# 论文观后感

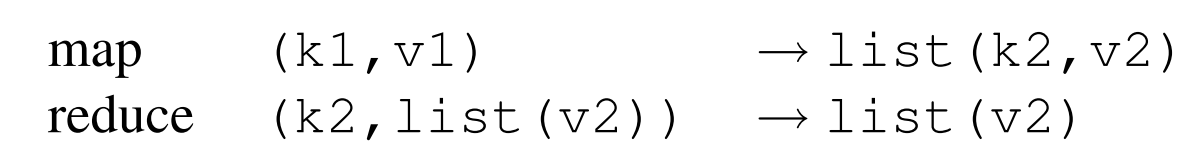
庞双双

## 1. MapReduce: Simplified Data Processing on Large Clusters

MapReduce是Google提出的一个软件架构，用于大规模数据集的并行运算。

MapReduce是一个编程范式，旨在使用map把大规模的问题分解成子问题，然后利用reduce把子问题的解汇总。这种编程范式特别适合应用于分布式系统。

要理解map和reduce的操作，最重要是要理解下式：



k1和v1是原始的输入key和value；

list(k2, v2)是map把k1和v1分布式计算后的中间结果集合；

reduce(k2, list(v2))是reduce函数根据k2的值来合并v2；

最终我们想要得到的结果是list(v2)。

MapReduce编程模型在Google内部成功应用于多个领域。成功在于几个方面：

首先，由于MapReduce封装了并行处理、容错处理、数据本地化优化、负载均衡等等技术难点的细节，这使得MapReduce库易于使用。即便对于完全没有并行或者分布式系统开发经验的程序员而言；

其次，大量不同类型的问题都可以通过MapReduce简单的解决。比如，MapReduce用于生成Google的网络搜索服务所需要的数据、用来排序、用来数据挖掘、用于机器学习，以及很多其它的系统；

第三，设计者实现了一个在数千台计算机组成的大型集群上灵活部署运行的MapReduce。这个实现使得有效利用这些丰富的计算资源变得非常简单，因此也适合用来解决Google遇到的其他很多需要大量计算的问题。

## 2. The Google File System

Google File System (GFS) 是由 Google 设计并实现的、一个面向大规模数据密集型应用的分布式文件系统，它不仅满足所有分布式文件系统共有的 高性能 、伸缩性 、可靠性 、可用性 ，还以 Google 自身 应用程序 和 技术环境 为基础进行了特有的设计 ，主要包括：

因为 GFS 使用设备数多，将组件失效视为常态事件。因此 持续监控 、错误检测 、灾难冗余 和 自动恢复 的机制必须包括在 GFS 中 。

主要对大文件的管理进行了优化 。

主要应用于 对文件尾部追加数据 的修改，而非覆盖原有数据的方式，如 “生产者-消费者” 队列，或者其他多路文件合并操作 。一旦写完后，对文件的操作通常是顺序读 。

采用了较弱的 一致性 要求，引用 原子性的记录追加 操作，保证多个客户端能够同时进行追加操作，不需要额外的同步操作来保证数据的一致性 。

目标程序绝大多数要求高速率、大批量地处理数据，极少要求对单一读写操作有严格的响应时间要求 。即 高性能的稳定网络带宽 比 低延迟 更重要 。

优点：master 和 chunkserver 的设计，将文件管理和文件存储分离；将文件分割成 chunk 存储，可并发访问，吞吐量较大；修改数据时控制流和数据流分离，充分利用每台机器的带宽；使用 lease 降低 master 工作负载，防止 split-brain 问题；对文件追加和顺序读的功能有优化；好的容错性。

缺点：只有一个 master ，元数据过多的话可能内存不够用；client 量很大的话，一个 master 负载过大；master 不能出错自动重启，出故障后人工切换 master 比较耗时；master 通过浏览所有的 chunk 进行垃圾回收效率太低；不擅长处理随机写问题、海量小文件存储；一致性过松，无法处理对一致性要求高的任务；GFS 被设计用于运行在单个数据中心的系统。

## 3. FastSGG: Efficient Social Graph Generation Using a Degree Distribution Generation Model

当今时代一个高效且可配置的社交图生成器空前重要；问题来源于生；要充分利用可取资源协助工作进行；论文要有所侧重，讲什么就专讲什么，严谨、克制；感觉论文编写者的水平很高；我的能力非常弱，以至于读论文都尤为困难。