

MapReduce

Шугаев Ильнур
VK.com
ilnur.shug@gmail.com

январь 2020 г.

Пример задачи

Поиск по логам

- Дневной лог событий рекламы в VK весит примерно 70 GB
- Хотим найти все вхождения события в течении года
- Нужно обработать 25.5 TB
- Чтение данных с диска со скоростью 100 MB/s займет примерно 70 часов

Пример задачи

Поиск по логам. Решение

- Специализированный код под каждую задачу

Map & Reduce

Типы

Большое количество программ можно выразить в виде последовательности функций map и reduce:

$$\begin{array}{lll} \text{map} & (k_1, v_1) & \rightarrow \text{list}(k_2, v_2) \\ \text{reduce} & (k_2, \text{list}(v_2)) & \rightarrow \text{list}(v_2) \end{array}$$

Данные функции можно легко распараллеливать

Примеры

WordCount

Map Для каждого слова в документе выдает $\langle \text{word}, 1 \rangle$

Reduce Суммирует счетчики для каждого слова

Примеры

WordCount. Pseudo code

```

1 map(String key, String value):
2     // key: document name
3     // value: document contents for each word w in value:
4     EmitIntermediate(w, "1");
5
6 reduce(String key, Iterator values):
7     // key: a word
8     // values: a list of counts
9     int result = 0;
10    for each v in values:
11        result += ParseInt(v);
12    Emit(AsString(result));
    
```

Примеры

Distributed Grep

Map Ищет строки подходящие под паттерн

Reduce Id function

Примеры

Inverted Index

Map Обрабатывает документы и выдает последовательность вида `⟨word, document ID⟩`

Reduce Для каждого слова выдает пару `⟨word, list(document ID)⟩`

Примеры

Distributed Sort

Map Из каждой записи извлекается ключ и выдается пара $\langle \text{key}, \text{record} \rangle$

Reduce Id function

Замечание

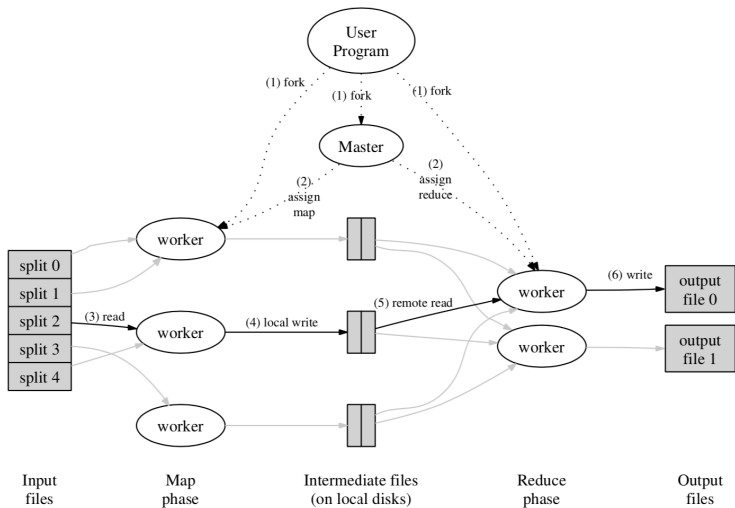
Корректность результата зависит от 1 (Partitioning function) и 2 (Ordering)

Цели MapReduce

Система должна взять на себя основную заботу о:

- 1 Параллелизме
- 2 Отказоустойчивости
- 3 Распределении данных
- 4 Распределении ресурсов и планировании

Схема работы



Распределение задач

- M - количество map'ов, R - количество reduce'ов
- Partitioning Function example — $hash(key) \bmod R$

Падение воркеров

- При падении Map-воркера перезапускается map задача и все reduce задачи, которые зависят от данных упавшего воркера
- При падении Reduce-воркера перезапускается только reduce задача

Падение мастера

При падении мастера прерывается весь процесс MapReduce вычислений

Код к данным

Хотим минимизировать обмен данными по сети.

Решение:

- 1 Мастер знает о том, где расположены данные, на которых нужно запустить тар задачи
- 2 Если это возможно, то следует запускать тар задачу на машине, на которой находятся нужные данные
- 3 В противном случае, мастер выбирает воркера, который ближе всего к данным (с точки зрения топологии сети)

Статусы воркеров

Мастер периодически спрашивает всех воркеров для уточнения статуса запущенных на них задач. Задача может находиться в одном из следующих состояний:

- idle
- in-progress
- completed

Замечание

Если воркер не отвечает в течении заданного времени, то все задачи воркера переходят в состояние idle

Backup tasks

Straggler

Определение

Straggler — воркер, которому необходимо сильно больше времени для завершения его задач в сравнении с остальными воркерами

Причины

- Медленное железо
- Данные слишком далеко
- ...

Backup tasks

Straggler. Решение проблемы

- Запуск дополнительных дублирующих задач, для задач, которые в состоянии in-progress, в ситуации, когда большинство задач уже в статусе completed
- Задача считается завершенной, когда она сама или дублирующая успешно завершаются

Partitioning functions

- Функция $hash(key) \bmod R$ создает слишком большие ограничения
- Пользователь может определить функцию партицирования самостоятельно

Ordering guarantees

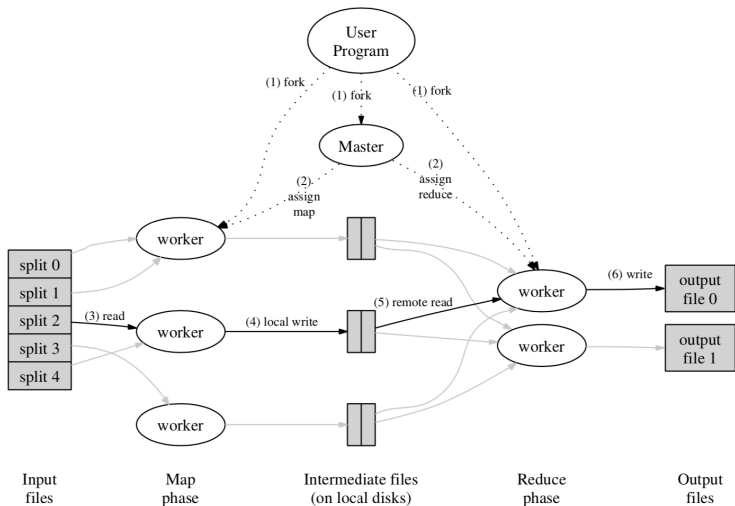
- 1 В результате выполнения map задачи, на локальном диске воркера образуется R партиций данных
- 2 Гарантируется, что данные на вход reduce задаче будут передаваться в порядке неубывания ключей

Замечание

Если $reduce = Id$ function, то после завершения всех reduce задач, мы получаем частично упорядоченный файл из R партиций, где данные внутри каждой партиции отсортированы

Ordering guarantees

Общая схема



Ordering guarantees

Distributed Sort

Несколько вариантов

- Использовать только один reduce воркер ($R = 1$)
- Использовать свою Partitioning function, которая разбивает данные на партии по ключу, а не по хэшу от ключа

Combiner function

Проблема. Word count

- Частоты слов распределены по Zipf Law¹
- Небольшое количество reduce задач получают большие списки на вход

¹https://en.wikipedia.org/wiki/Zipf%27s_law

Combiner function

Решение. Word count

Пользователь может задать Combiner функцию, которая на стороне map воркера будет делать предпосчет

Skipping bad records

Проблема

Если в данных есть записи, к которым код map/reduce функций не готов, то весь процесс завершится с ошибкой

Skipping bad records

Решение

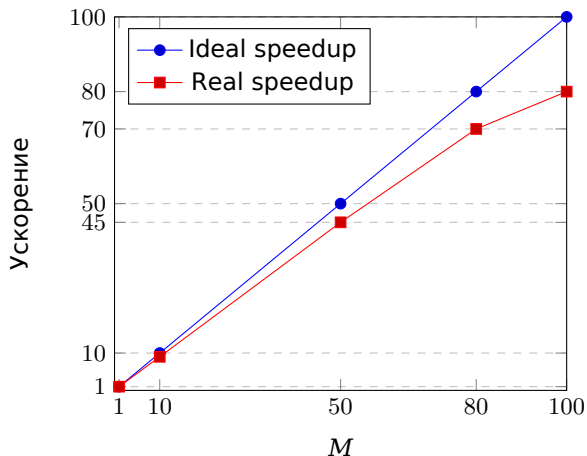
- 1 Можно попросить MapReduce запоминать записи, на который детерминировано появляются ошибки
- 2 Когда задача будет перезапущена она может успешно пропустить проблемные записи






Counters

Пример

Поиск по логам

Ускорение в зависимости от числа Map воркеров



-  Chu, Cheng-Tao и др. “Map-reduce for machine learning on multicore”. В: *Advances in neural information processing systems*. 2007, с. 281—288.
-  Dean, Jeffrey и Sanjay Ghemawat. “MapReduce: simplified data processing on large clusters”. В: *Communications of the ACM* 51.1 (2008), с. 107—113.
-  Lin, Jimmy. “Mapreduce is good enough? if all you have is a hammer, throw away everything that’s not a nail!” В: *Big Data* 1.1 (2013), с. 28—37.
-  Lin, Jimmy и Chris Dyer. “Data-intensive text processing with MapReduce”. В: *Synthesis Lectures on Human Language Technologies* 3.1 (2010), с. 1—177.
-  White, Tom. *Hadoop: The definitive guide*. ” O’Reilly Media, Inc.”, 2012.