorkItem(new WaitCallback(BeginTsp));
}
// Создает объект класса Tsp
// Выполняется в отдельной нити
private void BeginTsp(Object stateInfo)
{
    // Полагаем, что все данные введены корректно (проверяются в
    StartButton_Click)
    // Находим для каждого магазина расстояния до всех прочих городов, а также
    находим его соседей
    cityList.CalculateCityDistances(numberOfCloseCities);
    tsp = new Tsp();
    tsp.foundNewBestTour += new
    Tsp.NewBestTourEventHandler(tsp_foundNewBestTour);
    tsp.FindBestTour(populationSize, maxGenerations, wrGoupSize, mutationChance,
    seed, chanceUseCloseCity, cityList, numberOfCloseCities);
    tsp.foundNewBestTour -= new
    Tsp.NewBestTourEventHandler(tsp_foundNewBestTour);
    tsp = null;
}
// Выбирает XML-файл с координатами магазинов
private void selectFileButton_Click(object sender, EventArgs e)
{
    OpenFileDialog fileOpenDialog = new OpenFileDialog();
    fileOpenDialog.Filter = "XML(*.xml)|*.xml";
    fileOpenDialog.InitialDirectory = ".";
    fileOpenDialog.ShowDialog();
    fileNameTextBox.Text = fileOpenDialog.FileName;
}

```

```

// Формируем список магазинов по XML-файлу
private void openCityListButton_Click(object sender, EventArgs e)
{
    string fileName = "";
    try
    {
        if (tsp != null)
        {
            StatusLabel.Text = "Нельзя изменять список магазинов во время работы
алгоритма";
            StatusLabel.ForeColor = Color.Red;
            return;
        }
        fileName = this.fileNameTextBox.Text;
        // Читаем файл и формируем список магазинов
        cityList.OpenCityList(fileName);
        // Отображаем магазины на карте
        this.DrawCityList(cityList);
    }
    catch (FileNotFoundException)
    {
        MessageBox.Show("Файл не найден: " + fileName, "Ошибка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
    catch (InvalidCastException)
    {
        MessageBox.Show("Плохой формат XML-файла", "Ошибка",
MessageBoxButtons.OK, MessageBoxIcon.Error);
    }
}

```

```

catch
{
    MessageBox.Show("Не выбран файл", "Ошибка", MessageBoxButtons.OK,
    MessageBoxIcon.Error);
}
}

private bool stillRun()
{
    if (tsp != null) // Если выполняется поиск решения
    {
        StatusLabel.Text = "Нельзя изменять список магазинов во время работы
алгоритма";
        StatusLabel.ForeColor = Color.Red;
        return true;
    }
    return false;
}

// Очистка списка магазинов
private void clearCityListButton_Click(object sender, EventArgs e)
{
    if (stillRun()) return;
    cityList.Clear();
    this.DrawCityList(cityList);
}

// Размещает магазин на карте в точке мышиного удара
private void tourDiagram_MouseDown(object sender, MouseEventArgs e)
{
    if (stillRun()) return;
    cityList.Add(new City(e.X, e.Y)); // e.X, e.Y - координаты мыши в физической

```

системе координат

```
this.DrawCityList(cityList);
}
}
// Класс Program предоставляет метод Main для точки входа (entry point)
static class Program
{
    [STAThread] // Точка входа приложения
    static void Main()
    {
        Application.Run(new TspForm());
    }
}
// Класс TspEventArgs устанавливает аргументы события, при наступлении
// которого отображается очередной лучший маршрут
public class TspEventArgs : EventArgs
{
    // Конструктор, устанавливающий значения свойств события
    public TspEventArgs(Cities cityList, Tour bestTour, int generation, bool done, int
imprv)
    {
        this.cityList = cityList; // Список магазинов
        this.bestTour = bestTour; // Лучший маршрут
        this.generation = generation; // Номер поколения
        this.done = done; // Флаг завершения алгоритма
        this.imprv = imprv; // Число улучшений решения
    }
}
// Список магазинов (private копия и public)
private Cities cityList;
```

```

public Cities CityList
{
    get
    {
        return cityList;
    }
}

// Лучший маршрут (private копия и public)
private Tour bestTour;
public Tour BestTour
{
    get
    {
        return bestTour;
    }
}

// Номер поколения, в котором появился текущий лучший маршрут (private
копия и public)
private int generation;
public int Generation
{
    get
    {
        return generation;
    }
    set
    {
        generation = value;
    }
}

```

```

}
// Флаг завершения алгоритма (private копия и public)
private bool done = false;
public bool Done
{
    get
    {
        return done;
    }
    set
    {
        done = value;
    }
}
// Число улучшений маршрута (private копия и public)
private int imprv;
public int Imprv
{
    get
    {
        return imprv;
    }
    set
    {
        imprv = value;
    }
}
}
// Класс City обеспечивает формирование и хранение позиции магазина

```

```

(свойство Location),
// хранение в списке Distances расстояний от текущего магазина до всех прочих
магазинов,
// формирование (метод FindClosestCities) и хранение списка CloseCities
магазинов-соседей
public class City
{
// Конструктор, обеспечивающий формирование позиции магазина
public City(int x, int y)
{
Location = new Point(x, y); // System.Drawing.Point class
}
// Позиция магазина (private копия и public)
private Point location;
public Point Location
{
get
{
return location;
}
set
{
location = value;
}
}
// Расстояния от магазина до каждого из прочих магазинов (private копия и
public)
// Индекс списка - это номер соответствующего магазина
private List<double> distances = new List<double>();

```

```

public List<double> Distances
{
    get
    {
        return distances;
    }
    set
    {
        distances = value;
    }
}

// Список магазинов-соседей (private копия и public)
private List<int> closeCities = new List<int>();
public List<int> CloseCities
{
    get
    {
        return closeCities;
    }
}

// Находит для текущего магазина его магазины-соседи (их число равно
numberOfCloseCities)

// Помещает их в список closeCities
public void FindClosestCities(int numberOfCloseCities)
{
    double shortestDistance;
    int shortestCity = 0;
    int dCnt = Distances.Count;
    double[] dist = new double[dCnt];

```

```

// Копируем список расстояний магазина до прочих городов в массив dist
Distances.CopyTo(dist);
closeCities.Clear();
for (int i = 0; i < numberOfCloseCities; i++)
{
    shortestDistance = Double.MaxValue;
    for (int city = 0; city < dCnt; city++)
    {
        if (dist[city] < shortestDistance)
        {
            shortestDistance = dist[city];
            shortestCity = city;
        }
    }
    closeCities.Add(shortestCity);
    dist[shortestCity] = Double.MaxValue;
}
}
}

// Класс Cities содержит список городов
// Для каждого магазина указываются его координаты, расстояния до каждого
прочего магазина и список магазинов-соседей
public class Cities : List<City>
{
    // Находит и помещает в список city.Distances расстояния от выбранного
магазина до всех прочих магазинов
    // Находит для каждого магазина его numberOfCloseCities магазинов-соседей и
помещает их в список closeCities
    public void CalculateCityDistances(int numberOfCloseCities)

```

```

{
    foreach (City city in this)
    {
        city.Distances.Clear();
        Point pos = city.Location;
        for (int i = 0; i < Count; i++)
        {
            Point pos2 = this[i].Location;
            city.Distances.Add(Math.Sqrt(Math.Pow((double)(pos.X - pos2.X), 2D) +
Math.Pow((double)(pos.Y - pos2.Y), 2D)));
        }
    }
    foreach (City city in this)
    {
        // Находим для каждого магазина его numberOfCloseCities соседей
        city.FindClosestCities(numberOfCloseCities);
    }
}

// Формирует список магазинов по данным XML-файла
// Исключения:
// плохой формат XML-файла
// не задано имя XML-файла
// плохое имя файла (пара метр fileName)
// Фрагмент XML-файла
// <?xml version="1.0" encoding="utf-8" ?>
// <CitiesPos>
// <CityPos X="121" Y="132"/>
// <CityPos X="160" Y="370"/>
// ...

```

```

// </CitiesPos>

public void OpenCityList(string fileName)
{
    DataSet cityDS = new DataSet();

    try
    {
        this.Clear();
        cityDS.ReadXml(fileName);
        // Коллекция строк с координатами городов; X и Y - имена атрибутов тега
        City
        DataRowCollection cities = cityDS.Tables[0].Rows;
        foreach (DataRow city in cities)
        {
            this.Add(new City(Convert.ToInt32(city["X"]), Convert.ToInt32(city["Y"]))); //
или city[0], city[1]
        }
    }
    finally
    {
        cityDS.Dispose();
    }
}

// Класс Population обеспечивает формирование начальной популяции
class Population : List<Tour>
{
    // Лучший маршрут (private-копия и public), найденный генетическим
алгоритмом
    private Tour bestTour = null;

```

```

public Tour BestTour
{
    set
    {
        bestTour = value;
    }
    get
    {
        return bestTour;
    }
}

// Число улучшений решения (private-копия и public)
private int imprv = 0;
public int Imprv
{
    set
    {
        imprv = value;
    }
    get
    {
        return imprv;
    }
}

// Формирует популяцию - начальный набор случайных маршрутов
// populationSize - число создаваемых маршрутов (размер популяции)
// cityList - список магазинов
// rand - генератор случайных чисел. Обеспечивает различные результаты при
запусках

```

```

// chanceToUseCloseCity - вероятность выбора магазина из списка магазинов-
соседей
// numberOfCloseCities - число магазинов-соседей
public void CreateRandomPopulation(int populationSize, Cities cityList, Random
rand, int chanceToUseCloseCity, int numberOfCloseCities)
{
    int cCnt = cityList.Count; // Число магазинов
    // В маршруте на один магазин больше
    // Формируется цепочка магазинов, в которой первый и последний магазины
совпадают
    int tCnt = cCnt + 1;
    int city, cCity, nextCity;
    int tourCount;
    int step; // Номер участка маршрута
    bool[] cityUsage = new bool[cCnt]; // Список магазинов, не включенных в
маршрут (свободных магазинов)
    List<int> cities = new List<int>();
    Tour tourTmp, tour;
    // Можно ввести параметр "Начальный магазин маршрутов" или положить
cityNumber = rand.Next(cCnt)
    int cityNumber = 0; // Все маршруты в итоге будут начинаться с города
cityNumber
    for (tourCount = 0; tourCount < populationSize; tourCount++)
    {
        for (city = 0; city < cCnt; city++)
        {
            cities.Add(city);
            cityUsage[city] = false; // true, если магазин уже включен в маршрут
        }
    }

```

```

tourTmp = new Tour(cCnt);
cCity = rand.Next(cCnt); // магазин, добавляемый в маршрут
tourTmp[0] = cCity; // Первый магазин маршрута
cityUsage[cCity] = true; // магазин с номером включен в маршрут
cities.Remove(cCity); // Удаляем магазин из списка свободных городов
nextCity = cCity;
for (step = 1; step < cCnt; step++)
{
    // Подбираем следующий магазин маршрута
    if (rand.Next(100) < chanceToUseCloseCity)
    {
        // Берем первый свободный магазин из списка магазинов-соседей
        for (city = 0; city < numberOfCloseCities; city++)
        {
            nextCity = cityList[cCity].CloseCities[city];
            if (!cityUsage[nextCity]) break;
        }
    }

    // Если магазин уже в маршруте, то берем магазин из списка магазинов, не
    включенных в маршрут
    if (cityUsage[nextCity])
    {
        nextCity = cities[rand.Next(cities.Count)];
    }
    cityUsage[nextCity] = true;
    tourTmp[step] = nextCity; // Включаем магазин с номером в маршрут
    cities.Remove(nextCity);
    cCity = nextCity;
}

```

```

tour = new Tour(tCnt);
if (tourTmp[0] == cityNumber)
{
    for (step = 0; step < cCnt; step++)
    {
        tour[step] = tourTmp[step];
    }
}
else
{
    // Ищем позицию магазина с номером cityNumber
    int city0 = 0, m = -1, n = 0;
    for (step = 0; step < cCnt; step++)
    {
        if (tourTmp[step] == cityNumber)
        {
            city0 = step;
            break;
        }
    }
    // Ставим в маршруте tour на первое место магазин с номером cityNumber
    // и добавляем в tour магазина из tourTmp, расположенные после cityNumber
    for (step = city0 + 1; step < cCnt; step++)
    {
        m++;
        tour[m] = tourTmp[step];
    }
    // Добавляем в tour города из tourTmp, расположенные перед cityNumber
    for (step = 0; step < city0; step++)

```

```

    {
        n++;
        tour[m + n] = tourTmp[step];
    }
}

tour[cCnt] = cityNumber; // Первый и последний магазины маршрута
совпадают

// Вычисляем длину маршрута
tour.DetermineFitness(cityList);
// Добавляем маршрут в популяцию
Add(tour);
// Изменяем, если на то есть основание, информацию о лучшем маршруте
if ((bestTour == null) || (tour.Fitness < bestTour.Fitness))
{
    BestTour = tour;
}
}
}
}

// Класс Tour представляет маршрут по всем магазинам
public class Tour : List<int>
{
    // Конструктор, принимающий параметр capacity
    // capacity - размер маршрута. Равен числу магазинов + 1
    // Начальный пустой маршрут
    public Tour(int capacity)
    {
        this.Clear();
        for (int i = 0; i < capacity; i++)

```

```

    {
        this.Add(-1);
    }
}

// Протяженность маршрута (private копия и public)
private double fitness;
public double Fitness
{
    set
    {
        fitness = value;
    }
    get
    {
        return fitness;
    }
}

// Определяет общую протяженность маршрута
// cities - список магазинов маршрута. Используется, чтобы получить
расстояние между магазинами
public void DetermineFitness(Cities cities)
{
    this.Fitness = 0;
    for (int i = 0; i < this.Count - 1; i++)
    {
        this.Fitness += cities[this[i]].Distances[this[i + 1]];
    }
}

public static Tour Crossover(Tour parent1, Tour parent2, Cities cityList, Random

```

```

rand)
{
    int cCnt = cityList.Count;
    Tour child = new Tour(cCnt + 1); // Новый (дочерний) маршрут
    int step;    // Номер участка маршрута
    // Первоначально дочерний маршрут совпадает с parent1
    // Первый и последний магазин дочернего маршрута совпадают, как и у всех
прочих маршрутов    for (step = 0; step < cCnt + 1; step++)
    {
        child[step] = parent1[step];
    }
    for (int i = 0; i < 2; i++) // Можно ввести параметр "Число скрещиваний" (i <
nChanges взамен i < 2)
    {
        // Можно ввести параметр "Длина куска при скрещивании"; сейчас – это
кусочек с тремя городами
        child = makeChange(child, parent2, rand, cCnt, 3);
    }
    return child;
}
// child - дочерний маршрут
// parent2 - второй родительский маршрут (при мутации равен child)
static Tour makeChange(Tour child, Tour parent2, Random rand, int cCnt, int nCts)
{
    int step, step2, city, city2;
    List<int> cities = new List<int>(); // Удаляемые из child магазины
    List<int> cities2 = new List<int>(); // Добавляемые в child магазины
    List<int> cities3 = new List<int>(); // Города для восполнения child
    // nCts - протяженность заменяемого куска маршрута

```

```

// Берем из parent2 кусок маршрута с nCts магазинами и заменяем
симметричный кусок в дочернем маршруте
// При мутации parent2 = child
step2 = rand.Next(cCnt - nCts);
while (step2 == 0) step2 = rand.Next(cCnt - nCts);
for (step = step2; step < step2 + nCts; step++)
{
    city = child[step];
    city2 = parent2[step];
    child[step] = city2;
    cities.Add(city); // Удаляемые из child магазины
    cities2.Add(city2); // Добавляемые в child магазины
    cities3.Add(city); // магазины для восполнения child
}
// Исключаем из списков cities2 и cities3 магазины списка cities
int k, k2;
for (k = 0; k < nCts; k++)
{
    city = cities[k];
    for (k2 = 0; k2 < cities2.Count; k2++)
    {
        if (cities2[k2] == city)
        {
            cities2.Remove(city);
            cities3.Remove(city);
            break;
        }
    }
}
}

```

```

// Находим в дочернем маршруте оставшиеся в списке cities2 магазины и
// заменяем их на магазины из списка cities3
int clft = cities2.Count;
for (k = 0; k < clft; k++)
{
    city = cities2[k];
    for (step = 1; step < cCnt; step++)
    {
        if (child[step] == city)
        {
            city2 = cities3[rand.Next(cities3.Count)];
            child[step] = city2;
            cities3.Remove(city2);
            break;
        }
    }
}
return child;
}

// Мутация
// Случайным образом меняем местами 2 магазина маршрута или 2 куска
// маршрута
// child - дочерний маршрут
// rand - генератор случайных чисел
public static Tour Mutate(Tour child, Random rand)
{
    int tCnt = child.Count - 1;
    // Можно добавить параметр flipCity - "Тип мутации"
    // Если flipCity = true, то перестановка магазинов, иначе перестановка кусков

```

маршрута

```
bool flipCity = true;
int nMutations = 2;
for (int i = 0; i < nMutations; i++) // Можно добавить параметр nMutations -
"Число мутаций"
{
    if (flipCity)
    {
        int step = 0, step2 = 0;
        while (step == 0) step = rand.Next(tCnt);
        while (step2 == 0 || step2 == step) step2 = rand.Next(tCnt);
        int city = child[step];
        child[step] = child[step2];
        child[step2] = city;
    }
    else
    {
        // Можно ввести параметр mtnPiece - "Длина куска маршрута при мутации"
        int mtnPiece = 3;
        child = makeChange(child, child, rand, tCnt, mtnPiece);
    }
}
return child;
}
}
// Класс Tsp координирует решение задачи коммивояжера
class Tsp
{
    // Обработчик события "Найден текущий лучший маршрут"
```

```

// sender - объект, создавший событие
// e - аргументы события. Параметр содержит информацию о лучших
маршрутах

public delegate void NewBestTourEventHandler(Object sender, TspEventArgs e);
public event NewBestTourEventHandler foundNewBestTour;

// Генератор случайных чисел
Random rand;

// Список магазинов. Используется для расчета расстояний между магазинами
Cities cityList;

// Список всех маршрутов
Population population;

// Флаг завершения работы алгоритма (поиска новых поколений) (private копия
и public)
private bool halt = false;
public bool Halt
{
    get
    {
        return halt;
    }
    set
    {
        halt = value;
    }
}

// Старт вычислений (алгоритма)
// populationSize - число случайных маршрутов, создаваемых до начала
вычислений
// maxGenerations - максимальное число скрещиваний

```

// wrGoupSize - число маршрутов, проверяемых в каждом поколении. 2 первых берутся в качестве родительских маршрутов, чьи дочерние маршруты заменят 2 худших маршрута в группе

// mutationChance - вероятность мутации дочернего маршрута

// seed - заправка датчика случайных чисел

// chanceToUseCloseCity - вероятность выбора магазина-соседа

// cityList - список магазинов

// numberOfCloseCities - число магазинов-соседей

```
public void FindBestTour(int populationSize, int maxGenerations, int wrGoupSize,  
int mutationChance, int seed, int chanceToUseCloseCity, Cities cityList, int  
numberOfCloseCities)
```

```
{
```

```
    rand = new Random(seed);
```

```
    this.cityList = cityList;
```

```
    population = new Population();
```

```
    population.CreateRandomPopulation(populationSize, cityList, rand,  
chanceToUseCloseCity, numberOfCloseCities);
```

```
    displayTour(population.BestTour, 0, false, 0);
```

```
    int generation;
```

```
    for (generation = 0; generation < maxGenerations; generation++)
```

```
    {
```

```
        if (Halt)
```

```
        {
```

```
            break; // по требованию GUI
```

```
        }
```

```
        makeChildren(wrGoupSize, mutationChance, generation);
```

```
    }
```

```
    displayTour(population.BestTour, generation, true, population.Imprv);
```

```
}
```

```

// Случайным образом выбирает из популяции группу маршрутов
// После сортировки группы по возрастанию длины маршрута два первых
рассматриваются как родительские маршруты
// Они участвуют в скрещивании
// Дочерний маршрут заменяет худший маршрут текущей рабочей группы
// wrGoupSize - размер рабочей группы (число маршрутов в группе)
// mutationChance - вероятность мутации (в %) дочернего маршрута
// generation - текущее поколение
void makeChildren(int wrGoupSize, int mutationChance, int generation)
{
    int[] wrGoup = new int[wrGoupSize]; // Рабочая группа
    int i;
    // Один и тот же маршрут может попасть в рабочую группу неоднократно
    for (i = 0; i < wrGoupSize; i++)
    {
        wrGoup[i] = rand.Next(population.Count);
    }
    // Пузырьковая сортировка массива wrGoup по возрастанию длины маршрута
    bool swapped = true;
    int pass = 1, hold; // pass - номер прохода
    while (swapped && pass < wrGoupSize)
    {
        swapped = false;
        for (i = 0; i < wrGoupSize - pass; i++)
        {
            if (population[wrGoup[i]].Fitness > population[wrGoup[i + 1]].Fitness)
            {
                swapped = true; // Обмен
                hold = wrGoup[i];

```

```

    wrGoup[i] = wrGoup[i + 1];
    wrGoup[i + 1] = hold;
}
}
pass++;
}

bool fndBestTour = false;

// Можно ввести параметр attempMax - "Число попыток улучшить
популяцию"
int attemps = 0, attempMax = 1;
while (!fndBestTour && attemps < attempMax)
{
    attemps++;
    // 0, 1 - индексы массива wrGoup. Элементы массива wrGoup с этими
индексами - это родительские маршруты
    fndBestTour = makeAChild(wrGoup, mutationChance, 0, 1, wrGoupSize - 1,
fndBestTour);

    // Меняем родительские маршруты местами
    if (!fndBestTour) fndBestTour = makeAChild(wrGoup, mutationChance, 1, 0,
wrGoupSize - 2, fndBestTour);
}

if (fndBestTour) displayTour(population.BestTour, generation, false,
population.Imprv);
}

// wrGoup - рабочая группа маршрутов
// p1 - индекс первого родительского маршрута в wrGoup
// p2 - индекс второго родительского маршрута в wrGoup
// bad - индекс заменяемого (плохого) маршрута в wrGoup
// fndBestTour - равен true, если в результате скрещивания и мутации получен

```

новый лучший маршрут

```
bool makeAChild(int[] wrGoup, int mutationChance, int p1, int p2, int bad, bool
fndBestTour)
{
    int badTour = wrGoup[bad]; // badTour - индекс плохого маршрута в списке
population
    // Скрещиваем два лучших маршрута рабочей группы. Результат - это
дочерний маршрут
    // При выполнении условия rand.Next(100) < mutationChance применяем
операцию мутации
    // Заменяем результатом последний или предпоследний маршрут этой группы
(в списке population находится в позиции badTour)
    Tour child = Tour.Crossover(population[wrGoup[p1]], population[wrGoup[p2]],
cityList, rand);
    // Мутация дочернего маршрута
    if (rand.Next(100) < mutationChance) child = Tour.Mutate(child, rand);
    // Находим длину дочернего маршрута
    child.DetermineFitness(cityList);
    // Проверяем, имеет ли дочерний маршрут наименьшую длину
    if (child.Fitness < population.BestTour.Fitness)
    {
        fndBestTour = true; // Найден текущий лучший маршрут
        population.BestTour = child;
        population.Imprv++; // Число улучшений решения
    }
    // Можно ввести параметр takeIfBetter – "Брать только улучшающий дочерний
маршрут"
    bool takeIfBetter = false;
    if (takeIfBetter)
```

```

{
    if (child.Fitness < population[badTour].Fitness) population[badTour] = child;
}
else
{
    population[badTour] = child;
}
return fndBestTour;
}

// Порождает событие, воспринимаемое GUI, отображающим маршрут
// bestTour - лучший на текущий момент маршрут
// generationNumber - число порожденных поколений
// done - если true, то вычисления завершаются
// imprv - число улучшений решения
void displayTour(Tour bestTour, int generationNumber, bool done, int imprv)
{
    this.foundNewBestTour(this, new TspEventArgs(cityList, bestTour,
generationNumber, done, imprv));
}

```