CS 378 – Big Data Programming

Lecture 27
Partitioning Example,
Aggregation and Broadcast Variables
Review

• Assignment 12
  – Create user sessions
  – Order events by timestamp
  – Order sessions by user ID
  – Partition sessions by referring domain
  – Filter out large sessions (> 1000 events)
Partitioning - Review

• Prudent partitioning can greatly reduce the amount of communication (shuffle)

• If an RDD is scanned only once, no need

• If an RDD is reused multiple times in key-oriented operations
  – Partitioning can improve performance significantly
Partitioning

Figure 4-4, from Learning Spark
Partitioning
Figure 4-5, from Learning Spark
Example - Page Rank

• Walk through page rank algorithm for Spark

• See a more complex algorithm using Spark
  – Iterative

• Show benefits of partitioning, persistence
What is Page Rank?

Algorithm for weighting linked documents
Part of Google’s ranking algorithm – lots of other stuff included

Basic idea
Rank++ for inbound links
Rank++ for high rank links

In this image:
Size proportional to # inbound links

Basic Page Rank Algorithm
From Learning Spark, pp. 66-67

• Give each page an initial rank of 1

• On each iteration, have page $p$ send a contribution of $\frac{\text{rank}(p)}{\text{numNeighbors}(p)}$ to its neighbors

• Set each page’s rank to
  \[ 0.15 + 0.85 \times \text{contributionsReceived} \]
Page Rank - Example

Accumulators

• In our session generator app,
• Suppose we wanted to count the number of sessions that are filtered due to size (> 1000)

• How would we do this?

• How did we do this using Hadoop map-reduce?
Accumulators

• An accumulator provides a means for aggregating values from worker nodes back to the driver node.

• Create an accumulator from the context

• Increment the accumulator in functions passed to worker nodes
Accumulators

• For failures or re-evaluation, what happens?

• Actions:
  – Each task’s update applied only once

• Transformations:
  – No guarantee that task updates applied only once
  – Re-evaluation will update accumulator each time
Broadcast Variables

• If you want to access a read-only data structure from multiple transformations
  – It will be wrapped into each closure
  – Wasteful if the data is large

• A broadcast variable addresses this issue
  – Sent to each worker node only once
  – Accessible from closures sent to the workers
  – Data must be serializable