**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ  
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук  
Департамент программной инженерии

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Старший преподаватель департамента программной инженерии факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д. В. Пантюхин «\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2017 г. | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия» профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. В. Шилов «\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2017 г. |

|  |  |
| --- | --- |
| ***Подп. и дата*** |  |
| ***Инв. № дубл.*** |  |
| ***Взам. инв. №*** |  |
| ***Подп. и дата*** |  |
| ***Инв. № подл*** | RU.17701729.503200-01 81 01-1-ЛУ |

**ПРОГРАММА КЛАСТЕРИЗАЦИИ АЛГОРИТМОМ FASTDBSCAN**

**Пояснительная записка**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.503200-01 81 01-1-ЛУ**

Исполнитель  
студент группы БПИ165  
\_\_\_\_\_\_\_\_\_\_ / Соколов Д.В. /  
«\_\_» \_\_\_\_\_\_\_\_\_\_\_ 2017 г.

**Москва 2017**

УТВЕРЖДЕНRU.17701729.503200-01 51 01-1-ЛУ

|  |  |
| --- | --- |
| Подп. и дата |  |
| Инв. № дубл. |  |
| Взам. инв. № |  |
| Подп. и дата |  |
| Инв. № подл | RU.17701729.503200-01 81 01-1 |

**ПРОГРАММА КЛАСТЕРИЗАЦИИ АЛГОРИТМОМ FASTDBSCAN**

**Пояснительная записка**

**RU.17701729.503200-01 81 01-1**

**Листов 24**

**Москва 2017**

**Содержание**

[**1. Введение** 3](#_Toc451347290)

[**1.1. Наименование программы** 3](#_Toc451347291)

[**1.2. Основания для разработки** 3](#_Toc451347292)

[**2. Назначение и область применения** 4](#_Toc451347293)

[**2.1. Назначение разработки** 4](#_Toc451347294)

[**2.1.1. Функциональное назначение** 4](#_Toc451347295)

[**2.1.2. Эксплуатационное назначение** 4](#_Toc451347296)

[**2.2. Область применения** 4](#_Toc451347297)

[**3. Технические характеристики** 5](#_Toc451347298)

[**3.1. Постановка задачи на разработку программы** 5](#_Toc451347299)

[**3.2. Описание и обоснование алгоритма и функционирования программы** 5](#_Toc451347300)

[**3.2.1. Описание алгоритма и функционирования программы** 5](#_Toc451347301)

[**3.2.2. Обоснование выбора алгоритма** 6](#_Toc451347302)

[**3.3. Описание и обоснование метода организации входных и выходных данных** 7](#_Toc451347303)

[**3.3.1. Описание метода организации входных и выходных данных** 7](#_Toc451347304)

[**3.3.2. Обоснование метода организации входных и выходных данных** 7](#_Toc451347305)

[**3.4. Описание и обоснование выбора состава технических и программных средств** 7](#_Toc451347306)

[**3.4.1. Состав технических и программных средств** 7](#_Toc451347307)

[**3.4.2. Обоснование выбора состава технических и программных средств** 8](#_Toc451347308)

[**4. Технико-экономические показатели** 9](#_Toc451347309)

[**4.1. Предполагаемая потребность** 9](#_Toc451347310)

[**4.2. Ориентировочная экономическая эффективность** 9](#_Toc451347311)

[**4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами** 9](#_Toc451347312)

[**Приложение 1. Список используемой литературы** 10](#_Toc451347313)

[**Приложение 2. Описание и функциональное назначение классов** 11](#_Toc451347314)

[**Приложение 3. Описание и функциональное назначение полей, методов и свойств** 12](#_Toc451347315)

# **1. Введение**

## **1.1. Наименование программы**

Наименование программы: «Программа кластеризации алгоритмом FastDBscan».

## **1.2. Основания для разработки**

Основанием для разработки программы является Приказ НИУ ВШЭ № 2.3-02/0812-01 от 08.12.2016

Программа разрабатывается в рамках выполнения курсовой работы по теме «Программа кластеризации алгоритмом FastDBscan».

# **2. Назначение и область применения**

## **2.1. Назначение разработки**

### **2.1.1. Функциональное назначение**

Программа будет применяться для кластерного анализа входных данных, а также для демонстрации и исследования результата. У пользователя будет возможность ввести данные точек и один из вариантов кластеризации из доступных, выбрать параметры кластеризации и посмотреть отображаемый результат.

### **2.1.2. Эксплуатационное назначение**

Программа будет использоваться для кластерного анализа входных данных, а также для демонстрации и исследования результата кластеризации, оценки точности, качества и скорости работы различных доступных вариантов кластеризации.

## **2.2. Область применения**

Программа будет применяться для кластерного анализа входных данных, являющимися точками в двумерном пространстве, с возможность изменения параметров кластеризации и использованием алгоритмов FastDBscan и DBscan.

# **3. Технические характеристики**

## **3.1. Постановка задачи на разработку программы**

Программа должна обеспечивать возможность выполнения следующих функций:

1. кластеризация точек из файла с входными данными;
2. настройка параметров для доступных алгоритмов кластеризации;
3. вывод результат кластеризации в форму графика с точками различных цветов, где цвет характеризует принадлежность точки к кластеру;
4. запуск других алгоритмов кластеризации над другими данными и с другими параметрами без перезапуска программы.

## **3.2. Описание и обоснование алгоритма и функционирования программы**

### **3.2.1. Описание алгоритма и функционирования программы**

В программе используется два алгоритма кластеризации DBscan и K-means.

Входные данные для алгоритма DBscan:

1. Множество точек, над которыми будет выполняться алгоритм
2. Eps – максимальное расстояние между двумя точками, которые будем считать достижимыми между собой.
3. minPTS – минимальное количество точек в Eps-окрестности для каждой точки, которая может входить в кластер.

Алгоритм выполнения DBscan:

1. программа проходит по всем точкам и для каждой из них считает расстояние до текущей, т.е. ищет количество точек в Eps-окрестности данной.
2. Если количество точек в Eps-окрестности меньше чем minPTS, то сразу помечаем точку как шум и переходим к пункту 1 с другой не принадлежащей никакому кластеру точкой.
3. Если количество точек в Eps-окрестности больше или равно minPTS, то помечаем точку как принадлежащую текущему кластеру и переходим к пункту 4.
4. Если в Eps-окрестности есть точка, не принадлежащая никакому кластеру, то переходим к 1 пункту для этой точки.
5. Если в Eps-окрестности этой точки все остальные уже принадлежат какому-то кластеру, то переходим к 1 пункту с другой не принадлежащей никакому кластеру точкой и новым кластером.

Входные данные для K-means:

1. Множество точек, над которым будет выполняться кластеризация.
2. K – количество ожидаемых кластеров для K-means

Алгоритм выполнения K-means:

1. Генерируем K центроидов(точек) – центров будущих кластеров
2. Для каждой точки их входных данных считаем расстояние до каждого из центроидов. Помечаем точку как принадлежащую самому ближайшему центроиду.
3. Пересчитываем центроиды как среднее всех точек, которые принадлежат центроиду
4. Проверяем изменилась ли принадлежность точек к центроиду. Если принадлежность не изменилась – заканчиваем работу алгоритма, если изменилась переходим к пункту 2

Входные данные для FastDBscan:

1. Множество точек, над которым будет выполняться кластеризация.
2. K – количество ожидаемых кластеров для K-means
3. Eps – максимальное расстояние между двумя точками, которые будем считать достижимыми между собой.
4. minPTS – минимальное количество точек в Eps-окрестности для каждой точки, которая может входить в кластер.
5. T – процент точек из каждого кластера K-means, которые берутся для обработки в DBscan’е Алгоритм выполнения K-means.

Алгоритм выполнения FastDBscan:

1. Для множества входных точек выполняем алгоритм K-means.
2. От каждого полученного кластера K-means берем T % точек. Получаем новое множество E.
3. Выполняем над множеством E алгоритм DBscan.
4. Отображаем результат на множество точек – для каждого кластера K-means берем максимальный лежащий внутри него кластер DBScan’a, все остальные кластеры меньшего размера помечаем как шум.

Также в программе есть вариант использовать R\*-дерево. Использование этого дерева позволяет сократить время на поиск соседей в Eps-окрестности для DBscan’а.

После выполнения данных алгоритмов происходит отрисовка цветного графика данного множества точек, где цвет точки характеризует ее принадлежность к кластеру. Черный цвет – шум, остальные цвета – полученные кластеры.

### **3.2.2. Обоснование выбора алгоритма**

Алгоритм FastDBscan выбран поскольку он позволяет ускорить кластеризацию данных при сохранении точности, достигаемой оригинальным DBscan.

## **3.3. Описание и обоснование метода организации входных и выходных данных**

### **3.3.1. Описание метода организации входных и выходных данных**

#### **3.3.1.1. Описание метода организации входных данных**

Программа получает входные данные с помощью кнопки на Windows Forms, которая открывает проводник для поиска .txt файла с точками. Параметры и варианты кластеризации вводятся через текстовые поля, выпадающие списки и флажки.

#### **3.3.1.2. Описание метода организации выходных данных**

Программа выводит результат кластеризации в виде цветного графика в отдельном окне Windows Forms с информацией о количестве точек, времени и скорости выполнения выбранных алгоритмов.

### **3.3.2. Обоснование метода организации входных и выходных данных**

#### **3.3.2.1. Обоснование метода организации входных данных**

Описанный в п. 3.3.1.1 настоящего документа метод организации входных данных выбран, потому что он является наиболее оптимальным и удобным для пользователя.

#### **3.3.2.2. Обоснование метода организации выходных данных**

Описанный в п. 3.3.1.2 настоящего документа выбран, потому что цветной график наиболее наглядно показывает результат кластеризации.

## **3.4. Описание и обоснование выбора состава технических и программных средств**

### **3.4.1. Состав технических и программных средств**

#### **3.4.1.1. Состав технических средств**

Для нормального функционирования программы требуется компьютер, оснащенный следующими техническими компонентами:

1. процессор не ниже Intel Pentium/Celeron, AMD K6/Athlon/Duron или совместимый с ними с тактовой частотой не ниже 1 ГГц;
2. 512 Мб ОЗУ или более;
3. жесткий диск с объемом свободной памяти не менее 500Мб;
4. VGA-совместимые видеоадаптер и монитор с разрешением не ниже 800х600;
5. клавиатура и мышь.

#### **3.4.1.2. Состав программных средств**

Для нормального функционирования программы требуется компьютер, оснащенный следующими программными компонентами:

1. операционная система Microsoft Windows XP (SP2, SP3) / Vista / 7 / 8 / 8.1 / 10;
2. библиотека Microsoft .NET Framework 4.5 и выше.

### **3.4.2. Обоснование выбора состава технических и программных средств**

#### **3.4.2.1. Обоснование выбора состава технических средств**

При процессоре ниже указанного процесс кластеризации может занять довольно много времени.

При разрешении экрана ниже указанного окна программы могут отображаться некорректно.

Клавиатура и мышь требуются для ввода входных данных и управления программой.

#### **3.4.2.2. Обоснование выбора состава программных средств**

Поскольку программа выполнена с использованием библиотеки Microsoft .NET Framework 4.5.2, для ее выполнения требуется библиотека Microsoft .NET Framework версии не ниже 4.5, которая, в свою очередь, требует одну из указанных операционных систем.

# **4. Технико-экономические показатели**

## **4.1. Предполагаемая потребность**

Программа будет использоваться преподавателями, студентами и исследователями в области математики и информатики для решения задачи интеллектуального анализа данных, а также для демонстрации и исследования доступных в программе алгоритмов кластеризации, оценки их скорости и точности.

## **4.2. Ориентировочная экономическая эффективность**

Программа может бесплатно дать необходимый материал для преподавателей, студентов и исследователей в области математики и информатики, занимающихся исследованиями в интеллектуального анализа данных.

## **4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами**

Программа бесплатна и имеет русскоязычный интерфейс.

# **Приложение 1. Список используемой литературы**

1. Подбельский В. В. Язык C#. Базовый курс: учеб. пособие, 2-е изд., перераб. и доп. – М.: Финансы и статистика, 2013.
2. Microsoft Developer Network
3. Единая система программной документации: сборник, офиц. изд. – М.: ИПК Издательство стандартов, 2001.
4. Петцольд Ч. Программирование для Microsoft Windows на C#, 2002
5. Шилдт, Г. C# 4.0: полное руководство: пер. с англ. – М.: ООО «И. Д. Вильямс», 2013.
6. Вьет Тханг, Д. В. Пантюхин, А. И. Галушкин «Гибридный алгоритм кластеризации FastDBSCAN» <https://mipt.ru/upload/medialibrary/4a3/77-81.pdf>

# **Приложение 2. Описание и функциональное назначение классов**

Описываются только классы, содержащие вручную написанные поля, методы и свойства. Полностью автоматически сгенерированные классы не описываются.

*Табл. 1. Описание и функциональное назначение классов*

|  |  |
| --- | --- |
| Класс | Назначение |
| Form1 | Представляет форму Form1 – стартовое окно программы |
| Graphic | Представляет форму Graphic – окно с графиком из точек, полученных после кластеризации. |
| Point | Класс – точка на плоскости. |
| Dbscan | Статический класс. Представляет статические методы для кластеризации методом DBscan. |
| Dbscanwithrtree | Статический класс. Представляет статические методы для кластеризации методом DBscan с использованием R\*-дерева. |
| Kmeans | Статический класс. Представляет статические методы для кластеризации методом K-means. |
| WorkWithFilePoint | Статический класс. Представляет статические методы для получения элементов класса Point из файла. |
| ForRstartree | Класс, хранящий все поля и структуры данных, нужные для работы с R\*-дерева. |
| TRNode | Класс, представляющий узел в R\*-дереве и содержащий нужные методы для работы с узлом. |
| TRtree | Класс, представляющий R\*-дерево и содержащий нужные методы для работы с деревом. |

# **Приложение 3. Описание и функциональное назначение полей, методов и свойств**

Описываются только вручную написанные поля, методы и свойства. Полностью автоматически сгенерированные поля, методы и свойства не описываются.

*Табл. 2. Описание и функциональное назначение полей класса Form1*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| OnlyDBscan | private | bool | Булевская переменная. True – при кластеризации используется только алгоритм DBscan  False – алгоритм FastDBscan |
| UseRTree | private | bool | Булевская переменная. True – при кластеризации используется R\*-дерево  False – не используется R\*-дерево |

*Табл. 3. Описание и функциональное назначение методов класса Form1*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| button1\_Click | private | void | object, EventArgs | При нажатии на кнопку открывает проводник |
| button2\_Click | private | void | object, EventArgs | При нажатии на кнопку проверяет все введенные данные. Если они верны передает их в DrawGraph |
| closeToolStripMenuItem\_Click | private | void | object, EventArgs | При нажатии в меню кнопки “Закрыть” закрывает программу |
| DrawGraph | private | void | String, int, double, int, int | В зависимости от OnlyDBscan и UseRTree запускает выбранная вариация кластеризации |
| UseOnlyDBScanwithRtree | private | void | String,double,int | Выполняет кластеризацию DBscan с использованием R\*-дерева и открывает новое окно графика |
| UseFastDBscanwithRtree | private | void | String,int,double,int,int | Выполняет кластеризацию FastDBscan с использованием R\*-дерева и открывает новое окно графика |
| UseOnlyDBScan | private | void | String,double,int | Выполняет кластеризацию DBscan без R\*-дерева и открывает новое окно графика |
| UseOnlyFastDBScan | private | void | String,int,double,int,int | Выполняет кластеризацию FastDBscan без R\*-tree и открывает новое окно графика |
| GetFullPicture | private | void | List<Point>, int, int | Отражает результаты работы DBscan’а на результат работы K-means при работе FastDBscan без использования R\*-дерева |
| WorkWithDb | private | void | List<Point>, int, int | Обрабатывает переданный лист точек, которые принадлежат одному кластеру K-means. |
| readmetxtToolStripMenuItem\_Click | private | void | Object, EventArgs | Блокнотов открывает файл с пояснениями(reader.txt) |
| checkBox1\_Click | private | void | Object, KeyPressEventArgs | Изменяет флажок “Использовать R\*-tree” и переменную UseRTree |
| comboBox1\_SelectionChangeCommitted | private | void | Object, EventArgs | Изменяет значени выпадающего списка с вариантами:  1)FastDBscan  2)FBscan |

В классе Form1 отсутствуют вручную написанные свойства.

*Табл. 4. Описание и функциональное назначение полей класса Graphic*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| array | private | Color | Массив доступных цветов для кластеризации |

*Табл. 5. Описание и функциональное назначение методов класса FormGraph*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| Graphic | public | - | List<Point>, int, Stopwatch, Stopwatch, string | Конструктор класса FormGraph для FastDBscan |
| Graphic | public | - | List<Point>, int, Stopwatch, string | Конструктор класса FormGraph для DBscan |
| DrawGraph | private | void | List<Point>, int, string | Метод для отрисовки результата FastDBscan |
| DrawGraph | private | void | List<Point>, int, string, string | Метод для отрисовки результата DBscan |
| zedGraph\_ContextMenuBuilder | private | void | ZedGraphControl, ContextMenuStrip, System.Drawing.Mouse, ZedGraphControl.ContextMenuObjectState | Меняем список вариантов работы с графиком, который отображается при нажатии на правую кнопку |
| pointitem\_Click | private | void | Object, EventArgs | Включает отображение доступных подсказок на графике |
| zedGraph\_Point\_show | string | void | ZedGraphControl, GraphPane, CurveItem, int | Метод для отображения координаты точки при наведении |
| newMenuItem\_Click | private | void | Object, EventArgs | Включает отображение доступных подсказок на графике |
| zedGraph\_Cluster\_show | private | void | ZedGraphControl, GraphPane, CurveItem, int | Метод для отображения номера кластера при наведении на точку |
| toolStripLabel1\_Click | private | void | Object, EventArgs | Метод для отображения времени кластеризации |

В классе Graphic отсутствуют вручную написанные свойства.

*Табл. 6. Описание и функциональное назначение полей класса DBscan*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| eps | public | double | Статическая переменная. Eps – расстояние для DBscan. |
| minPTS | private | int | Статическая переменная. minPTS – минимальное количество точек для Eps-окрестности |

В классе DBscan отсутствуют вручную написанные свойства.

*Табл. 7. Описание и функциональное назначение методов класса DBscan*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| Clustering | public | Int | List<Point>, double, int | Метод, запускающий кластеризацию DBscan. Возвращает целое число – количество полученных кластеров. |
| CheckAllPoint | private | void | List<Point> | Метод, который для каждой точке проверяет, есть ли в eps окрестности каждой точки хотя бы minPTS точек. |
| StartClusterization | private | void | List<Point>, Point, cluster | Рекурсивный метод, который запускается от каждой точки, прошедшей проверку на плотность |

В классе Kmeans отсутствуют вручную написанные поля и свойства.

*Табл. 8. Описание и функциональное назначение методов класса Kmeans*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| Cluster | public | Int[] | List<Point>,int,int | Производит алгоритм кластеризации K-means |
| InitMeans | private | List<Point> | Int,List<Point>,int | Метод генерирует центроиды. |
| Normalized | private | void | List<Point> | Метод выполняет нормализацию данных |
| UpdateMeans | private | bool | List<Point>,List<Point> | Метод для пересчета центроидов. |
| UpdateClustering | public | double | List<Point>,List<Point> | Пересчитывает принадлежность точек к центроидам |
| MinIndex | private | int | Double[] | Возвращает индекс центроида, до которого минимальное расстояние до точки |

*Табл. 9. Описание и функциональное назначение полей класса Point*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| x | public | double | Абцисса точки |
| y | public | double | Ордината точки |
| db | public | int | Кластер DBscan |
| IsVisited | public | bool | Посещена ли точка при выполнении DBscan |
| minPTScheck | public | bool | Выполнено ли условие на количество точек в Eps-окрестности |
| xnorm | public | double | Нормализованная абцисса |
| idx | public | int | Id точки в дереве |
| findpoint | public | List<Point> | Лист точек в Eps-окрестности |
| ynorm | public | double | Нормализовання ордината |
| cluster | public | int | Номер кластера K-means |

В классе Point отсутствуют вручную написанные свойства.

*Табл. 10. Описание и функциональное назначение методов класса* Point

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| Point | public | - | Double, double | Конструктор точки с двумя координатами |
| Distance | public | double | Point | Возвращает расстояние между текущей точкой и переданной. |
| Point | public | - | Double, double, int | Конструктор с двумя координатами и id |

*Табл. 11. Описание и функциональное назначение полей класса WorkWithFilePoint*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| xmin | public | int | Самая минимальная абцисса из входных данных. |
| xmax | public | int | Самая максимальная абцисса из входных данных. |
| ymin | public | int | Самая минимальная ордината из входных данных. |
| ymax | public | int | Самая максимальная ордината из входных данных. |
| number\_of\_stroke | public | int | Текущий номер строки при считывании из файла |

В классе WorkWithFilePoint отсутствуют вручную написанные свойства.

*Табл. 12. Описание и функциональное назначение методов класса WorkWithFilePoint*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| GetPointFromFile | public | List<Point> | string | Метод, считывающий из файла по заданном пути, все точки |
| GetPointFromString | private | Point | string | Метод, выполняет объекта класса Point из входной строки. |

*Табл. 13. Описание и функциональное назначение полей класса Dbscanwithrtree*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| eps | private | double | Статическая переменная. Eps – расстояние для DBscan. |
| minPTS | private | int | Статическая переменная. minPTS – минимальное количество точек для Eps-окрестности |

В классе Dbscanwithrtree отсутствуют вручную написанные свойства.

*Табл. 14. Описание и функциональное назначение методов класса Dbscanwithrtree*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| Clustering | public | List<TRObject> | List<TRObject>, double, int, out int | Метод, запускающий кластеризацию DBscan с использованием R\*-дерева. По ссылке возвращает количество полученных кластеров. |
| CheckAllPoints | private | void | TRtree | Проверяет все точки дерева на плотность |
| Check | private | bool | TRtree, Point | Проверяет выполняется ли правило плотности в данной точке |
| Startclust | private | void | TRtree, List<Point>, int | Рекурсивный метод, который запускается от каждой точки, прошедшей проверку на плотность |

*Табл. 15. Описание и функциональное назначение полей класса ForRStartree*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| MAX\_M | public | int | Минимальное количество точек в узле |
| MIN\_M | private | int | Минимальное количество объектов в узле |
| TAxis | public | enum | По какой оси будет разделение |
| TBound | protected | enum | Граница по которой будет происходить сортировка(левая или правая) |
| Point | public | struct | Структура описывающая точка( внутри double X, double Y) |
| TMBR | public | struct | Структура, описывающая прямоугольник(Point Left, Point Right), где Left – верхняя левая точка, Right – правая нижняя |

В классе *ForRStartree* отсутствуют вручную написанные методы и свойства.

*Табл. 16. Описание и функциональное назначение полей класса TRNode*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| fmbr | public | TMBR | Ограничивающий прямоугольник для узла |
| Fparent | public | int | Индекс узла родителя в массиве узлов дерева |
| FChildren | public | Int[] | Список индексов дочерних узлов |
| FObject | public | Point[] | Массив точек, принадлежащих узлу |
| FisLeaf | public | bool | Является ли узел листом |

*Табл. 17. Описание и функциональное назначение методов класса TRNode*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| getIsLeaf | public | void | bool | Есть ли дочерние узлы |
| getChild | public | int | int | Возвращает индекс дочернего узла |
| getObject | public | Point | int | Метод для доступа к точкам в узле |
| setChild | public | void | Int,int | Метод присваивания дочернего узла |
| setObject | public | void | Int,Point | Метод присваивания узлу точки |
| setParent | public | void | int | Метод присваивания узла-родителя |
| copy | public | void | TRNode | Метод копирования узла |
| TRNode | public | - | - | Конструктор создания узла |
| isIntersected | public | bool | TMBR,TMBR | Статический метод для проверки пересечения двух прямоугольников |
| IsIntersected | public | bool | TMBR | Метод для проверки пересечения текущего узла с переданным прямоугольником |
| Overlap | public | double | TMBR | Возвращает площадь перекрытия узла с заданной областью |
| Area | public | double | TMBR | Возвращает площадь переданного прямоугольника |
| margin | public | double | - | Возвращает периметр узла |

*Табл. 18. Описание и функциональное назначение полей класса TRtree*

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Назначение |
| FNodeArr | public | TRNode | Массив узлов дерева |
| FRoot | private | int | Ссылка на корневой узел в массиве узлов |
| FHeight | Public | Int[] | Список индексов дочерних узлов |

*Табл. 19. Описание и функциональное назначение методов класса TRtree*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| QuickSort | public | void | Point[],int,int,Taxis | Сортировка точек |
| QuickSort | public | void | Int[], int, int, Taxis, TBound | Сортировка узлов по mbr |
| splitNodeRStart | public | void | Int,Point | Разделяет узел на 2 в соответствии с алгоритмами R\*-дерева |
| splitNodeRStart | public | void | Int,int | Разделяет узел на 2 в соответствии с алгоритмами R\*-дерева |
| updateMBR | public | void | TRNode | Обновляет mbr узла |
| updateMBR | public | void | int | Обновляет mbr объекта |
| chooseSplitAxis | public | TAxis | Point,int | Возвращает ось по которой будет разделяться узел |
| chooseSplitAxis | public | TAxis | Int,int | Возвращает ось по которой будет разделяться узел |
| chooseSubtree | public | int | Point,int | Выбирает оптимальный узел для вставки объекта |
| TRtree | public | - |  | Конструктор создания дерева |
| insertObject | public | void | Point | Метод вставки объекта в дерево |
| findObjectinarea | public | List<Point> | TMBR, int, List<Point>, int | Проверяет пересечения области поиска с одним из узлов |
| findobjectinarea | public | List<TRObject> | TMBR, int, List<Point>, int | Проверяет есть ли объекты в узле, для которых выполняется условие плотности точек |
| newNode | private | void | int | Создает новый узел и возвращает его индекс в списке узлов |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Изм. | Номера листов | | | | Всего листов (страниц) в документе | № документа | Входя- щий № сопро- водит. докум. и дата | Под- пись | Дата |
| изменен- ных | заменен- ных | новых | аннулиро- ванных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |