МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ М. В. ЛОМОНОСОВА

ФИЛИАЛ МОСКОВСКОГО ГОСУДАРСТВЕННОГО УНИВЕРСИТЕТА ИМЕНИ М. В. ЛОМОНОСОВА В ГОРОДЕ СЕВАСТОПОЛЕ

Факультет «Компьютерной математики» Направление подготовки «Прикладная математика и информатика»

07.06.22 (бакалавр)

**ОТЧЕТ**

**по лабораторной работе №5**

**«Локализация цепно-рекуррентного множества динамической системы»**

Работу выполнил: студент группы ПМ-201

Хаметов Марк Владимирович

Руководитель: профессор кафедры прикладной математики и информатики Осипенко Георгий Сергеевич

Севастополь, 2022

**Оглавление**

[Постановка задачи 3](#_Toc101350858)

[Теоретическая часть 4](#_Toc101350859)

[Алгоритм решения 6](#_Toc101350860)

[Результаты 7](#_Toc101350861)

[Полученные изображения 8](#_Toc101350862)

[Использованная литература 9](#_Toc101350864)

# 

# Постановка задачи

Построить символический образ цепно-рекуррентного множества полученного локализацией графа для отображения Жулия. Отображение Жулия выглядит следующим образом:

xn = xn-1^2 - yn-1 ^2 + a

yn = 2\*yn-1\*xn-1 + b

Известно что при параметрах a и b равным 0.3 и 0.2 соответственно в области [-1.5;1.5] x [-1.5;1.5] существует замкнутое инвариантное множество, которое располагается в ограниченной области.

Источником формулы отображения Жулия является Википедия.

# Теоретическая часть

Замкнутое инвариантное множество является аттрактором тогда и только тогда, когда существует такая окрестность этого множества, что все точки множества, которые попали в данную окрестность, образуют исходное множество.

Сильно связанные вершины графа – это такие вершины ‘а‘ и ‘б‘ ориентированного графа, что одновременно существуют и ориентированный путь из ‘а‘ в ‘б‘, и из ‘б‘ в ‘а‘.

Для реализации мы разбиваем интересующую нас область на ячейки в виде прямоугольников одинакового размера. Нумерация ячеек производится по строкам отсчитывая от верхнего левого угла. Например:

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| 5 | 6 | 7 | 8 |
| 9 | 10 | 11 | 12 |

Применяем отображение Жулия к точкам каждой ячейки. Для каждой ячейки находим в какие ячейки попадает результат. Для этого применяем формулу:

n = [(x - xmin) / hx ] + 1 + [(ymin - y) / hy] \* [(xmax - xmin) / hx]

Для полученного списка строим граф состоящий из номеров ячеек в качестве вершин и связей между вершинами определенными наличием перехода между ячейками.

Далее мы применяем алгоритм для нахождения сильно связанных вершин графа. В нашем случае был использован алгоритм Тарьяна. Ячейки не имеющие соответствующих сильно связанных вершин отбрасываются.

Для локализации множества мы переразбиваем оставшиеся ячейки и повторяем алгоритм для них. Для переразбиения мы делим каждую ячейку на 4 равных ячеек. Такие итерации можно проводить пока вас не устроит результат. В нашем случае производилось 6 итераций.

Тогда то количество ячеек, на которое мы делим область, для нашего случая можно рассчитать следующей формулой на языке Haskell:

ghci> map (36\*) (map (4^) [0..5])

[36,144,576,2304,9216,36864]

Алгоритм Тарьяна взят с данного источника: www.geeksforgeeks.org/tarjan-algorithm-find-strongly-connected-components/

Определения получены по следующим ссылкам: ru.wikipedia.org/wiki/Компонента\_сильной\_связности

ru.wikipedia.org/wiki/Замкнутое\_множество

## Алгоритм решения

Алгоритм решения состоит из следующих шагов:

1. Инициализация переменных и получение параметров от пользователя.
2. Инициализация списков для хранения полученных точек;
3. Получаем значения от пользователя и, для оптимизации, считаем то, что не меняется внутри цикла
4. Начало цикла для первой ячейки. Пункты 5-8.
5. Генерируем 100 случайных точек внутри ячейки.
6. Считаем отображения для этих точек.
7. Считаем номера тех ячеек, в которые попали точки при отображении. Заносим в список.
8. Сортируем и удаляем совпадающие результаты. Выводим значения в консоль.
9. Конец цикла. Повторяем для следующей ячейки.
10. Для полученого списка ячеек строим граф связей между ними.
11. Используем алгоритм Тарьяна для нахождения сильно связанных вершин графа.
12. Отбрасываем ячейки без соответствующих сильно связанных вершин.
13. Переразбиваем оставшиеся ячейки.
14. Для нового списка ячеек можно перейти к 4 пункту алгоритма.

Алгоритм расписан самостоятельно.

# 

# Результаты

Программа написана на языке Python. Для шести итераций при делении области на 36 начальных ячеек получается итоговое деление области на 36864 ячеек. Исполнение программы занимает менее минуты.

## Полученные изображения

## 

xn = xn-1^2 - yn-1 ^2 + a

yn = 2\*yn-1\*xn-1 + b

## Где a = 0.3 и b = 0.2. Для области [-1.5;1.5] x [-1.5;1.5].

## На первом изображении (левое верхнее) область разделена на 36 ячеек.

## На втором изображении область разделена на 144 ячеек.

## На третьем изображении область разделена на 576 ячеек.

## На четвертом изображении область разделена на 2304 ячеек.

## На пятом изображении область разделена на 9216 ячеек.

## На шестом изображении область разделена на 36864 ячеек.

## 

# Использованная литература

1. “Введение в символический анализ динамических систем” Г.С.Осипенко, Н.Б.Ампилова.
2. <https://ru.wikipedia.org/wiki/Множество_Жюлиа>
3. <https://www.geeksforgeeks.org/tarjan-algorithm-find-strongly-connected-components/>
4. <https://ru.wikipedia.org/wiki/Компонента_сильной_связности>
5. <https://ru.wikipedia.org/wiki/Замкнутое_множество>