Моделирование и анализ информационных систем. Т. 23, № 6 (2016), с. 777–783
Modeling and Analysis of Information Systems. Vol. 23, No 6 (2016), pp. 777–783

DOI: 10.18255/1818-1015-2016-6-777-783

УДК 004.9

Генерация графа социальной сети с использованием Apache Spark

Белов Ю. А., Вовчок С.И.

получена 24 октября 2016

Аннотация. Планируется создать метод кластеризации графа социальной сети. Для тестирования будущего метода возникла необходимость в генерации графа, по своей структуре схожего с лежащими в основе существующих социальных сетей. В статье представлен алгоритм для распределенной генерации такого графа. Учитываются основные свойства социальной сети: степенное распределение количества сообществ для пользователей, плотные пересечения сообществ и другие. В данном алгоритме учтены проблемы, присутствующие в подобных работах других авторов, например, проблема кратных ребер при генерации. Особенностью созданного алгоритма стала реализация, зависящая от такого параметра как количество сообществ, а не от количества пользователей, как это делается в других работах. Это связано с особенностью развития структуры реальной существующей социальной сети. В работе перечислены свойства ее графа. Описана таблица, содержащая необходимые для алгоритма переменные. Составлен пошаговый алгоритм генерации. Для него определены соответствующие математические параметры. Генерация происходит распределенно с помощью фреймворка Apache Spark. Подробно описано, каким образом происходит разделение задач с помощью данного фреймворка. В алгоритме используется модель Эрдеша–Реньи для случайных графов как наиболее подходящая и достаточно простая для реализации. Основными преимуществами созданного метода являются использование малого количества ресурсов, по сравнению с другими подобными генераторами, и скорость выполнения. Быстрота достигается за счет распределенной работы и того, что при распределенной работе алгоритма в любой момент пользователи сети имеют свои уникальные номера и упорядочены по этим номерам, поэтому не требуется их сортировка. Разработанный алгоритм будет способствовать не только созданию эффективного метода кластеризации. Он может быть полезен в других областях, связанных, например, с поисковыми системами социальных сетей.

Ключевые слова: социальная сеть, генерация

Для цитирования: Белов Ю. А., Вовчок С.И., "Генерация графа социальной сети с использованием Apache Spark", *Моделирование и анализ информационных систем*, **23**:6 (2016), 777–783.

Об авторах:
Белов Юрий Анатольевич, orcid.org/0000-0002-4221-6470, канд. физ.-мат. наук, доцент,
Ярославский государственный университет им. П.Г. Демидова,
ул. Советская 14, г. Ярославль, 150003 Россия, e-mail: belov45@yandex.ru

Вовчок Сергей Иванович, orcid.org/0000-0003-0535-5786, аспирант,
Ярославский государственный университет им. П.Г. Демидова,
ул. Советская 14, г. Ярославль, 150003 Россия, e-mail: vof4ok@gmail.com

Введение

В настоящее время уже существуют методы для генерации графа социальной сети, многие из которых рассматривались при подготовке данной работы. Большинство методов генерируют граф, не схожий по перечисленным далее в работе свойствам со структурой реальной социальной сети. Внимание привлекла работа 2014 года [1]. В ней разработан процесс генерации графа, основанный на модели AGM [2] и названный CKB. Нами была поставлена цель создать алгоритм для генерации графа социальной сети, усовершенствовав CKB, в частности минимизировав количество необходимой для хранения структур памяти, решив проблему кратных ребер при реализации.

Итоговый алгоритм стал интересен своей реализацией на основе количества сообществ сети, а не количества пользователей. Такая идея лучше передает процесс развития структуры социальной сети. Первоначально в сети появляется группа знакомых людей, образующих сообщество. Далее, благодаря рекламе, в сети регистрируются новые группы знакомых людей и отдельные пользователи из различных городов и стран. Постепенно часть из них образуют большие по количеству членов сообщества. Этот процесс и стал основой первой части алгоритма, в которой генерируются независимые сообщества с пользователями. Связи между пользователями образуются по модели Эрдеша–Реньи [4], как наиболее подходящей [3] для быстрого выполнения алгоритма.

Уже внутри социальной сети на следующем этапе ее развития ранее незнакомые друг другу пользователи из различных групп начинают контактировать и тем самым образуют новые связи. В результате пользователь сети становится участником нескольких сообществ. Социальная сеть получает более реалистичную структуру. Это стало идеей второй части алгоритма.

В алгоритме использована идея модели Чунг-Лу [5], в которой существует возможность выбора показателя степенного распределения вершин. Итоговый метод пошагово описывается далее. Программная реализация производится на фреймворке Apache Spark [6]. Планируется рядом экспериментов подтвердить эффективность алгоритма по отношению к существующим и подтвердить выполнение в полученном графе свойств социальной сети.

1. Постановка задачи

Необходимо сгенерировать граф $G = (V, E)$, где $|V| = A_1$ и $|E| = B$. Сообщество C_i является подграфом графа G, размер сообщества $|C_i| = r_{C_i}$. Количество сообществ равно A_2. Все вместе они образуют вершинное покрытие графа. Количество вхождений j-й вершины в разные сообщества — b_j.

Вершина $j \in C_i$ имеет внутреннюю степень d^{int}_{j,C_i} и внешнюю степень d^{ext}_{j,C_i}. Внутренняя — определяется как количество ребер, соединяющих j с другими вершинами в C_i. Внешняя степень — соответственно количество ребер, соединяющих j с остальной частью графа. Общая степень вершины j равна $d_j = d^{int}_{j,C_i} + d^{ext}_{j,C_i}$.

Количество ребер внутри сообщества C_i равно $B_{C_i} = \frac{1}{2}\Sigma_{\forall j \in C_i} d^{int}_{j,C_i}$.

Задача состоит в том, чтобы сгенерировать граф социальной сети G со следующими свойствами:

1. Размеры сообществ графа распределены по степенному закону с экспонентой $\alpha > 0$:
$$\forall i \in \mathbb{N} \Rightarrow p(r_{C_i}) \sim r_{C_i}^{-\alpha}; \tag{1}$$

2. В графе присутствует большая компонента связности:
$$\exists G^* \subset G \colon |V^*| \sim A_1, \forall i, j \in V^* \exists \omega_1, \omega_2, \ldots, \omega_k \in E^* \exists t_l \in V \colon \\ \omega_0 = (i, t_0), \omega_1 = (t_0, t_1), \ldots, \omega_D = (t_{D-1}, j); \tag{2}$$

3. Граф имеет диаметр $D \approx \frac{\ln A_1}{\ln \ln A_1}$:
$$\forall 0 < \epsilon < 1 \colon (1-\epsilon)\frac{\ln A_1}{\ln \ln A_1} \leq D \leq (1+\epsilon)\frac{\ln A_1}{\ln \ln A_1}, \tag{3}$$
где
$$D = \max_{i,j \in V^*}(d_{i,j}),$$
$$d_{i,j} = min_{\omega_s \in E^*}(|\{\omega_1, \omega_2, \ldots, \omega_k \mid \omega_0 = (i, t_0), \omega_1 = (t_0, t_1), \ldots, \omega_k = (t_k, j)\}|);$$

4. Пользователи могут не иметь связей и/или не входить ни в одно сообщество:
$$\forall i \in N \Rightarrow d_i \geq 0, b_i \geq 0; \tag{4}$$

5. Сообщества могут пересекаться:
$$\exists i, j \in \mathbb{N} \colon C_i \bigcap C_j \neq \emptyset; \tag{5}$$

6. С большой вероятностью каждое сообщество C_i является связным графом:
$$\forall k \in \mathbb{N} \Rightarrow P(C_k = (V_k, E_k) \colon \forall i, j \in C_k \exists \omega_t \in E_k \colon \\ \omega_0 = (i, t_0), \omega_1 = (t_0, t_1), \ldots, \omega_l = (t_l, j)) \geq 1 - \frac{1}{r_{C_i}}; \tag{6}$$

7. Плотность ребер внутри сообществ больше, чем средняя плотность ребер во всем графе G:
$$\forall i \in \mathbb{N} \Rightarrow \frac{d_{C_i}}{r_{C_i}(r_{C_i} - 1)} > \frac{b}{A_1(A_1 - 1)}; \tag{7}$$

8. Количество ребер внутри сообщества больше, чем количество ребер, соединяющих вершины сообщества с остальной частью графа:
$$\forall i \in \mathbb{N} \Rightarrow d_{C_i} > \Sigma_{\forall j \in C_i} d_{j,C_i}^{ext}; \tag{8}$$

9. Количество ребер в сообществе растет суперлинейно с размером сообщества:
$$\forall i \in \mathbb{N} \Rightarrow d_{C_i} \propto r_{C_i}^{1+\gamma}, \gamma \in (0, 1); \tag{9}$$

10. Пересечение сообществ более плотно, чем их непересекающаяся часть:
$$\forall i, j \in \mathbb{N}(i \neq j), C_{i \bigcap j} = C_i \bigcap C_j \Rightarrow \\ \frac{d_{C_{i \bigcap j}}}{r_{C_{i \bigcap j}}(r_{C_{i \bigcap j}} - 1)} > \frac{d_{C_i}}{r_{C_i}(r_{C_i} - 1)}. \tag{10}$$

Свойства взяты из результатов исследований структуры реальной социальной сети [1]

2. Предложенный метод

Основные шаги алгоритма следующие:

1. Распределенно и независимо генерируются сообщества пользователей в виде списков связанных друг с другом пользователей.

2. Объединяя пары пользователей из различных сообществ в одну вершину, получаем граф социальной сети.

3. Подробный алгоритм предложенного метода

Таблица 1. Входные параметры для генерации
Table 1. Input parameters for generation

Параметры Parameters	**Значение Value**	**По умолчанию By default**
A_2	Количество сообществ	—
b_{min}	Минимальное число сообществ, в которое входит один пользователь	1
b_{max}	Максимальное число сообществ, в которое входит один пользователь	10000
x_{min}	Минимальный размер сообщества	2
x_{max}	Максимальный размер сообщества	10000
$\alpha > 1$	Экспонента степенного распределения размеров сообщества	$2,5$
$\beta > 1$	Экспонента степенного распределения числа вхождений пользователя в сообщества	$2,5$

1. Генерируем последовательность для степенного распределения размеров сообществ: ставим в соответствие каждому j-му сообществу d_j.

2. Вводим вспомогательные переменные D_j: $D_0 = 1$, $D_1 = d_1$, $D_2 = D_1 + d_2$, $\cdots$, $D_{A_2} = D_{A_2-1} + d_{A_2}$. Таким образом, D_j равно количеству пользователей для первых j сообществ. Т. е. если $d_1 = 5$; $d_2 = 4$; $d_3 = 3$; $d_4 = 5$, то $D_1 = 5$; $D_2 = 9$; $D_3 = 12$; $D_4 = 17$.

3. Вычисляем количество пользователей A_1 создаваемой социальной сети, а также — X_{A_1}, равное поправке на объединение в одну вершину пар пользователей из различных сообществ. В данной работе предполагается, что X_{A_1} — это математическое ожидание числа вхождений пользователей в сообщества ($E[b]$),

которое вычисляется по формуле

$$E[b] = \int_{b_{min}}^{b_{max}} bp(b)db = \frac{(1-\beta)(b_{max}^{2-\beta} - b_{min}^{2-\beta})}{(b_{max}^{1-\beta} - b_{min}^{1-\beta})(2-\beta)}. \tag{11}$$

A_1 рассчитывается исходя из уравнения:

$$A_1 \cdot E[b] = A_2 \cdot E[x], \tag{12}$$

где $E[x]$ — математическое ожидание размеров сообществ, рассчитывается аналогично $E[b]$.

4. Распределенно (с использованием нескольких компьютеров, объединенных в параллельную вычислительную машину) генерируем сообщества и составляем для каждого из них «списки» пользователей при помощи инструментов фреймворка Apache Spark. В терминологии Apache Spark в вычислительном кластере существует главный узел, называемый *мастер*, и ведомые узлы, называемые *слэйвами*. Для генерации «списка» пользователей для сообщества передаем на слэйвы данные о каждом i-м сообществе. Данными являются номера пользователей от $D_{i-1}+1$ до D_i для i-го сообщества. В сгенерированном «списке» для i-го сообщества содержатся уникальные номера пользователей от $D_{i-1}+1$ до D_i. Таким образом, все пользователи распределены в «списки» сообществ без повторения. Количество сообществ равно A_2. Их размер распределен по степенному закону.

5. Распределенно для каждого «списка» на слэйвах запускаем модель Эрдеша–Реньи [4] для случайных графов, в результате чего получаем для каждого пользователя в «списке» связанных с ним пользователей из данного «списка». Так как пользователи имеют уникальные номера, на этом шаге имеем возможность проверять при добавлении связи, нет ли уже добавленной идентичной. Тем самым в дальнейшем убираем возможность существования кратных ребер в сообществах. После выполнения данного шага алгоритма у нас есть независимые сообщества пользователей со списками связей между пользователями.

6. «Сцепляем» полученные на предыдущем шаге списки связей в один упорядоченный общий список пользователей. Список каждого i-го сообщества «сцепляется» с $i+1$-м. Эта операция возможна благодаря уникальным номерам пользователей. Она не требует дополнительной сортировки пользователей по номерам, так как изначально списки в сообществах упорядочены. В результате получается общий список сообществ, каждое из которых сгенерировано с помощью алгоритма Эрдеша–Реньи.

7. Вводим значение «*», необходимое для того, чтобы помечать элемент списка, который не будет использован в дальнейшем.

8. Из двух разных сообществ выбираем двух пользователей. Для этого случайным образом задаем i и j : $i \neq j$ и $0 \leq i, j \leq A_2$. Выбираем пользователей

Z_1 в отрезке от $(D_{i-1}+1)$-го элемента списка до D_i-го и Z_2 — от $(D_{j-1}+1)$-го до D_j-го, не равные «∗». Нам необходимо объединить данных пользователей, сделав из них одного. Для этого в общем списке добавляем к Z_1 все связи Z_2. А также выполняем проход от $(D_{j-1}+1)$-го до D_j-го элемента, заменяя значения смежных с элементами вершин равных Z_2 на Z_1. Сам элемент Z_2 заменяем на «∗».

9. Предыдущий шаг повторяем X_{A_1} раз, т. е. количество поправок, указанное ранее, на объединение в одну вершину пар пользователей из различных сообществ.

10. Создаем пустой граф $G(V, E)$ на A_1 вершин. Переносим данные, не равные «∗», из полученного списка в данный граф.

В результате выполнения алгоритма получаем граф социальной сети.

Список литературы / References

[1] Чихрадзе К. К. и др., “Использование модели социальной сети с сообществами пользователей для распределенной генерации случайных социальных графов”, *Машинное обучение и анализ данных*, **1**:8 (2014), 1027–1047; [Chikhradze K. K. et al., “On a model of social network with user communities for distributed generation of random social graphs”, *Machine Learning and Data Analysis*, **1**:8 (2014), 1027–1047, (in Russian).]

[2] Yang J., Leskovec J., “Community-affiliation graph model for overlapping network community detection”, *IEEE 12th Conference (International) on Data Mining*, 2012.

[3] Raigorodskii A., “Models of Random Graphs and Their Applications to the Web-Graphs Analysis”, *RUSSIR-2015*, Lomonosov Moscow University, Moscow Istitute of Physics and Technology, Yandex, Moscow, Russia, 25–29 August, 2015.

[4] Erdos P., Renyi A., “On the evolution of random graphs”, *Bull. Inst. Int. Statist. Tokyo*, **38** (1961), 343–347.

[5] Aiello W., Chung F., Lu L., “On the evolution of random graphs”, *A random graph model for power law graphs*, http://people.math.sc.edu/lu/papers/power.pdf.

[6] Карау Х., и др., *Изучаем Spark: молниеносный анализ данных*, ДМК Пресс, М., 2015, 304 с.; [Karau K. et al., *Learning Spark: Lightning-Fast Data Analysis*, DMK Press, Moscow, 2015, 304 pp., (in Russian).]

[7] Вовчок С. И., “Создание метода кластеризации графа социальной сети”, *Новые информационные технологии в науке: сборник статей Международной научно-практической конференции МЦИИ ОМЕГА САЙНС. Ч. 2*, 2016, 34–36; [Vovchok S. I., “Sozdanie metoda klasterizatsii grafa sotsialnoy seti”, *Novye informatsionnye tekhnologii v nauke: sbornik statey Mezhdunarodnoy nauchno-prakticheskoy konferentsii MTsII OMEGA SAYNS. Part 2*, 2016, 34–36, (in Russian).]

Belov Y. A., Vovchok S.I., "Generation of a Social Network Graph by Using Apache Spark", *Modeling and Analysis of Information Systems*, **23**:6 (2016), 777–783.

DOI: 10.18255/1818-1015-2016-6-777-783

Abstract. We plan to create a method of clustering a social network graph. For testing the method there is a need to generate a graph similar in structure to existing social networks. The article presents an algorithm for the graph distributed generation. We took into account basic properties such as

power-law distribution of the users communities number, dense intersections of the social networks and others. This algorithm also considers the problems that are present in similar works of other authors, for example, the multiple edges problem in the generation process. A special feature of the created algorithm is the implementation depending on the communities number parameter rather than on the connected users number as it is done in other works. It is connected with a peculiarity of progressing the existing social network structure. There are properties of its graph in the paper. We described a table containing the variables needed for the algorithm. A step-by-step generation algorithm was compiled. Appropriate mathematical parameters were calculated for it. A generation is performed in a distributed way by Apache Spark framework. It was described in detail how the tasks division with the help of this framework runs. The Erdos-Renyi model for random graphs is used in the algorithm. It is the most suitable and easy one to implement. The main advantages of the created method are the small amount of resources in comparison with other similar generators and execution speed. Speed is achieved through distributed work and the fact that in any time network users have their own unique numbers and are ordered by these numbers, so there is no need to sort them out. The designed algorithm will promote not only the efficient clustering method creation. It can be useful in other development areas connected, for example, with the social networks search engines.

Keywords: social network, generation

About the authors:
Yuriy A. Belov, orcid.org/0000-0002-4221-6470, PhD,
P.G. Demidov Yaroslavl State University,
4 Sovetskaya ul., Yaroslavl 150003, Russia, e-mail: belov45@yandex.ru

Sergei I. Vovchok, orcid.org/0000-0003-0535-5786, graduate student,
P.G. Demidov Yaroslavl State University,
14 Sovetskaya ul., Yaroslavl 150003, Russia,
e-mail: vof4ok@gmail.com