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»
01.03.02 (бакалавр)

ОТЧЕТ

по лабораторной работе №6

«Топологическая сортировка графа. Построение Аттрактора и фильтрация динамической системы»

Работу выполнил:
студент группы ПМ-401
Хаметов Марк Владимирович

Руководитель: профессор
кафедры прикладной
математики и информатики
Осипенко Георгий Сергеевич

Севастополь, 2023

Оглавление

Постановка задачи

Дан ориентированный циклический граф. Необходимо провести топологическую сортировку графа. Граф получен при построении символического образа динамической системы. Динамическая система получена из дифференциального уравнения.

Решение найдено на примере уравнения Дуффинга:

$$\ddot{x} + k\dot{x} + \alpha x + \beta x^3 = B\cos(\omega t)$$

Произведем замену y=ẋ и получим динамическую систему уравнений:

$$\begin{cases} \dot{x} = y \\ \dot{y} = -k\dot{x} - \alpha x - \beta x^3 + B\cos(wt) \end{cases}$$

В области [-2;-2]x[2;2] необходимо построить достаточно малое разбиение области на ячейки для символического образа аттракторов динамической системы.

Для построения символического образа берется область. Эту область впоследствии разбивают на ячейки. Это разбиение называют покрытием. Ячейки соответствующие аттракторам закрашиватся.

Теоретическая часть

Сильно связанные вершины графа – это подмножества таких вершин ориентированного графа, между которыми существует путь в обоих направлениях.

Разбиение области на ячейки в данной работе - это разбиение на прямоугольники одинакового размера. Длина ребер задается пользователем. Нумерация ячеек идет в порядке сначала слева направо, затем сверху вниз. Тогда обозначим ячейку $M(i)$, где i номер вершины графа.

Тогда вершины графа это номера ячеек. Ребро исходящее из вершины соответствует отображению из соответствующей ячейки в другую ячейку. Номер полученной ячейки задает конечную точку ребра. Так как мы не можем отобразить каждую точку в области, мы отображаем k равномерно распределенных точек каждой ячейки. Это число задается пользователем.

Для реализации отображения этих точек мы используем метод Рунге-Кутта. Шаг алгоритма задается пользователем. Начальное время алгоритма равняется нулю, а конечное время равняется периоду функции в правой части уравнения. В случае уравнения Дуффинга период равен:

$$T = \frac{2*\pi}{\omega}$$

По теореме 5.1 из источника [1]: Пусть P(d) - это окрестность равная объединению всех ячеек соответствующих возвратным вершинам графа, где d - это длина стороны ячейки.

$$P(d) = \{\cup\ M(i),\ i - \text{возвратная}\}$$

Тогда аттрактор динамической системы совпадает с пересечением множеств P(d) по формуле:

$$Q = \bigcap_{d>0} P(d)$$

По теореме 5.2 из источника [1]: При уменьшении размера ячейки новая

окрестность оказывается вложена в старую. Из этого следует то, что уменьшение диаметра ячеек приводит к меньшему размеру окрестности. Таким образом, последовательность окрестностей монотонно убывает и сходится к цепно-рекуррентному множеству по формуле:

$$\lim_{k\to\infty} P_k = \bigcap_k P_k = Q.$$

Для подсчета номера ячейки полученного после отображения точки области применяем формулу:

$$n = [(x - x_{min}) / h_x] + 1 + [(y_{min} - y) / h_y] * [(x_{max} - x_{min}) / h_x]$$

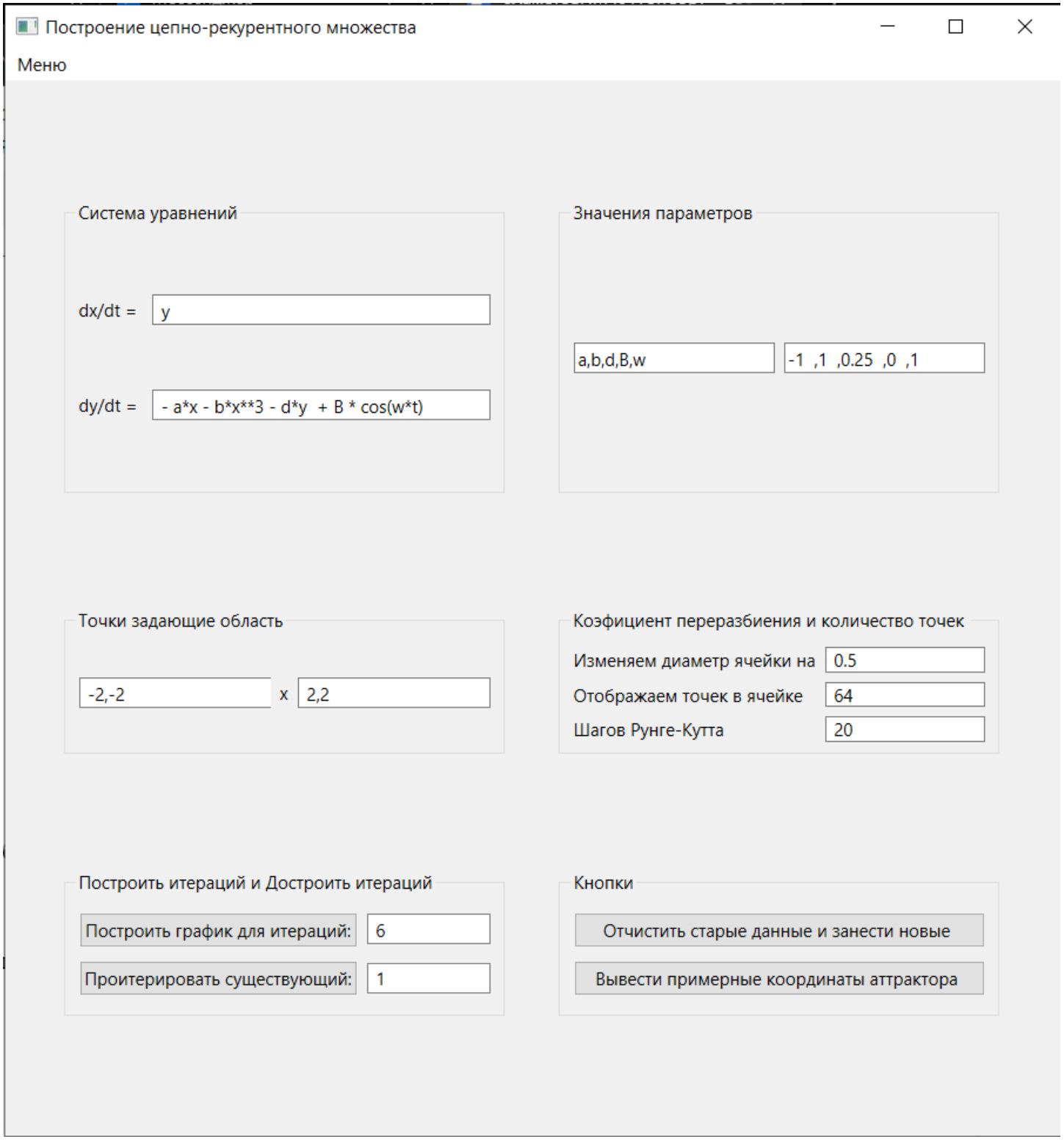
Для нахождения сильно связанных вершин графа и их топологической сортировки была использована функция kosaraju_strongly_connected_components из библиотеки networkx для языка программирования python. В результате работы алгоритма реализованного функцией мы получаем компоненты сильной связности в порядке соответствующем порядку прохождения аттракторов динамической системы

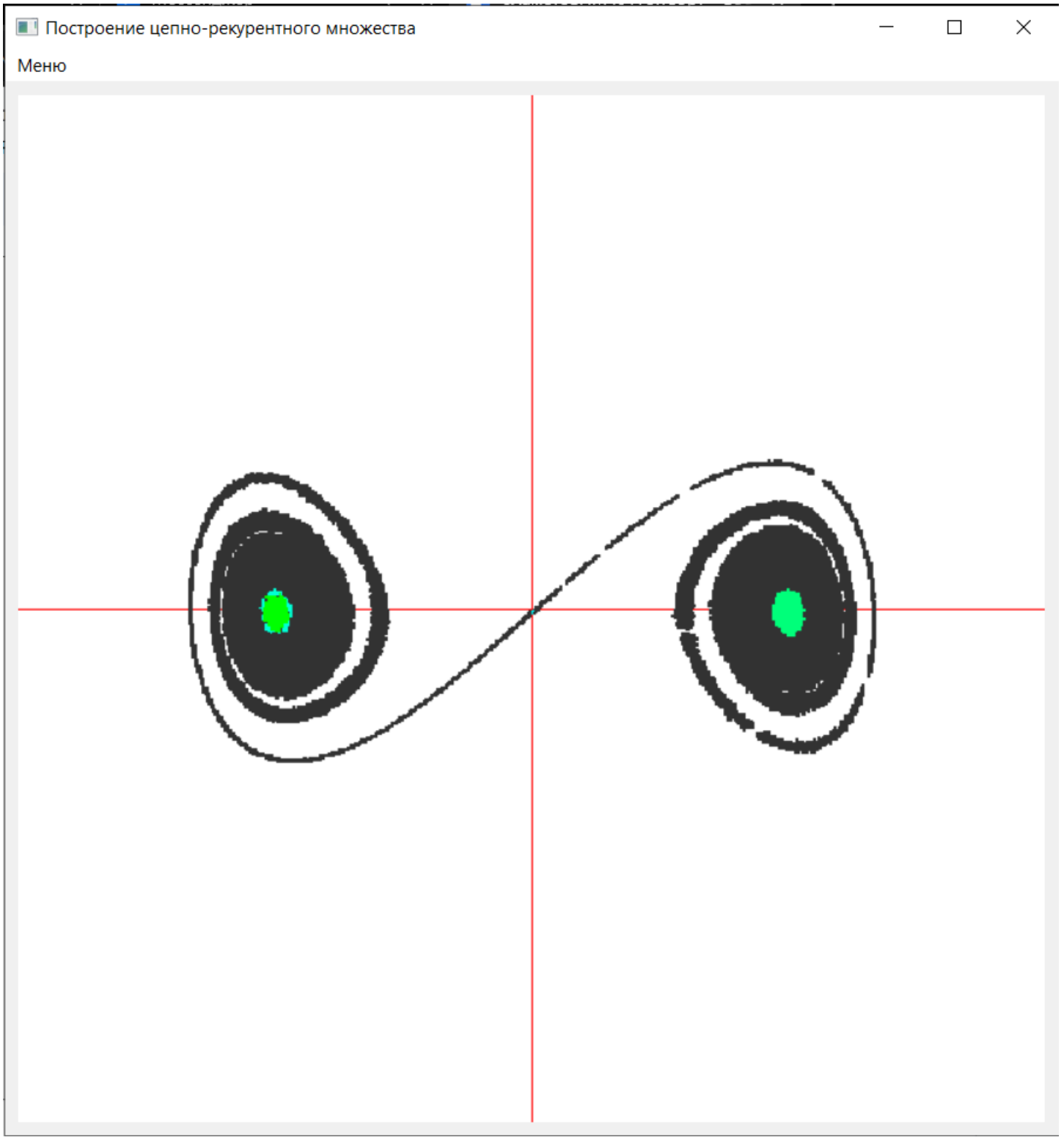
Так как разбиение на необходимое количество ячеек сразу приводит к применению сложного алгоритма к большому количеству ячеек, мы используем алгоритм локализации для построения символического образа. Для этого мы разбиваем область на малое количество ячеек и ищем сильно связанные вершины графа. Затем мы отбрасываем остальные вершины, для этого мы создаем список ячеек с которыми необходимо продолжить работу. Мы разбиваем область на ячейки размером меньше. Далее мы применяем алгоритм только для ячеек из списка. Алгоритм продолжает работу до момента достижения желаемого размера ячеек.

Для нахождения примерных точек аттракторов динамической системы для каждой компоненты сильной связности состоящей из 2 вершин и более:

1. Для каждой ячейки соответствующей компоненте отображаем точки этой ячейки.
2. Если точка попала в свою ячейку, то считаем результат отображения аттрактором, которому соответствует компонента сильной связности .

Интерфейс программы




Построение цепно-рекурентного множества
Меню

Построение цепно-рекурентного множества

Меню

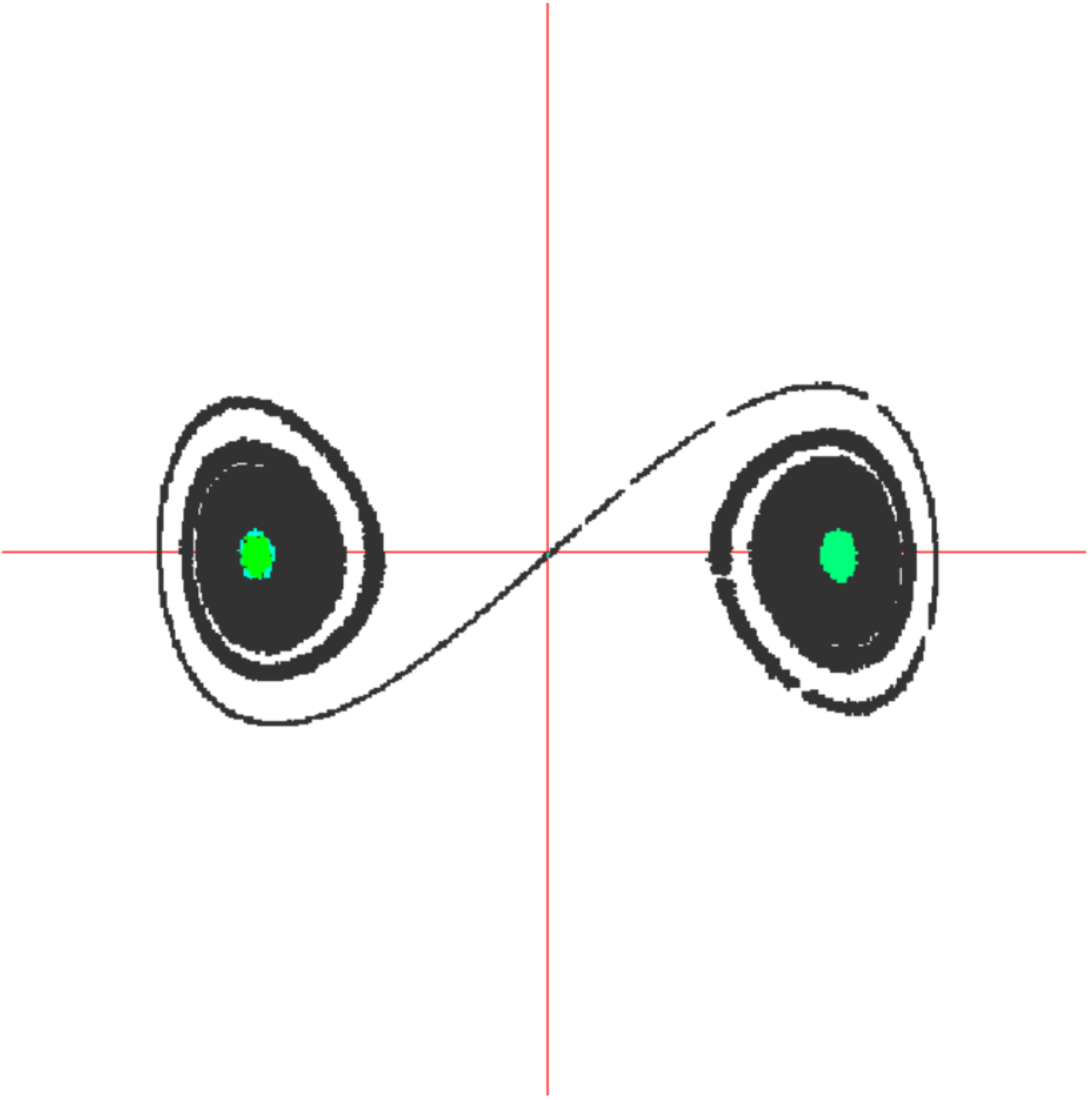
1 итерация. Занято времени 0.04504060745239258
Количество ячеек 256.0
2 итерация. Занято времени 0.16702604293823242
Количество ячеек 1024.0
3 итерация. Занято времени 0.5970368385314941
Количество ячеек 4096.0
4 итерация. Занято времени 1.6686086654663086
Количество ячеек 16384.0
5 итерация. Занято времени 3.733623743057251
Количество ячеек 65536.0
6 итерация. Занято времени 21.436715602874756
Количество ячеек 262144.0
На этой итерации также был нарисован график и посчитана топологическая сортировка графа
Предполагаемые точки аттракторов:
1.0627499999999999, -0.08036327515778718
1.0180624999999999, 0.13889266203703704
-1.0449062500000001, 0.09345721281828705
-1.0477291666666677, -0.18812872468171293
0.000875, -0.01661458334463614

7 итерация. Занято времени 154.08138132095337
Количество ячеек 1048576.0
На этой итерации также был нарисован график и посчитана топологическая сортировка графа
Предполагаемые точки аттракторов:
-0.983234375, 0.06150908063111482
0.993484375, 0.0775543984081127
-1.0076354166666668, -0.08610539240377921
0.0004375, -0.008307291668079517

Результаты

Программа написана на языке Python. Для создания интерфейса использовалась библиотека PySide6. Для расчетов использовались библиотеки networkx, math, time.

Для измерений результатов область задавалась [-2, -2]x[2, 2] и на каждой итерации длина ребра ячейки делилась на два, изначальная длина ребра была равна 0.5 .



Для параметров $\propto$ = -1 , β = 1 , k = 0.25 , B = 0 , w = 1 . Для 64 отображаемых точек в

виде сетки 8 на 8 из ячейки. Для 20 шагов алгоритма Рунге-Кутта каждой точки. Седьмая итерация и алгоритм топологической сортировки заняли 2,5 минуты. Всего программа исполнялась 3 минуты. Количество ячеек на финальном изображении равнялось 1048576.

Ячейки соответствующие компонентам сильной связности нарисованы оттенками зеленого. Эти ячейки показывают примерную область нахождения аттрактора. Было найдено 4 предполагаемых аттрактора: -0.983,0.061; 0.993,0.077; -1.007,-0.086; 0, -0.008;

Использованная литература

1. “Введение в символический анализ динамических систем” Г.С.Осипенко, Н.Б.Ампилова.
2. https://networkx.org/documentation/stable/reference/index.html
3. https://networkx.org/documentation/stable/reference/algorithms/generated/networkx.algorithms.components.kosaraju_strongly_connected_components.html
4. https://ru.wikipedia.org/wiki/Алгоритм_Косарайю
5. https://web.archive.org/web/20090812054837/http://rain.ifmo.ru/cat/view.php/vis/graph-general/scc-2008/algorithm
6. https://habr.com/ru/articles/537290/
7. https://ru.wikipedia.org/wiki/Компонента_сильной_связности
8. https://cyberleninka.ru/article/n/otsenka-pokazateley-lyapunova-metodamisimvolicheskogo-analiza
9. https://zetcode.com/gui/pysidetutorial/drawing/
10. https://github.com/Zenoro/ODU-solutions
11. https://www.freecodecamp.org/news/lambda-expressions-in-python/
12. https://www.geeksforgeeks.org/topological-sorting/
13. https://stackoverflow.com/questions/17200117/how-to-get-the-object-name-from-within-the-class
14. https://srinikom.github.io/pyside-docs/PySide/QtCore/QRectF.html
15. https://en.wikipedia.org/wiki/Coordinate_system
16. https://stackoverflow.com/questions/60918473/how-do-i-convert-pixel-screen-coordinates-to-cartesian-coordinates
17. https://www.pythonguis.com/tutorials/pyside6-plotting-pyqtgraph/
18. https://stackoverflow.com/questions/17200117/how-to-get-the-object-name-from-within-the-class
19. https://eltehhelp.xyz/wp-content/uploads/2021/09/image-2.png
20. https://eltehhelp.xyz/wp-content/uploads/2021/09/image-1.png