CSE 4/587
Data Intensive Computing

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall

Dr. Shamshad Parvin
shamsadp@buffalo.edu
313 Davis Hall

Spark Demo

Announcements

References

- **Advanced Analytics with Spark** by S. Ryza, U. Laserson, S. Owen and J. Wills
- **Apache Spark documentation**
  - http://spark.apache.org/
  - https://spark.apache.org/examples.html
- **Pyspark Examples**
  - https://github.com/apache/spark/tree/master/examples/src/main/python
- **Resilient Distributed Dataset: A Fault-tolerant Abstraction for in-Memory Cluster Computing.** M. Zaharia et al.
  - https://www.usenix.org/system/files/conference/nsdi12/nsdi12-final138.pdf

Running Spark

Spark can be installed following instructions here: https://spark.apache.org/docs/latest/quick-start.html

In today's demos we'll be using the Python API, PySpark

Running Spark

Once installed, we have two ways to run a Spark program:

Interactively via the Spark Shell (command: pyspark)

- Provides us with a REPL to try commands
- Provides a GUI to show us what our Spark programs are doing

Batch jobs via python or spark-submit

- Let's us specify a python script containing a Spark job we want to run

Word Count in Spark (Python API)

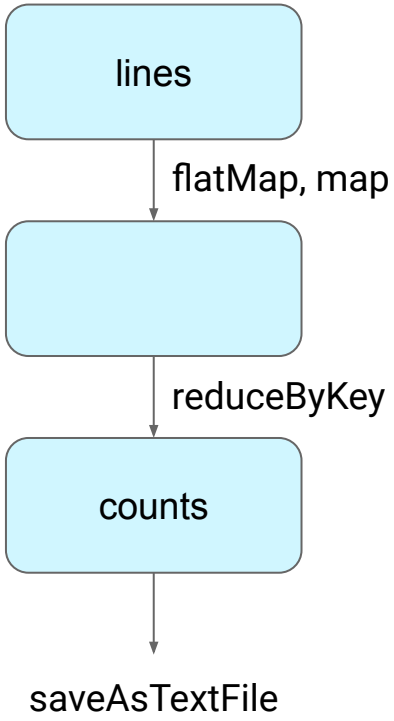
```
def countWords(sc, file):
  lines = sc.textFile(file)
  counts = lines.flatMap(lambda x: x.split(' ')) \
                .map(lambda x: (x, 1)) \
                .reduceByKey(lambda a,b: a + b)
  return counts

from pyspark.context import SparkContext
sc = SparkContext('local', 'test')

countWords(sc, "frankenstein.txt").saveAsTextFile("output")
```

Word Count in Spark

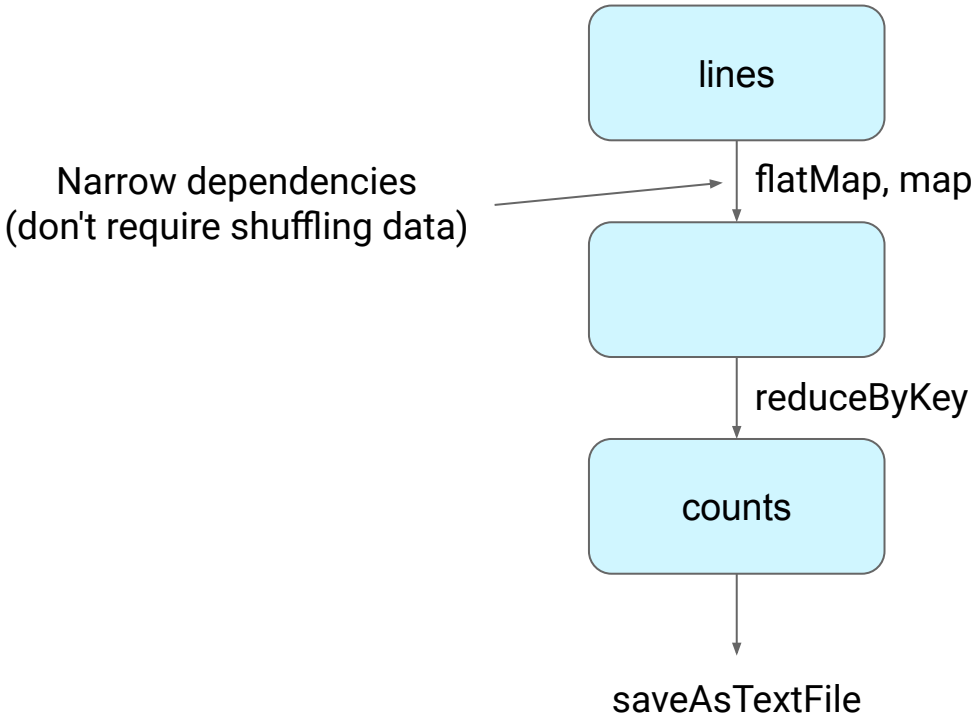
What does the RDD DAG generated by the word count program look like?

How many stages does it have?

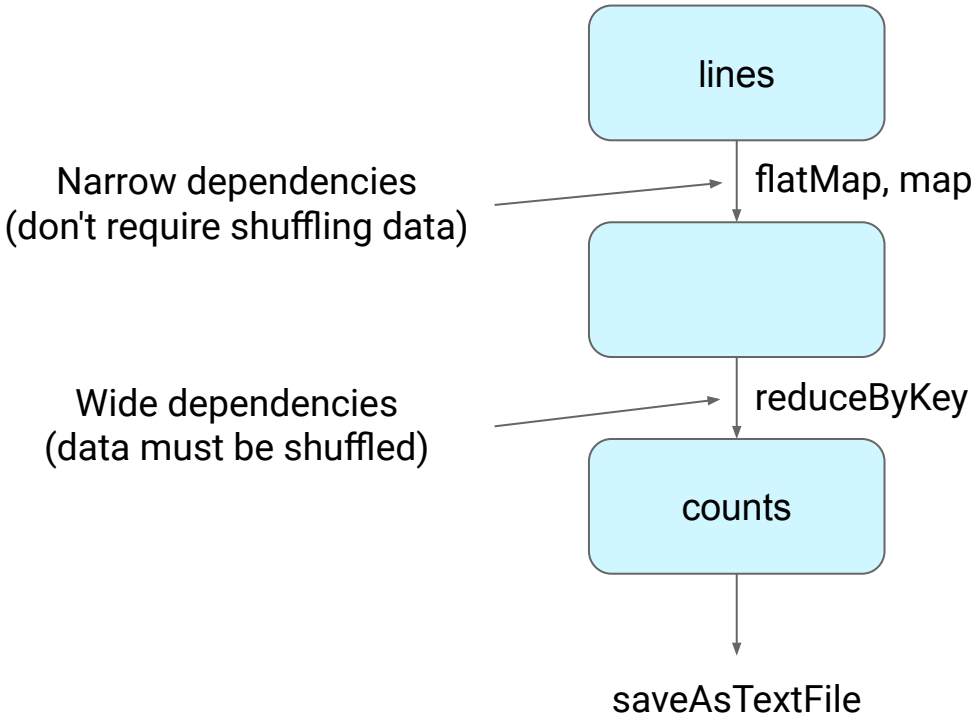
Word Count in Spark



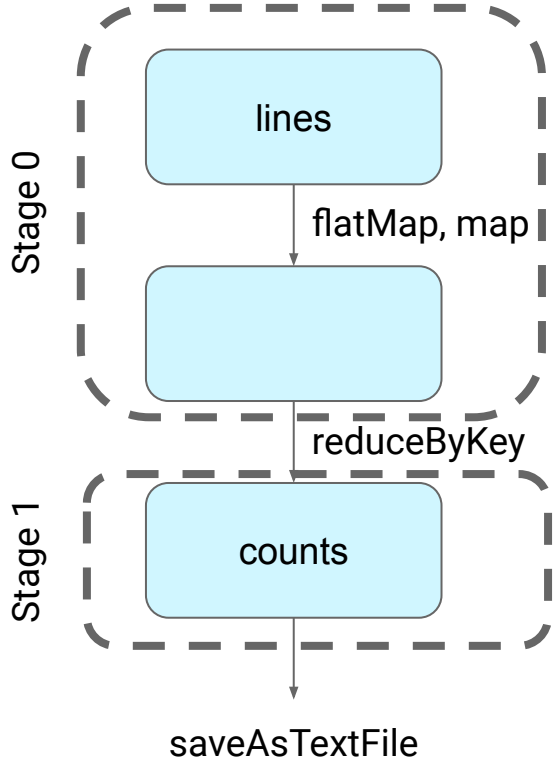
Word Count in Spark



Word Count in Spark



Word Count in Spark



Adding Joins…

```
def countWords(sc, files):
  output = None
  for file in files:
    lines = sc.textFile(file)
    counts = lines.flatMap(lambda x: x.split(' ')) \
                  .map(lambda x: (x, 1)) \
                  .reduceByKey(lambda a,b: a + b)

    if output == None:
      output = counts
    else:
      output = output.fullOuterJoin(counts)
  return output
```

Adding Joins…

```
def countWords(sc, files):
  output = None
  for file in files:
    lines = sc.textFile(file)
    counts = lines.flatMap(lambda x: x.split(' ')) \
                  .map(lambda x: (x, 1)) \
                  .reduceByKey(lambda a,b: a + b)

    if output == None:
      output = counts
    else:
      output = output.fullOuterJoin(counts)
  return output
```

Join the counts of many text files

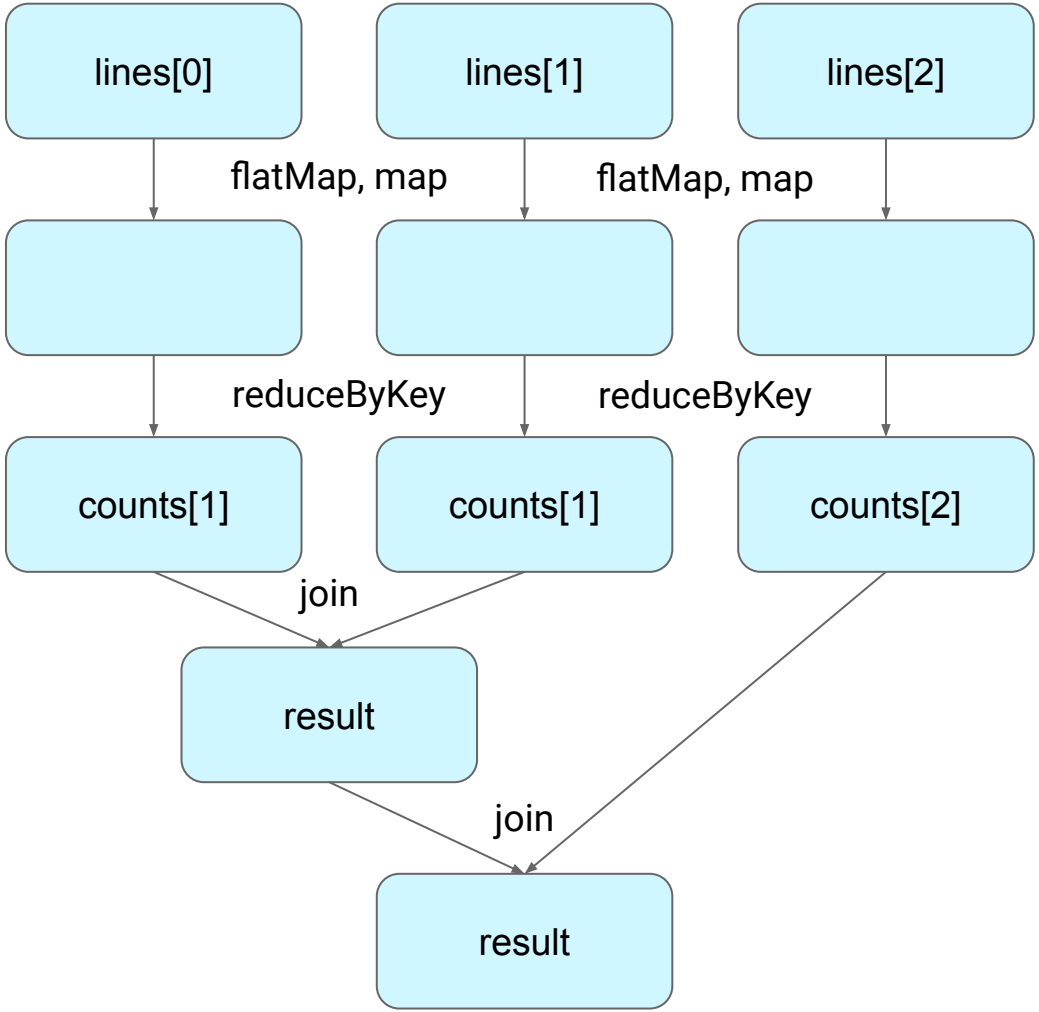
Adding Joins...

Now what does the RDD DAG generated by the word count program look like (let's say for 3 text files)?

How many stages does it have?

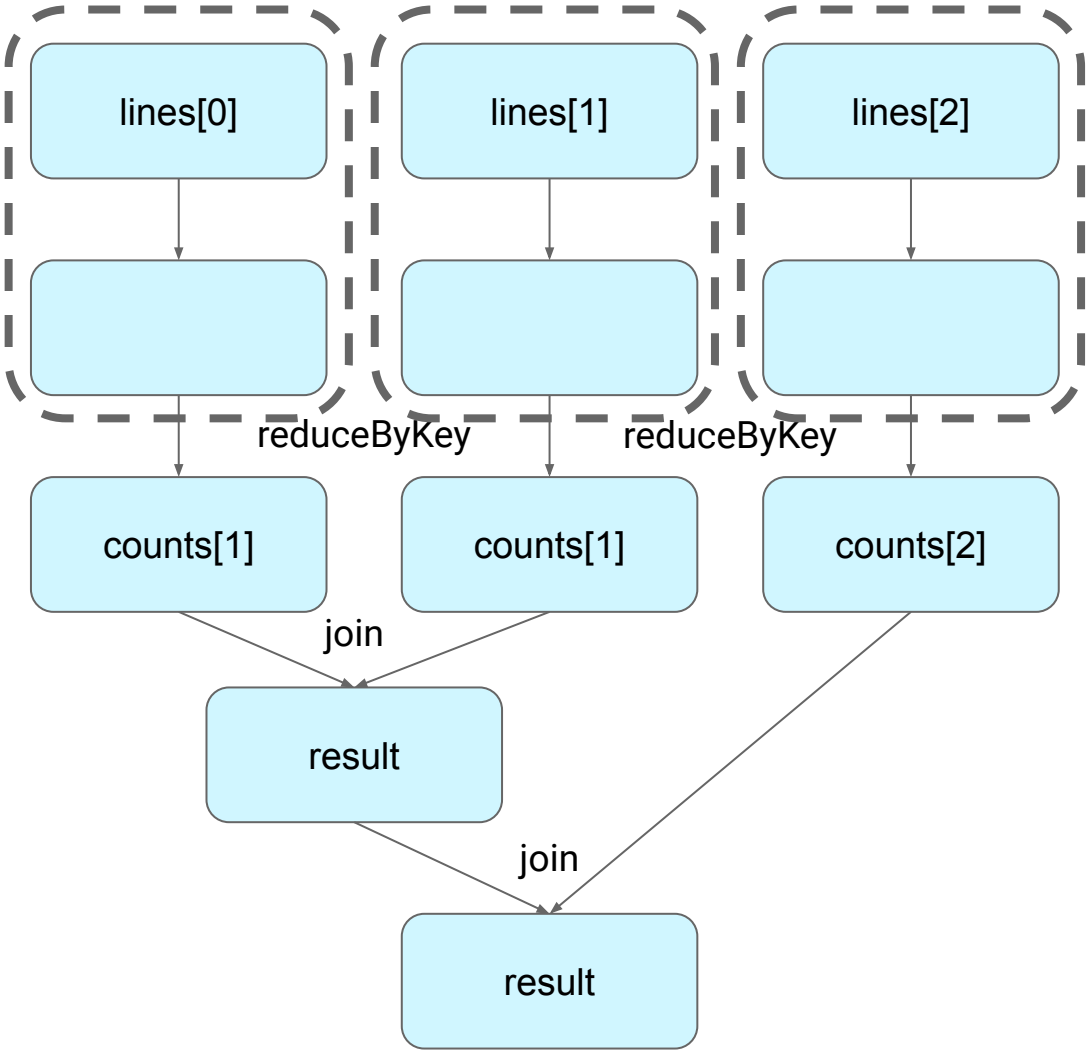
Adding Joins

How is this DAG broken into stages?



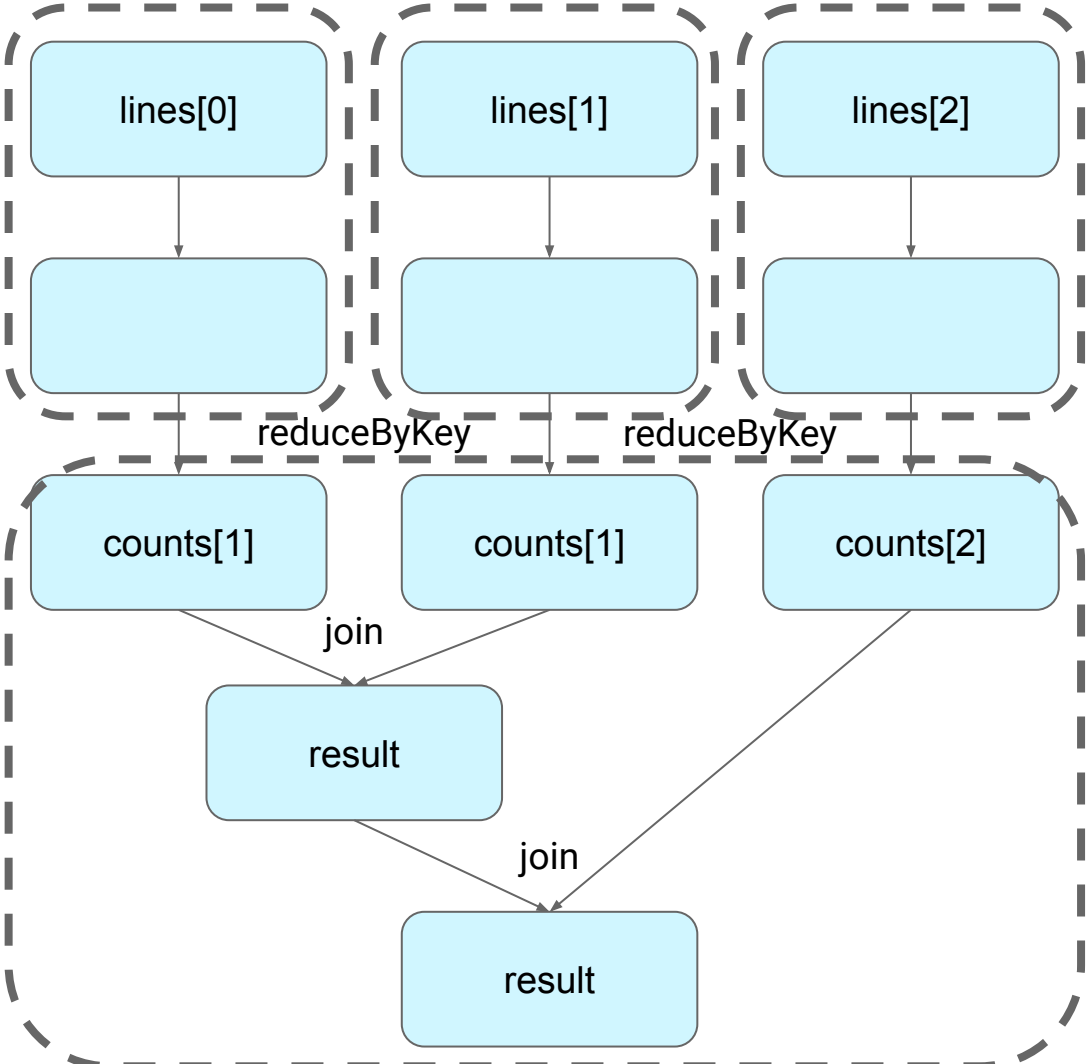
Adding Joins

We still have a stage per reduceByKey…but what about the joins?



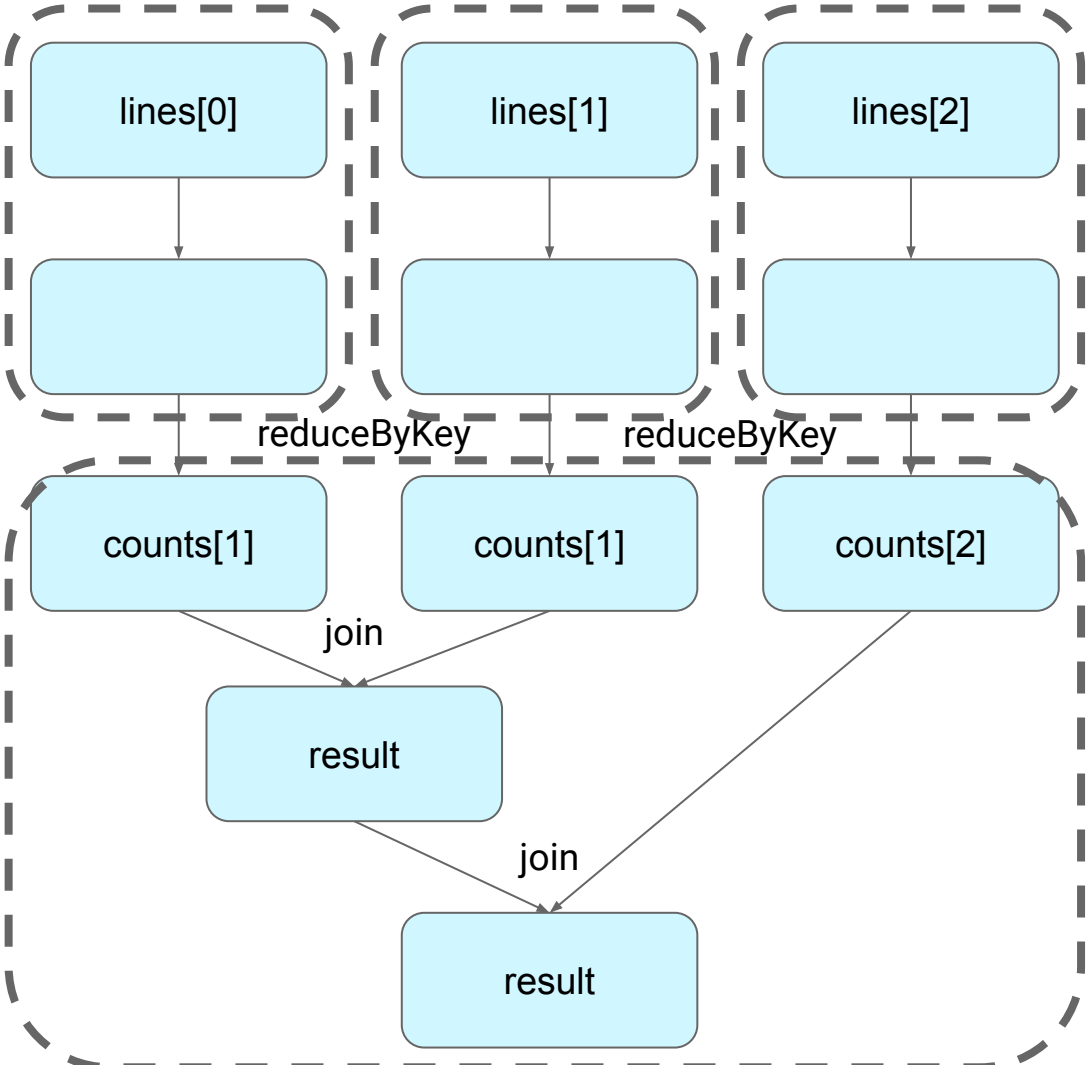
Adding Joins

If the parents have the same partitioning scheme, the joins are narrow and can be in one stage!



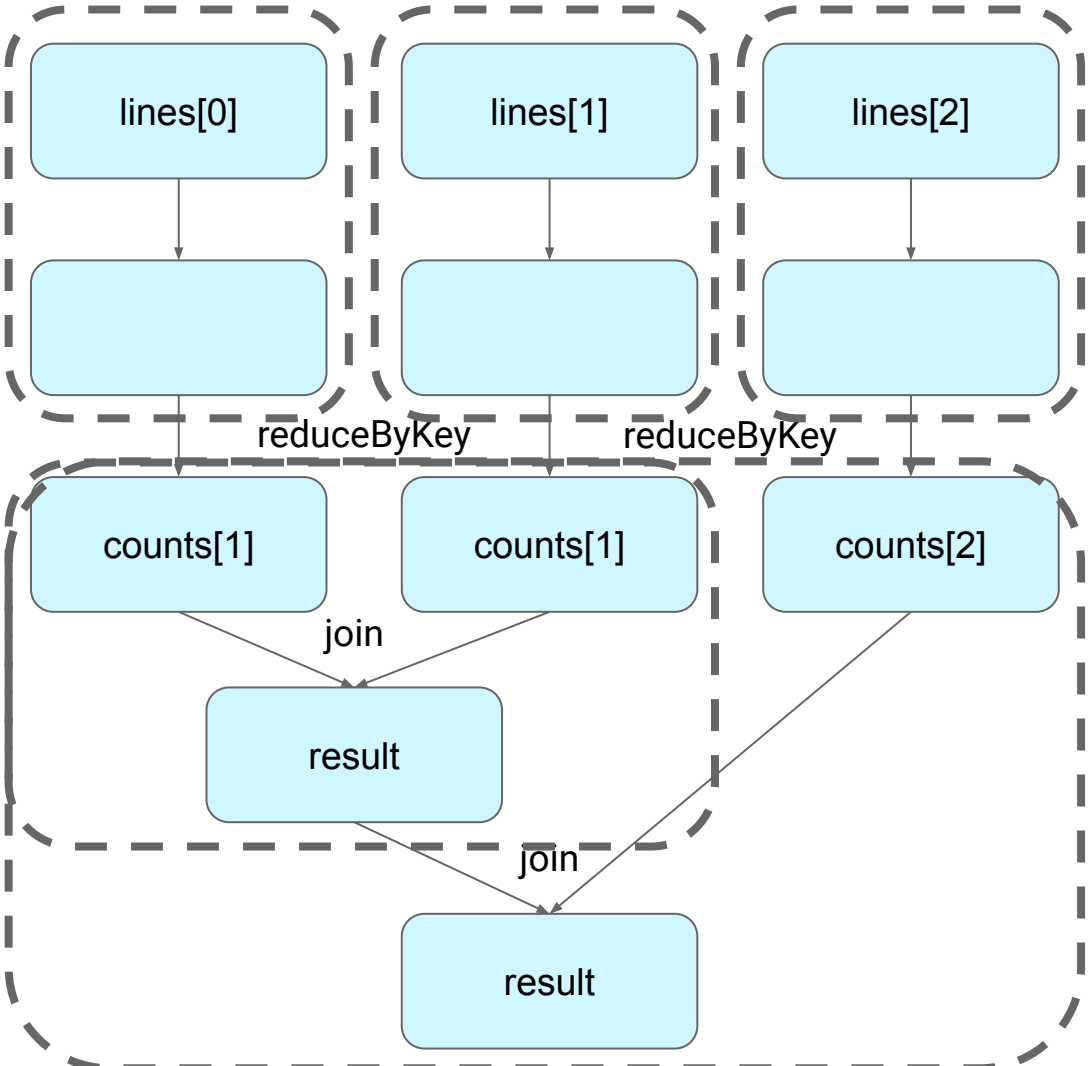
Adding Joins

Also note that due to lazy evaluation, we can include ALL 3 count RDDs in the same stage for joins!



Adding Joins

If the parents do not have the same partitioning, the joins are wide and must be in their own stages.

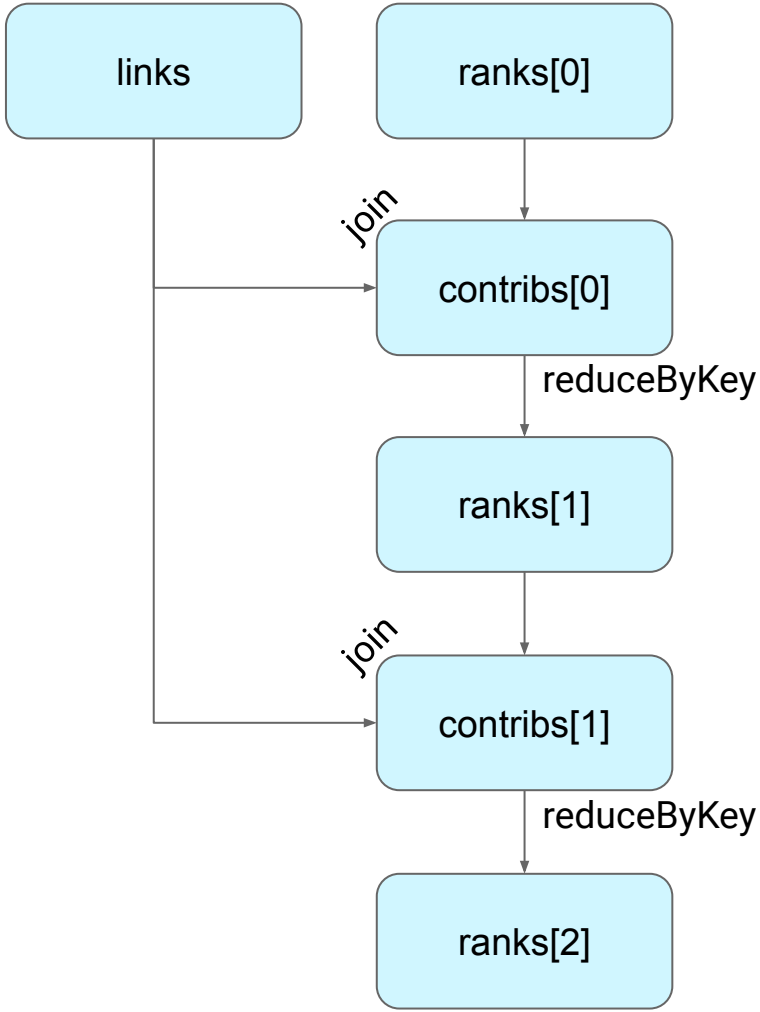


PageRank in Spark

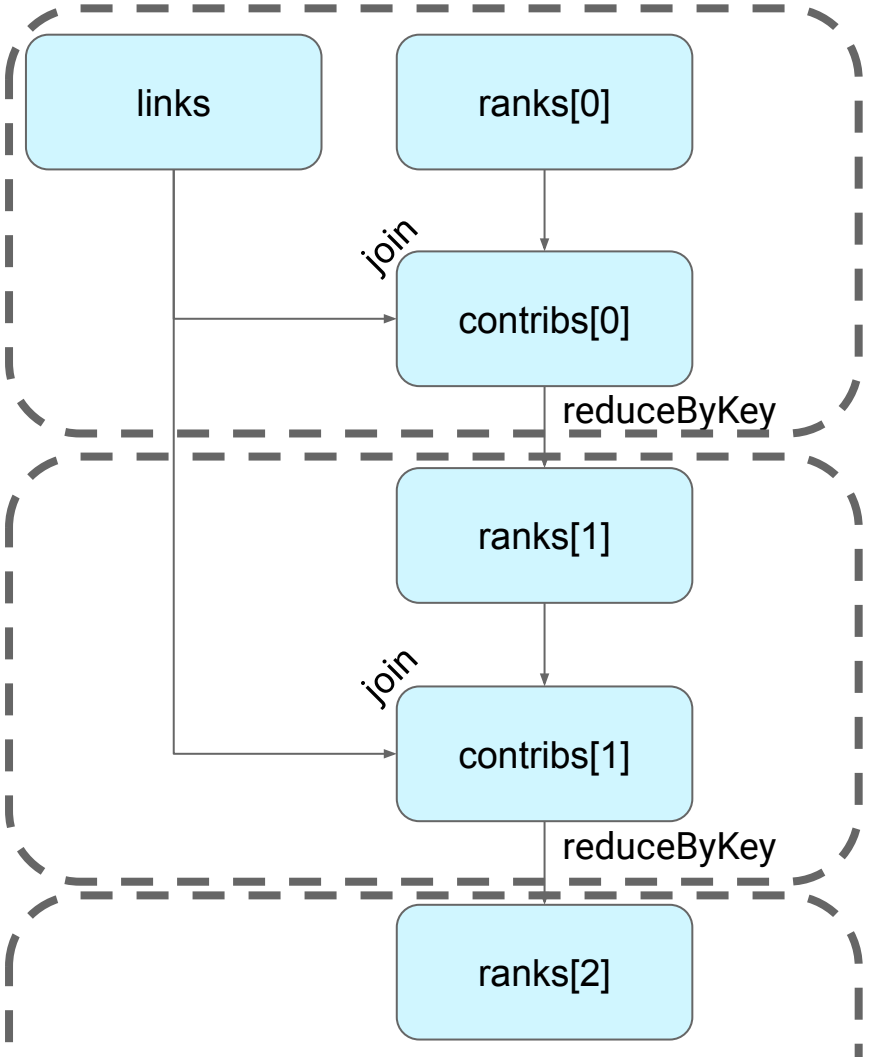
```
def pageRank(sc, file, iters, cache = False):
  lines = sc.textFile(file)
  links = lines.map(lambda urls: parseNeighbors(urls)) \
               .groupByKey()
               .cache()
  N = links.count()
  ranks = links.map(lambda u: (u[0], 1.0/N))

  for i in range(iters):
    contribs = links.join(ranks) \
                    .flatMap(lambda u: computeContribs(u[1][0], u[1][1]))

    ranks = contribs.reduceByKey(lambda a,b: a+b) \
                    .mapValues(lambda rank: rank * 0.85 + 0.15*(1.0/N))
  return ranks
```

PageRank in Spark

Assuming we partition intelligently, how does this DAG get broken into stages?



PageRank in Spark



PageRank in Spark

Note: If links and ranks are partitioned differently, then the joins will need their own stages as well!