MapReduce 简介

分而治之

将计算任务划分成互相没有依赖的部分

任务划分-任务合并

Map 与 Reduce

借鉴了函数式语言：map-将一个操作映射到一组数据上，reduce-结果整理

为这两种主要操作进行抽象处理

Map：将一个键值对 `(k1, v1)` 映射到一组键值对 `[(k2, v2)]`

Reduce：将一组键相同的键值对 `(k2,[v2])` 映射到另一组键值对 `[(k3, v3)]`

基于 Map 和 Reduce 的并行计算模型：

- 各个 map 对划分的数据并行处理并产生不同结果
- 各个 reduce 也各自并行计算，各自负责处理不同的中间结果数据
- reduce 之前必须等待所有 map 处理完，需要同步，对中间结果进行收集整理
- 汇总所有 reduce 输出结果可以得到最终结果

MapReduce 构架

MapReduce 提供了一个统一的计算框架来

- 划分和调度任务
- 划分和分布存储数据
- 处理同步
- 收集整理结果
- 系统通信，负载均衡
- 处理出错情况

程序员只用关心具体的业务逻辑

MapReduce 主要设计思想

横向扩展：采用便宜易扩展的低端商用服务器

失效是常态：用的是便宜商用服务器，但是系统容错性强

处理向数据迁移：数据/代码互相定位，发挥数据本地化的优点

顺序处理，避免随机访问：所有计算被组织成流式操作

为开发者隐藏系统细节：将程序员和系统细节隔离开

可扩展性：数据层面和系统层面