Применение алгоритмов выявления сообществ для социологического исследования блогов: результаты пилотного исследования

Анастасия Кинчарова

Данная статья содержит некоторые результаты проекта «Разработка методологии сетевого и семантического анализа блогов для социологических задач»[1].

Введение

Содержательным фокусом проекта в целом является концептуализация ислама в блогах и анализ социальных сетей блогеров, пишущих и комментирующих посты об исламе. Одной из основных задач является группировка (выделение кластеров, сообществ) единиц наблюдения с помощью а) семантического и б) сетевого анализа и сопоставление двух полученных видов группировки для проверки гипотезы об их совпадении, хотя бы частичном.

Гипотеза, из которой мы исходим в рамках анализа социальных сетей блогеров, состоит в том, что в полной сети блогеров, пишущих об исламе в определённый период времени, можно выделить группы, внутри которых блогеры связаны друг с другом сильнее, чем с теми, кто находится за пределами этой группы. Блогеров мы рассматриваем как вершины сети, комментарии к постам каждого блогера – в качестве связей между блогерами.

Основная методическая проблема сетевого анализа состоит в подборе релевантного метода группировки и программного обеспечения, способного реализовать этот метод . В качестве метода группировки выбран метод выделения сообществ (community detection), в качестве программного обеспечения – программный пакет igraph в программной среде R.

В данной статье представлены результаты пилотного исследования одной эгосети с помощью различных алгоритмов выделения сообществ. Цель этого пилотного исследования – множественная:

[1] В данной научной работе использованы результаты, полученные в ходе выполнения проекта «Разработка методологии сетевого и семантического анализа блогов для социологических задач», выполненного в рамках Программы «Научный фонд ГУ-ВШЭ» в 2011 году.

а) опробовать процесс сбора данных, выяснить, какие трудности и вопросы при этом возникают, разрешить их на концептуальном уровне;

в) сформировать файл данных для программной среды R и пакета igraph, который будет необходим в качестве образца для построения файла данных полной сети и инструкции по формированию массива данных для полной сети;

б) визуально представить изучаемую эгосеть и оценить её структуру;

г) опробовать различные алгоритмы выделения сообществ в социальных сетях на пилотной эгосети с помощью программного пакета igraph в программной среде R и по результатам анализа оценить их релевантность в контексте поставленных в исследовании задач.

Формирование массива данных

Как было указанно во введении, объектом пилотного исследования, описанного здесь, является одна эгосеть, то есть сеть, выстроенная вокруг одного блогера. Эта сеть включает его самого и его посты об исламе за определённый выбранный период, его комментаторов и их посты об исламе за этот же период, комментаторов комментаторов и их посты об исламе за тот же период и т.д. до того момента, пока в журналах блогеров, комментировавших «исламский» пост, не окажется постов на данную тематику в указанный период времени.

Процесс формирования эгосети происходил следующим образом. С помощью поиска по блогам поисковой системы Яндекс был выбран «стартовый» пост. При поиске задавалось поисковое слово, принадлежащее к перечню «исламской» лексики – в данном случае это было словосочетание «Аллах акбар»; область поиска – блоги на платформе «Живой Журнал» (поскольку именно она ограничивает объект исследования в данном проекте в целом). Также задавался временной интервал поиска – с 14 февраля 2011 г. по 21 февраля 2011 г., и тип записей, среди которых должен был происходить поиск – в данном случае это записи (то есть посты), из поиска исключались комментарии и микроблоги (Рис. 1).

Итак, помимо того, что сеть ограничена тематикой – ислам, она также ограничена временным интервалом в одну неделю. Это оправдано, поскольку актуальность записей в блоге длится довольно недолго, это проявляется, в частности, в том, что комментарии обычно оставляют в ближайшие к дате написания поста дни. Таким образом, социальные сети в блогосфере описаны в определённом временном интервале, что соответствует их природе. Со временем сети могут значительно трансформироваться, и на больших интервалах времени актуальным стал бы скорее их лонгитюдный анализ, однако мы не ставим перед собой такую задачу.

Из поиска также были исключены микроблоги, поскольку, во-первых, они ведутся не в Живом Журнале, а на других платформах, а во-вторых, они не соответствуют требованиям к анализируемым текстам в рамках семантического анализа, с результатами которого предполагается сопоставлять результаты сетевого анализа.

Заметим, что словосочетание «Аллах акбар» регулярно встречается в контекстах, не имеющих отношения к исламу, что вызывает вопрос, является ли это свидетельством проникновения исламского дискурса в контексты, далёкие от ислама, или это случай, когда первоначально привязанное к исламу выражение утратило свой основной смысл. В первом случае мы должны интерпретировать такое употребление, во втором – можем просто проигнорировать употребление этого выражения как нерелевантного. Представляется, что первое объяснение более оправдано, и мы должны хотя бы попытаться объяснить употребление «исламской» лексики в неисламских контекстах.

Процесс формирования массива данных проходил следующим образом: фиксировался адрес URL «стартового» поста (имеющий вид http://abcd.livejournal.com/123456) и все адреса URL журналов комментаторов этого поста (имеющие вид http://efgh.livejournal.com/).

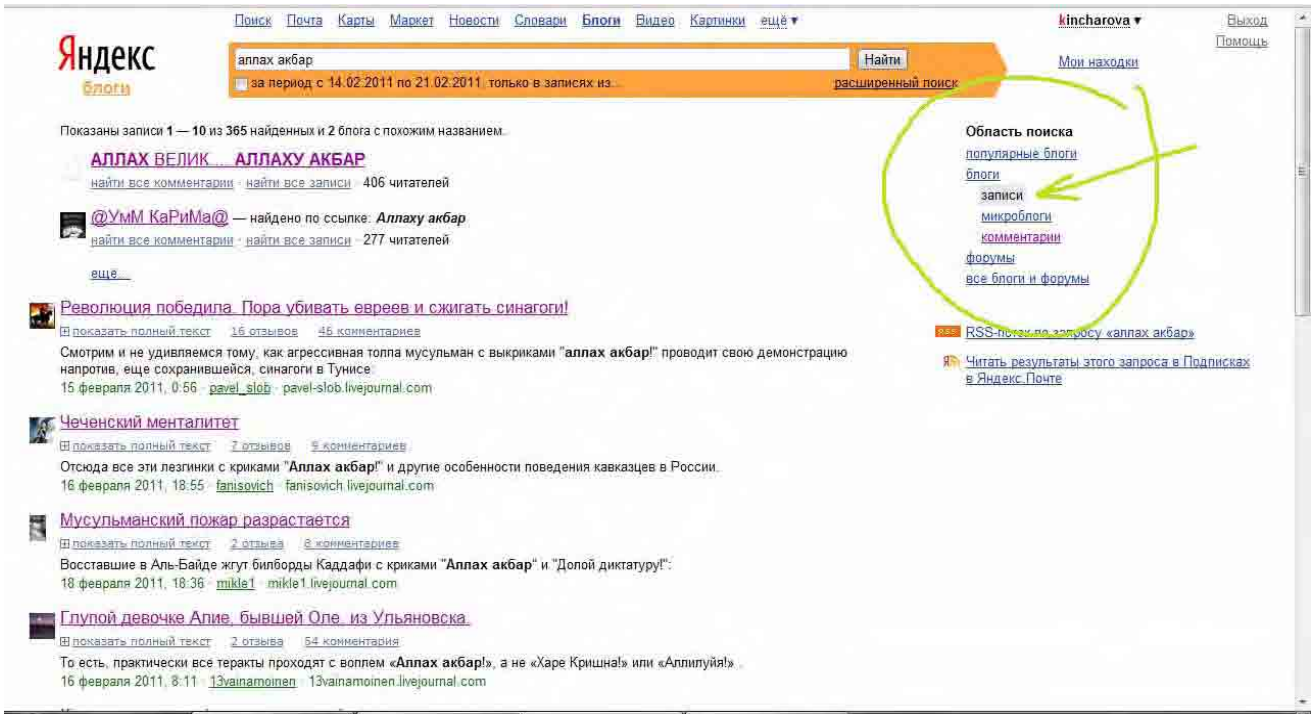


Рис. 1. Поиск «стартового» поста

Ряд вопросов, которые возникли в процессе формирования исходного массива данных, и которые необходимо разрешить:

- Включать ли в список комментаторов только комментаторов первого уровня, то есть комментаторов собственно поста, или включать и тех, кто подключается к обсуждению на более низких уровнях, то есть комментирует комментарии, а не сам пост (рис. 2)? Ответ на этот вопрос с содержательной точки зрения должен в пользу скорее второго варианта, поскольку само наличие комментария является свидетельством включённости в обсуждение тем, связанных с исламом.

- Включать ли в список комментаторов тех, кто уже комментировал этот пост раньше (то есть выше в списке комментариев)? С содержательной точки зрения это имеет смысл, так как может свидетельствовать о большой

значимости для комментатора обсуждаемой темы. Вместе с тем, фиксировать все комментарии к посту одного и того же автора на всех уровнях обсуждения, означает сильно перегружать массив данных. В результате более рациональным мы посчитали фиксировать каждого комментатора каждого поста только один раз.

Очевидно, что из числа комментаторов следует исключать ботов, отправляющих сообщения о том, что пост попал в рейтинг и т.д. Эти сообщения генерируются автоматически роботом, которого, разумеется, нельзя считать участником социальной сети. При ручном сборе данных игнорировать таких комментаторов не составляет труда, при машинном можно постараться исключить комментаторов, в имени которых присутствует сочетание букв «bot» (обычно оно есть в имени робота).

Рис. 2. Комментатор, включающийся в обсуждение, начавшееся с комментария другого комментатора

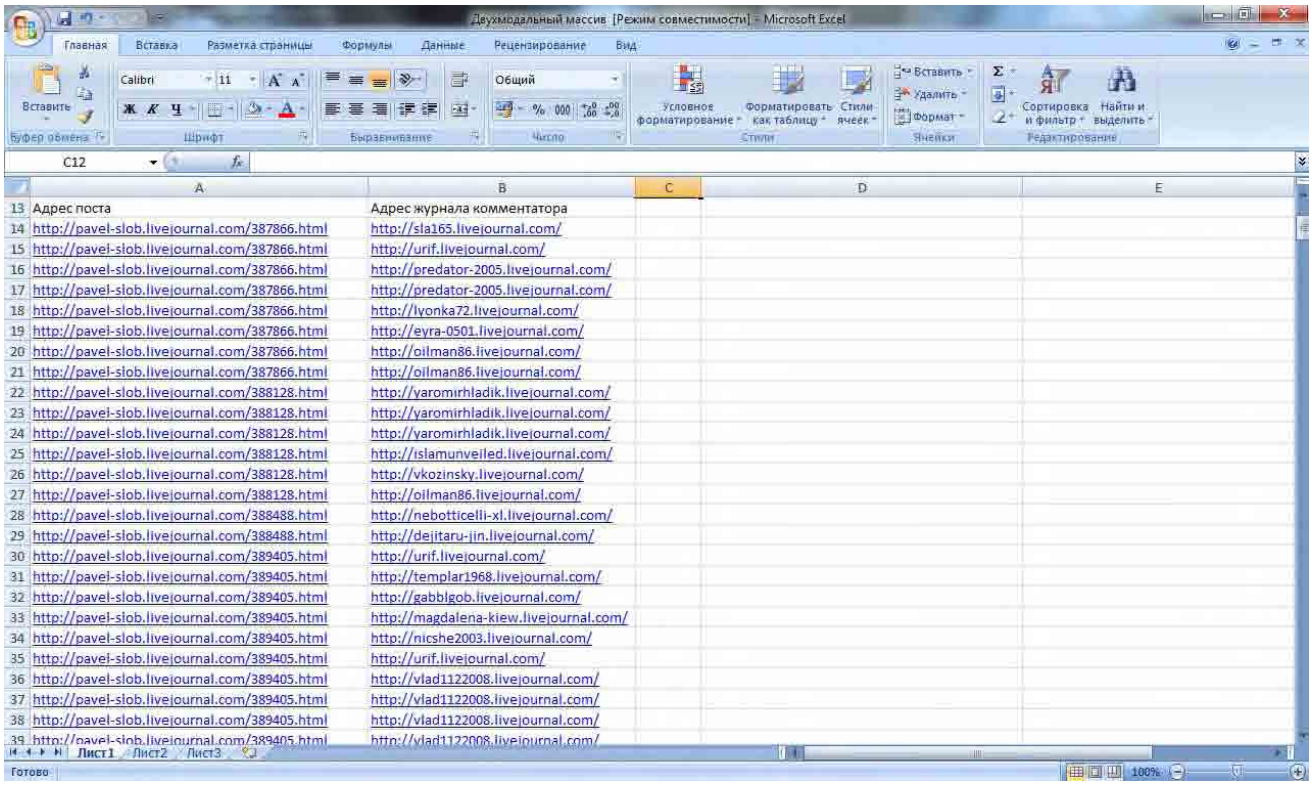
После того, как были зафиксированы адреса журналов всех комментаторов «стартового» поста, журнал автора «стартового» поста был проверен на наличие других записей, содержащих «исламскую» лексику и относящихся к изучаемому периоду. Для постов, отвечавшим этим требованиям, были также зафиксированы адрес поста и адреса журналов комментаторов. После этого мы переходили на URL –адреса журналов комментаторов (которые имеют вид http://efgh.livejournal.com/) и находили посты, которые относились к выбранному периоду (14.02.2011 – 21.02.2011) и содержали «исламскую» лексику. Процесс повторялся. Нам удалось дойти до границ сети, то есть в определённый момент у комментаторов не обнаружилось постов, затрагивавших исламскую тематику в указанный период. Это произошло на 13-м круге анализа.

Итак, «сырой» массив исходных данных представлял собой таблицу, в которой фиксировались два типа данных: адрес поста, содержащего «исламскую» лексику, и адрес журнала комментатора (Рис. 3).

Следующим шагом является приведение данного массива в вид и перенос в формат, приемлемый для работы в программной среде R.

Данный массив несёт информацию о социальной сети, связи в которой имеют направление – от комментатора к конкретному посту (направленная сеть), и которая содержит вершины двух типов – посты и журналы (бимодальная сеть). Формат входного файла, в котором будут содержаться данные для анализа в программной среде R, который мы изберём, должен отражать эти свойства сети. Кроме того, этот формат должен быть компактным, поскольку сеть, формирующаяся в основном исследовании, будет большой.

Проблема, которую необходимо решить при формировании входного графа для работы в R, связана с необходимостью зафиксировать связь поста и автора. Построить граф с более чем двумя типами вершин с помощью имеющегося программного обеспечения невозможно, а построение нескольких двумерных графов сильно усложнит анализ.



	A	B
13	Адрес поста	Адрес журнала комментатора
14	http://pavel-slob.livejournal.com/387866.html	http://sla165.livejournal.com/
15	http://pavel-slob.livejournal.com/387866.html	http://urif.livejournal.com/
16	http://pavel-slob.livejournal.com/387866.html	http://predator-2005.livejournal.com/
17	http://pavel-slob.livejournal.com/387866.html	http://predator-2005.livejournal.com/
18	http://pavel-slob.livejournal.com/387866.html	http://lyonka72.livejournal.com/
19	http://pavel-slob.livejournal.com/387866.html	http://eyra-0501.livejournal.com/
20	http://pavel-slob.livejournal.com/387866.html	http://oilman86.livejournal.com/
21	http://pavel-slob.livejournal.com/387866.html	http://oilman86.livejournal.com/
22	http://pavel-slob.livejournal.com/388128.html	http://yaromirhladik.livejournal.com/
23	http://pavel-slob.livejournal.com/388128.html	http://yaromirhladik.livejournal.com/
24	http://pavel-slob.livejournal.com/388128.html	http://yaromirhladik.livejournal.com/
25	http://pavel-slob.livejournal.com/388128.html	http://islamunveiled.livejournal.com/
26	http://pavel-slob.livejournal.com/388128.html	http://vkozinsky.livejournal.com/
27	http://pavel-slob.livejournal.com/388128.html	http://oilman86.livejournal.com/
28	http://pavel-slob.livejournal.com/388488.html	http://nebotticelli-xl.livejournal.com/
29	http://pavel-slob.livejournal.com/388488.html	http://dejitaru-jin.livejournal.com/
30	http://pavel-slob.livejournal.com/389405.html	http://urif.livejournal.com/
31	http://pavel-slob.livejournal.com/389405.html	http://templar1968.livejournal.com/
32	http://pavel-slob.livejournal.com/389405.html	http://gabblgob.livejournal.com/
33	http://pavel-slob.livejournal.com/389405.html	http://magdalena-kiew.livejournal.com/
34	http://pavel-slob.livejournal.com/389405.html	http://nicshe2003.livejournal.com/
35	http://pavel-slob.livejournal.com/389405.html	http://urif.livejournal.com/
36	http://pavel-slob.livejournal.com/389405.html	http://vlad1122008.livejournal.com/
37	http://pavel-slob.livejournal.com/389405.html	http://vlad1122008.livejournal.com/
38	http://pavel-slob.livejournal.com/389405.html	http://vlad1122008.livejournal.com/
39	http://pavel-slob.livejournal.com/389405.html	http://vlad1122008.livejournal.com/

Рис. 3. Первичные данные пилотного исследования

Наше предложение состоит во включении автора в список комментаторов с приданием ему веса 10 (или любого другого, значительно отличающегося в большую сторону от веса остальных комментаторов). Другим решением будет

приписать атрибут авторства или комментирования каждому блогеру, связанному с конкретным постом.

Относительно возможности обнаружения сообществ в бимодальных сетях в документации к пакету igraph информации нет, в источниках, в которых предлагается алгоритм, также нет специальной информации о том, как использовать данные алгоритмы применительно к бимодальному графу. Очевидно, что в данном исследовании граф для проведения процедуры выделения сообществ должен содержать вершины одного типа.

Поскольку полученный массив данных содержит двудольный граф, а алгоритмы выделения сообществ работают с однодольными графами, необходимо привести двудольных граф к однодольному. Для этого в пакете igraph для R есть функция bipartite.projection. Данная функция превращает двудольный граф в два однодольных, состоящих из вершин одного из типов.

Однако bipartite.projection работает таким образом, что применительно к комментаторам он устанавливает для каждого из них связь со всеми, кто комментировал тот же пост. Такая сеть будет отличаться от сети, где устанавливается связь от комментатора к автору поста. С содержательной точки зрения правомерность использования графа, полученного в результате проекции, можно оценить по-разному: если предполагать, что комментатор поста читает все комментарии к этому посту, которые были написаны и раньше, и позже его собственного, то такая сеть вполне адекватна реальности. Однако если точно можно сказать только то, что автор комментария прочитал сам пост, и ничего нельзя сказать о том, читал ли он комментарии, этот вариант является неправомерным.

В данной статье мы откажемся от использования в качестве пилотной бимодальную сеть и будем использовать одномодальную, построенную на основе направленных связей блогеров – от комментатора к автору поста.

Итак, во входном файле данных будет список комментаторов, список комментируемых и список весов (количество постов данного блогера, к которым оставил комментарии данный комментатор, при этом несколько комментариев данного комментатора к одному и тому же посту учитываться не будут). Проблема необходимости обозначать связь автора с его постом фактически исчезает – такие связи просто удаляем (в противном случае мы получим петлю (loop), которая может затруднить некоторые процедуры анализа, а в содержательном не получим никакого приращения информации).

Далее мы приводим обзор и описываем опыт использования алгоритмов выделения сообществ, которые доступны в программном пакете igraph для среды R. Всего их шесть, все они предполагают измерение модульности.

При оценке алгоритмов мы будем использовать тестовую сеть, состоящую из 301 вершины и 346 ребер, эта сеть направленная, взвешенная, содержит вершины одного типа, которые представляют собой блогеров, связи – комментарии к постам друг друга.

Результатом работы алгоритма должна быть информация о принадлежности каждой вершины к тому или иному сообществу. Алгоритмы, реализация которых в R не даёт такой возможности, нам не подходят, поскольку именно принадлежность вершины (блогера) к тому или иному сообществу является самым значимым выводом сетевого анализа в нашем исследовании.

Что касается работы с направленными и взвешенными графами, то в принципе мы можем пренебречь тем, что наша сеть по своей природе направленная и взвешенная. Однако в любом случае это большая сеть и алгоритм для выделения сообществ должен быть рассчитан на работу с такой сетью.

Задача сетевого анализа

Задачей сетевого анализа является «разделить» всю сеть блогеров, в чьих постах за определённый период содержатся высказывания об исламе, на некие кластеры, группы и определить принадлежность каждого блогера к тому или иному кластеру. Заметим, что мы будем искать только один кластер для каждого блогера, хотя в сети могут существовать и вероятнее всего существуют блогеры, которые принадлежат одновременно к разным группам, сообществам. Ограничение связано с тем, что проанализированные нами методы и алгоритмы анализа в настоящее время не позволяют производить такого рода вычисления.

Что такое сообщество? С содержательной точки зрения – это группа вершин сети, участники которой связаны друг с другом значительно теснее, чем с остальными вершинами сети. На изображении сети такие группы выглядят как «сгустки» вершин, соединённые друг с другом множеством связей.

С формальной точки зрения можно говорить, что «структура графа образована сообществами, если он отличается от случайного графа» [1, 90]. В случайном графе связи между вершинами распределены относительно равномерно, поэтому если применительно к изучаемому графу обнаруживается, что это не так, то можно говорить о существовании в данной сети сообществ. Случайный граф выступает здесь нулевой моделью, с которой сравнивают изучаемый граф.

В целом разделение сети на кластеры – не новая задача в рамках сетевого анализа, однако методы и алгоритмы, которые мы применяем в данном исследовании, были разработаны относительно недавно (начиная с 2000-х годов).

До этого времени идея разделения сетей на некие имеющие содержательный смысл части развивалась в двух направлениях: во-первых, в связи с идеями разделения графа (graph partitioning) в рамках теории графов и компьютерных науках, во-вторых, в социологии как метод иерархической кластеризации [2].

Однако оба эти метода для полноценного выделения сообществ в сети были недостаточно хороши. Компьютерные науки ставят задачу разделения графа следующим образом: необходимо разделить некоторое количество вершин сети на известное количество групп определённого размера таким образом, чтобы количество связей между вершинами было минимальным [1, 90]. Такая постановка задачи актуальна при построении, например, компьютерных сетей.

Однако так ставить задачу применительно к социальным сетям невозможно, потому что исследователь, как правило, не знает, сколько сообществ обнаружится в данной сети и какого они будут размера. Очень вероятно, что они будут не равны между собой по величине.

Группа методов иерархической кластеризации предоставляет более подходящие возможности анализа сообществ. В основе этих методов лежит измерение сходства между парами вершин в соответствии с некими значимыми критериями. Затем осуществляется последовательное объединение вершин, обнаруживших наибольшее сходство. Данная группа методов иерархической кластеризации называется агломеративной, и у неё есть ряд недостатков, о которых упоминают, например, М. Ньюман и М. Гирван [3].

Другой вариант иерархической кластеризации состоит, наоборот, в выделении в отдельный кластер вершин, имеющих наименьшие показатели сходства, при том, что первоначально вся сеть рассматривается как отдельный кластер. Данная группа методов называется дивизимными методами. Оба варианта иерархической кластеризации в большой степени зависят от того, как измеряется сходство между вершинами. Кроме этой зависимости иерархическая кластеризация часто ошибочно определяет принадлежность даже важных вершин, рассматривает слабо связанные с другими вершины как отдельные кластеры и плохо измерима [1, 93].

Не так давно был предложен метод, который стал своего родом прорывом в исследовании сообществ и подтолкнул ученых к развитию этого направления. Речь идет о функции модульности, которую предложили М. Ньюман и М. Гирван [3], и которая сейчас используется в большинстве алгоритмах выделения сообщество. Модульность (modularity) – это функция, позволяющая оценить «качество разделения графа на кластеры» [1, 86].

Согласно этой функции «субграф представляет собой сообщество, если число ребер внутри субграфа превышает ожидаемое число внутренних ребер, которое этот субграф имел бы в нулевой модели» [1,86]. Нулевой моделью здесь предполагается, что ожидаемая последовательность степеней соответствует реальной последовательности степеней графа. Это более строгое требование, чем требование соответствия распределения степеней в графе [1, 86].

Модульность является «одновременно глобальным критерием определения сообщества, функцией качества и ключевым ингредиентом наиболее популярных методов кластеризации графа» [1, 86].

Алгоритмов выделения сообществ, в том числе использующих функцию модульности, в настоящее время разработано много, об этом свидетельствует обширный обзор С. Фортунато [1]. Однако не все они могу т быть использованы в социологических исследованиях, поскольку зачастую не интегрированы в доступную к использованию неспециалистом программную среду.

Тестирование алгоритмов выделения сообществ с помощью пакета igraph в программной среде R

Алгоритм «walktrap.community»

Данный алгоритм основан на идее, что случайные блуждания оказываются «пойманы» в той части графа, внутри которого связи плотнее, чем окружающие, то есть там, где находится сообщество [4].

Вычислительная сложность данного алгоритма составляет $O(n^2 \log n)$ для редких сетей.

Исходя из особенностей такого понятия, как случайные блуждания, авторы алгоритма рассчитывают расстояние между вершинами или между группами вершин, которое в свою очередь зависит от вероятности достижения от одной вершины до другой в процессе случайного блуждания. Данный показатель оказывается большим, если вершины находятся в разных сообществах в графе, и маленьким, если они находятся в одном сообществе. Далее они осуществляют иерархическую кластеризацию агломеративным способом на основе метода Уорда: вершины объединяются в сообщества на основе выбора наименьшего среднего квадрата расстояний между ними. После того, как вершина присоединена к какому-либо сообществу, расстояния между вершинами и сообществами пересчитываются.

Оценка качества полученной модели осуществляется с помощью показателя модульности, о котором была речь выше – максимальный показатель модульности соответствует наилучшему варианту выделения сообществ.

Авторы сравнили данный алгоритм с рядом других алгоритмов выделения сообществ, из представленных здесь алгоритмов среди них был алгоритм М. Гирван и М. Ньюмана, и данный алгоритм превзошел их по ряду показателей, в том числе по вычислительной скорости (массив в 100 000 вершин данный алгоритм обрабатывает около двух часов).

Алгоритм «walktrap.community» опробован на тестовой сети блогеров (301 вершина, 346 ребер). После расчетов был проведён визуальный анализ – сопоставлялась визуальная репрезентация графа и результаты разбиения на сообщества, проведённые с помощью функции walktrap.community. При этом вершины были сгруппированы для большей наглядности с помощью алгоритма построения графов Фруктермана-Рейнголда (Fruchterman-Reingold).

Итак, после загрузки файла с данными, задается команда на определение сообществ. Результат представляет собой несколько блоков информации. Во-первых, список слияний. Изначально каждая вершина представляет собой отдельный кластер и имеет номер от 0 до n-1 (если в массиве n вершин). Первое сообщество, получившееся в результате слияния двух вершин, получает номер n, второе – n+1 и т.д., поэтому в принципе можно отследить порядок объединения и какие именно вершины участвовали в нём на каждом шаге.

```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

> g <- read.graph("graph9.10.2011.short.ncol", format="ncol", directed=TRUE)
> walktrap.community(g, weights = E(g)$weight)
$merges
       [,1] [,2]
  [1,]    1   20
  [2,]   18  294
  [3,]   57  122
  [4,]  103  302
  [5,]    9  211
  [6,]  112  301
  [7,]   39  281
  [8,]   64   65
  [9,]   53  204
 [10,]   79  306
 [11,]   66  176
 [12,]  213  307
 [13,]   13  147
 [14,]   14   95
 [15,]   30  119
 [16,]  193  303
 [17,]   73   74
 [18,]  210  310
 [19,]   50  225
 [20,]  272  311
 [21,]   56  316
 [22,]  197  313
 [23,]  286  312
 [24,]  249  322
 [25,]   85  265
 [26,]   19   22
 [27,]   32   33
 [28,]    5  293
 [29,]   77  138
 [30,]   90  292
 [31,]  118  242
 [32,]  133  331
 [33,]    3   43
 [34,]   27   28
 [35,]  173  308
 [36,]    6    7
```

Второй блок данных – величина показателя модульности. Можно проследить, на каком шаге слияния кластеров, она была наибольшей. Этот шаг 282, показатель модульности равен 0,802803139. Это наилучший вариант деления.

```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

[291,]  585  590
[292,]  362  588
[293,]  586  589
[294,]  497  583
[295,]  591  593
[296,]  425  595
[297,]  592  596
[298,]  559  597
[299,]  450  598
[300,]  594  599

$modularity
  [1]  0.000000000 -0.010903565 -0.001161101  0.006205229  0.010995565  0.015938176  0.025391951  0.032548903  0.037386824  0.042367504  0.051602383
 [12]  0.058987748  0.066116148  0.071033380  0.075861785  0.080759982  0.085651835  0.088132657  0.092618442  0.097573743  0.102478285  0.107357448
 [23]  0.112261990  0.116995222  0.121887075  0.124380587  0.126823341  0.129294647  0.131750090  0.134224568  0.136721253  0.139227455  0.141720968
 [34]  0.144179584  0.146673096  0.151558604  0.156571008  0.161332792  0.163842166  0.166342024  0.168835536  0.173644906  0.176128901  0.178628759
 [45]  0.181036616  0.183501577  0.185995089  0.188466395  0.190912321  0.193402661  0.195870794  0.198342100  0.200775336  0.203275194  0.205635465
 [56]  0.208128977  0.210584421  0.213065244  0.215555584  0.218064958  0.220504540  0.222909225  0.225415427  0.227877215  0.230202590  0.232645344
 [67]  0.235113477  0.237543541  0.239900640  0.242390980  0.244878148  0.247314557  0.249811242  0.252276203  0.254753853  0.257076055  0.259544188
 [78]  0.262034528  0.264436041  0.266862933  0.269353273  0.271792854  0.274146781  0.276605397  0.279057668  0.281544836  0.284009796  0.286446205
 [89]  0.288765235  0.291198472  0.293622192  0.296096670  0.298447424  0.300931419  0.303393207  0.305829616  0.308227957  0.310724641  0.313211809
[100]  0.315527667  0.317992627  0.320422692  0.322928894  0.325416061  0.327836608  0.330330121  0.332791909  0.335241008  0.337588589  0.340072585
[111]  0.342543890  0.345031058  0.347343743  0.349776980  0.352210216  0.354637108  0.357032276  0.359513099  0.361930474  0.364389089  0.366733499
[122]  0.369043012  0.371504800  0.373963416  0.376393480  0.378861613  0.381345608  0.383829604  0.386253323  0.388699249  0.391180072  0.393521309
[133]  0.395913304  0.398219645  0.400709985  0.403187635  0.405681148  0.408111212  0.410566655  0.412987203  0.415471198  0.417809262  0.420267878
[144]  0.422571046  0.424959869  0.427402623  0.429911997  0.432392820  0.434727712  0.437027708  0.439502186  0.441954457  0.444381349  0.446836792
[155]  0.449222443  0.451519266  0.453850986  0.456331808  0.458771390  0.461065041  0.463514139  0.465842687  0.468225165  0.470648884  0.473101155
[166]  0.475391634  0.477717009  0.480153418  0.482599344  0.484978650  0.487265956  0.489588158  0.492037257  0.494457804  0.496741937  0.499060967
[177]  0.501437101  0.503879855  0.506313091  0.508594052  0.510909910  0.513355836  0.515728797  0.518006586  0.520319271  0.522758853  0.525188917
[188]  0.527463533  0.529833322  0.532142834  0.534414278  0.536920480  0.539356890  0.541663230  0.543931501  0.546298117  0.548601285  0.551034522
[199]  0.553397966  0.555697961  0.557994785  0.560355056  0.562648707  0.565005806  0.567296284  0.569713658  0.572127861  0.574538891  0.576946748
[210]  0.579351433  0.581752946  0.584246458  0.586752660  0.589255690  0.591761892  0.594264921  0.596764779  0.601307666  0.603778972  0.606247105
[221]  0.608712066  0.611173854  0.613632470  0.616113293  0.618590943  0.621065421  0.623536727  0.626004860  0.628469821  0.630972851  0.633475880
[232]  0.635955255  0.638497802  0.641007176  0.643519723  0.646032270  0.648544517  0.651057363  0.653569910  0.655739837  0.657897074  0.660409621
[243]  0.662922168  0.665431543  0.667487263  0.672042840  0.674352353  0.676817314  0.679304481  0.681788477  0.686597847  0.688555222  0.690997976
[254]  0.695321968  0.699979062  0.708855459  0.710974627  0.713176278  0.719210197  0.724083016  0.726525769  0.734818443  0.737007404  0.740338432
[265]  0.748783382  0.752641029  0.759772602  0.765803349  0.771564441  0.774927193  0.780827872  0.783013660  0.786604826  0.788543167  0.790319715
[276]  0.792229505  0.793977501  0.795858739  0.796921496  0.797099150  0.798133355  0.802803139  0.800277903  0.802368520  0.801010729  0.793929915
[287]  0.797279978  0.796911978  0.787388411  0.781509939  0.782426765  0.724911014  0.724780945  0.704414088  0.704090502  0.511017772  0.485870096
[298]  0.201711197  0.166370575  0.134675050  0.000000000
```

Далее программа выдаёт имена вершин.

```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

[177]  0.501437101  0.503879855  0.506313091  0.508594052  0.510909910  0.513355836  0.515728797  0.518006586  0.520319271  0.522758853  0.525188917
[188]  0.527463533  0.529833322  0.532142834  0.534414278  0.536920480  0.539356890  0.541663230  0.543931501  0.546298117  0.548601285  0.551034522
[199]  0.553397966  0.555697961  0.557994785  0.560355056  0.562648707  0.565005806  0.567296284  0.569713658  0.572127861  0.574538891  0.576946748
[210]  0.579351433  0.581752946  0.584246458  0.586752660  0.589255690  0.591761892  0.594264921  0.596764779  0.601307666  0.603778972  0.606247105
[221]  0.608712066  0.611173854  0.613632470  0.616113293  0.618590943  0.621065421  0.623536727  0.626004860  0.628469821  0.630972851  0.633475880
[232]  0.635985255  0.638497802  0.641007176  0.643519723  0.646032270  0.648544817  0.651057363  0.653569910  0.655739837  0.657897074  0.660409621
[243]  0.662922168  0.665431543  0.667487263  0.672042840  0.674352353  0.676817314  0.679304481  0.681788477  0.686597847  0.688555222  0.690997976
[254]  0.695321968  0.699979062  0.708855459  0.710974627  0.713176278  0.719210197  0.724083016  0.726525769  0.734818443  0.737007404  0.740338432
[265]  0.748783382  0.752641029  0.759772602  0.765803349  0.771564441  0.774927193  0.780827872  0.783013660  0.786604826  0.788543167  0.790319715
[276]  0.792229505  0.793977501  0.795858739  0.796921496  0.797099150  0.798133355  0.802803139  0.800277903  0.802368520  0.801010729  0.793929915
[287]  0.79727997 8  0.796911978  0.787388411  0.781509939  0.782426765  0.724911014  0.724780945  0.704414088  0.704090502  0.511017772  0.485870096
[298]  0.201711197  0.166370575  0.134675050  0.000000000

$labels
  [1] "23marazumatik23"      "abu-yunus"            "444ellendg"           "chaotickgood"         "4epenoug"
  [6] "jenna-kristiana"      "a343"                 "enverkadirov"         "abacad"               "nurmagomedova"
 [11] "abdulkhamid"          "kurbanov05dag"        "abdur644"             "assal"                "kolobok1973"
 [16] "timourgazi"           "abortnik"             "absentee-lj"          "malikit"              "alstrangers"
 [21] "mmurtaza"             "muslim-ru"            "rystam4ik"            "tim-ar"               "umm-muslimka"
 [26] "adamka2010"           "stalin-ist"           "adiaphora22"          "griseopallidus"       "a-eliseev"
 [31] "semargl-bird"         "aerys"                "agathpher"            "systemity"            "aid-a"
 [36] "aisiati"              "mirty"                "ai-zhilin"            "aka-human"            "pavel-slob"
 [41] "akazakof"             "akolbeseder"          "soloroma"             "aleksgoro"            "alenka-moscva"
 [46] "alersandra"           "alexeymitin"          "alex-nik"             "alexx71"              "alil-muhammad"
 [51] "v-sidorov"            "alsharkasi"           "dorogoi-brat"         "alteisa"              "andrey2"
 [56] "snowtomcat"           "anglares"             "eyra-0501"            "anti-nlp"             "antioha"
 [61] "arctic-lane"          "ardalion5000"         "argonov"              "arnold65"             "arpadhaizy"
 [66] "gabblgob"             "nicshe2003"           "assabur"              "azrail-thul42"        "baba-gamp"
 [71] "darqawi"              "bacian"               "11000"                "baikonur113"          "junayd-dag"
 [76] "baizhi"               "baltvilks"            "heshilao"             "banderrass"           "ben-ehlmann"
 [81] "bessonov-perm"        "b-graf"               "bilan-tsuro"          "bk77"                 "black-sea-white"
 [86] "strezzen"             "booooo"               "brewbuilder"          "brusilov-14"          "buzuzi"
 [91] "vakeli"               "camrad"               "cczy"                 "cdissident"           "cheslavkon"
 [96] "chudokol"             "claire-1973"          "cowboy-pete"          "den-shi"              "cpmaker"
[101] "crazycat-meyr"        "dadrov"               "darbazi-anti"         "darginec"             "dariarusanova"
[106] "darwish-asiya"        "d-avaliani"           "gelavasadze"          "dejitaru-jin"         "deputat-yehu"
[111] "d-i-k-o-b-r-a-z"      "dimon-73"             "dionalco"             "discofuhrer"          "dr-repellent"
[116] "ecilop1"              "edn"                  "egogeo"               "elf-ociten"           "el-murid"
[121] "emmyixrenznaet"       "konha"                "sohryu-1"             "fa"                   "gluraba"
[126] "fanny-caplan"         "fatracoon"            "felishte"             "felixwankel"          "felyzen"
```

А затем указывает, к какой группе относится каждая вершина. Этот показатель дается применительно к тому варианту разделения на группы, который характеризует наибольшая величина модульности.

```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

[171] "kavadim"                  "kazan-love"           "keeperpine"           "khoroshayashpit"      "koba777"
[176] "kondratjev"               "kostigoff"            "kroshka-rena"         "kurtis10"             "lactoriacornuta"
[181] "ladi-jane"                "leoramschtain"        "loly-girl"            "lyonka72"             "ma535468"
[186] "maarulla"                 "magdalena-kiew"       "major7007"            "makhk"                "mamirgov"
[191] "margret-stwt"             "maxalex"              "meharher"             "mike-2304"            "mikes68"
[196] "misha-samarsky"           "mos-art"              "ms-rowshan"           "murix"                "muslim-qazaqi"
[201] "nalya-om"                 "na-vihod"             "nazhdak-vellesa"      "nebotticelli-xl"      "nedogurok"
[206] "nezloygad"                "neznayushiy"          "nikadubrovsky"        "nikola1989"           "noxx77"
[211] "n-ramazan"                "sestramobislame"      "o55"                  "oilman86"             "ostranna"
[216] "savall"                   "oton75"               "oubique"              "pan-baklazhan"        "patetlao"
[221] "petrvictorovich"          "phago-lov"            "povischuk"            "pravda-religion"      "praviloman"
[226] "pravover"                 "predator-2005"        "proffcom"             "pronzus"              "proxodimez"
[231] "psr1913plus16"            "puffinus"             "qingtiger"            "qi-tronic"            "quillonweqip"
[236] "rada-maslova"             "red-ptero"            "remeslenik"           "renar"                "rositsa"
[241] "rukhakasia"               "ru-partizanen"        "ruveticus"            "ryboved"              "sabinulya"
[246] "samael-harah"             "samuel-smith"         "sandroba"             "savmaxru"             "serezha-la"
[251] "sergiej"                  "shavu"                "shomer-shomron"       "shraibman"            "shtabss"
[256] "sla165"                   "som"                  "starik-henc"          "sunsniper"            "superrex"
[261] "suresnois"                "surgeon05"            "svinkina"             "svjatko"              "t-34-85"
[266] "tanasquel"                "tanatt"               "templar1968"          "terem-pro"            "the-berlin-cake"
[271] "theodore-r"               "theriandrora"         "to-be-friends"        "toddyav"              "totalitarizm86"
[276] "trezvy-gruzchik"          "udif76"               "uhva"                 "umar-azzam"           "umm-talib"
[281] "uppsss"                   "urif"                 "usama-ben-yorik"      "valeriuus"            "vaykuneci"
[286] "vkozinsky"                "vlad1122008"          "vlip"                 "v-valentinov"         "werwolf-tamboff"
[291] "whitesip"                 "wizard-sk"            "woicha"               "wolf-kitses"          "woodweend"
[296] "www.facebook.com/Loly.garu" "xdrider"            "x-erx"                "yaromirhladik"        "yosha-orlow"
[301] "yrbys"

$membership
  [1]  1  1  5  5  5  5  7  7  1  1  1  1  6 15  6  2  6  2  2  1  1  1  1  1  1  2  2  8  8 16  6  5 11 11 15  3  3 11  3  3  2 19  3  5 11 11 11  6  1  2
 [51]  2  1  2 17 18 18  4  4  4  2  4  6  1  6  0  0  0  1  1  2  2  0  0  1  1  6 10 10 15  1  6  6  1  6 14 14 15 15  3  9  9 15  0  5  9  6 19 13 13  8
[101]  3  6 10  2  2  2 12 12  3  0  6  0  1 14  6  0  2 10  6  6 17  7  4  1  1  0  3  6  3  0  0  6  3  6  3 11 10 10 10  1  6  1  1  8  9  3  3 15  5  5
[151]  6  6  4 10  7  0  2 17  0 10  2  3  3  1  0 11  1  2 17  1  0  6  9  0  1 13  0  1  9  8  3 16  1  3 11  1  3  1  6  1 17 13 11  4  6  6 14 15  5  2
[201]  0  1 16  3 17  4  2  5 14  5  1  1 11  3  0  0 16 16  0  6 17  8  1  3  4  2  3  1  6  1  6  4  3  5  2  3 12 11  0  6 14  2  6 13  5  1  0 11  2 15
[251]  0  4  0  5  6  3 10 11  2  6 10  1  8  4  0 14  6  3  1  0  2 17  0 19  5 14  8 15  2  1  3  3 15  1 18  3  3  3  5  1  0  5  9  5  2  1  6 13  3  4
[301] 10

attr(,"class")
[1] "igraph.walktrap"
>
```

Итак, мы получили весьма высокий показатель модульности 0,802803139, которому соответствует следующее распределение блогеров по сообществам:

Таблица 1

Распределение вершин по сообществам в результате применения алгоритма walktrap.community

№	Имя блогера	Сообщество, к которому принадлежит вершина	
		при random walk = 4шага	при random walk = 2шага
65	arpadhaizy	0	4
66	gabblgob	0	4
67	nicshe2003	0	4
72	bacian	0	14
73	l1000	0	14
93	cczy	0	4
110	deputat-yehu	0	27
112	dimon-73	0	14
116	ecilop1	0	4
126	fanny-caplan	0	27
130	felyzen	0	23
131	senseisekai	0	23
156	incogn1too	0	4
159	ishkur777	0	4
165	jidovska-morda	0	4
171	kavadim	0	14
174	khoroshayashpit	0	4
177	kostigoff	0	4
201	nalya-om	0	4
215	ostranna	0	22
216	savall	0	22
219	pan-baklazhan	0	4
239	renar	0	4
247	samuel-smith	0	14
251	sergiej	0	14
253	shomer-shomron	0	4
265	t-34-85	0	4
270	the-berlin-cake	0	4
273	to-be-friends	0	4
291	whitesip	0	4
1	23marazumatik23	1	3
2	abu-yunus	1	3
9	abacad	1	3
10	nurmagomedova	1	3
11	abdulkhamid	1	28
12	kurbanov05dag	1	28
20	alstrangers	1	3
21	mmurtaza	1	3
22	muslim-ru	1	3
23	rystam4ik	1	3
24	tim-ar	1	3
25	umm-muslimka	1	1
49	alexx71	1	3
52	alsharkasi	1	1
63	argonov	1	3

68	assabur	1	1
69	azrail-thul42	1	3
74	baikonur113	1	1
75	junayd-dag	1	1
80	ben-ehlmann	1	3
83	bilan-tsuro	1	3
113	dionalco	1	3
124	fa	1	3
125	g1uraba	1	3
140	giovinco	1	3
142	gool	1	3
143	gorezz77	1	3
164	jelecka	1	1
167	ka-ib	1	3
170	kasaty21	1	3
175	koba777	1	3
178	kroshka-rena	1	26
183	loly-girl	1	3
186	maarulla	1	3
188	major7007	1	3
190	mamirgov	1	3
202	na-vihod	1	3
211	n-ramazan	1	3
212	sestramobislame	1	3
223	povischuk	1	3
228	proffcom	1	3
230	proxodimez	1	3
246	samael-harah	1	3
262	surgeon05	1	3
269	terem-pro	1	3
280	umm-talib	1	3
284	valeriuus	1	26
290	werwolf-tamboff	1	3
296	www.facebook.com/Loly.garu	1	3
16	timourgazi	2	1
18	absentee-lj	2	0
19	malikit	2	0
26	adamka2010	2	0
27	stalin-ist	2	0
41	akazakof	2	0
50	alil-muhammad	2	0
51	v-sidorov	2	0
53	dorogoi-brat	2	0
60	antioha	2	1
70	baba-gamp	2	0
71	darqawi	2	0
104	darginec	2	0
105	dariarusanova	2	0
106	darwish-asiya	2	0
117	edn	2	0
157	isa-abu-ali	2	0
161	islamkniga	2	0
168	kaminskii-88	2	0
200	muslim-qazaqi	2	0
207	neznayushiy	2	0
226	pravover	2	0
235	quillonweqip	2	1
242	ru-partizanen	2	1
249	savmaxru	2	0
259	sunsniper	2	0
271	theodore-r	2	0

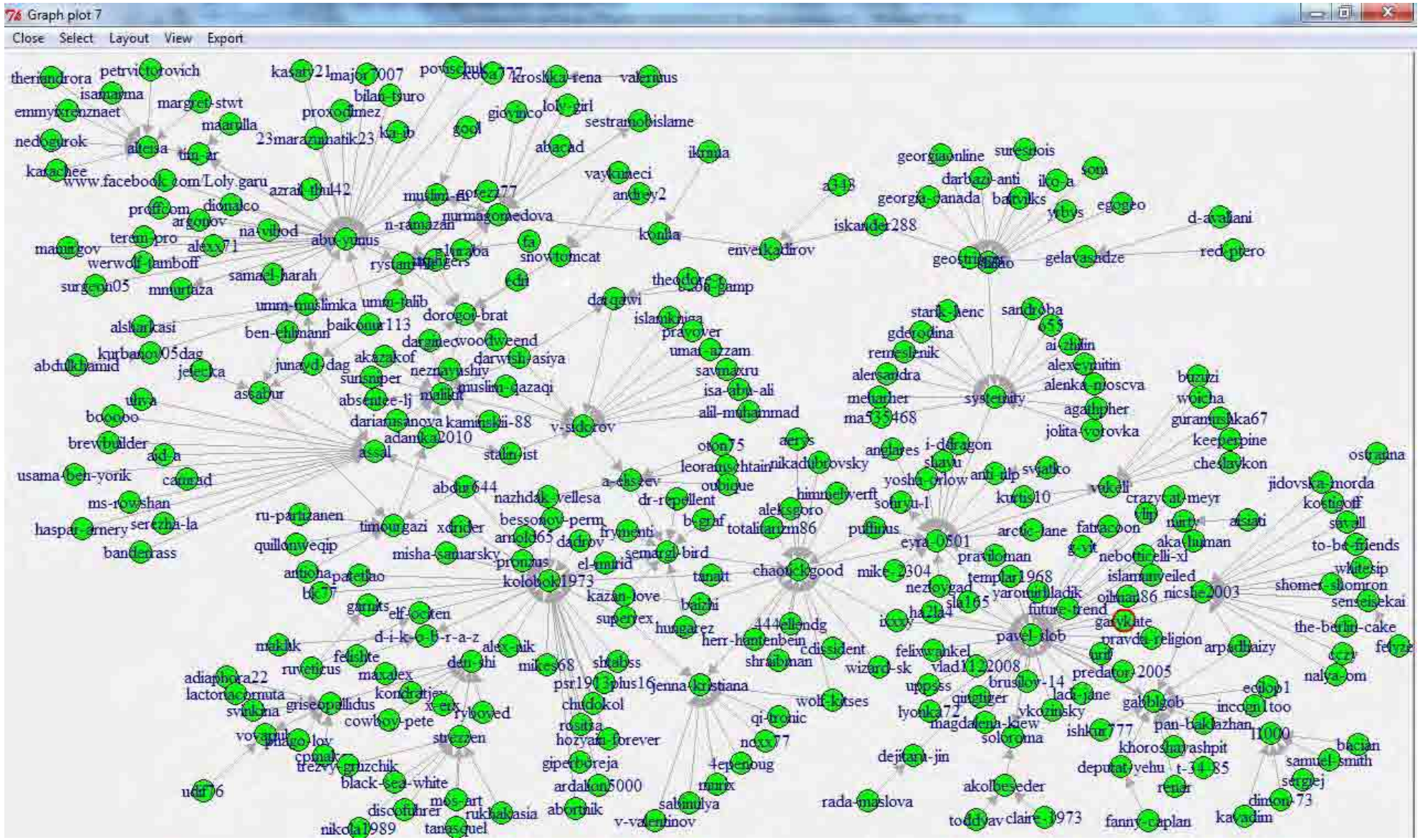
279	umar-azzam	2	0
295	woodweend	2	0
36	aisiati	3	25
37	mirty	3	25
39	aka-human	3	7
40	pavel-slob	3	7
43	soloroma	3	7
89	brusilov-14	3	7
101	crazycat-meyr	3	7
109	dejitaru-jin	3	24
127	fatracoon	3	7
129	felixwankel	3	7
133	future-trend	3	7
135	garykate	3	7
146	g-vit	3	7
147	ha2la4	3	7
162	islamunveiled	3	7
163	ixxxy	3	2
181	ladi-jane	3	7
184	lyonka72	3	7
187	magdalena-kiew	3	7
204	nebotticelli-xl	3	7
214	oilman86	3	7
224	pravda-religion	3	7
227	predator-2005	3	7
233	qingtiger	3	7
236	rada-maslova	3	24
256	sla165	3	7
268	templar1968	3	7
281	uppsss	3	7
282	urif	3	7
286	vkozinsky	3	7
287	vlad1122008	3	7
288	vlip	3	7
299	yaromirhladik	3	7
57	anglares	4	12
58	eyra-0501	4	12
59	anti-nlp	4	21
61	arctic-lane	4	12
123	sohryu-l	4	12
153	i-ddragon	4	12
194	mike-2304	4	12
206	nezloygad	4	12
225	praviloman	4	12
232	puffinus	4	12
252	shavu	4	12
264	svjatko	4	21
300	yosha-orlow	4	12
3	444ellendg	5	2
4	chaotickgood	5	2
5	4epenoug	5	2
6	jenna-kristiana	5	2
32	aerys	5	2
44	aleksgoro	5	2
94	cdissident	5	2
149	herr-hantenbein	5	2
150	himmelwerft	5	2
199	murix	5	2
208	nikadubrovsky	5	2
210	noxx77	5	2
234	qi-tronic	5	2

245	sabinulya	5	2
254	shraibman	5	2
275	totalitarizm86	5	2
289	v-valentinov	5	2
292	wizard-sk	5	2
294	wolf-kitses	5	2
13	abdur644	6	1
15	kolobok1973	6	2
17	abortnik	6	2
31	semargl-bird	6	2
48	alex-nik	6	2
62	ardalion5000	6	2
64	arnold65	6	2
76	baizhi	6	2
81	bessonov-perm	6	2
82	b-graf	6	2
84	bk77	6	2
96	chudokol	6	2
102	dadrov	6	2
111	d-i-k-o-b-r-a-z	6	17
115	dr-repellent	6	2
119	elf-ociten	6	2
120	el-murid	6	2
128	felishte	6	17
132	frymenti	6	2
134	garnits	6	2
141	giperboreja	6	2
151	hozyain-forever	6	2
152	hungarez	6	2
172	kazan-love	6	2
189	makhk	6	17
195	mikes68	6	2
196	misha-samarsky	6	2
220	patetlao	6	2
229	pronzus	6	2
231	psr1913plus16	6	2
240	rositsa	6	2
243	ruveticus	6	2
255	shtabss	6	2
260	superrex	6	2
267	tanatt	6	2
297	xdrider	6	2
7	a343	7	6
8	enverkadirov	7	6
122	konha	7	6
155	ikrima	7	6
28	adiaphora22	8	5
29	griseopallidus	8	5
100	cpmaker	8	5
144	vovapub	8	5
180	lactoriacornuta	8	5
222	phago-lov	8	5
263	svinkina	8	5
277	udif76	8	5
90	buzuzi	9	15
91	vakeli	9	15
95	cheslavkon	9	15
145	guramushka67	9	15
173	keeperpine	9	15
179	kurtis10	9	15
293	woicha	9	15

77	baltvilks	10	9
78	heshilao	10	9
103	darbazi-anti	10	9
118	egogeo	10	9
137	georgia-canada	10	9
138	georgiaonline	10	9
139	geostringer	10	9
154	iko-a	10	9
160	iskander288	10	9
257	som	10	9
261	suresnois	10	9
301	yrbys	10	9
33	agathpher	11	8
34	systemity	11	8
38	ai-zhilin	11	8
45	alenka-moscva	11	8
46	alersandra	11	8
47	alexeymitin	11	8
136	gderodina	11	8
166	jolita-vorovka	11	8
185	ma535468	11	8
193	meharher	11	8
213	o55	11	8
238	remeslenik	11	8
248	sandroba	11	8
258	starik-henc	11	8
107	d-avaliani	12	18
108	gelavasadze	12	18
237	red-ptero	12	18
98	cowboy-pete	13	13
99	den-shi	13	13
176	kondratjev	13	13
192	maxalex	13	13
244	ryboved	13	13
298	x-erx	13	13
85	black-sea-white	14	10
86	strezzen	14	10
114	discofuhrer	14	10
197	mos-art	14	10
209	nikola1989	14	10
241	rukhakasia	14	10
266	tanasquel	14	10
276	trezvy-gruzchik	14	10
14	assal	15	1
35	aid-a	15	1
79	banderrass	15	1
87	booooo	15	1
88	brewbuilder	15	1
92	camrad	15	1
148	haspar-arnery	15	1
198	ms-rowshan	15	1
250	serezha-la	15	1
278	uhva	15	1
283	usama-ben-yorik	15	1
30	a-eliseev	16	16
182	leoramschtain	16	16
203	nazhdak-vellesa	16	16
217	oton75	16	16
218	oubique	16	16
54	alteisa	17	11
121	emmyixrenznaet	17	11

158	isamarma	17	11
169	karachee	17	11
191	margret-stwt	17	11
205	nedogurok	17	11
221	petrvictorovich	17	11
272	theriandrora	17	11
55	andrey2	18	19
56	snowtomcat	18	19
285	vaykuneci	18	19
42	akolbeseder	19	20
97	claire-1973	19	20
274	toddyav	19	20

Сопоставим оба варианта применения алгоритма (с шагом 4 и шагом 2, шаг – длина случайного блуждания) с изображением сети. Можно сказать, что полученная в результате применения алгоритма «ловушки блужданий» структура сообществ достаточно хорошо отражает существующие связи комментирования блогеров. Если этот метод позволит обработать тот объем данных, который мы получим в исследовании всего Живого Журнала, он может быть использован для анализа.


Graph plot 7
Close
Select
Layout
View
Export

Алгоритм «label.propagation.community»

Данный алгоритм, по заявлению его авторов, достаточно прост и быстр в реализации, что располагает к его использованию в работе с большими сетями. Он основан на идее распространения ярлыков (которое напоминает процесс распространения инфекции), и не требует никаких входных данных, кроме списка ребер сети (можно также задаваться веса этих связей) [8].

Процедура «распространения ярлыков» состоит в том, что первоначально каждая вершина сети имеет свой собственный ярлык, затем начинается итеративный процесс и на каждой итерации часть вершин принимает ярлыки своих «соседей», а именно тот ярлык, который более представлен среди соседей данной вершины. Процесс продолжается до тех пор, пока итерации не прекратятся. Когда они прекращаются, можно делать вывод относительно структуры сообществ данной сети [5].

Критерием для остановки итераций является условие, чтобы «каждая вершина имела по меньшей мере столько же соседей внутри своего сообщества, сколько она имеет соседей из других сообществ» [5].

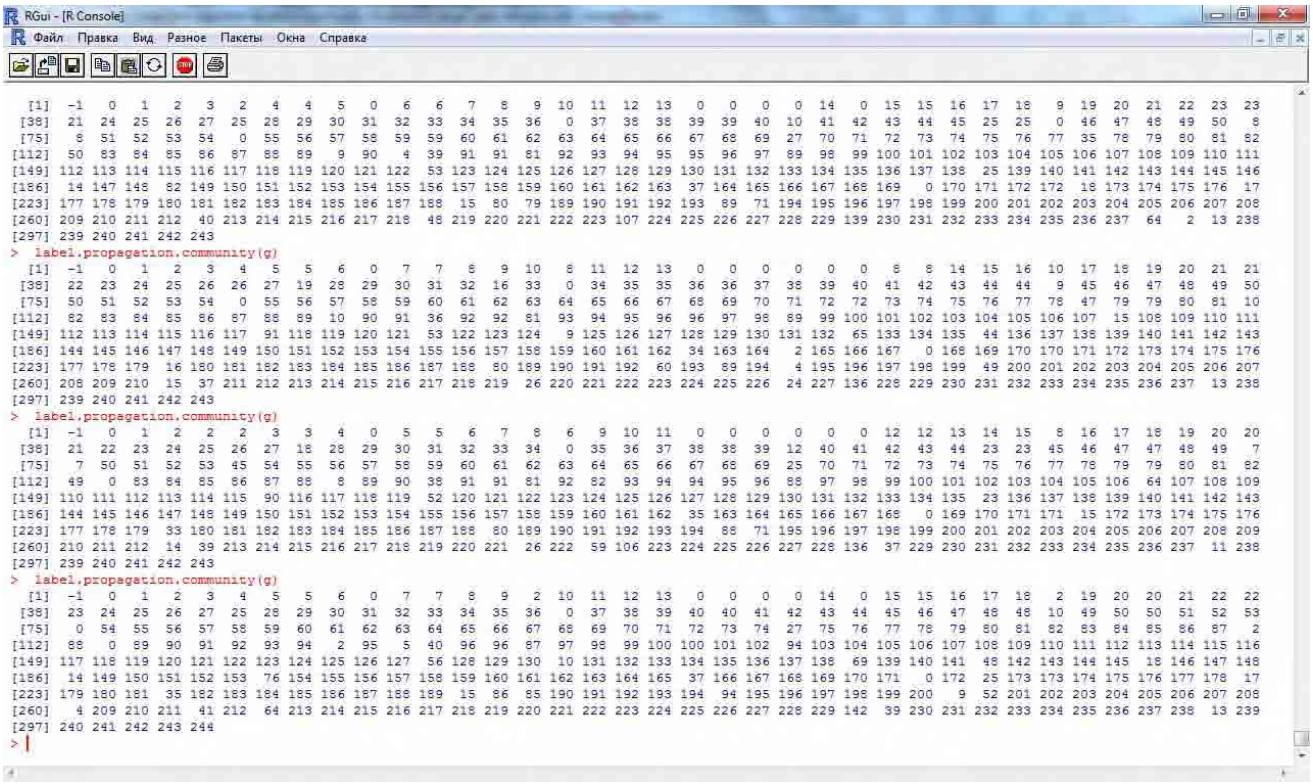
Процедура применения функции «label.propagation.community» допускает первоначальное присвоение всем вершинам сети определенного ярлыка, а также фиксацию ярлыков для некоторых вершин. Однако в нашем исследовании эти функции задействовать нет необходимости, поскольку если для тестовой сети можно выделить несколько «центров притяжения» и зафиксировать их ярлык, то для сети, для которой будут делаться выводы всего исследования, это будет технически невозможно в силу её величины.

Результаты применения этой процедуры – ярлыки, то есть сообщества, к которым в конечном счёте отнесены вершины, могут немного различаться при нескольких попытках запуска этой процедуры. Это связано с тем, что процесс «распространения ярлыков» начинается каждый раз со случайным образом выбранных вершин (при условии, что мы не зафиксировали ни одной вершины). Иногда возможность получить несколько разных, но значимых решений может быть полезна, поскольку реальные сети с содержательной точки зрения могут иметь несколько приемлемых вариантов деления на сообщества (в этом случае можно говорить о пересекающихся сообществах). Конечно, это справедливо только при условии, что деление на сообщества будет эффективным (то есть что их будет относительно немного).

Этот алгоритм не выдает показатель модульности, её нужно рассчитывать отдельно. Однако в нашем случае этого можно не делать: результаты применения этой процедуры в нашем случае показывают, что она не позволила разделить все вершины на сообщества, так как в результате процесса таких сообществ оказалось слишком много. Возможно, результат был бы более приемлемым, если зафиксировать вершины с наибольшими входящими степенями.

Неудовлетворительные результаты для тестовой сети связаны, скорее всего, с тем, что эта сеть – редкая. В статье не сказано о том, как различаются результаты на

тесных и редких сетях, видимо, это и есть слабое место этого алгоритма. Кроме того, этот алгоритм не позволяет задавать количество итераций, тогда как в статье написано, что с увеличением этого количества результат улучшается. Таким образом, можно признать данный алгоритм неподходящим для использования в нашем исследовании.



Алгоритм «edge.betweenness.community»

В первую очередь надо отметить, что данный алгоритм был первым, который базируется на идее модульности [1].

Как пишут сами авторы – М. Ньюман и М. Гирван, особенности их алгоритма заключаются в том, что происходит итеративное исключение ребер с наибольшей промежуточностью, и после каждого исключенного ребра значения промежуточности пересчитываются [3]. Кроме того, они ввели функцию модульности для оценки качества выделения сообществ.

Авторы утверждают, что ребра с наибольшей промежуточностью находятся именно на границах между сообществами, тогда как ребра с малой промежуточностью – внутри сообществ. Промежуточность авторами определяется на основе геодезического (то есть кратчайшего) пути между вершинами, они рассматривают и возможности вычисления промежуточности случайного блуждания (random walk betweenness) и промежуточности тока (current flow betweenness), однако для практических целей рекомендуют первый, наименее требовательный с точки зрения вычислительной мощности вариант, дающий результаты, незначительно отличающиеся от других.

Важным для нашего исследования соображением М. Ньюмана и М. Гирван является то, что применительно к направленным сетям во многих случаях лучше

всего просто игнорировать данную характеристику сети при выделении сообществ, считая направленную связь индикатором взаимодействия, неважно, какого. При этом «подогнать» алгоритм под особенности направленных сетей, как они утверждают, является простейшей задачей.

Что касается модульности, то её определение мы рассмотрели в начале данного текста. Ньюман и Гирван говорят о ней как о «мере качества определённого деления сети» (a measure of the quality of a particular division of a network) [3]. Она измеряет «фракции ребер в сети, которые соединяют вершины того же типа (т.е. ребра внутри сообщества) минус ожидаемый уровень того же количественного показателя в сети с таким же делением на сообщества, но случайно соединёнными вершинами. Если количество внутренних ребер не более чем случайно, то Q=0. Величины, близкие к Q=1, которая является максимумом, отражают сильную структуру сообществ» [3]. На практике, как пишут авторы, величина модульности обычно располагается в интервале между 0,3 и 0,7. Модульность подсчитывается для каждой итерации, т.е. при каждом удалении ребра, и отслеживается момент, когда она достигает наивысшей точки. Эта точка свидетельствует о наилучшем делении сети на сообщества.

Данный алгоритм нацелен на одномодальные, ненаправленные, невзвешенные сети, однако, как говорят авторы, «генерализации к более сложным типам сетей, разумеется, возможны». Ограничением является вычислительная сложность алгоритма – $O(n^3)$ для редких графов, отсюда величина сети должна быть не более 10000 вершин.

Тем не менее, при выполнении этого алгоритма сам показатель модульности по умолчанию не выводится, а также не выводится принадлежность вершин к сообществу.

В качестве результата выводится а) список удалённых связей, б) показатель промежуточности соответствующих удалённых связей, в) слияния (вернее, наоборот, деления компонентов) и г) шаг, на котором произошло отделение данного компонента от графа.

```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

> edge.betweenness.community (g, directed = TRUE, edge.betweenness = TRUE, merges = TRUE, bridges = TRUE, labels = TRUE)
$removed.edges
  [1] 330 274  17 113  11  15  14  13 114  97 239  12 318  16 177 197 238 310 144 178  78 180  54 125  41  79 222 299 221 230 229 275 228 103 137 162 291
 [38]  29 138 186 323  21  53  59 266  52  91  99 112 115 129 135 267 285   5  19  25  44  96  98 120 182 193 233 315 108 126   0   1   2   3   4   6   7
 [75]   8   9  10  18  20  22  23  24  26  27  28  30  31  32  33  34  35  36  37  38  39  40  42  43  45  46  47  48  49  50  51  55  56  57  58  60  61
[112]  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  80  81  82  83  84  85  86  87  88  89  90  92  93  94  95 100 101 102 104 105 106
[149] 107 109 110 111 116 117 118 119 121 122 123 124 127 128 130 131 132 133 134 136 139 140 141 142 143 145 146 147 148 149 150 151 152 153 154 155 156
[186] 157 158 159 160 161 163 164 165 166 167 168 169 170 171 172 173 174 175 176 179 181 183 184 185 187 188 189 190 191 192 194 195 196 198 199 200 201
[223] 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 223 224 225 226 227 231 232 234 235 236 237 240 241 242 243 244 245 246
[260] 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 268 269 270 271 272 273 276 277 278 279 280 281 282 283 284 286 287 288
[297] 289 290 292 293 294 295 296 297 298 300 301 302 303 304 305 306 307 308 309 311 312 313 314 316 317 319 320 321 322 324 325 326 327 328 329 331 332
[334] 333 334 335 336 337 338 339 340 341 342 343 344 345

$edge.betweenness
  [1] 1151.0  420.0  359.5  213.0  126.5  196.0   74.0  111.5   71.0   66.0   62.0   54.0   48.0   45.0   38.0   37.5   36.0   34.0   30.0   29.0   23.0
 [22]   20.0   17.0   16.5   16.0   15.0   14.0   14.0   13.0   12.0   11.0   11.0    9.5    9.0    8.0    8.0    8.0    6.0    6.0    6.0    6.0    5.0
 [43]    5.0    4.0    4.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    3.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0    2.0
 [64]    2.0    2.0    1.5    2.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
 [85]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[106]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[127]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[148]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[169]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[190]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[211]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[232]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[253]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[274]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[295]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[316]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0
[337]    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0    1.0

$merges
     [,1] [,2]
[1,]  300   77
[2,]  299   57
[3,]  298   39
[4,]  297   98
[5,]  296   14
[6,]  295    1
```

Однако для нас важнее всего именно показатель модульности и принадлежность вершин к сообществам, который мы можем запросить с помощью функций modularity и community.to.membership программного пакета igraph. При измерении показателей принадлежности к сообществам и модульности (которая связана с конкретной конфигурацией сообществ), необходимо задать количество итераций, для которого нужно произвести измерения. Мы перебрали несколько вариантов количества итераций, и оказалось, что величина модульности оказывается наибольшей при количестве итераций, равном 260, она составляет в этом случае 0,6498004. Вместе с тем, данное деление дает 41 сообщество, при этом 19 из них состоит из 1 вершины – ими оказываются так называемые «висячие» вершины, которые кроме этого являются комментаторами комментаторов более или менее популярных в масштабе маленькой тестовой сети блогеров. Такое деление имеет смысл в некоторых случаях, однако для «большого» исследования Живого Журнала является не лучшим

вариантом.

```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

[293,]  513  591
[294,]  523  512
[295,]  575  594
[296,]  560  593
[297,]  592  596
[298,]  493  597
[299,]  589  598
[300,]  595  599

$bridges
  [1] 346 345 344 343 342 341 340 339 338 337 336 335 334 333 332 331 330 329 328 327 326 325 324 323 322 321 320 319 318 317 316 315 314 313 312 311 310
 [38] 309 308 307 306 305 304 303 302 301 299 298 297 296 295 294 293 292 291 290 289 288 287 286 285 284 283 282 281 280 279 278 277 276 275 274 273 272
 [75] 271 270 269 268 267 266 265 264 263 262 261 260 259 258 257 256 255 254 253 252 251 250 249 248 247 246 245 244 243 242 241 240 239 238 237 236 235
[112] 234 232 231 230 229 228 227 226 225 224 223 222 221 220 219 218 217 216 215 214 213 212 211 210 209 208 207 206 205 204 203 202 201 200 199 198 197
[149] 196 195 194 193 192 191 190 189 188 187 186 185 184 183 182 181 180 179 178 177 176 175 174 173 172 171 170 169 168 167 166 165 164 163 162 161 160
[186] 159 158 157 156 155 154 153 152 151 150 149 148 147 146 145 144 143 142 141 139 138 137 135 134 133 132 131 130 129 128 127 126 125 124 123 122 121
[223] 120 119 118 117 116 115 113 112 111 109 108 107 106 105 104 103 102 101 100  99  98  97  96  95  94  93  92  91  90  89  88  87  86  85  84  83  82
[260]  81  80  79  78  77  76  75  74  73  72  71  70  69  68  67  65  64  63  62  61  60  59  58  57  55  54  52  51  50  49  48  42  40  38  37  35  28
[297]  27  25  22  10

$labels
  [1] "23marazumatik23"        "abu-yunus"              "444ellendg"             "chaotickgood"           "4epenoug"
  [6] "jenna-kristiana"        "a343"                   "enverkadirov"           "abacad"                 "nurmagomedova"
 [11] "abdulkhamid"            "kurbanov05dag"          "abdur644"               "assal"                  "kolobok1973"
 [16] "timourgazi"             "abortnik"               "absentee-lj"            "malikit"                "alstrangers"
 [21] "mmurtaza"               "muslim-ru"              "rystam4ik"              "tim-ar"                 "umm-muslimka"
 [26] "adamka2010"             "stalin-ist"             "adiaphora22"            "griseopallidus"         "a-eliseev"
 [31] "semargl-bird"           "aerys"                  "agathpher"              "systemity"              "aid-a"
 [36] "aisiati"                "mirty"                  "ai-zhilin"              "aka-human"              "pavel-slob"
 [41] "akazakof"               "akolbeseder"            "soloroma"               "aleksgoro"              "alenka-moscva"
 [46] "alersandra"             "alexeymitin"            "alex-nik"               "alexx71"                "alil-muhammad"
 [51] "v-sidorov"              "alsharkasi"             "dorogoi-brat"           "alteisa"                "andrey2"
 [56] "snowtomcat"             "anglares"               "eyra-0501"              "anti-nlp"               "antioha"
 [61] "arctic-lane"            "ardalion5000"           "argonov"                "arnold65"               "arpadhaizy"
 [66] "gabblgob"               "nicshe2003"             "assabur"                "azrail-thul42"          "baba-gamp"
 [71] "darqawi"                "bacian"                 "11000"                  "baikonur113"            "junayd-dag"
 [76] "baizhi"                 "baltvilks"              "heshilao"               "banderrass"             "ben-ehlmann"
 [81] "bessonov-perm"          "b-graf"                 "bilan-tsuro"            "bk77"                   "black-sea-white"
 [86] "strezzen"               "booooo"                 "brewbuilder"            "brusilov-14"            "buzuzi"
 [91] "yakeli"                 "camrad"                 "cczy"                   "cdissident"             "cheslavkon"
```

```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

> modularity(g, memb$membership)
[1] 0.6259564
> memb <- community.to.membership(g, ebc$merges, steps=260)
> memb
$membership
  [1] 22  1 23  2 24  2 25  7 26  1 27  1 28  1  2  4 29 30  1  1  1  1  1  1  1 31  1 32 11 17  2 33  0  0  1 34  2  0  2  2  1  9  2  2  0  0  0  2  1  1
 [51]  1  1  1 13  3  3  2  2 20  4  2  2  1  2  5  5  5  1  1  1  1  6  6  1  1  2  7  7  1  1  2  2  1  2  8  8  1  1  2  2  2  1  5  2  2  2  9 10 10 11
[101]  2  2  7  1  1  1 12 12 18 14 15  6  1  8  2  5  1  7 19  2 13 16  2 35  1 14  2 15  2 36  5  2  2 19  2  0  7  7  7  1  2 37  1 11  2  2  2  1  2  2
[151]  2  2  2  7 16  5  1 13  5  7  1  2  2  1  5  0  1  1 13  1  6  2  2  5  1 10  5 21  2 11  2 17  1  2  0 38  2  1 15  1 13 10  0  2  2  2  8  1  2  1
[201]  5  1 17  2 13  2  1  2  8  2  1  1  0  2 39  5 17 17  5  2 13 11  1  2  2  1  2  1  2  1  2  2  2  2  4 18 12  0  5  2  8  4 19 10  2  1  6  0  1  1
[251]  6  2  5  2  2  2  7  0  1  2  7  1 11 20  5  8  2  2  1  5  1 13  5  9  2  8 40  1  1  1  2  2  1 21  3  2  2  2  2  1  5  2  2  2  1  1  2 10  2  2
[301]  7

$csize
 [1] 14 75 92  3  4 20  6 13  8  3  6  6  3  8  2  3  2  5  2  3  2  2  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1

> modularity(g, memb$membership)
[1] 0.6498004
> memb <- community.to.membership(g, ebc$merges, steps=270)
> memb
$membership
  [1] 20  1 21  1 22  1  0  0  1  1 23  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  2  2 15  1  1  3  3  1 24  1  3  1  1  1  8  1  1  3  3  3  1  1  1
 [51]  1  1  1 11  4  4  1  1 18  1  1  1  1  1  5  5  5  1  1  1  1  6  6  1  1  1  0  0  1  1  1  1  1  1  7  7  1  1  1  1  1  1  5  1  1  1  8  9  9  2
[101]  1  1  0  1  1  1 10 10 16 12 13  6  1  7  1  5  1  0 17  1 11 14  1 25  1 12  1 13  1 26  5  1  1 17  1  3  0  0  0  1  1 27  1  2  1  1  1  1  1  1
[151]  1  1  1  0 14  5  1 11  5  0  1  1  1  1  5  3  1  1 11  1  6  1  1  5  1  9  5 19  1  2  1 15  1  1  3 28  1  1 13  1 11  9  3  1  1  1  7  1  1  1
[201]  5  1 15  1 11  1  1  1  7  1  1  1  3  1 29  5 15 15  5  1 11  2  1  1  1  1  1  1  1  1  1  1  1  1 16 10  3  5  1  7  1 17  9  1  1  6  3  1  1
[251]  6  1  5  1  1  1  0  3  1  1  0  1  2 18  5  7  1  1  1  5  1 11  5  8  1  7 30  1  1  1  1  1  1  1 19  4  1  1  1  1  1  5  1  1  1  1  1  1  9  1  1
[301]  0

$csize
 [1]  14 177   7  14   3  20   6   8   3   6   3   8   2   3   2   5   2   3   2   2   1   1   1   1   1   1   1   1   1   1   1

> modularity(g, memb$membership)
[1] 0.454793
> memb <- community.to.membership(g, ebc$merges, steps=271)
> memb
$membership
  [1] 20  0 21  0  0  0  1  1  0  0 22  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  2  2 15  0  0  3  3  0 23  0  3  0  0  0  8  0  0  3  3  3  0  0  0
 [51]  0  0  0 11  4  4  0  0 18  0  0  0  0  0  5  5  5  0  0  0  0  6  6  0  0  0  1  1  0  0  0  0  0  0  7  7  0  0  0  0  0  0  5  0  0  0  8  9  9  2
[101]  0  0  1  0  0  0 10 10 16 12 13  6  0  7  0  5  0  1 17  0 11 14  0 24  0 12  0 13  0 25  5  0  0 17  0  3  1  1  1  0  0 26  0  2  0  0  0  0  0  0
```

Данный алгоритм, кроме того, едва ли применим для определения сообществ в «большом» графе, на котором будут строиться выводы исследования, ещё и в силу относительно высоких вычислительных требований. Таким образом, данный алгоритм является не вполне приемлемым для использования в нашем исследовании.

Алгоритм «spinglass.community»

Данный алгоритм использует модель спинового стекла и имитации отжига (spin-glass model and simulated annealing). В основе подхода лежит идея выделения сообществ в сети как «нахождения основного состояния спинового стекла бесконечного диапазона» [6].

Авторы связывают «структурные свойства графа, которые определяют термин «сообщество», с тем что устойчивые структуры сообщества соотносятся с минимумами энергетического ландшафта системы» [6]. Авторы интерпретируют такие термины как «когезия и адгезия в рамках исследования сетей в качестве меры принадлежности вершин к одному и тому же или разным сообществам» [6]. Основное состояние рассматривается ими как «минимальная энергия или максимально модульная конфигурация» сообщества, и с ней они связывают некоторые характеристики, которые определяют сообщество.

Интересной особенностью этого алгоритма является возможность определить сообщество, включающее конкретную вершину, не определяя другие сообщества.

Данный метод предполагает задание количество вращений (spins), которое представляет собой максимальное количество сообществ, которые предполагает получить исследователь. Он может работать с направленными графами, но направление игнорируется при его реализации, может учитывать взвешенные графы.

В данном случае, так же, как и в предыдущем, определение наилучшего варианта реализации алгоритма происходит за счет подбора.

В результатах расчетов показывается сообщество, к которому отнесена вершина, размер сообществ и показатель модульности.

```
RGui - [R Console]
Файл Правка Вид Разное Пакеты Окна Справка

> g <- read.graph("graph9.10.2011.short.ncol", format="ncol", directed=TRUE)
> spingless.community(g, spins=2)
$membership
  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 1 0 1 1 1 1 1 1 0 0 0 0 0 1 1 0 1 1 1 1 1 1 0 0 0 0 0 0 1 1 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 0 0 1 1 1 1 1 0 0
 [76] 0 0 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 1 0 0 0 1 1 1 1 1 0 0 1 1 1 0 0 1 1 0 1 0 1 0 1 1 0 0 0 1 0 0 0 0 1 1 0 1 1 1 1 0 1 0 1 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0
[151] 0 0 0 0 0 1 1 1 1 0 1 1 1 0 1 0 0 1 1 0 1 0 0 1 0 1 1 0 0 1 1 1 0 1 0 1 1 0 0 0 1 1 0 0 0 0 1 0 0 1 1 0 1 1 1 0 1 0 1 0 0 0 0 1 1 1 1 1 1 0 1 1 0 1 0
[226] 1 1 0 0 0 0 0 1 0 0 1 0 0 1 0 1 0 0 1 0 0 1 0 1 0 1 0 1 0 0 1 0 0 1 0 0 0 1 0 1 1 0 1 0 1 1 1 1 1 0 1 1 0 1 1 1 1 1 0 0 1 1 1 1 0 0 1 0 0 1 1 0 0 1 1 0
[301] 0

$csize
[1] 165 136

$modularity
[1] 0.4640471

$temperature
[1] 0.1618977

> spingless.community(g, spins=3)
$membership
  [1] 2 2 0 0 0 0 2 2 2 2 2 2 2 2 0 2 0 0 0 2 2 2 2 2 2 0 0 1 1 0 0 0 1 1 2 1 1 1 1 1 0 1 1 0 1 1 1 0 2 0 0 2 0 2 0 0 1 1 1 2 1 0 2 0 1 1 1 2 2 0 0 1 1 2 2
 [76] 0 2 2 2 2 0 0 2 0 0 0 2 2 1 1 1 2 1 0 1 0 1 0 0 1 1 0 2 0 0 0 2 2 1 1 0 1 2 0 0 1 0 2 0 0 2 2 1 2 2 1 1 0 1 1 1 0 1 0 1 1 2 2 2 2 0 2 2 1 1 1 1 2 0 0
[151] 0 0 1 2 2 1 0 2 1 2 0 1 1 2 1 1 2 0 2 2 1 0 1 1 2 0 1 2 1 1 1 0 2 1 1 2 1 2 0 2 2 0 1 1 0 0 0 2 0 0 1 2 0 1 2 1 0 0 0 0 2 2 1 1 1 1 0 0 1 0 2 1 2 1 1
[226] 0 1 2 0 2 0 0 1 0 2 1 2 1 1 0 0 2 0 0 0 2 1 1 0 2 1 1 1 0 0 1 2 1 0 0 2 2 1 1 1 0 0 1 2 1 0 2 1 0 0 0 1 2 0 2 1 1 2 2 0 1 1 1 0 2 1 0 1 0 0 2 0 0 1 1
[301] 2

$csize
[1] 103 106  92

$modularity
[1] 0.6219759

$temperature
[1] 0.1487163

> spingless.community(g, spins=4)
$membership
  [1] 3 3 1 1 1 1 3 3 3 3 3 3 2 2 1 2 1 2 2 3 3 3 3 3 3 2 2 3 3 2 1 1 1 1 2 0 0 1 0 0 2 0 0 1 1 1 1 1 3 2 2 3 2 3 2 2 1 1 1 2 1 1 3 1 0 0 0 2 3 2 2 0 0 2 2
 [76] 1 1 1 2 3 1 1 3 1 3 3 2 2 0 1 1 2 0 1 1 1 0 3 3 3 0 1 1 2 2 2 1 1 0 0 1 0 3 3 1 0 2 1 1 1 3 3 1 3 3 0 0 1 0 0 0 1 0 1 0 1 1 1 1 3 1 3 3 3 1 0 0 2 1 1
[151] 1 1 1 1 3 0 2 3 0 1 2 0 0 2 0 1 3 2 3 3 0 1 1 0 3 3 0 3 1 3 0 2 3 0 1 3 0 3 1 3 3 3 1 1 1 1 3 2 1 2 0 3 2 0 3 1 2 1 3 1 3 3 1 0 0 0 2 2 0 1 3 3 3 0 1
```

```
RGui - [R Console]
Файл Правка Вид Разное Пакеты Окна Справка

[1] 0.7796287

$temperature
[1] 0.1117703

> spingless.community(g, spins=10)
$membership
  [1] 4 4 6 6 6 6 2 2 4 4 4 4 3 3 6 3 6 0 0 4 4 4 4 9 4 0 0 1 1 0 6 6 2 2 3 5 5 2 5 5 0 5 5 6 2 2 2 6 4 0 0 4 0 9 0 0 7 7 7 3 7 6 4 6 8 8 8 3 4 0 0 8 8 3 3
 [76] 6 2 2 3 4 6 6 4 6 1 1 3 3 5 7 7 3 8 6 7 6 5 1 1 1 5 6 2 0 0 0 2 2 5 8 6 8 4 1 6 8 0 2 6 6 9 2 7 4 4 8 5 6 5 8 8 6 5 6 5 2 2 2 2 4 6 4 4 1 7 5 5 3 6 6
[151] 6 6 7 2 2 8 0 9 8 2 0 5 5 3 8 2 4 0 9 4 8 6 7 8 4 1 8 4 7 1 5 0 4 5 2 9 5 4 6 4 9 1 2 7 6 6 1 3 6 0 8 4 0 5 9 7 0 6 1 6 4 4 2 5 8 8 0 0 8 6 9 1 4 5 7
[226] 0 5 4 6 4 6 7 5 6 3 5 2 2 8 6 1 3 6 1 6 4 8 2 0 3 8 7 8 6 6 5 2 2 0 6 2 4 1 7 8 1 6 5 4 8 0 9 8 5 6 1 1 3 0 4 5 5 3 4 0 5 5 5 6 4 8 6 7 6 0 4 6 1 5 7
[301] 2

$csize
 [1] 33 22 33 20 43 36 54 20 30 10

$modularity
[1] 0.7937142

$temperature
[1] 0.1223105

> spingless.community(g, spins=11)
$membership
  [1]  2  2  3  3  3  3 10 10  2  2  2  2  8  8  9  8  9  0  0  2  2  2  2  6  2  0  0  5  5  9  9  3  1  1  8  4  4  1  4  4  0  4  4  3  1  1  1  9  2  0
 [51]  0  2  0  6  0  0  3  3  3  8  3  9  2  9  7  7  7  8  2  0  0  7  7  8  8  3 10 10  8  2  9  9  2  9  5  5  8  8  4  3  3  8  7  3  3  9  4  5  5  5
[101]  4  9 10  0  0  0 10 10  6  7  6  7  2  5  9  7  0 10  9  9  6 10  3  2  2  7  4  6  4  7  7  9  4  9  4  1 10 10 10  2  9  2  2  5  3  4  4  8  3  3
[151]  9  9  3 10 10  7  0  6  7 10  0  4  4  8  7  1  2  0  6  2  7  9  3  7  2  5  7  2  3  5  4  9  2  4  1  6  4  2  6  2  6  5  1  3  9  9  5  8  3  0
[201]  7  2  9  4  6  3  0  3  5  3  2  2  1  4  7  7  9  9  7  9  6  5  2  4  3  0  4  2  9  2  9  3  4  3  8  6 10  1  7  9  5  8  9  5  3  2  7  1  0  8
[251]  7  3  7  3  9  4 10  1  0  9 10  2  5  3  7  5  3  4  2  7  0  6  7  4  3  5  5  8  0  2  4  4  8  2  0  4  4  4  3  2  7  3  3  3  0  2  9  5  4  3
[301] 10

$csize
 [1] 28 14 43 41 34 22 15 30 20 35 19

$modularity
[1] 0.7920138

$temperature
[1] 0.1140397
```

Данный алгоритм зависит от показателя spins, и для больших графов не подходит. Уже на графе в 301 вершину расчеты для 10-ти групп требуют несколько секунд. При попытке провести расчеты для 1300 вершин и 25 групп на домашнем компьютере работу программы пришлось прервать через несколько часов.

Алгоритм «fastgreedy.community»

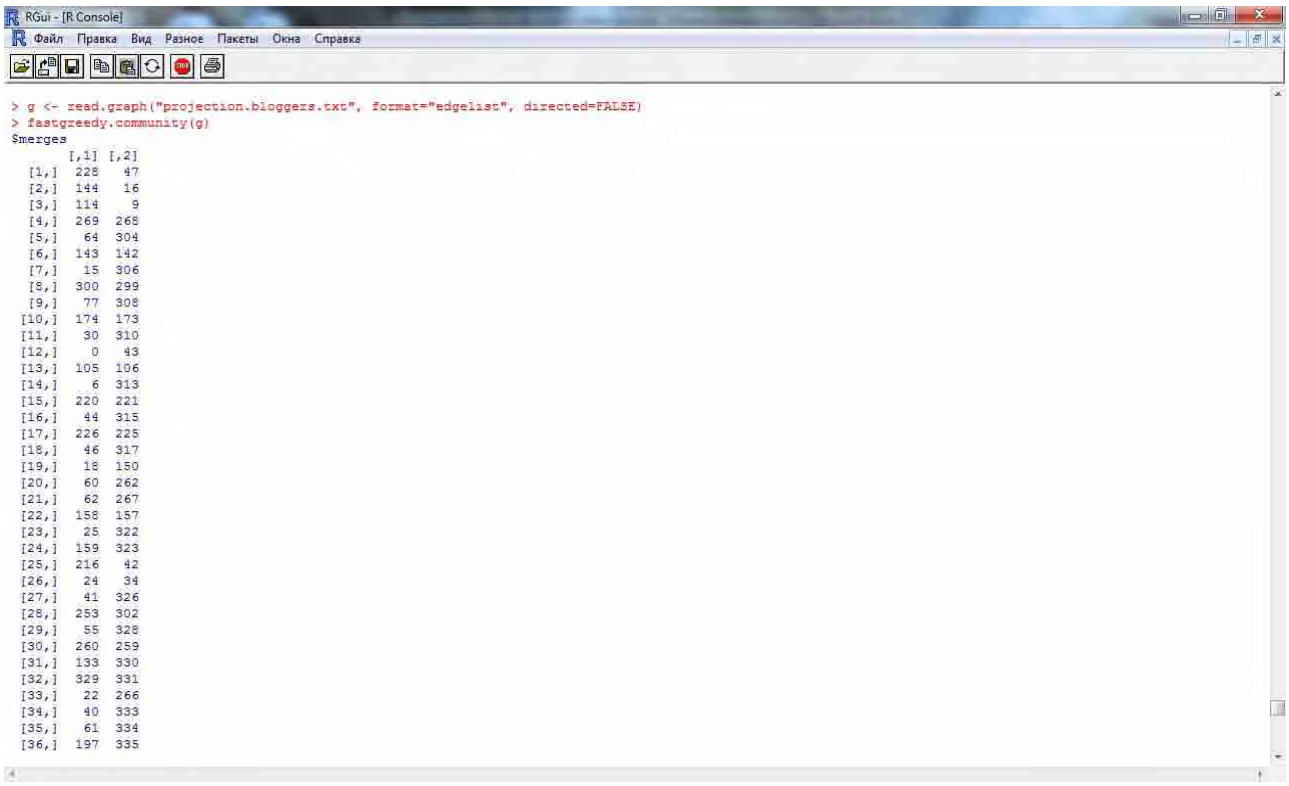
Данный алгоритм предназначается прежде всего для анализа больших сетей, он позволяет быстро производить расчеты с сетями до сотен тысяч вершин и миллионов ребер [7]. Скорость расчетов для редких сетей составляет $O(n \log^2 n)$. При этом имеет такие ограничения, как жесткие требования к формату загружаемых файлов, также граф должен быть ненаправленным. Это связано с тем, что «неподходящие» форматы требуют довольно много памяти компьютера и поэтому снижают производительность при расчетах. Приведение файла в формат «edgelist», то есть пары чисел, означающие связанные вершины, значительно сокращает затраты времени и оперативной памяти.

В основе алгоритма лежит иерархическая агломеративная кластеризация редких иерархически организованных сетей. Как и остальные описанные здесь алгоритмы выделения сообществ, он основан на показатели модульности.

Результаты расчетов представляют собой, во-первых, список слияния субграфов, во-вторых – значения модульности. Максимальное значение модульности отмечает то слияние, на котором возникает оптимальное число и конфигурация сообществ.

Поскольку данный алгоритм не может работать с файлами формата ncol, а тестовая сеть, на которой испытывались остальные алгоритмы, содержится в файле такого формата, для опробования этого алгоритма использовалась другая сеть, содержащаяся в файле формата edgelist, и представляющая собой проекцию «блогеры» двумодальной сети «блогеры-посты», первоначально полученной на этапе формирования массива данных.

Для неё максимальное значение модульности составило 0,7439494 (7.439494e-01), оптимальная структура сообществ для данной сети включает девять сообществ.



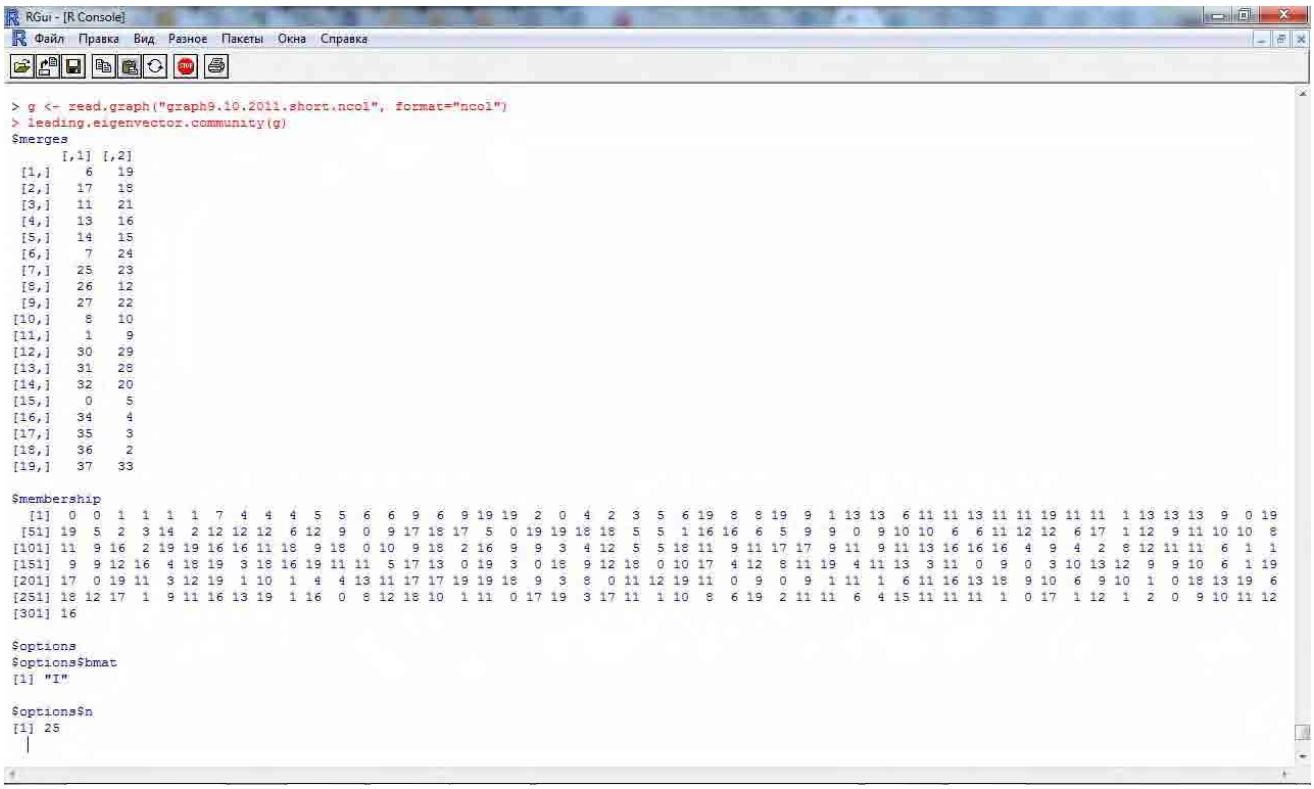
```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

[296,]  589  539
[297,]  593  579
[298,]  595  594
[299,]  596  597
[300,]  598  599

$modularity
  [1] -5.600326e-03 -4.839870e-03 -4.079704e-03 -3.319537e-03 -2.559661e-03 -1.046858e-03 -2.869809e-04  1.226981e-03  1.986857e-03  3.490394e-03
 [11]  4.250270e-03  5.760757e-03  6.520634e-03  7.280510e-03  8.797947e-03  9.557824e-03  1.106368e-02  1.182355e-02  1.332477e-02  1.408436e-02
 [21]  1.484395e-02  1.560353e-02  1.636283e-02  1.785450e-02  1.936151e-02  2.012051e-02  2.087923e-02  2.238914e-02  2.314786e-02  2.460738e-02
 [31]  2.536581e-02  2.685313e-02  2.882000e-02  2.957843e-02  3.104490e-02  3.289884e-02  3.401954e-02  3.580745e-02  3.826257e-02  4.138490e-02
 [41]  4.517444e-02  4.963118e-02  5.475514e-02  6.054630e-02  6.694067e-02  7.399154e-02  8.124889e-02  8.883954e-02  9.687385e-02  1.052281e-01
 [51]  1.138497e-01  1.229170e-01  1.319226e-01  1.331785e-01  1.351466e-01  1.378267e-01  1.412189e-01  1.453232e-01  1.501396e-01  1.556681e-01
 [61]  1.619087e-01  1.688614e-01  1.761834e-01  1.773539e-01  1.785232e-01  1.795145e-01  1.812642e-01  1.822502e-01  1.830970e-01  1.838554e-01
 [71]  1.853497e-01  1.873907e-01  1.888432e-01  1.909040e-01  1.935519e-01  1.954539e-01  1.979351e-01  2.008646e-01  2.042788e-01  2.084306e-01
 [81]  2.133200e-01  2.189470e-01  2.253115e-01  2.273123e-01  2.300668e-01  2.331003e-01  2.347521e-01  2.371360e-01  2.402519e-01  2.441000e-01
 [91]  2.486801e-01  2.539922e-01  2.600365e-01  2.668129e-01  2.679897e-01  2.699230e-01  2.721696e-01  2.739966e-01  2.765774e-01  2.797238e-01
[101]  2.809574e-01  2.822704e-01  2.832547e-01  2.849158e-01  2.873275e-01  2.902755e-01  2.938565e-01  2.954672e-01  2.971813e-01  2.984728e-01
[111]  2.997632e-01  3.005868e-01  3.016884e-01  3.034010e-01  3.058396e-01  3.088768e-01  3.103746e-01  3.123577e-01  3.145435e-01  3.160644e-01
[121]  3.173339e-01  3.187361e-01  3.208889e-01  3.233414e-01  3.248496e-01  3.260543e-01  3.272225e-01  3.288604e-01  3.308785e-01  3.320693e-01
[131]  3.332589e-01  3.340173e-01  3.355290e-01  3.377322e-01  3.392137e-01  3.414516e-01  3.444460e-01  3.452044e-01  3.467212e-01  3.489574e-01
[141]  3.497159e-01  3.512032e-01  3.529743e-01  3.544419e-01  3.557416e-01  3.577838e-01  3.605684e-01  3.640956e-01  3.682170e-01  3.727739e-01
[151]  3.766981e-01  3.782057e-01  3.801717e-01  3.828915e-01  3.863651e-01  3.882417e-01  3.908688e-01  3.942465e-01  3.967694e-01  3.979672e-01
[161]  3.999213e-01  4.026318e-01  4.036465e-01  4.054119e-01  4.077517e-01  4.106534e-01  4.134630e-01  4.153418e-01  4.177726e-01  4.190867e-01
[171]  4.201907e-01  4.218396e-01  4.242391e-01  4.249969e-01  4.265016e-01  4.287532e-01  4.317516e-01  4.354968e-01  4.398695e-01  4.406265e-01
[181]  4.421147e-01  4.435986e-01  4.458330e-01  4.488181e-01  4.525538e-01  4.570400e-01  4.577964e-01  4.593092e-01  4.615680e-01  4.623244e-01
[191]  4.638372e-01  4.660543e-01  4.668104e-01  4.675662e-01  4.683218e-01  4.690773e-01  4.698325e-01  4.705875e-01  4.713413e-01  4.728489e-01
[201]  4.751103e-01  4.780733e-01  4.816717e-01  4.830664e-01  4.852035e-01  4.880832e-01  4.917054e-01  4.960700e-01  5.011772e-01  5.068479e-01
[211]  5.076017e-01  5.091093e-01  5.113663e-01  5.142888e-01  5.150426e-01  5.165502e-01  5.188116e-01  5.195645e-01  5.203151e-01  5.218163e-01
[221]  5.240682e-01  5.270706e-01  5.307802e-01  5.352158e-01  5.366475e-01  5.388299e-01  5.417628e-01  5.453450e-01  5.460956e-01  5.475968e-01
[231]  5.498487e-01  5.528025e-01  5.560505e-01  5.567973e-01  5.582910e-01  5.605315e-01  5.635189e-01  5.672531e-01  5.717342e-01  5.757986e-01
[241]  5.765361e-01  5.780113e-01  5.802240e-01  5.831743e-01  5.868622e-01  5.912877e-01  5.964508e-01  6.020803e-01  6.057448e-01  6.064824e-01
[251]  6.079575e-01  6.101702e-01  6.131206e-01  6.168084e-01  6.209994e-01  6.257845e-01  6.311048e-01  6.327577e-01  6.350881e-01  6.380957e-01
[261]  6.417807e-01  6.461430e-01  6.511827e-01  6.568998e-01  6.632941e-01  6.703659e-01  6.781149e-01  6.863687e-01  6.871034e-01  6.885502e-01
[271]  6.907091e-01  6.935534e-01  6.970759e-01  7.007279e-01  7.049182e-01  7.098346e-01  7.154769e-01  7.218452e-01  7.289395e-01  7.367598e-01
[281]  7.373992e-01  7.380383e-01  7.386771e-01  7.393160e-01  7.399545e-01  7.405487e-01  7.411427e-01  7.417363e-01  7.423155e-01  7.428944e-01
[291]  7.434342e-01  7.438199e-01  7.439494e-01  7.434605e-01  7.385842e-01  7.205530e-01  6.999425e-01  6.717008e-01  6.054422e-01  4.365827e-01
[301]  5.551115e-17

>
```

Таким образом, данный алгоритм является приемлемым для применения на полной сети комментирования постов об исламе в Живом Журнале.

Алгоритм «leading.eigenvector.community»

Статья, в которой обоснован этот алгоритм, посвящена ещё одному взгляду на модульность – основной показатель, позволяющий выделять сообщества в сети. В данном случае модульность выражается в виде собственных значений и собственных векторов того, что авторы называют матрицей модульности, которая является «характеристикой сети и сама по себе не зависима ни от каких разделений сети на сообщества» [3].

Практический смысл такого подхода состоит в создании новых алгоритмов выделения сообществ, а также возможности заметить первоначально неочевидную дву- или более модальную структуру сети. Представление модульности в матричном виде позволяет рассматривать «задачу оптимизации как спектральную проблему в линейной алгебре. Этот подход ведет к семье новых компьютерных алгоритмов для выделения сообществ, которые дают результаты, сопоставимые с лучшими прежними методами». Кроме того, данный подход позволяет лучше понять устройство сети как структуры сообществ.



```
RGui - [R Console]
Файл  Правка  Вид  Разное  Пакеты  Окна  Справка

> g <- read.graph("graph9.10.2011.short.ncol", format="ncol")
> leading.eigenvector.community(g)
$merges
      [,1] [,2]
 [1,]    6   19
 [2,]   17   18
 [3,]   11   21
 [4,]   13   16
 [5,]   14   15
 [6,]    7   24
 [7,]   25   23
 [8,]   26   12
 [9,]   27   22
[10,]    8   10
[11,]    1    9
[12,]   30   29
[13,]   31   28
[14,]   32   20
[15,]    0    5
[16,]   34    4
[17,]   35    3
[18,]   36    2
[19,]   37   33

$membership
  [1]  0  0  1  1  1  1  7  4  4  4  5  5  6  6  9  6  9 19 19  2  0  4  2  3  5  6 19  8  8 19  9  1 13 13  6 11 11 13 11 11 19 11 11  1 13 13 13  9  0 19
 [51] 19  5  2  3 14  2 12 12 12  6 12  9  0  9 17 18 17  5  0 19 19 18 18  5  5  1 16 16  6  5  9  9  0  9 10 10  6  6 11 12 12  6 17  1 12  9 11 10 10  8
[101] 11  9 16  2 19 19 16 16 11 18  9 18  0 10  9 18  2 16  9  9  3  4 12  5  5 18 11  9 11 17 17  9 11  9 11 13 16 16 16  4  9  4  2  8 12 11 11  6  1  1
[151]  9  9 12 16  4 18 19  3 18 16 19 11 11  5 17 13  0 19  3  0 18  9 12 18  0 10 17  4 12  8 11 19  4 11 13  3 11  0  9  0  3 10 13 12  9  9 10  6  1 19
[201] 17  0 19 11  3 12 19  1 10  1  4  4 13 11 17 17 19 19 18  9  3  8  0 11 12 19 11  0  9  0  9  1 11  1  6 11 16 13 18  9 10  6  9 10  1  0 18 13 19  6
[251] 18 12 17  1  9 11 16 13 19  1 16  0  8 12 18 10  1 11  0 17 19  3 17 11  1 10  8  6 19  2 11 11  6  4 15 11 11 11  1  0 17  1 12  1  2  0  9 10 11 12
[301] 16

$options
$options$bmat
[1] "I"

$options$n
[1] 25
```

Для реализации данного метода «рассчитывается собственный вектор матрицы модульности для самого большого позитивного собственного значения и потом разделения вершин на два сообщества, основанных на знаках соответствующих элементов в собственном векторе. Если все элементы в собственном векторе одного знака, то это означает, что сеть не имеет скрытой структуры сообществ» [8].

Входной граф должен быть ненаправленным, поскольку данный метод требует симметричную матрицу.

В качестве результатов программа выдает матрицу слияний и список сообществ, к которым принадлежат в итоге каждая из вершин.

Недостатком данного алгоритма является малая вычислительная скорость, которая не позволит работать с большими объемами данных.

Заключение

В заключение приведём таблицу, в которой обобщены результаты нашего анализа алгоритмов выделения сообществ в отношении их применимости в исследовании большой сети комментирования постов об исламе в Живом Журнале (табл. 2).

Наиболее релевантный для нашего исследования алгоритм выделения сообществ - fastgreedy.community, также может оказаться применимым алгоритм walktrap.community. В первом случае будет необходима более тщательная подготовка данных для анализа, поскольку данный алгоритм более требователен, чем остальные, к структуре входных данных. Вместе с тем, он специально

разработан для анализа больших массивов данных, и в этом смысле является наиболее перспективным.

Алгоритм walktrap.community может также быть использован в том случае, если размер итоговой сети не превысит несколько сотен тысяч вершин. Данный алгоритм менее требователен к структуре входных данных и показал свою эффективность в применении к реальным данным. Результат разделения сети на сообщества, полученный с его помощью, соответствует визуальному представлению сети с использованием алгоритма Фруктермана-Рейнголда.

Итак, мы рассмотрели процесс сбора первичных данных для сетевого анализа блогов об исламе, выявили содержательные проблемы, связанные с работой с бимодальными сетями, а также проанализировали возможности алгоритмов выявления социальных сетей и получили содержательные результаты по разделению вершин сети на сообщества.

Таблица 2

Алгоритмы выявления сообществ, представленные в программном пакете igraph (на основе R)

Функция	Вычислительная скорость/ может ли работать с большими сетями	Работа с направленными графами	Работа с взвешенными графами	Ограничения по формату входного графа	Выдает показатель модульности *	Выдает список принадлежности вершин к сообществам	Источник	Примечания
walktrap.community	O(n2log n)	Работает, но направление связи игнорирует	да	нет	да	да	P. Pons, M. Latapy. Computing communities in large networks using random walks.	Результаты анализа достаточно хорошо отражают визуальную репрезентацию сети
label.propagation.community	O(n)	нет	да	нет	нет	да	U.N. Raghavan, R. Albert, S. Kumara. Near linear time algorithm to detect community structures in large-scale networks.	Качество выделения сообществ на тестовых графах из нескольких сотен вершин неудовлетворительно (очень много вершин отнесены к отдельным сообществам), возможно, алгоритм не пригоден для анализа редких сетей.
spinglass.community	С большими сетями скорее нет, скорость зависит от заданного количества сообществ, на которые нужно разделить исходный граф	Работает, но направление связи игнорирует	да	нет	да	да	J. Reichardt and S. Bornholdt. Statistical mechanics of community Detection.	Для больших сетей и большого количества сообществ высокие требования к вычислительной мощности. На домашнем ПК сеть из 301 вершины и 1314 связей при заданных 25 возможных сообществах обработать не получилось – компьютер «завис».

leading.eigenvector.community	O(n2m)	нет	да, но веса не учитывает	нет	нет	да	M.E.J. Newman. Finding community structure using the eigenvectors of matrices.	Скорость расчетов не позволит работать с большими графами
fastgreedy.community	O(nlog2n)	нет	да	только простые форматы (simple formats)	да	нет	A. Clauset, M.E.J. Newman, C. Moore. Finding community structure in very large networks.	Вычислительные возможности алгоритма соответствуют нашим задачам, однако следует выяснить, можно ли идентифицировать вершины, которые в данном алгоритме обозначены только номерами, после выделения сообществ
edge.betweenness.community	O(nm2)	нет	нет	нет	нет	нет	M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks.	Алгоритм хорошо структурирует граф, однако слишком много вершин оказывается отдельными сообществами. Кроме того, скорость расчетов не позволит работать с большими графами.

*Модульность можно рассчитать и отдельно с помощью функции modularity, если известна принадлежность каждой вершины к сообществу

Библиография

1. S. Fortunato. Community detection in graphs. Physics Reports (2010) Volume: 486, Issue: 3-5.

2. M.E.J. Newman. Finding community structure using the eigenvectors of matrices // Physical Review E 74, 036104 (2006).

3. M.E.J. Newman and M. Girvan. Finding and evaluating community structure in networks // Physical Review E 69, 026113 (2004).

4. P. Pons, M. Latapy. Computing communities in large networks using random walks // Journal of Graph Algorithms and Applications (JGAA) vol. 10, no. 2, pages 191-218, 2006.

5. U.N. Raghavan, R. Albert, S. Kumara. Near linear time algorithm to detect community structures in large-scale networks // Physical Review E - Statistical, Nonlinear and Soft Matter Physics (2007) Volume: 76, Issue: 3 Pt 2, Publisher: APS, Pages: 036106.

6. J. Reichardt and S. Bornholdt. Statistical mechanics of community Detection // Physical Review E 74, 016110 (2006).

7. A. Clauset, M.E.J. Newman, C. Moore. Finding community structure in very large networks // Physical Review E 70, 066111 (2004).

8. Documentation for package ‘igraph’ version 0.5.5-2. September 20, 2011 // http://igraph.sourceforge.net/doc/R/igraph.pdf.