livelessons
video instruction from technology experts

Lesson 4: Spark Internals

4.4 RDD Deep Dive: Dependencies and Lineage

Functions Revisited

```
import random
flips = 1000000

# lazy eval
coins = xrange(flips)
```

nothing runs here

```
# lazy eval, nothing executed
heads_rdd = sc.parallelize(coins) \
              .map(lambda i: random.random()) \
              .filter(lambda r: r < 0.51)
```

```
head_count = heads_rdd.count()
```

Everything runs here

What is an RDD?

An ***Abstraction!***

What is an RDD?

Lineage (required)

1. Set of partitions for current RDD (data)

2. List of dependencies

3. Function to compute partitions (functional paradigm)

4. Partitioner to optimize execution

5. Potential preferred location for partitions

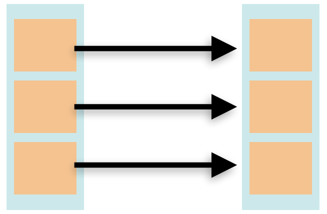
Optimization (optional)

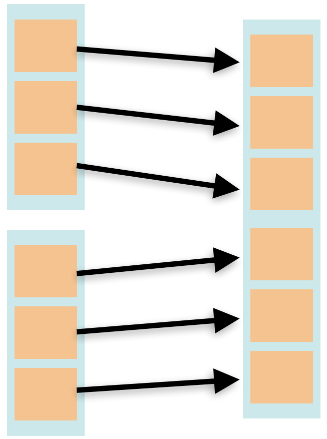
RDD as an interface

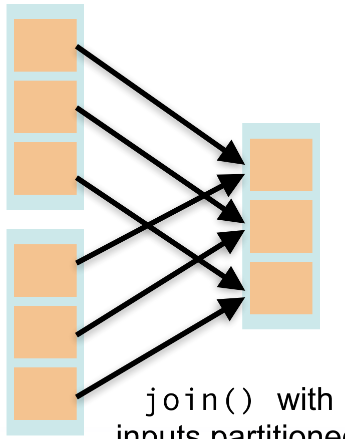
Operation	***Meaning***
`partitions()`	Return a list of `Partition` objects
`dependencies()`	Return a list of dependencies
`compute(p, parent)`	Compute the elements of `Partition` ***p*** given its ***parent*** `Partitions`
`partitioner()`	Return metadata specifying whether this RDD is hash/range partitioned
`preferredLocations(p)`	List nodes where `Partition` ***p*** can be accessed quicker due to data locality

Partition Dependencies

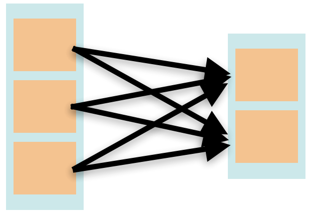
Narrow (can pipeline)

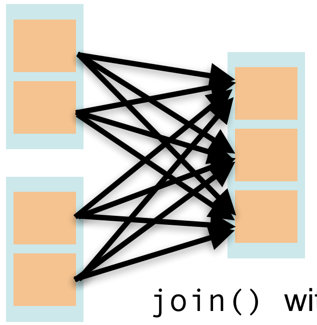




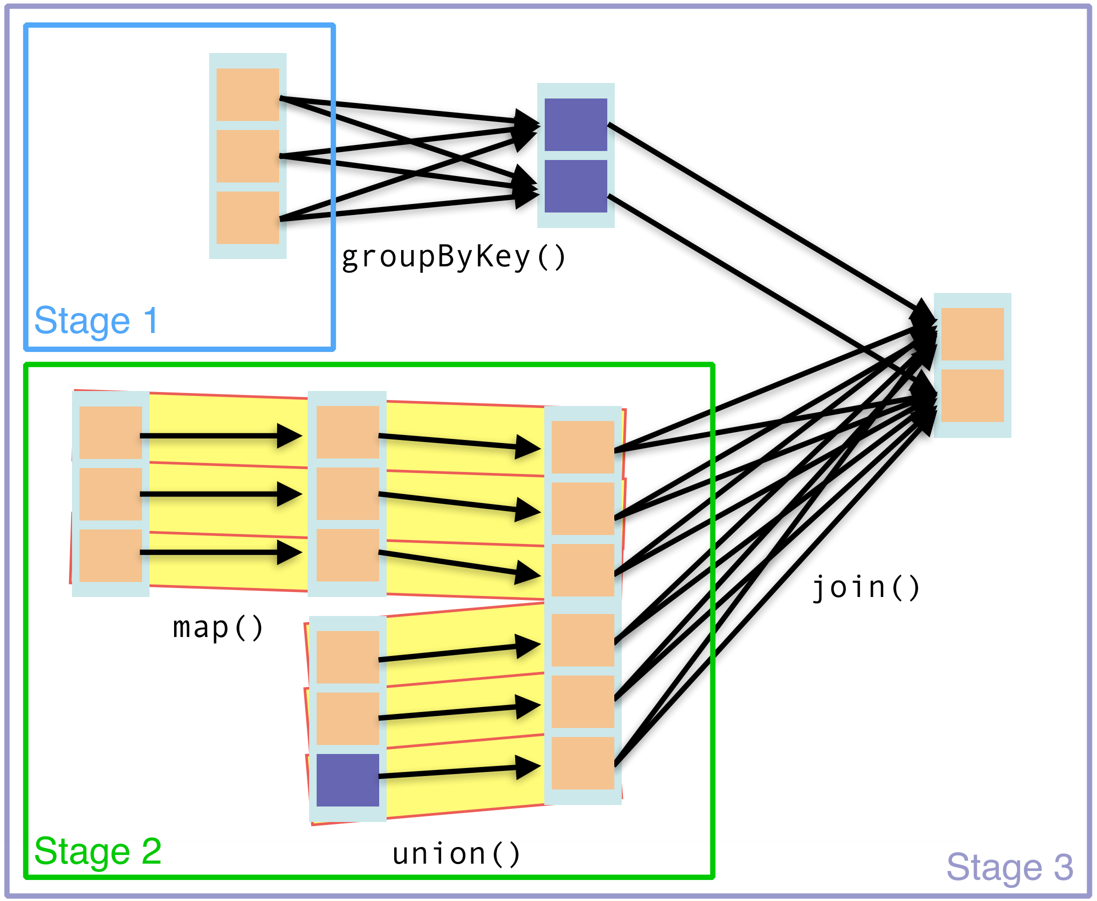


Wide (shuffle)





Partition Dependencies


groupByKey()
Stage 1
map()
union()
Stage 2
join()
Stage 3