МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование кафедры)
01.03.02. Прикладная математика и информатика
(код и наименование направления подготовки, специальности)
Системное программирование и компьютерные технологии
(направленность (профиль)/специализация)

БАКАЛАВРСКАЯ РАБОТА

на тему Муравьиный алгоритм в решении задачи коммивояжера

Студент Д.А. Синельников
(И.О. Фамилия) (личная подпись)

Руководитель А.В. Очеповский
(И.О. Фамилия) (личная подпись)

Консультант Н.В. Ященко
(И.О. Фамилия) (личная подпись)

Допустить к защите

Заведующий кафедрой к.т.н., доцент, А.В. Очеповский
(ученая степень, звание, И.О. Фамилия) (личная подпись)
«_____»_____20_____г.

Тольятти 2018

АННОТАЦИЯ

Тема бакалаврской работы: «Муравьиный алгоритм в решении задачи коммивояжера».

Работа выполнена студентом Тольяттинского Государственного Университета института математики, физики и информационных технологий, группы ПМИб-1401, Синельниковым Денисом Александровичем. Выпускная квалификационная работа посвящена разработке программного кода для решения задачи коммивояжера с помощью муравьиного алгоритма на языке программирования Java.

Объект исследования бакалаврской работы – вычислительный процесс нахождения оптимального пути муравьиным алгоритмом.

Предмет исследования бакалаврской работы – приложение, определяющая кратчайший путь с помощью муравьиного алгоритма для решения задачи коммивояжера.

Цель бакалаврской работы – разработка приложения для анализа работы муравьиного алгоритма в задаче коммивояжера.

Для достижения цели работы необходимо решить следующие задачи:

- программная реализация алгоритма оптимизации с помощью муравьиного алгоритма;
- вычислительный эксперимент;
- анализ данных вычислительного эксперимента.

Результаты бакалаврской работы:

- разработан муравьиный алгоритм для решения задачи коммивояжера;
- программная реализация решения задачи коммивояжера с помощью муравьиного алгоритма.

Бакалаврская работа представлена на 46 странице, включает 21 иллюстраций, 2 таблицы, 7 формул, список используемой литературы состоящий из 24 источников.

ABSTRACT

The title of the graduation work is The Ant Algorithm for the Solving the Travelling Salesman Problem.

This graduation work is devoted to the development of code for solving the traveling salesman problem using “ant algorithm” in the Java programming language.

The object of this graduation work is the computational process of finding the optimal path with the ant algorithm.

The subject of the work is the creation of a program that determines the shortest path using the ant algorithm for solving the travelling salesman problem.

The purpose of the graduation work is to design an application for a visual representation of the ant algorithm and computational experiments.

The graduation work consists of an introduction, three parts, conclusions and references. Much attention is paid to the description of the ant algorithm for solving travelling salesman problem.

The first part describes the travelling salesman problem, its history and algorithms for the solution of this problem, the ant algorithms, solving of travelling salesman problem using ant colony’s algorithms.

The second part considers the development of the application that solves the travelling salesman problem with the help of the ant algorithm. It contains the classes of algorithms, the implementation of the algorithm, the application interface.

In conclusion, there is an overall assessment.

As a result, the program code was developed, described and tested to solve the traveling salesman problem with the help of the ant algorithm.

The graduation work consists of an explanatory note on _ pages, including _ figures, _ tables, the list of _ references including _ foreign sources and _ appendices on 1 CD.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

С помощью наглядной демонстрации работы алгоритмов можно обеспечить более высокий уровень методов решения задач оптимизации, чтобы при прохождении вычислительных экспериментов была такая возможность, как не только продемонстрировать работу алгоритма на примере, но и модифицировать алгоритм, позволяя увидеть разницу в решении задачи при использовании различных подходов, а также возможность изменения исходных данных для демонстрации работы одного алгоритма в различных условиях.

Задача коммивояжёра дает возможность получить решение с использованием различных алгоритмов. Одним из наиболее результативных алгоритмов считается алгоритм оптимизации подражанием муравьиной колонии (ant colony optimization) [6].

Задача коммивояжёра считается довольно значимой транспортной задачей, которая занимается планированием транспортных перевозок. Для решения данной задачи требуется определить наиболее лучший маршрут объезда всех городов. Задача коммивояжера может решать проблемы распределения общественного транспорта города, выбор наилучшего пути для проезда курьера и т.п.

Известна одна из программных реализаций муравьиного алгоритма, написанная на языке Java, обеспечивающая наглядное представление о механизме работы алгоритма с возможностью его модификации. Однако это решение не позволяет вносить изменения в исходные данные, а также не обеспечивает достаточного функционала для визуального проектирования графов.

Таким образом, актуальность темы бакалаврской работы обусловлена тем, что даже при наличии большого количества уже реализованных муравьиных алгоритмов необходимо и дальше реализовывать, и улучшать данные алгоритмы для поиска наилучшего результата, т.к. применение автоматизированных систем в области транспортной логистики – один из способов экономии ресурсов.

Данная бакалаврская работа отличается высокой практической значимостью. В ходе его создания была разработана программа, решающая задачу коммивояжера с помощью муравьиного алгоритма, позволяющая сделать процесс выбора оптимального пути наиболее результативным.

Программная реализация и визуализация данного алгоритма позволят проектировать графы и наглядно демонстрировать механизм действия муравьиного алгоритма решения задачи коммивояжёра.

Объект исследования ВКР – задача коммивояжера.

Предмет исследования ВКР – применимость муравьиного алгоритма для решения задачи коммивояжера.

Цель работы – исследование применимости муравьиного алгоритма для решения задачи коммивояжера**.**

Задачами работы являются:

- реализация алгоритма оптимизации с помощью муравьиного алгоритма;
- разработка муравьиного алгоритма для приложения.

Обзор по главам:

1) Первая глава описывает задачу коммивояжера, ее историю, алгоритмы решения данной задачи, муравьиные алгоритмы, решение задач коммивояжера с помощью муравьиных алгоритмов;

2) Вторая глава – разработка приложения, с помощью которого решается задача коммивояжера с помощью муравьиного алгоритма, описание классов, реализация алгоритма, интерфейс приложения, проведение вычислительного эксперимента;

ГЛАВА 1 ОБЗОР СУЩЕСТВУЮЩИХ СПОСОБОВ РЕШЕНИЯ ЗАДАЧИ КОММИВОЯЖЕРА

1.1 Транспортные задачи

Свойственным показателем каждого нынешнего города считается сформированный рынок продуктов и услуг. Пред фирмами, предлагающими данные элементы потребительского рынка собственным покупателям, стоит цель правильно осуществить работу, существующих в их директиве транспортных средств.

В связи с актуальностью проблемы, появилось огромное количество задач, которые возможно совместить в единую группу «транспортных задач». В данную группу вступает колоссальное разнообразие разных задач, которые объединены тем, то что целевая роль в них носит тот либо другой экономический смысл.

Таким образом, к примеру, традиционная транспортная задача состоит в развозке продукта некоторыми транспортными средствами конкретному числу покупателей. Теоретическая задача о загрузке рюкзака, обладающая собственной целью более выгодного заполнения ограниченного места, благополучно используется с целью оптимальной загрузки автомобильного транспорта. Очень важны и проблемы концепции, значимость которых в нынешних реалиях невозможно преуменьшать.

Особенную роль в транспортных задачах занимает задача коммивояжера, содержащая определение наиболее выгодного маршрута, проходящего посредством установленных пунктов. Кроме большой актуальности и фактического использования задача коммивояжера обладает значительным общетеоретическим значением: она применяется в качестве модели с целью исследования эвристических алгоритмов разных оптимизационных задач. В настоящий период колоссальный интерес уделяется способностям решения задачи коммивояжера и других транспортных проблем с поддержкой геоинформационных систем.

1.2 Общая характеристика задачи коммивояжера

Задача коммивояжёра (Traveling Salesman Problem) – одна из самых известных задач дискретной оптимизации. Задача заключается в нахождении самого выгодного маршрута, проходящего между отмеченных городов, в которой, как минимум один раз, с дальнейшим возвращением в начальный город. Задача коммивояжёра является NP-трудной, т.е. имеется возможность проверить правильность её решения за полиномиальное время.

Имеется ряд индивидуальных ситуаций единой постановки задачи, в частности геометрическая задача коммивояжёра (кроме того именуемая как планарная или евклидовая, если матрица расстояний отображает дистанции среди точек на плоскости), треугольная задача коммивояжёра симметричная и асимметричная задачи коммивояжёра.

Кроме того, имеется обобщение задачи, именуемая обобщённой задачей коммивояжёра.

1.2.1 Постановка задачи коммивояжера

Рассматривается соответствующая определение задачи. Курьер обязан привезти товар в конкретное количество зон и возвратиться в место отправления. Установить, в котором порядке он обязан обойти клиентов, для того чтобы путь занял минимальный период.

В связи с тем, то что эта работа посвящена решению задачи коммивояжера, сделан вывод, то что ребра, проходящие через все пары пунктов и «затраты» которые идут на проезд через два пункта находятся в зависимости от направления движения.

Под затратами в этой работе подразумевается время.

Как было отмечено, решение задачи ищется среди циклов в которых никак не происходит вторичных посещений мест.

Совокупность прохода для задачи показана графом $G = (V, E)$, где V – вершины графа, E – ребра графа. Вес ребра $Cij > 0$ эквивалентен времени проезда среди смежных вершин графа.

С целью комфортной программной реализации граф показывается в виде матрицы, ее размерность соответствует числу вершин в графе. Значение элемента матрицы dij аналогичен ценности веса ребра Cij в графе, оно устанавливает время проезда среди пунктов i и j. Где $i = j$, $dij = M$. Элемент M олицетворяет безграничность, которая не допускает переход «в себя».

1.2.2 Математическая модель задачи коммивояжера

Задача коммивояжера – задача целочисленная. Пусть $x_{ij} = 1$, когда путешественник переходит из i-ого города в j-ый и $x_{ij} = 0$, если этого не происходит. Для наглядности, введем (n + 1) город, который находится в том месте, где и первоначальный город, то есть расстояния от (n + 1) города до абсолютно другого, который отличен от первого, равен расстоянием от первоначального города. Также, если из первоначального города возможно только лишь выйти, то в (n + 1) город можно всего лишь прийти. Установим дополнительные целые значения, равные номеру визита данного города на пути $u_1 = 0$, $u_{n+1} = n$. Ради избегания замкнутых путей, необходимо покинуть первый город и вернуться в (n + 1) определим дополнительные ограничения, которые связывают переменные x_{ij} и переменные u_i. I = {1, ... , n} – множество городов, матрица (cij) – попарные промежутки через города, $1 \leq i, j \leq n$. Формула (4) представляет математическую модель задачи коммивояжера.

$$min \quad _{i \in I} \quad _{j \in I} c_{ij} x_{ij} \tag{1}$$

$$u_i - u_j + n x_{ij} \leq n - 1, j = 2 \ldots n + 1, i = 1 \ldots n, i \neq j, \tag{2}$$

где

$$i = 1, j \neq n + 1, 0 \leq u_i \leq n, x_{in+1} = x_{i1}, i = 2 \ldots n. \tag{3}$$

1.3 Современные алгоритмы с помощью которых решается задача коммивояжера

1.3.1 Точные алгоритмы

Алгоритм полного перебора (АПП) реализовывает отбор в пространстве

N решений с помощью перебора абсолютно всех альтернатив. Итогом деятельности данного метода считается четкое решение.

Минус АПП – это его временная сложность, площадь поиска увеличивается экспоненциально (скорость роста пропорциональна значению самой величины), по этой причине, если N никак не является существенно малым применяют эвристические и поисковые алгоритмы.

Метод ветвей и границ – это развитие АПП. Задача метода состоит в добавлении контроля критерия для ограничивающей функции, исходящего с знания проблемы, согласно который в конкретном уровне возможно прекратить создание этой ветви дерева перестановок. Он сохраняет все без исключения позитивные качества АПП, однако не достаточно годен для вопросов, в месте, где N не считается существенно небольшим.

Преимущества этого алгоритма – его можно распараллелить и данный алгоритм имеет точное решение.

1.3.2 Эвристические алгоритмы

Метод включения дальнего (МВД) состоит в следующем: города, предельно далёкие друг от друга, ни в коем случае никак не станут смежными в цепи. Данные 2 города и станут базовыми с целью последующего решения. Далее вновь находится вершина, предельно далёкая от вершин, ранее заключенных в цепь. Находится наименьшая сумма длин ребер среди обнаруженной вершиной и двумя смежных вершин в цепи, то что задает место в цепи обнаруженной вершине.

BV-метод базируется на рассмотрении существующего эталонного маршрута и его оптимизации. Решение условно заключается из двух стадий:

1) получение начального эталонного решения. Решение предполагает собой наилучшее решение с абсолютно всех решений, приобретенных на базе «жадного» метода;

2) оптимизация начального решения – заключается в модернизации приобретенного первоначального эталонного маршрута с поддержкой BV-

модификаторов.

Данный алгоритм имеет квадратичную сложность, который предоставляет приближенное решение и может быть распараллелен на 2-м этапе.

1.3.3 Поисковые алгоритмы

Генетический алгоритм(ГА) и муравьиный алгоритм (Ant Colony System – ACS) считаются «фаворитами» поисковых алгоритмов.

Наиболее наилучшим из числа поисковых алгоритмов считается ГА. Правда, он также обладает минусами, связанными с преждевременной сходимостью. ГА предоставляет приближенное решение задачи, он так же может быть распараллелен, блок схема генетического алгоритма изображена на рисунке 1.

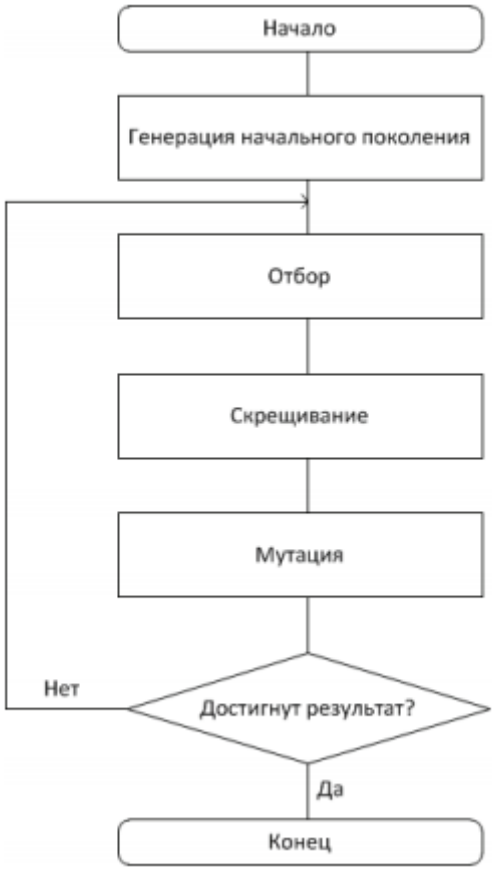

Рисунок 1 – Блок-схема работы генетического алгоритма.

Муравьиный алгоритм является развитием алгоритма «Система

муравьёв». Его основные отличия:

1) в функции подбора нового города очевидно существует баланс между использованием тех знаний которые накопились и исследованием новых решений;

2) при глобальном обновлении феромона его прибавление происходит только лишь к дугам, относящимся глобальному короткому пути;

3) до тех пор, пока муравьи находят решение, совершается локальное обновление феромона.

1.4 Положения муравьиного алгоритма для задачи коммивояжера и ее решения

Идея муравьиного алгоритма – моделирование реальной муравьиной колонии. Колония предполагает собой концепцию с весьма элементарными правилами автономного действия муравьев. Несмотря на простоту поведения любого муравья, действия всей колонии оказалось разумным. Таким образом, основой поведения муравьиной колонии предназначается низкоуровневая связь, вследствие которой, колония предполагает собой разумную систему, блок схема муравьиного алгоритма показана на рисунке 2.

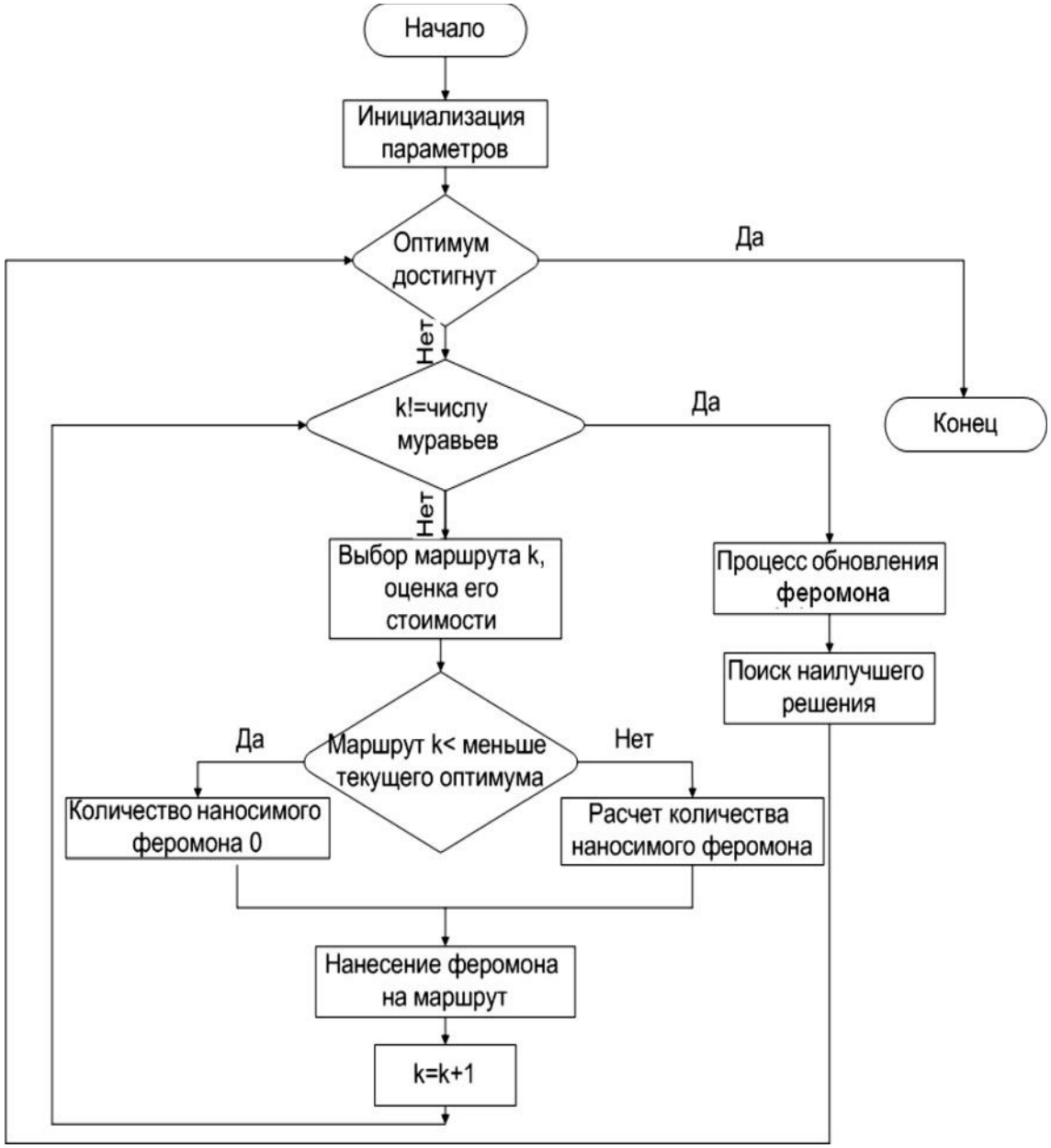

Рисунок 2 – Блок-схема муравьиного алгоритма

Феромон – специально химическое вещество, которое определяет взаимодействие между муравьями, которые откладывают данное вещество на пройденный путь. При выборе пути, муравей не берет только лишь факт короткого пути, но и учитывает опыт остальных муравьев, данную информацию, муравей получает из феромонов на каждом пути. Выходит, что феромоны определяют желание муравья сделать выбор между тем или другим маршрутом. Хотя при данном подходе нельзя избежать попадание в локальный оптимум. Эта возникшая проблема решается с помощью испарений феромонов.

Память муравья (список табу) — это пройденные муравьем городов, в которые зайти еще раз запрещено. При использовании списка табу муравей абсолютно точно не попадет в один город два раза. Этот список растет при прохождении пути и сбрасывается в старте любой итерации алгоритма. Тогда Ji,k— «список городов, которые еще необходимо посетить муравью k, находящемуся в городе i. Ji,k является дополнением к памяти муравья» [6, с. 72].

Видимость – это обратная расстоянию $\eta_{ij} = 1/D_{ij}$, где D_{ij} – расстояние между городами i и j. «Видимость представляет собой локальную информацию, характеризующую желание муравья посетить город j из города i – чем она выше, тем ближе город и тем сильнее желание его посетить».

Вероятность перехода муравья k в город j из города i определяется следующим соотношением:

$$P_{ij,k}(t) = \frac{r_{ij}(t)^{\alpha} * n_{ij}^{\beta}}{\sum_{i \in J_{ik}} r_{ij}(t)^{\alpha} * n_{ij}^{\beta}}, j \in J_{i,k}, \quad P_{ij,k}(t) = 0, j \notin J_{i,k}; \tag{4}$$

где α и β – параметры, которые задают вес следа феромона и видимость города при выборе следующего города.

При $\alpha = 0$ будет подобран наиближайший город, это сходится с жадным алгоритмом. Если $\beta = 0$, в таком случае функционирует только феромонное усиление, что влечет за собою быстрое нахождение маршрутов к одному субоптимальному решению.

Непосредственный выбор следующего города осуществляется по принципу «колеса рулетки»: для каждого муравья генерируется маршрут движения из последнего местонахождения случайным образом, с учётом вероятностей перехода. Затем для каждого из полученных маршрутов рассчитывается целевая функция общей протяжённости маршрута.

По завершении маршрута каждый муравей k откладывает на ребре (i, j) некоторое количество феромона, формула:

$$\Delta r_{ij,k}\ t\ =\ \begin{matrix}\frac{Q}{L_k(t)},\ i,j\ \in T_k\ t\ ;\\ 0,\ i,j\ \notin\ T_k\ t\ ;\end{matrix}, \tag{5}$$

где $Tk(t)$ – маршрут, пройденный муравьём k на итерации t, L$k(t)$ – длина этого маршрута, Q – некоторый регулируемый параметр.

Правило обновления феромона:

$$r_{ij}\ t+1\ =\ 1-p\ *r_{ij}\ t\ +\Delta r_{ij}\ t\ ;\Delta r_{ij}\ t\ =\ \ {}_{k=1}^{m}\Delta r_{ij,k}\ t\ , \tag{6}$$

где ρ – регулируемый параметр, принадлежащий отрезку [0;1], m – количество муравьёв.

С целью наглядно увидеть, как решается задача, изображенная на рисунке 3, которая находит минимальный путь в графе. Толщина линий отображает интенсивность прохождения муравьев на данном участке. В начале, вероятность перехода из одной вершины в другую вершину равна. С течением времени надобность, а значит и вероятность выбора самого короткого пути увеличивается, из-за того, что количество откладываемого феромона обратно пропорционально длине маршрута и задается формулой 6.

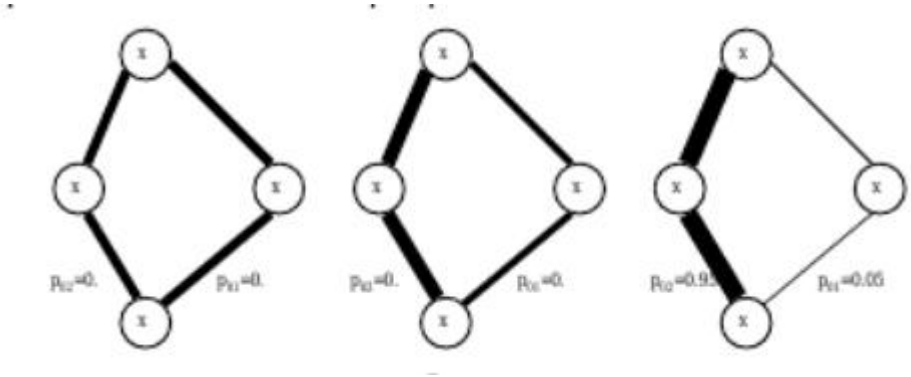

Рисунок 3 – Распределение вероятности

ГЛАВА 2 РАЗРАБОТКА ПРИЛОЖЕНИЯ РЕШЕНИЯ ЗАДАЧ КОММИВОЯЖЁРА С ПОМОЩЬЮ МУРАВЬИНОГО АЛГОРИТМА

2.1 Постановка требований к приложению

Была поставлена задача программной реализации муравьиного алгоритма решения задачи коммивояжёра с возможностью модификации исходных данных и регулируемых параметров.

Муравьиный алгоритм является самым эффективным из рассмотренных алгоритмов в параграфе 1.2, поэтому использовать его является предпочтительным.

Были выявлены следующие требования программы:

1) программа должна работать под любой операционной системой;
2) иметь оконный интерфейс;
3) иметь поле для рисования графа, для создаваемых графов должно быть предусмотрено отображение рёбер;
4) на нарисованном графе программа должна уметь искать оптимальный путь обхода всех вершин (решать задачу коммивояжера);
5) этапы решения задачи коммивояжера должны быть отображены на экран. Интервал отображения может быть задан автоматически, либо каждый шаг должен быть запущен вручную;
6) процесс решения задачи коммивояжера может быть прерван и изначальный граф может быть отредактирован путем добавления или изменения вершин;
7) должна быть возможность удалить существующий граф;
8) по завершении работы алгоритма программа должна показывать длину маршрута, найденного в ходе выполнения алгоритма;
9) пользователь должен иметь возможность удобно настраивать опции для муравьиный колонии, такие как количество муравьев, альфа параметр, бета параметр.

2.2 Обоснование выбора технических средств

Язык программирования является формальным языком, который указывает на набор инструкций, которые могут быть использованы для производства различных видов продукции. Языки программирования обычно состоят из инструкций для компьютера. Язык программирования, выбранный для разработки, может в значительной степени повлиять на процесс разработки, так и на конечный результат.

Для разработки алгоритма для решения задачи коммивояжера применяются различные языки программирования. Рассмотрим основные из них.

Язык JAVA. JAVA — это объектно-ориентированный язык программирования, который был разработан компанией Sun Microsystems и официально был выпущен 23 мая 1995 года. Проектировщики JAVA пытались разработать язык, который могли бы быстро изучить программисты. Также они хотели, чтобы язык был знаком большинству программистов, для простоты перехода. Отсюда, в JAVA проектировщиками было удалено множество сложных особенностей, которые существовали в C и C++. Особенности, такие как манипуляции указателя, перегрузка оператора и т.д. в JAVA не существуют.

В JAVA всё может быть объектом. Так основное внимание уделяется свойствам и методам, которые оперируют данными в нашем приложении и нет концентрации только на процедурах. Свойства и методы вместе описывают состояние и поведение объекта.

Отличительными особенностями этого языка являются:

1) написанные на языке JAVA приложения являются кроссплатформенными, т.е. могут выполняться на различных платформах и операционных системах;

2) за счет встроенной в систему многопоточности, в графическом интерактивном приложении получается достичь максимально быстрой реакции на ввод пользователя, а, следовательно, и высокой производительности;

3) время разработки приложений сокращено за счет того, что система построена на основе интерпретатора, т.е. скомпилированный единожды код может интерпретироваться на любой платформе;

4) благодаря механизму обнаружения возможных ошибок на этапе компиляции, динамической проверке, а также исключению ситуаций, подверженных ошибкам язык JAVA является довольно надежным языком программирования;

5) благодаря механизму сборки мусора память автоматически освобождается, и исключается возможность появления ошибок, связанных с неправильным использованием памяти, что значительно упрощает для разработчика процесс написания программы, который может меньше уделять внимания техническим тонкостям, и больше – модели бизнес-процессов.

Язык C++ был разработан сотрудником компании Bell Labs Бьерном Страуструпом в начале 1980-х годов. На сегодняшний день C++ является одним из популярнейших языков программирования и довольно часто используется в системном программировании. На нем пишутся высокоэффективные приложения, в том числе операционные системы, драйвера, приложения для встраиваемых систем, высокопроизводительных серверов, а также развлекательные приложения (игры). Он сочетает в себе свойства как высокоуровневых, так и низкоуровневых языков, а также является мультипарадигмальным языком программирования, т.е. он поддерживает объектно-ориентированное, процедурное, обобщенное и метапраграммирование. Одним из принципов разработки было сохранение совместимости с языком C, от которого и был унаследован синтаксис языка.

Особенностями языка C++ являются:

1) непосредственная и всесторонняя поддержка множества стилей программирования (процедурное программирование, абстракция данных, объектно-ориентированное программирование и обобщённое программирование);

2) совместимость с языком C, что позволяет легко переходить с одного языка программирования на другой;

Язык C# появился в 1998-2001 годах. C# относится языкам с C-подобным синтаксисом, из них синтаксис данного языка наиболее близок к C++ или JAVA. Он испытал влияние множества языков программирования, таких как C++, Pascal, Модула, Smalltalk, Delphi.

Особенности языка:

1) C#, развивается быстрее, относительно других языков;

2) C# определён стандартами ECMA и ISO, которые задают синтаксис языка;

3) C# тоже является кроссплатформенным стандартом. Изначально этот язык ориентировался на разработку программ для платформы Windows. Но в дальнейшем появились реализации для других платформ. Все же платформы, отличные от Windows, не поддерживаются официальным производителем.

Определение основных критериев

Для выбора языка программирования для системы управления подписками необходимо определить и учитывать различные критерии, которые могут повлиять на процесс создания данной системы и результат.

При выборе языка программирования необходимо учесть, что готовый проект будет являться серверным приложением и иметь базу данных, поэтому язык должен поддерживать доступ к базам данных.

Ввиду того, что передача информации будет осуществляться по сети, язык должен быть распределенным и обеспечить безопасность.

Готовую систему будет проще модернизировать, улучшать и поддерживать, если она будет составлена из определенного количества модулей, поэтому язык должен поддерживать создание модульных программ.

На стоимость законченной системы в значительной степени может повлиять такой критерий, как наличие свободной лицензии.

Сравнение языков по основным критериям

Лицензия

По этому критерию JAVA можно признать лидером, т.к. она основана на более открытой культуре с более высокой конкурентностью фирм в разных областях функциональности. Популярности JEE также помогает, что Oracle предлагает бесплатный комплект разработки, SDK.

C#, среда выполнения CLI, большинство библиотек могут свободно реализовываться без лицензии.

Модульность

JAVA является объектно-ориентированным языком, и как следствие можно создавать модульные программы, исходный код которых может использоваться многократно, что значительно упрощает доработку и дальнейшее улучшение уже готовых программ. Благодаря этому можно определить и такие критерии, как масштабируемость и гибкость.

Так же под этот критерий подходят языки C++, C# т.к. они так же являются объектно-ориентированными.

Надежность

JAVA учел причины написания ошибок в других языках и решил данные проблемы, посредством некоторых ограничений и проверок синтаксиса. В отличии от других рассматриваемых языков, программист, разрабатывающий на языке JAVA избавлен от неверных указателей, неправильного распределения и утечки памяти.

Безопасность

JAVA предназначен для использования в сети или в определенной среде. Специалисты Oracle продолжают находить ошибки в механизмах безопасности всех версий языка JAVA, делая таким образом его безопасным языком.

Итоги сравнения языков программирования приведены в таблице 1.

Таблица 1 – Сравнительная таблица языков программирования

	JAVA	C++	C#
Лицензия	+	-	+
Модульность	+	+	+
Надежность	+	-	-
Безопасность	+	+	+
Итог:	4	2	3

Таким образом, в качестве языка программирования для данной работы выбран язык JAVA.

2.3 Разработка архитектуры ПО

Проанализировав требования поставленной задачи в качестве основной платформы разработки был выбран язык java, так как он имеет возможность запускать исполняемые файлы на любой ОС и выводить на экран сложные интерфейсы.

Была разработана высокоуровневая архитектура приложения, изображенная на рисунке 4.

Javafx является графической библиотекой, позволяющей рисовать сложные фигуры и взаимодействовать с ними, поэтому он был выбран в качестве основной библиотеки.

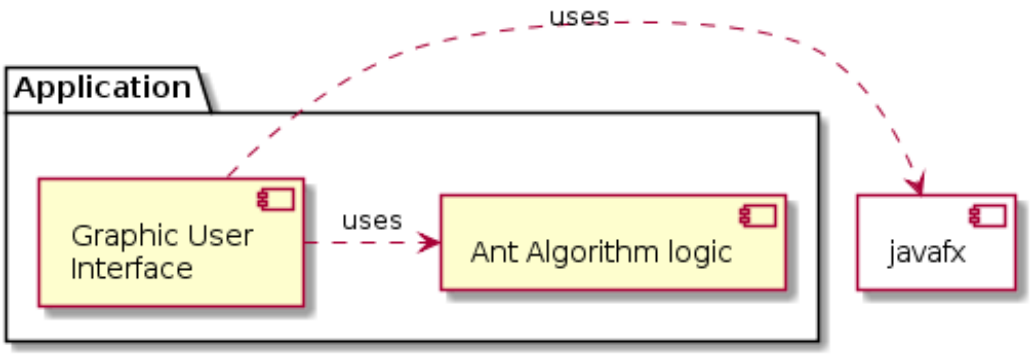

Рисунок 4 – Диаграмма компонентов

Алгоритм работы приложения представлен на рисунке 5.

Ant Algorithm logic – это абстракция выполняющая задачу коммивояжёра с помощью муравьиного алгоритма представленного на рисунке 2.

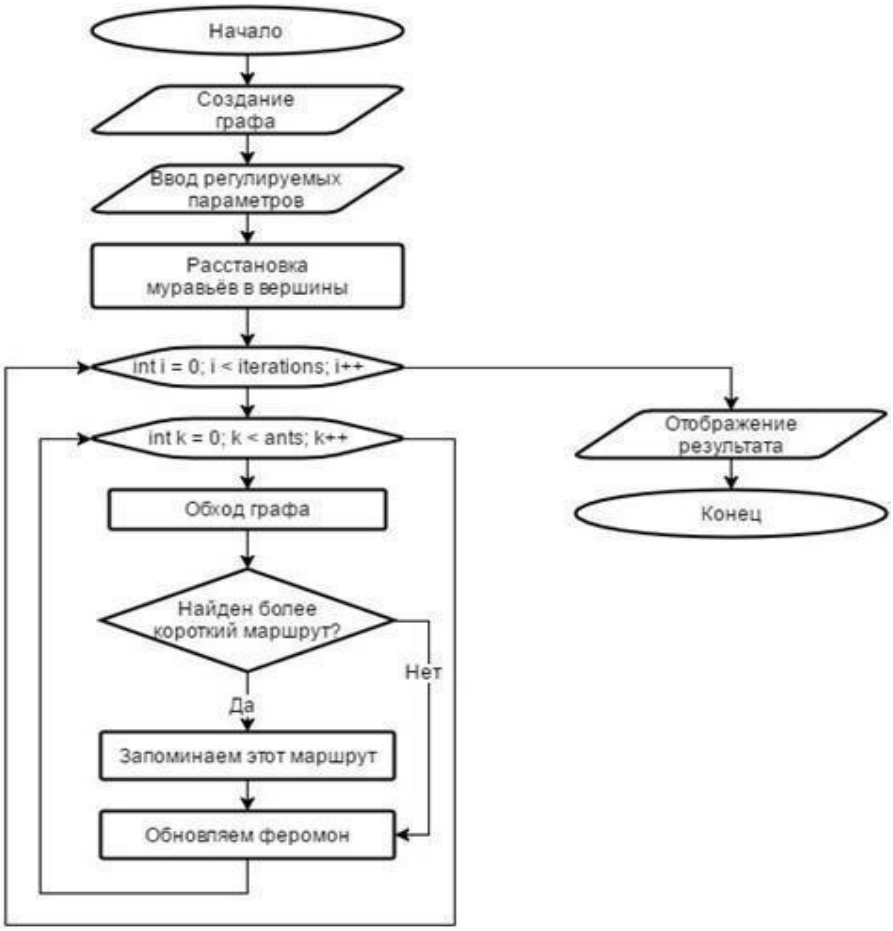

Рисунок 5 – Блок-схема описывающую работы приложения

Диаграмма последовательности случая решения задачи коммивояжера изображена на рисунке 6.

Случай поэтапного решения задачи коммивояжера с помощью приложения представлен на рисунке 7.

Автоматическое решения задачи коммивояжера изображен на рисунке 7.

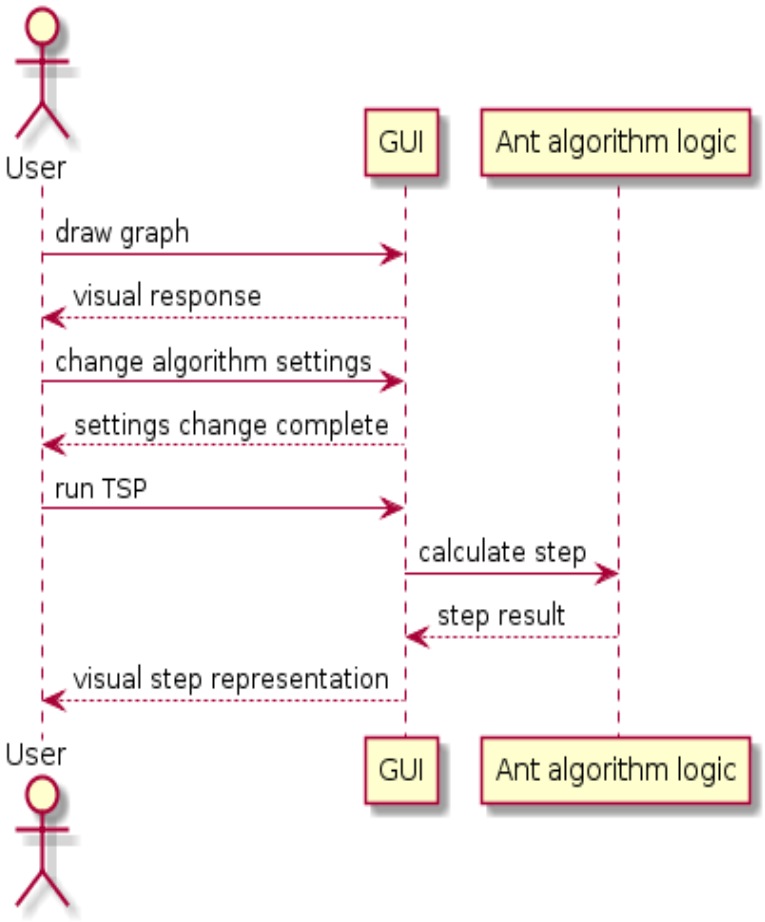

Рисунок 6 – Диаграмма последовательности решения задачи коммивояжера

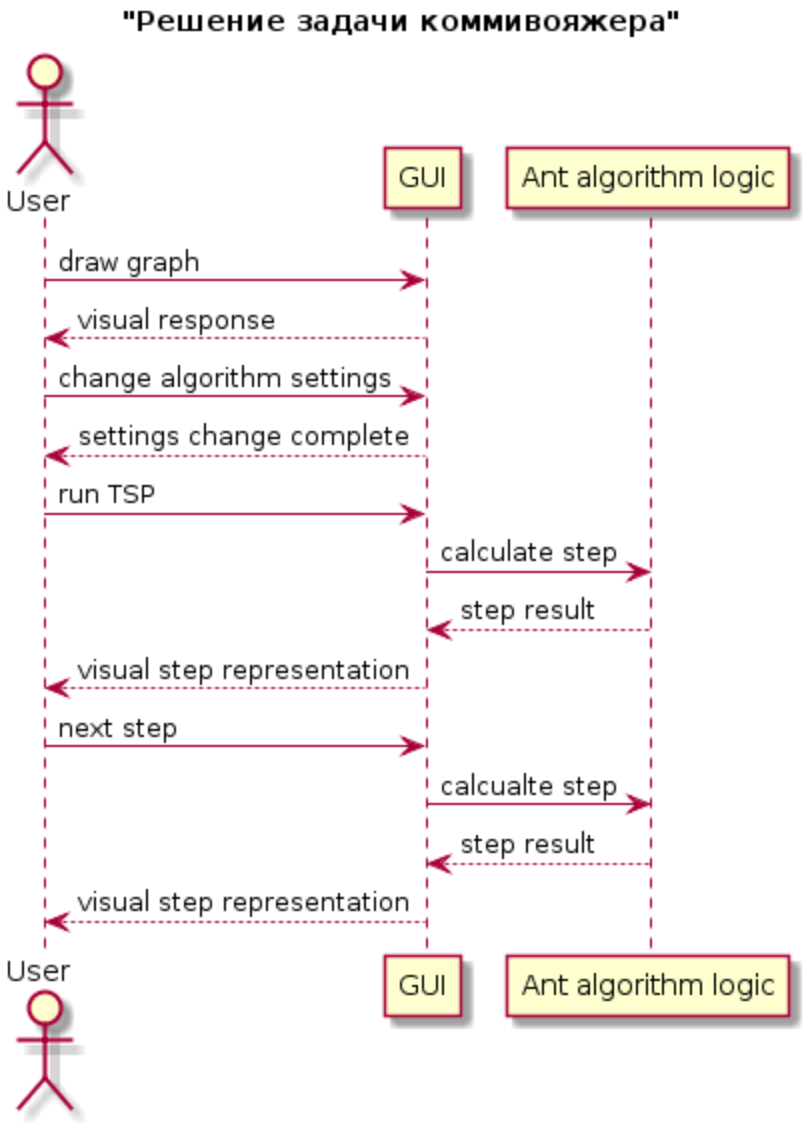

Рисунок 7 – Диаграмма последовательности поэтапной последовательности задачи коммивояжера

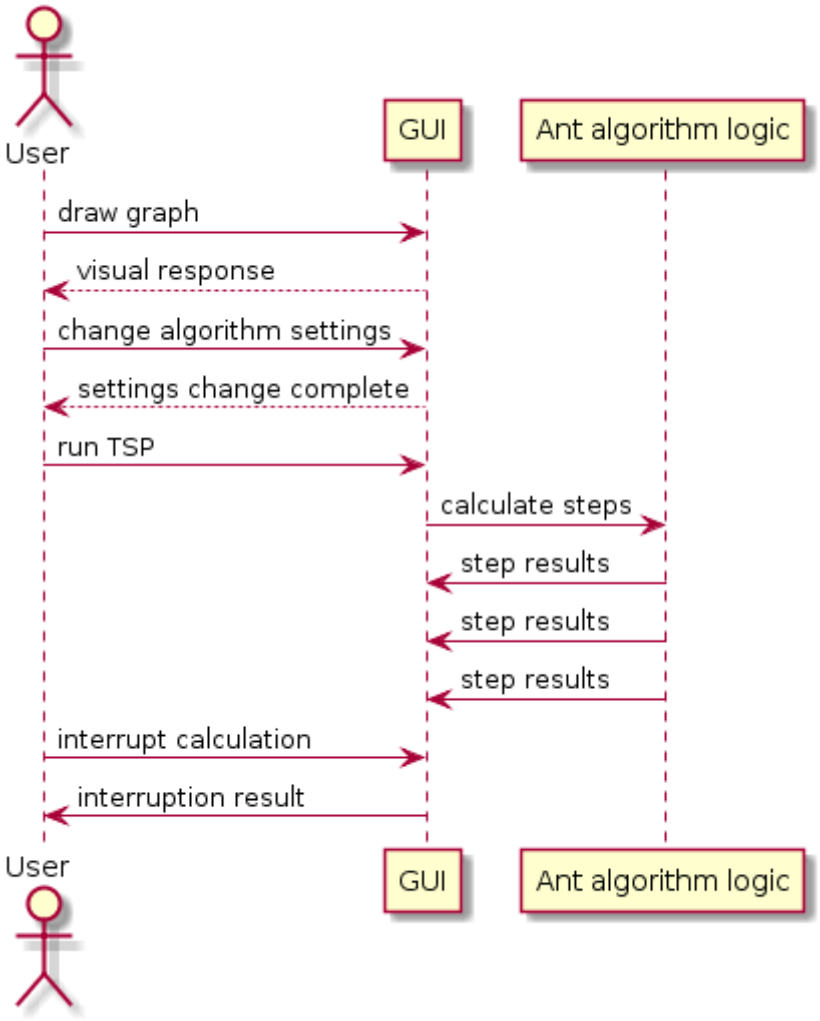

Рисунок 8 – Диаграмма компонентов автоматического решения задачи коммивояжера

2.4 Реализация алгоритма

Для вычисления оптимального маршрута в задаче коммивояжёра в программе применяется алгоритм оптимизации подражанием муравьиной колонии.

Были выделены и разработаны следующие Java-классы. Проведем краткий обзор по самым важным составляющим программы:

1) класс AntColonySystemTSP – который реализует работу муравьиного алгоритма для решения задачи коммивояжера;

2) интерфейс AntOptimizationTSP – который описывает алгоритм для решения задачи коммивояжера;

3) класс SingleAnt – который описывает поведение муравья;

4) класс GraphicalSolverFrame – который реализует графическое

отображение решения;

5) класс PointCord – который реализует отображение точек в графе;

6) класс Main – с которого начинается работа программы.

Диаграмма классов показана в приложении А.

Временная сложность алгоритма оценена как $Q(t * m * n^2)$, где *t* – число итераций, *m* – количество муравьёв, *n* – количество вершин в графе.

При испарении феромона необходимо учитывать, что уровень феромона на рёбрах не должен достигать нулевого уровня, иначе переход по таким рёбрам будет невозможен и полученное решение будет субоптимальным.

Механизм обновления феромонов показан следующим правилом, указанным на рисунке 9.

```
maxPheromone = 1 / ro * ((double)coumputeTourLenght(getBestSoFarTour()));
minPheromone = maxPheromone * (1 - Math.pow(0.05,1.0/((double)noNodes)))/(noNodes);
if(stagnationiterations > maxStagnaterations) {
reinitPheromones();
stagnationiterations = 0;
}
```

Рисунок 9 – Обновление феромона

Механизм перезапуска феромонов показан на рисунке 10.

```
public void reinitPheromones(){
    int i,j;
    for(i = 0; i < noNodes; i++)
       for(j = 0; j < noNodes; j++)
          pheromone[i][j] = tau0;
    for(i = 0; i < noNodes; i++)
       pheromone[i][i] = 0;
    maxPheromone = tau0;
    minPheromone = maxPheromone*(1-Math.pow(0.05, 1.0/((double)noNodes)))/(noNodes);
  }
```

Рисунок 10 – Перезапуск феромонов

Механизм реализации муравья в алгоритме, реализуется на поведении реального муравья, муравей ходит в случайном порядке через города, прокладывая феромоны.

Феромоны оставляются муравьями, после прохода через маршрут, там, где след от феромона больше, муравьи начинают ходить чаще, тем самым с

каждой итерацией муравьиного алгоритма, пути на которых меньшее количество феромонов теряют популярность среди муравьев и значение феромонов в программе обнулятся на том или ином маршруте.

На основе блок-схемы изображенной на рисунке 5, которая описывает работу программы, составлялся программный код на языке Java в классе GeneticAntSystemTSP. Фрагмент кода представлен на рисунке 11.

```
public GeneticAntSystemTSP(int noNodes, int noAnts, double alfa,
            double beta, double ro,
            int maxStagnationIterations, int geneticIterations){
    this.noNodes = noNodes;
    this.noAnts = noAnts;
    this.alfa = alfa;
    this.beta = beta;
    this.ro = ro;
    this.iterations = 0;
    this.stagnationIterations = 0;
    this.maxStagnationIterations = maxStagnationIterations;
    this.geneticIterations = geneticIterations;
    dist = new int[noNodes][noNodes];
    pheromone = new double[noNodes][noNodes];
    choiceInfo = new double[noNodes][noNodes];
    ants = new SingleAnt[noAnts];
    for(int i = 0; i < noAnts; i++){
      ants[i] = new SingleAnt(noNodes);
    }
    ga = new GeneticAlgorithmTSP(noAnts, noNodes, 0.1, false, false);
    bestSoFarTour = new int[noNodes+1];
  }
```

Рисунок 11 – Алгоритм в программном коде

2.5 Разработка приложения

Решение поставленной задачи требует наличия информации о количестве вершин и расстояниях между ними. Для реализации получения информации был выбран способ, который позволяет создать граф непосредственно в программе.

Граф может быть создан пользователем вручную. По умолчанию дан пустой граф.

Длины рёбер графа вычисляются автоматически, исходя из координат вершин на плоскости.

С целью сравнения эффективности применения муравьиного алгоритма графах дополнительно реализован генетический алгоритм, позволяющий получить точное решение для задачи коммивояжёра. Для работы этого алгоритма достаточно данных о количестве вершин и расстояниях между ними.

Полученная информация представляется в программном продукте пользователю в виде графа с обозначенным маршрутом обхода вершин.

Для обеспечения демонстрации поведения муравьиного алгоритма в различных условиях пользователь программы имеет возможность регулировать некоторые параметры, влияющие на работу муравьиного алгоритма.

К этим параметрам относятся:

1) количество муравьев;
2) альфа параметр;
3) бета параметр;
4) рейтинг испарения феромонов.

2.5.1 Этапы разработки приложения

1. Проектирование. Определяются цели и задачи, способы их решения, а также определяется структура данных и язык программирования, на котором будет написано приложение.

2. Разработка алгоритма. Разработка алгоритма который будет реализован в программе.

3. Создание интерфейса. В программную среду разработки вводятся необходимые управляющие элементы: кнопки, текстовые поля, флажки, переключатели и другие элементы.

4. Отладка. Все управляющие элементы связываются программным кодом и путем ввода конкретных значений происходит проверка работоспособности кода и отлавливание возможных ошибок.

5. Заключительный этап. Идет компиляция кода и создание дистрибутива. Компиляция – процесс перевода программного кода в машинный язык, понятный каждому компьютеру. Здесь же идет подключение необходимых программных библиотек для полной работоспособности

приложения.

2.6 Интерфейс приложения

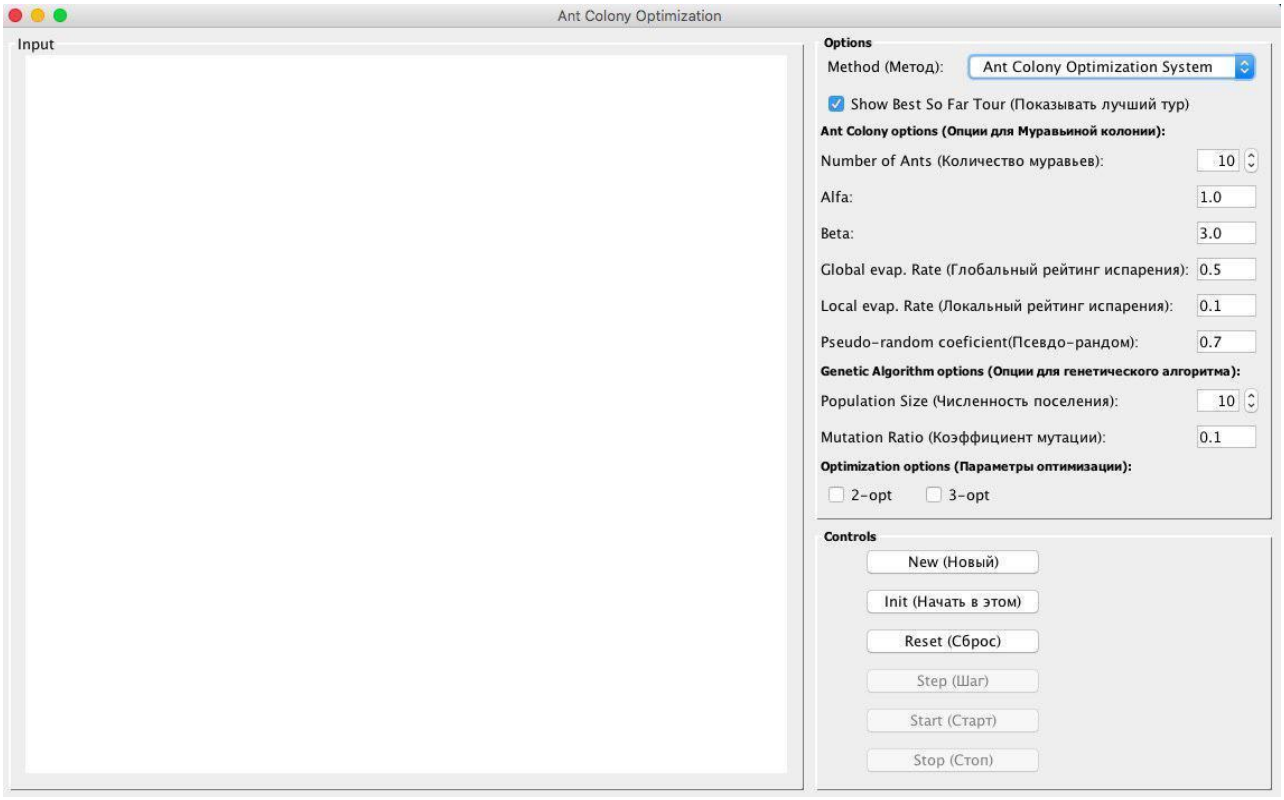

Рисунок 12 – Главное окно приложения

В интерфейсе приложения предусмотрены следующие компоненты:

1) панель, на которой отображаются граф и полученный маршрут;

2) текстовые поля ввода параметров: количество муравьев, Alfa, Beta, рейтинг испарения;

3) переключатель для показывания лучшего пути;

4) функциональные кнопки для работы приложения: новый граф, начать в этом графе, сброс, шаг, старт и стоп;

5) информационные ветки, показывающие количество итераций и пройденный путь.

2.7 Описание работы приложения

При запуске программы открывается главное окно, представленное на рисунке 12.

2.7.1 Создание графа

В программе предусмотрен способ создания графа в окне программы.

Пользователь самостоятельно размещает вершины, кликая мышкой по панели в желаемых местах. На месте клика создаётся вершина. Пример выбора графов изображен на рисунке 13.

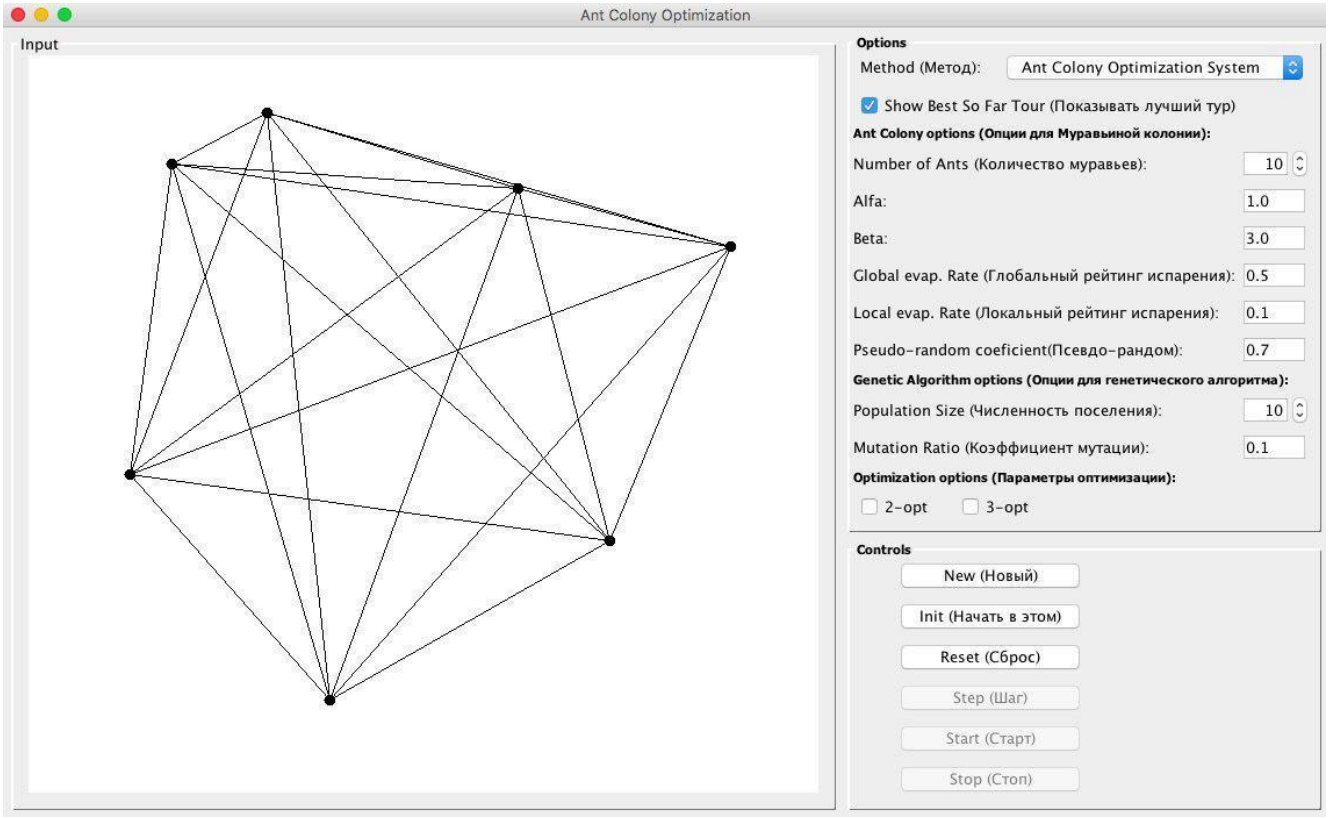

Рисунок 13 – Создание графов

После указывания графов в приложении выбираем нужные параметры в правой вкладке “Options” изображенной на рисунке 14.

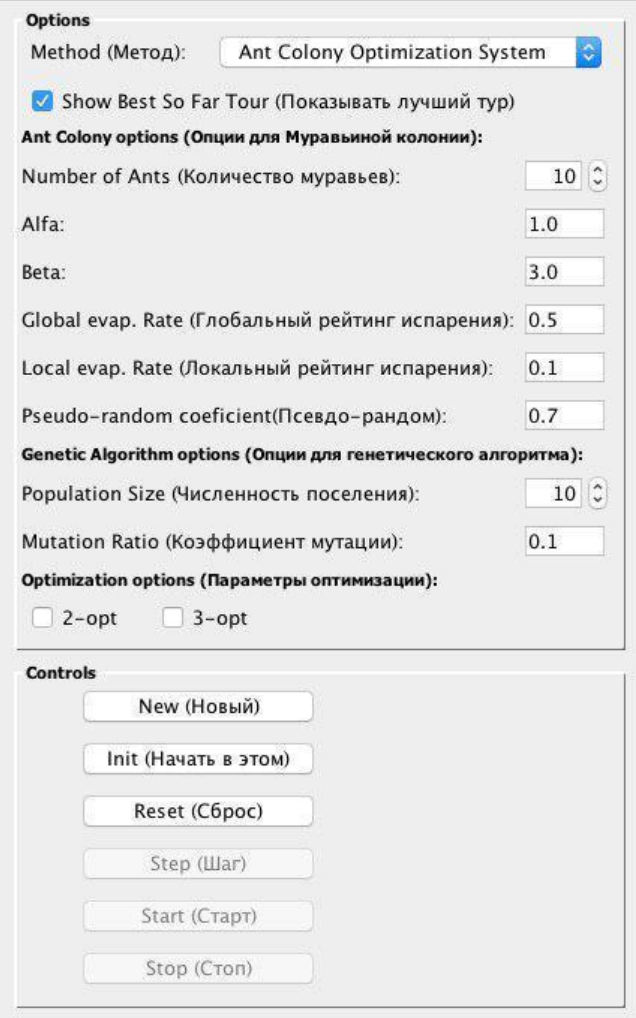

Рисунок 14 – Выбор параметров в приложении

2.7.2 Работа приложения

После размещения графов и выбора параметров в приложении, нужно нажать кнопку “Init(начать в этом графе)”, которая обозначает, то что начать работу в данном графе или же нажать кнопку “New(Новый)”, что бы стереть уже построенный граф и создать новый и в окне приложения. После нажатия кнопки “Init”, будет предложен выбор между кнопками “Step(Сделать шаг)” или “Start(Старт)”, работа приложения после нажатия кнопки “Init” изображена на рисунке 15.

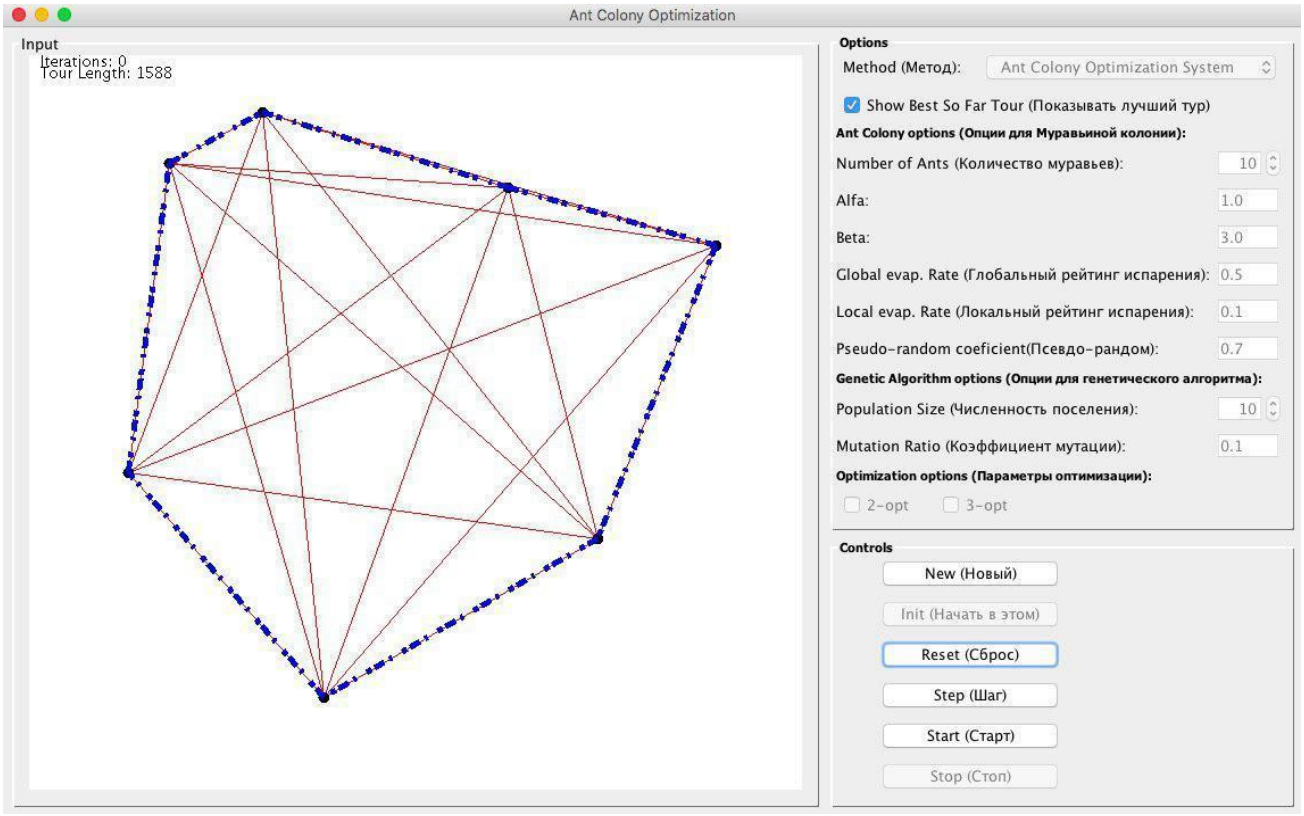

Рисунок 15 – Работа приложения после того, как нажать кнопку “Init”(Начать в этом)”

После того как мы нажмем кнопку “Step(Шаг)”, мы перейдем на следующую по счету итерацию, работа кнопки изображена на рисунке 16.

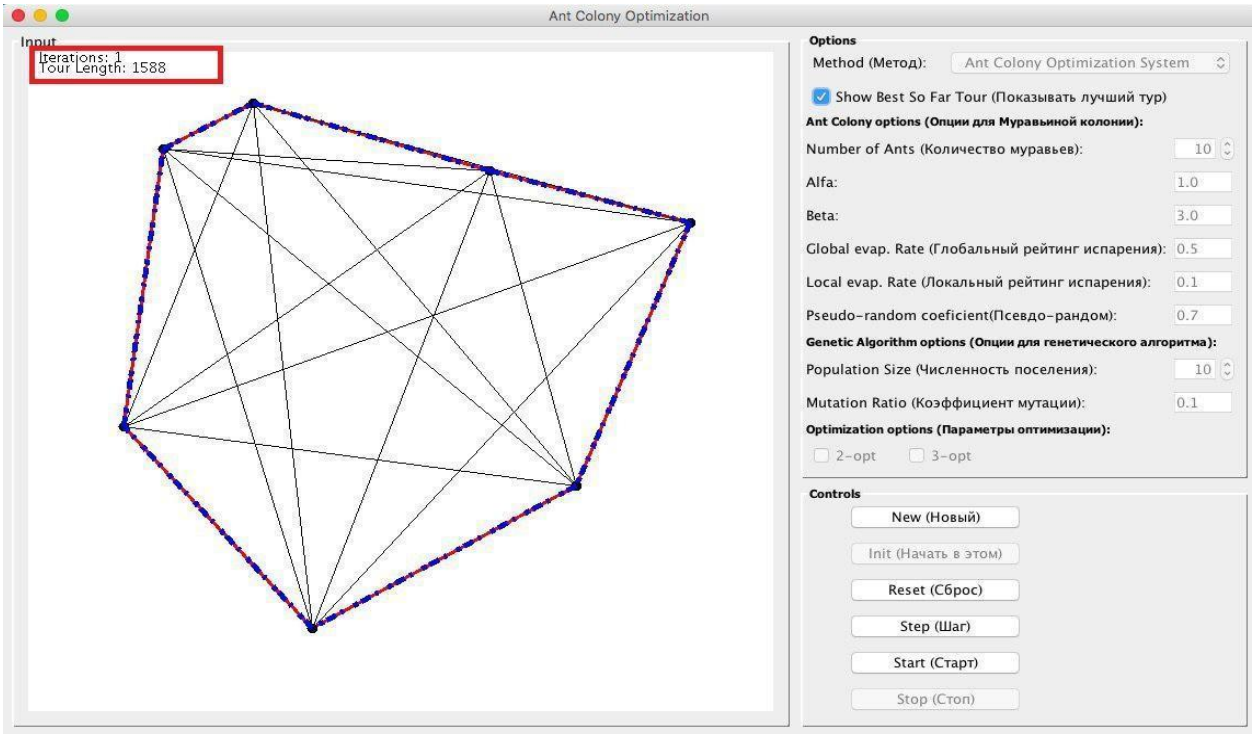

Рисунок 16 – Работа приложения после нажатия кнопки “Step”

Для того что бы пользователю перейти к автоматическому решению задачи коммивояжера нужно нажать кнопку “Start(Старт)”, тогда программа автоматически начнет высчитывать оптимальный путь. Оптимальный путь показан в окне “Input” и называется “Tour Length”. Работа приложения после нажатия кнопки “Start” показана на рисунке 17.

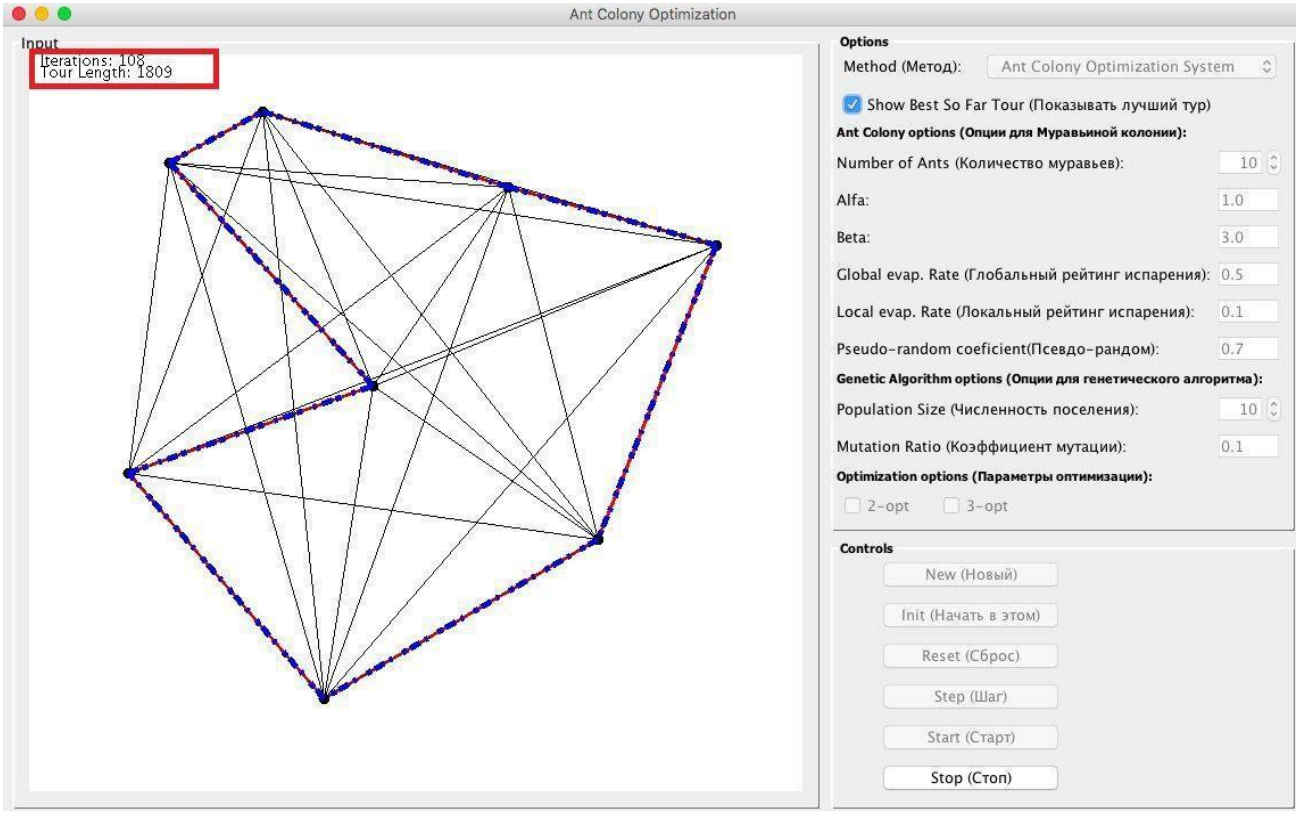

Рисунок 17 – Работа приложения после нажатия кнопки “Start”

Для того, чтобы остановить работу приложения в любой момент необходимо нажать кнопку “Stop”, кнопка остановит приложение на текущей итерации и даст выбор между нажатием кнопки “Reset(Сбросить)”, “Step(Шаг)”, “Start(Старт)” и “New(Новый)”.

В приложении присутствует функция, графически показывать лучший тур который находится в данный момент, чтобы включить отображение данной функции, необходимо поставить галочку в соответствующем меню “Show Best So Far Tour”, изображенная на рисунке 18.

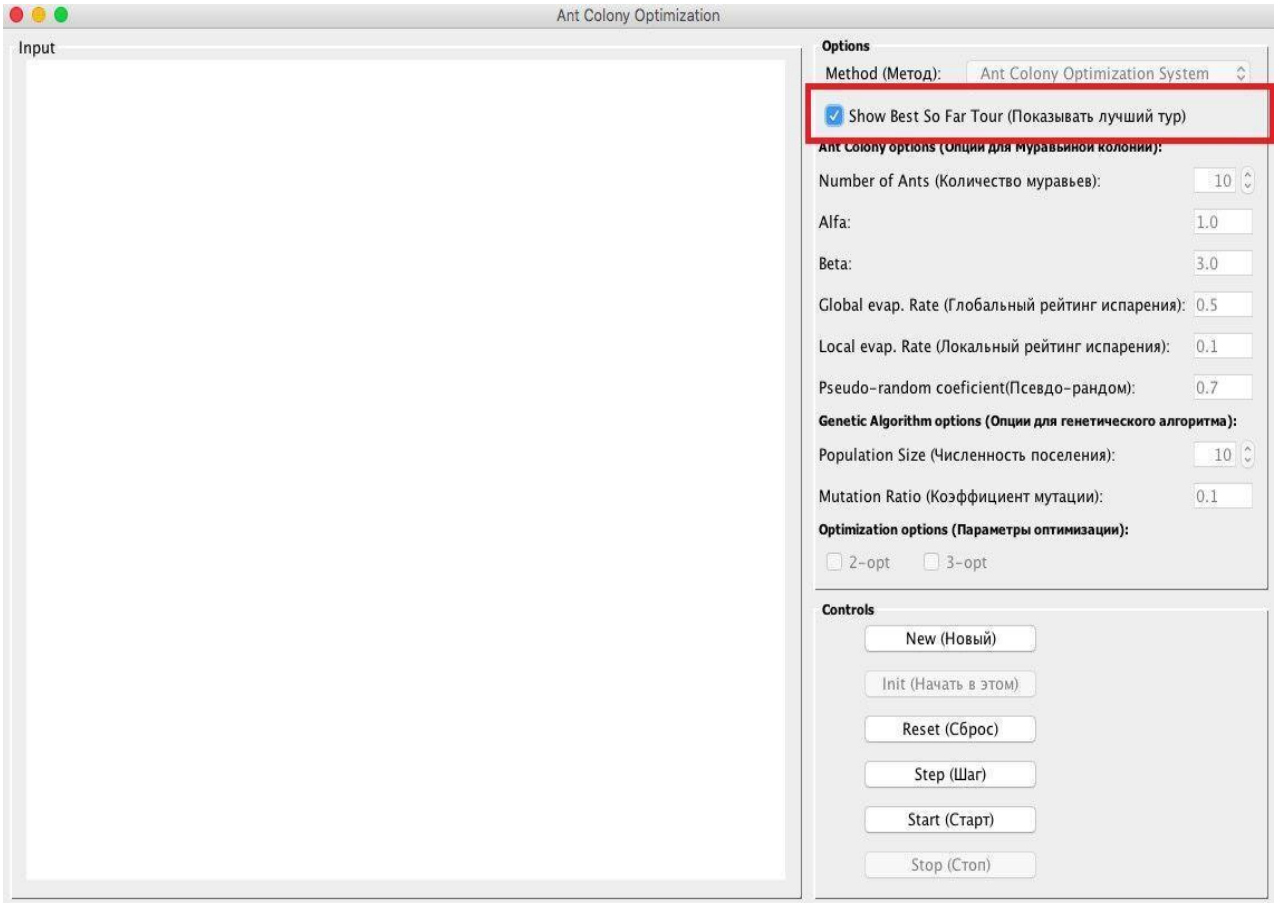

Рисунок 18 – Пункт меню “Show Best So Far Tour”

2.7.3 Получение результатов

В результате выполнения предшествующих шагов будет создан некоторый граф, с индивидуальными параметрами.

После задания графа и выставления регулируемых параметров пользователю необходимо выбрать алгоритм и нажать на соответствующую кнопку.

После завершения работы алгоритмов автоматически просматриваются результаты работы приложения.

2.8 Проведение вычислительного эксперимента

Для проведения вычислительного эксперимента были использованы исходные данные из работы [12], из данной работы была взята таблица под номером 3 и была модифицирована и использована в данной работе в таблице 2.

Таблица 2 демонстрирует зависимость времени поиска и найденного

оптимального маршрута от значений регулируемых параметров на примере тестовой задачи, изображенной на рисунке 19.

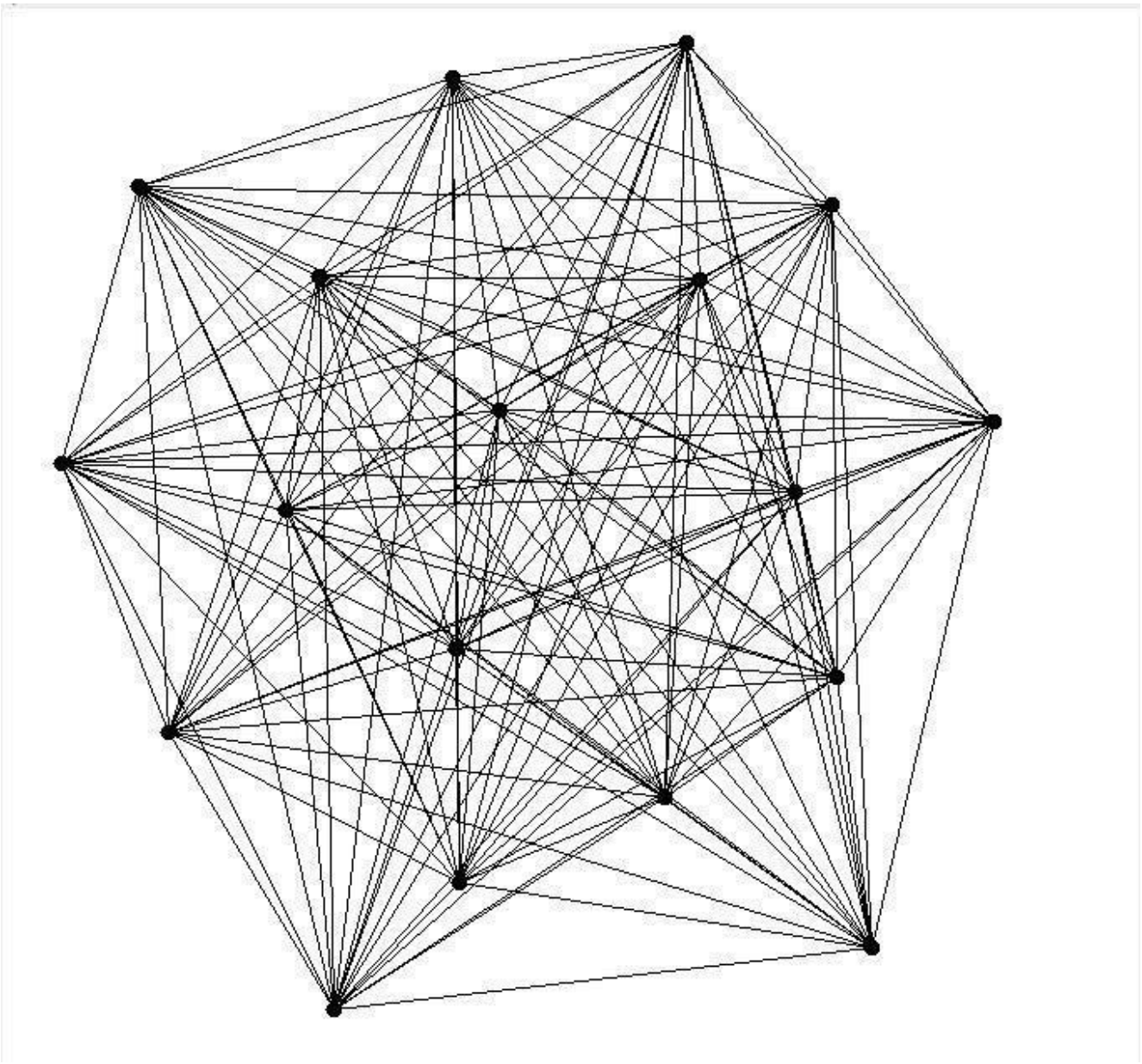

Рисунок 19 – Графы тестовой задачи

Таблица 2 – Применение муравьиного алгоритма для решения задачи коммивояжера муравьиным алгоритмом с различными параметрами к тестовой задаче.

№	α	β	Муравьи	Количество итераций	Количество городов	Q	Результат	Время (сек)
1	1	1	20	100	17	100	2567	6,43
2	1	1	20	980	17	100	2564	51,54
3	0	1	20	20	17	100	3200	85,52
4	1	0	20	1000	17	100	3095	74,91
5	1	1	20	950	17	100	2680	12,60
6	1	1	20	900	17	600	2657	38,86
7	1	1	20	500	17	600	2564	82,54
8	1	1	50	550	17	600	2570	22,38
9	1	1	50	50	17	600	2531	6,74
10	1	1	50	60	17	600	2610	12,5
11	2	1	50	70	17	600	2636	19,10
12	2	1	50	80	17	600	2846	51,88
13	2	1	50	1	17	600	3219	1,89
14	2	1	1	1	17	600	3254	0,55
15	2	1	130	200	17	580	2601	120,89
16	1	1	60	400	17	580	2692	32,64
17	2	1	60	600	17	580	2720	51,80
18	0	0	60	250	17	580	2763	55,02

В соответствии с полученными результатами можно сделать следующие выводы:

1) при неизменном числе вершин *n* и изменении количества муравьёв *m* и итераций *t*, временные затраты изменяются согласно приведённой оценке

временной сложности Q ($t * m * n^2$);

2) изменение параметров m и t влияет на временные затраты, но не гарантирует получения лучшего результата. Это происходит в силу вероятностного подхода к выбору вершин муравьями;

3) изменение феромонного коэффициента Q не оказывает значительного влияния на нахождение оптимального решения.

2.9 Динамика нахождения решений

Для изучения динамики нахождения решений муравьиным алгоритмом для решения задачи коммивояжера на каждой его итерации сравниваются длины пути, найденного на данной итерации, и лучшего из найденных путей. Динамика поиска минимального пути показана на рисунке 20.

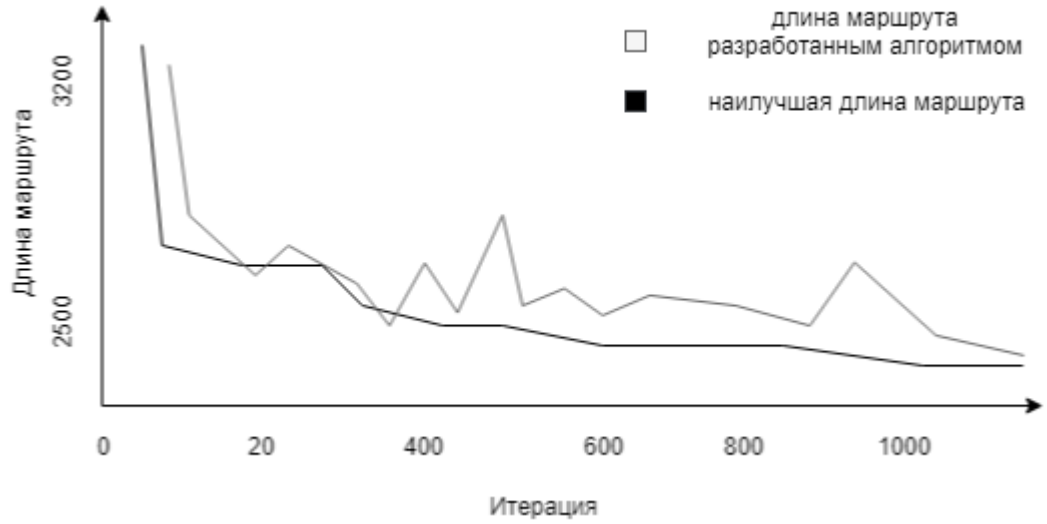

Рисунок 20 – Динамика поиска минимального пути задачи коммивояжера с помощью муравьиного алгоритма

Серой линией на рисунке 18 обозначены значения длин маршрутов, полученных на каждой итерации, чёрной – наилучшие решения, найденные в ходе работы алгоритма. Несовпадение этих линий говорит о том, что муравьиный алгоритм на каждой итерации находит новое решение.

Для подтверждения вывода возьмём среднеквадратичное несоответствие длин маршрутов, обнаруженных муравьями в протекающей итерации. Эти значения, показанные на рисунке находятся выше нуля, что говорит о поиске разных маршрутов муравьями, показанных на рисунке 19.

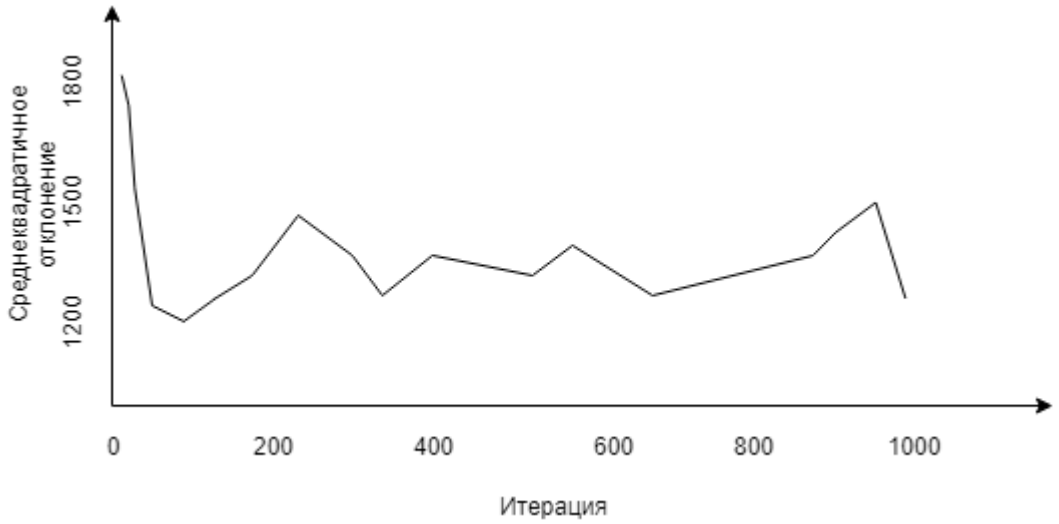

Рисунок 21 – Разброс решений

На рисунке 21, изображен разброс решений при использовании муравьиного алгоритма для решения задачи коммивояжера. На рисунке можно увидеть то, что среднеквадратичное отклонение никак не зависит от количества итераций.

ЗАКЛЮЧЕНИЕ

В рамках бакалаврской работы все цели и задачи выполнены. Было рассмотрены муравьиные алгоритме в задаче коммивояжера, был выделен объект исследования, проанализирована предметная область, определены цели и задачи работы. Проведен анализ существующих алгоритмов для решения задачи коммивояжера и выявлен наиболее подходящий для реализации цели выпускной квалификационной работы.

При анализе задачи коммивояжера было выявлено, что для решения задач бакалаврской работы потребуется написать свою программную реализацию муравьиного алгоритма, а не использовать существующие, так как существующие разработки были сделаны для других конкретных целей, поэтому и наша разработка будет носить такой же характер.

Был спроектирован интерфейс пользователя и использован муравьиный алгоритм.

Алгоритм и интерфейс были реализованы в среде разработки IntelliJ IDEA с использованием языка программирования Java. В ходе выполнения работы реализованы следующие функции:

- ввод пользователем параметров для алгоритма;
- отображение результатов на каждой итерации;
- реализована визуализация муравьиного алгоритма.

Затем, муравьиный алгоритм был применен к задаче коммивояжера, для того что бы найти оптимальный путь прохождения в той или иной задаче.

Итогом бакалаврской работы является разработанный алгоритм и приложение которое решает задачу коммивояжера с помощью муравьиного алгоритма, позволяющая, вручную вводить график, проходить по каждой итерации в задаче. Работа соответствует целям и задачам, которые были поставлены.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Майника Э. Алгоритмы оптимизации на сетях и графах // М.: Мир, 1981. – 323 с.
2. Макконнелл, Дж. Основы современных алгоритмов // М.: Техносфера, 2004. – 368 с.
3. Новиков Ф.А. Дискретная математика для программистов // М.: Питер, 2002. – 368 с.
4. Сергеев С.И. Задача коммивояжера. Вопросы теории / И. И. Меламед, И. Х. Сигал // Автомат. и телемех. 1989. – 33 с.
5. Машин Т. JavaFX 2.0. Разработка RIA-приложений // БХБ-Питербург 2012. – 320с.
6. Корте Б. Комбинаторная оптимизация. Теория и алгоритмы // Московский центр непрерывного математического образования 2015. – 720 с.
7. Гуц А.К. Математическая логика и теория алгоритмов // URSS 2016. – 128 с.
8. Штовба С.Д. Муравьиные алгоритмы // [Журнал] Exponenta Pro. 2003. – 49-75с.
9. Панченко Т. В. Генетические алгоритмы: учебно-методическое пособие // Т.В. Панченко. – Астрахань: Изд. дом «Астраханский университет», 2007. – 87 с.
10. Хорстманн К.С. Java. Библиотека профессионала. Том 1. Основы / Гари Корнелл. // Изд-во Addison-Wesley, 2012. – 864 с.
11. Босуэлл Д. Читаемый код, или Программирование как искусство. Полное руководство. // СПб.: Изд-во Питер, 2012. – 208 с.
12. Евтушенко Д.А. Программная реализация муравьиного алгоритма решения задачи коммивояжера. // Тюмень, 2016. – 46-48 с.
13. Муравьиные алгоритмы / Хабр [Электронный ресурс] : Режим доступа: https://habrahabr.ru/post/105302/ – Загл. с экрана – Яз рус., англ – (Дата обращения: 21.05.2018)

14. S. Lin, Computer solutions of the traveling salesman problem [Электронный ресурс] / S. Lin // Bell System Technical Journal. – 1965. – Pages 2245–2269.

15. G.A. Croes, A Method for Solving Traveling-Salesman Problems [Text] / G.A. Croes // Operations Research. – 1958. – Pages 791-812.

16. M. Dorigo. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem // [Журнал] IEEE Transactions on Evolutionary Computation Vol. 1, №1, 1997. – Pages 53-66.

17. A modified ant system to achieve better balance between intensification and diversification for the traveling salesman problem [Text] / Yuzhe Yan, Han-suk Sohn, German Reyes // Dept. of Industrial Engineering, New Mexico State University, Las Cruces, NM 88003, USA – 2016. – Pages 1-21.

18. Solving the Travelling Salesman Problem Using the Ant Colony Optimization [Text] / Ivan Brezina Jr. Zuzana Čičková // University of Economics in Bratislava Faculty of Economic Informatics Department of Operations Research and Econometrics, Slovakia – 2011. – Pages 1-13.

19. Overview of Java application configuration frameworks [Text] / Victor S. Denisov // MGU Lomonosov University. – Russia – 2013. – Pages 1-12.

20. Simplifying and improving ant-based clustering [Text] / Swee ChuanTan, Kai MingTing Shyh WeiTeng // Monash University, Faculty of Information Technology. – Australia – 2011. – Pages 46-48.

21. Application of an Improved Ant Colony Optimization on Generalized Traveling Salesman Problem [Text] / Kan Jun-man, Zhang Yi // Energy Procedia, Volume 17, Part A, 2012. – Pages 319-325.

Приложение А

Relationships - Class Diagram