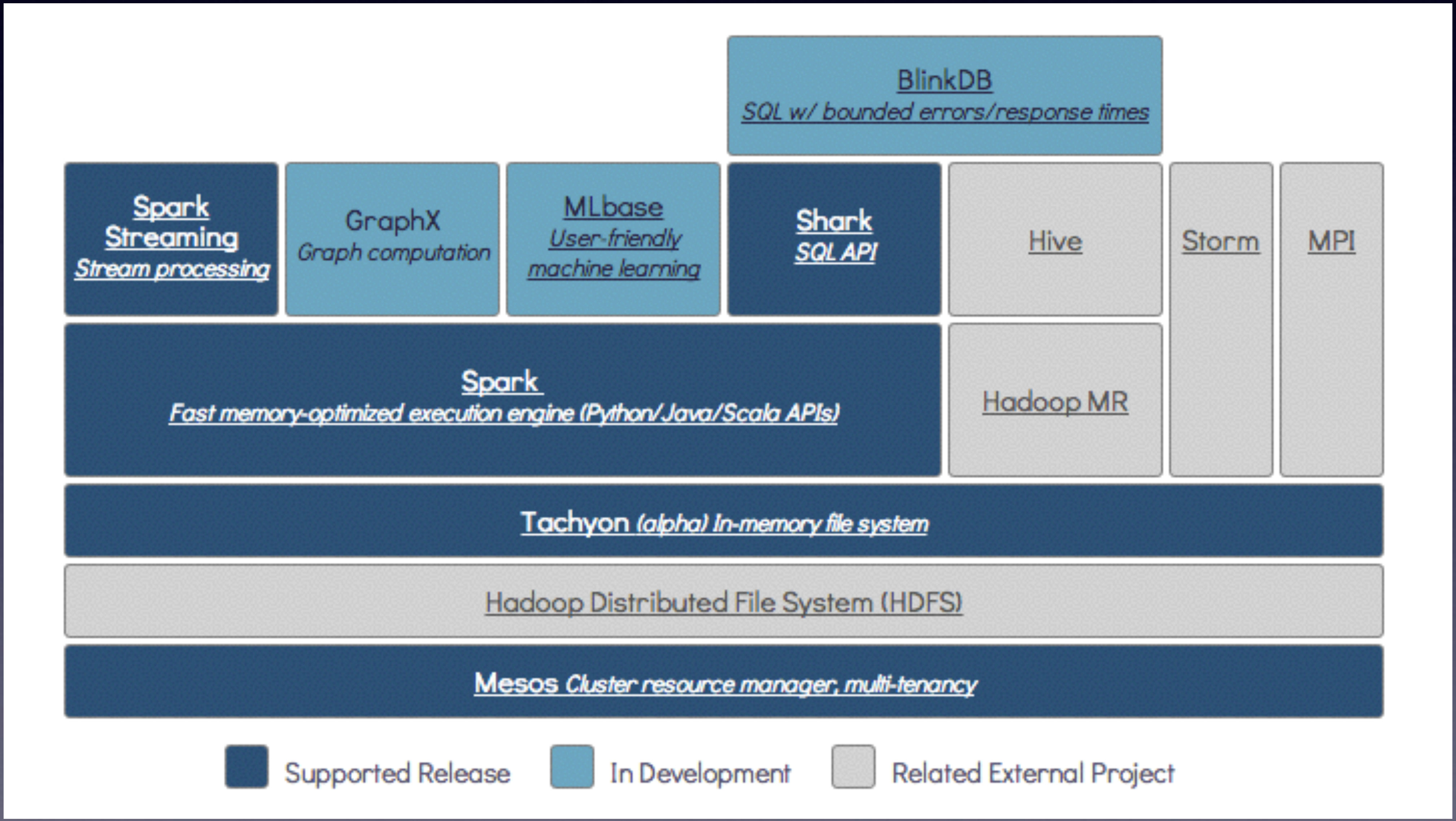
# Spark概述与编程模型

陈 超
@CrazyJvm

# What’s Spark

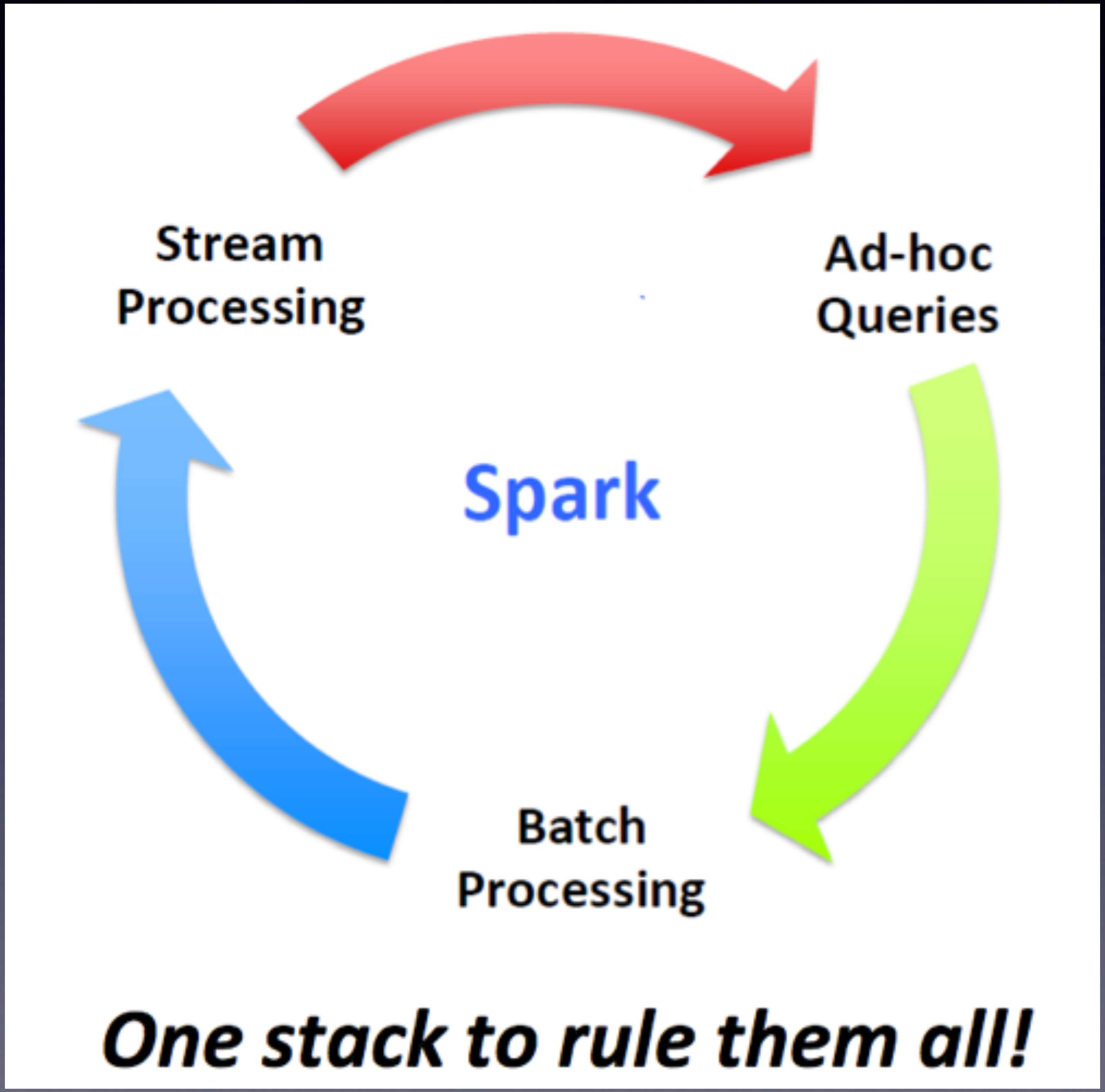
- Apache Spark is an open source cluster computing system that aims to make data analytics fast — both fast to run and fast to write

# BDAS

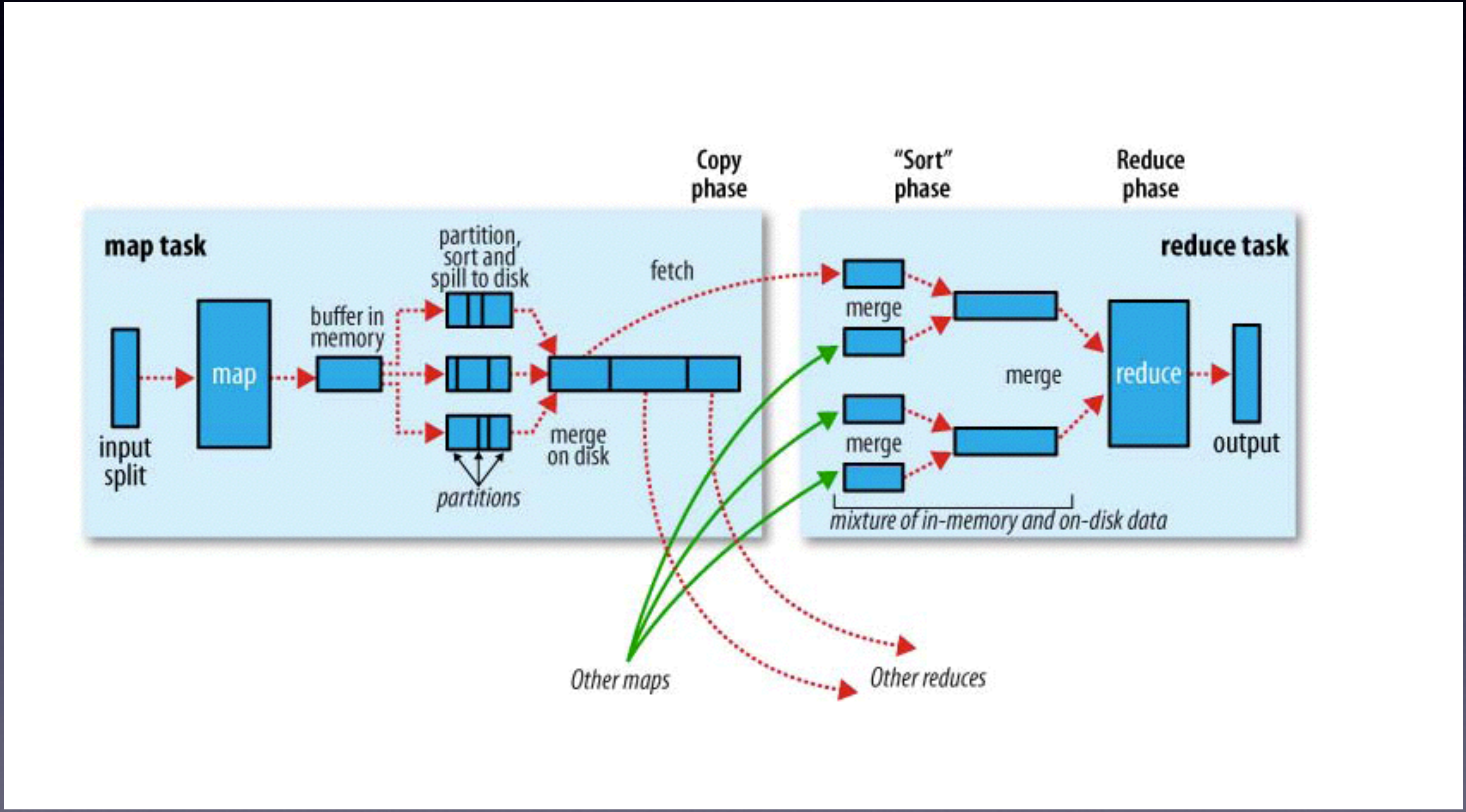

BlinkDB
SQL w/ bounded errors/response times
Spark Streaming
Stream processing
GraphX
Graph computation
MLbase
User-friendly machine learning
Shark
SQL API
Hive
Storm
MPI
Spark
Fast memory-optimized execution engine (Python/Java/Scala APIs)
Hadoop MR
Tachyon (alpha) In-memory file system
Hadoop Distributed File System (HDFS)
Mesos Cluster resource manager, multi-tenancy
Supported Release
In Development
Related External Project

*the Berkeley Data Analytics Stack*

# 搞定所有！

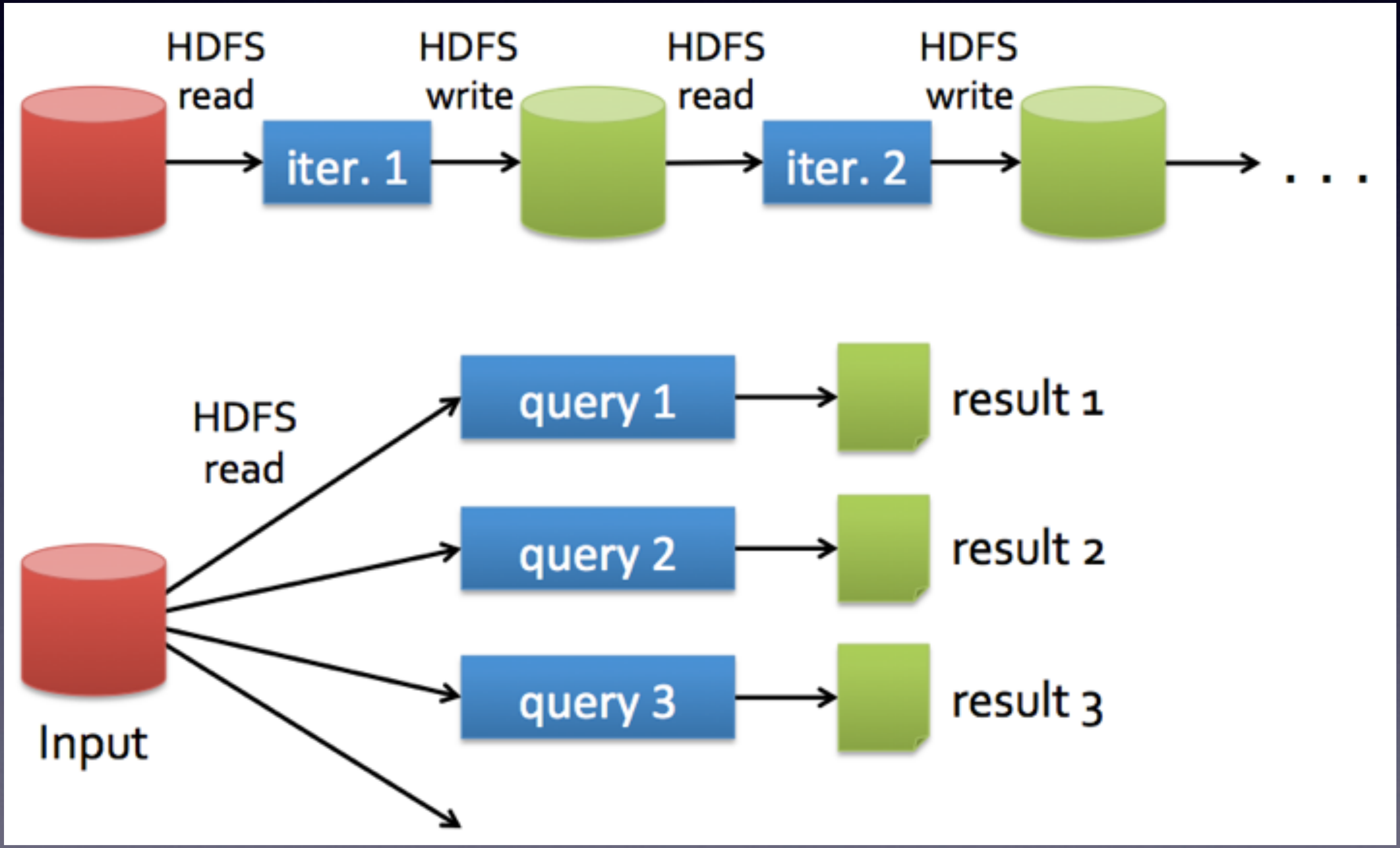

Stream Processing
Ad-hoc Queries
Spark
Batch Processing
One stack to rule them all!

## 回顾Hadoop


Copy phase
"Sort" phase
Reduce phase
map task
reduce task
partition, sort and spill to disk
fetch
buffer in memory
map
input split
merge on disk
partitions
merge
merge
merge
reduce
output
mixture of in-memory and on-disk data
Other maps
Other reduces

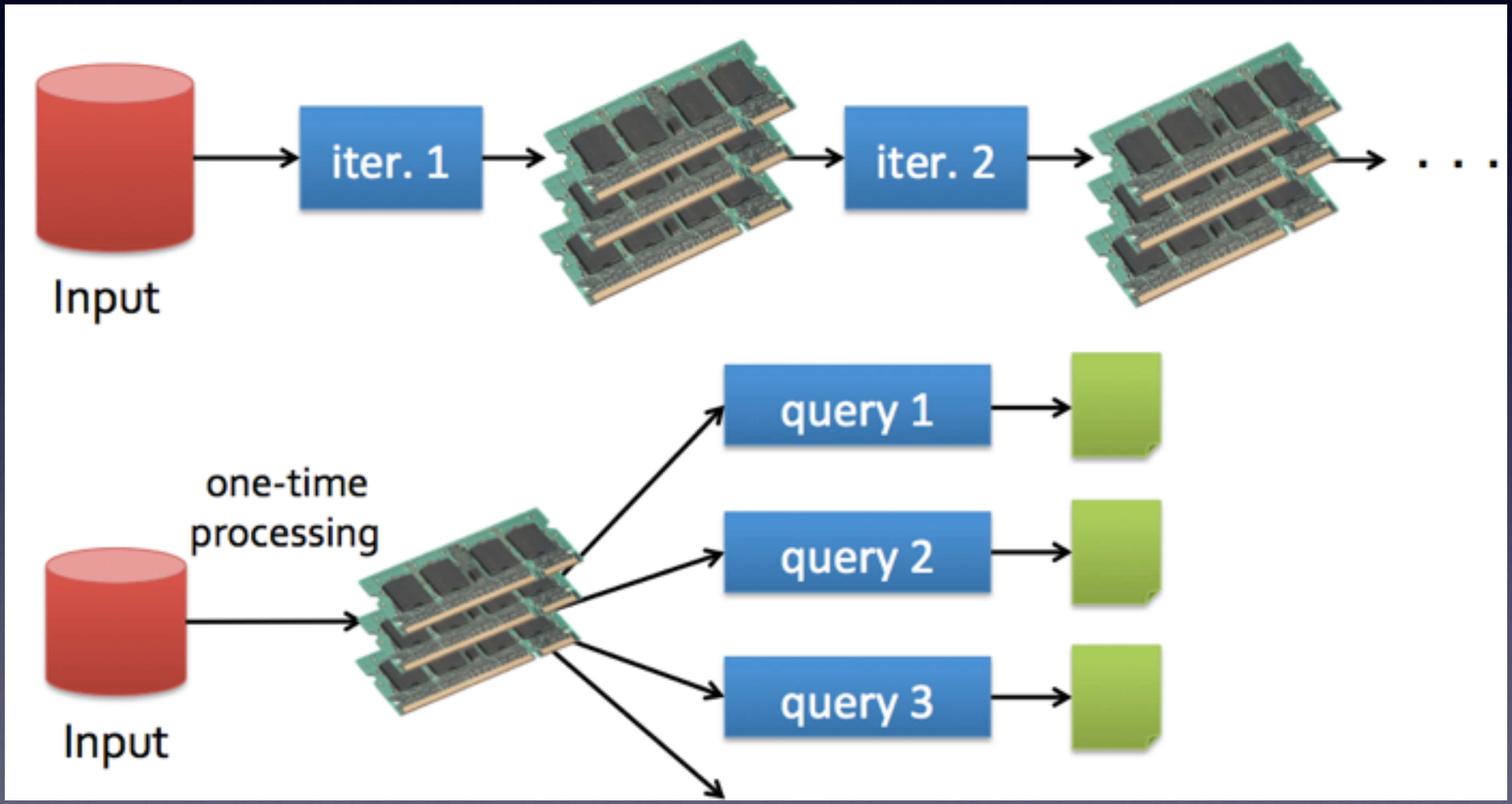
# Hadoop的数据共享？慢！

- 为什么慢？？？额外的复制，序列化和磁盘IO开销。



小象学院 CHINA HADOOP

# Spark的共享数据？快！



## Spark的快只是因为内存?

- 内存计算
- DAG

很多优化措施其实是想通的，譬如说delay scheduling.

# Spark API呢?

- 支持3种语言的API
  - Scala(很好)
  - Python(不错)
  - Java(…)

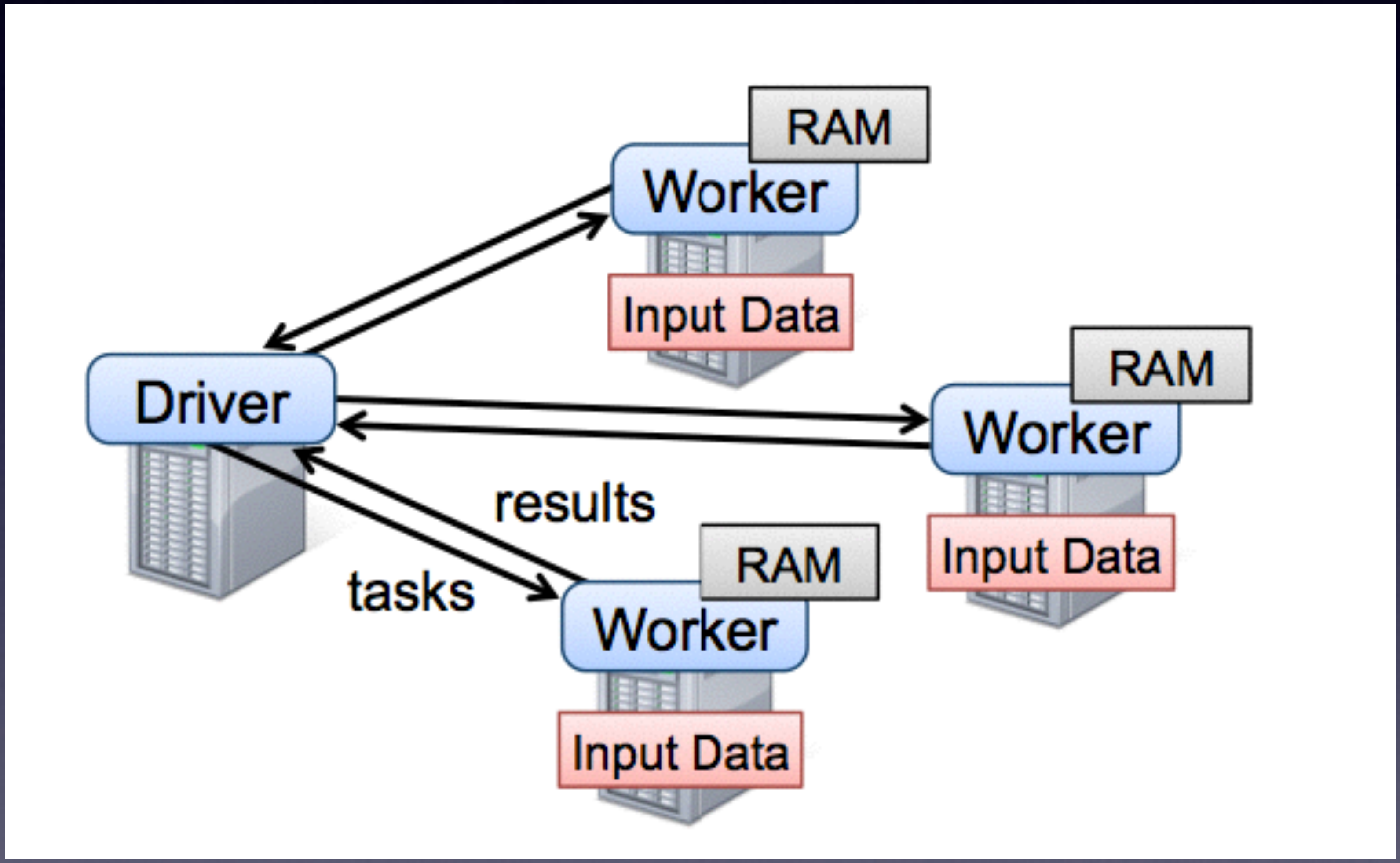
## 通过哪些模式运行Spark呢?

- 有4种模式可以运行
  - local(多用于测试)
  - Standalone
  - Mesos
  - YARN

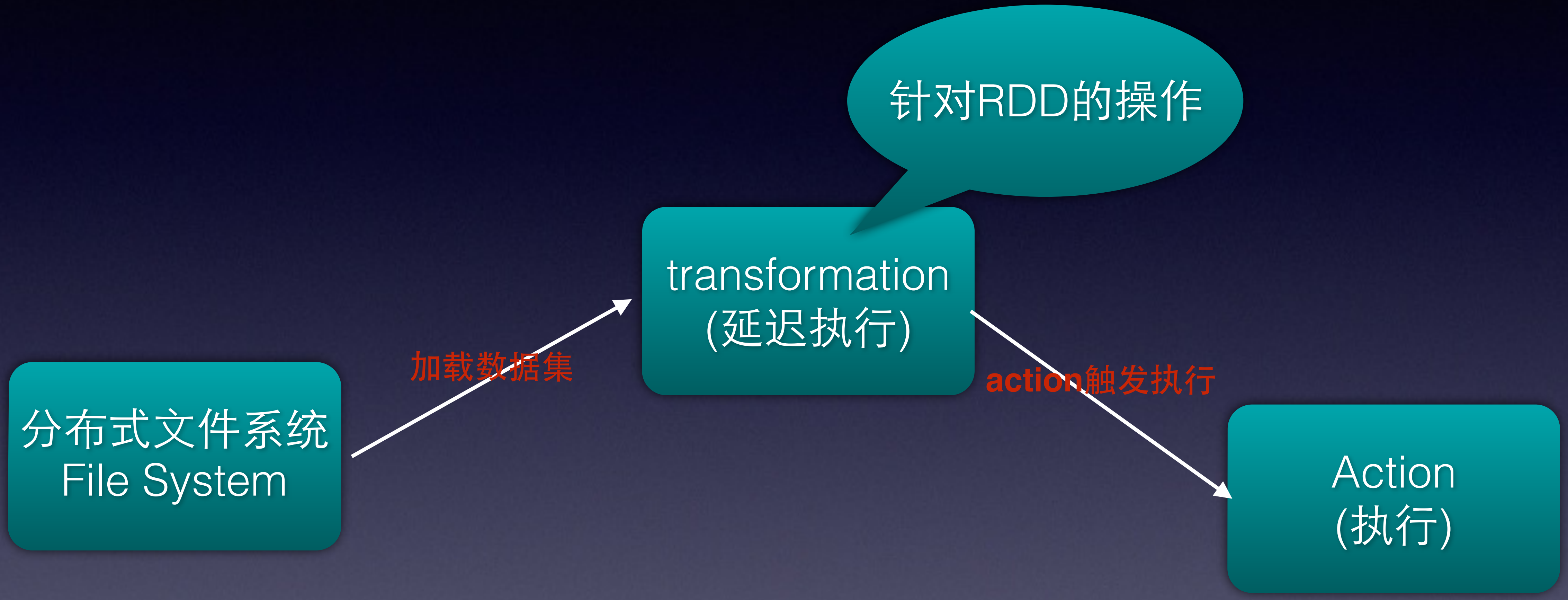
# 其实…一切都以RDD为基础

- Resilient Distributed Dataset
  - A list of partitions
  - A function for computing each split
  - A list of dependencies on other RDDs
  - Optionally, a Partitioner for key-value RDDs (e.g. to say that the RDD is hash-partitioned)
  - Optionally, a list of preferred locations to compute each split on (e.g. block locations for an HDFS file)

# Spark runtime



# 流程示意



ps:RDD可以从集合直接转换而来，也可以由从现存的任何Hadoop InputFormat而来,亦或者HBase等等。

# first demo!

```
lines = sc.textFile("hdfs://...")

errors = lines.filter(_.startsWith("ERROR"))

errors.persist()

Mysql_errors = errors.filter(_.contains("MySQL")).count

http_errors = errors.filter(_.contains("Http")).count
```

加载进来成为RDD

transformation

缓存RDD

action

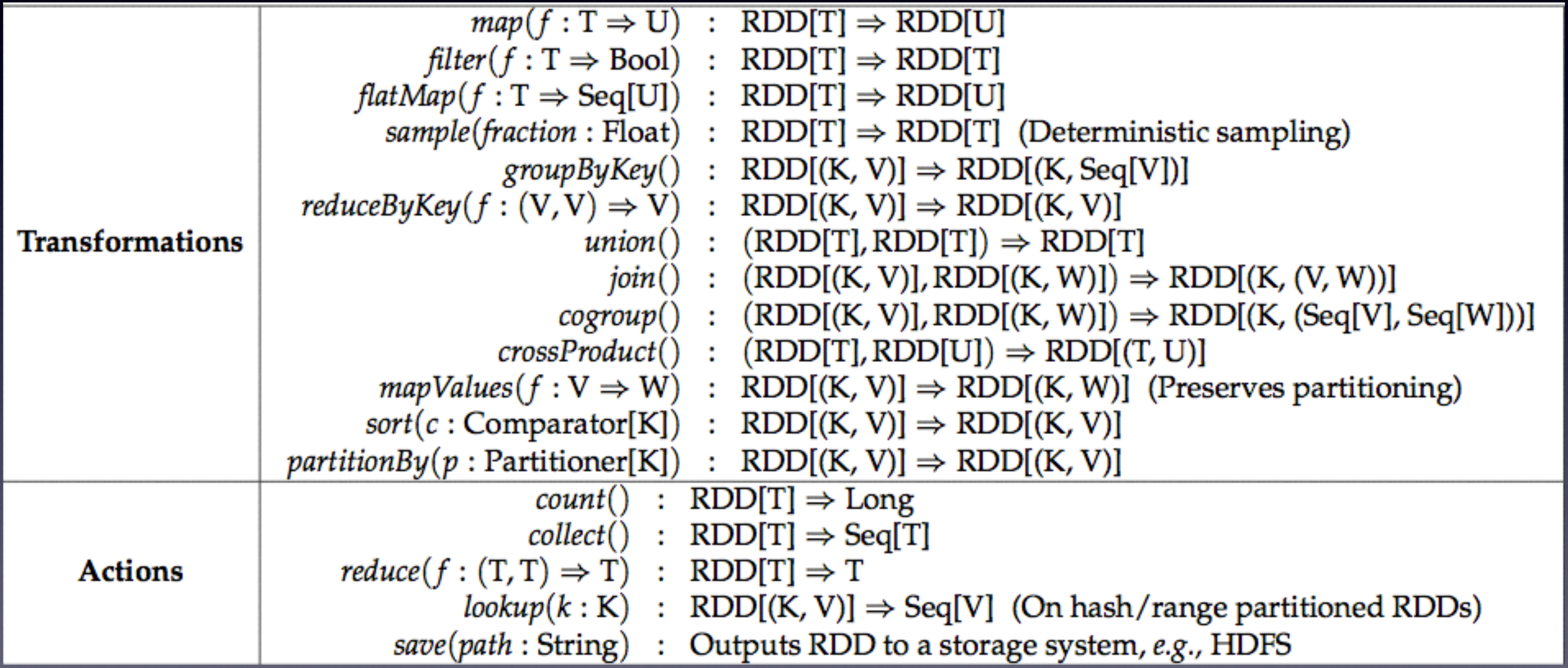
# 缓存策略

```
class StorageLevel private(
   private var useDisk_ : Boolean,
   private var useMemory_ : Boolean,
   private var deserialized_ : Boolean,
   private var replication_ : Int = 1)
```

```
val NONE = new StorageLevel(false, false, false)
val DISK_ONLY = new StorageLevel(true, false, false)
val DISK_ONLY_2 = new StorageLevel(true, false, false, 2)
val MEMORY_ONLY = new StorageLevel(false, true, true)
val MEMORY_ONLY_2 = new StorageLevel(false, true, true, 2)
val MEMORY_ONLY_SER = new StorageLevel(false, true, false)
val MEMORY_ONLY_SER_2 = new StorageLevel(false, true, false, 2)
val MEMORY_AND_DISK = new StorageLevel(true, true, true)
val MEMORY_AND_DISK_2 = new StorageLevel(true, true, true, 2)
val MEMORY_AND_DISK_SER = new StorageLevel(true, true, false)
val MEMORY_AND_DISK_SER_2 = new StorageLevel(true, true, false, 2)
```

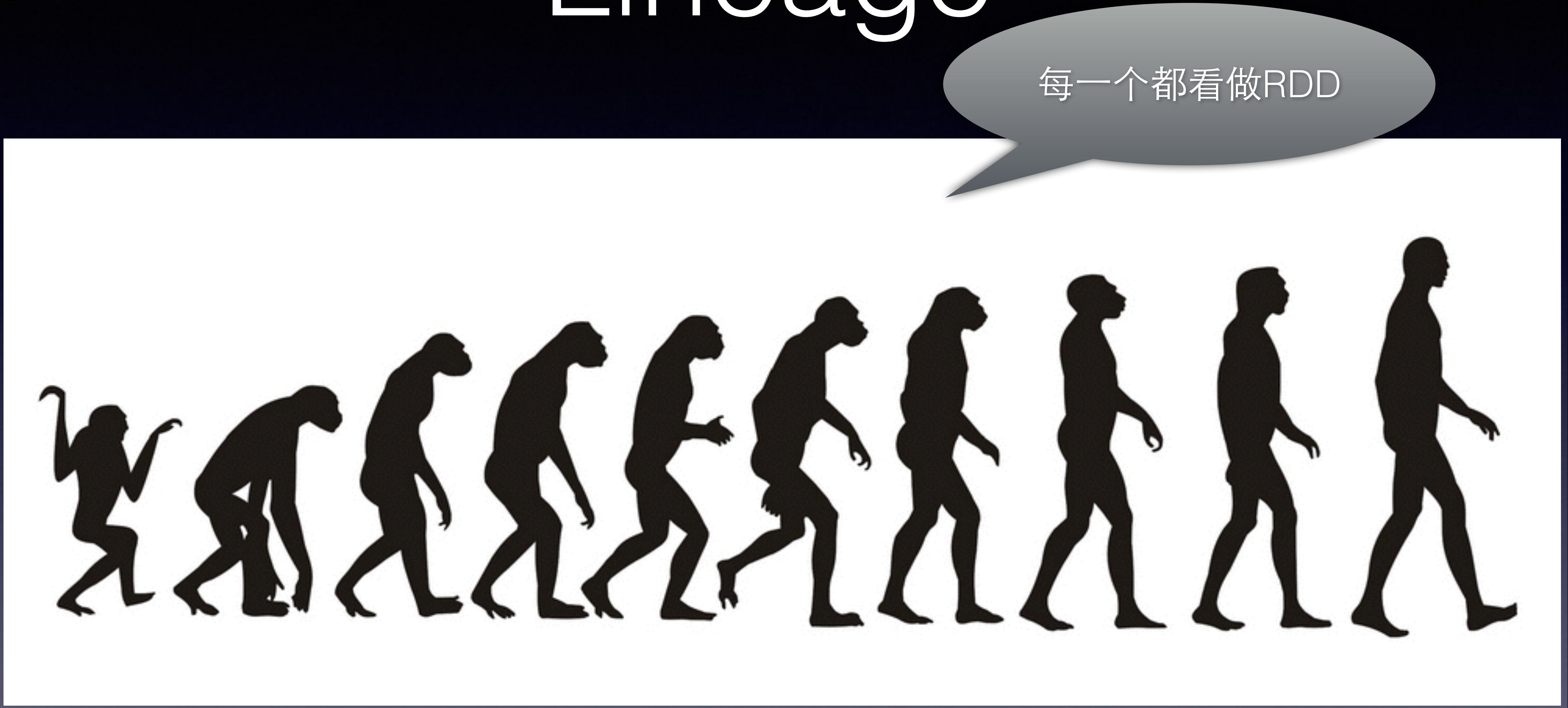
cache默认

# transformation & action



| | | |
|---|---|---|
| **Transformations** | *map*(*f* : T ⇒ U) | : RDD[T] ⇒ RDD[U] |
| | *filter*(*f* : T ⇒ Bool) | : RDD[T] ⇒ RDD[T] |
| | *flatMap*(*f* : T ⇒ Seq[U]) | : RDD[T] ⇒ RDD[U] |
| | *sample*(*fraction* : Float) | : RDD[T] ⇒ RDD[T] (Deterministic sampling) |
| | *groupByKey*() | : RDD[(K, V)] ⇒ RDD[(K, Seq[V])] |
| | *reduceByKey*(*f* : (V, V) ⇒ V) | : RDD[(K, V)] ⇒ RDD[(K, V)] |
| | *union*() | : (RDD[T], RDD[T]) ⇒ RDD[T] |
| | *join*() | : (RDD[(K, V)], RDD[(K, W)]) ⇒ RDD[(K, (V, W))] |
| | *cogroup*() | : (RDD[(K, V)], RDD[(K, W)]) ⇒ RDD[(K, (Seq[V], Seq[W]))] |
| | *crossProduct*() | : (RDD[T], RDD[U]) ⇒ RDD[(T, U)] |
| | *mapValues*(*f* : V ⇒ W) | : RDD[(K, V)] ⇒ RDD[(K, W)] (Preserves partitioning) |
| | *sort*(*c* : Comparator[K]) | : RDD[(K, V)] ⇒ RDD[(K, V)] |
| | *partitionBy*(*p* : Partitioner[K]) | : RDD[(K, V)] ⇒ RDD[(K, V)] |
| **Actions** | *count*() | : RDD[T] ⇒ Long |
| | *collect*() | : RDD[T] ⇒ Seq[T] |
| | *reduce*(*f* : (T, T) ⇒ T) | : RDD[T] ⇒ T |
| | *lookup*(*k* : K) | : RDD[(K, V)] ⇒ Seq[V] (On hash/range partitioned RDDs) |
| | *save*(*path* : String) | : Outputs RDD to a storage system, *e.g.*, HDFS |

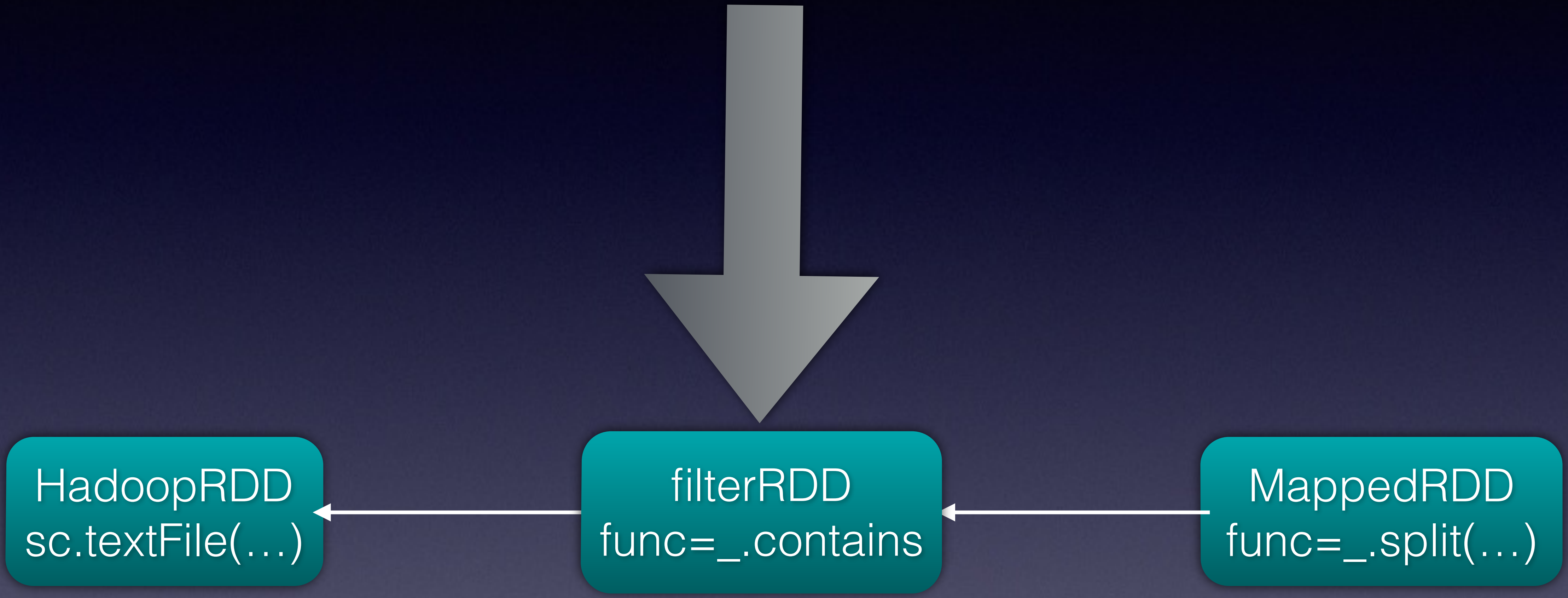
## Lineage



但是假如每次都快到进化完的时候就挂了，那岂不是每次都要从头进化？何不在中间制作个拷贝呢？！

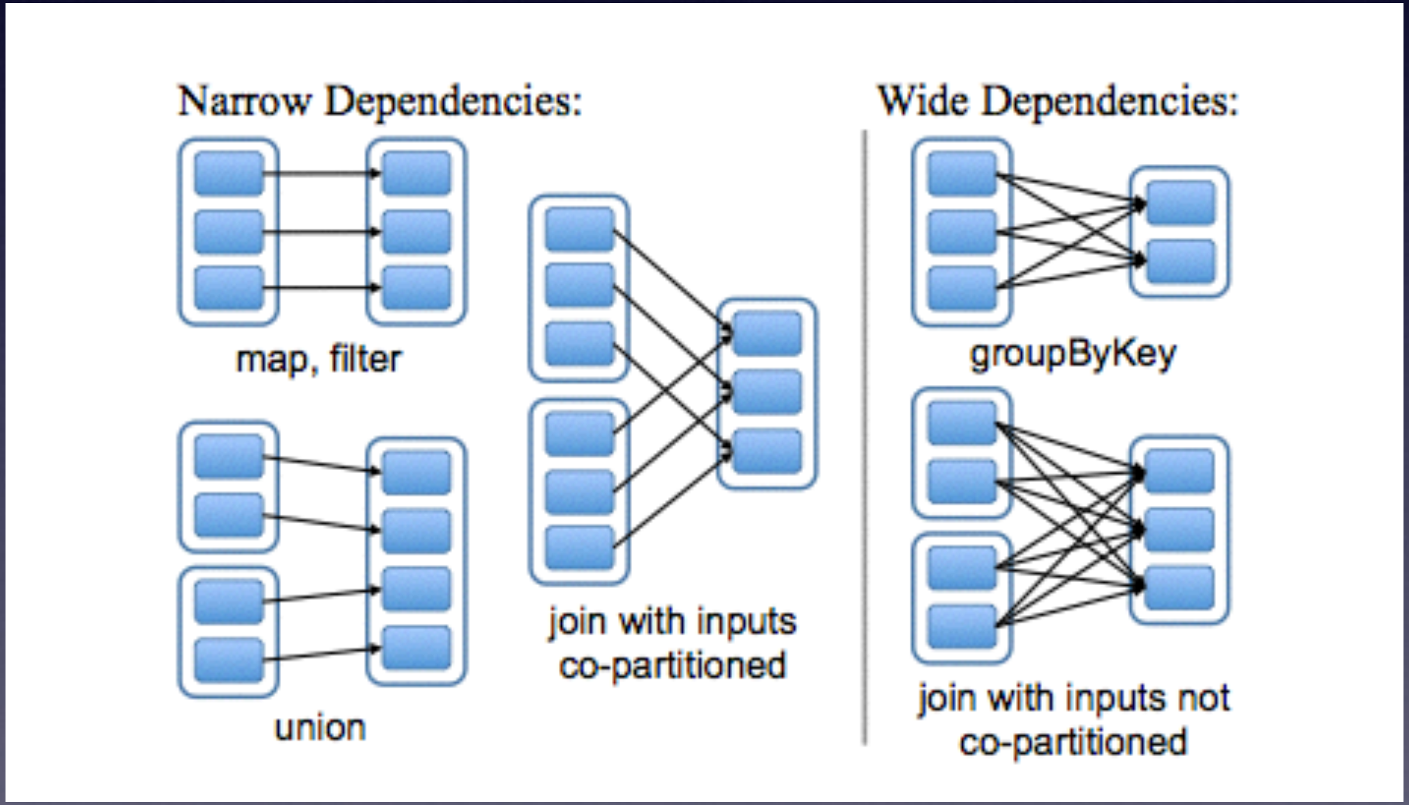
# 容错

```
val logs = sc.textFile(…).filter(_.contains("spark")).map(_.split('\t')(1))
```



每个RDD都会记录自己依赖于哪个(哪些)RDD，万一某个RDD的某些partition挂了，可以通过其它RDD并行计算迅速恢复出来。

# Dependency


Narrow Dependencies:
Wide Dependencies:
map, filter
union
join with inputs co-partitioned
groupByKey
join with inputs not co-partitioned

# 集群配置

```
spark-env.sh

export JAVA_HOME=
export SPARK_MASTER_IP=
export SPARK_WORKER_CORES=
export SPARK_WORKER_INSTANCES=
export SPARK_WORKER_MEMORY=
export SPARK_MASTER_PORT=
export SPARK_JAVA_OPTS="-verbose:gc -XX:-PrintGCDetails -XX:+PrintGCTimeStamps”
```

```
slaves
xx.xx.xx.2
xx.xx.xx.3
xx.xx.xx.4
xx.xx.xx.5
```

# 版本选择?

- 自己编译 — 可能会遇到某些问题
- pre-built版本

let’s have a try!

interactive shell & programming in IDE

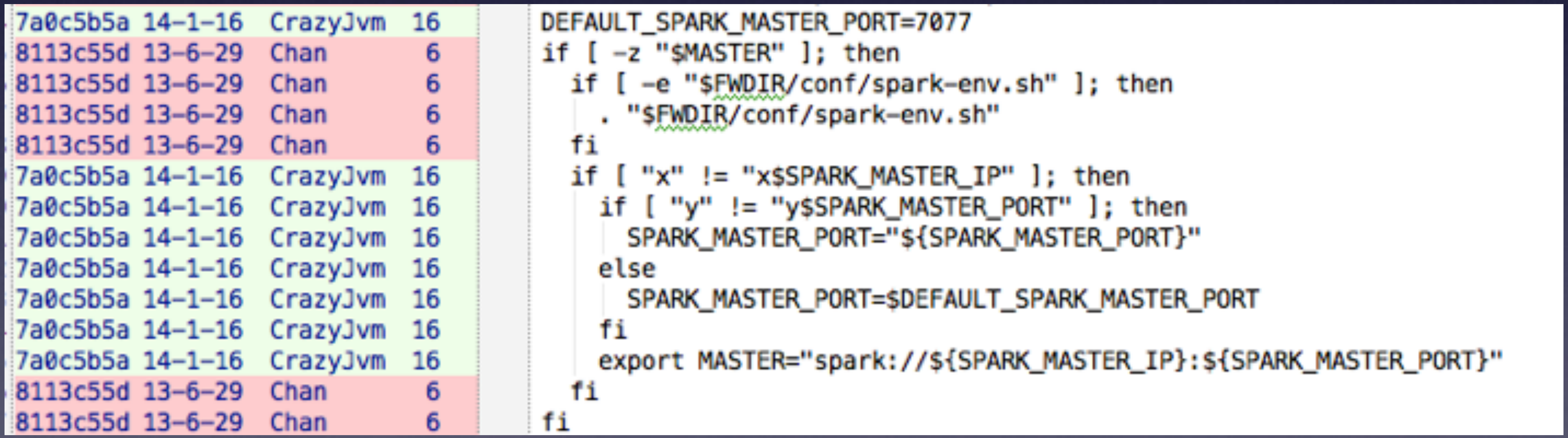
# shell运行

几个本地线程

- MASTER=local[4] ADD_JARS=code.jar ./spark-shell
- MASTER=spark://host:port
- 指定executor内存：export SPARK_MEM=25g

# spark-shell注意

spark-shell intends to set MASTER automatically if we do not provide the option when we start the shell , but there's a problem. The condition is "if [[ "x" != "x$SPARK_MASTER_IP" && "y" != "y $SPARK_MASTER_PORT" ]];" we sure will set SPARK_MASTER_IP explicitly, the SPARK_MASTER_PORT option, however, we probably do not set just using spark default port 7077. So if we do not set SPARK_MASTER_PORT, the condition will never be true. We should just use default port if users do not set port explicitly I think.



```
7a0c5b5a 14-1-16  CrazyJvm  16    DEFAULT_SPARK_MASTER_PORT=7077
8113c55d 13-6-29  Chan       6    if [ -z "$MASTER" ]; then
8113c55d 13-6-29  Chan       6      if [ -e "$FWDIR/conf/spark-env.sh" ]; then
8113c55d 13-6-29  Chan       6        . "$FWDIR/conf/spark-env.sh"
8113c55d 13-6-29  Chan       6      fi
7a0c5b5a 14-1-16  CrazyJvm  16      if [ "x" != "x$SPARK_MASTER_IP" ]; then
7a0c5b5a 14-1-16  CrazyJvm  16        if [ "y" != "y$SPARK_MASTER_PORT" ]; then
7a0c5b5a 14-1-16  CrazyJvm  16          SPARK_MASTER_PORT="${SPARK_MASTER_PORT}"
7a0c5b5a 14-1-16  CrazyJvm  16        else
7a0c5b5a 14-1-16  CrazyJvm  16          SPARK_MASTER_PORT=$DEFAULT_SPARK_MASTER_PORT
7a0c5b5a 14-1-16  CrazyJvm  16        fi
7a0c5b5a 14-1-16  CrazyJvm  16        export MASTER="spark://${SPARK_MASTER_IP}:${SPARK_MASTER_PORT}"
8113c55d 13-6-29  Chan       6      fi
8113c55d 13-6-29  Chan       6    fi
```

# IDE

- 推荐Intellij IDEA
- 加入依赖
- coding
- 打包
- 运行

## Demo with IDE

谢谢大家！