УДК 004.9:519.178:316.35
DOI 10.25205/1818-7900-2021-19-2-76-91

Модели импорта данных из Твиттера

В. А. Попов, А. А. Чеповский

Национальный исследовательский университет «Высшая школа экономики»
Москва, Россия

Аннотация

Описываются алгоритм импорта данных из социальной сети Twitter и построение взвешенных социальных графов. Для импорта данных за основу берутся заданные посты, скачиваются пользователи, имевшие с ними какое-либо из зафиксированных взаимодействий. Далее алгоритм ориентируется на заданную конфигурацию и по ней вычисляет веса на ребрах полученного графа. Конфигурация учитывает тип взаимодействия пользователей между собой. Авторы вводят понятие (F, L, C, R)-модели информационного взаимодействия.

Авторы описывают разработанный алгоритм и реализованное программное обеспечение для построения взвешенных графов. В статье показано применение алгоритма и трех моделей на примере как отдельного поста, так и серии постов.

Ключевые слова

анализ социальных сетей, импорт данных из социальных сетей, модели информационного взаимодействия, выделение сообществ

Благодарности

Работа выполнена при финансовой поддержке РФФИ в рамках научного проекта № 19-07-00806

Для цитирования

Попов В. А., *Чеповский А. А.* Модели импорта данных из Твиттера // Вестник НГУ. Серия: Информационные технологии. 2021. Т. 19, № 2. С. 76–91. DOI 10.25205/1818-7900-2021-19-2-76-91

Twitter Data Import Models

V. A. Popov, A. A. Chepovskiy

National Research University Higher School of Economics
Moscow, Russian Federation

Abstract

In this paper, the authors describe an algorithm for importing data from the social network Twitter and building weighted social graphs. To import data, the given posts are taken as a basis, users who have had any of the recorded interactions with them are downloaded. Further, the algorithm focuses on the given configuration and uses it to calculate the weights on the edges of the resulting graph. The configuration takes into account the type of user interaction with each other. The authors introduce the concept of (F, L, C, R)-model of information interaction.

The authors describe the developed algorithm and implemented software for constructing weighted graphs. The paper shows the application of the algorithm and three models on the example of both a single post and a series of posts.

Keywords

social network analysis, import of data from social networks, information interaction models, community detection

Acknowledgements

The study was funded by RFBR according to the research project no. 19-07-00806

For citation

Popov V. A., Chepovskiy A. A. Twitter Data Import Models. *Vestnik NSU. Series: Information Technologies*, 2021, vol. 19, no. 2, p. 76–91. (in Russ.) DOI 10.25205/1818-7900-2021-19-2-76-91

Введение

В последнее время социальные сети вошли в повседневную жизнь многих людей. Ежесекундно в сети генерируется огромный объем информации, передаются десятки тысяч сообщений и создаются сотни тысяч постов и комментариев. Все пользователи в социальных сетях имеют различные взаимодействия между собой, создавая единый социальный граф и множество потоков распространения информации. Изучение данных структур интересно для понимания действий конкретных людей и общества в целом, многие рекомендательные и рекламные системы работают на основе обработки данных о пользователях и их действиях в социальных сетях. Государственные структуры также занимаются изучением информационного потока в сетях для обеспечения безопасности. Поэтому тема изучения графов взаимодействующих объектов, в том числе социальных графов, формируемых на основе данных из социальных сетей, является актуальной в наши дни [1–4].

Одним из важных методов анализа социальных графов является подход к выявлению групп общения пользователей путем выделения на основе алгоритмов обработки графов неявных сообществ [5; 6]. Для большей эффективности данной операции требуется использовать взвешенный социальный граф, основанный на совершенных действиях пользователей. В этой связи возникает задача разработки алгоритмов для построения таких графов на основе особенностей каждой отдельной социальной сети.

Взвешенные графы

Твиттер (Twitter) – социальная сеть, в которой у каждого пользователя есть своя страница с краткой информацией. На ней пользователи могут размещать тексты до 280 символов – постить твиты (Tweets). Также есть возможность прикрепить фото или видео в постах. Каждый пользователь имеет свое уникальное имя (Username). При создании нового аккаунта каждому присваивается имя из случайных латинских букв и цифр длиной в 15 символов, которое впоследствии можно изменить на любое не занятое. Каждый пользователь может подписаться на других, т. е. читать их посты в своей ленте. Таким образом, у каждого пользователя формируется два списка: список Читателей (Followers) и список Читаемых (Following).

Все посты-твиты имеют уникальный числовой идентификатор (id), и у каждого пользователя есть возможность совершить три действия над ними: поставить отметку «Нравится» (Like), поделиться постом у себя на странице (Retweet) и написать комментарий (Reply). Каждый комментарий имеет такую же структуру, как и основной пост, т. е. считается твитом.

Действия пользователей фиксируются на их личных страницах. В главном разделе находятся все твиты и ретвиты этого пользователя, в разделе «Твиты и ответы» (Tweets & replies) расположены все тексты пользователя, т. е. его оригинальные посты, комментарии, адресованные другим людям, и тексты при ретвитах. На странице «Медиа» (Media) сохраняются все медиафайлы пользователя (изображения и видео к твитам). Последний раздел «Нравится» (Likes) содержит все твиты, которым выбранный пользователь поставил соответствующую оценку – Like.

Таким образом, в социальной сети Twitter существует 2 глобальных объекта со своими полями: пользователь и пост. Каждый пользователь имеет следующие атрибуты: уникальный идентификатор, краткая информация о себе, список своих оригинальных постов, ретвитов и комментариев, список постов, которым пользователь поставил Like. А у каждого поста присутствуют: автор, текст, комментарии и списки пользователей, которые поставили отметку Like и / или ретвитнули этот пост. Учитывая весь функционал данной социальной сети, было выделено 4 вида взаимоотношений (взаимодействий) между пользователями:

- подписка друг на друга;
- лайки постов других пользователей;
- комментарии к постам других пользователей;

- ретвиты записей других пользователей.

На основе этих связей имеется возможность сформировать социальный граф пользователей. Социальный граф – граф, вершины которого представляют собой социальные объекты (ими могут быть аккаунты пользователей, сообщества), а ребра описывают отношения между ними. Причем, можно сделать как метаграф, содержащий ребра, соответствующие каждому из четырех взаимоотношений, так и отдельные графы для каждого вида взаимодействия.

Но наибольший интерес представляют взвешенные графы, вершины которых соответствуют пользователям, а ребра строятся в случае наличия хотя бы одного из взаимоотношений. При этом вес на ребрах в данном случае можно определять на основании данных об имевших место взаимодействиях по-разному, в зависимости от задач, поставленных перед исследователями сети скачанных объектов. Суммарный вес ребра может быть посчитан, например, как линейная комбинация весов, соответствующих каждому из зафиксированных взаимодействий пользователей. Эти отдельные веса для взаимоотношений и были приняты как параметры при построении взвешенных социальных графов.

Таким образом, определим (F, L, C, R)-модель информационного взаимодействия в сети Twitter как взвешенный граф $G(V, E)$, у которого на множестве ребер E весовая функция $w(\mathrm{e})$ с неотрицательными значениями задана следующим образом:

$$w(\mathrm{e}) = F \times \delta_e^F + L \times \delta_e^L + C \times \delta_e^C + R \times \delta_e^R,$$

где

$$\delta_e^F = \begin{cases} 2, & \text{если оба инцидентных пользователя подписаны друг на друга,} \\ 1, & \text{если хотя бы один из инцидентных пользователей подписан на другого,} \\ 0, & \text{иначе;} \end{cases}$$

$$\delta_e^L = \begin{cases} 2, & \text{если между инцидентными пользователями есть взаимые Like,} \\ 1, & \text{если есть хотя бы один Like между инцидентными пользователями,} \\ 0, & \text{иначе;} \end{cases}$$

$$\delta_e^C = \begin{cases} 2, & \text{если между инцидентными пользователями есть взаимые Reply,} \\ 1, & \text{если есть хотя бы один Reply между инцидентными пользователями} \\ 0, & \text{иначе;} \end{cases}$$

$$\delta_e^R = \begin{cases} 2, & \text{если между инцидентными пользователями есть взаимые Retweet,} \\ 1, & \text{если есть хотя бы один Retweet между инцидентными пользователями,} \\ 0, & \text{иначе.} \end{cases}$$

Под взаимностью тут понимаются действия от каждого из двух пользователей, но не обязательно к одному и тому же твиту. Так, например, (1, 3, 2, 4)-модель соответствует ситуации, когда вес подписок – 1, вес лайков – 3, вес комментариев – 2, вес ретвитов – 4.

Импорт данных

Входными данными для задачи импорта является список исходных постов, заданных оператором. Это могут быть обнаруженные ранее посты схожей или противоположной направленности об одном событии или группе событий, предположительные вбросы ложной / искаженной информации, рекламные сообщения и т. п. Никаких существенных ограничений на количество этих постов или их свойства не налагается.

Задача состоит в импорте данных об имевшем место взаимодействии пользователей социальной сети с этими постами и между собой. На основе этих данных на следующем шаге и будет построен социальный граф. А результат работы импорта представляет собой файл в специальном XML-подобном унифицированном формате AVS, который позволяет задать

вершины и ребра графа. Различные данные о вершинах и ребрах графа задаются в AVS-файле как атрибуты вершин и ребер соответственно. Важно отметить, что основные данные (уникальные имена пользователей-вершин и веса ребер) сохраняются в самом AVS-файле. В то же время тексты постов и комментариев могут суммарно быть достаточно большого объема, поэтому для каждого пользователя сохраняются в отдельном файле, ссылка на который находится в одном из атрибутов соответствующей вершины графа, описанной в AVS-файле. Далее такой файл может быть обработан сторонним программным обеспечением.

В ходе импорта данных можно выделить два основных этапа. На первом этапе для заданного списка исходных постов в зависимости от задач и настроенной конфигурации скачиваются списки пользователей, провзаимодействовавших с заданными постами. Это могут быть три списка пользователей:

- лайкнувшие данную запись;
- репостнувшие данную запись;
- прокомментировавшие данную запись.

На основе этой информации путем объединения пользователей из всех трех множеств формируется список вершин будущего социального графа. Более никакие вершины в граф не добавляются.

На втором этапе импортируются заданные согласно конфигурации взаимоотношения между пользователями. Как было сказано ранее, на основе данных из Твиттера можно построить различные взвешенные графы с учетом заданных параметров – весов каждого из 4 типов взаимодействий пользователей.

В итоге второго этапа работы формируются ребра графа, далее им присваиваются веса в зависимости от выбора модели, и сформированный граф записывается в AVS-файл.

Подходы к импорту данных из Twitter

Базовым методом для получения данных из Твиттера является использование API (Application Programming Interface) [1]. Но он имеет ряд ограничений: например, количество запросов на получение списка фолловеров пользователя лимитировано пятнадцатью запросами за пятнадцать минут, также сам функционал API ограничен, с его помощью нельзя скачать списки постов, которые лайкнул, ретвитнул или прокомментировал выбранный пользователь.

И главное – с сентября 2018 г. Твиттер ограничил выдачу новых аккаунтов для разработчиков. Платформа Твиттер для разработчиков отличается от обычных аккаунтов тем, что дополнительно предоставляется множество продуктов и инструментов API, которые позволяют автоматизировать многие процессы Твиттер в реальном времени. Например, можно настроить автоматический постинг твитов с других платформ, делать автоматические интеграции рекламы и совершать импорт данных.

Поэтому для импорта данных в финальной реализации приложения был выбран метод веб-скрапинга (получение данных путем извлечения их со страниц веб-ресурсов) [7], который также имеет свои ограничения. Так, при скачивании списков пользователей, лайкнувших или прокомментировавших выбранный пост, Твиттер ограничивает выдаваемые списки только 50–80 пользователями. Но, с другой стороны, данный метод позволяет импортировать всю информацию о пользователях (список читателей, ретвитов, лайков, комментариев), а ограничения на количество запросов в минуту не такие жесткие, как при использовании API. Здесь существенен тот факт, что спустя определенное время Твиттер начинает блокировать запросы с одного ip-адреса, поэтому приходится искусственно уменьшать частоту запросов, а также делать паузы в скачивании. Но, несмотря на это, метод веб-скрабинга позволяет скачивать информацию о взаимодействии между пользователями в более короткий срок.

[1] Twitter API. URL: https://developer.twitter.com/en/docs/basics/getting-started (дата обращения 15.02.2021).

Импорт вершин социального графа

Для импорта социального графа сначала выбирается, какая информация по всем указанным в исходных данных постам будет импортирована. Для оператора предоставлено три опции для множественного выбора: лайки, ретвиты и комментарии. При выборе этих элементов будут скачаны соответствующе списки пользователей. Иначе говоря, при выборе лайков, по каждому посту будет скачан список пользователей, которые лайкнули их, при выборе ретвитов будут импортированы списки ретвитнувших посты и при выборе комментариев соответственно формируются списки прокомментировавших.

Для импорта этих данных используется инструмент автоматизации действий веб-браузера – Selenium [2], а именно программная библиотека языка Python Selenium WebDriver [3] с использованием ChromeDriver [4] и пакет Python для анализа HTML Beautiful Soup [5]. Сначала с помощью веб-драйвера происходит аутентификация в Твиттере, затем для каждого поста посещаются и скачиваются HTML-коды следующих URL-страниц:

- при выборе лайков: https://twitter.com/author/status/postID/likes/;
- при выборе ретвитов: https://twitter.com/author/status/postID/retweets/;
- при выборе комментариев: https://twitter.com/author/status/postID/.

Дополнительно для страницы с комментариями перед скачиванием HTML-кода совершается прокрутка страницы до самого низа. Это необходимо, чтобы охватить весь список комментариев, которые изначально не помещаются при посещении страницы.

После этого скачанные HTML-коды страниц обрабатываются с помощью библиотеки Beautiful Soup. Для страниц каждого вида была проанализирована структура хранения данных и были определены нужные теги, по которым хранятся имена пользователей, совершивших действие. На этом этапе из HTML-кодов страниц для каждого поста формируются и сохраняются списки пользователей, лайкнувших, ретвитнувших и прокомментировавших посты (в зависимости от изначально выбранных флагов). Далее все списки пользователей объединяются в один users_list – список пользователей-вершин будущего социального графа.

Результатом данного этапа является формирование списка вершин. Первый этап производится единожды для заданных постов. Второй этап построения ребер может быть применен к списку вершин многократно, его итогом будет уже создание графа в зависимости от параметров.

Настройка импорта ребер социального графа

Далее для импорта ребер выбирается, какие виды отношений между пользователями будут использоваться для построения взвешенного графа. Есть четыре возможности: «Подписка», «Лайки», «Комментарии» и «Ретвиты», при выборе которых для каждого пользователя из предыдущего пункта (список users_list) будут импортированы списки его фолловеров, списки постов, которые он лайкает, комментирует, ретвитит соответственно. На основе этих списков далее будут построены ребра графа.

Также при выборе возможности любого типа, пользователю предоставляется выбор веса этого взаимодействия, что повлияет на веса ребер в итоговом графе.

Рассмотрим на примерах механизм работы данного этапа. Пусть выбран только пункт «Подписка» с весом F. Это означает, что если два пользователя не подписаны друг на друга,

[2] Инструмент автоматизации действий веб-браузера. URL: https://www.selenium.dev (дата обращения 15.02.2021).

[3] Библиотека Python Selenium. URL: https://selenium-python.readthedocs.io (дата обращения 15.02.2021).

[4] ChromeDriver. URL: https://sites.google.com/a/chromium.org/chromedriver (дата обращения 15.02.2021).

[5] Библиотека Python Beautiful Soup. URL: https://www.crummy.com/software/BeautifulSoup/ (дата обращения 15.02.2021).

то ребра между ними не будет; если присутствует односторонняя подписка, то вес ребра между вершинами будет F; а при взаимной подписке – $2F$.

При выборе флага «Лайки» с весом L смотрятся последние 50 отметок Like каждого пользователя. Если среди этого списка из 50 твитов присутствует хотя бы один лайк первого пользователя второму, то к весу ребра между ними прибавляется L, если аналогично у второго есть лайк к первому, то вес ребра становится $2L$. Учитывается только наличие или отсутствие лайков с каждой стороны, если количество лайков с одной стороны больше одного, то вес ребра все равно увеличивается только на L.

При активации флага «Комментарии» с весом C импортируются последние 50 текстов пользователя (его оригинальные посты и комментарии к другим). При наличии комментария к другому пользователю вес C, при взаимных – $2C$. Аналогично лайкам и комментариям учитывается только наличие или отсутствие комментариев с каждой стороны.

Аналогичная схема при выборе флага для «Ретвитов». С весом R скачиваются последние 50 постов пользователя включая ретвиты. Если среди них присутствует ретвит другого пользователя, то к весу ребра между ними прибавляется R, при взаимных репостах вес ребра увеличивается до $2R$. Так же как в лайках, учитывается только наличие или отсутствие ретвитов с обеих сторон, а количество больше одного не влияет на увеличение веса.

В итоге при формировании взвешенного графа все веса по каждому выбранному взаимодействию между двумя пользователями складываются, и получается итоговый вес ребра в графе. Разберем пример: пусть выбраны флаги «Подписка», «Лайки», «Комментарии» и «Ретвиты» с весами 1, 3, 2 и 4 соответственно. Два пользователя имеют взаимную подписку; первый ставит лайки второму, но второй не ставит первому; второй комментирует посты первого, но первый не комментирует посты второго; ретвиты они оба не делают. Тогда вес ребра между ними будет равен $(1 + 1) + 3 + 2 + 0 = 7$.

Описанные веса ребер строятся для каждой пары пользователей из списка вершин графа users_list, и на основе этих данных формируется итоговый взвешенный граф.

Также на данном этапе есть еще одна возможность – скачать последние 40 текстов (посты, комментарии, тексты к ретвитам) каждого пользователя, являющегося вершиной социального графа.

После выбора всех настроек начинает работать алгоритм импорта данных пользователей. В этом алгоритме также применяется Selenium WebDriver с использованием ChromeDriver и пакет Python Beautiful Soup. А также применяется библиотека Python Twint [6], основанная на методе скрапинга веб-страниц. Далее детально рассмотрим этапы его работы.

Импорт ребер социального графа

Для каждой выбранной опции запускается свой алгоритм. Соответствующий алгоритм применяется для всех пользователей из общего списка вершин users_list.

1. **Импорт подписок.** С помощью Selenium WebDriver и Python Beautiful Soup скачивается список читателей каждого пользователя. Все списки объединяются в единую структуру данных – словарь, где никам пользователей из users_list соответствуют списки их фолловеров.

2. **Импорт лайков.** Также с помощью Selenium WebDriver и Python Beautiful Soup для каждого пользователя импортируется список последних 50 постов, которые лайкает выбранный юзер, который впоследствии преобразуется в pandas DataFrame. Далее скачанные данные обрабатываются, и для каждого пользователя из users_list формируется два списка: список id постов и список никнеймов пользователей, которых пользователь отметил лайком. После этого полученные списки по всем пользователям также объединяются в словарь.

[6] Библиотека Python Twint. URL: https://github.com/twintproject/twint (дата обращения 15.02.2021).

3. **Импорт ретвитов.** Для данного пункта используется метод Profile библиотеки Python Twint, с помощью которой импортируется 50 оригинальных постов и ретвитов каждого пользователя с идентификатором автора поста. По этому идентификатору отсеиваются ретвиты от оригинальных постов пользователя и формируется список репостов. Далее, аналогично лайкам, с каждым пользователем в словаре сопоставляется список никнеймов и id постов, которые он ретвитил.

4. **Импорт комментариев и текстов.** В этом случае применяется метод Search, который возвращает оригинальные твиты, ответы и тексты к ретвитам пользователя в виде структуры pandas DataFrame, где есть следующие поля: id, conversation_id, reply_to, retweet, text. Поле retweet указывает, является ли данный объект ретвитом. Далее по conversation_id разделяем комментарии и оригинальные посты автора. Если пост является комментарием, то conversation_id равен id оригинального поста, к которому сделан данный комментарий, и reply_to – имя автора этого поста. А в случае оригинального поста conversation_id равно id, а поле reply_to пустое. Таким образом, из общей структуры можно выделить список постов и пользователей, которых комментирует автор, и составить словарь аналогичный лайкам, где каждому пользователю из users_list будет соответствовать два списка.

Более того, с помощью данного метода можно также скачать последние тексты пользователей, разделяя их по видовым группам (оригинальный пост, комментарий или текст ретвита).

Несомненным плюсом использования Python Twint при импортировании ретвитов и комментариев является тот факт, что импорт списков по каждому пользователю можно совершать в несколько потоков. Так, с помощью библиотеки Python Multiprocessing импорт работает в 4 потока.

После этого для каждого пользователя все скачанные ранее списки объединяются, и формируется единая структура данных pandas DataFrame. Затем все списки пользователей пересекаются со списком ников users_list. Так в структуре данных остается информация о лайках, ретвитах, комментариях только пользователей из списка вершин графа. Процесс пересечения происходит уже локально, после импорта данных, а не во время скачивания. Это позволяет сократить время работы, совершая меньшее количество запросов, а также сохранить все данные о пользователях для дальнейшего расширенного анализа. В итоге, после выполнения этой части алгоритма, на основе получившейся структуры строится финальный граф с учетом всех весов ребер.

В конце данного этапа после описанных выше процедур сам граф сохраняется в формате AVS. При необходимости можно выбрать другие варианты импорта взаимоотношений или изменить их веса. Скачанная до этого информация будет использована снова, а недостающая информация о пользователях будет импортирована. Другими словами, при смене только весов взаимодействий ничего дополнительного скачиваться не будет, будет только сформирован заново взвешенный граф, а при добавлении нового типа взаимодействия сначала импортируется дополнительная информация, после чего на основе старых и новых данных сформируется новый граф. Далее полученные графы импортируются в новые AVS-файлы.

Пример импорта данных – скачивание одного поста по (1, 3, 2, 4)-модели

В сети Twitter по состоянию на вечер 03.02.2021 был скачан следующий пост (от 02.02.2021), посвященный новости об успешном завершении третьей фазы клинических испытаний Sputnik V – вакцины от новой коронавирусной инфекции Covid-19: https://twitter.com/sputnikvaccine/status/1356580985127792650.

Исходный граф, построенный на основе импорта из Twitter по описанному в данной работе алгоритму в соответствии с (1, 3, 2, 4)-моделью, содержит 268 вершин и 100 ребер (рис. 1). Средний вес ребра равен 2,3, максимальная степень в графе – 12. При этом 174 вершины

изолированные – это пользователи, которые взаимодействовали с исходным постом, но никак не взаимодействовали с остальными скачанными пользователями (и те с ними – тоже).

После удаления изолированных вершин и применения к графу алгоритма Infomap [8] для разбиения на непересекающиеся сообщества получаем граф $\boldsymbol{G'}$. В нем выделено 15 сообществ (рис. 2). Три самых больших из них – $\boldsymbol{S_0}$, $\boldsymbol{S_1}$ и $\boldsymbol{S_2}$ содержат 41, 15 и 10 вершин соответственно. В каждом из этих сообществ, в свою очередь, аналогично может быть выделена структура (рис. 3).

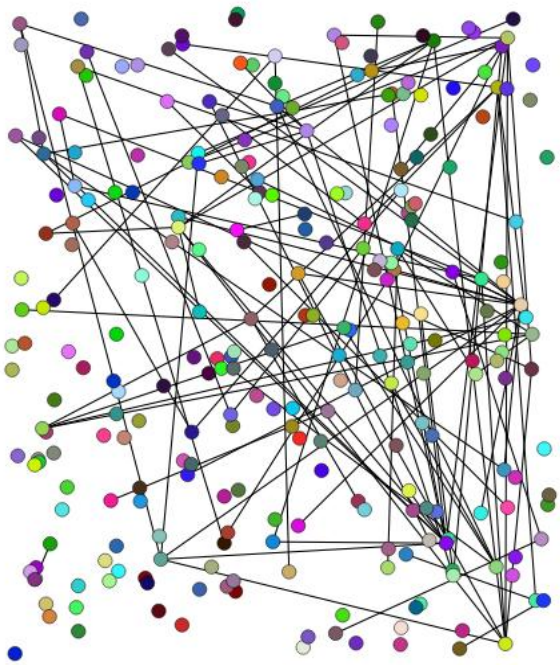


Рис. 1. Исходный граф **G**
Fig 1. The original graph **G**

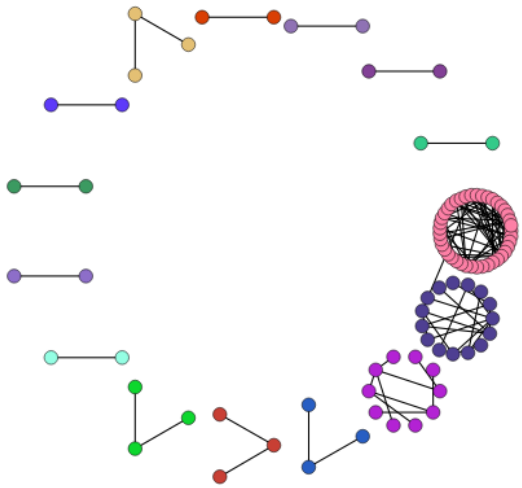
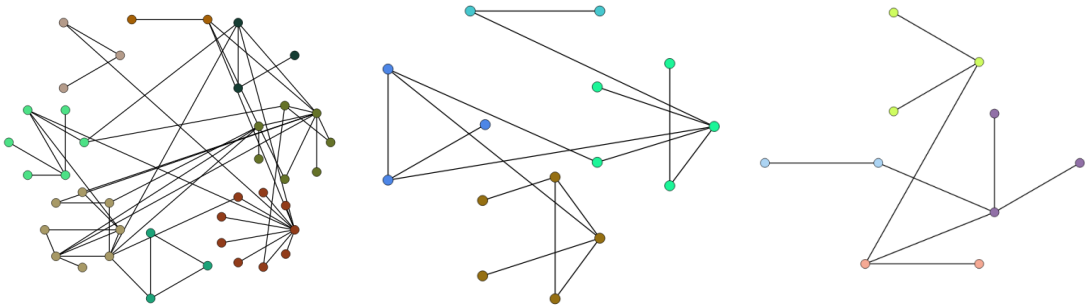


Рис. 2. Разбиение графа $\boldsymbol{G'}$ на 15 сообществ
Fig. 2. Splitting the graph $\boldsymbol{G'}$ into 15 communities



а б в

Рис. 3. Внутреннее разбиение сообществ: *а* – $\boldsymbol{S_0}$; *б* – $\boldsymbol{S_1}$; *в* – $\boldsymbol{S_2}$
Fig 3. Internal partition of the community: *a* – $\boldsymbol{S_0}$; *b* – $\boldsymbol{S_1}$; *c* – $\boldsymbol{S_2}$

Как видно, сообщество $\boldsymbol{S_2}$ не обладает выраженными вершинами-лидерами. Такая конструкция сообщества была названа в работе [9] «созвездием третьего рода». В сообществе $\boldsymbol{S_1}$ и особенно в $\boldsymbol{S_0}$ уже можно выделить лидеров мнений. Из четырех вершин с наибольшими степенями две получают их за счет всех взаимных действий (рис. 4), при этом не имея особых связей с остальными вершинами графа.

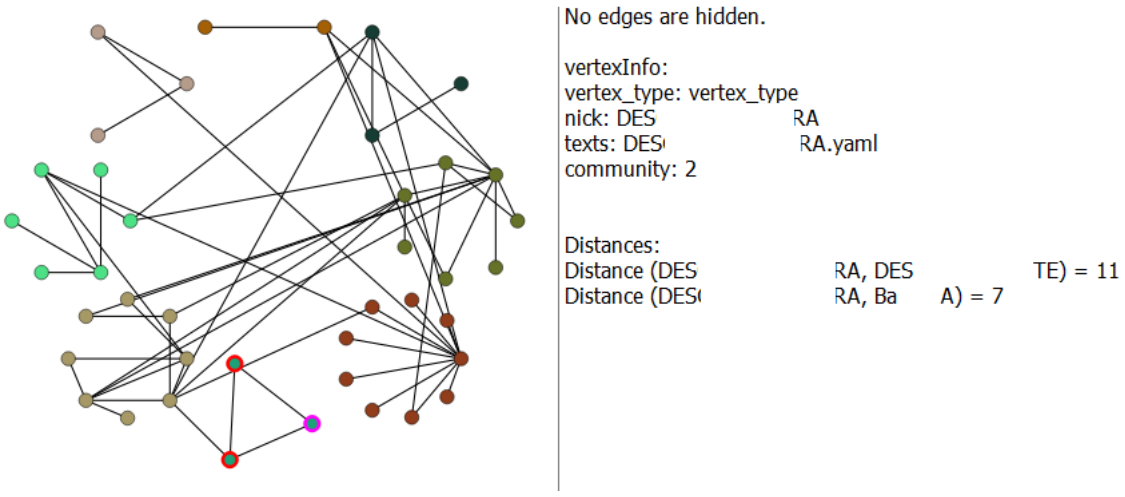


Рис. 4. Вершины с высокими степенями, но малым числом связей
Fig. 4. Vertices with high degrees but few connections

Представленный подход к импорту позволяет получить и исследовать таблицу со значениями по степеням и взвешенным степеням вершин. Посмотрим на вершины с высокими значениями этих показателей (табл. 1).

Две другие вершины в сообществе $\boldsymbol{S_0}$ являются явно выраженными лидерами мнений, на что указывают как высокая степень, так и связи с остальными вершинами (рис. 5). Таким образом, описанный алгоритм и $(1, 3, 2, 4)$-модель построения графов позволяют получить взвешенные графы, в которых можно выделять содержательные сообщества и искать лидеров мнений.

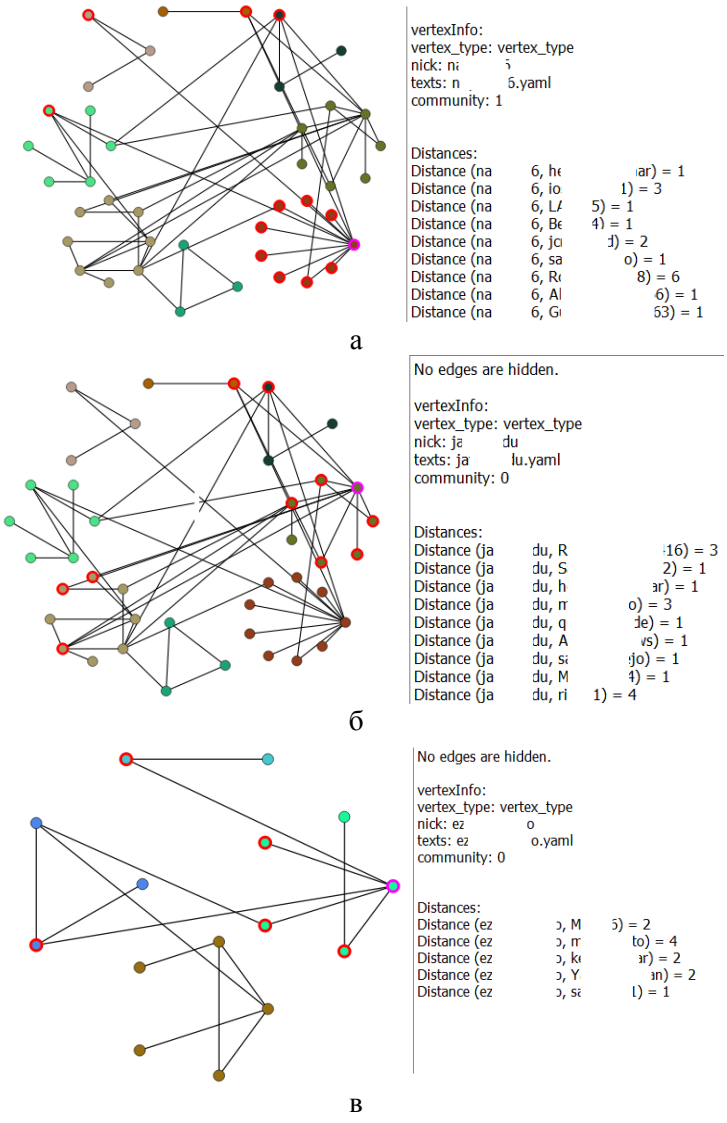
Таблица 1

Вершины графа G' с высокими значениями степеней

Table 1

Vertices of the graph G' with high degree values

Номер сообщества	Вершина	Рисунок	Степень вершины	Взвешенная степень вершины
0	na***6	5, *а*	12	25
0	DES***RA	4	2	18
0	DES***TE	4	3	17
0	ja***u	5, *б*	10	17
1	ez***o	5, *в*	5	11



а

б

в

Рис. 5. Вершина na***6 – лидер в S_0 (*а*);
вершина ja***u – лидер в S_0 (*б*); вершина ez***o – лидер в S_1 (*в*)

Fig 5. Vertex na***6 – leader in S_0 (*a*);
vertex ja***u – leader in S_0 (*b*); vertex ez***o – leader in S_1 (*c*)

Пример второй – скачивание нескольких постов и разные модели

В сети Twitter по состоянию на вечер 27.05.2020 были скачаны 8 актуальных постов, касающихся действий властей города Москвы в рамках борьбы с новой коронавирусной инфекцией Covid-19, а также относящиеся к ним комментарии, лайки, ретвиты. Среди постов были как официальные заявления руководства города в СМИ и своих аккаунтах, так и остросоциальные сообщения провокационного характера:

- https://twitter.com/rianru/status/1265550010000818180
- https://twitter.com/VRSoloviev/status/1265627361397166080
- https://twitter.com/rianru/status/1265630484488470528
- https://twitter.com/rianru/status/1265627223907860481
- https://twitter.com/tass_agency/status/1265628021538553861
- https://twitter.com/izvestia_ru/status/1265633807451009024
- https://twitter.com/moslenta/status/1265627960746352642
- https://twitter.com/meduzaproject/status/1265627619078389760
- https://twitter.com/EchoMskRu/status/1265627900373536768
- https://twitter.com/SobolLubov/status/1265629006478614529

Скачивание проходило с генерацией взвешенного графа в трех вариациях со следующими параметрами:

- (1, 3, 0, 0)-модель: без учета комментариев и ретвитов (вес подписок – 1, вес лайков – 3);
- (1, 3, 0, 4)-модель: без учета комментариев, но с учетом ретвитов (вес подписок – 1, вес лайков – 3, вес ретвитов – 4);
- (1, 3, 2, 4)-модель: с учетом комментариев и ретвитов (вес подписок – 1, вес лайков – 3, вес комментариев – 2, вес ретвитов – 4).

Также были скачаны и соответствующие тексты для дальнейшего анализа, ссылки на которые хранятся в AVS-файлах графов как длинные атрибуты. По итогам получено три соответствующих графа (табл. 2).

Таблица 2

Данные о скачанных графах

Table 2

Data about downloaded graphs

Граф	n	m	Mean edges weight	Mean vertex degree	Max vertex degree	Взвешенный диаметр
$\boldsymbol{G_1}$	632	845	2.257	1.337	92	23
$\boldsymbol{G_2}$	632	906	2.599	1.434	97	27
$\boldsymbol{G_3}$	632	1002	2.767	1.585	126	28

Далее для каждого из графов был применен алгоритм Infomap [8] для выделения неявных сообществ (табл. 3, 4). Необходимо отметить, что в полученных графах в соответствии с алгоритмом скачивания имеется большое количество висячих вершин с ребрами единичного веса. Данные вершины соответствуют пользователям, имевшим взаимодействие с исходными постами, но при этом с остальными вершинами связанные лишь фолловерством.

В результате анализа графа сторонним программным обеспечением получены лидеры мнений небольших сообществ, участвовавших в обсуждении и распространении исходных постов. Причем в ряде случаев в зависимости от графа сообщества формируются по-разному, что подчеркивает полезность скачивания по разным конфигурациям и использование соответствующих моделей.

Таблица 3

Данные о разбиении графов

Table 3

Data on graph partitioning

Граф	Число выделенных сообществ	Число сообществ			
		более 3вершин	3 вершины	2 вершины	1 вершина
G_1	255	26	6	6	217
G_2	242	34	7	3	198
G_3	216	33	4	6	173

Таблица 4

Данные о разбиении графов

Table 4

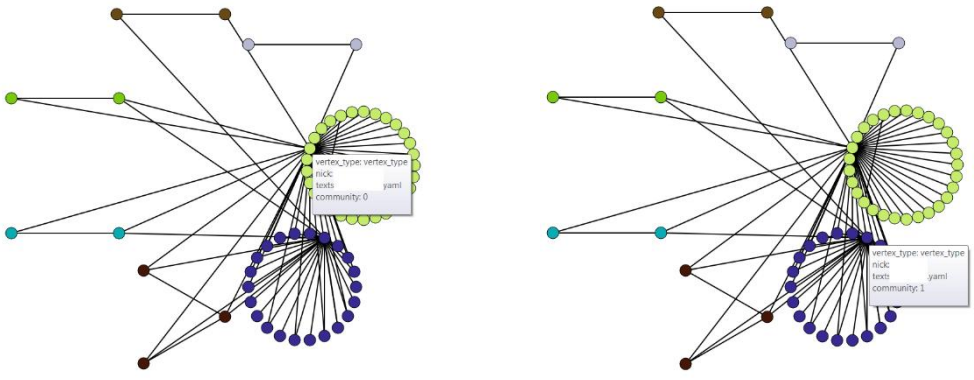
Data on graph partitioning

Граф	Размеры топ-5 сообществ	Средний размер		
		сообщества	сообщества без учета единичных	топ-10
G_1	62-39-32-30-25	2,47	10,92	26,9
G_2	40-37-36-31-30	2,61	9,86	26,7
G_3	49-44-44-33-25	2,92	10,67	29,3

Посмотрим на следующие примеры.

Пример 1. На графе G_1 самое большое сообщество состоит из 62 вершин, причем внутри имеется явное разделение (рис. 6).

Для графа G_2 те же вершины разбились на сообщества иначе – разделившись на два разных за счет большего веса на ребрах между ними (рис. 7).



а б

Рис. 6. Один из лидеров сообщества, состоящего из двух центральных и пяти близких к ним (*а*); еще один лидер того же сообщества (*б*)

Fig. 6. One of the leaders of a community consisting of two central and five close to them (*a*); another of the leaders of the same community (*b*)

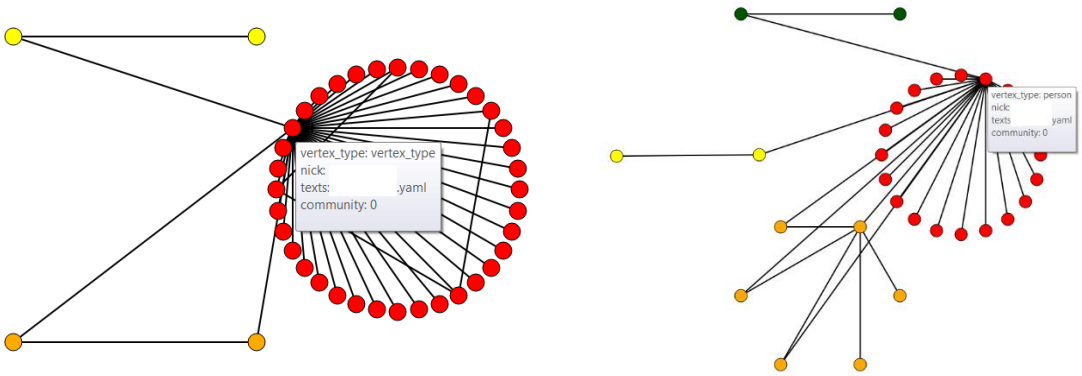
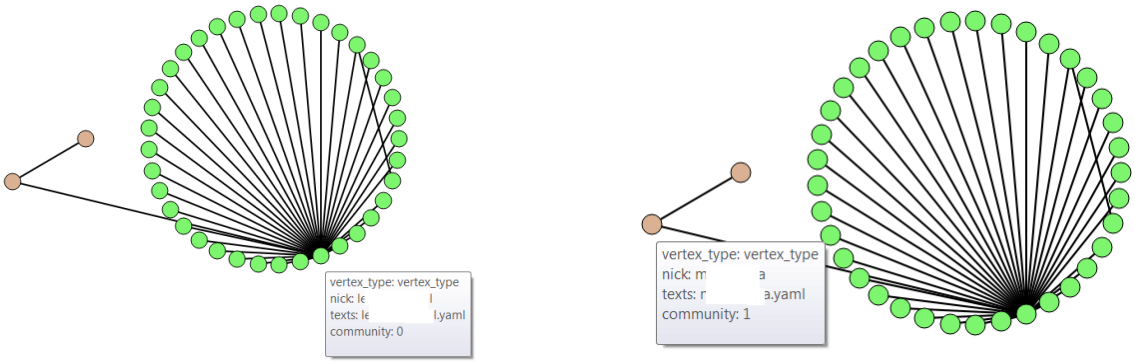
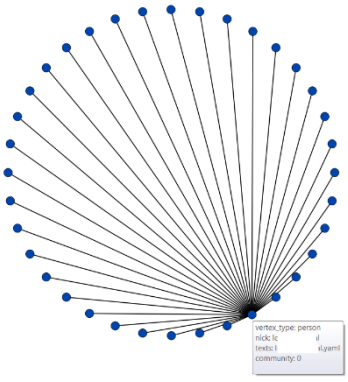


Рис. 7. Первый из лидеров теперь в своем сообществе (*а*); второй из лидеров также в своем сообществе (*б*)
Fig. 7. The first of the leaders is now in his community (*a*); the second of the leaders is also in his community (*b*)

Пример 2 состоит в наличии «хвоста» у одного из сообществ для графа $\boldsymbol{G_1}$ и его отсутствия для $\boldsymbol{G_2}$ (рис. 8).



а б



в

Рис. 8. Лидер сообщества @le***al (офиц. СМИ), граф $\boldsymbol{G_2}$ (*а*);
наличие «хвоста» в сообществе @le***al в графе $\boldsymbol{G_2}$ (*б*);
сообщество @le***al в графе $\boldsymbol{G_2}$ уже без «хвоста» (*в*)
Fig. 8. Community Leader @le** * al (ofic. Media), graph $\boldsymbol{G_2}$ (*a*);
the presence of a “tail” in the @le***al community in the $\boldsymbol{G_2}$ column (*b*);
the @le***al community in the $\boldsymbol{G_2}$ column is already without a “tail” (*c*)

Пример 3 состоит в рассмотрении вершины, соответствующей пользователю с ником ***@Mr***ay***. В графе G_1 у этой вершины 47 смежных, суммарная степень равна 73. При этом в графе G_2 степень вершины растет в 1,25 раза, а ее взвешенная степень растет почти в 2 раза (табл. 5). Это связано с большим числом ретвитов данного пользователя.

Таблица 5

Данные о вершине @Mr***ay

Table 5

Data on the vertex @Mr***ay

Граф	Степень		Взвешенная степень	
	вершины	вершины в своем сообществе	вершины	вершины в своем сообществе
G_1	47	19	73	43
G_2	59	29	145	100
G_3	65	39	171	128

Например, исследуемая вершина теперь имеет общее ребро с весом 4 с вершиной, соответствующей пользователю ***@ib***hP***. Его появление обусловлено одним ретвитом. Но общее перестроение графа повлекли и обратные примеры, часть пользователей ушла в другие сообщества, например, в графе G_1 вершина ***@y***an*** и ее небольшое сообщество из 4 вершин были в общем сообществе с ***@Mr***ay***, а в графе G_2 – уже нет. При этом вес вершины ***@Mr***ay*** значительно вырос за счет ретвитов (9, *а*, *б*). Для графа G_3, в котором добавлены еще и комментарии, рост взвешенной степени вершины уже не такой резкий, но продолжается, тут вершина ***@y***an*** и ее сообщество возвращаются (рис. 9, *в*).

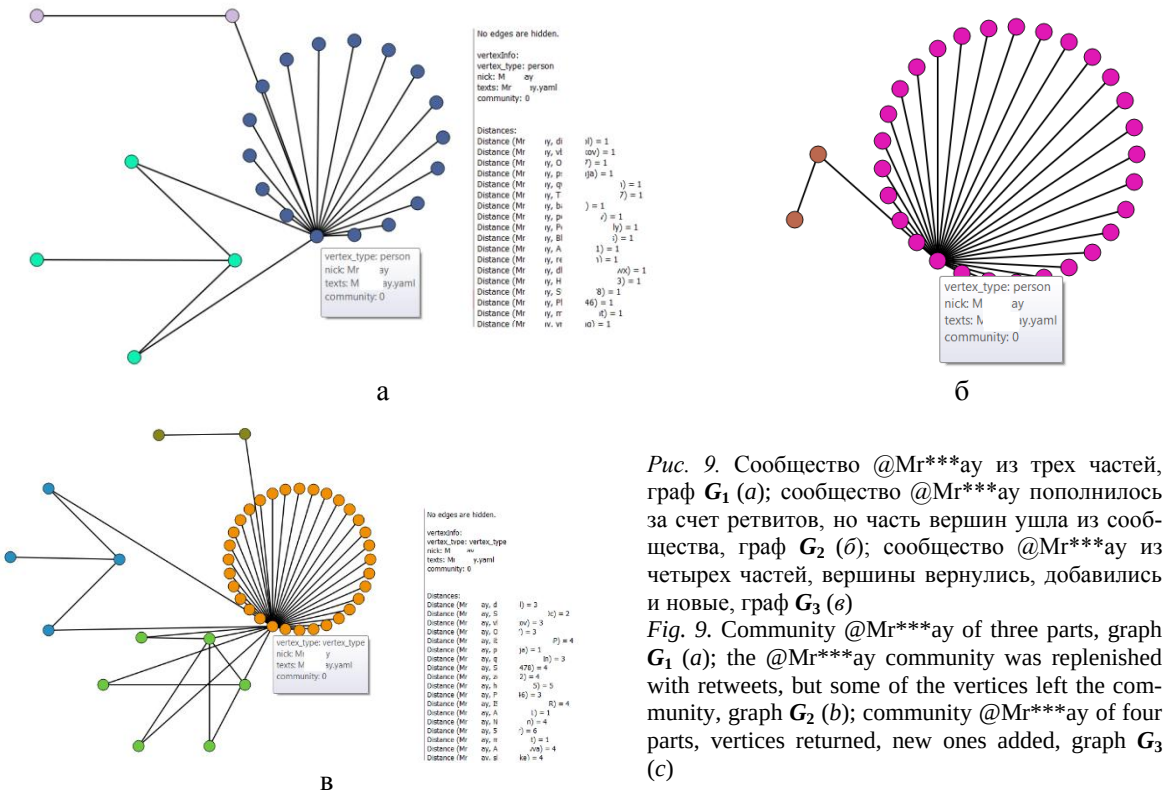


Рис. 9. Сообщество @Mr***ay из трех частей, граф G_1 (*а*); сообщество @Mr***ay пополнилось за счет ретвитов, но часть вершин ушла из сообщества, граф G_2 (*б*); сообщество @Mr***ay из четырех частей, вершины вернулись, добавились и новые, граф G_3 (*в*)

Fig. 9. Community @Mr***ay of three parts, graph G_1 (*a*); the @Mr***ay community was replenished with retweets, but some of the vertices left the community, graph G_2 (*b*); community @Mr***ay of four parts, vertices returned, new ones added, graph G_3 (*c*)

Корректируя конфигурацию для скачивания вершин, можно добиться графов с разным содержанием в зависимости от поставленной оператором цели. Первую конфигурацию – (1, 3, 0, 0)-модель – мы называем **графом общего сходства пользователей**. Третью конфигурацию – (1, 3, 2, 4)-модель – мы называем **графом информационного взаимодействия пользователей**. Таким образом, построены подходы к импорту данных и анализу катализаторов информационного воздействия в сети Twitter.

Заключение

В рамках данной работы описан выработанный авторами комплексный анализ возможностей и ограничений по импорту данных из социальной сети Twitter в контексте взаимодействия пользователей. В работе представлена методика построения (*F*, *L*, *C*, *R*)-моделей информационного взаимодействия, представлены примеры ее использования для выявления лидеров мнения групп пользователей и соответствующих катализаторов.

Авторами разработаны алгоритм и структуры данных для него, позволяющие построить взвешенные социальные графы на основе данных о взаимодействии пользователей Twitter. Описано реализованное программное обеспечение для импорта данных пользователей на основе заданных изначально оператором твитов-источников и конфигурационных параметров модели информационного взаимодействия.

Приведенные примеры показывают высокую степень актуальности достигнутых результатов. Авторам представляется целесообразным дальнейшее развитие методики построения моделей информационного взаимодействия и ее расширение на другие социальные сети.

Список литературы

1. **Лещёв Д. А., Сучков Д. В., Хайкова С. П., Чеповский А. А.** Алгоритмы выделения групп общения // Вопросы кибербезопасности. 2019. Т. 32, № 4. С. 61–71.
2. **Соколова Т. В., Чеповский А. А.** Анализ профилей сообществ социальных сетей // Системы высокой доступности. 2018. Т. 14, № 3. С. 82–86.
3. **Коломейченко М. И., Поляков И. В., Чеповский А. А., Чеповский А. М.** Выделение сообществ в графе взаимодействующих объектов // Фундаментальная и прикладная математика. 2016. Т. 21, № 3. С. 131–139.
4. **Roth M., Ben-David A., Deutscher D.** Suggesting Friends Using the Implicit Social Graph. In: KDD'10, July 25–28, 2010. Washington, DC, USA, 2010.
5. **Girvan M., Newman M.** Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 2002, vol. 99, no. 12, p. 7821–7826.
6. **Blondel V. D., Guillaume J. L., Lambiotte R., Lefebvre E.** Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008, no. 10, P10008. 12 p.
7. **Mitchell R.** Web Scraping with Python. Sebastopol, O'Reilly Media, 2015.
8. **Rosvall M.**, **Axelsson D., Bergstrom C. T.** The map equation. *The European Physical Journal Special Topics*, 2009.
9. **Chepovskiy A. A., Leshchev D. A., Khaykova S. P.** Core Method for Community Detection. In: Complex Networks & Their Applications IX. Vol. 1: Proceedings of the Ninth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2020. Springer, 2021, p. 38–50. DOI 10.1007/978-3-030-65347-7_4

References

1. **Leschyov D. A., Suchkov D. V., Khaykova S. P., Chepovskiy A. A.** Algorithms to reveal communication groups. *Voprosy kiberbezopasnosti*, 2019, vol. 32 (4), p. 61–71. (in Russ.) DOI 10.21681/2311-3456-2019-4-61-71

2. **Sokolova T. V., Chepovskiy A. A.** Analiz profilej soobshchestv social'nykh setej. *Sistemy vysokoj dostupnosti*, 2018, vol. 14, no. 3, p. 82–86. (in Russ.)
3. **Kolomejchenko M. I., Polyakov I. V., Chepovskiy A. A., Chepovskiy A. M.** Vydelenie soobshchestv v grafe vzaimodejstvuyushchikh ob'ektov. *Fundamental'naya i prikladnaya matematika*, 2016, vol. 21, no. 3, p. 131–139. (in Russ.)
4. **Roth M., Ben-David A., Deutscher D.** Suggesting Friends Using the Implicit Social Graph. In: KDD'10, July 25–28, 2010, Washington, DC, USA, 2010.
5. **Girvan M., Newman M.** Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 2002, vol. 99, no. 12, p. 7821–7826.
6. **Blondel V. D., Guillaume J. L., Lambiotte R., Lefebvre E.** Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, 2008, no. 10, P10008. 12 p.
7. **Mitchell R.** Web Scraping with Python. Sebastopol, O'Reilly Media, 2015.
8. **Rosvall M.**, **Axelsson D., Bergstrom C. T.** The map equation. *The European Physical Journal Special Topics*, 2009.
9. **Chepovskiy A. A., Leshchev D. A., Khaykova S. P.** Core Method for Community Detection. In: Complex Networks & Their Applications IX. Vol. 1: Proceedings of the Ninth International Conference on Complex Networks and Their Applications COMPLEX NETWORKS 2020. Springer, 2021, p. 38–50. DOI 10.1007/978-3-030-65347-7_4

Материал поступил в редколлегию
Received
29.03.2021

Сведения об авторах

Попов Владимир Александрович, студент магистратуры, Национальный исследовательский университет «Высшая школа экономики» (Москва, Россия)

vapopov_1@edu.hse.ru

Чеповский Александр Андреевич, кандидат физико-математических наук, доцент, Национальный исследовательский университет «Высшая школа экономики» (Москва, Россия)

aachepovsky@hse.ru

Information about the Authors

Vladimir A. Popov, Master's Student, National Research University Higher School of Economics (Moscow, Russian Federation)

vapopov_1@edu.hse.ru

Alexander A. Chepovskiy, PhD (Mathematics), Associate Professor, National Research University Higher School of Economics (Moscow, Russian Federation)

aachepovsky@hse.ru