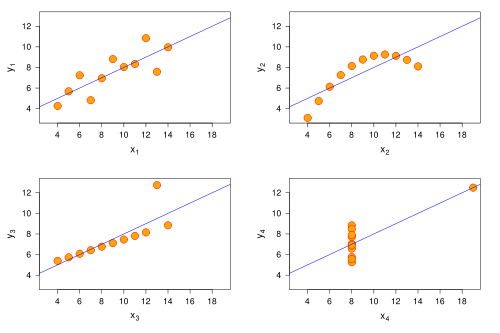
**ВВЕДЕНИЕ**

Визуализация данных - один из методов, используемых для передачи данных или информации путем их представления в виде визуальных объектов (например, точек или линий), содержащихся в графике.

Первичной целью визуализации данных является четкая и эффективная передача информации с помощью статистической графики, графиков и информационной графики (инфографики).

Визуализация данных – это и искусство, и наука. Как эстетичность и красота, так и функциональность должны идти рука об руку, обеспечивая понимание довольно разреженного и сложного набора данных, передавая его ключевые аспекты более интуитивно понятным способом.

Отличным примером важности визуализации данных является так называемый квартет Энскомба — четыре набора числовых данных, у которых простые статистические свойства (среднее значение, дисперсия, корреляция) идентичны, но их графики существенно отличаются.



Уже из этого можно сделать вывод, что при анализе данных визуализация данных не только желательна, а даже необходима.

Область применения визуализации достаточно велика – она используется в научных и статистических исследованиях (в частности, в прогнозировании, интеллектуальном анализе данных, бизнес-анализе), в педагогическом дизайне для обучения и тестирования, в новостных сводках и аналитических обзорах. Сфера анализа вредоносного программного обеспечения не стала исключением в потребности визуализации.

**1. АНАЛИЗ ВРЕДОНОСН ЫХ ЭЛЕМЕНТОВ**

Из-за растущей угрозы со стороны вредоносного программного обеспечения исследование уязвимых систем становится все более важным. Необходимость регистрации и анализа активности охватывает сети, отдельные компьютеры, а также мобильные устройства. Хотя существуют различные автоматические подходы и методы для обнаружения и идентификации вредоносных программ, фактический анализ постоянно растущего числа подозрительных образцов является трудоемким процессом для аналитиков вредоносного ПО. Использование визуализации может помочь поддержать этот процесс анализа в отношении исследования, сравнения и обобщения образцов вредоносных программ.

Существует два основных подхода к анализу вредоносного ПО: статический и динамический.

* Статический анализ описывает методы, которые не требуют фактического выполнения проверяемого образца. В зависимости от глубины анализа файл может быть проверен на его основные свойства (например, тип файла, контрольная сумма), легко извлекаемая информация (например, строки, информация об импортах DLL) или полностью разобран.
* Динамический анализ выполняет файл в главной системе, при этом различные инструменты контролируют выполнение проверяемого образца и записывают соответствующую информацию в лог выполнения. В зависимости от глубины анализа, это могут быть как простые операции файловой системы, так и целые инструкции, захваченные через отладчик.

Несмотря на то, что динамический анализ может показать больше при базисном (имеющем небольшую глубину) анализе, при более продвинутом анализе вредоносного ПО используют именно статический анализ – при помощи дизассемблеров, методов обратной разработки и так далее. Причиной этому является зависимость динамического анализа от среды анализа: данные, полученные при этом типе анализа, могут являться не полными в силу текущих свойств соответствующей системы. Однако, статический анализ проводится медленнее, так как аналитику приходится разбираться в структуре исполняемого файла.

Исследование кибератак также является насущной проблемой аналитиков на сегодняшний день. Кибератака – это любой тип наступательного маневра, который совершается отдельными лицами, группами или организациями и который атакует компьютерные информационные системы, инфраструктуру, компьютерные сети, персональные компьютеры с целью различных злонамеренных действий.

Выделяют следующие виды атак в интернете:

* Вандализм — порча интернет-страниц, замена содержания оскорбительными или пропагандистскими элементами.
* Пропаганда — рассылка обращений пропагандистского характера или вставка пропаганды в содержание других интернет-страниц.
* Сбор информации — взлом частных страниц или серверов для сбора секретной информации и/или её замены на фальшивую, полезную другому государству.
* Отказ сервиса — атаки с разных компьютеров для нарушения функционирования сайтов или компьютерных систем.
* Вмешательства в работу оборудования — атаки на компьютеры, которые занимаются контролем над работой гражданского или военного оборудования, что приводит к его отключению или поломке.
* Атаки на пункты инфраструктуры — атаки на компьютеры, обеспечивающие жизнедеятельность городов, их инфраструктуры, таких как телефонные системы, водоснабжения, электроэнергии, пожарной охраны, транспорта и так далее.

Правительства и бизнес-организации прилагают огромные усилия для обеспечения безопасности своих данных. Они используют различные типы инструментов и методов для ведения бизнеса, в то время как злоумышленники пытаются обойти существующие защиты для доступа к важным данным.

Обеспечивать безопасность систем с каждым днём становится всё сложнее из-за появления новых типов вредоносных программ. Понимание моделей атаки дает более глубокое понимание уязвимости сети, которое, в свою очередь, может использоваться для защиты сети от будущих атак.

Существует несколько целей, для которых используется анализ вредоносных элементов.

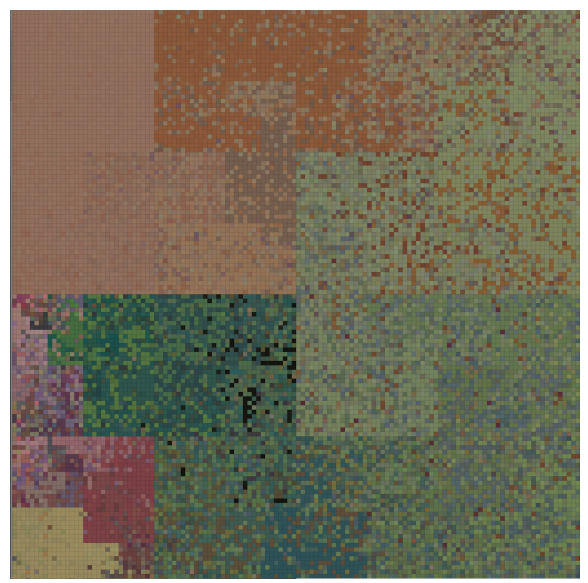
* Индивидуальный анализ используется для анализа отдельного образца вредоносного ПО, выявления целей его выполнения и пути их достижения.
* Классификация вредоносного ПО. Здесь анализ используется для определения семейства вредоносных программ, к которому относится заданный образец.
* Анализ выборки образцов. В этом случае анализ используется для выявления схожести в поведении большого количества образцов вредоносного ПО с целью получения какой-либо статистики.

**2. ПИКСЕЛЬНЫЕ МЕТОДЫ ВИЗУАЛИЗАЦИИ**

2**.1 ОБЩИЕ СВЕДЕНИЯ**

В зависимости от цели анализа существует множество различных техник визуализации данных.

Методы визуализации приобретают все большее значение при изучении и анализе больших объемов многомерной информации. Одним из важных методов визуализации, который особенно интересен для визуализации больших наборов многомерных данных, является класс пиксельных методов – dense pixel display. Основная идея пиксельных методов визуализации состоит в том, чтобы одновременно представлять как можно больше объектов данных на экране путем сопоставления каждого значения данных с пикселем экрана, учитывая его цвет и расположение.



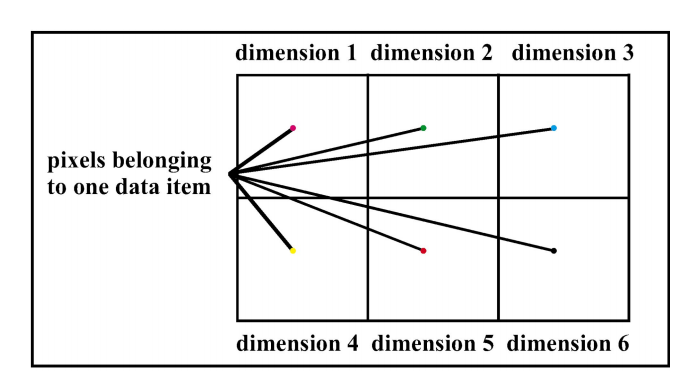
Первые очевидные плюсы такого метода визуализации заключаются в следующем:

* Отсутствие пустого места. Каждый объект представляется пикселем, так что заполнение графика происходит максимальным образом.
* Отсутствие перекрытий. Так как каждый объект занимает определенное место на графике, возможность перекрытия одним образцом другого пропадает – визуализация более понятна для восприятия.
* Можно визуализировать действительно огромные наборы данных.

+ минусы методы

**2.2 СВОЙСТВА ПИКСЕЛЬНОГО ГРАФИКА**

Все пиксельные методы визуализации разделяют экран на несколько областей – окон. Для набора данных размерности (то есть каждый образец из данных имеет измерений (атрибутов, свойств)) экран разделяется на окон – по одному на каждое измерение. В некоторых случаях происходит разделение на окно – дополнительное -ое окно используется для отображения какой-либо результирующей функции по всем остальным свойствам. Внутри окон значения данных упорядочены в соответствии с заданной общей сортировкой. Корреляции, функциональные зависимости и другие интересующие отношения между измерениями могут быть обнаружены путем связывания соответствующих областей в нескольких окнах.



При этом возникает несколько вопросов в построении графика:

1. Сопоставление значений данных с цветами: нужно тщательно спроектировать отображение значений на графике, чтобы оно стало интуитивно понятным.

2. Расположение (компоновка) пикселей внутри окон. Решение этого вопроса зависит от входных данных и задачи визуализации. В большинстве случаев, проблема расположения может быть формально описана как проблема оптимизации, и различные техники визуализации подходят для различных задач оптимизации.

3. Форма окон. Например, при прямоугольной форме окон, как показано на рисунке X, для наборов данных с большим количеством атрибутов, окна для разных измерений иногда расположены далеко друг от друга, из-за чего становится трудно найти какие-либо связи между этими атрибутами. Как и в предыдущем пункте, проблема формы окон может быть представлена как проблема оптимизации.

4. Упорядочивание окон для измерений. В большинстве случаев не существует определенного порядка расположения окон измерений на графике. Чтобы обнаружить зависимости и корреляции между атрибутами, представленными в под окнах, нужно разместить связанные измерения рядом друг с другом. Из этого снова следует проблема оптимизации.

В этой работе мы остановимся на более подробном рассмотрении только второй проблемы, а для остальных будем использовать следующие параметры:

* Цвета для отображения значений будет зависеть от двух цветов, которые задает пользователь, – начального и конечного.
  + В случае, когда атрибут представляет собой дискретную величину, разделим цветовой диапазон от начального до конечного цвета на количество уникальных элементов во входных данных. Далее каждому значению из уникальных сопоставим цвет из получившегося градиента.
  + Для непрерывных величин пользователю будет предлагаться ввести количество шагов, на которые будет разбит диапазон значений. Цветовой диапазон разбивается на такое же количество шагов, и эти два разбиения сопоставляются между собой. Таким образом, если рассматриваемая величина попадает в -ый шаг диапазона значений, она будет отображена -ым цветом в градиенте.
* Прямоугольная форма окна. Пользователю предоставляется выбор ширины и высоты графика.
* Упорядочивание окон для измерений будем проводить по порядку поступления атрибутов данных.

**2.3 РАСПОЛОЖЕНИЕ ПИКСЕЛЕЙ В ПИКСЕЛЬНОМ ГРАФИКЕ**

В этом пункте рассмотрим проблему расположения пикселей более подробно. Этот вопрос является одним из наиболее важных, поскольку только хорошее расположение позволит обнаружить кластеры или корреляции между измерениями. Для рассмотрения проблемы компоновки нужно различать наборы данных, которые имеют естественный порядок объектов (например, зависящие от времени данные – такие, как вызовы API или импорты DLL-библиотек в исполняемом файле) и наборы данных без естественного порядка.

В этой работе будем рассматривать только первый тип. Или расписать про второй тоже, но там еще сложнее?

Определим проблему расположения пикселей в пиксельном графике для набора данных с естественным порядком объектов.

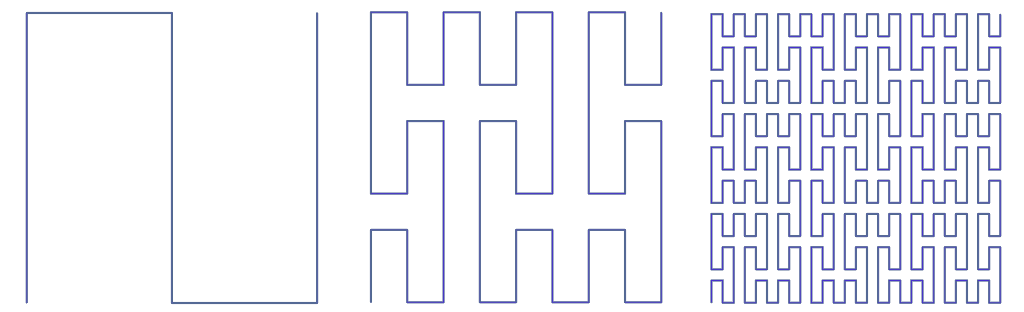
Проблема компоновки пикселей для множества объектов , расположенных в соответствии с их естественным порядком, заключается в нахождении отображения этого множества на окно размера , т.е. биективного отображения , такого, чтобы минимизировать функцию

, где

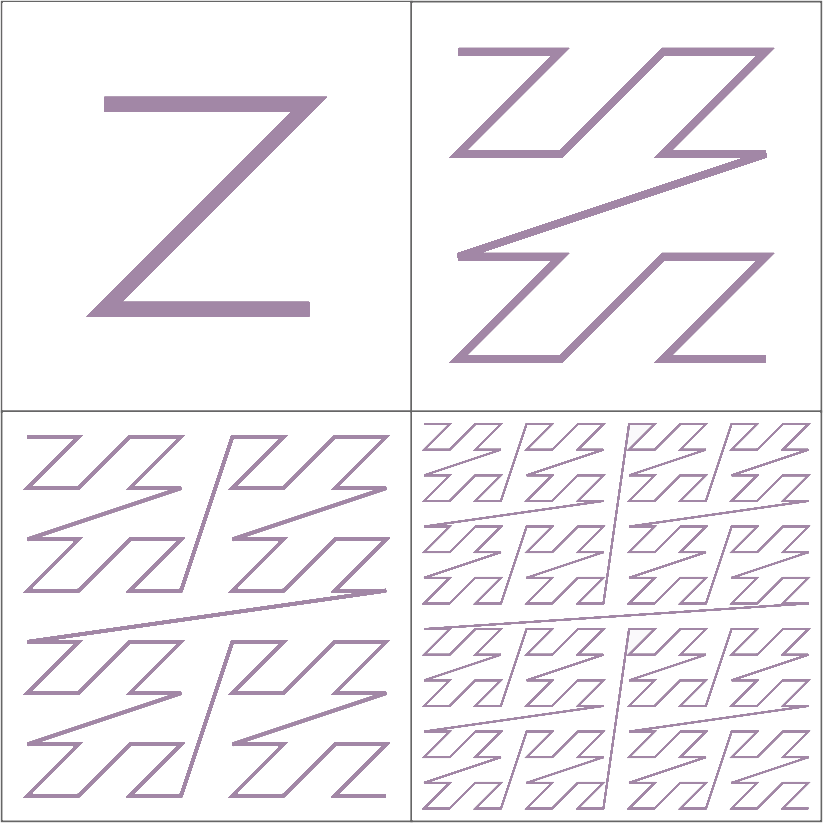
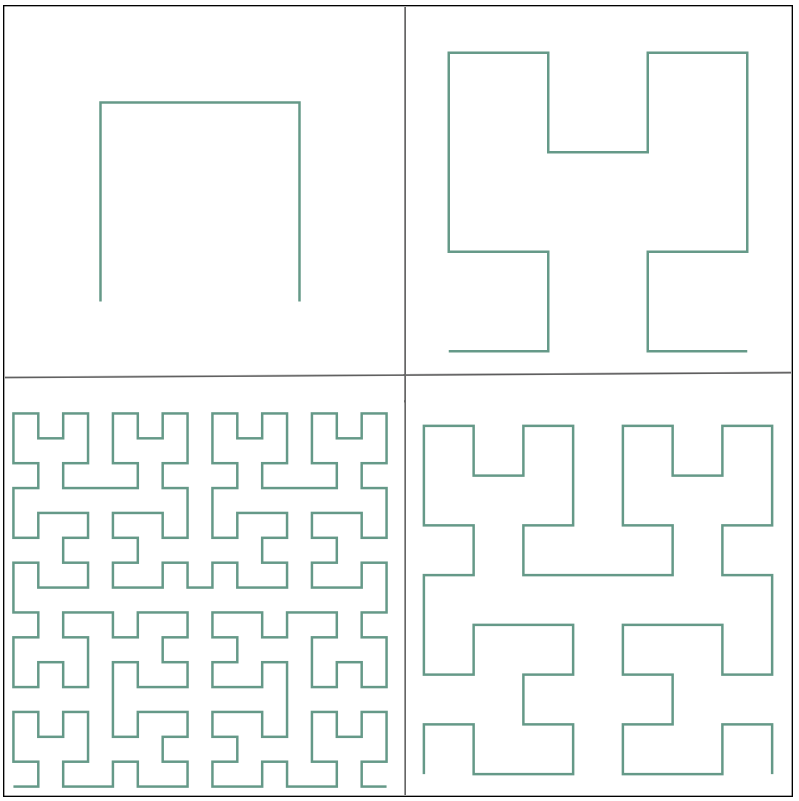
- -расстояние между пикселями, соответствующими элементам и .

Таким образом, мы находим функцию, при которой минимизируется суммарное расстояние между получившимися пикселями относительно минимального расстояния двух пикселей в прямоугольном окне размера .

Задача отображения упорядоченных одномерных наборов данных в двумерное измерение рассматривалась математиками еще в прошлом веке. Так называемые *заполняющие пространство кривые* решают в точности выше указанную задачу оптимизации и обеспечивают минимизацию заданной функции.

**

Семейство заполняющих пространство кривых, или кривые Пеано – семейство параметрических кривых, образ которых содержит квадрат (или, в более общем смысле, открытые области пространства). Другими словами, кривая представляют собой непрерывное отображение из единичного отрезка в единичный квадрат, то есть проходит через любую точку этого квадрата.



Примерами заполняющих пространство кривых являются кривые Пеано, Мортона и Гильберта.

+ Разбор примера моей визуализации сета атак.