Алгоритм выделения сообществ в социальных сетях

М. И. КОЛОМЕЙЧЕНКО, А. А. ЧЕПОВСКИЙ
Национальный исследовательский университет «Высшая школа экономики»

А. М. ЧЕПОВСКИЙ
Национальный исследовательский университет «Высшая школа экономики»
e-mail: achepovskiy@hse.ru

УДК 004.421.2:519.178

Ключевые слова: социальная сеть, большие данные, алгоритмы теории графов.

Аннотация

В статье предлагается алгоритм нахождения подграфов с заданными свойствами больших графов, описывающих социальные сети. Описывается вычислительный эксперимент, подтверждающий эффективность предлагаемого алгоритма.

Abstract

M. I. Kolomeychenko, A. A. Chepovskiy, A. M. Chepovskiy, An algorithm for detecting communities in social networks, Fundamentalnaya i prikladnaya matematika, vol. 19 (2014), no. 1, pp. 21—32.

In this paper, we propose an algorithm to find subgraphs with given properties in large social networks. A computational experiment that confirms the effectiveness of the proposed algorithm is presented.

1. Проблема выделения сообществ как задача анализа графа

Распознавание структуры, скрытой в реальных социальных сетях, является ключевой задачей, решение которой необходимо для понимания организации сложных сетей [1, 2]. Сложные системы обычно организуются в блоки, которые имеют определённые роли и функции. В представлении сети такие блоки являются множеством вершин с высокой плотностью внутренних связей, в то время как связи между блоками имеют сравнительно низкую плотность. Данные блоки называются сообществами или модулями и встречаются в разнообразном множестве структур взаимосвязанных объектов [5, 6].

Фундаментальная и прикладная математика, 2014, том 19, № 1, с. 21—32.
© 2014 *Издательский дом «Открытые системы»*

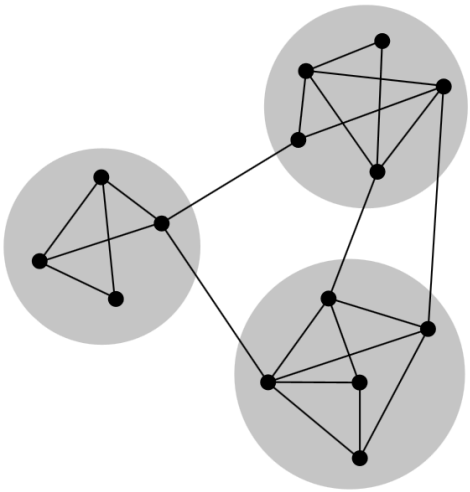

Рис. 1. Пример сети, разбитой на сообщества

Правильно подобранное представление поможет выявить важную информацию о сети, упростит восприятие и подчеркнёт скрытые структурные особенности и особенности взаимосвязей объектов. На рис. 1 приведён пример сети, разбитой на сообщества с высокой плотностью связей внутри сообществ и низкой плотностью связей между сообществами.

Выделение модулей (сообществ) является актуальной задачей теории графов. Мы рассматриваем задачу нахождения по заданному мультиграфу разбиения вершин на сообщества, при котором минимизируется заданная целевая функция.

Было разработано множество алгоритмов с использованием методов из различных дисциплин, таких, как физика, биология, информатика, прикладная математика и социология [1, 2, 5, 6]. Однако не все алгоритмы надёжны и могут быть применены на практике.

Приведём несколько алгоритмов, предложенных ранее для выделения сообществ.

В алгоритме иерархического разбиения М. Гирвана и М. Ньюмана [6, 17] связи удаляются итеративно в соответствии со значением одной из метрик для связей или вершин [1, 16]. В самых распространённых реализациях используется метрика связи, или «центральность по посредничеству» [1], определяемая как количество кратчайших путей между всеми парами вершин, проходящих через заданную связь:

$$\mathrm{betw}(l) = \sum_{s \neq t,\ s \in V,\ t \in V} \frac{\sigma_{st}(l)}{\sigma_{st}}, \tag{1}$$

где l — связь, для которой считается метрика, V — множество вершин графа, σ_{st} — количество кратчайших путей между вершинами s и t, $\sigma_{st}(l)$ — количество кратчайших путей между вершинами s и t, проходящих через связь l.

В самой распространённой реализации процедура удаления связей завершается, когда достигает максимума модулярность результирующего разбиения [15]:

$$Q = \sum_{i,j} \left[\frac{A_{ij}}{2m} - \frac{k_i k_j}{4m^2} \right] \sigma(c_i, c_j), \tag{2}$$

где m — количество связей, A — матрица смежности графа, k_i — степень вершины i,

$$\sigma(c_i, c_j) = \begin{cases} 1, & c_i = c_j, \\ 0, & c_i \neq c_j, \end{cases}$$

c_i — номер класса, к которому принадлежит вершина i. Модулярность Ньюмана—Гирвана (2) определяет качество разбиения сети на сообщества. Используется быстрый жадный алгоритм оптимизации модулярности [4]. Этот метод является ускоренной реализацией предыдущей техники, предложенной М. Ньюманом [14]. Начиная с множества изолированных вершин связи оригинального графа итеративно добавляются так, чтобы максимально возможным образом увеличить метрику модулярности, описанную выше.

Быстрый метод оптимизации модулярности [3] предусматривает многоступенчатую процедуру, основанную на локальной оптимизации модулярности Ньюмана—Гирвана для соседей каждой вершины. После разбиения графа сообщества заменяются на супервершины, в результате чего получаются сеть с меньшими степенями вершин. Процедура итеративно продолжается до тех пор, пока модулярность нельзя увеличить.

Помимо указанных выше алгоритмов, существует ещё несколько популярных подходов для выделения сообществ в сетевых структурах [3, 7, 13, 18]. Однако приведённые в [10] результаты тестирования показывают, что большинство разработанных алгоритмов неприменимы для реальных сетей больших размеров либо по причине низких скоростных характеристик, либо из-за низкого качества выделения сообществ.

В данной статье предлагается метод, с помощью которого можно эффективно выделять сообщества в сети. Он основан на модификации алгоритма для нахождения сообществ. Данная модификация использует метрику модулярности графа, предложенную в [8, 19—21].

2. Методика выделения сообществ

Предлагаемая методика предусматривает проведение предобработки данных, которая заключается в получении графа на основе данного мультиграфа. Кратные связи в мультиграфе заменяются на одну с весом, равным количеству кратных связей между парой вершин.

Сопоставим задачу выделения значимых структур из данных с задачей сжатия данных.

Рассмотрим задачу получения уникальных имён для вершин. Простой метод получения уникальных имён для вершин — использование кода Хаффмана, который обеспечивает сжатие информации путём назначения самых коротких кодовых слов самым частым событиям или объектам и длинных кодовых слов менее частым. Затем для создания общего представления сети необходимо отделить важную структурную информацию от несущественных деталей. В основе метода лежит двухуровневое разбиение информации. Метод использует уникальные имена для более общих объектов (сообществ) и повторяющиеся имена для более мелких деталей (обозначения вершин в каждом модуле).

Использование двухуровневого описания позволяет закодировать случайное блуждание с использованием меньшего количества информации, чем при одноуровневой схеме кодирования. Кроме того, данный подход поможет извлечь структуру сети на основе предположения [12], что, в общем случае, случайное блуждание будет проводить статистически больше времени внутри групп вершин, чем при перемещении от группы к группе.

Таким образом, разбиение на сообщества будет описываться верхним уровнем двухуровневого кодирования. Основная идея заключается в том, что нет необходимости непосредственно выполнять двухуровневое кодирование. Достаточно лишь считать верхнюю границу длины кодового слова, необходимого для кодирования вершины. Процесс разбиения сети на сообщества будет минимизировать данную метрику.

Пусть M — разбиение n вершин на m модулей:

$$\alpha = 1, \ldots, n, \quad i = 1, \ldots, m.$$

Пусть $L(M)$ — верхняя граница длины кодового слова, выступающая в качестве метрики качества полученного разбиения. Для вычисления метрики $L(M)$ для заданного разбиения M сначала применим теорему Шеннона об источнике шифрования, которая гласит, что, когда используется n кодовых слов для описания n состояний случайной величины X, которые встречаются с частотами p_i, средняя длина кодового слова не может быть меньше энтропии случайной переменной X, представимой в виде

$$H(X) = \sum_{i=1}^{n} p_i \log_2 p_i. \tag{3}$$

При вычислении метрики $L(M)$ будет использоваться формула (3). Для заданной сети зафиксируем разбиение M на сообщества. Введём случайную величину $\mathcal{Q}$, которая принимает значения от 1 до m (количество сообществ) с вероятностями $q^{(i)}$,где $i = 1, \ldots, m$. Для каждого сообщества i введём случайную величину $\mathcal{P}^i$, которая принимает значения от 1 до n_i (количество вершин в сообществе i) с вероятностями ρ_k^i, где $k = 1, \ldots, n_i$. Расчёт метрики $L(M)$ основывается на вычислении энтропии введённых случайных величин $\mathcal{Q}$ и $\mathcal{P}^i$

следующим образом:

$$L(M) = qH\Big(\mathcal{Q}) + \sum_{i=1}^{m} p_i H(\mathcal{P}^i\Big), \tag{4}$$

где

$$q = \sum_{i=1}^{m} q_i -$$

вероятность перехода между сообществами на каждом шаге случайного блуждания, q_i — вероятность покинуть сообщество i, p_α — вероятность посетить вершину α,

$$p_i = \sum_{\alpha \in i} p_\alpha + q_i -$$

вероятность остаться в сообществе i. Имеем

$$H(\mathcal{Q}) = -\sum_{i=1}^{m} \frac{q_i}{\sum\limits_{j=1}^{m} q_j} \log_2 \left(\frac{q_i}{\sum\limits_{j=1}^{m} q_j} \right), \tag{5}$$

$H(\mathcal{Q})$ — энтропия переходов между модулями, нижняя граница средней длины кодового слова для именования модулей;

$$H(\mathcal{P}^i) = \frac{q_i}{q_i + \sum\limits_{\beta \in i} p_\beta} \log_2 \left(\frac{q_i}{q_i + \sum\limits_{\beta \in i} p_\beta} \right) - \sum_{\alpha \in i} \frac{p_\alpha}{q_i + \sum\limits_{\beta \in i} p_\beta} \log_2 \left(\frac{p_\alpha}{q_i + \sum\limits_{\beta \in i} p_\beta} \right), \tag{6}$$

$H(\mathcal{P}^i)$ — энтропия перемещения внутри модуля i, нижняя граница средней длины кодового слова для именования вершин в модуле i. Из заданных формул (5) и (6) можно получить более детальное описание метрики $L(M)$ (формула (4)):

$$L(M) = \left(\sum_{i=1}^{m} q_i \right) \log_2 \left(\sum_{i=1}^{m} q_i \right) - 2 \sum_{i=1}^{m} q_i \log_2 q_i - \sum_{\alpha=1}^{n} p_\alpha \log_2 p_\alpha + \\ + \sum_{i=1}^{m} \left(q_i + \sum_{\alpha \in i} p_\alpha \right) \log_2 \left(q_i + \sum_{\alpha \in i} p_\alpha \right). \tag{7}$$

Заметим, что

$$\sum_{\alpha=1}^{n} p_\alpha \log_2 p_\alpha$$

не зависит от разбиения сети на сообщества. Поэтому, когда мы будем оптимизировать разбиение сети, необходимо будет хранить все изменения: q_i — вероятность, с которой случайное блуждание входит и выходит из сообществ, и $\sum\limits_{\alpha \in i} p_\alpha$ — долю времени, которую случайное блуждание проводит в каждом из сообществ.

Метрика качества полученного разбиения может быть легко посчитана для любого разбиения, обновление и пересчёт этой метрики будет быстрой операцией.

Любой численный метод, разработанный для нахождения разбиения сети, который оптимизирует целевую функцию, может быть использован для минимизации метрики $L(M)$.

Рассмотрим метод для неориентированных взвешенных сетей. Введём относительный вес вершины α:

$$w_\alpha = \sum_{l \in l_\alpha} w_l, \tag{8}$$

где l_α — множество инцидентных связей вершины α, w_l — вес связи l. Тогда с учётом (8) можно задать относительный вес сообщества i:

$$w_i = \sum_{\alpha \in i} w_\alpha. \tag{9}$$

Пусть w_i^{exit} — вес выхода из сообщества i:

$$w_i^{\text{exit}} = \sum_{l \in l_i^{\text{exit}}} w_l, \tag{10}$$

где l_i^{exit} — множество связей, выходящих из сообщества i.

Введём общий вес связей, соединяющих модули:

$$w^{\text{exit}} = \frac{\sum\limits_{i=1}^{m} w_i^{\text{exit}}}{2}.$$

При условии использования неориентированных взвешенных сетей формулы (8)—(10) преобразуют формулу (7) так, что метрика качества разбиения $L(M)$ будет выглядеть следующим образом:

$$L(M) = w^{\text{exit}} \log_2 w^{\text{exit}} - 2\sum_{i=1}^{m} w_i^{\text{exit}} \log_2 w_i^{\text{exit}} - \\ - \sum_{\alpha=1}^{n} w_\alpha \log_2 w_\alpha + \sum_{i=1}^{m} (w_i^{\text{exit}} + w_i) \log_2 (w_i^{\text{exit}} + w_i).$$

Теперь рассмотрим случай ориентированного взвешенного графа. Для начала необходимо ввести новый параметр — вероятность телепортации τ. Телепортация случайного блуждания — это равновероятный переход в случайную вершину в каждый момент времени во время процесса случайного блуждания;

$$p_\alpha = \frac{1}{n},$$

где p_α — вероятность начать случайное блуждание из вершины α.

Введём вероятность перехода из вершины α в вершину β:

$$w_{\alpha\beta}^{\mathrm{norm}} = \frac{w_{\alpha\beta}}{\sum\limits_{l \in l_\alpha^{\mathrm{out}}} w_l}, \tag{11}$$

где l_α^{out} — множество вершин, выходящих из вершины α, $w_{\alpha\beta}$ — вес связи между α и β. Если не существует выходящей связи из α в β, то $w_{\alpha\beta} = 0$.

В процессе случайного блуждания на каждом шаге осуществляется переход в соседнюю вершину с вероятностью $1 - \tau$ и в случайную вершину сети с вероятностью τ. При условии что сеть является ориентированной и взвешенной, вероятность выхода из сообщества i, используемая в формуле (7), будет выглядеть следующим образом:

$$q_i = \tau \frac{n - n_i}{n} \sum_{\alpha \in i} p_\alpha + (1 - \tau) \sum_{\alpha \in i} \sum_{\beta \notin i} p_\alpha w_{\alpha\beta}^{\mathrm{out}}, \tag{12}$$

где n_i — количество вершин в сообществе i.

В качестве обобщения процесса случайного блуждания можно ввести параметр τ_α — вероятность телепортации в вершину α, где

$$\sum_{\alpha=1}^{n} \tau_\alpha = 1.$$

Тогда вероятность выхода из сообщества i (формула (12)) примет вид

$$q_i = \tau \left(1 - \sum_{\alpha \in i} \tau_\alpha\right) \sum_{\alpha \in i} p_\alpha + (1 - \tau) \sum_{\alpha \in i} \sum_{\beta \notin i} p_\alpha w_{\alpha\beta}^{\mathrm{out}}.$$

Любой жадный алгоритм (быстрый, но не точный) или метод Монте-Карло (точный, но медленный) могут быть использованы для минимизации $L(M)$. В [3, 4, 14, 15] аналогичный метод использован лишь для структурного разделения сети на сообщества. Зачастую в социальных сетях содержится множество дополнительной информации [1, 2], хранящейся в различных типах данных. Поэтому при выделении сообществ необходимо использовать информационное содержимое сети для улучшения качества разбиения и его детальной настройки.

Пусть S — множество всевозможных строк, $T = \{\mathrm{int}, \mathrm{double}, \mathrm{string}\}$ — множество типов, $\mathrm{at} = (\mathrm{name}, \mathrm{type})$ — тип атрибута, где $\mathrm{name} \in S$, $\mathrm{type} \in T$, $TC = \{\mathrm{at}_i\}$ — множество (коллекция) типов атрибутов, $D_{\mathrm{at.type}}$ — множество всех допустимых значений типа атрибута $\mathrm{at} \in TC$, $A = (t, \mathrm{value})$ — атрибут, где $t \in TC$, $\mathrm{value} \in D_{t.\mathrm{type}}$, $G = (V, E)$ — мультиграф, где $V = \{V_i\}$ — множество вершин, $E = \{E_i\}$ — множество рёбер, такое что $E_i \in V \times V$. Пусть $\varphi\colon TC \to 2^V$ — отображение, задающее множество вершин, обладающих заданным атрибутом, $\varphi_t\colon \varphi(t) \to D_{t.\mathrm{type}}$, где $t \in TC$ — отображение, задающее значения атрибута в вершинах, содержащих данный тип атрибута, $\psi\colon TC \to 2^E$ — отображение, задающее множество связей, обладающих заданным атрибутом, $\psi_t\colon \psi(t) \to D_{t.\mathrm{type}}$, где $t \in TC$, — отображение, задающее значения атрибута

в связях, содержащих данный тип атрибута. Пусть ST — множество типов атрибутов, которые должны приниматься во внимание при выделении сообществ. Пусть $\mu_{ST}(v_i, v_j) \in [0; 1]$ — мера близости двух вершин по заданному набору типов атрибутов. Тогда для случая неориентированного графа вес сообщества i (формула (9)) примет вид

$$w_i = \left(\sum_{\alpha,\beta \in i} \frac{1 - \mu_{st}(\alpha, \beta)}{|i|^2} \right) \sum_{\alpha \in i} w_\alpha.$$

Для случая ориентированного графа вероятность перехода из вершины α в вершину β (формула (11)) преобразуется к виду

$$w_{\alpha\beta}^{\mathrm{norm}} = \frac{w_{\alpha\beta}}{\sum\limits_{l \in l_\alpha^{\mathrm{out}}} w_l} \big(1 - \mu_{st}(\alpha, \beta)\big).$$

Методология работы алгоритма с учётом баланса между качеством разбиения сети и скоростью работы алгоритма будет выглядеть следующим образом: соседние вершины соединяются в сообщества, которые в свою очередь соединяются в суперсообщества, и т. д. Затем в случайном порядке каждая вершина перемещается в соседнее сообщество с целью максимальной минимизации целевой функции. Если минимизировать целевую функцию нельзя, то вершина остаётся в первоначальном сообществе. Данная процедура повторяется до тех пор, пока нельзя будет переместить ни одну вершину и уменьшить целевую функцию. На каждом шаге генерируется случайная последовательность выбираемых вершин. Данным алгоритмом будет быстро найдено хорошее разбиение сети на сообщества.

Общую схему работы алгоритма выделения сообществ можно представить следующим образом.

1. Изначально каждая вершина считается отдельным сообществом. Вычисляется и запоминается целевая функция $L(M)$. Переход к пункту 2.
2. Случайное блуждание формирует последовательность вершин. Переход к пункту 3.
3. С учётом частоты встречаемости вершин в полученной последовательности выбираются подмножества вершин, которые объединяются в сообщества. Переход к пункту 4.
4. Для заданного разбиения вычисляется метрика $L(M)$. Если её значение стало меньше, то разбиение M сохраняется и продолжается работа алгоритма (переход к пункту 2). Иначе, если значение целевой функции $L(M)$ не уменьшилось, переход к пункту 5.
5. Полученное разбиение M считать результатом выделения сообществ в сети.

Найденные сообщества будут обладать следующим свойством: плотность связей внутри сообществ намного выше плотности связей между сообществами.

3. Вычислительные эксперименты

Тестирование алгоритма проведено в несколько этапов: тестирование на сгенерированных графах и тестирование на подграфах реальных социальных сетей.

Первый этап тестирования проводился на сгенерированных графах. Для оценки и тестирования алгоритма выделения сообществ необходимы графы с заданной структурой сообществ, которую алгоритм будет выделять. В [9, 11] предложена LFR-модель, при помощи которой можно протестировать алгоритм на разных конфигурациях сетей и сообществ в них. Рассмотрим LFR-модель как специальный случай планарной модели разбиения на l сообществ, при которой:

— размеры сообществ различны и распределены по степенному закону. Пусть l — количество сообществ в сети, $n_{\max}$ — максимальная степень сообщества (количество вершин, входящих в сообщество), $n_i \in \{1, 2, 3, \ldots, n_{\max}\}$ — степень i-го сообщества. Тогда

$$P(n_i = n) = \frac{c_1}{n^{\tau_1}},$$

где $c_1 = \mathrm{const}$, $\tau_1 = \mathrm{const}$;

— степени вершин распределены по степенному закону. Пусть N — количество вершин в сети, $k_{\max}$ — максимальная степень вершины, $k_i \in \{1, 2, 3, \ldots, k_{\max}\}$ — степень i-й вершины. Тогда

$$P(k_i = k) = \frac{c_2}{k^{\tau_2}},$$

где $c_2 = \mathrm{const}$, $\tau_2 = \mathrm{const}$.

Введём следующие обозначения: k_i^{in} — количество соседей i-й вершины, лежащих в том же сообществе, k_i^{out} — количество соседей i-й вершины, лежащих в других сообществах, c — сообщество вершины i, n_c — количество вершин, лежащих в сообществе c, k_c^{out} (k_c^{in}) — количество доступных соединений снаружи (внутри) сообщества c, сумма степеней вершин снаружи (внутри) данного сообщества, $\langle k \rangle$ — средняя степень вершины:

$$\langle k \rangle = \frac{\sum\limits_{i=1}^{N} k_i}{N}.$$

Тогда

$$k_c^{\mathrm{out}} \sim (N - n_c)\langle k \rangle, \quad k_c^{\mathrm{in}} \sim n_c \langle k \rangle.$$

Вероятность связи вершины i с вершинами из других сообществ вычисляется по формуле

$$p_i^{\mathrm{out}} = \frac{k_i^{\mathrm{out}}}{k_c^{\mathrm{out}}} \sim \frac{k_i^{\mathrm{out}}}{(N - n_c)\langle k \rangle}.$$

Вероятность связи вершины i с вершинами своего сообщества вычисляется по формуле

$$p_i^{\mathrm{in}} = \frac{k_i^{\mathrm{in}}}{k_c^{\mathrm{in}}} \sim \frac{k_i^{\mathrm{in}}}{n_c \langle k \rangle}.$$

Тестирование скорости работы алгоритма проводилось на сгенерированных графах. Приведём зависимость времени выполнения алгоритма от средней степени узла, где количество вершин — 10 000, максимальная степень узла — 40, вероятность связи между вершинами из разных сообществ — 10 %, минимальный размер сообщества — 100, максимальный размер сообщества — 1 000 (рис. 2). У сетей из теста отличалась средняя степень узла, что в свою очередь отразилось на количестве связей: у графа со средней степенью узла «4» 53 152 связи, у графа со средней степенью узла «35» 350 440 связей.

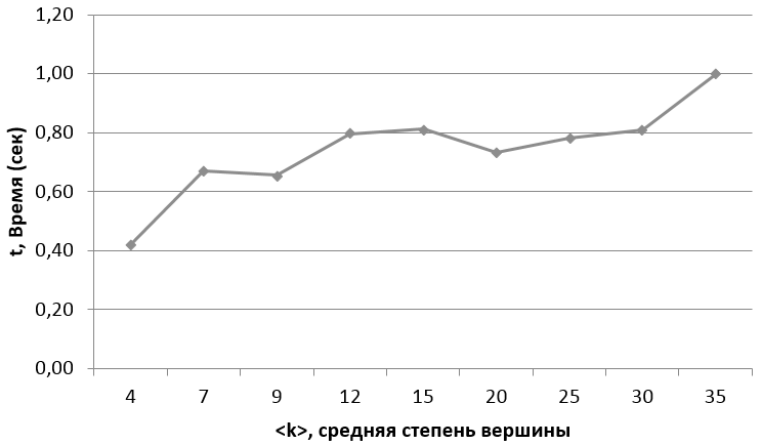

Рис. 2. Зависимость времени выполнения алгоритма от средней степени узла

Из рис. 2 видно, что представленную зависимость можно аппроксимировать линейной функцией, но при анализе асимптотики алгоритма необходимо учитывать структурные особенности конкретной сети. Для подтверждения факта линейной зависимости между временем выполнения алгоритма и средней степенью вершины требуются дополнительные исследования на различных конфигурациях сетей.

Приведём примеры анализа реальных подграфов социальных сетей, где вершины — это профили социальной сети, а связи — отношения дружбы между ними. Данные взяты из социальной сети «ВКонтакте» (`vk.com`) с использованием «API Вконтакте».

На рис. 3, 4 показаны результаты выделения сообществ в социальных сетях. Слева расположены вершины сети, размещёные случайным образом, справа приведены размещения по результатам работы алгоритма выделения сообществ. Все элементы одного сообщества размещаются по окружности относительно друг друга, сами сообщества располагаются также по окружности относительно друг друга. В данных примерах (рис. 3 и 4) продемонстрировано выделение сообществ из графа, описывающего отношения «дружбы». Видно, что вершины из

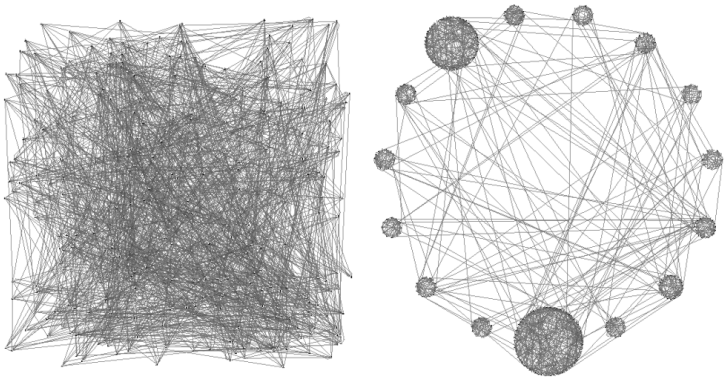

Рис. 3. До и после выделения сообществ. Количество вершин — 300, количество связей — 1357

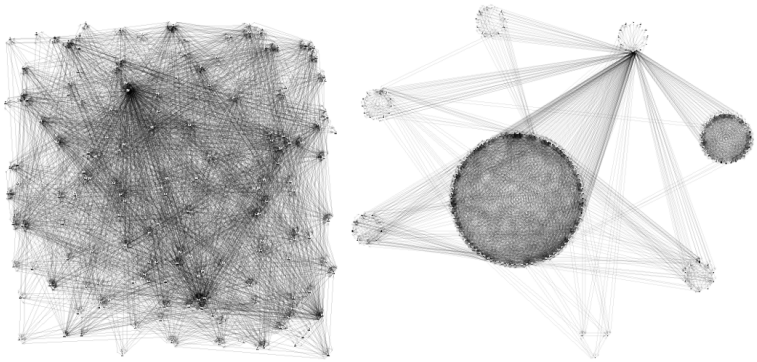

Рис. 4. До и после выделения сообществ. Количество вершин — 172, количество связей — 3026

одного сообщества почти все связаны между собой, но при этом слабо связаны с вершинами из других сообществ. Кроме того, видно, что в сети присутствует несколько объектов, которые имеют связи с вершинами из всех сообществ.

Таким образом, предложенная в данной работе модификация существующих методов применима к большим данным реальных социальных сетей и эффективно (с линейной оценкой сложности) решает задачу выделения сообществ с учётом информационного содержания.

Литература

[1] Батура Т. В. Методы анализа компьютерных социальных сетей // Вестн. НГУ, Сер. Информационные технологии. — 2012. — Т. 10, вып. 4. — С. 13—28.

[2] Чураков А. Н. Анализ социальных сетей // СоцИс. — 2001. — № 1. — С. 109—121.

[3] Blondel V. D., Guillaume J.-L., Lambiotte R., Lefebvre E. The Louvain method for community detection in large networks // J. Statist. Mech. Theory Experiment. — 2008. — P. 108—121.

[4] Clauset A., Newman M. E., Moore C. Finding community structure in very large networks // Phys. Rev. — 2004. — Vol. E 70, no. 6. — 066111.

[5] Fortunato S. Community detection in graphs // Phys. Rep. — 2010. — Vol. 486. — P. 75—174.

[6] Girvan M., Newman M. E. Community structure in social and biological networks // Proc. Natl. Acad. Sci. USA. — 2002. — Vol. 99. — P. 7821—7826.

[7] Guimera R., Sales-Pardo M., Amaral L. A. N. Modularity from fluctuations in random graphs and complex networks // Phys. Rev. — 2004. — Vol. E 70, no. 2. — 025101.

[8] Lambiotte R., Rosvall M. Ranking and clustering of nodes in networks with smart teleportation // Phys. Rev. — 2012. — Vol. E 85, no. 5. — 056107.

[9] Lancichinetti A., Fortunato S. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities // Phys. Rev. — 2009. — Vol. E 80, no. 1. — 016118.

[10] Lancichinetti A., Fortunato S. Community detection algorithms: a comparative analysis // Phys. Rev. — 2009. — Vol. E 80, no. 5. — 056117.

[11] Lancichinetti A., Fortunato S., Radicchi F. Benchmark graphs for testing community detection algorithms // Phys. Rev. — 2008. — Vol. E 78, no. 4. — 046110.

[12] Lovasz L. Random walks on graphs: a survey // Combinatorics, Paul Erdős is Eighty / D. Miklós, V. T. Sós, T. Szőnyi, eds. — (Bolyai Soc. Math. Stud.; Vol. 2). — Budapest, 1996. — P. 1—46.

[13] Massen C. P., Doye J. P. K. Identifying communities within energy landscapes // Phys. Rev. — 2005. — Vol. E 71. — 046101.

[14] Newman M. E. Fast algorithm for detecting community structure in networks // Phys. Rev. — 2004. — Vol. E 69. — 066133.

[15] Newman M. E. Modularity and community structure in networks // Proc. Natl. Acad. Sci. USA. — 2006. — Vol. 103. — P. 8577—8582.

[16] Newman M. E. Networks: An Introduction. — Oxford: Oxford Univ. Press, 2010.

[17] Newman M. E., Girvan M. Finding and evaluating community structure in networks // Phys. Rev. — 2004. — Vol. E 69. — 026113.

[18] Radicchi F., Castellano C., Cecconi F., Loreto V., Parisi D. Defining and identifying communities in networks // Proc. Natl. Acad. Sci. USA. — 2004. — Vol. 101. — P. 2658—2663.

[19] Rosvall M., Axelsson D., Bergstrom C. T. The map equation // Eur. Phys. J. Special Topics. — 2009. — Vol. 178, no. 1. — P. 13—23.

[20] Rosvall M., Bergstrom C. T. An information-theoretic framework for resolving community structure in complex networks // Proc. Natl. Acad. Sci. USA. — 2007. — Vol. 104, no. 18. — P. 7327—7331.

[21] Rosvall M., Bergstrom C. T. Maps of information flow reveal community structure in complex networks // Proc. Natl. Acad. Sci. USA. — 2008. — Vol. 105, no. 4. — P. 1118—1123.