* 批处理计算

批处理计算主要面向离散计算场景，计算的数据是静态数据，数据在计算前已经获取并保存，在计算过程中不会发生变化。例如电商领域统计一年的销售额，由于上一年的数据已经存在并且不会再增加和修改，因此该计算可以被允许计算一段时间而不必立即返回结果（即：允许进行离线处理）。

MapReduce就是典型的批处理计算系统。MapReduce最早是由Google公司研究提出的一种面向大规模数据处理的并行分布式计算模型和方法。

MapReduce的灵感来源于函数式语言（比如[Lisp](https://baike.baidu.com/item/Lisp/22083" \t "_blank)）中的内置函数map和reduce。在MapReduce中，一次计算主要分为Map（映射）和Reduce（规约）两个阶段。待计算的大数据被预先划分成多个数据块分别存储于各个计算节点。当计算作业被提交之后，该作业会被划分成若干Map任务和若干Reduce任务并由计算资源控制节点负责任务的调度和分配。各节点上的数据首先经过Map阶段的计算，形成中间结果（通常采用键值对Key-Value的方式）保存于负责执行Map任务的本地节点。中间结果经过排序后分发给各Reduce任务。各Reduce任务的计算节点从各Map任务计算节点处读取各自Reduce计算所需的中间结果，然后计算得出最终的结果并输出。

MapReduce通常使用 Java 语言在 Hadoop 集群上执行，但是可以借助 Hadoop Stream 的功能，使用Python编程实现。

【例】Python MapReduce进行单词计数

在当前文件夹下有一个文件words.txt，其中的内容为：“”,使用Python 语言按照MapReduce框架编程统计words.txt文件中不同单词的出现总次数。

完成Mapper阶段，读入words.txt，并给出每一次单词出现的次数（1次），mapper.py程序代码：

# 读入文件

file = open("words.txt",'r',encoding='utf-8')

while True:

# 依次读入文件的每一行

line = file.readline()

# 如果读到文件末尾，则退出

if not line:

break

# .strip() 函数是去除字符串前后的空白符（包括空格、换行符等）

# 使用空格拆分每一行中每个单词

words = line.strip().split()

# 输出每个单词对应一个数字 1

for word in words:

print('%s\t%s' % (word, 1))

该程序执行后，会输出：aa 1

bb 1

cc 1

dd 1

aa 1

bb 1

cc 1

然后使用上面的输出作为Reducer阶段的输入，完成最终单词计数统计的任务，reducer.py程序代码：

import sys

current\_word = None

current\_count = 0

word = None

# 从标准输入获取输入一行数据

for line in sys.stdin:

# 去除该行数据前后的空白符

line = line.strip()

# 使用制表符拆分输入

word, count = line.split('\t')

try:

# 获取单词的计数的整数形式数值

count = int(count)

except ValueError: #count如果不是数字的话，直接忽略掉

continue

# 如果连续两次的单词相同，则累加单词计数

if current\_word == word:

current\_count += count

else:

# 如果读入了一个新的的单词

# 打印输出上一个单词计数统计信息

if current\_word:

print("%s\t%s" % (current\_word, current\_count))

# 令current\_count和current\_word为新的单词计数信息

current\_count = count

current\_word = word

# 最后输出最后一个单词计数统计信息

if word == current\_word:

print("%s\t%s" % (current\_word, current\_count))

需要将mapper.py的程序的输出作为reducer.py的输入，使用操作系统的流API，完成程序的执行，命令为：

python mapper.py |sort | python reducer.py

其中：

| 为流运算符，含义为将前一个程序的输出作为后一个程序的输入；

sort 执行完mapper之后对结果进行排序，保证相同单词处于相邻位置的，这样可以简化执行reducer的执行步骤。

程序执行结果为：

aa 2

bb 2

cc 2

dd 1

可以看成是统计出words.txt中每个单词计数的个数。

如果将words.txt文件放在 Hadoop 的 HDFS文件系统中，则上传程序代码和执行步骤就可以对更大容量的数据文件进行单词计数统计。