INT303 W8

LARGE-SCALE COMPUTING

在进行 large-scale computing 时，往往需要分布式计算（distribute computation）。分布式计算会用到大量处理器，处理器有故障的可能，这意味着如果处理器数量足够多，每天都有很多处理器故障。

之前的做法是把数据集中到一台处理器上进行运算，但拷贝数据需要时间，因此一些想法是：bring computation to data （直接在数据所在的处理器上运算）；store files multiple times for reliability（把文件多备份几个，以防万一）。

Spark/Hadoop 解决了上述的问题：

- **Storage Infrastructure: File system**
- Programming model: MapReduce, Spark

STORAGE INFRASTRUCTURE

如果某个处理器故障，how to store data persistently?

- **Distributed File System**
  - Provides global file namespace
- **Typical usage pattern**:
  - Huge files (100s of GB to TB)
  - Data is rarely updated in place
  - Reads and appends are common

DISTRIBUTED FILE SYSTEM

Chunk servers

- File is split into contiguous chunks (文件被分为连续的块)
- Typically each chunk is 16-64MB
- Each chunk replicated (usually 2x or 3x) (每块备份2到3份)
- Try to keep replicas in different racks (把副本保存在不同的机架上)

Master node

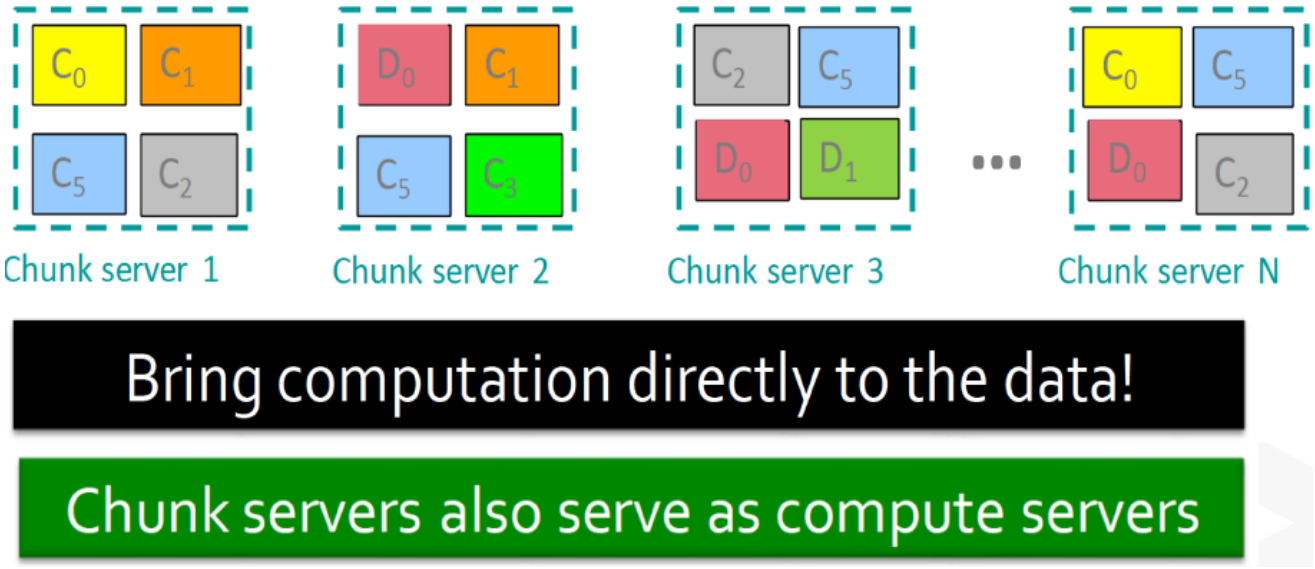
- a.k.a. Name Node in Hadoop's HDFS (HDFS, Hadoop distributed file system)
- Stores metadata about where files are stored
- Might be replicated

Client library for file access

- Talks to master to find chunk servers
- Connects directly to chunk servers to access data

Reliable distributed file system

- Data kept in "chunks" spread across machines
- Each chunk replicated on different machines
- Seamless recovery from disk or machine failure (从磁盘或机器故障中无缝恢复)



Mapreduce: DISTRIBUTED COMPUTING PROGRAMMING MODEL

MapReduce is a style of programming designed for:

- Easy parallel programming
- Invisible management of hardware and software failures
- Easy management of very-large-scale data

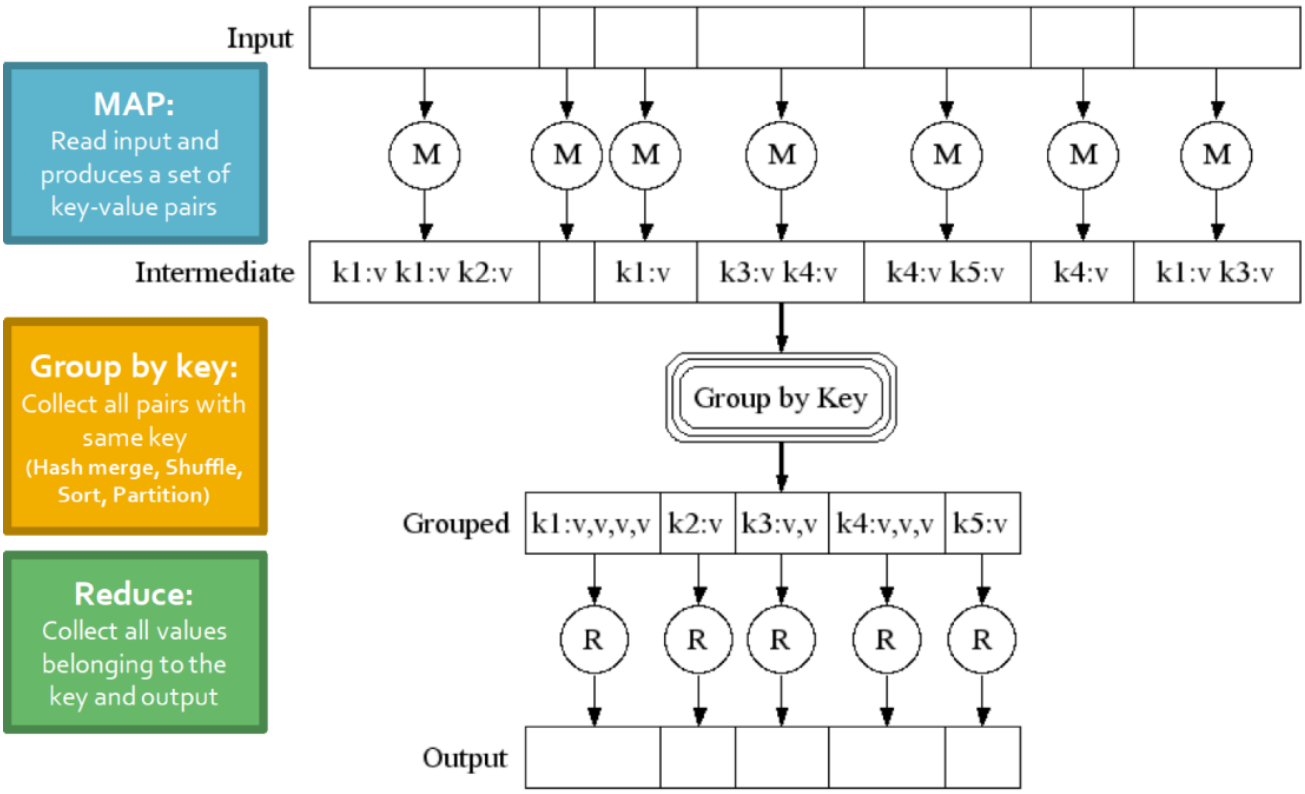
它有几个实现，包括 Hadoop，**Spark**，Flink，和谷歌原本的 MapReduce。

MapReduce

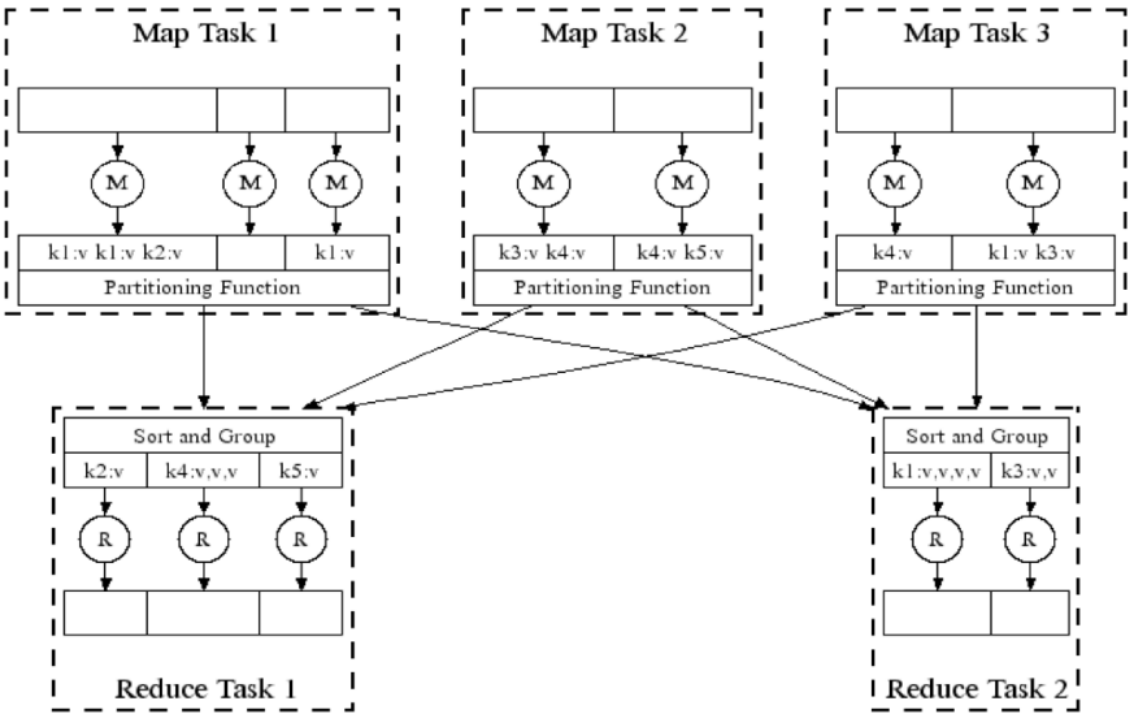
3 steps of MapReduce:

- Map:
    - Apply a user-written ***Map function*** to each input element
        - ***Mapper*** applies the Map function to a single element

        - Many mappers grouped in a ***Map task*** (the unit of parallelism)
    - The output of the Map function is a set of 0, 1, or more key-value pairs.
- Group by key: Sort and shuffle
    - System sorts all the key-value pairs by key, andoutputs key-(list of values) pairs
- Reduce:
    - User-written Reduce functions applied to each key-(list of values)



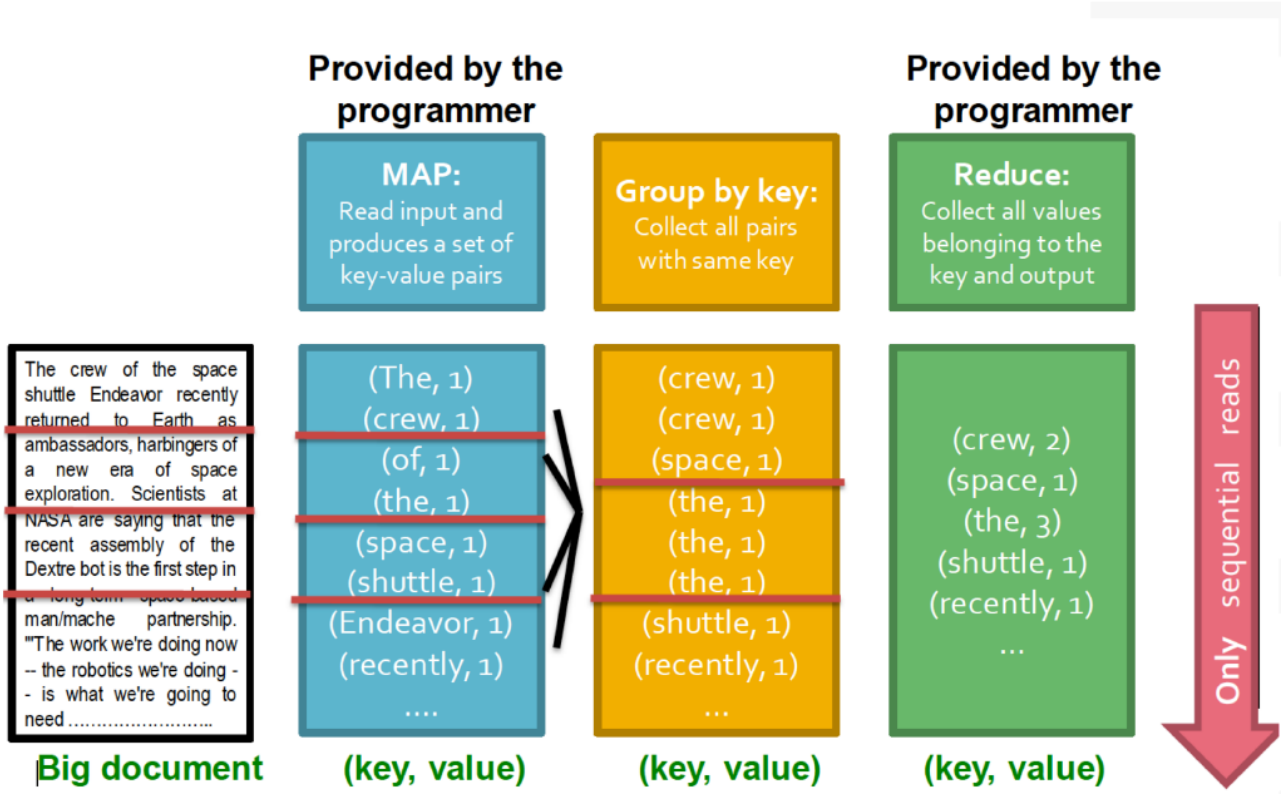
MapReduce: In Parallel



All phases are distributed with many tasks doing the work

- Each mapper/reducer must generate the same number of output key/value pairs as it receives on the input. (**Wrong**)
- The output type of keys/values of mappers/reducers must be of the same type as their input. (**Wrong**)
- The inputs to reducers are grouped by key. (**True**)
- It is possible to start reducers while some mappers are still running. (**Wrong**)

EXAMPLE: WORD COUNTING



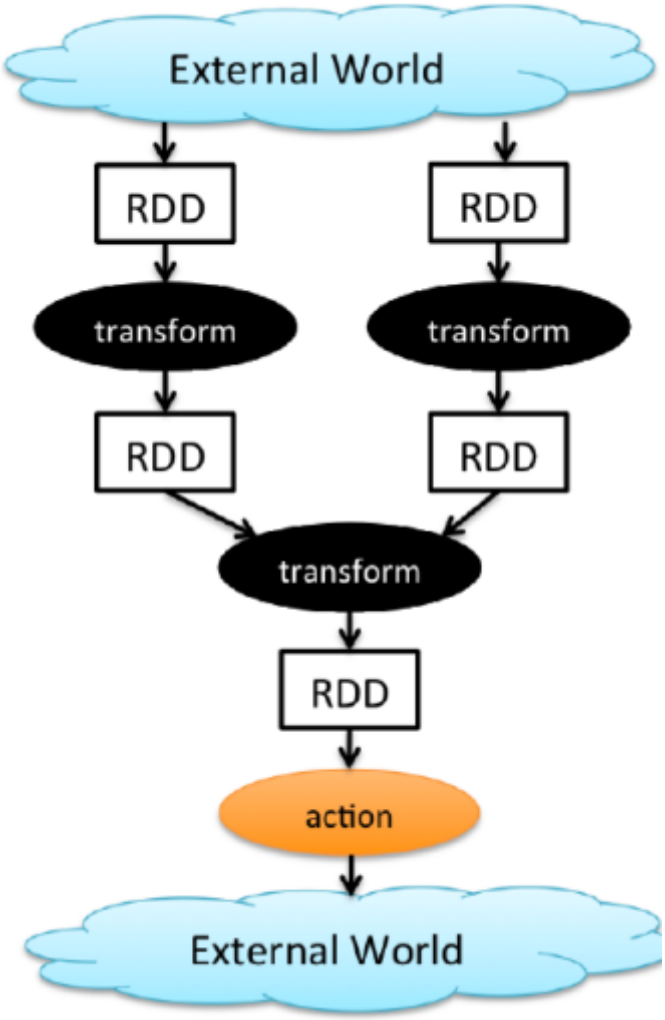
Spark: Extends MapReduce

Open source software (Apache Foundation, Apache 基金会)
Supports Java, Scala and Python

Spark: RDD

Key construct/idea: ***Resilient Distributed Dataset (RDD)***

- 记录分区的集合 (由 key-value pairs 组成)，进行分布式储存
- RDD 分布在集群中，并且是 read-only 的
- RDD 把 dataset 缓存在 memory (内存) 中
  - 有不同的 storage levels
  - 如果内存不够，数据可能存入 disk (磁盘)
- RDDs 可以从 Hadoop 中创建，或 transforming other RDDs (可以 stack RDDs)
- RDDs 可以对 dataset 中所有元素进行相同的操作



Spark RDD Operations

Transformations build RDDs 通过确定的操作，或其他 RDDs:

- Transformations include ***map, filter, join, union, intersection, distinct***
- **Lazy evaluation**: Nothing computed until an action requires it

Actions to return value or export data

- Actions include ***count, collect, reduce, save***
- Actions can be applied to RDDs; actions force calculations and return values