CSE 4/587
Data Intensive Computing

Dr. Eric Mikida
epmikida@buffalo.edu
208 Capen Hall
Announcements and Feedback

- Project Phase 3 due Friday
List two benefits that Spark has over MapReduce
List **two** benefits that Spark has over MapReduce

1. Keeps data in memory as much as possible (good for iterative apps)
2. Better for productivity (higher level constructs, more operations)
3. Suitable for entire pipeline (cleaning, EDA, modeling, production)
4. Support for streaming data
Name one other technology in the Hadoop ecosystem that improves programmer productivity with MapReduce
Name one other technology in the Hadoop ecosystem that improves programmer productivity with MapReduce

1. PIG
2. Hive/HBASE
In one sentence explain the primary way fault-tolerance is achieved in MapReduce.
In **one sentence** explain the primary way fault-tolerance is achieved in MapReduce.

Data is divided into blocks, and the blocks are replicated across multiple nodes/racks in the cluster.
In one sentence explain the primary way fault-tolerance is achieved in Spark.
In **one sentence** explain the primary way fault-tolerance is achieved in Spark

The series of transformations used to derive an RDD are stored as a lineage graph that can be re-executed if data is lost.
Explain the difference between a transformation and an action in Spark
Explain the difference between a transformation and an action in Spark

Transformations on an RDD do not trigger any computation. An action requires computation to be performed.
Explain the difference between a narrow dependency and a wide dependency in Spark
Explain the difference between a narrow dependency and a wide dependency in Spark

For a narrow dependency, each parent partition has at most one child partition. For a wide dependency a parent may have multiple child partitions.
Explain the difference between a narrow dependency and a wide dependency in Spark

For a narrow dependency, each parent partition has at most one child partition. For a wide dependency, a parent may have multiple child partitions.

Bonus: What does this mean for how these computations are performed?
Explain the difference between a narrow dependency and a wide dependency in Spark

For a narrow dependency, each parent partition has at most one child partition. For a wide dependency a parent may have multiple child partitions.

Bonus: What does this mean for how these computations are performed?

Narrow can be pipelined. Wide may require data to be shuffled.
Spark HW Q1f(i)

Name one transformation that results in a narrow dependency, draw a DAG
Spark HW Q1f(i)

Name **one** transformation that results in a narrow dependency, draw a DAG

1. map
2. filter
3. union
Spark HW Q1f(ii)

Same as above for wide dependency
Same as above for wide dependency

1. reduceByKey
2. groupByKey
3. join*
lines = sc.textFile(sys.argv[1]).map(lambda r: r[0])
K = int(sys.argv[2])
convergeDist = float(sys.argv[3])
data = lines.map(parseVector).cache()
kPoints = data.takeSample(False, K, 1)
tempDist = 1.0

while tempDist > convergeDist:
    closest = data.map(lambda p: (closestPoint(p, kPoints), (p, 1)))
    pointStats = closest.reduceByKey(lambda p1_c1, p2_c2: (p1_c1[0] + p2_c2[0], p1_c1[1] + p2_c2[1]))
    newPoints = pointStats.map(lambda st: (st[0], st[1][0] / st[1][1])).collect()

    tempDist = sum(np.sum((kPoints[iK] - p) ** 2) for (iK, p) in newPoints)
    for (iK, p) in newPoints:
        kPoints[iK] = p
Given the above spark application, draw the lineage graph DAG for the RDD `newPoints`
Spark HW Q2a

Given the above spark application, draw the lineage graph DAG for the RDD `newPoints`
Identify in the above code one instance of:

i. A transformation that results in a wide dependency
ii. A transformation that results in a narrow dependency
iii. An action
Identify in the above code one instance of:

i. A transformation that results in a wide dependency
   `closest.reduceByKey(...)`

ii. A transformation that results in a narrow dependency
    `data.map(...)`

iii. An action
    `data.takeSample(...), or .collect()`
How many "jobs" will the above code run?
How many "jobs" will the above code run?

1 per action =

1 for takeSample + 1 per iteration for collect until convergence
Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)
Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)

2 stages:

- first stage is map, map, reduceByKey
- second stage is map
Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)

Spark HW Q2d
Spark HW
Q2d

Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage).
Spark HW Q2e

What algorithm is the above code an implementation of?
What algorithm is the above code an implementation of?

k-means clustering
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
Given the above spark application, draw the lineage graph DAG for the RDD `newPoints` ranks
Given the above spark application, draw the lineage graph DAG for the RDD `newPoints` and `ranks`.
Identify in the above code one instance of:

i. A transformation that results in a wide dependency

ii. A transformation that results in a narrow dependency

iii. A transformation that may result in a narrow dependency OR a wide dependency

iv. An action
Identify in the above code one instance of:

i. A transformation that results in a wide dependency
   
   `groupByKey(...) or reduceByKey(...)`

ii. A transformation that results in a narrow dependency
   
   `map(...), flatMap(...), or mapValues(...)`

iii. A transformation that may result in a narrow dependency OR a wide dependency
   
   `join(...)`

iv. An action
   
   `count()`
How many "jobs" will the above code run?
How many "jobs" will the above code run?

1 per action = 1 (just the count action)
Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)

3 stages:

- Stage one contains map and groupByKey
- Stage two contains map, join*, flatMap, and reduceByKey
- Stage three contains mapValues

* assuming that join is narrow
Spark HW Q3d

Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)
Spark HW Q3d

Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)
Spark HW Q3d

Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)

3 stages:

Stage one contains map and groupByKey

Stage two contains map, join*

Stage three contains flatMap, and reduceByKey

Stage four contains mapValues

* assuming that join is wide
Spark HW Q3d

Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)
Spark HW Q3d

Based on your DAG, determine how it is broken up into stages (state the number of stages, and name the transformations in each stage)

For a wide join we need 4 stages
What algorithm is the above code an implementation of?
What algorithm is the above code an implementation of?

PageRank