ОРГАНИЗАЦИЯ ПРОЦЕССА РАСПРЕДЕЛЕННОЙ ОБРАБОТКИ БОЛЬШИХ ОБЪЁМОВ ДАННЫХ С ИСПОЛЬЗОВАНИЕМ КЛАСТЕРА APACHE HADOOP

Слисенко К. Ю., Сиротко С. И., Кириченко А. Н.
Кафедра информатики, Кафедра электронных вычислительных машин, Белорусский государственный университет информатики и радиоэлектороники
Минск, Республика Беларусь
E-mail: kslisenko@gmail.com, sergeyis@bsuir.by, ext_kir@outlook.com

В статье рассматривается проблема организации процесса распределенной обработки больших объёмов данных применительно к анализу логов серверов и приложений. Анализ логов позволяет выявить тенденции нагрузки и доступа к ресурсам системы и далее на основе этого определить факты сетевых атак и несанкционированного доступа. В рамках решения поставленной задачи были проанализированы средства обработки больших объёмов данных и выбран фреймворк Apache Hadoop, позволяющий запускать вычисления в кластере. Описан процесс организации обработки данных с применением данного фреймворка.

Введение

Потребность в обработке больших объемов данных возникает при решении как научно-исследовательских, так и практических задач, становясь порой нетривиальной пролемой. Примерами могут служить обработка данных множества экспериментов или анализ логов сетевых среверов и приложений.

Рассмотрим в качестве инструмента для решения подобных задач фреймворк Apache Hadoop. Данный фреймворк, содержит набор средств, которые позволяют организовать распределенный вычислительный процесс для решения широкого круга задач анализа данных и машинного обучения.

I. Проблемы при обработке больших объёмов данных

Удобным для исследования примером задачи обработки большого объема данных является анализ логов, потребность в котором возникает при администрировании, отладке, сопровождении программных систем. Возникающая при этом проблема большого объема входных может быть решена в том числе путем распределения обработки в кластере из множества машин. При организации данного процесса возникают следующие проблемы:

1. Масштабируемость вычислений: должна быть возможность использования одной и той же программы для анализа данных различного объема и выполнения вычислений как на одной машине, так и на кластере;
2. Отказаустойчивость: при отказе одной из машин не должны теряться исходные данные, потери промежуточных вычислений должны быть минимальными. Должно быть организовано автоматическое восстановление после сбоев;
3. Хранение больших объёмов данных: необходима высокая надёжность и скорость доступа. По возможности вычисления должны производиться локально к расположению данных. Передача данных по сети должна быть минимальной;
4. Кросплатформенность: программы обработки данных не должны быть привязаны к специализированному оборудованию. Должна быть возможность производить вычисления на различных платформах, в том числа и на кластере, состоящем из персональных компьютеров.

II. Принцип Map Reduce

Для анализа логов был выбран принцип Map-Reduce, лежащий в основе вычислений на фреймворке Apache Hadoop. Map-Reduce может быть применен для решения широкого круга задач: анализа изображений, вычислений на графах, машинного обучения и так далее.

Вычислительный процесс разбивается на две стадии: Map и Reduce. Для каждой стадии необходимо реализовать соответствующию программную функцию. На вход функциям приходят наборы пар ключ-значение, такая же структура данных ожидается на выходе. Каждая из функций описывает отображение входного набора в выходной.

На стадии Map происходит предварительная обработка входных данных , представленными в виде пар ключ-значение:

$$(k_1, v_1), (k_2, v_2), ..., (k_n, v_n)$$

Такой же формат имеют и выходные данные:

$$(k'_1, v'_1), (k'_2, v'_2), ..., (k'_n, v'_n)$$

На стадии Reduce происходит свёртка предварительно обработанных данных. Пары с одинаковым ключём комбинируются. Функция выполняет вычисления над набором скомбинированных значений, поступающих на вход:

$$(k'_1, [v_{11}, v_{12}, ..., v_{1n}]), ..., (k'_n, [v_{n1}, v_{n2}, ..., v_{nn}])$$

Выход стадии Reduce является результатом обработки данных:

$$(k'_1, v''_1), (k'_2, v''_2), ..., (k'_n, v''_n)$$

При выполнении вычислений входные данные разбиваются на фрагменты, для каждого такого фрагмента применяется функция. Обработка данных может быть распараллелена и распределена по разным машинам, программный код может быть выполнен на машине, содержащий данные: в результате процесс вычислений становится линейно-масштабируемым, что актуально для анализа логов серверов и приложений.

Функции Map и Reduce не описывают процесс считывания и записи данных и логики координирования процесса вычислений, не зависят от платформы исполнения. Релазованные функции могут быть применены для обработки данных как малого, так и большого объёма.

Для анализа логов принцип Map-Reduce может быть применён следующим образом: функция Map принимает на вход одну запись лога и выделяет ip-адрес источника и данные о запрашиваемом ресурсе. На выходе функции Map ключом будет ip-адрес, значением – данные о ресурсе. На вход функции Reduce поступят данные о ресурсах, сгруппированные по ip-адресам. На выходе функции Reduce ключём будет ip-адрес, значением – данные о ресурсе, количество запросов к которому было максимальным. Таким образом вычисляются с каких ip-адресов какие ресурсы запрашиваются больше всего.

III. Обзор фреймворка Apache Hadoop

Выбранный для анализа логов фреймворк Apache Hadoop позволяет обрабатывать больше объёмы данных и организовывать распределённые вычисления. Для хранения данных используется распределенная файловая система HDFS, для организации вычислений – описанный выше принцип Map-Reduce. Фреймворк выполняет разбивку всего процесса на небольшие задания. Каждое такое задание содержит входные данные, конфигурацию и программу для выполнения. Задания, в свою очередь, разделяется на подзадачи Map и Reduce.

Основными компонентами вычислительного процесса являются координатор вычислений и исполнители. Координатор выполняет разбивку на задания, исполнители выполняют вычисления. Каждое задание оперирует с небольшим фрагментом данных. В случае сбоя координатор отправляет задание другому исполнителю, теряется только небольшая часть промежуточных вычислений. Разработчику функций Map и Reduce не требуется закладывать логику координации процесса и восстановления в случае сбоев. Промежуточные результаты вычислений автоматически сохраняются в файловой системе.

Одной из особенностью фреймворка является использование распределенной файловой системы HDFS, которая оптимизирована под хранение файлов большого размера и позволяет реплицировать данные между несколькими машинами.

Apache Hadoop позволяет выполняет масштабирование вычислений и предоставляет средства для организации отказаустойчивости. Для проведения вычислений не требуется наличие специализированного оборудования. Немаловажным преимуществом Apache Hadoop является возможность работы в кластере.

IV. Организация сложных процессов вычислений

Процесс анализа логов может состоять из множества этапов. Для организации такого процесса недостаточно пары Map-Reduce функций. При организации подобных вычислений возникают следующие трудности:

1. Последующие этапы вычислительного процесса могут зависеть от результата выполнения предыдущих этапов;
2. Вычисления могут выполняться с использованием разных средств, не только Map-Reduce программ;
3. Процесс получения исходных данных может быть частью самого вычислительного процесса. Очередная порция логов для анализа может поступать по мере накопления определённого объёма данных либо, к примеру, по уведомлению о нарушении безопасности;
4. Возникает необходимость производить вычисления периодически по заданному графику. Для анализа логов необходимо иметь отчёты за различные периоды времени.

В качестве объекта исследования подходов к организации вычислений и обработки данных был выбран фреймворк Apache Oozie, интегрированный с Apache Hadoop. Процесс вычислений описывается в виде направленного ациклического графа. Фреймворк предоставляет возможность описывать зависимости вычислительных задач друг от друга, запуска вычислений по графику и по условию, уведомлений по завершению операции. Эти возможности актуальны как для задачи анализа серверных логов, так и для других задач анализа данных и машинного обучения.

V. Список литературы

1. White, T. Hadoop the definitive guide, Third edition / T. White // O'Reilly Media, 2012. – 656 p.
2. Gillick, D. MapReduce: Distributed Computing for Machine Learning / D. Gillick, A. Faria, J. DeNero // Berkeley University, CS262A. – 2006
3. Cheng-Tao C. Map-Reduce for Machine Learning on Multicore / C. Cheng-Tao, K. Sang Kyun, L. Yi-An // Stanford University, Stanford CA 94305-9025. – 2008