COMP9313: Big Data Management

Introduction to MapReduce and Spark

动机 计算单元

Motivation of MapReduce

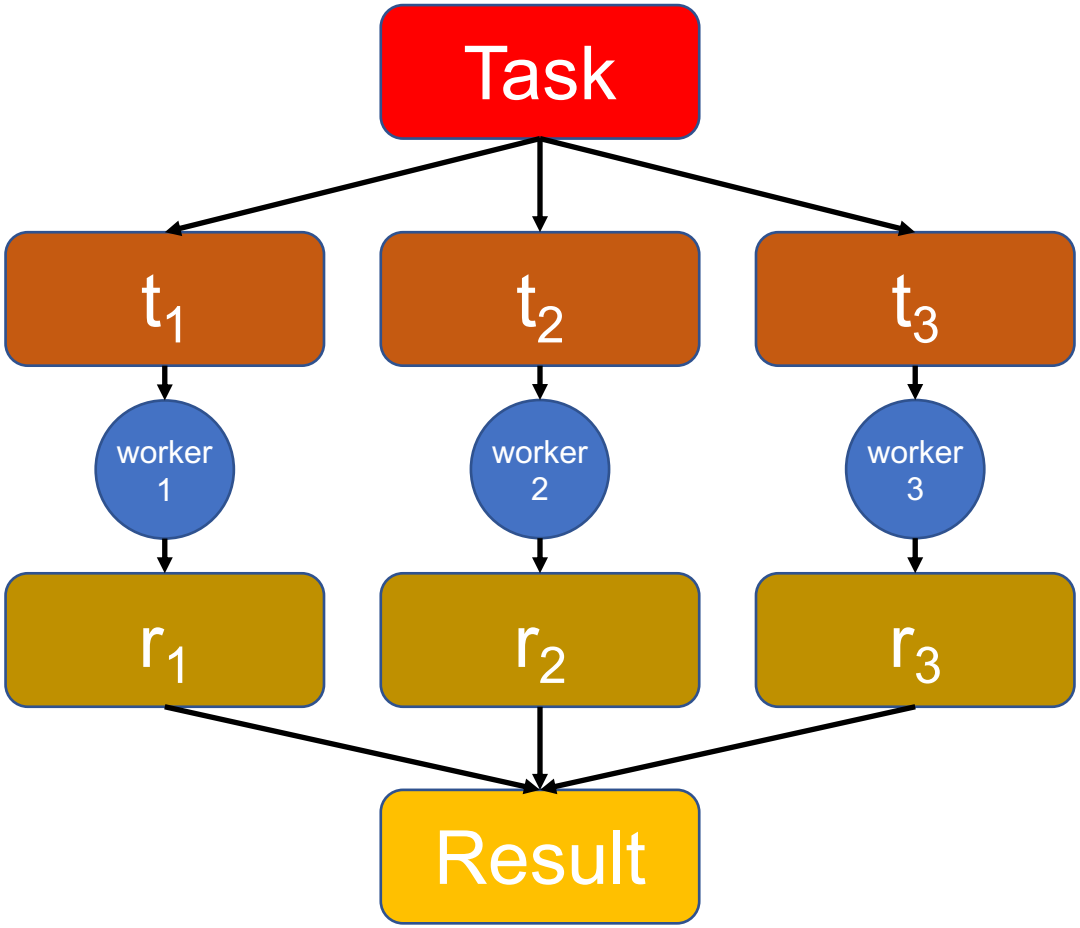
- Word count
  - output the number of occurrence for each word in the dataset.
- Naïve solution: 天真的解决方案

```
word_count(D):
    H = new dict
    For each w in D:
        H[w] += 1
    For each w in H:
        print (w, H[w])
```

- How to speed up?
  如何加快

Motivation of MapReduce

- Make use of multiple workers



There are some problems…

- Data reliability 数据可靠度
- Equal split of data 等分数据
- Delay of worker
- Failure of worker
- Aggregation the result 聚合结果

- We need to handle them all! In traditional way of parallel and distributed processing.
  传统的并行和分布式处理方式

MapReduce

编程框架

- MapReduce is a programming framework that

使我们能够在分布式环境中对大型数据集执行分布式和并行处理

  - allows us to perform distributed and parallel processing on large data sets in a distributed environment

无需担心可靠性，容错性等问题

  - no need to bother about the issues like reliability, fault tolerance etc

提供了编写代码逻辑的灵活性，而无需关心系统的设计问题

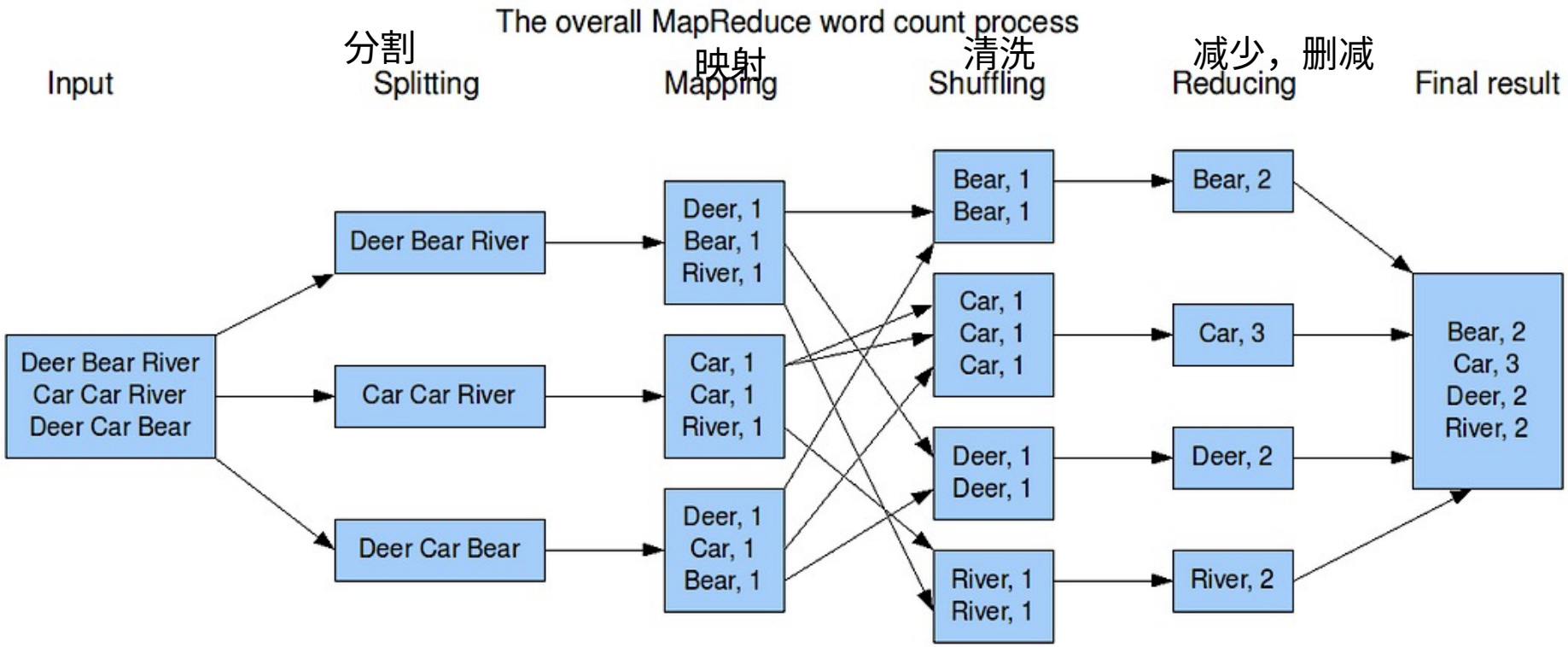
  - offers the flexibility to write code logic without caring about the design issues of the system

MapReduce是面向大数据并行处理的计算模型、框架和平台，它隐含了以下三层含义：
1）MapReduce是一个基于集群的高性能并行计算平台（Cluster Infrastructure）。它允许用市场上普通的商用服务器构成一个包含数十、数百至数千个节点的分布和并行计算集群。
2）MapReduce是一个并行计算与运行软件框架（Software Framework）。它提供了一个庞大但设计精良的并行计算软件框架，能自动完成计算任务的并行化处理，自动划分计算数据和计算任务，在集群节点上自动分配和执行任务以及收集计算结果，将数据分布存储、数据通信、容错处理等并行计算涉及到的很多系统底层的复杂细节交由系统负责处理，大大减少了软件开发人员的负担。
3）MapReduce是一个并行程序设计模型与方法（Programming Model & Methodology）。它借助于函数式程序设计语言Lisp的设计思想，提供了一种简便的并行程序设计方法，用Map和Reduce两个函数编程实现基本的并行计算任务，提供了抽象的操作和并行编程接口，以简单方便地完成大规模数据的编程和计算处理

Map Reduce

- MapReduce consists of Map and Reduce
- Map
  - Reads a block of data 读取数据块
  - Produces key-value pairs as intermediate outputs 产生键值对作为中间输出
- Reduce
  - Receive key-value pairs from multiple map jobs 从多个映射作业接收键-值对
  - aggregates the intermediate data tuples to the final output 将中间数据元组聚合到最终输出

A Simple MapReduce Example



Pseudo Code of Word Count

```
Map(D):
        for each w in D:
                emit(w, 1)
                        key value pair


Reduce(t, counts): # e.g., bear, [1, 1]
        sum = 0
        for c in counts:
                sum = sum + c
        emit (t, sum)
                key value pair
```

Advantages of MapReduce

- Parallel processing 并行处理
  - Jobs are divided to multiple nodes
  - Nodes work simultaneously 节点同时工作
  - Processing time reduced

- Data locality 数据局部性
  - Moving processing to the data 处理 移到数据上
    - Opposite from traditional way 与传统方式相反

We will discuss more on MapReduce, but not now…

Motivation of Spark

简化

- MapReduce greatly simplified big data analysis on large, unreliable clusters. It is great at one-pass computation. 一站式计算非常出色

不可靠的集群

- But as soon as it got popular, users wanted more:

更复杂的，多道分析

  - more **complex**, multi-pass analytics (e.g. ML, graph)

更多互动式临时查询

  - more **interactive** ad-hoc queries
  - more **real-time** stream processing

更实时的流处理

Limitations of MapReduce

- As a general programming model:
  - more suitable for one-pass computation on a large dataset
  - hard to compose and nest multiple operations
  - no means of expressing iterative operations
- As implemented in Hadoop
  - all datasets are read from disk, then stored back on to disk
  - all data is (usually) triple-replicated for reliability

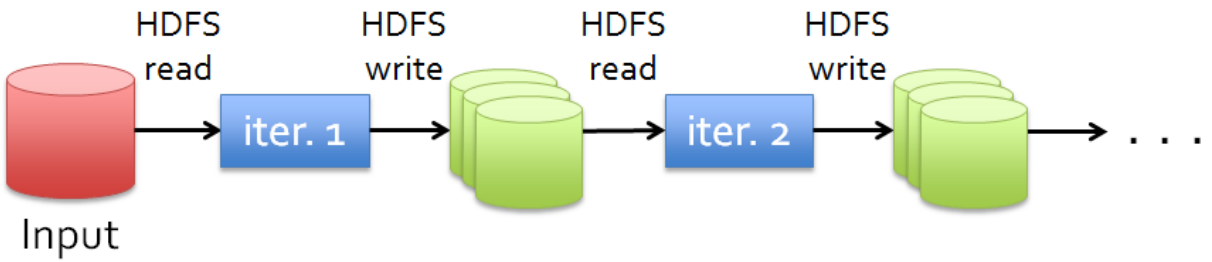
更适合对大型数据集进行一遍计算

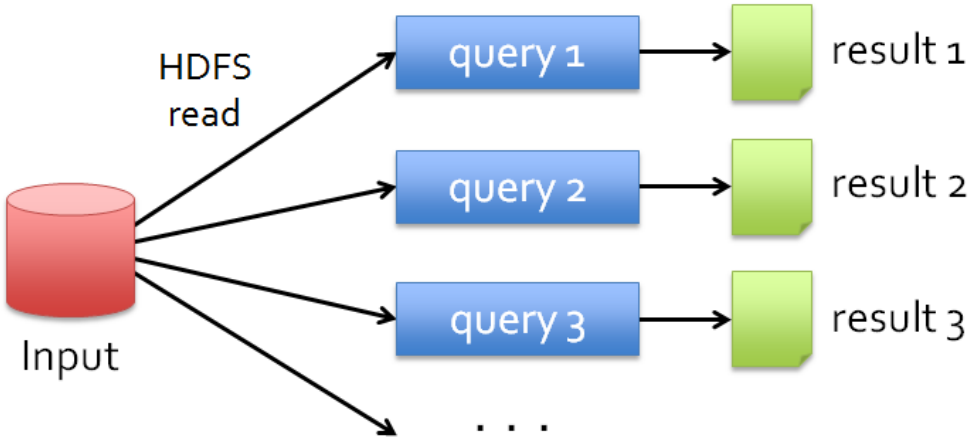
难以编写和嵌套多个操作

无法表达迭代运算

为了可靠性，所有数据(通常)都进行了三次复制

Data Sharing in Hadoop MapReduce





由于复制、序列化和磁盘IO而导致速度慢

- **Slow** due to replication, serialization, and disk IO

复杂的应用程序、流媒体和交互式查询都需要MapReduce所缺少的东西

- Complex apps, streaming, and interactive queries all need one thing that MapReduce lacks:
  - Efficient primitives for **data sharing**

用于数据共享的高效基元

What is Spark?

Apache Spark是一个用于实时处理的开源集群计算框架

- Apache Spark is an open-source cluster computing framework for real-time processing.

Spark提供了用于对整个集群进行编程的接口

- Spark provides an interface for programming entire clusters with
  - implicit data parallelism 隐式数据并行
  - fault-tolerance 容错性；容错能力
- Built on top of Hadoop MapReduce
  - extends the MapReduce model to efficiently use more types of computations

Spark Features

- Polyglot 多语言
- Speed
- Multiple formats 多种格式
- Lazy evaluation 惰性求值
- Real time computation
- Hadoop integration 集成
- Machine learning 机器学习

Spark Eco-System

结构

Spark Architecture

- Master Node 负责集群内的作业执行
  - takes care of the job execution within the cluster
- Cluster Manager 集群管理器
  - allocates resources across applications 跨应用程序分配资源
- Worker Node
  - executes the tasks 执行任务