  
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
 «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Кафедра \_\_\_\_\_ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ\_\_\_\_\_\_\_\_\_\_\_\_\_

УТВЕРЖДАЮ

Зав. каф. Соболь Б.В.

"\_\_\_\_\_\_"\_\_\_\_\_\_\_\_2019 г.

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

К отчету по курсу «Основы программной инженерии»

на тему:\_ «Разработка ИС распределения доставки ресурсов при помощи генетического алгоритма»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Автор работы \_\_Калюжный Николай Александрович, Колесникова Алёна Игоревна, Ляшенко Марина Сергееевна\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Направление\_\_\_\_\_Информационные системы и технологии\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Курс \_\_\_\_4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Группа\_\_ВИС41\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_Венцов Н.Н.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (Ф.И.О.)

Работа защищена \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(дата) (оценка)

г. Ростов-на-Дону,

2019г.



МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
 «ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Кафедра \_\_\_\_\_\_\_\_ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ\_\_\_\_\_\_\_\_\_\_\_\_\_

УТВЕРЖДАЮ

Зав. каф. Соболь Б.В.

"\_\_\_\_\_\_"\_\_\_\_\_\_\_\_2019г.

**ЗАДАНИЕ**

на индивидуальную работу

Студенты Калюжный Н.А., Колесникова А.И., Ляшенко М.С. Курс\_4\_Группа\_ВИС41\_

Тема «Разработка ИС распределения доставки ресурсов при помощи генетического алгоритма»

Срок представления отчета к защите "\_\_\_\_\_\_\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019г.

**Исходные данные для проектирования:**

XML-документ, содержащий координаты маршрутов транспорта.

ИС позволяет найти наикратчайший маршрут движения транспорта по заданным значениям

**1 Содержание пояснительной записки курсовой работы**:

ВВЕДЕНИЕ

1 Постановка задачи

2 Разработка ИС

ЗАКЛЮЧЕНИЕ

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

ПРИЛОЖЕНИЕ

**2 Перечень графического материала:** 8 изображений\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель курсовой работы \_\_\_\_\_\_\_\_\_\_ Венцов Н.Н.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата (**фамилия**, инициалы)

Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись

Оглавление

[Введение 4](#_Toc4981343)

[Постановка задачи 4](#_Toc4981344)

[Разработка ИС 5](#_Toc4981345)

[Начальный этап 5](#_Toc4981346)

[Цикл смены поколений 6](#_Toc4981347)

[Заключение 9](#_Toc4981348)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 9](#_Toc4981349)

[ПРИЛОЖЕНИЕ 10](#_Toc4981350)

# Введение

В настоящее время все более актуальными становятся задачи оптимизации, поиска, реализации распределенных и (или) параллельных систем. Многие из них легко реализуемы простыми математическими методами, но некоторые задачи требуют к себе особого подхода. Эти задачи либо не разрешимы простыми методами, либо их решение потребует значительного времени и объема ресурсов. Для решения подобного рода задач существуют особые эвристические методы и алгоритмы. К их числу относятся генетические алгоритмы.

Генетический алгоритм — это адаптивные методы функциональной оптимизации, основанные на компьютерном имитационном моделировании биологической эволюции. Основные принципы ГА были сформулированы Голландом, и хорошо описаны во многих работах и на ряде сайтов в Internet. В настоящее время существует ряд теорий биологической эволюции (Ж.-Б.Ламарка, П.Тейяра де Шардена, К.Э.Бэра, Л.С.Берга, А.А.Любищева, С.В.Мейена и др.), однако, ни одна из них не считается общепризнанной.

Наиболее известной и популярной, конечно, является теория Чарльза Дарвина, которую он представил в работе "Происхождение Видов" в 1859 году, и тем ни менее, не смотря на свои недостатки, именно теория Дарвина традиционно моделируется в ГА. Она применима не к отдельным особям, а к популяциям – большому количеству особей одного вида, т.е. способных давать плодовитое потомство, находящейся в определенной статичной или динамичной внешней среде. В основе модели эволюции Дарвина лежат случайные изменения отдельных материальных элементов живого организма при переходе от поколения к поколению. Целесообразные изменения, которые облегчают выживание и производство потомков в данной конкретной внешней среде, сохраняются и передаются потомству, т.е. наследуются. Особи, не имеющие соответствующих приспособлений, погибают, не оставив потомства или оставив его меньше, чем приспособленные. Поэтому в результате естественного отбора возникает популяция из наиболее приспособленных особей, которая может стать основой нового вида.

# Постановка задачи

Есть некий склад, на котором хранится один вид ресурса. Есть n-большое потребители, которым необходимо доставить этот ресурс, таким образом, чтобы суммарный пробег транспортного средства был минимальным.

# Разработка ИС

Программа написана на языке C# в среде Visual Studio 2017. Полный код программы находится в Приложении.

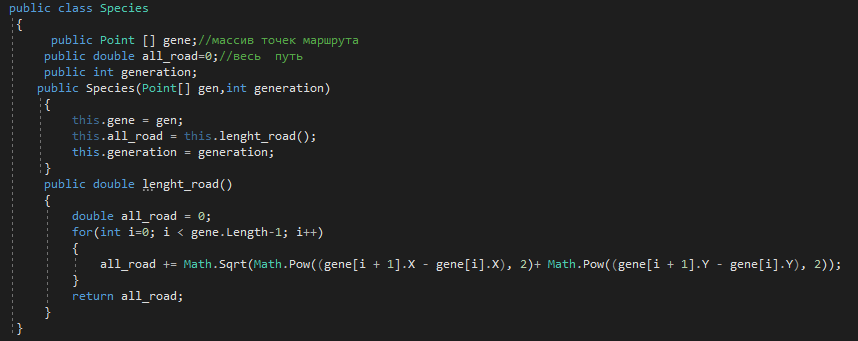
## Начальный этап

Для начала работы генетического алгоритма, создаём начальную популяцию из 9 особей(маршрутов), обладающих 12 признаков(потребителей), которые хранятся в XML-файле (Рис.1).



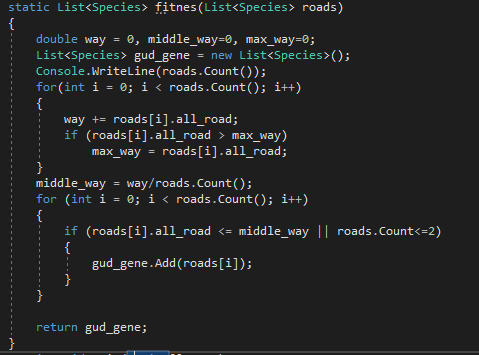
*Рисунок 1 – Хранилище начальной популяции*

После создания популяции производим оценка приспособленности каждой особи по её признакам с использованием критериев. Критерием данной задачи было минимизация маршрута передвижения транспорта. Для того, чтобы проще было проводить оценку, координатные критерии были перестроены в длину маршрута (Рис.2)



*Рисунок 2 – Преобразование координатных значений в длину пути*

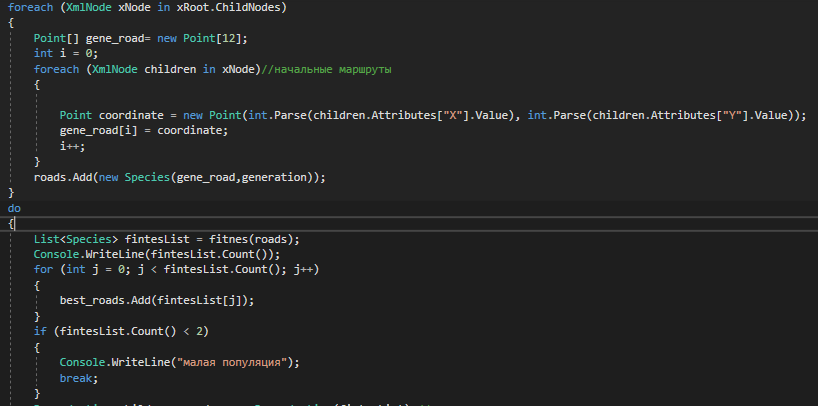
После чего используем Fitness-функцию (функция приспособленности) для поиска приоритетных критерию особей (Рис.3).



*Рисунок 3 – Fitness-функция (функция приспособленности)*

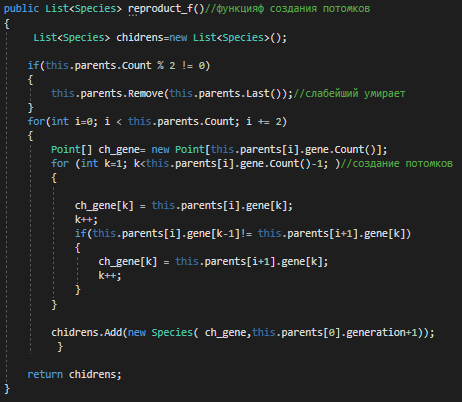
## Цикл смены поколений

В начале цикла смены поколений формируем новое поколение, для которого осуществляем пропорциональный отбор особей, которые могут участвовать в продолжении рода. Отбираются только те особи популяции, у которых количество информации в фенотипе и генотипе о выживании и продолжении рода положительно, причем вероятность выбора пропорциональна этому количеству информации (Рис.4).



*Рисунок 4 – Отбор особей для новой популяции*

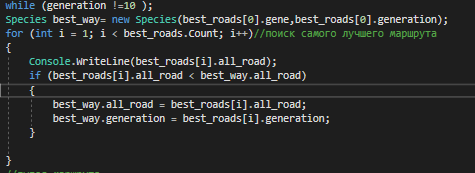
Отобранных особей для новой популяции подвергаем кроссоверу(скрещиванию). Если кроссовер происходит, то потомки получают по половине случайным образом определенных признаков от каждого из родителей. Если кроссовер не происходит, то потомство не появляется (Рис.5)



*Рисунок 5 – Функция создания потомка кроссовером*

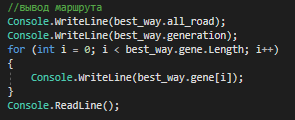
Для нового поколения проводится оценка приспособленности особей, которая проводилась для начальной популяции и после чего цикл нового поколения с использование отбора и кроссовера повторяется до тех пор, пока все пары поколений не будут рассмотрены.

Как только все пары предыдущего поколения пройдут отбор и кроссовер, переносим наиболее приспособленных особей из старого поколения в новое и заменяем старое поколение новым (Рис.6).

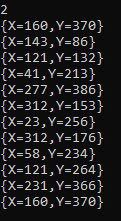


*Рисунок 6 – Поиск приспособленных особей*

Когда будут найдены самые приспособленные особи, мы их выводим на консоль (Рис.7). Выглядит это в таком виде (Рис.9).



*Рисунок 7 – Вывод лучшего маршрута (кодовая версия)*



*Рисунок 8 – Вывод лучшего маршрута (консольная версия)*

# Заключение

В ходе выполнения работы, были получены следующие результаты:

1. Разработана программа для распределения доставки ресурсов;
2. Проведено исследование генетического алгоритма для выбора оптимальных параметров;
3. Разработанная программа находит решение в допустимое время;

На основании проведенной работы можно сделать выводы о том, что в

настоящее время генетические алгоритмы являются эффективным

вычислительным средством при решении задач оптимизации, для которых

нет специального алгоритма решения.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Панченко Т. В. Генетические алгоритмы: учебно-методическое

пособие / Т.В. Панченко. – Астрахань: Изд. дом «Астраханский

университет», 2007. – 87 с.

1. De Jong, K.A. Introduction to the second special issue on genetic

algorithms. /K.A. De Jong. – Machine Learning, 5(4). – p. 351-353.

1. Батищев Д. И., Неймарк Е. А., Старостин Н. В. Применение

генетических алгоритмов к решению задач дискретной оптимизации / Д. И.

Батищев, Е. А. Неймарк, Н. В. Старостин. – Н. Новгород: 2007. – 85 с.

# ПРИЛОЖЕНИЕ

Программный код ИС

***Species.cs***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml;

using System.Xml.Linq;

using System.Drawing;

namespace GA

{

public class Species

{

public Point [] gene;//массив точек маршрута

public double all\_road=0;//весь путь

public int generation;

public Species(Point[] gen,int generation)

{

this.gene = gen;

this.all\_road = this.lenght\_road();

this.generation = generation;

}

public double lenght\_road()

{

double all\_road = 0;

for(int i=0; i < gene.Length-1; i++)

{

all\_road += Math.Sqrt(Math.Pow((gene[i + 1].X - gene[i].X), 2)+ Math.Pow((gene[i + 1].Y - gene[i].Y), 2));

}

return all\_road;

}

}

}

***Reproduction.cs***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Drawing;

namespace GA

{

class Reproduction

{

public List<Species> parents;

public List<Species> chidrens ;

public Reproduction(List<Species> parents)

{

this.parents = parents;

this.chidrens=this.reproduct\_f();

}

/\* private bool live\_or\_die(Point[] gene)//проврка выживаемости потомка

{

bool flag = true;

for(int i=1; i<gene.Length-1; i++)

{

for(int j = 1; j < gene.Length - 1; j++)

{

if(gene[j]==gene[i] && j != i)

{

flag= false;

break;

}

}

if (flag == false)

break;

}

return flag;

}\*/

public List<Species> reproduct\_f()//функцияф создания потомков

{

List<Species> chidrens=new List<Species>();

if(this.parents.Count % 2 != 0)

{

this.parents.Remove(this.parents.Last());//слабейший умирает

}

for(int i=0; i < this.parents.Count; i += 2)

{

Point[] ch\_gene= new Point[this.parents[i].gene.Count()];

for (int k=1; k<this.parents[i].gene.Count()-1; )//создание потомков

{

ch\_gene[k] = this.parents[i].gene[k];

k++;

if(this.parents[i].gene[k-1]!= this.parents[i+1].gene[k])

{

ch\_gene[k] = this.parents[i+1].gene[k];

k++;

}

}

chidrens.Add(new Species( ch\_gene,this.parents[0].generation+1));

}

return chidrens;

}

}

}

***Program.cs***

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Xml;

using System.Xml.Linq;

using System.Drawing;

namespace GA

{

class Program

{

static List<Species> fitnes(List<Species> roads)

{

double way = 0, middle\_way=0, max\_way=0;

List<Species> gud\_gene = new List<Species>();

Console.WriteLine(roads.Count());

for(int i = 0; i < roads.Count(); i++)

{

//Console.WriteLine("{0} - {1}",roads[i].all\_road, roads[i].generation);

way += roads[i].all\_road;

if (roads[i].all\_road > max\_way)

max\_way = roads[i].all\_road;

}

middle\_way = way/roads.Count();

for (int i = 0; i < roads.Count(); i++)

{

if (roads[i].all\_road <= middle\_way || roads.Count<=2)

{

gud\_gene.Add(roads[i]);

}

}

return gud\_gene;

}

static void Main(string[] args)

{

List<Species> roads = new List<Species>();

List<Species> best\_roads = new List<Species>();

XmlDocument road = new XmlDocument();

int generation = 1;

road.Load("Road.XML");

XmlElement xRoot = road.DocumentElement;

foreach (XmlNode xNode in xRoot.ChildNodes)

{

Point[] gene\_road= new Point[12];

int i = 0;

foreach (XmlNode children in xNode)//начальные маршруты

{

Point coordinate = new Point(int.Parse(children.Attributes["X"].Value), int.Parse(children.Attributes["Y"].Value));

gene\_road[i] = coordinate;

i++;

}

roads.Add(new Species(gene\_road,generation));

}

do

{

List<Species> fintesList = fitnes(roads);

Console.WriteLine(fintesList.Count());

for (int j = 0; j < fintesList.Count(); j++)

{

best\_roads.Add(fintesList[j]);

}

if (fintesList.Count() < 2)

{

Console.WriteLine("малая популяция");

break;

}

Reproduction childrens\_road = new Reproduction(fintesList);//создание потомков

roads.Clear();

roads = childrens\_road.chidrens;

generation++;

}

while (generation !=10 );

Species best\_way= new Species(best\_roads[0].gene,best\_roads[0].generation);

for (int i = 1; i < best\_roads.Count; i++)//поиск самого лучшего маршрута

{

Console.WriteLine(best\_roads[i].all\_road);

if (best\_roads[i].all\_road < best\_way.all\_road)

{

best\_way.all\_road = best\_roads[i].all\_road;

best\_way.generation = best\_roads[i].generation;

}

}

//вывод маршрута

Console.WriteLine(best\_way.all\_road);

Console.WriteLine(best\_way.generation);

for (int i = 0; i < best\_way.gene.Length; i++)

{

Console.WriteLine(best\_way.gene[i]);

}

Console.ReadLine();

}

}

}