### МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

### ФГБОУ ВО «Пермский государственный национальный исследовательский университет»

Физический факультет

Кафедра радиоэлектроники и защиты информации

АНАЛИЗ ДАННЫХ СОЦИАЛЬНЫХ СЕТЕЙ

Курсовая работа

|  |  |
| --- | --- |
|  | Работу выполнил  студент 3 курса специальности  “Информационная безопасность автоматизированных систем”  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.М. Абдульманов |
| Оценка работы научным руководителем  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка, выставленная руководителем)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата, подпись руководителя) | Научный руководитель:  к. ф.-м. н., старший преподаватель кафедры радиоэлектроники и защиты информации  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ И.Н. Черепанов |
| Оценка работы комиссией по защите курсовых работ  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка, выставленная комиссией)  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата, подпись председателя комиссии) |  |

Пермь, 2017

# ОГЛАВЛЕНИЕ

1. ВВЕДЕНИЕ

2. СРАВНЕНИЕ ПОПУЛЯРНЫХ В РОССИИ СОЦИАЛЬНЫХ СЕТЕЙ: ВКОНТАКТЕ, ОДНОКЛАССНИКИ, МОЙ МИР, FACEBOOK, TWITTER, INSTAGRAM

3. ОБЗОР ЛИТЕРАТУРЫ

4. МЕТОДИКА ИССЛЕДОВАНИЯ

5. ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

6. ЗАКЛЮЧЕНИЕ

7. СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

ПРИЛОЖЕНИЕ №1

ПРИЛОЖЕНИЕ №2

# ВВЕДЕНИЕ

В течение последних десятилетий аудитория пользователей Всемирной паутины неуклонно росла. Согласно статистике, сейчас 84 миллиона россиян в возрасте от 16 лет выходят в интернет хотя бы раз в месяц [1]. 83% из них делают это по крайней мере раз в сутки [2].

На сегодняшний день социальные сети являются самыми посещаемыми ресурсами в интернете. Это легко объяснить. В социальных сетях люди делятся фото, видео и музыкой, отслеживают мировые новости, доносят свои мысли и мнения. В общераспространенном понимании социальная сеть – это сообщество людей, объединенных общими интересами, общим делом или имеющих другие причины для непосредственного общения между собой [Воронкин]. Можно сказать, что это определение также подходит и для интернет-соцсетей.

Как уже было сказано, в таких сетях накапливается большое, даже огромное количество различного контента. И если должным образом обработать и представить эту информацию, можно получить наглядную, отражающую реальность социальную картину общества. Можно делать из этого выводы, прогнозировать поведение членов этого общества. Несомненно, разработка новых и интеграция уже существующих способов и моделей анализа данных социальных сетей - это актуальная задача, решению которой и посвящена данная работа. Ее результаты могут использоваться как база для создания различных web- и мобильных приложений, ведь сейчас в пространстве рунета существует пробел, который может заполнить эта программа.

**Цель** данной работы: создать приложение для анализа данных социальных сетей.

В связи с этим перед началом работы поставлены следующие **задачи**:

1. Рассмотреть существующие алгоритмы построения социальных графов и выбрать наиболее предпочтительный для адаптации.
2. Изучить способы обмена информацией с платформой ВКонтакте API.
3. Разработать приложение на языке Python, осуществляющее обработку списка друзей пользователя и создающего его социальный граф.
4. Собрать статистику подписчиков группы “Пермский государственный университет (ПГНИУ)” и на основании этих данных построить гистограмму, отражающую способность студентов ПГНИУ поддерживать стабильные социальные отношения, то есть уточнить значение числа Данбара для нашей выборки.

# СРАВНЕНИЕ ПОПУЛЯРНЫХ В РОССИИ СОЦИАЛЬНЫХ СЕТЕЙ: ВКОНТАКТЕ, ОДНОКЛАССНИКИ, МОЙ МИР, FACEBOOK, TWITTER, INSTAGRAM

Социальные сети стремительно ворвались в нашу жизнь. Многие люди не мыслят жизни без таких сайтов, как Facebook, Одноклассники, Вконтакте и т.д.

На таблице 1 можно видеть сравнение 6 социальных сетей, наиболее распространенных в СНГ, с точки зрения посещаемости и аудитории [3, 4].

Таблица 1. Сравнительная таблица социальных сетей

|  |  |  |  |
| --- | --- | --- | --- |
| **Социальная сеть** | **Логотип** | **Описание** | **Ежемесячная аудитория (млн человек), 2016** |
| ВКонтакте |  | ВКонтакте занимает первое место по популярности на территории Российской Федерации. Аудитория на 70-80% состоит из учащихся школ и ВУЗов и прочей молодежи, ищущей новых знакомств и развлечений. Во многом данная социальная сеть пользуется спросом, благодаря возможности загрузки и скачивания огромных объемов контента – видео, музыки, фильмов. | 87 |
| Одноклассники |  | Проект ориентирован на средне-старшую возрастную группу и у молодежи, которая является основной движущей силой интернета, спросом не пользуется. Сайт имеет не самый интуитивно понятный интерфейс, что тоже не добавляет ему привлекательности. Но идея поиска и общения бывших одноклассников и одногруппников сделала Одноклассники второй по популярности социальной сетью в России. | 73 |
| Facebook |  | Facebook является крупнейшей социальной сетью мира. Кроме того, Facebook популярен и у владельцев крупного и среднего бизнеса – более 50% крупных международных корпораций имеют официальные представительства в этой социальной сети. Российские пользователи пока еще не особо полюбили проект (из-за более привычного ВКонтакте), но их количество увеличивается с каждым годом. | 14,4 |
| Instagram |  | Бесплатное приложение для обмена фотографиями и видеозаписями с элементами социальной сети, позволяющее снимать фотографии и видео, применять к ним фильтры, а также распространять их через свой сервис и ряд других социальных сетей. Оно снискало популярность среди российских пользователей в силу таких факторов, как удобство просмотра фото, большая коллекция фильтров, отметка пользователей. | 18,5 |
| Twitter |  | Социальная сеть для публичного обмена сообщениями при помощи веб-интерфейса, SMS, средств мгновенного обмена сообщениями или сторонних программ-клиентов для пользователей интернета любого возраста. Публикация коротких заметок в формате блога получила название «микроблогинг». Твиттер часто используется для передачи новостей как личного, так и общественного значения [5]. | 7,8 |
| Мой мир |  | Знакомая всем обитателям рунета соцсеть. Вследствие огромного количества почтовых ящиков, зарегистрированных в домене @mail.ru, в Моем мире создана такая же большая база аккаунтов. Но их валидность при этом ставится под сомнение.  Один из главных плюсов платформы - взаимодействие с широким спектром других служб Mail.Ru, таких как Работа@mail.ru, Товары@mail.ru, Знакомства@mail.ru и др. | 16,1 |

Из таблицы можно сделать вывод: универсальной социальной сети нет, но можно пользоваться с той, которая будет наиболее удобна для нужд конкретного человека.

Все они являются кроссплатформенными и поддерживают свои API [6].

* **API** – коллекция готовых HTTP-запросов, а также определение структуры HTTP-ответов, для выражения которых используют XML или JSON форматы. С помощью API программисты пишут приложения, значительно расширяющие возможности сайта.

В своей работе я выбрал для анализа данных социальную сеть ВКонтакте. Она имеет самую большую базу пользователей в России и привычный интерфейс. Также VK API имеет все необходимые для моих нужд методы.

# ОБЗОР ЛИТЕРАТУРЫ

Е. Д. Патаркин дает следующее определение социальной сети [7]:

**Социальная сеть** – это платформа, на базе которой участники могут устанавливать отношения друг с другом.

Тогда социальная интернет-сеть – это интерактивный многопользовательский веб-сайт, обладающий рядом обязательных **качеств**:

* содержание (контент) сайта создается исключительно или преимущественно его пользователями;
* сайт представляет собой автоматизированную среду, в рамках которой пользователи имеют возможность создавать связи с другими пользователями (социальные связи) или социальные объекты (тематические группы);
* пользователи имеют возможность получать статическую и динамическую информацию об объектах, существующих в данной социальной среде, о социальных связях между ними;
* пользователям доступны функции коммуникации с другими пользователями и социальными объектами.

**Функции** социальных сетей:

1. Коммуникационная.

В рамках коммуникационной функции люди устанавливают контакты, обмениваются новостями, информацией (фото, видео, аудиоматериалы, ссылки на сайты, комментарии, сообщения), кооперируются для достижения совместных целей (сплочение и удержание социальных связей).

1. Информационная.

Поток информации имеет двустороннюю направленность, т.к. участники общения выступают попеременно и в роли коммуникатора, и в роли реципиента.

1. Социализирующая.

Саморазвитие, рефлексия в системе «друзей» и «групп»).

1. Самоактуализирующая (самопрезентация).
2. Идентификационная.

При создании индивидуального профиля пользователь наполняет его информацией о себе – имя, дата рождения, семейный статус, школа, ВУЗ, интересы и пр., что позволяет осуществлять поиск анкет по заданным признакам.

1. Функция формирования идентичности.

Согласно теории Фестингера (1954 г.), человек склонен сравнивать себя с теми людьми, с которыми у него есть большее количество схожих черт. Кроме того, согласно теории когнитивного диссонанса, похожие люди положительно оценивают друг друга. Это основной механизм, который позволяет человеку четко формулировать свои позиции в отношении других людей и групп.

1. Развлекательная.

Социальные сети позволяют обмениваться не только текстовыми сообщениями, но и мультимедийными файлами, кроме того, следует отметить значение виджетов – мини-программ развлекательного характера, создаваемых сторонними производителями для расширения возможностей пользователя (игры, медиа-приложения и т.д.).

Осознание этих качеств и функций позволяет нам четко представлять, над чем предстоит дальнейшая работа.

Ценность социальной сети можно определить и количественно. Директор и один из основателей американской Национальной Радиовещательной Компании Д. Сарнов [8] показал, что ценность телевизионной (радиовещательной) сети возрастает прямопропорционально числу их зрителей (слушателей) n. Таким образом, ценность сети тем выше, чем больше число ее элементов (узлов). Узлы же социальных сетей связаны между собой, что выгодно отличает ее от моноцентричных сетей, например, телевизионной. Г. Рейндгольд отметил [8]: «Когда сеть похожа на телевизионную и вещает что-то людям, ценность ее услуг возрастает линейно. Когда же сеть дает возможность отдельным узлам вступать в контакт друг с другом, ценность возрастает в квадратичной зависимости. А когда та же самая сеть располагает средствами для создания ее участниками целых групп, ценность возрастает экспоненциально»

Закон Меткалфа гласит, что полезность сети пропорциональна квадрату численности пользователей этой сети ≈n2 [9]. То есть ценность всей системы возрастает даже быстрее, чем число ее узлов n. Учитывая то, что каждый узел сети может быть соединен с n−1 другими узлами, то ценность для него пропорциональна n−1. Поскольку в сети всего n узлов, то ценность всей сети пропорциональна n(n−1). Под ценностью здесь понимается количество возможных подключений [10] (рис. 1).

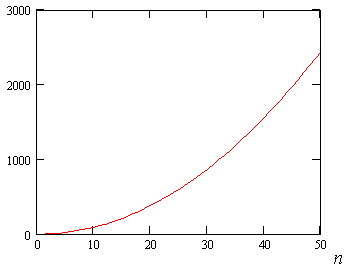


Рисунок 1. Иллюстрация графической зависимости ценности социальной сети при увеличении числа ее участников от 0 до 50 в приложении к законам Меткалфа.

На основании высокой полезности социальных сетей с точки зрения скорости распространения информации можно сделать вывод, что необходимость исследований в этом направлении очевидна. И они действительно проводятся. Возьмем для рассмотрения алгоритм построения графа дружеских связей.

Википедия [14] предлагает следующее определение графа дружеских связей (социального графа):

* **Социальный граф —** граф, узлы которого представлены социальными объектами, такими как пользовательские профили с различными атрибутами (например: имя, день рождения, родной город), сообщества, медиаконтента и так далее, а рёбра — социальными связями между ними.

На специализированных ресурсах вроде Habrahabr [13] или GeekTimes приводится множество его реализаций. На рисунках 2 и 3 показаны кластеризованные социальные графы, визуализированные с помощью платформ Gephi и Wolfram Mathematica.

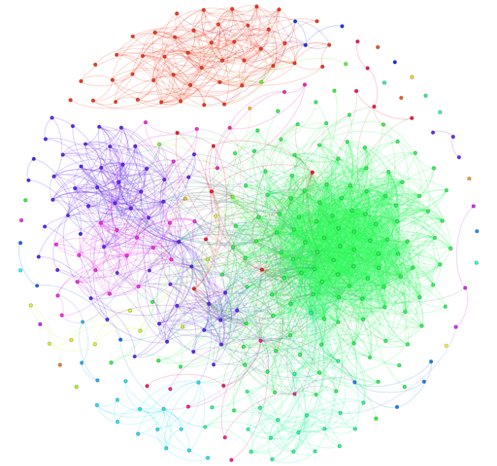


Рисунок 2. Кластеризованный социальный граф, построенный с помощью платформы Gephi

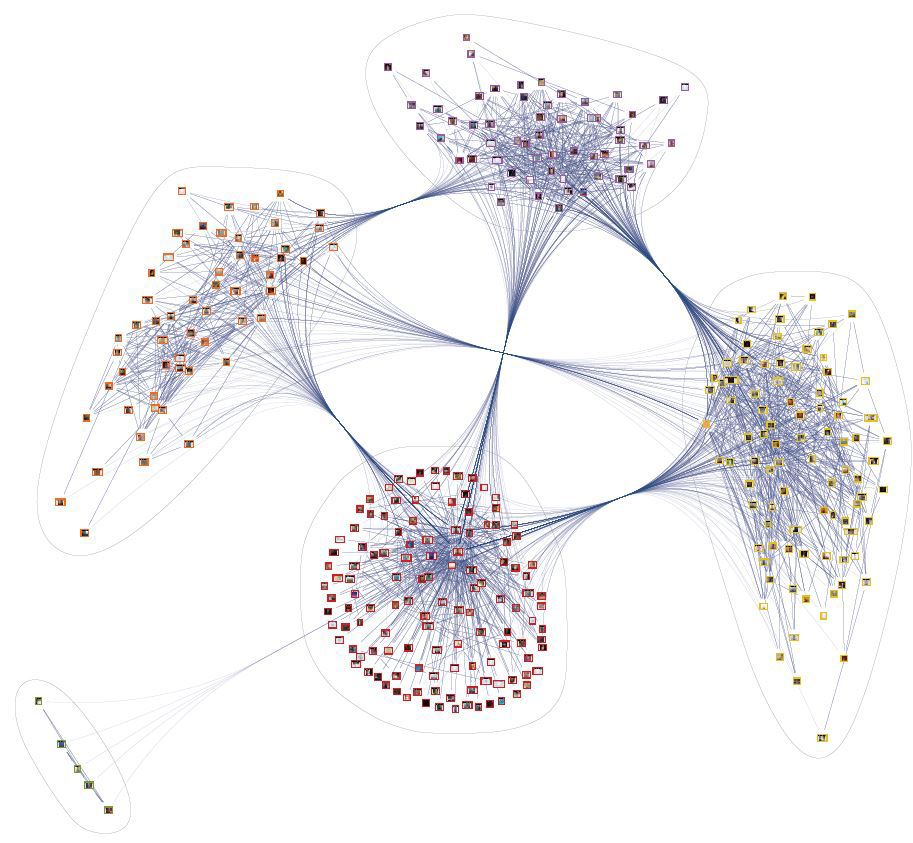


Рисунок 3. Кластеризованный социальный граф, построенный с помощью платформы Wolfram Mathematica

С помощью социальных графов решают такие задачи, как: идентификация пользователей; социальный поиск; генерация рекомендаций по выбору «друзей», медиа-контента, новостей; выявление «реальных» связей или сбор открытой информации для моделирования графа. В своей работе я попытаюсь создать программу, способную визуализировать социальные связи любого пользователя сайта ВКонтакте.

Большой интерес представляет такой универсальный показатель, как **число Данбара** (рис. 4).



Рисунок 4. Антрополог Робин Данбар

С глубокой древности до настоящего времени люди объединялись в группы, и когда эти группы достигают численности примерно 150, их эффективность резко падает, вследствие чего им приходится дробиться на более мелкие подразделения, создавать четкую иерархию. Этими подразделениями могут быть военные отряды, филиалы фирм или другие социальные группы. Люди качественнее выполняют задачи, будучи частью команды (кластера). При этом подавляющее большинство членов кластера должно быть связано преимущественно только с другими членами этого же кластера, и только отдельные представители соединяют этот кластер с другими.

* **Число Данбара** – это ограниченное число связей, которые человек может стабильно поддерживать [15].

Это число не является константой. В разных условиях оно варьируется от 120 до 230, но в большинстве ситуаций исследователи принимают число Данбара равным **150**. В своей работе я попытаюсь вычислить данный показатель для студентов Пермского университета и сравнить его с общепринятым.

# МЕТОДИКА ИССЛЕДОВАНИЯ

Как уже было сказано, целью данной работы является создание приложения, которое визуализирует два объекта:

1. Социальный граф дружеских связей
2. Гистограмма, иллюстрирующая, какое максимальное количество дружеских связей способны поддерживать члены социальной сети (расчет числа Данбара).

### 4.1. СОЗДАНИЕ СОЦИАЛЬНОГО ГРАФА

Задача реализации социального графа делится на несколько подзадач [11]:

1. Получение токена
2. Получение данных
3. Визуализация графа

Для этого будут использоваться следующие инструменты:

1. Язык программирования Python 3.6
2. Библиотека requests
3. Библиотека d3.js
4. Веб-интерфейс VK API
5. Браузер Mozilla Firefox для просмотра созданного графа

Вот несколько причин выбрать в качестве языка программирования именно Python:

* Python вышел на 1-е место для обучения программированию в университетах США
* Python входит в топ-5 по популярности и стремительно поднимается в этом рейтинге
* Хорошее знание языка Python никогда не оставит программиста без работы
* Большое количество библиотек на все случаи жизни, постоянно пополняемых за счет сообщества

В качестве браузера был выбран Mozilla Firefox в силу того, что в нем можно   
делать HTTP-запросы к серверу без перезагрузки страницы (объект XMLHTTPRequest). Библиотеки requests и d3 будут использоваться для выполнения запросов к api.vk.com и для визуализации графа соответственно. В вопросе взаимодействия с сайтом выбор между использованием VK API и ручным парсингом HTML очевиден: VK API намного удобнее.

Для работы с VK API необходимо получить токен. *Access token* – это уникальный ключ доступа, позволяющий точно идентифицировать аккаунт. Он представляет собой строку из латинских букв и цифр. Для его получения необходимо создать свое Standalone-приложение ВКонтакте и в настройках найти его ID. Затем вставить номер приложения в специальный запрос, который следует ввести в адресную строку браузера:

https://oauth.vk.com/authorize?client\_id=ID\_приложения&

scope=photos,audio,video,docs,notes,pages,status,offers,questions

,wall,groups,messages,email,notifications,stats,ads,offline

,docs,pages,stats,notifications&response\_type=token

После отправки запроса серверу и подтверждения всех разрешений он вернет строку, содержащую токен, а также предупредит о том, что его потеря может привести к потере доступа к аккаунту, поэтому пользователь должен хранить свой токен в тайне. Данный access\_token можно использовать в приложении для получения разнообразной от сервера ВКонтакте.

Для получения данных, необходимых для создания графа, требуется использовать несколько методов, имеющихся в пакете VK API. Необходимые методы представлены в таблице 2.

Таблица 2. Некоторые методы пакета VK API

|  |  |
| --- | --- |
| **Метод** | **Описание** |
| users.get() | Принимает как один или несколько ID и список запрашиваемых полей, например, фото, и возвращается значения этих полей |
| friends.getMutual() | Принимает два идентификатора: source\_id и target\_id. Метод возвращает список ID общих друзей между парой пользователей. |
| friends.get() | Принимая ID пользователя, возвращает список ID его друзей и расширенную информацию о них. |
| execute | Универсальный метод, который позволяет запускать последовательность других методов, сохраняя и фильтруя промежуточные результаты. |

Метод Execute хорош тем, что позволяет разом выполнить до 25 обращений к методам API. Это позволяет обойти ограничение сайта ВКонтакте на максимальное число запросов в секунду (не более трех).

*Пара слов о теории графов.*

* **Теория графов** – раздел дискретной математики, изучающий свойства графов. В общем смысле граф представляется как множество вершин (узлов), соединённых рёбрами.

В социальной интернет-сети в вершинах находятся пользователи, ребрам соответствует отношение «друзья» (рис. 5).



Рисунок 5. Модель социального графа

Алгоритм получения данных представлен на блок-схеме (рис. 6).

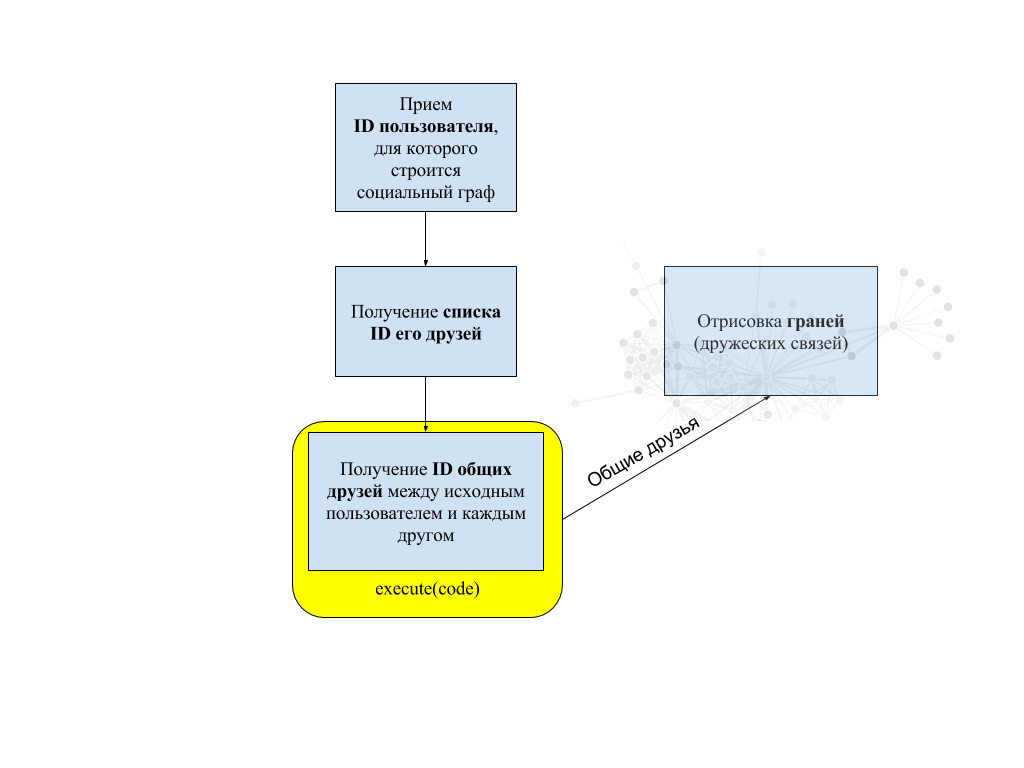


Рисунок 6. Алгоритм получения данных для создания социального графа

Затем на основании этой информации генерируется json-файл, который затем будет использоваться при создании веб-страницы с анимированным графом. В нем есть поля links и nodes, хранящие информацию о вершинах (пользователях) и связях между ними (дружеских отношениях).

* **JSON** (JavaScript Object Notation) - простой формат обмена данными, удобный для чтения и написания как человеком, так и компьютером. Основан на подмножестве языка программирования JavaScript.

Пример созданного json-файла представлен ниже:

{"links": [{"source": 161, "target": 0, "value": 1}, {"source": 36, "target": 0, "value": 1} ]… {"nodes": ["group": 1, "name": "Дмитрий Нечаев", "photo": "https://pp.userapi.com/c636521/v636521570/3820c/\_PG6Hn6eo3U.jpg"}]}

Для отображения этого графика в браузере также необходимы файлы d3.v3.min.js и index.html.

Список кастомизируемых параметров файла index.html представлен на таблице 3:

Таблица 3. Список наиболее важных кастомизируемых параметров файла index.html

|  |  |  |
| --- | --- | --- |
| **Параметр** | **Используется** | **Описание параметра** |
| width | 2000 px | Длина графа в окне браузера |
| height | 2000 px | Высота графа в окне браузера |
| d3.layout.force().linkDistance(P) | P = 5 | Длина отображаемых связей |
| .link {fill: none; stroke: #P;} | P = #825ac7 | Цвет связей |
| node.append("image")  .attr("xlink:href",function(d) {return d.photo;})  .attr("x", P1)  .attr("y", P2)  .attr("width", P3)  .attr("height", P4); | P1 = -8  P2 = -8  P3 = 16  P4 = 16 | Определяет положение изображения по отношению к точке (вершине) |
| .node text { pointer-events: none;  font: P; } | P = 10px sans-serif | Кегль и шрифт текста (имя, фамилия) |

Затем остается только открыть html-файл браузером Mozilla Firefox (или любым другим, предварительно выполнив команду ‘python -m http.server 8000’).

Интересно, что первым данный вид графов (Force-Directed Graph) в 1993 году скомпилировал Дональд Кнут, один из известнейших ученых в области информатики, автор монументальной монографии «Искусство программирования» [16]. Он составил граф взаимосвязей героев романа Виктора Гюго «Отверженные» (рис. 7).

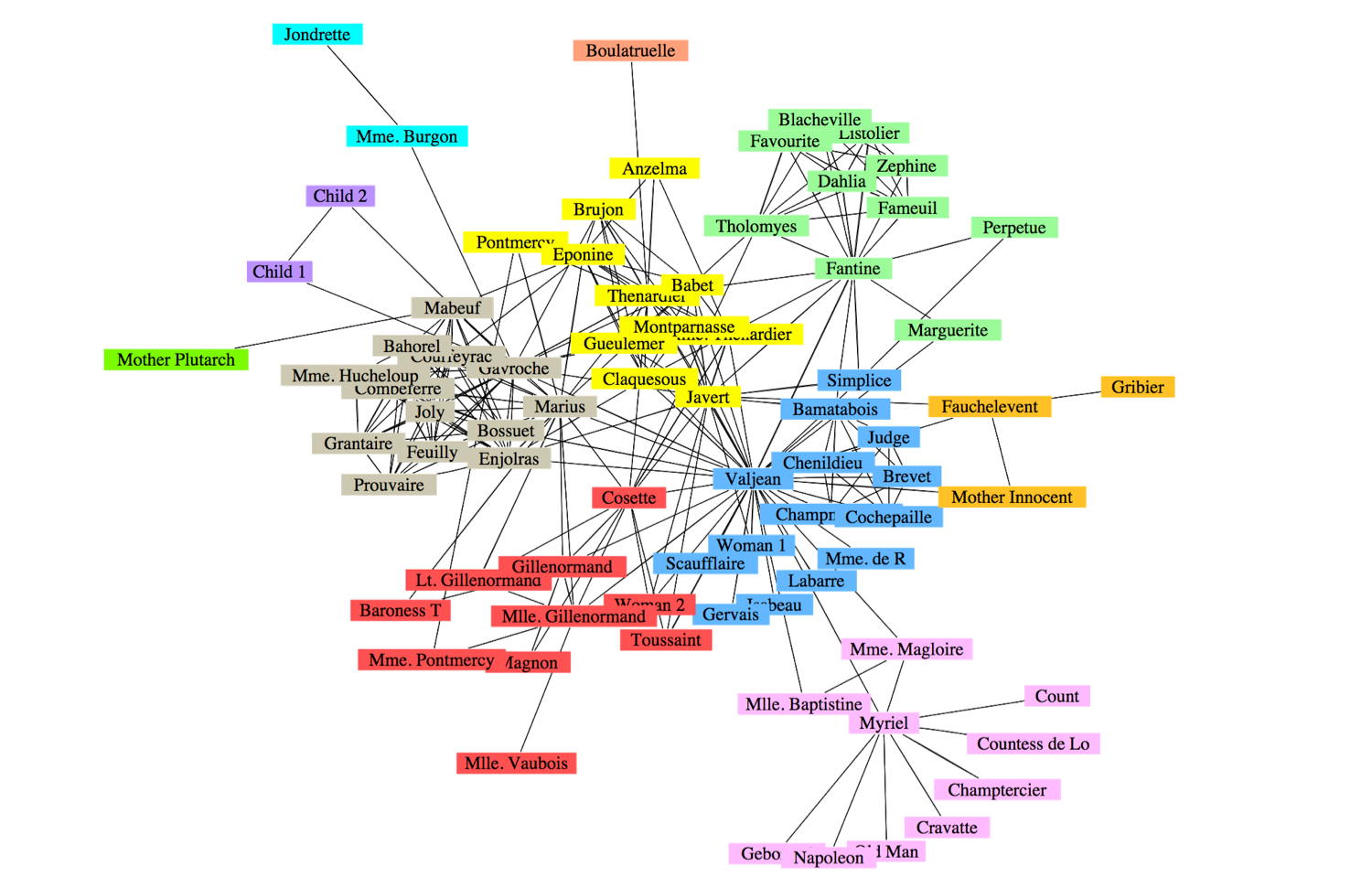


Рисунок 7. Герои романа “Отверженные” на социальном графе

### 4.2. ВЫЧИСЛЕНИЕ ЧИСЛА ДАНБАРА

Для вычисления числа Данбара необходимо совершить несколько действий:

1. Получение данных о количестве друзей пользователей группы «Пермский государственный университет»
2. Сохранение их в файл
3. Анализ данных с использованием методов математической статистики
4. Создание гистограммы

Для этого будут использоваться следующие инструменты:

1. Язык программирования Python 3.6
2. Надстройка языка программирования JavaScript – VKScript
3. Библиотека requests
4. Web-интерфейс VK API
5. Microsoft Excel
6. Web-сервис создания графиков Datavisual

Кажущийся довольно тривиальным Microsoft Excel вполне подходит для обработки больших массивов данных, предлагая при этом мощные статистические методы, поэтому было решено остановиться на этом знакомом инструменте. Существует много онлайн-платформ, предлагающих построение графиков и гистограмм, например, infogr.am, onlinecharts.ru и т.д., но я решил попробовать новый сервис Datavisual, потому что он не требует регистрации и имеет простой интерфейс.

Для получения списка пользователей группы логично было бы воспользоваться методом groups.getMembers, но VK API за один запрос возвращает не более 1000 участников, а в секунду можно сделать не более 3 запросов. Чтобы поднять скорость получения данных с 3000 до 100000, нужно снова воспользоваться уже знакомым методом execute, запускающимся из хранимой процедуры на сервере ВКонтакте. Процедура принимает id группы и возвращает список с данными ее пользователей. Подсчет 12000 подписчиков ПГНИУ происходит меньше чем за секунду.

Основную сложность в этой задаче представляет подсчет количества друзей у каждого пользователя и занесение их в список. В ходе работы мне не удалось обойти ограничение VK API, поэтому получение числа друзей осуществлялось обычным запросом friends.get, принимающим на входе id человека.

После заполнения списка этот массив экспортируется в файл и открывается программой для работы с электронными таблицами (Microsoft Excel).

*Пара слов о математической статистике.*

Более 10000 значений, полученных в результате предыдущих операций, необходимо преобразовать, чтобы получить группированный статистический ряд абсолютных частот.

**Группированным статистическим рядом абсолютных частот** (рис.8) называется последовательность пар чисел

где — центр k-го интервала группировки и — число элементов выборки, попавших в k-й интервал.

Числа ( k = 1,…,m ) называются **абсолютными частотами**.

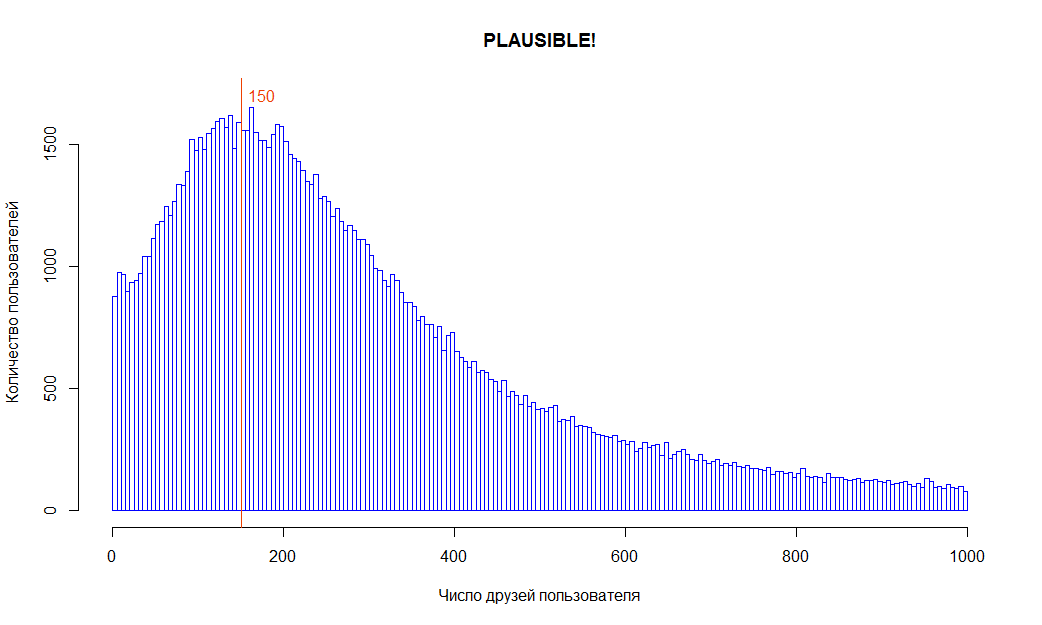


Рисунок 8. Пример группированного статистического ряда абсолютных частот

Алгоритм построения группированного статистического ряда абсолютных частот [12]:

1. Находим минимальный и максимальный элемент выборки. В данной работе это xmin = 0 и xmax = 1000. Значения элементов выборки ограничены тысячей, а хвост из пользователей с [1001,9999] друзей отброшен, т.к., скорее всего, это известные личности или аккаунты с накрученными друзьями, а они будут только искажать результат.
2. Находим длину интервала группировки h = = 1000 / 200 = 5.

Здесь m = 200 - число интервалов группировки, взято из соображений наглядности.

1. Находим правые границы интервалов группировки: xk = xmin + kh (k = 1,..., 200).
2. Находим центры ẋk интервалов группировки по формуле: ẋk = ẋk - h/2 (k = 1,..., 200).
3. Для каждого интервала группировки (xk-1 , xk) находим число ṅk элементов выборки, попавших в этот интервал.

Число ṅk – это и есть количество пользователей с соответствующим числом друзей. Например, если при ẋk = 128 значение ṅk = 144, то это означает, что из всей выборки в 10000 человек у 144 человек по 128±2,5 друзей.

На рис. 9 показано окно программы Microsoft Excel с расчетными данными, на основании которых построена гистограмма, предназначенная уточнить число Данбара для студентов ПГНИУ.



Рисунок 9. Окно программы Microsoft Excel с расчетными данными, необходимыми для построения графика группированного статистического ряда абсолютных частот.

# ПОЛУЧЕННЫЕ РЕЗУЛЬТАТЫ

### СОЗДАНИЕ СОЦИАЛЬНОГО ГРАФА

Созданное приложение быстро справляется с поставленной задачей: составить json-файл со значениями вершин и связей. После нескольких тестов с разными аккаунтами среднее время выполнения операции составило 3.5 секунд. Это хороший результат, который удалось оптимизировать благодаря использованию метода Execute. Для пользователя с десятью друзьями операция завершилась менее, чем за секунду. Работу приложения можно увидеть на рис. 10

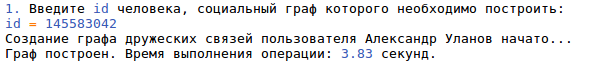


Рисунок 10. Работа приложения, собирающего данные для создания социального графа

Используя полученный json-файл, в окне браузера открывается граф, содержащий определенное количество вершин и связей между ними. Каждой вершине соответствует друг исходного пользователя, каждой связи – отношение «дружба» между его друзьями. Для облегчения идентификации пользователя используется его фотография и имя (рис. 11).

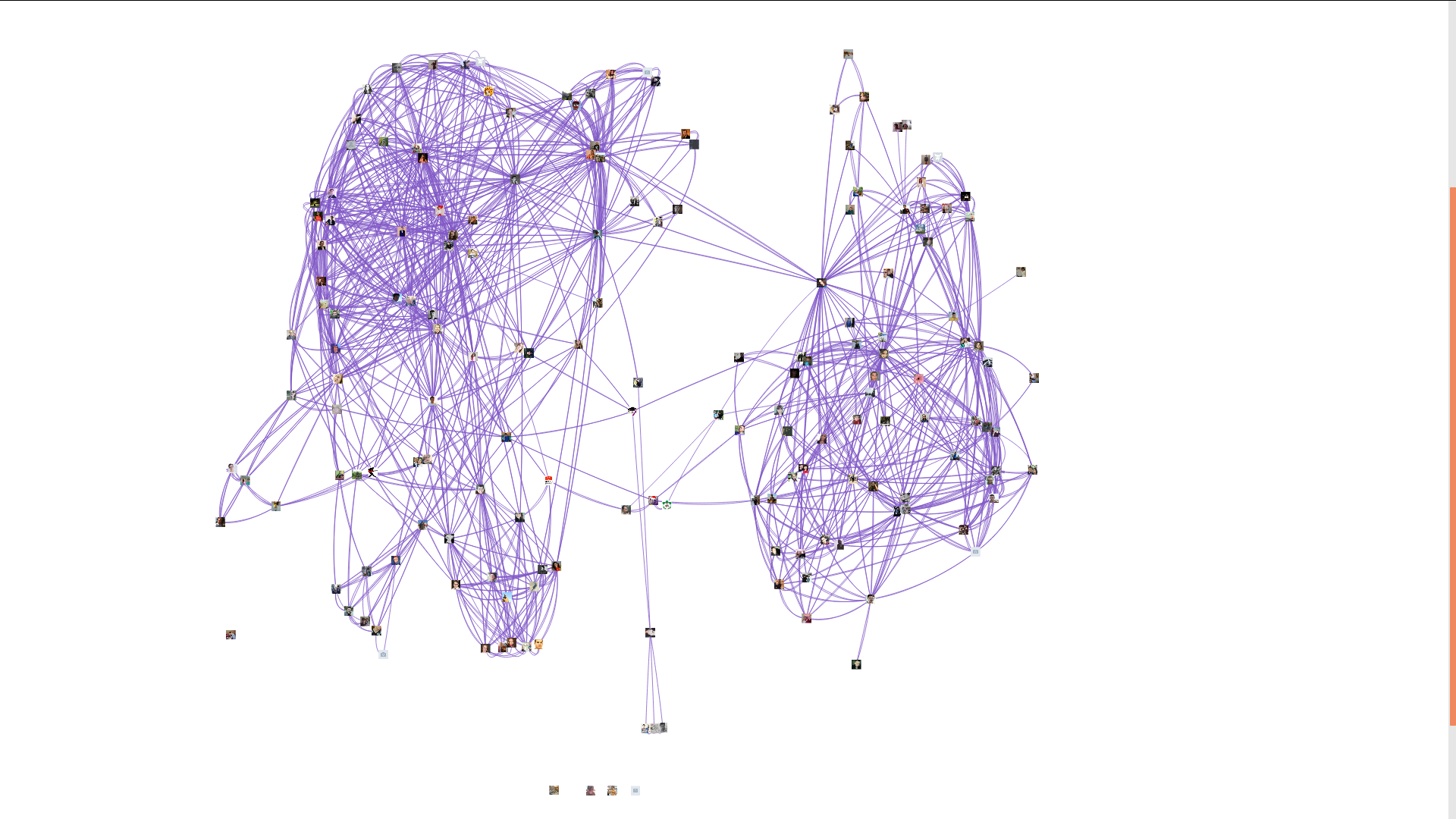


Рисунок 11. Социальный граф дружеских связей пользователя Александр Уланов

Элементы интерактивны. Вершины можно передвигать, но затем они возвращаются практически в исходное положение, чтобы не нарушить структуру графа. В местах большого скопления связей можно выделить кластеры, внутри которых находятся люди связанные определенной характеристикой, например члены одной семьи, односельчане или учащиеся одного ВУЗа (рис. 12).

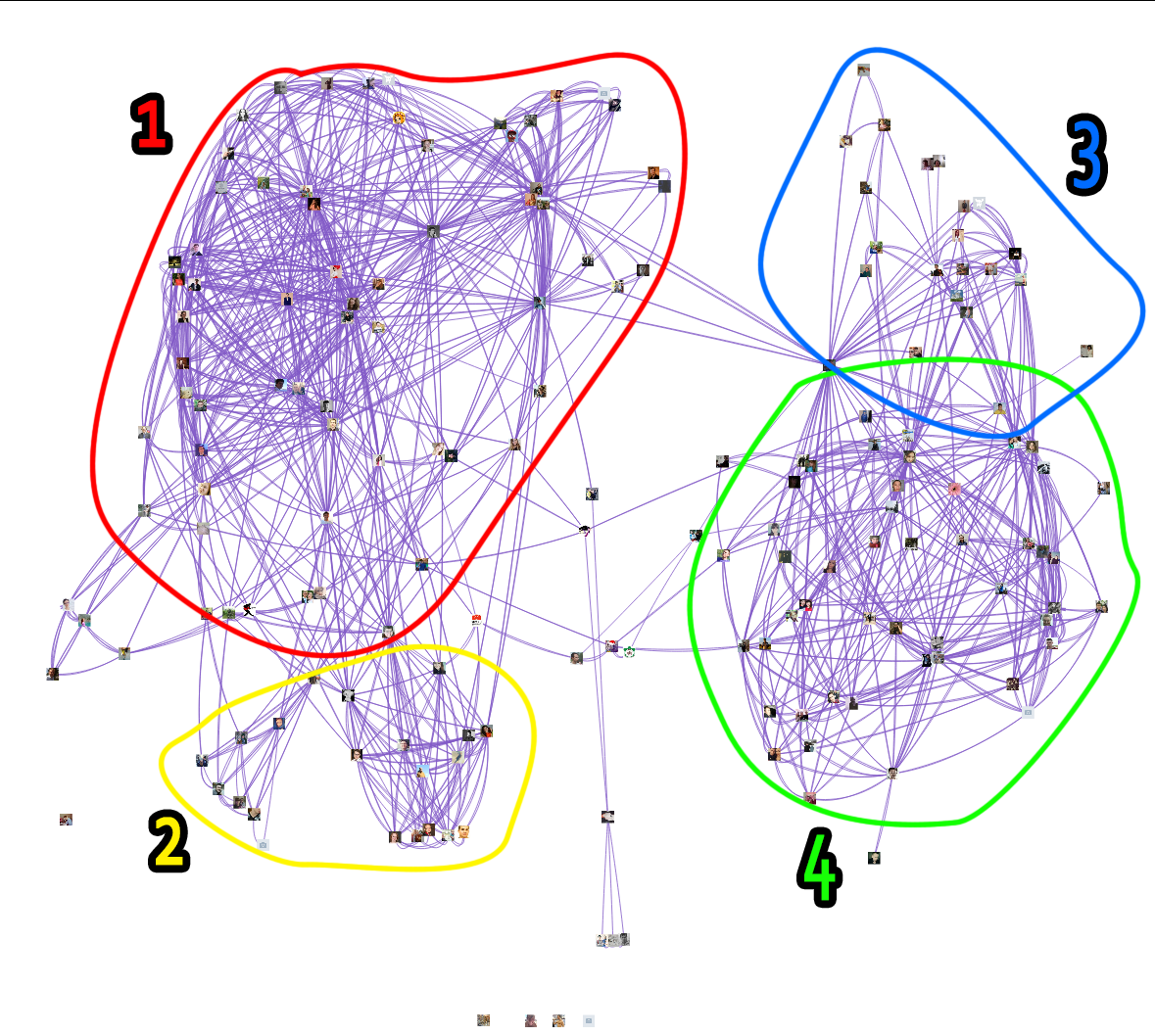


Рисунок 12. Социальный граф дружеских связей пользователя Александр Уланов с кластерами.

1 - учащиеся одного ВУЗа, 2 - сообщество фотографов, 3 - семья, 4- односельчане

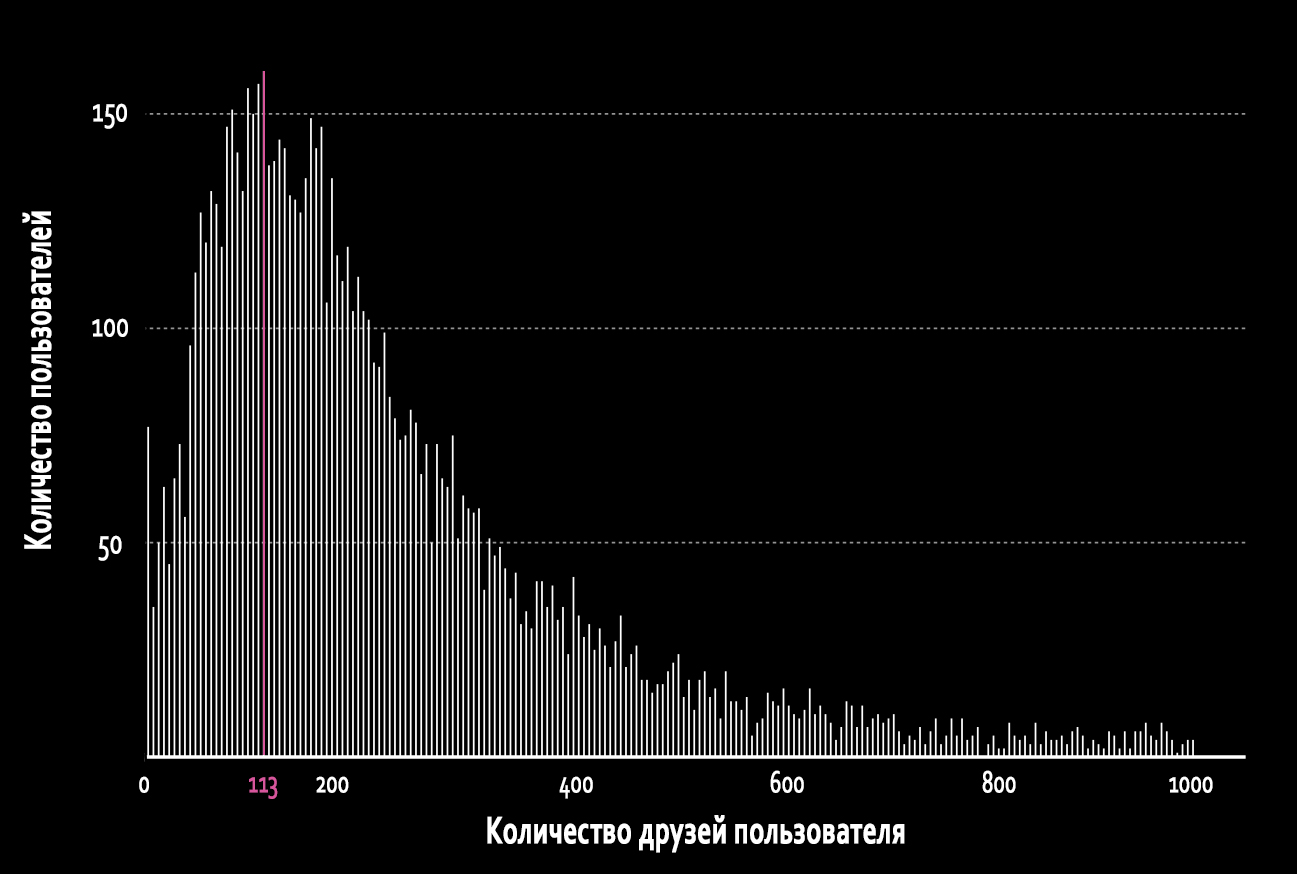
### 5.2. ВЫЧИСЛЕНИЕ ЧИСЛА ДАНБАРА

Как можно увидеть на рис. Для 11585 активных пользователей группы «Пермский государственный университет (ПГНИУ)» процесс создания списка друзей занял 77 минут. Как можно заметить, из общей выборки с помощью исключения были убраны 636 удаленных и заблокированных пользователей. Разработанное приложение на протяжении всей своей работы выводило сообщения, позволяющие отслеживать удаленных пользователей и общее число обработанных аккаунтов (рис. 13).



Рисунок 13. Работа программы VKDataAnalysys

В результате анализа полученной информации в Microsoft Excel и экспорта таблицы в сервис DataVisual представилось следующее распределение (рис. 14):



Проанализировав данные более 10 тысяч участников группы «Пермский государственный университет (ПГНИУ)», мы пришли к выводу, что максимальное количество пользователей группы поддерживают стабильные социальные отношения только с 113 людьми. Это отличается то стандартной оценки в 150 человек, что может быть связано с ограниченностью и неравномерностью выборки. Но на гистограмме также можно заметить, что большое количество людей имеют около 200 друзей. Аппроксимируя эти значения, можно прийти к привычному значению – 150 человек.

Конечно, было бы лучше использовать какую-нибудь другую метрику, чтобы улучшить качество выборки, например, регулярность личной переписки или выставление оценок «Мне нравится», но получить первое не представляется возможным без согласия пользователей, а отсутствие второго не гарантирует прекращение дружеских отношений.

# ЗАКЛЮЧЕНИЕ

Цель моей курсовой работы можно считать выполненной: было создано приложение для анализа данных социальных сетей. Оно выполняет две функции: строит социальный граф дружеских связей заданного пользователя и на основе статистики подписчиков группы «Пермский государственный университет (ПГНИУ)» выдает готовые данные, которые потом можно использовать для визуализации этой информации, а именно построения гистограммы, отражающую способность студентов ПГНИУ поддерживать стабильные социальные отношения.

В ходе работы я изучил способы обмена информацией с платформой ВКонтакте API, стал гораздо лучше разбираться в тонкостях языка Python, впервые познакомился с синтаксисом языка JavaScript, когда возникла необходимость использовать его ответвление (VKScript) для создания хранимой процедуры.

Также одним из главных итогов написания этой курсовой работы я считаю получение опыта использования академических знаний в области программирования для решения прикладных задач, связанных с социологией, математической статистикой и теорией графов.

# CПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Интернет в России: динамика проникновения. Зима 2015 - 2016 гг. [Online] [Cited: Апрель 24, 2017.] <http://fom.ru/SMI-i-internet/12610>
2. Самые популярные социальные сети в России 2016 [Online] [Cited: Апрель 24, 2017.] <http://www.pro-smm.com/populyarnye-socialnye-seti-v-rossii-2016/>
3. Annual Report Mail.ru Group [Online] [Апрель 24, 2017.] <https://corp.imgsmail.ru/media/files/mail.rugrouparfy2015.pdf>
4. Какая социальная сеть лучше: ВКонтакте, Одноклассники или Facebook? [Online] [Cited: Апрель 24, 2017.] <http://livelenta.com/kakaya-socialnaya-set-luchshe-vkontakte-odnoklassniki-ili-facebook.html>
5. Твиттер [Online] [Cited: Апрель 24, 2017.] [https://ru.wikipedia.org/wiki/Твиттер](https://ru.wikipedia.org/wiki/%D0%A2%D0%B2%D0%B8%D1%82%D1%82%D0%B5%D1%80)
6. API [Online] [Cited: Апрель 24, 2017.] <https://ru.wikipedia.org/wiki/API>
7. Патаракин Е. Д. Педагогический дизайн социальной сети Scratch // Образовательные технологии и общество (Educational Technology & Society). – 2013. – Т.16. – № 2. – С. 505–528.
8. Г. Рейнгольд. Умная толпа: новая социальная революция. – М.: ФАИР ПРЕСС, 2006. – 416 с.
9. Закон Меткалфа [Online] [Cited: Апрель 24, 2017.]

[https://ru.wikipedia.org/wiki/Закон\_Меткалфа](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BA%D0%BE%D0%BD_%D0%9C%D0%B5%D1%82%D0%BA%D0%B0%D0%BB%D1%84%D0%B0)

1. А.С. Воронкин Социальные сети: эволюция, структура, анализ // Образовательные технологии и общество (Educational Technology & Society). – 2014. – Т. 17. – № 1. – С. 15
2. Анализ дружеских связей VK с помощью Python [Online] [Cited: Апрель 24, 2017.] <https://habrahabr.ru/post/221251/>
3. Статическая обработка выборки [Online] [Cited: Апрель 24, 2017.] <http://www.math-pr.com/exampl_sts1.html>
4. Анализ дружеских связей VK с помощью Python [Online] [Cited: Апрель 24, 2017.] <https://habrahabr.ru/post/221251/>
5. Социальный граф [Online] [Cited: Апрель 24, 2017.] [https://ru.wikipedia.org/wiki/Социальный\_граф](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D1%86%D0%B8%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%B3%D1%80%D0%B0%D1%84)
6. Число Данбара и пользователи ВКонтакте [Online] [Cited: Апрель 24, 2017.] <https://habrahabr.ru/post/189226/>
7. The Stanford GraphBase: A Platform for Combinatorial Computing [Online] [Cited: Апрель 24, 2017.] <http://www-cs-faculty.stanford.edu/~uno/sgb.html>
8. Статическая обработка выборки [Online] [Cited: Апрель 24, 2017.]  [http://www.math-pr.com/exampl\_sts1.html](http://www-cs-faculty.stanford.edu/~uno/sgb.html)

# ПРИЛОЖЕНИЕ №1. Основной модуль VKDataAnalysis.py

#!/usr/bin/python  
# -\*- coding: utf8 -\*-  
  
import vk  
import sys  
import time  
import requests # Элегантное и быстрое выполнение HTTP-запросов. "HTTP For Humans!"  
import pickle # Сохранение и загрузка сложных объектов в Python  
import json # JavaScript Object Notation  
from concurrent.futures import ThreadPoolExecutor # Многопоточность  
  
api\_v = '5.21' # Версия VK API  
token = '654baf5ea528a303051f73881df90080075900244621156c9b2989eaf713e6cf63a4e4a3a00c85cbc429c' # Токен (Arbeit Frei)  
session = vk.Session(access\_token=token) # Создание сессии (Arbeit Frei)  
api = vk.API(session);   
max\_workers = 2 # Максимальное количество рабочих потоков при глубинном поиске друзей  
delay = 0.5 # Время (сек), которое при неудачном запросе к серверу нужно подождать одному из потоков перед следующим запросом  
deep = 1 # Глубина поиска  
  
group\_id = '40202469' # Группа ПЕРМСКИЙ УНИВЕРСИТЕТ (12215 человек)  
total = '12200' # Количество пользователей в группе, по которым производить поиск  
permians\_id\_list = [] # Список ID пермяков  
permians\_friends\_list = [] # Список количества друзей пермяков   
  
def force(f, delay=delay):  
 """  
 При неудачном запросе сделать паузу и попробовать снова  
 """  
 def tmp(\*args, \*\*kwargs):  
 while True:  
 try:  
 res = f(\*args, \*\*kwargs)  
 break  
 except KeyError:  
 time.sleep(delay)  
 return res  
 return tmp  
  
class VkException(Exception):  
 def \_\_init\_\_(self, message):  
 self.message = message  
  
 def \_\_str\_\_(self):  
 return self.message   
  
class VkFriends():  
 """  
 Находит друзей, находит общих друзей  
 """  
 parts = lambda lst, n=25: (lst[i:i + n] for i in iter(range(0, len(lst), n)))  
 make\_targets = lambda lst: ",".join(str(id) for id in lst)  
  
 def \_\_init\_\_(self, \*pargs):  
 """  
 Инициализация  
 """  
 try:  
 self.token, self.my\_id, self.api\_v, self.max\_workers = pargs  
 self.my\_name, self.my\_last\_name, self.photo = self.base\_info([self.my\_id])  
 self.all\_friends, self.count\_friends = self.friends(self.my\_id)  
 except VkException as error:  
 sys.exit(error)  
  
 def request\_url(self, method\_name, parameters, access\_token=False):  
 """read https://vk.com/dev/api\_requests"""  
  
 req\_url = 'https://api.vk.com/method/{method\_name}?{parameters}&v={api\_v}'.format(  
 method\_name=method\_name, api\_v=self.api\_v, parameters=parameters)  
  
 if access\_token:  
 req\_url = '{}&access\_token={token}'.format(req\_url, token=self.token)  
  
 return req\_url  
  
 def base\_info(self, ids):  
 """  
 Справка: https://vk.com/dev/users.get  
 """  
 r = requests.get(self.request\_url('users.get', 'user\_ids=%s&fields=photo' % (','.join(map(str, ids))))).json()  
 if 'error' in r.keys():  
 raise VkException('Error message: %s Error code: %s' % (r['error']['error\_msg'], r['error']['error\_code']))  
 r = r['response'][0]  
 # Проверяем, если id из settings.py не деактивирован  
 if 'deactivated' in r.keys():  
 raise VkException("User deactivated")  
  
 return r['first\_name'], r['last\_name'], r['photo']  
  
 def friends(self, id):  
 """  
 read https://vk.com/dev/friends.get  
 Принимает идентификатор пользователя  
 """  
 r = requests.get(self.request\_url('friends.get',  
 'user\_id=%s&fields=uid,first\_name,last\_name,photo,country,city,sex,bdate' % id)).json()['response']  
 #r = list(filter((lambda x: 'deactivated' not in x.keys()), r['items']))  
 return {item['id']: item for item in r['items']}, r['count']  
  
 def common\_friends(self):  
 """  
 read https://vk.com/dev/friends.getMutual and read https://vk.com/dev/execute  
 Возвращает в словаре кортежи с инфой о цели и списком общих друзей с инфой  
 """  
 result = []  
 # разбиваем список на части - по 25 в каждой  
 for i in VkFriends.parts(list(self.all\_friends.keys())):  
 r = requests.get(self.request\_url('execute.getMutual',  
 'source=%s&targets=%s' % (self.my\_id, VkFriends.make\_targets(i)), access\_token=True)).json()['response']  
 for x, id in enumerate(i):  
 result.append((self.all\_friends[int(id)], [self.all\_friends[int(i)] for i in r[x]] if r[x] else None))  
 return result  
  
class D3(VkFriends):  
 """  
 Генерирует json, дабы можно было заюзать http://bl.ocks.org/mbostock/4062045 (Force-Directed Graph)  
 """  
  
 def \_\_init\_\_(self, token, my\_id, api\_v, max\_workers):  
 VkFriends.\_\_init\_\_(self, token, my\_id, api\_v, max\_workers)  
 self.friendships = self.common\_friends()  
 self.js = {"nodes": [], "links": []}  
 self.write\_json(self.to\_json())  
  
 def to\_json(self):  
 """  
 Создание JSON-комбинации  
  
 Из self.friendships сначала составляем узлы, затем ребра.  
 Иначе ребра вначале могут ссылаться на несуществующие узлы.  
 """  
 for i in self.friendships:  
 self.js['nodes'].append({"name": "%s %s" % (i[0]['first\_name'], i[0]['last\_name']),  
 "group": 1, "photo": i[0]['photo']})  
 for i in self.friendships:  
 if i[1]:  
 find\_world = '%s %s' % (i[0]['first\_name'], i[0]['last\_name'])  
 for d in self.js["nodes"]:  
 if find\_world in d.values():  
 for c in i[1]:  
 find\_friend = '%s %s' % (c['first\_name'], c['last\_name'])  
 for e in self.js["nodes"]:  
 if find\_friend in e.values():  
 self.js['links'].append({"source": self.js["nodes"].index(e),  
 "target": self.js["nodes"].index(d), "value": 1})  
  
 return json.JSONEncoder().encode(self.js)  
  
 def write\_json(self, json):   
 """  
 Запись JSON-комбинации в файл  
 """  
 with open("web/miserables.json","w") as f:  
 f.write(json)  
  
def save\_values(friends\_list, file\_name):  
 f = open(file\_name + '.txt','w')  
 for index in permians\_friends\_list:  
 f.write(str(index) + '\n')  
 f.close()  
  
def get\_friends\_list(id\_list, friends\_list):  
 permians\_deleted = 0 # Удаленные пользователи группы  
 for i in range(len(permians\_id\_list)):  
 try:  
 friends = api.friends.get(user\_id = permians\_id\_list[i])  
 permians\_friends\_list.append(len(friends))  
 time.sleep(0.25) # 4 операций в секунду (при попытке сделать больше ВК будет выдавать ошибки)  
 except Exception as e:  
 print(e)  
 permians\_deleted += 1  
 if (i % 10 == 0):  
 print('Число созданных ассоциаций (c друзьями / всего) = ' + str(len(permians\_friends\_list)) + '/' + str(i))  
 print('Количество удаленных пользователей = ' + str(permians\_deleted))  
  
def get\_id\_list(id\_list, group\_id, total):  
 tek = 0  
 number\_of\_cycles = round(int(total) / 25000 + 1) # Количество циклов прохода по группе. Единовременно нельзя, т.к. за один запрос запрещено больше 25к раз  
 print('Количество циклов: ' + str(number\_of\_cycles))  
 for i in range(number\_of\_cycles):  
 r = requests.post('https://api.vk.com/method/execute.group\_getMembers?group\_id='+  
 group\_id+'&tek='+str(tek)+'&access\_token='+token+'&total='+total) # Хранимая процедура на сервере  
 response = r.json()['response']  
 permians\_id\_list.extend(response) # Заполнение списка id пользователей  
 tek += 25000 # Инкрементирование начального значения Tek с шагом 25000  
 # print('Tek = '+ str(tek))  
 permians\_count = len(permians\_id\_list) # Общее число пользователей группы  
 print('Количество проанализированных пользователей: ' + str(permians\_count))  
  
  
if \_\_name\_\_ == '\_\_main\_\_':  
 my\_id = str(input('1. Введите id человека, социальный граф которого необходимо построить: \nid = '))  
 user = api.users.get(user\_id = my\_id)  
 print('Создание графа дружеских связей пользователя ' + user[0]['first\_name'] + ' ' + user[0]['last\_name'] + ' начато...')  
 begin\_time = time.time()  
 a = D3(token, my\_id, api\_v, max\_workers)  
 end\_time = time.time()  
 print('Граф построен. Время выполнения операции: ' + str(round(end\_time - begin\_time, 2)) + ' секунд.')  
  
 group\_name = api.groups.getById(group\_id = group\_id)[0]['name'] # Получение имени группы по id  
 print('\n2. Начат сбор статистики друзей пользователей группы ' + group\_name)  
 begin\_time = time.time() # Начало отсчета времени выполнения  
 get\_id\_list(permians\_id\_list, group\_id, total) # Получение списка ID пользователей какой-либо группы  
 get\_friends\_list(permians\_id\_list, permians\_friends\_list) # Получение списка, хранящего количество друзей этих пользователей  
 end\_time = time.time() # Окончание отсчета времени выполнения  
 save\_values(permians\_friends\_list, 'output') # Сохранение значений  
 print("На сбор статистики по друзьям пользователей группы " +   
 group\_name + "ушло " +   
 str(round(end\_time - begin\_time, 2)) + " секунд. Записаны данные " +   
 str(len(permians\_friends\_list)) + " человек.")

# ПРИЛОЖЕНИЕ №2. Файл index.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<title>Социальный граф дружеских связей</title>

<style>

.link {

fill: none;

stroke: #825ac7;

}

.node text {

pointer-events: none;

font: 10px sans-serif;

}

</style>

</head>

<body>

<script src="d3.v3.min.js"></script>

<script>

var width = 2000,

height = 2000;

var color = d3.scale.category20();

var force = d3.layout.force()

.linkDistance(10)

.linkStrength(1)

.size([width, height]);

var highlight\_color = "blue";

var highlight\_trans = 0.1;

var svg = d3.select("body").append("svg")

.attr("width", width)

.attr("height", height);

d3.json("miserables.json", function(error, graph) {

var nodes = graph.nodes.slice(),

links = [],

bilinks = [];

graph.links.forEach(function(link) {

var s = nodes[link.source],

t = nodes[link.target],

i = {}; // intermediate node

nodes.push(i);

links.push({source: s, target: i}, {source: i, target: t});

bilinks.push([s, i, t]);

});

force

.nodes(nodes)

.links(links)

.start();

var link = svg.selectAll(".link")

.data(bilinks)

.enter().append("path")

.attr("class", "link");

var node = svg.selectAll(".node")

.data(graph.nodes)

.enter().append("g")

.attr("class", "node")

.call(force.drag);

node.append("image")

.attr("xlink:href",function(d) {return d.photo;})

.attr("x", -8)

.attr("y", -8)

.attr("width", 16)

.attr("height", 16);

node.append("title")

.text(function(d) { return d.name; });

force.on("tick", function() {

link.attr("d", function(d) {

return "M" + d[0].x + "," + d[0].y

+ "S" + d[1].x + "," + d[1].y

+ " " + d[2].x + "," + d[2].y;

});

node.attr("transform", function(d) {

return "translate(" + d.x + "," + d.y + ")";

});

});

});

</script>

</body>

</html>