解剖MapReduce的心脏

让我们一起探索，奇迹发生的地方

@赛赛

目录

聊点啥？

1. **生活 + 经验 = 思想**
   - 斗地主
   - 掰玉米

2. **模型 + 实践 = 相识**
   - 经典 WordCount
   - 如何求各个部门的总工资？

3. **解剖 + 思考 = 升华**
   - 心脏是啥？

1

体会生活

斗地主，统计一副扑克牌的张数够不够（红桃多少张）？

架构思想来源于人们生活！

1

感受经验

掰玉米！还是忍不住，要提提！

架构思想源自于人们的工作经验！

1

生活 + 经验 = 思想

分而治之，解说来自于百度百科。

"分而治之"(Divide and conquer)方法(又称"分治术")，是有效算法设计中普遍采用的一种技术。

所谓"分而治之" 就是把一个复杂的算法问题按一定的"分解"方法分为等价的规模较小的若干部分，然后逐个解决，分别找出各部分的解，把各部分的解组成整个问题的解，这种朴素的思想来源于人们生活与工作的经验 也完全适合于技术领域。诸如软件的体系结构设计、模块化设计都是分而治之的具体表现。

中文名	分而治之方法	思　想	把复杂问题分解为等价的小问题
外文名	Divide and conquer	应　用	软件的体系结构设计、模块化设计
属　于	有效算法设计一种	又　称	分治术

架构思想来源于人们生活与工作经验！

模型

MapReduce是啥？ 是一个为了简化海量数据处理而提出的编程模型，思想就是分而治之。

编程模型

- ✓ 我们只需要通过编写 **Map 函数**和 **Reduce 函数**来指定想要进行的计算；
- ✓ 不用去纠结背后复杂的容错、数据分发、负载均衡等等一系列技术细节问题。

分而治之

- ✓ **Map 函数**负责“**分**”，即把复杂的任务分解为若干个“简单的任务”来处理。
- ✓ **Reduce 函数**负责对 map 阶段的**结果进行汇总**。

让我们慢慢去体会！

模型

来自于论文的伪代码，尝试一起读读经典 WordCount！

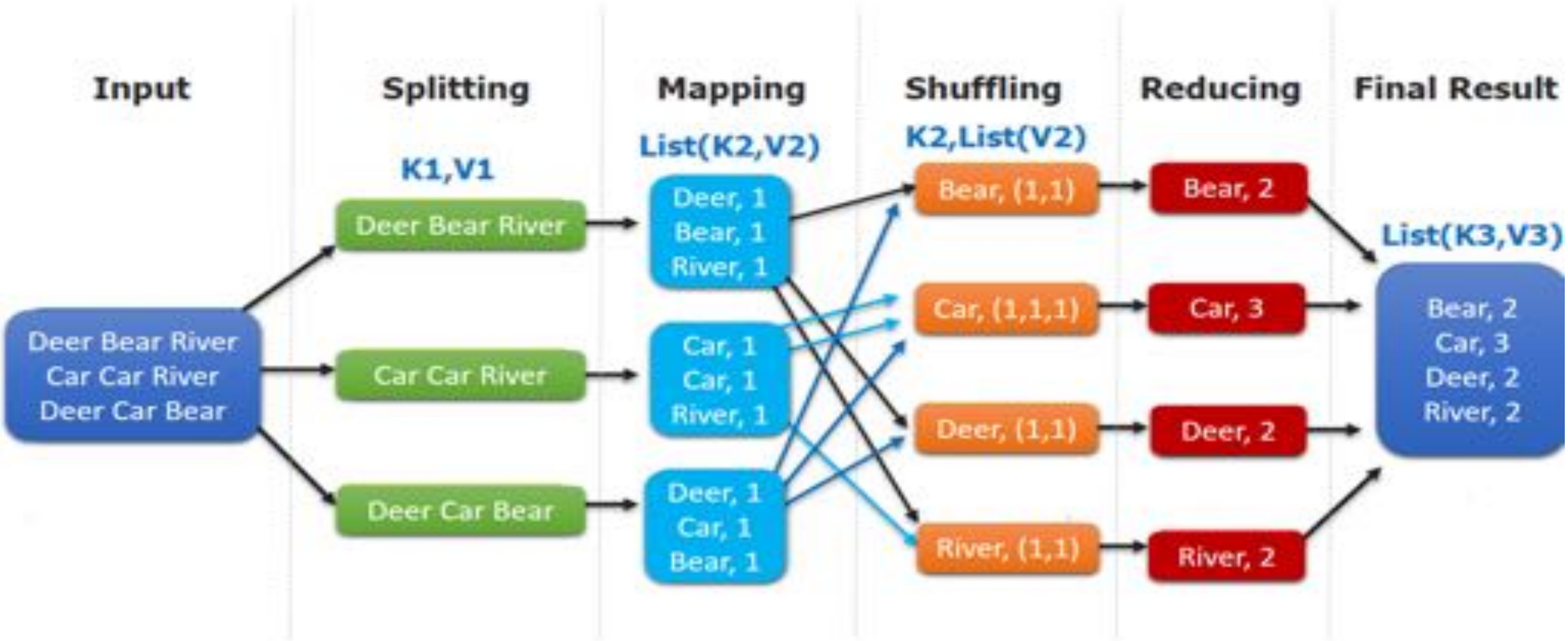
```
map(String key, String value):
   // key: document name
   // value: document contents
   for each word w in value:
      EmitIntermediate(w, "1");

reduce(String key, Iterator values):
   // key: a word
   // values: a list of counts
   int result = 0;
   for each v in values:
      result += ParseInt(v);
   Emit(AsString(result));
```

实践

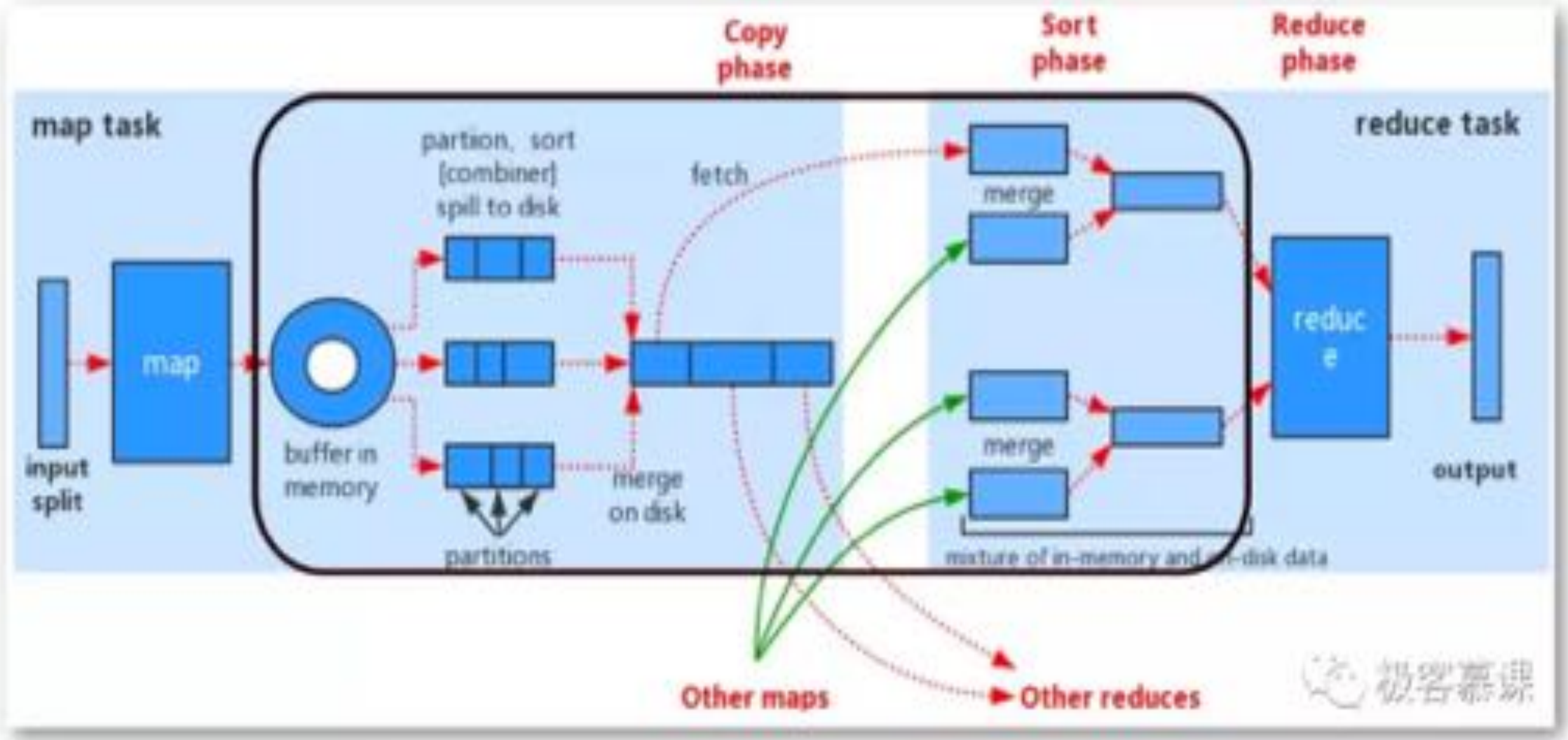
说一千道一万，不妨亲自操刀试试看！

模型 + 实践 = 相识

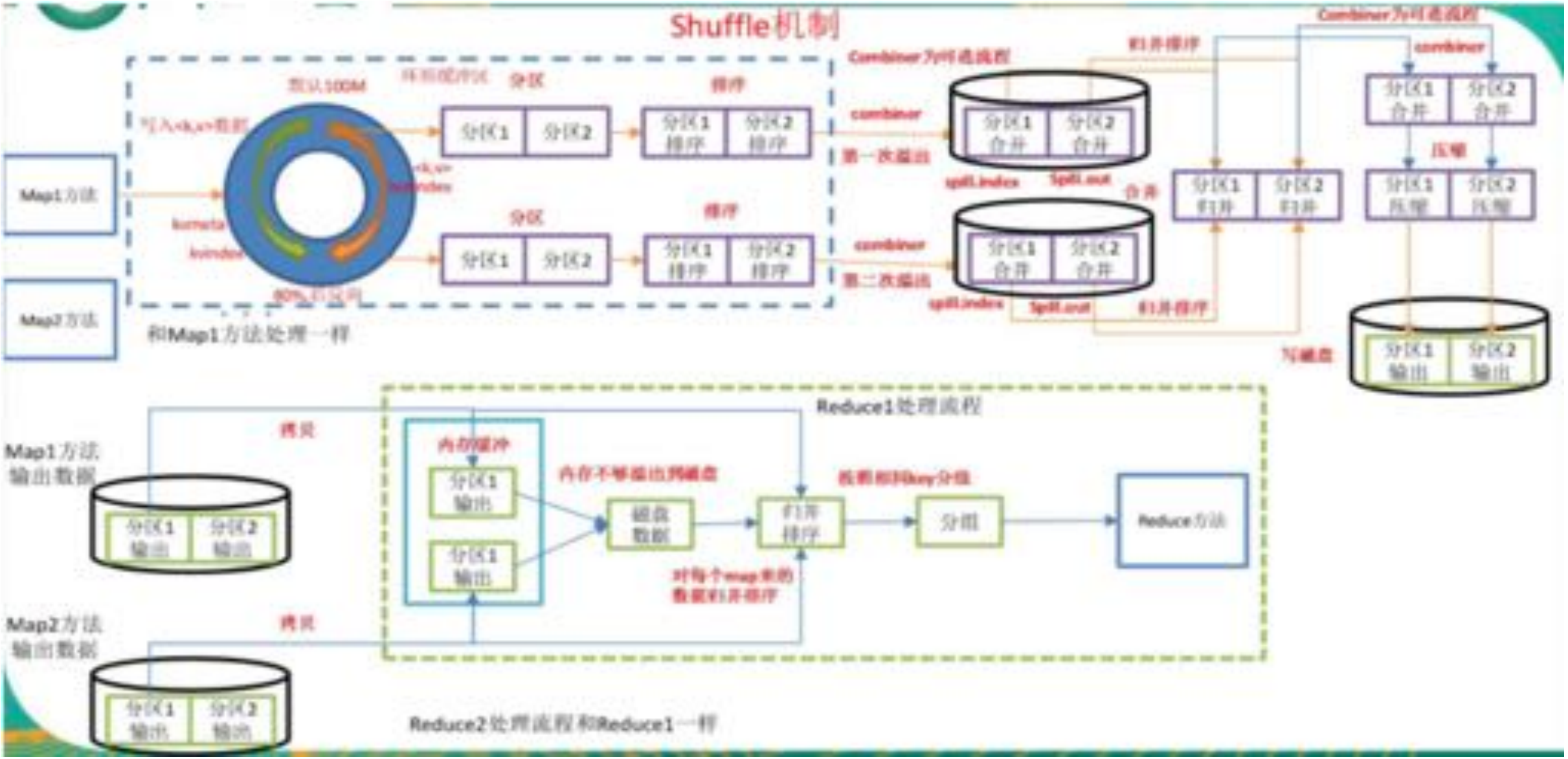


https://github.com/heibaiying/BigData-Notes/blob/master/notes/Hadoop-MapReduce.md

解剖“心脏”，一个奇迹发生的地方！



解剖“心脏”，一个奇迹发生的地方！



思考

十万个为什么

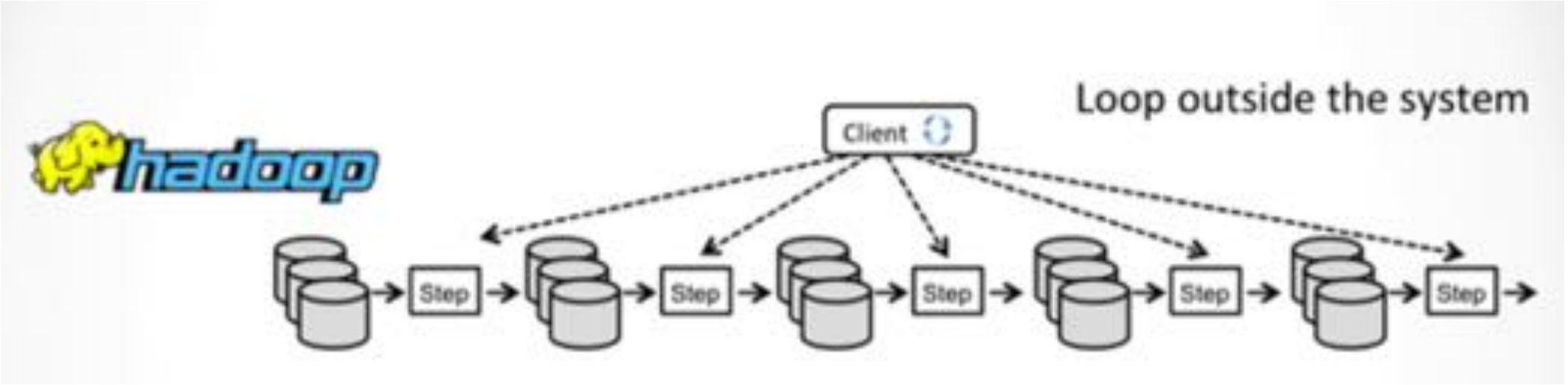
- ✓ 为什么要有缓冲区？
- ✓ 为什么要设计成环形缓冲区？
- ✓ 为什么要分区？
- ✓ 为什么要排序？
- ✓ 为什么支持合并？
- ✓ 。。。。。。

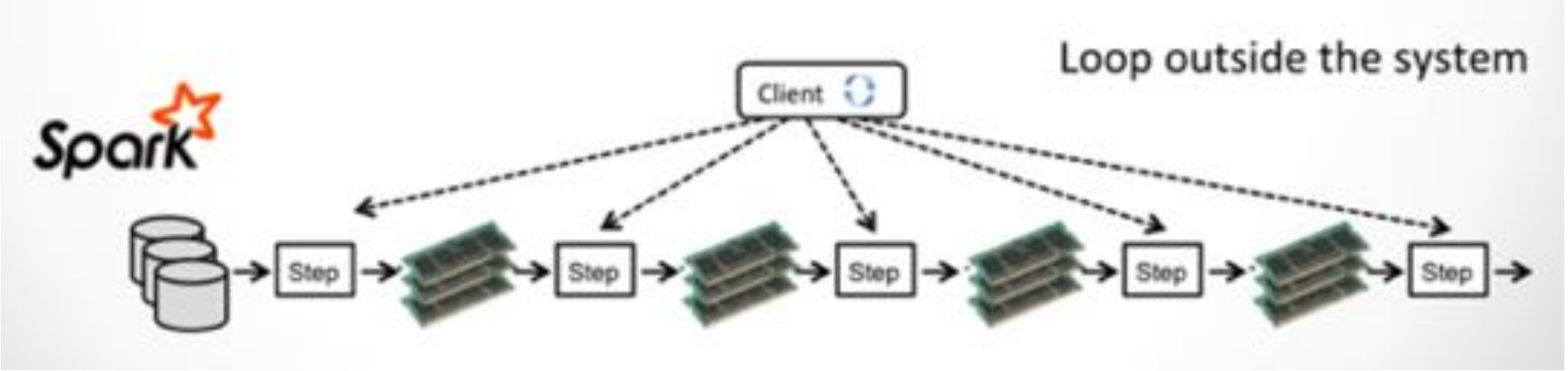
缺点

- ✓ MR 计算框架大多中间数据走磁盘，实时性低！
- ✓ MapReduce的输入数据是静态的，无法支持流计算！
- ✓ 大量的磁盘 IO！
- ✓ 过于底层，编程稍显复杂！
- ✓ 。。。。。。

重要原则：计算靠近数据（代码靠近数据、数据靠近代码）

解剖 + 思考 = 升华


hadoop
Loop outside the system
Client
Step
Step
Step
Step
Step


Spark
Loop outside the system
Client
Step
Step
Step
Step
Step

感谢！