ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

УДК 004.05

ИСПОЛЬЗОВАНИЕ РАСПРЕДЕЛЕННОЙ ОБРАБОТКИ ДАННЫХ ПРИ РАЗРАБОТКЕ ПРИЛОЖЕНИЯ «АВТОМАТИЗИРОВАННАЯ СИСТЕМА КОММУНИКАЦИЙ СТУДЕНТОВ, АДМИНИСТРАЦИИ И ПОТЕНЦИАЛЬНЫХ РАБОТОДАТЕЛЕЙ»

В.И. БАКЛАН
(Представлено: канд. техн. наук, доц. И.Б. БУРАЧЁНОК)

Рассмотрены особенности фреймворка Spark, а также основные практики и подходы распределённой обработки большого количества данных на примере разработанного приложения «Автоматизированная система коммуникаций студентов, администрации и потенциальных работодателей». Показаны преимущества использования фреймворка Spark в составе интегрированной системы.

Сегодня проблема трудоустройства выпускников после окончания ВУЗов чрезвычайно актуальна, так как молодой специалист сталкивается с довольно жёсткими условиями рынка труда. Проблема трудоустройства выпускников имеет место и для нашего ВУЗа, поэтому создание автоматизированной системы коммуникаций студентов, администрации и потенциальных работодателей, позволяющей установить связь между ними является одной из актуальнейших задач нашего региона.

Для реализации автоматизированной системы коммуникаций студентов, администрации и потенциальных работодателей использовался большой стек современных, пользующихся популярностью среди разработчиков технологий: для реализации backend-части – Scala и Java; для frontend-части – TypeScript; для построения API – Spring Boot Framework на Java, Play Framework и Akka; для построения системы сообщений – Scala, React, Material UI; для хранения данных – СУБД PostgreSQL; для хранение кеша и в качестве брокера сообщений – Redis.

Система развёрнута на платформе Amazon Web Services. База данных размещается в RDS, доступ к Redis осуществляется через ElasticCache, развёртывание кластера производится при помощи ECS. Вся инфраструктура, включая конфигурации сетей, настройки и развёртывания сервисом написана кодом в файлах конфигураций и доступна к развёртыванию с использованием Cloud Formation. Для упаковки приложений в контейнеры для их последующей контейнеризации используется Docker и Docker-Compose.

Исходя из того, что разработанное приложение потенциально может генерировать огромные объёмы данных и мета-информацию, на основании которых можно отслеживать поведение пользователей, сервиса, расставлять приоритеты в выборе последующих подпроектов и анализировать тренды, которые развиваются в рамках сервиса, то такая система не имеет физических ограничений по нагрузке, данные могут генерироваться в любом объёме, следует более подробно остановиться на рассмотрении использования при разработке приложения распределённой обработки данных.

Термин Big Data появился сравнительно недавно. Google Trends показывает начало активного роста употребления словосочетания начиная с 2011 года. Большие данные (англ. big data) – серия подходов, инструментов и методов обработки структурированных и неструктурированных данных огромных объёмов и значительного многообразия для получения воспринимаемых человеком результатов, эффективных в условиях непрерывного прироста, распределения по многочисленным узлам вычислительной сети, сформировавшихся в конце 2000-х годов, альтернативных традиционным системам управления базами данных и решениям класса Business Intelligence. Таким образом, Big Data – это не какой-то конкретный объем данных и не сами данные, а методы их обработки, которые позволяют распределено их обрабатывать. Эти методы применимы не только к огромным массивам данных, но и к маленьким данным.

Существует три основных принципа обработки больших данных:

- горизонтальная масштабируемость;
- отказоустойчивость;
- локальность данных.

В качестве примера обработки большого количества данных можно рассмотреть ClickStream. Clickstream – это потом кликов пользователей, сгруппированный по идентификатору сессии. Так как на начальных этапах объём данных не огромный, обработка будет производиться на кластере с одной нодой и все кликстрим-события будут передаваться в аналитическую систему с помощью Apache Kafka.

Apache Kafka – диспетчер сообщений на Java платформе. В Kafka – есть тема сообщения, в которую издатели пишут сообщения и есть подписчики в темах, которые читают эти сообщения, все сообщения в процессе диспетчеризации пишутся на диск и не зависит от потребителей.

Непосредственным обработчиком будет являться приложение, написанное на языке Python с использованием фреймворка Spark.

Spark – это проект Apache, который позиционируется как инструмент для «молниеносных кластерных вычислений». Проект разрабатывается процветающим свободным сообществом, в настоящий момент является наиболее активным из проектов Apache.

Spark предоставляет быструю и универсальную платформу для обработки данных. По сравнению с Hadoop Spark ускоряет работу программ в памяти более чем в 100 раз, а на диске – более чем в 10 раз.

Spark был выбран в связи с растущим сообществом разработчиков, поддержкой множества систем и языков программирования, большинством встроенных элементов.

Задача аналитического сервиса состоит в генерации поведенческой «карты» пользователя, что в будущем позволит аналитикам и проект-менеджерам продумывать стратегию развития продукта на основании поведенческих данных пользователя в человеко-читаемом формате.

Для уменьшения нагрузки на систему, планируется использование Spark Streaming с Batch-режимом, который позволит обрабатывать поточные данные в небольших батчах, что позволит сократить единоразовую нагрузку и иметь возможность дозировать нагрузку на кластер.

Так как приложение изначально планируется использовать ограниченному количеству пользователей, то для аналитического кластера может использоваться минимальная конфигурация с одной master-нодой и одной worker-нодой, а также минимальным уровнем репликации данных в Kafka. В представленном примере аналитической обработки допускается потеря данных, так как клик является не чувствительной информацией, и потеря допустимой части данных не является критичной.

Таким образом, использование представленного стека технологий позволяет написать систему, которая позволит выполнять обработку любого количества данных, в зависимости от требований и финансовых ограничений. Для последующего увеличения количества данных или времени обработки достаточно расширить кластер, а не переписывать существующий код. Использования языка Python позволяет писать код, не задумываясь о тонкостях языка, что позволяет ускорить процесс разработки и последующей доработки.

ЛИТЕРАТУРА

1. Architectural Styles and the Design. Диссертация Roy Thomas Fielding. [Электронный ресурс]. / ics.uci.edu. – Режим доступа: https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation.pdf/. – Дата доступа: 20.09.2020.
2. Zaharia M. Spark: The definitive guide // Zaharia M, Chambers B. – O'Reilly Media, Inc, 2018. – 606 с.