**CS5543 Real-Time Big Data Analytics**

**MapReduce & Spark Programing**

ICE-6

10/4/2016

Name: Bill Capps

Class ID:

**MapReduce & Spark Programming: Compute Single Source Shortest Path**

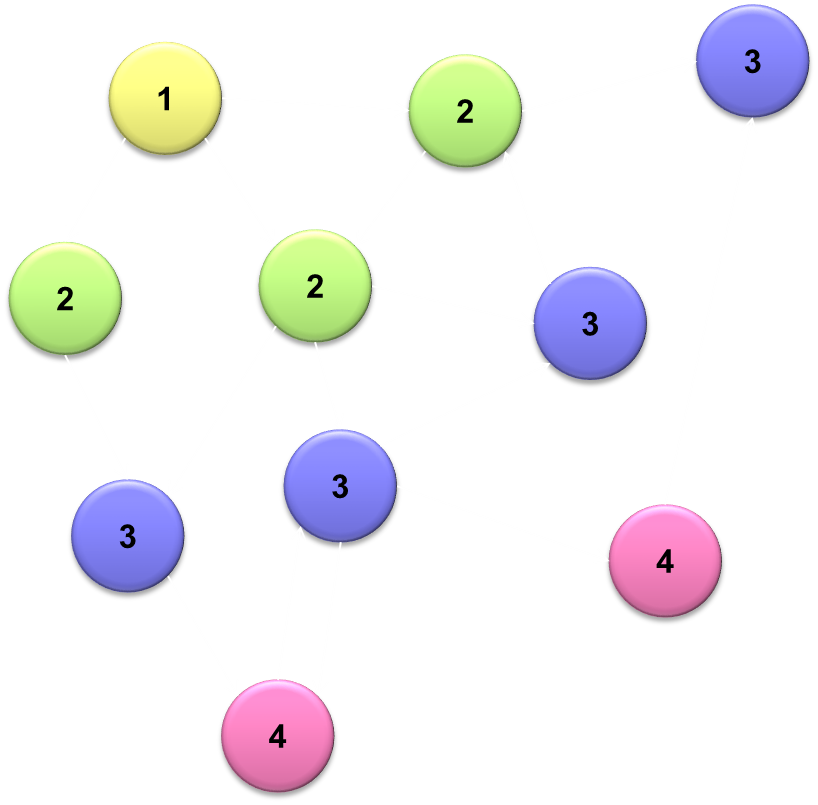
**Problem:** find shortest path from a source node to one or more target nodes.

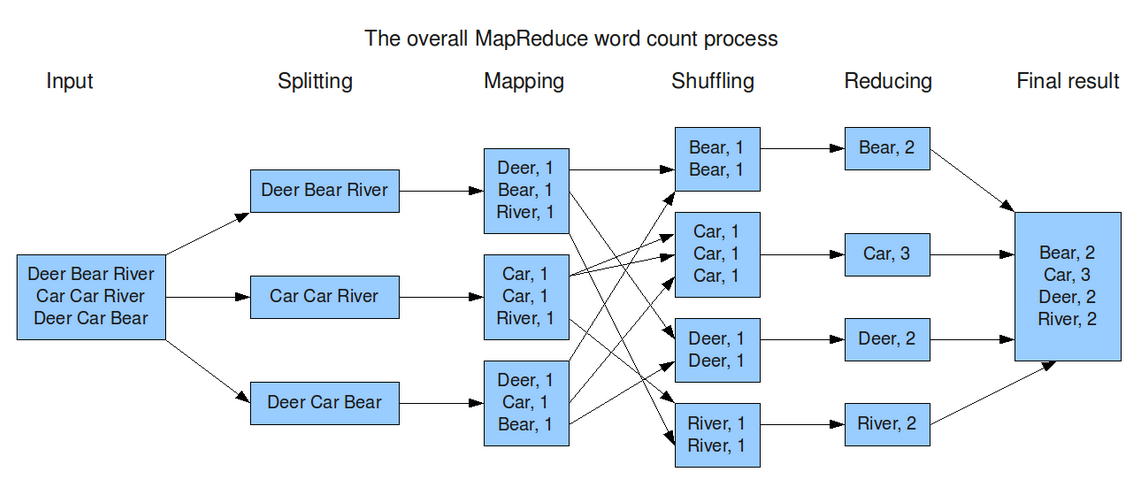
Given the problem description,

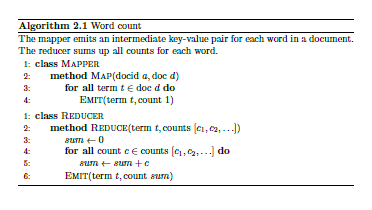
1. Draw the MapReduce Diagram.
2. Sketch the MapReduce Algorithm.
3. Sketch the Spark Scala implementation
4. Sketch the Storm Topology

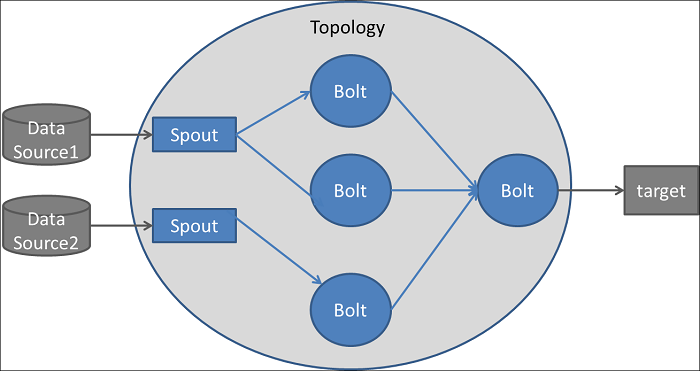
**Example: Visualization of Parallel Breadth-First Search**

**Hint:** MapReduce: parallel Breadth-First Search (BFS). For the sake of simplicity, consider equal edge weights. Solution to the problem can be defined inductively. Multiple iterations are needed to explore entire graph.

****

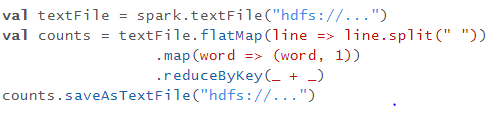






Storm Topology Example

**Spark Scala Code for WordCount**



|  |  |
| --- | --- |
| **flatMap(func)** | Similar to map, but each input item can be mapped to 0 or more output items (so func should return a Seq rather than a single item). |
| **reduceByKey**(*func*, [*numTasks*]) | When called on a dataset of (K, V) pairs, returns a dataset of (K, V) pairs where the values for each key are aggregated using the given reduce function *func*, which must be of type (V,V) => V. Like in groupByKey, the number of reduce tasks is configurable through an optional second argument. |