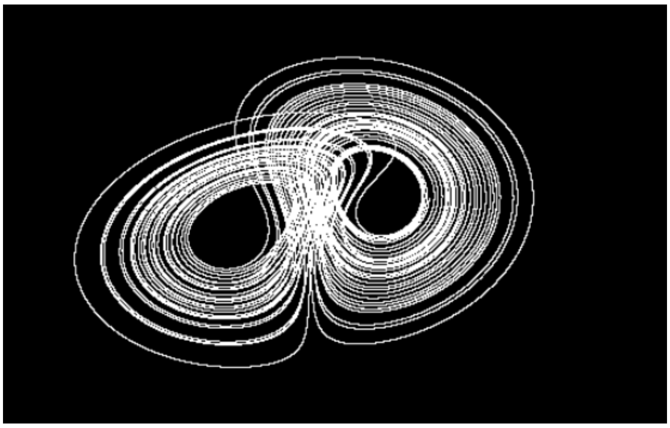

Рисунок 2 - Набор из орбит аттрактора Лоренца с заданным набором параметров σ, r, b

С геометрической точки зрения хаотические аттракторы намного сложнее классических. Их структура очень хаотична и запутанна, а порой даже не предсказуема.

«На самом деле, чем величественней наука,
Тем сильнее ощущение тайны»
В.Набоков /

ЛИТЕРАТУРА

1. Hasselblatt B., Katok A. A First Course in Dynamics: Witha Panorama of Recent Developments. Canibridge University Press. 2003.

УДК 519.1

Студ. Ю.А. Карленок
Науч. рук. доц. Е.И. Ловенецкая
(кафедра высшей математики, БГТУ)

ПОСТРОЕНИЕ МИНИМАЛЬНОГО ОСТОВНОГО ДЕРЕВА

Английским математиком Артуром Кэли была сформулирована задача: как соединить *n* городов железнодорожными линиями так, чтобы не строить лишних дорог. При этом известна стоимость строительства дороги между каждой парой городов. Требуется найти сеть дорог, соединяющую все города и имеющую минимально возможную стоимость. Эта задача относится к теории графов. Под графом понимается множество точек (вершин графа), часть из которых соединена отрезками. Отрезки, соединяющие вершины графа между собой, называются ребрами.

Пусть каждому ребру графа ставится в соответствие некоторое число – вес ребра. Это число может означать цену ребра, его длину, в нашем случае это стоимость железной дороги, которая может определяться длиной пути и дополнительными сложностями рельефа.

В теории графов выделяются понятия дерева и цикла. Деревья – графы, обладающие тем свойством, что, пройдя из одной вершины в другую, вернуться в неё снова можно лишь повернув назад. Циклы таким свойством не обладают. Вернуться в вершину можно не поворачивая назад, а пройдя полный круг цикла.

Дерево, содержащее все вершины данного графа, называется остовным деревом. Минимальное остовное дерево – остовное дерево, суммарный вес ребер которого будет минимальным.

Таким образом, задача А. Кэли – это задача построения минимального остовного дерева [1,2]. Целью данной работы была разработка программы, которая решает данную задачу для задаваемого пользователем графа.

В программе реализовано построение графов с использованием графической визуализации. Суть программы проста. На рабочей области левой клавишей мыши вы расставляете точки (максимальное число – 100). Точки нумеруются от 0 до 99. Наведя на какую-нибудь из них, можно узнать её номер. В панели слева можно вписать вес ребра, которое вы хотите создать. Введя какое-то число и нажав левой клавишей мыши на две уже имеющиеся точки, вы создадите ребро с данным весом. Если вы хотите узнать вес какого-то созданного ребра, в нижней панели-консоли можно найти данное ребро по его весу и двум вершинам (см. рис.1). Когда вы разместите все точки, свяжете их ребрами и укажете их вес, то вы можете нажать на кнопку «Добавить». После этого цветными линиями будут выделены все ребра, образующие минимальное остовное дерево данного графа (см. рис.2).

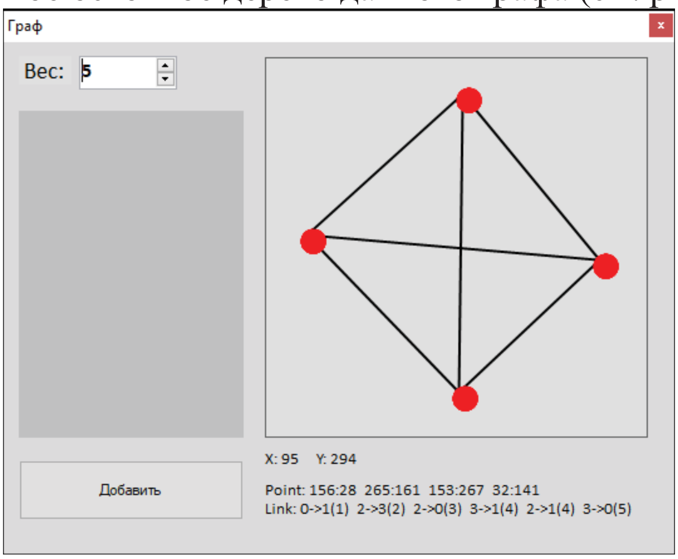

Рисунок 1 - Исходный связный граф

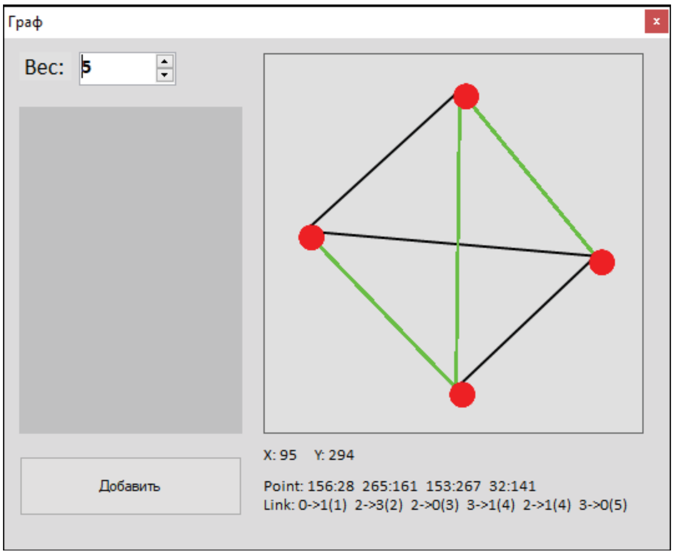

Рисунок 2 - Минимальное остовное дерево данного графа

Весь алгоритм в программе базируется на 3 массивах: двух массивах классов и одном массиве, записывающем использованные точки.

Первый массив *V* объектов класса отвечает за графическое представление точек: при щелчке по рабочей области программа определяет координаты *X* и *Y* курсора мыши и записывает их как элемент массива класса, а на рабочей области рисуется окружность заданного радиуса *R* с центром в точке(*X*; *Y*), изображающая вершину графа.

Второй массив *E* объектов класса отвечает за ребра. Он содержит в себе 3 целочисленные переменные: номера двух точек, образующих ребро, и вес этого ребра. Вводя вес ребра в текстовом поле, мы записываем показатель веса в переменную. Кликнув на 2 точки, мы «переносим» их номера в переменные, отвечающие за идентификацию этих точек.

Последний массив *T* участвует в построении минимального остовного дерева. Он имеет фиксированный размер, в отличие от двух предыдущих. Его емкость зависит от числа созданных точек. Изначально он пуст – все его ячейки имеют необрабатываемое значение – *null*.

Для построения минимального остовного дерева в программе использован метод Краскала, суть которого в следующем. Первоначально в минимальное остовное дерево добавляется ребро с минимальным весом. Далее на каждом шаге из всех оставшихся ребер снова выбираем ребро с минимальным весом и снова добавляем в дерево. Повторяем процедуру до тех пор, пока все вершины не будут соединены между собой. Важно то, что в процессе «создания» минимального остовного дерева на каждом шаге следует проверять и отбрасывать ребра, которые могут создать цикл.

При нажатии на кнопку «Добавить» начинается процесс сортировки: мы выстраиваем все ребра по весу от меньшего к большему, т.к. метод Краскала базируется на ребрах с минимальным весом. Сортировка выполняется за счет введения дополнительной переменной. Если вес первого ребра больше веса второго, то параметры первого ребра мы записываем в созданную переменную, далее первому присваиваем параметры второго, после чего во второе ребро записываем значение созданной нами переменной. Путем такого перекрестного обмена два ребра поменялись номерами. Если же вес первого ребра меньше веса второго, то обмен не нужен. Создается цикл, выполняющий этот алгоритм до тех пор, пока все ребра не будут отсортированы по возрастанию весов.

После сортировки мы можем приступить к алгоритму добавления ребер в минимальное остовное дерево. Берется первое ребро (после сортировки – ребро с минимальным весом). Его вершины записываются в массив T, отвечающий за использование точек. Берется следующее ребро. Для каждого нового ребра проводится проверка: если две его вершины уже имеются в этом массиве, то ребро откидывается и программа начнет проверять следующее ребро. Таким образом отбрасываются ребра, которые образуют циклы. Если же данных точек не встречалось или в массиве T имеется только одна точка из двух, входящих в данное ребро, то в массив записываются номера точек, задействованных в ребре, но ещё не записанных в массиве. Что касается самого ребра, то из массива ребер E мы достаем информацию о двух точках данного ребра. Далее благодаря массиву точек V мы узнаем координаты X и Y каждой из этих двух точек. Ну и теперь уже не составит труда соединить две эти точки линией другого цвета. Таким образом мы графически отметили ребро, которое входит в минимальное остовное дерево. Данный алгоритм выполняется до тех пор, пока не будут проверены все имеющиеся ребра, или пока не заполнится массив T, т.е. пока в дерево не добавятся все вершины. По завершении работы алгоритма поверх нашего исходного графа цветными линиями будут отмечены все ребра, входящие в минимальное остовное дерево.

Задача построения минимального остовного дерева широко используются на практике, в том числе при конструировании и трассировке сверхбольших интегральных схем, при проектировании коммуникаций, дорог, трубопроводов, в компьютерных и телекоммуникационных сетях.

ЛИТЕРАТУРА

1. Окулов, С.М. Дискретная математика. Теория и практика решения задач по информатике [Электронный ресурс]: учебное пособие / С.М. Окулов. – 2-е изд. (эл.). – М.: БИНОМ. Лаборатория знаний, 2012. – 422 с.

2. Белоусов, А.И. Дискретная математика: учебное пособие / А.И. Белоусов, С.Б. Ткачев. – М.: Изд-во МГТУ им. Н.Э. Баумана, 2006. – 744 с.

УДК 519.854

Студ. А.Л. Чернявский
Науч. рук. доц. Е.И. Ловенецкая
(кафедра высшей математики, БГТУ)

ЗАДАЧА ШТЕЙНЕРА

Своё название эта задача получила в честь Якоба Штейнера, швейцарского геометра, жившего в XIX веке. Это задача о соединении заданного набора точек некоторой сетью прямых отрезков так, чтобы длина этой сети была минимальной. При этом можно использовать дополнительные точки.

В настоящее время различные вариации задачи Штейнера широко используются на практике, в том числе при конструировании и трассировке сверхбольших интегральных схем, при синтезе сетей передачи данных, при проектировании коммуникаций, дорог, трубопроводов, в компьютерных и телекоммуникационных сетях и т.д. Задача Штейнера нашла применение даже в биологии при построении эволюционного дерева для группы биологических видов.

Сложность задачи Штейнера в том, что для ее решения необходимо добавить новые точки, называемые *точками Штейнера* и служащие в качестве узлов искомой кратчайшей сети.

Известны следующие свойства задачи Штейнера [1].

1. Для числа t точек Штейнера в сети выполняется неравенство $0 \le t \le n-2$, где n – количество исходных точек.

2. Каждая точка Штейнера соединена ровно с тремя другими точками, *исходными* либо *дополнительными*, причем угол между ребрами равен 120°.

3. Если некоторая вершина сети Штейнера соединена только с двумя другими точками, то угол между двумя ребрами должен быть не меньше 120°.