National Research University Higher School of Economics (Higher School of Economics/HSE)
Faculty of Computer Science
Bachelor's Programme Data Science and Business Analytics
01.03.02 Applied Mathematics and Computer Science

Internship report

Fulfilled by

Krauze Natalya Olegovna
Pavleeva Maria Valeryevna
Smotrova Kristina Dmitrievna

(signature)

Checked by

(job or academic title)	(*surname, initials)*	(signature)

Moscow, 2019

Content:

Educational Internship Schedule (Plan)

№	**Calendar period**	**Work Plan**	Internship Supervisor’s signature/
1	01.07.2019	1. Organizational (induction) meeting	
2	01.07.2019	2. Instructing on the requirements of labor protection, safety, fire safety and internal labor regulations	
3	01.07.2019 - 13.07.2019	3. Fulfillment of Individual Assignment	
4	01.07.2019 - 13.07.2019	4. Consultation	
5	14.07.2019	5. Preparation and submission of the Report	

Formulation of the problem

Our task was to analyze groups of the social network VKontakte using graph algorithms. As a community for analysis, we took the “Квесчн” group, there were 442 users (at the time of our work), most of whom are the students of the Faculty of Computer Science.

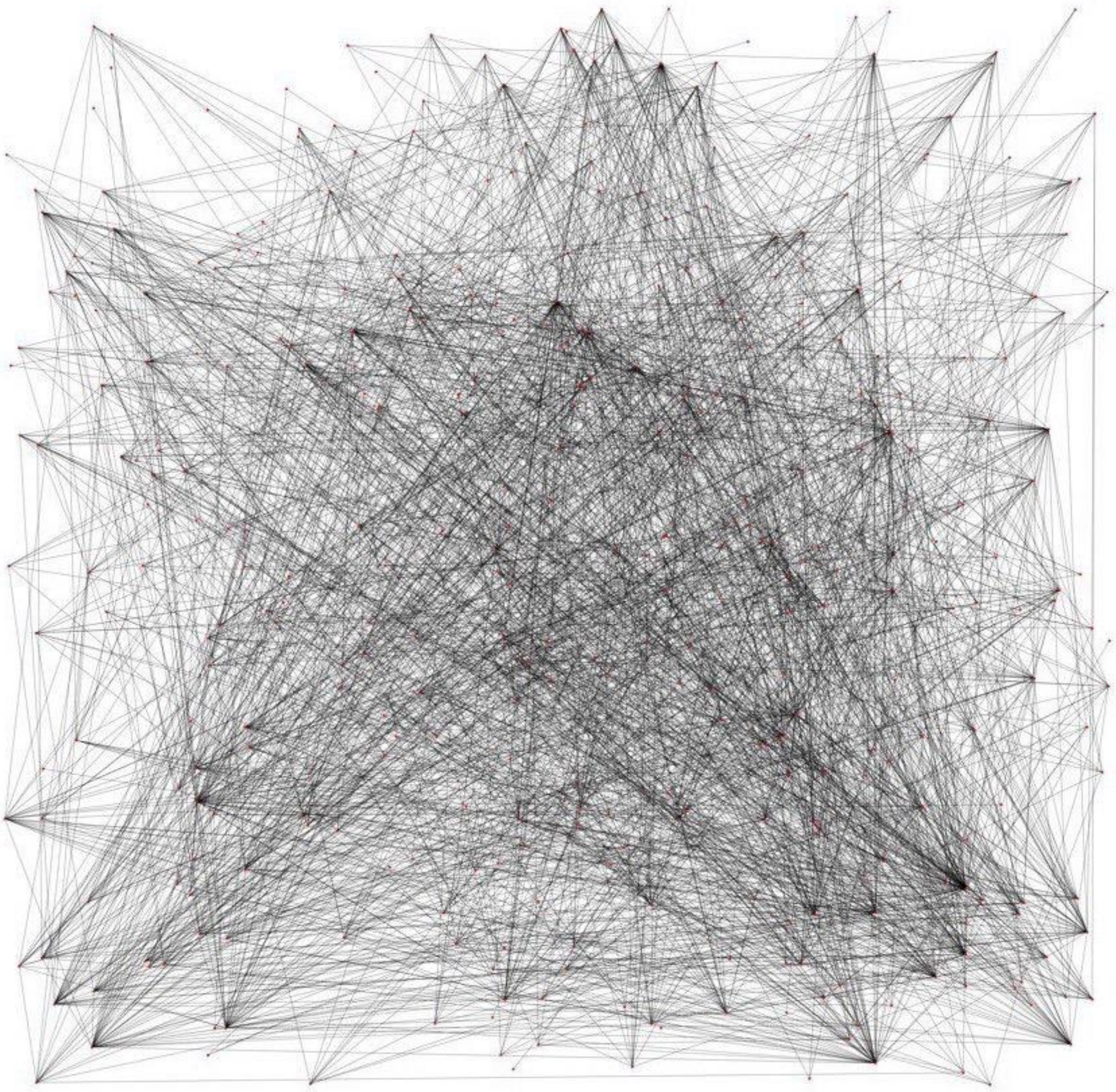
Globally, a task can be divided into the following components: obtaining data about users of a group using VK API, building a graph from the received data, launching and analyzing the work of algorithms on the graph of users. Additionally, user graph visualization was implemented using the NetworkX library.

Data receiving

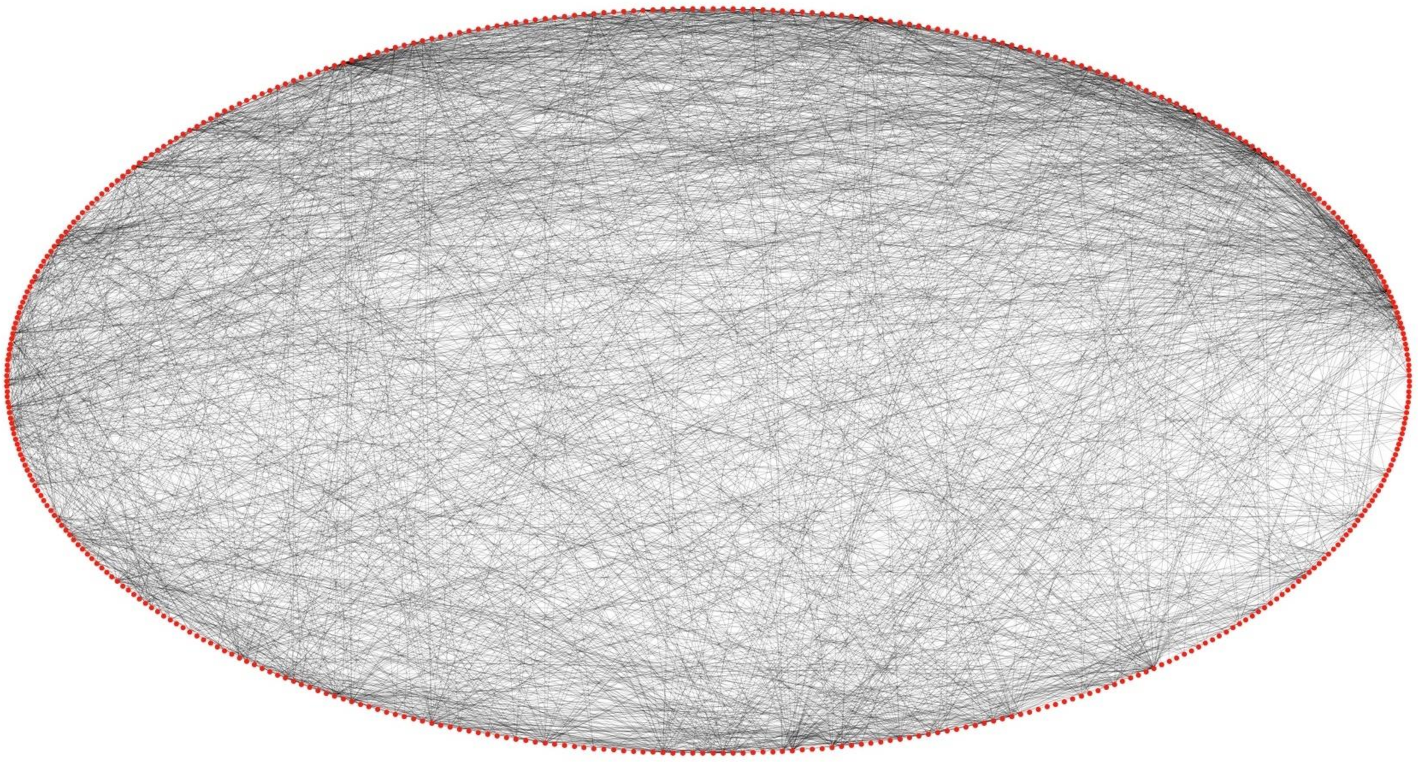
In order to obtain data, we used VK API requests using the “requests” library. Firstly, we got access to the group members list: this is done by the “getMembers” method. After that, for each of the users, it was necessary to obtain a list of all his friends, then to check for all users of the group, whether they are in the user's friends list (this brute force takes a linear number of users, since we put the list of users in an unordered set, where the search operation is performed in constant time), therefore, the general asymptotic behavior of obtaining information about the connections between users required quadratic time. In the process of obtaining the list of friends, the following problems arose: some users have closed accounts, so information about their friends cannot be obtained. Another part of the users was blocked by the VK administration or had deleted pages, from where it was also impossible to get information. There were also restrictions on the number of requests per second (no more than 3), so we had to enter the program “in a dream” for 1 second so that it did not complete its work in emergency.

Compilation and visualization of the graph

Based on the data obtained, we constructed several graph representations: as an adjacency matrix, as a list of adjacent vertices, and as a list of edges. The vertices in this graph were users, and the users who were each other's friends are connected by edges. The resulting graph was sparse (it had about 2300 edges out of 97,000 possible). Then we decided to visualize the graph despite its size in order to better understand how the relationships between users are arranged. For this, we used NetworkX library. We built several visualizations: with a random arrangement of vertices on a plane and with a location of them over the oval shape.



Visualization of a graph with a random arrangement of vertices on a plane



Graph visualization with ovoid ordering of vertices on a plane

Graph analysis

As it was mentioned earlier, the resulting graph was quite sparse. But this is not the main problem. It was originally planned to analyze the graph using the Karger's algorithm to find the minimum cut. But this only makes sense for connected graphs. Our graph is not connected. Therefore, we decided to search for connectivity components, based on the DFS algorithm. As a result, in our graph there are about 65 connectivity components: one large (about 300 users) and a lot of small ones (mostly they are either isolated vertices or separate edges between a pair of vertices). A large component of connectivity consisted of students of the FCS, the rest were either blocked users, or fake accounts, or pairs of people from other faculties who, apparently, found the group in recommendations and were interested in the subject of posts.
In the end, we tried to apply the "Career" algorithm to our large connectivity component and found that the size of the minimum cut is 8, which means that our components are divided into 2 large clusters of users, among which there are only 8 connections (edges).

Bibliography:

1) https://vk.com/dev/manuals - VK API manual
2) https://networkx.github.io/documentation/stable/tutor.. - NetworkX tutorial

3) https://e-maxx.ru/algo/stoer_wagner_mincut – Karger’s algorithm explanation
4) https://en.wikipedia.org/wiki/Karger%27s_algorithm

Conclusion

In the process, we mastered many new tools and libraries: used VK API requests using the “request” library, built user graph visualizations using NetworkX, studied and implemented new algorithms, namely, the Karger’s algorithm and the search algorithm of connectivity components based on the search in depth (DFS). We also learned to interpret the results of the algorithms from a practical point of view.