Учеба/Магистратура/1 курс 1 семестр/Иностранный язык/"geometry.cfg

Учеба/Магистратура/1 курс 1 семестр/Иностранный язык/"textcomp.cfg

Учеба/Магистратура/1 курс 1 семестр/Иностранный язык/"bblopts.cfg
Учеба/Магистратура/1 курс 1 семестр/Иностранный язык/"14pt.ldf
Учеба/Магистратура/1 курс 1 семестр/Иностранный язык/"russian.ldf
Учеба/Магистратура/1 курс 1 семестр/Иностранный язык/"russian.cfg

Учеба/Магистратура/1 курс 1 семестр/Иностранный язык/"epstopdf.cfg

1 Анатомия Reddit: Обзор академических исследований

1.1 Введение

Понимание динамики и структуры человеческого общения — центральная тема исследований в вычислительных общественных науках. Растущая доступность цифрового отслеживания человеческого взаимодействия на крупном масштабе данных позволила выявить множество разнообразных феноменов. Так, например, журналы телефонных звонков привели в выявлению «пакетности» человеческого общения, которая проявляется в коротких «периодах» интенсивного разговора, сопровождаемых длинными паузами; социальные сети и электронная почта подтвердили «тесность» мира, т.е. факт того, что среднее расстояние в сети взаимодействий между людьми непропорционально мало по сравнению с её размером; сообщения в микроблогах привели к открытию механизмов, приводящих к образованию информационных каскадов; и т.д. Если старые исследования изначально основывались на изучении взаимодействия двух субъектов, то появление новых каналов связи (микроблогов и онлайн-форумов) создало возможность изучения коллективных переговоров.

Коллективные переговоры не были изобретены современными средствами массовой информации. Например, вспомните демократические дебаты в Сенате Греции. Как таковые, они были и остаются основным способом обмена мнениями и принятия коллективных решений. Онлайн-форумы предоставляют место, где интернет-пользователи могут размещать вопросы или комментарии, которые могут провоцировать возникновение дискуссий с другими участниками сообщества. Понимание принципов взаимодействия людей на онлайн-форумах имеет важные теоретические последствия для понимания принципов коллективного мышления, а также может применяться на практике с целью создания положительного пользовательского опыта, увеличения вовлеченности и способствования демократическому процессу. Цель этой статьи — предоставить обзор академических исследований дискуссионных онлайн-платформ, или

онлайн-форумов, и собрать воедино разнообразные вопросы, рассматриваемые в литературе. В большей степени наше внимание будет обращено к так называемой «главной странице Интернета» — веб-сайту Reddit (REDDIT.COM) — который в настоящее время является крупнейшим онлайн-форумом в мире. Следует отметить, что многие дискуссионные онлайн-платформы схожи по своей архитектуре и тоже ранее изучались, например, в сравнительных исследованиях; к ним относятся Digg, Hacker News, Slashdot, Epinions, Meneame, Barrapunto и даже Википедия.

Остальная часть этой статьи организована следующим образом. Раздел 2 представляет наборы данных, которые могут быть извлечены из Reddit и были широко использованы исследователями. Затем академические исследования разделяются в зависимости от объекта их основного внимания (посты или пользователи). Они приведены в Разделах 3 и 4 соответственно. Статья завершается обсуждением результатов и перспектив для будущих исследований.

1.2 Наборы данных

Reddit был запущен в 2005 году и является агрегатором социальных новостей, рейтингом веб-контента и сайтом для ведения дискуссий, находящимся на 6-м месте в списке самых посещаемых веб-сайтов в мире с 234 миллионами уникальных пользователей (по состоянию на февраль 2018 года). Схематичная структура Reddit представлена на рис. 1. Зарегистрированные пользователи размещают посты, состоящие из заголовка и внешней ссылки или самостоятельного написанного контента, которые моментально становятся доступными для всей аудитории Reddit с возможностью оценки и комментирования. Система оценок допускает повышение (+1) или понижение (-1) рейтинга постов и комментариев только зарегистрированными пользователями. Комментарии формируют дерево обсуждений, в котором корневым узлом назначется сам пост, а все остальные узлы представляют собой комментарии. Если два узла формируют отношение «ответ на», то между ними присутствует связь.

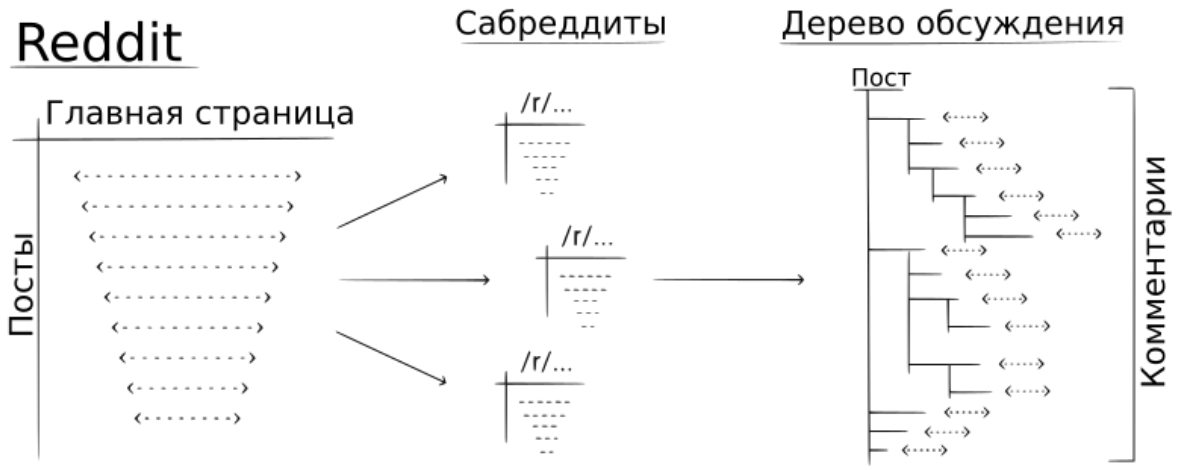

Рис. 1: Схематичная структура платформы Reddit. Точка входа — главная страница Reddit, обеспечиваемая постами из сабреддитов, на которые подписан зарегистрированный пользователь (или из всех сабреддитов, для анонимных пользователей) и располагает их в соответствии с голосами пользователей и возрастом поста. Пользователь может проследовать на страницу конкретного сабреддита, где лента сокращается до постов, относящихся только к выбранному сабреддиту. Каждый пост может быть положительно или отрицательно оценен и имеет прикреплённую секцию комментариев. Комментарии структурированы как дерево по отношению «ответ на» к другим комментариям или к самому посту.

Огромное информационное пространство Reddit поделено на сабреддиты — создаваемые пользователями сообщества, объединённые определённой темой. Название одного из сабреддитов является неотъемлемой частью каждого опубликованного поста. Каждый сабреддит и сам Reddit обладают так называемой «главной страницей» — лентой, где заголовки постов и ссылки для голосования и комментирования показываются пользователям. На порядок отображения постов влияют два фактора: 1) время и 2) рейтинг, иначе называемый «кармой» и представляющий собой разницу между положительными и отрицательными голосами. Посты с большим рейтингом имеют более высокий шанс попадания на главную страницу. Однако со временем новая информация заменяет в ленте старую. Пользователи могут отслеживать сабреддиты, но не других пользователей, что обозначает главное отличие от социальных сетей типа Facebook или Twitter, где пользователи подписываются на определённого человека, а не контент. Другие платформы имеют схожую структуру. Например, Slashdot (запущен в 1997 году) состоит из новостей вместе с комментариями, модерируемыми выбранными пользователями, а не от-

крытой системой голосования. В нём доступно только ограниченное количество тематических подразделов. Hacker News (запущен в 2007 году) сильно похожее на Reddit онлайн-сообщество, но только с двумя предварительно созданными «сабреддитами». Digg (запущен в 2004) в данный момент выступает в роли агрегатора новостей, однако раньше он был социально-курируемой платформой с возможностью публикации постов, комментирования и системой голосования, как на Reddit. Meneame — испанский аналог Digg, Barrapunto — тоже испанская версия Slashdot. И т.д.

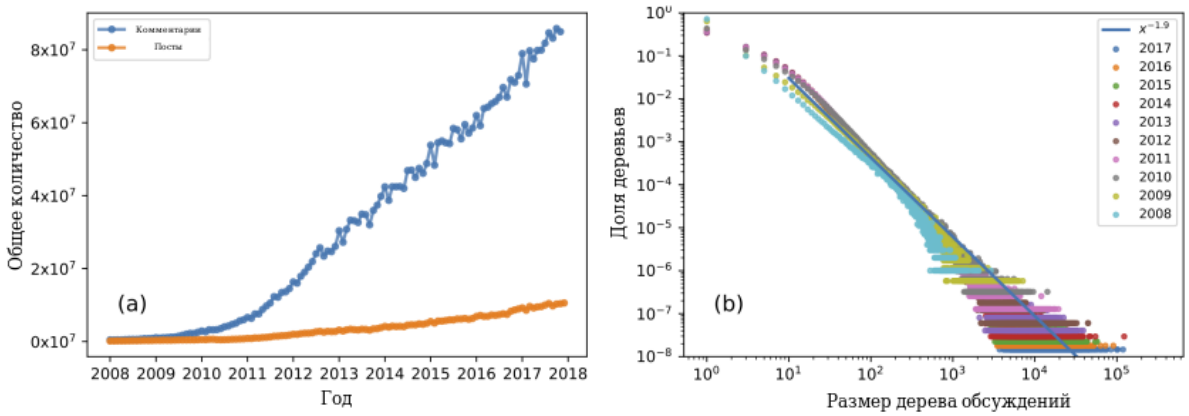

Рис. 2: Эволюция Reddit с января 2008 года по январь 2018 года: a) ежемесячное количество постов и комментариев, b) распределение объёмов дискуссий. Вы можете заметить экспоненциальное увеличение количественной активности, однако распределение объёмов дискуссий придерживается одинаковой формы, близкой к показательной функции со степенью α, находящейся в районе 1.7 для ранних лет и 1.9 для поздних.

Reddit занял центральное место в научной литературе благодаря открытости, богатству и качеству его данных, что позволяет проводить продольные исследования всей системы и, что самое важное, обеспечить воспроизводимость результатов. Джейсон Баумгартнер, под псевдонимом Stuck_In_The_Matrix, проделал огромный объём работы, когда попытался собрать полный набор постов и комментариев, опубликованных с момента создания сайта. Иллюстрации в этой главе были созданы на основе этого набора данных. Например, основные числа по росту сайта и общих размерах дискуссий можно найти в рис. 2. Его репозиторий данных также содержит данные с платформы Hacker News.

Несмотря на признанное качество, набор данных не является без-

упречным. Гаффни и Мэтиас сообщают о нескольких несоответствиях в данных. Например, данные по комментариям и постам до 2008 года сильно повреждены, 80% постов отсутствуют, также как и 90% данных о постах за один месяц между 2009 и 2010 годами. В общем, на интервале с января 2006 по февраль 2016, авторы сообщают о 0.043% потерянных комментариев и 0.65% потерянных постов. Риски присутствия некорректных данных очевидны, однако на крупномасштабных исследованиях они могут быть безопасно проигнорированы из-за их относительной незначительности. Система поддерживала экспоненциальный рост, поэтому объём данных за ранние годы пренебрежимо мал по сравнению с современными числами. Собираемые повторно данные должны быть свободны от ошибок, присутствующих в данных до 2008 года.

Хотя отсутствующие данные и вызывают разумные несоответствия, сообщается, что комментарии от авторов, чьи аккаунты были удалены, составляют около 25% от всех данных, что создаёт большие ограничения для исследований, направленных на пользователей. В таком случае имя автора заменяется на стандартное имя «[deleted]», но вся остальная информация об опубликованном поста или комментарии остаётся в системе.

2 Curriculum Vitae

Personal Details

Dmitry Karikh
13, Veshnyakovskaya ul., Moscow. Russia, 111539.
8 (915) 123-45-67 E-mail: karikh.d@gmail.com

29th August 1996
Russian
Single
Children (none)

Education

2014-2018	MIREA — Russian Technological University Bachelor's degree in Information Systems and Technologies
2003-2014	City Budget Educational Institution «Secondary School №1028»

Professional Experience

2016-2018	Laboratory assistant of the department of higher mathematics 2, MIREA — Russian Technological University.
2015-2016	Laboratory assistant of the department of higher mathematics 2, Moscow Technological University.

Skills

Programming languages	Python (Flask, Mako, etc.), Java (Android), JavaScript (Web), Bash, C (basic level).
Other technical languages	HTML, CSS, JSON, YAML.
Software	Debian-based Linux distributions, Docker, Rancher, Jenkins CI, SSH, Git (GitHub, Gitea, standalone), Huginn, Distributed monitoring (Netdata + InfluxDB + Grafana).
Hardware	Soldering, AVR microcontrollers, Raspberry Pi. Assembly and repair of personal computers and servers. Ethernet and Wi-Fi networks.
Languages	Russian: native English: intermediate (writing, reading); fluent (speaking)
Interests	Computer games, Online forums

3 Docker, Inc.

Hello everyone, my name is Dmitry Karikh. I am a master student at MIREA. Today, I will tell you about the Docker Incorporation.

First, I will give you some basic information about the company. Then, I will tell you about their products. And at the end I will mention the reason for their success.

So, starting out, the company was founded as dotCloud, Inc. in 2010 by Solomon Hykes. Later, in October 2013 it was renamed to Docker, Inc. In 2015 Docker, a privately held startup company, was estimated to be valued at over $1 billion, making it what is called a "unicorn company".

Moving on, let's talk about the company's products. Docker, Inc. mostly specializes on software development and automation of code deployment inside the software containers. Their main product is an open-source containerization project named Docker. Other products are also related to Docker: Swarm – an orchestration engine that allows to build clusters, Docker Hub – a platform for sharing of container images created by users or organizations, etc. There are two main editions of Docker: Community and Enterprise. First one is free to use for anyone, open-source and has more frequent releases. The Enterprise Edition costs from $750 to $3000 per year and is based on the Community Edition thus having the same functionality, plus there are options to use certified container images, reliable Docker Datacenter and get the official same day support service.

Finally a word about the success of Docker. The company has acquired many innovative startups including Orchard, SocketPlane, Kinematic, Tutum, Unikernel Systems, Conductant and Infinit. Due to these technologies and the availability of the source code, Docker has been able to attract the attention of many industry giants, including Red Hat, Cisco, Google, and Microsoft, thus becoming the industry standard of containerization.

I hope you will be interested to learn more about this company. Thank you for listening to my presentation.

4 Dockerization Impacts in Database Performance Benchmarking

4.1 Introduction

Benchmarking has long been used for the comparison of software and hardware systems or software versions. Though, when done right, benchmarking is surprisingly hard as conflicting goals such as reproducibility, portability, understandability, fairness, ease-of-use, and relevance need to be balanced. When focusing on reproducibility and ease-of-use, an engineer is likely to encounter two main challenges:

1. Correctly installing and configuring both benchmarking client and the system under test (SUT) can be challenging, or at least involves a lot of effort;

2. For reproducibility reasons, benchmark runs need to be repeated several times – preferably on a fresh system setup which aggravates the first challenge.

A solution that naturally lends itself to these challenges is to use containers as a convenient deployment mechanism for preconfigured, ready-to-use experimental setups. However, it is unclear whether this will affect benchmarking results. Most existing studies on this topic are measuring the overheads that various applications might incur when running inside containers instead of on bare metal or inside a virtual machine. These, however, all quantify the overhead that is induced by Docker for a certain workload or application. Neither of these studies measures indirect effects of Docker that, for instance, a database benchmark running against an SUT on another machine might experience. For such a benchmark, indirect effects might lead to volatile and unpredictable changes in benchmarking results rendering results at least partially obsolete.

This article aims to answer the question whether it is safe to dockerize database benchmarks, that is to say, whether dockerization of benchmarking client and/or SUT has observable effects on measurement results.

4.2 Experiments

The authors of this article have carefully designed a set of experiments that not only quantifies possible dockerization impacts on benchmarking results but also explores whether different standard settings of both benchmarking client and SUT can further influence potential impacts. These settings include configurations that put either I/O, CPU or memory under stress.

The main objects of study are YCSB (a standard cloud storage benchmark) and Apache Cassandra (a widely used NoSQL system) running on unified Amazon EC2 instances (virtual servers).

There are four dockerization variants in this case of database benchmarking:

1. None of the testing components are dockerized;
2. Either a benchmarking client or a SUT is dockerized;
3. Both components are dockerized.

Apache Cassandra has various configuration options that may affect the overall performance of the database. All combinations of the following options were tested.

1. One of the compaction strategies:
   (a) Size Tiered;
   (b) Time Window;
   (c) Level;
2. Key cache was either disabled or set to «auto» (=4% of the available heap size and up to a maximum of 100MB);

YCSB's Workload A consists of 50% read and 50% write operations against the database. For the m3.medium instances, 100k records and 3kk operations were used. m3.large instances used the same number of records and number of operations was increased to 9kk to achieve sufficiently long-running experiments. This configuration results in about 20 minutes per benchmark run for both instance types. Also the benchmarking client was configured to use a various number of simultaneous threads (10, 25, 50 on m3.medium instances and 30, 75, 150 on m3.large instances).

After the end of each benchmarking process, the virtual machine was automatically destroyed and recreated from the same prebuilt image to ensure a consistent environment. Docker images were also prebuilt for the same reason.

4.3 Results

A dockerized component introduces an additional layer of complexity into a system. Thus, from the beginning, authors expected a decrease in throughput of systems when using Docker containers and the lowest throughput in fully dockerized setups. As expected, they found such impacts on measurement results. In summary, their findings show that the dockerization of benchmarking components typically leads to increasing latencies and decreasing throughputs, especially for larger instance types. However, they also observed that in some cases Docker actually increased throughput.

4.4 Implications

The authors of this article have worded the next five implications based on the results:

1. Results of dockerized benchmarks can be acceptable when comparing different database systems.

2. Benchmark setups should be as close as possible to the production environment that they try to emulate.

3. When evaluating system configurations or implementation alternatives, it may be an option to dockerize the benchmark.

4. In many cases it may be acceptable to dockerize the benchmark as long as it stays dockerized and no configuration changes are made.

5. Repeating sufficiently long experiments is always important in benchmarking.

4.5 Conslusion

Based on their results, authors of the article derived a number of implications regarding their main question whether it is safe to dockerize database benchmarks: Overall, dockerization of benchmarking system and SUT is not a good idea. Dockerized results can and should only be used for a general ranking of SUTs as the concrete influence of Docker on measurement results depends on too many factors. However, this also implies that if the production environment is dockerized, then the corresponding benchmarking environment should be as well.

5 Course Book, p. 8, ex. F

1. In banking, you can **make a fortune** with the big bonuses and retire at 35.

2. When you **work flexitime**, you can arrange your own schedule, so this is very convenient when you have children.

3. People who work in sales often have the opportunity to **earn commission** on top of a basic salary.

4. Luke is ambitious and does not want to be a sales assistant all his life. In fact, he hopes to **get a promotion** annd become Assistant Manager very soon.

5. Many student **do part-time work** when they are at university because it fits in with wheir studies.

6. Goran is 59, but he does not want to **take early retirement**. In fact, he is taking on more work!