

УДК 004.852

DOI: 10.15827/2311-6749.18.3.7

РАЗРАБОТКА ПРИЛОЖЕНИЯ ДЛЯ ПОСТРОЕНИЯ РЕКОМЕНДАЦИЙ СООБЩЕСТВ СОЦИАЛЬНОЙ СЕТИ ВКОНТАКТЕ

*О.С. Самохвалова, студентка, its34play@yandex.ru; К.Д. Уляшев, студент,
qwertypalmo@gmail.com*

*(Санкт-Петербургский государственный электротехнический университет «ЛЭТИ»
им. В.И. Ульянова (Ленина) (СПбГЭТУ «ЛЭТИ»), ул. Профессора Попова, 5,
г. Санкт-Петербург, 197376, Россия)*

Целью работы являлась разработка рекомендательной системы для социальной сети ВКонтакте. Предметом рекомендаций были выбраны сообщества в данной социальной сети. В качестве основной идеи алгоритма рекомендаций использован принцип коллаборативной фильтрации, но в более простом виде: рекомендации составляются на основе знаний о сообществах, в которых состоят друзья пользователя. В результате выполнения работы было разработано приложение с интерфейсом пользователя, позволяющее получить список рекомендованных сообществ социальной сети ВКонтакте.

Ключевые слова: *социальные сети, рекомендации, построение рекомендаций, Neo4j, NoSQL, графовые БД, social networks.*

В настоящее время в связи с ростом количества информации по любым темам встает вопрос о том, как найти интересную или необходимую информацию в сети интернет, в библиотечных системах или хранилищах данных. В связи с этим стали активно развиваться алгоритмы контекстного поиска, которые позволяют предложить пользователю то, что его, скорее всего, интересует. Одной из разновидностей контекстного поиска являются рекомендательные системы.

Рекомендательные системы – это инструменты автоматической генерации предложений товаров и услуг на основе знаний о потребностях клиентов. Они стали появляться еще в 2000-х [1], однако настоящий подъем в этой области произошел примерно 5–10 лет назад.

На данный момент существует немало успешных самостоятельных рекомендательных сервисов, таких как Last.fm [2], Pandora [3] и IMDb [4]. Помимо этого, все крупные социальные сети используют подобные алгоритмы для рекомендаций контента своим пользователям, например, Instagram [5] предлагает потенциально интересные пользователю профили, а ВКонтакте [6] составляет плейлисты с теми музыкальными композициями, которые, по мнению рекомендательной системы, должны понравиться пользователю.

В социальной сети ВКонтакте созданы тысячи сообществ по интересам, популярность которых стремительно набирает обороты, но инструменты сети не предоставляют пользователям возможность получить рекомендации в этой области. Также в интернете сложно найти удобное и доступное решение данной проблемы. Разработанное авторами приложение, реализующее построение рекомендаций на основе интересов друзей, решает ее.

Для выполнения задачи использовался принцип коллаборативной фильтрации [7]. Коллаборативная фильтрация предполагает расчет рекомендаций на основе оценок других пользователей (в данном случае учитывалась информация о тех сообществах, в которых состоят друзья пользователя), составляется граф интересов и делается рекомендация.

В итоге было разработано веб-приложение с простым и удобным интерфейсом, с помощью которого пользователь может получить две версии корректного списка рекомендованных сообществ: на основе информации о подписках его друзей с учетом и без учета количества общих интересов с пользователем.

Задачи данной работы – исследование существующих решений по рекомендациям сообществ пользователям ВКонтакте и реализация приложения, позволяющего пользователям получать список рекомендуемых к просмотру сообществ ВКонтакте.

Обзор предметной области

Перед созданием приложения был осуществлен поиск аналогичных инструментов для построения рекомендаций сообществ популярных социальных сетей, но все найденные решения рекомендовали все что угодно, кроме сообществ. В рамках исследования методов решения был проведен анализ приложений, которые также исследовали рекомендательные алгоритмы для социальных сетей.

Сравним существующие аналоги.

В статье [8] рассматривается проблема автоматизации процесса веб-серфинга и фильтрации контента, а также описаны проектирование и реализация мультиагентной рекомендательной системы EZSurf, обеспечивающей анализ интересов и предоставление рекомендаций пользователям социальной сети ВКон-

также на основе данных из профиля конкретного пользователя. Авторами статьи спроектирована рекомендательная система, реализующая метод фильтрации содержимого (content-based) и использующая профиль пользователя социальной сети ВКонтакте.

В работе [9] исследована применимость алгоритма Random Walk with Restarts в крупных социальных сетях для рекомендации музыкальных треков. Предложены модификации оригинального алгоритма, позволяющие улучшить качество рекомендаций, а также способ составления рекомендации в режиме реального времени по запросу пользователя. Так как оригинальный алгоритм при работе использует структуры данных, которые приобретают колоссальные размеры в реальных социальных сетях, был представлен «облегченный» вариант алгоритма, описаны качество и время его работы.

Статья [10] посвящена разработке рекомендательной системы для пользователей интернет-сервиса Steam, специализирующегося на продаже компьютерных игр. Разрабатываемая система использует гибридную модель фильтрации, состоящую из коллаборативной фильтрации (collaboration filtration), а также из фильтрации, основанной на контенте (content-based). Такие системы направлены на компенсацию недостатков одного подхода внедрением другого.

Для сравнения аналогов использовались определенные критерии.

Основная идея работы – создание простого рабочего приложения, чтобы любой мог воспользоваться данным инструментом. Также предполагалось, что для пользователя имеет значение время, необходимое для решения поставленной задачи, то есть одна из главных целей – высокая скорость работы алгоритмов рекомендаций. И третий критерий – простота алгоритма, под которой подразумевается низкая сложность вычислений, производимых данным алгоритмом.

Результаты сравнения рассмотренных решений и алгоритма, полученного в результате выполнения данной работы, сведены в таблицу.

Сравнение по критериям

Название статьи	Доступность приложениям	Скорость построения рекомендаций	Простота алгоритма
Построение рекомендаций сообществ социальной сети «ВКонтакте»	+	Алгоритм срабатывает за несколько секунд, но загрузка данных занимает несколько минут	Примитивный алгоритм. Основная сложность заключается в запросах к базе Neo4j
Разработка рекомендательной системы на основе данных из профиля социальной сети «ВКонтакте»	–	–	Сложный мультиагентный механизм
Рекомендации треков в социальных сетях	–	Требует значительных временных ресурсов	Алгоритм несет в себе нетривиальные расчеты
Альтернативная рекомендательная система для пользователей Steam	–	–	Алгоритм несет в себе нетривиальные расчеты

Сделаем определенные выводы. Рассмотренные рекомендательные системы используют более сложные алгоритмы, из чего можно предположить, что время их работы значительно выше значений, полученных при тестировании авторского приложения. Надо отметить, что рассмотренные аналоги не связаны с построением рекомендаций именно сообществ сети ВКонтакте. Так что, созданное авторами решение потенциально найдет отклик среди пользователей социальной сети ВКонтакте.

Обзор подходов и выбор метода решения

Существует множество подходов [7] для поиска рекомендаций, самыми популярными из которых являются коллаборативная фильтрация, content-based и гибридный подход.

- Коллаборативная фильтрация на данный день самый популярный вид рекомендательных систем. Системы коллаборативной фильтрации – это модели, которые пытаются предсказать, насколько вам понравится тот или иной продукт, исходя из входных данных о том, как вы и другие пользователи оценивали этот и другие продукты в прошлом. Хотя такие системы достаточно эффективны, их точность в значительной степени зависит от того, насколько пересекаются оцениваемые объекты между отдельными пользователями. Чем больше и чаще эти пересечения, тем точнее будет система. Этот факт ограничивает сферу применения данной модели; если каждый пользователь будет располагать своей уникальной информацией, которую никто другой не видел, коллективные рекомендательные системы будут бессильны.

ны. Также одним из основных недостатков коллаборативной фильтрации является проблема холодного старта: такая система не может построить рекомендации для нового пользователя, так как еще ничего про него не знает, аналогично с новыми продуктами – они никому не будут рекомендованы, пока не накопится информация о них.

- Content-based – это система, которая ищет объекты, похожие на те, что пользователь уже положительно оценил. Описываемый подход чаще всего используется для текстов (документов, сайтов, блогов) или объектов, которые можно описать ключевыми словами. В рекомендательных системах контентного типа полезность рекомендуемых объектов находится из оценок пользователя, данных схожим объектам. Например, чтобы посоветовать человеку документ, контентная система пытается найти сходство между различными объектами, которые ранее получили у пользователя высокую оценку (сходство может заключаться в тематике, авторе и даже в похожих названиях глав в документе).

- Демографический подход заключается в сравнении характеристик объекта с характеристиками пользователя. Рассматриваться могут как простые дискретные свойства (возраст, пол, место проживания, национальность), так и более сложные, например, интерес пользователя к определенному объекту. Некоторые из них система может легко определить самостоятельно: страну и даже город, в котором проживает пользователь, можно определить по IP, а предпочитаемый язык узнать из свойств браузера, используя JavaScript. Другие данные можно попросить пользователя указать явно. Несмотря на то, что объем полученной информации о пользователе будет достаточно ограничен, она может помочь принять важные решения, например, кому рекомендовать куклу, а кому видео для взрослых.

- Статистический подход – это системы, основанные на статистических данных, собранных от пользователей. Проще говоря, это привычные нам рейтинги самых-самых: самые скачиваемые, самые читаемые, самые популярные и т.п. Данный подход решает проблему холодного старта.

- Ассоциативный подход строит рекомендации на основе данных о том, какие объекты используются вместе. Яркий пример применения таких систем – анализ покупок пользователей. Например, кто-то купил себе новый телефон, и вполне вероятно, что к нему понадобятся наушники, чехол, зарядка, дополнительная карта памяти и прочие аксессуары. Рекомендательные системы здесь могут быть бессильны, ведь у этих объектов нет никаких общих параметров и сравнить их не представляется возможным, их объединяет лишь то, что они используются вместе.

- Гибридный подход объединяет в себе основные принципы сразу нескольких подходов. Концепция данного подхода заключается в том, что разные системы могут эффективно взаимодействовать, тем самым компенсируя недостатки друг друга.

Многие из алгоритмов, используемых рекомендательными системами, пришли из области машинного обучения, которая занимается алгоритмами для обучения, прогнозирования и принятия решений. Эти алгоритмы реализуют нетривиальные математические вычисления, так как в их задачу входят не только построение списка рекомендаций, но и масштабируемость, анализ поведения пользователей и предсказание их дальнейших действий. В рамках выполнения данной работы решение вышеперечисленных задач избыточно, поэтому использование данных алгоритмов нецелесообразно.

Полученная рекомендательная система основана на использовании нереляционной БД Neo4j [11]. Вся логика построения рекомендаций реализована через запросы к базе. Такой подход позволяет максимально облегчить алгоритм и ускорить его. По принципу построения рекомендаций данный способ имеет общие черты с коллаборативной фильтрацией: в качестве основы рекомендаций лежит информация о сообществах друзей пользователя, то есть используется информация о том, как оценили то или иное сообщество другие пользователи социальной сети.

Исследование модели данных

Авторы провели исследование аналогов выбранной модели данных для SQL СУБД, в ходе которого выяснено, что для решения поставленной задачи больше подходит NoSQL-модель. Причины такого вывода:

- NoSQL-модель занимает меньше ресурсов памяти по сравнению с SQL-моделью; модели Neo4j и SQL не подразумевают коллекций, однако, если говорить о коллекции как о наборе однородных данных, то стоит отметить, что в авторской NoSQL-модели различаются два набора: множество вершин графа и множество ребер, в то время как SQL-модель содержит четыре набора: таблицы Person, Friend, Group, Take_Part_In (рис. 1);

- SQL-запросы были бы более громоздкими и сложными, чем NoSQL-запросы.

В таблицу Person входит информация о пользователе и его друзьях: personId (int) – id пользователя в БД; name (String) – имя пользователя в социальной сети; vkId (int) – id пользователя в социальной сети; avatar (String) – ссылка на фотографию пользователя в профиле.

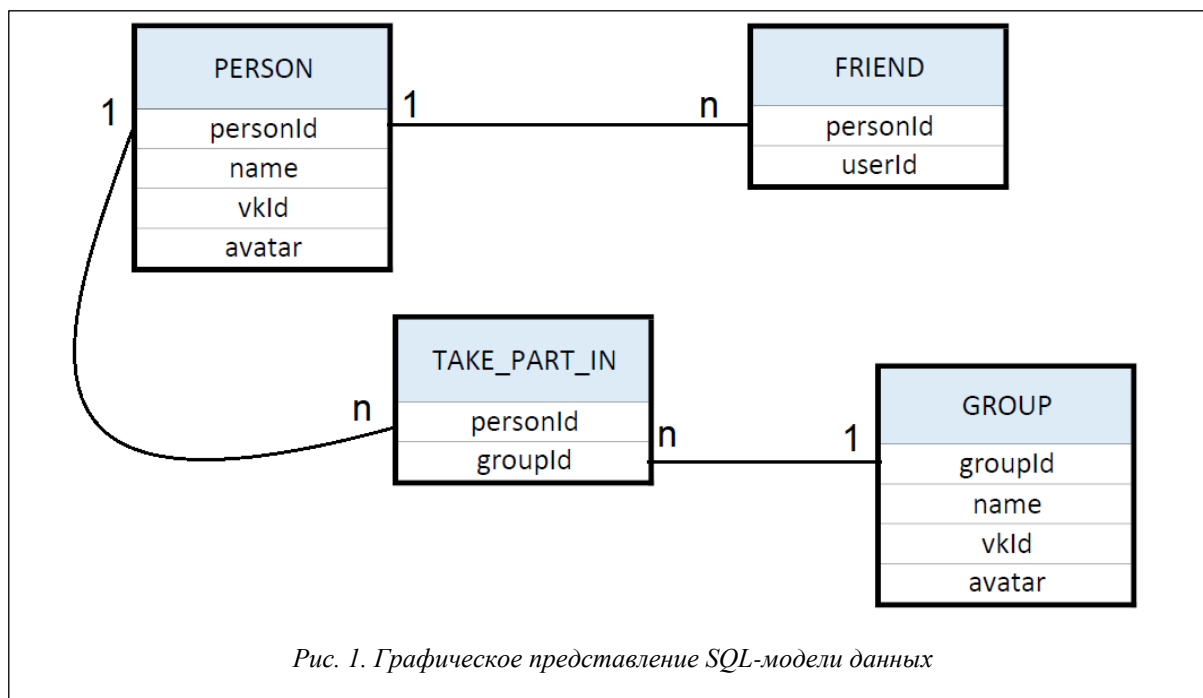


Таблица Group содержит информацию о сообществах, в которых состоит пользователь и/или его друзья: groupId (int) – id сообщества в БД; name (String) – имя сообщества в социальной сети; vkId (int) – id сообщества в социальной сети; avatar (String) – ссылка на фотографию сообщества в социальной сети.

Таблица Friend описывает связи между пользователем и его друзьями: personId (int) – id пользователя в БД (друзья); userId (int) id пользователя в БД (пользователь приложения).

Таблица Takes_Part_In описывает связи между людьми и сообществами: personId (int) – id пользователя в БД; groupId (int) – id сообщества в БД.

Описание решения и используемые технологии

Полученный инструмент представляет собой веб-приложение с интерфейсом пользователя. Разработка осуществлялась с использованием технологии Spring Boot [12]. Интерфейс пользователя реализован с помощью фреймворка Angular 5 [13].

Для работы приложения необходимо загрузить данные. Загрузка данных в программу происходит автоматически после авторизации пользователя с помощью профиля ВКонтакте. Авторизация реализована с помощью средств Spring Security [14] и протокола OAuth [15]. Используя VK API [16], приложение получает информацию об авторизованном пользователе и его друзьях, а также о сообществах, в которых они состоят. Стоит отметить, что было разработано API также для импорта данных вручную. После загрузки данные сохраняются в БД Neo4j (рис. 2).

Вершины графа бывают двух типов – Person и Group. Корневая вершина имеет тип Person и хранит идентификатор пользователя, остальные вершины типа Person хранят идентификаторы его друзей, вершины-листья имеют тип Group и хранят идентификатор сообщества. Вершины имеют следующие свойства: id(int) – id, присваиваемый БД конкретной вершине; name(String) – имя пользователя в социальной сети ВКонтакте; vkId(int) – id пользователя в социальной сети ВКонтакте; avatar(String) – ссылка на фотографию пользователя из профиля ВКонтакте.

Ребра между корнем и Person-вершинами создают связи FRIEND (связь пользователя с другом). Ребра между Person-вершинами и Group-вершинами создают связи TAKES_PART_IN (связь человека с сообществом). Каждый режим построения алгоритмов реализуется посредством одного запроса к БД.

Представим запросы к БД.

Рекомендации на основе количества друзей: MATCH (g:Group), (g)-[r]-(p2:Person), (p:Person) WHERE NOT (g)-[:TAKES_PART_IN]-(p) AND p.vkId = {0} RETURN g as group, COLLECT(p2) as friends, COUNT(r) AS rating ORDER BY rating DESC LIMIT {1};

Рекомендации на основе общих интересов с друзьями: MATCH (g:Group), (g)-[r]-(p3:Person), (p:Person) WHERE NOT (g)-[:TAKES_PART_IN]-(p) and p.vkId = {0} WITH g, p3 MATCH (p1:Person)-[:TAKES_PART_IN]-(g1:Group)-[:TAKES_PART_IN]-(p3) WHERE p1.vkId = {0} " + "WITH g, p3, COUNT(g1) as weight RETURN g as group, COLLECT(p3) as friends, SUM(weight) as rating ORDER BY rating DESC LIMIT {1};

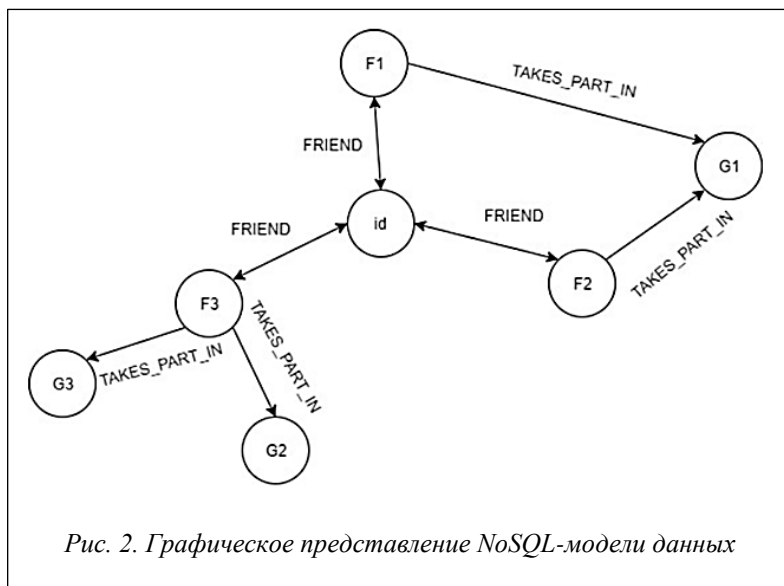


Рис. 2. Графическое представление NoSQL-модели данных

Рекомендательная система работает в двух режимах.

Первый режим загружает из БД все Group-вершины, которые не имеют связей с корневой вершиной, и ранжирует их в порядке убывания количества связей типа TAKE_PART_IN. Иными словами, рассматриваются сообщества, в которых состоят друзья пользователя, но не состоит сам пользователь, составляется рейтинг по количеству вхождений в сообщество друзей.

Второй режим сначала анализирует и назначает каждой некорневой Person-вершине вес: чем больше количество Group-вершин, с которыми и корневая вершина, и рассматриваемая Person-вершина

имеют связи типа TAKE_PART_IN, тем больше вес. После назначения весов повторяются все действия, но ранжирование происходит в порядке убывания суммы всех весов Person-вершин, которые имеют с рассматриваемой Group-вершиной связь типа TAKE_PART_IN. Проще говоря, рассматриваются сообщества, в которых состоят друзья пользователя, но не состоит сам пользователь, составляется рейтинг по количеству вхождений в сообщество друзей, но с учетом информации о том, сколько общих сообществ у пользователя с другом. Чем больше общих сообществ у пользователя и друга, тем больший вес в рейтинге имеют сообщества, в которых состоит этот друг.

После завершения поиска рекомендаций полученная информация выводится на интерфейс пользователя (рис. 3) в удобном для восприятия виде: результаты работы алгоритмов визуально отделены друг от друга, карточки, отображающие необходимую информацию, выводятся в порядке релевантности.

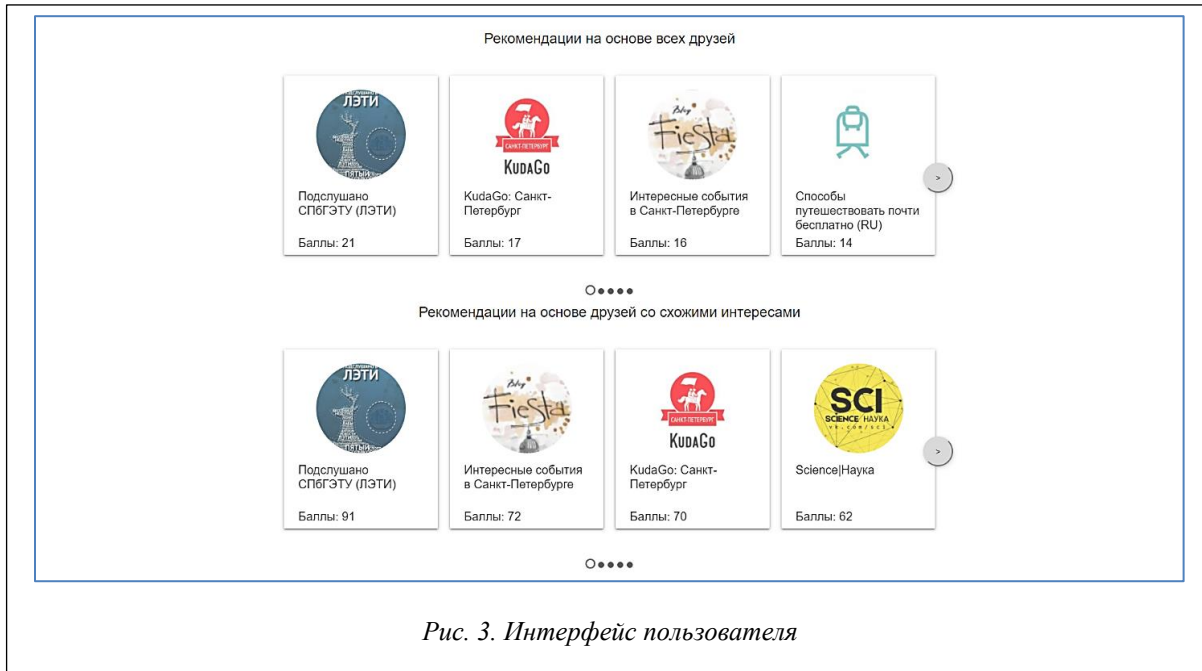


Рис. 3. Интерфейс пользователя

Главным минусом данного решения является время, необходимое на загрузку данных в базу: при больших объемах данных приложение не сможет отработать за приемлемое для пользователя время, несмотря на то, что сам алгоритм построения рекомендаций занимает всего несколько секунд. Для пользователя с небольшим количеством друзей время работы программы в целом занимает от полутора до трех минут. Под небольшим количеством подразумевается до 200 человек.

Тестирование приложения проводилось на компьютере ACER Core i5 3210M 2500 Mhz, память 4 Гб DDR3, SSD 500 Гб, Windows 10.

Заключение

В рамках создания рекомендательной системы для социальной сети ВКонтакте были рассмотрены наиболее распространенные подходы для поиска рекомендаций. Было выяснено, что большинство рассмотренных алгоритмов используют сложные математические вычисления и решают ряд побочных задач, что говорит о нецелесообразности их использования для решения поставленных задач.

Разработанное приложение [17] успешно решает задачу построения рекомендаций сообществ для пользователей социальной сети ВКонтакте. Используемый в рекомендательной системе алгоритм показал свою рентабельность, так как при простоте реализации он гарантирует минимальные временные затраты. В дальнейшем планируется оптимизировать загрузку данных из социальной сети ВКонтакте, чтобы максимально сократить время работы программы.

Литература

1. Ломаш Д.А., Хлопин К.В. Реализация рекомендательных систем на основе алгоритмов коллаборативной и контекстной фильтрации // Вестн. РГУПС. 2013. № 1. С. 75–84.
2. Last.fm // <https://www.last.fm/ru/>
3. PANDORA MUSIC // <https://www.pandora.com/>
4. IMDb // <http://www.imdb.com/>
5. Instagram // <https://www.instagram.com>
6. VK // <https://vk.com>
7. Хабрахабр // habrahabr URL: <https://habrahabr.ru/post/58309/> (дата обращения: 08.05.2018).
8. Авхадеев Б.Р., Воронова Л.И., Охупкина Е.П. Разработка рекомендательной системы на основе данных из профиля социальной сети «ВКонтакте» // Вестн. Нижневартковского гос. ун-та. 2014. № 3. С. 68–76.
9. Дзюба А.А. Рекомендации треков в социальных сетях: магистер. дис. СПб: СПбГУ, 2012.
10. Гажа К.В., Калмыков В.В., Митянина А.В. Альтернативная рекомендательная система для пользователей Steam // Науч. дискуссия. 2017. № 1. С. 78–93.
11. Noe4j // <https://neo4j.com/>
12. Spring Boot // <https://projects.spring.io/spring-boot/>
13. Angular 5 // <https://angular.io/>
14. Spring Security // <https://projects.spring.io/spring-security/>
15. OAuth // <https://oauth.net/>
16. VK API // <https://vk.com/dev/methods>
17. https://github.com/moevm/nosql-2017-social_network_recomendations