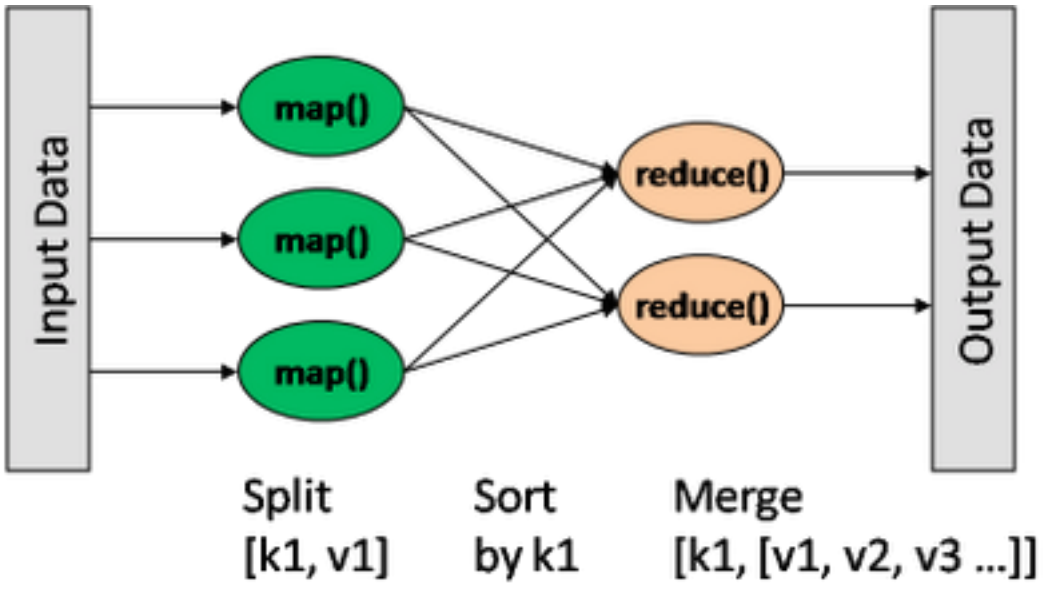
Graph Algorithms (Shortest path and Matching) with MapReduce in Cloud

Ariful Azad, Department of CS, Purdue University

---

1. **Motivation:**
    1. **What is cloud computing to me? :** When I took the course I did not have any concrete idea on cloud computing. As I learnt the idea it seems more and more an economical shift rather than a new computational paradigm from the programmer perspective. The programming methodology in cloud is still similar to distributed / parallel computing.
    2. **How cloud computing can contribute to my research?** : My research interest lies in application of graph algorithm in computational biology. Security and privacy is a concern to me but is not related to my research. I am more interested in developing algorithms for very large-scale datasets that can run efficiently and fast on cloud (preferably with low cost). When I search on the web about cloud computing application a good number of the papers show up mainly based on large-scale data analysis using some form of Google's MapReduce. I decided to learn it and the simplicity of the concept attracts me. However, the technique is difficult to apply in graph algorithms since graph partitioning is a nontrivial to parallelize. I found couple of resources and was interested to investigate it more. This guide me to chose this project!

2. **Problem Description:** MapReduce is a framework that was developed at Google for processing large amount of data (>1TB) that are distributed across thousand of machines.



Properties:

1. Mappers and Reducers run in parallel. No dependency between difference instances of mapper (and instances of reducer as well). For graph application this is very difficult to achieve since often there is a dependency between different parts.
2. Sometimes it is impossible to implement the algorithm in a single map-reduce stage. For example single source shortest path. In that case we require multiple map-reduce stages.

In this project I will implement single source shortest path and a graph-matching algorithm using MapReduce.

3. **Scope of the project:**
    1. Install a library that will help running MapReduce at Amazon and possibly at Yahoo Cloud. The obvious choice is free framework Apache's Hadoop.
    2. Develop the map and reduce function for identifying shortest path in a very large graph.
    3. Try to improve performance (challenge: Is there any other program for large scale to compare??). Idea of the algorithm is found from a slide in Google Code University, no paper found.
    4. Develop the map reduce function for a matching algorithm in large bipartite graph.

4. **What I have done so far:** I have read 2 papers and 3 slides on this. I have also watched 3 lectures on MapReduce on Google Code University. Now I have concrete idea about what to develop. I also set up a Hadoop image on my local machine for local testing of a program. However, Hadoop is still need to be installed in a real cloud. I ma planning to install it in Amazon EC2 and Yahoo cloud.

5. **Previous works:** This is a super active research area now since I found so many recent research papers on MapReduce. However, not many works focus on dependent data like graphs. Frankly I found 2/3 papers and 3 slides on this. Those will be my starting point.

6. **References:**

    1. A very practical introduction to MapReduce: http://code.google.com/edu/parallel/mapreduce-tutorial.html
    2. Seminal paper on MapReduce: Dean, Jeff and Ghemawat, Sanjay. **MapReduce: Simplified Data Processing on Large Clusters.**
    3. Lectures on graph algorithms using MapReduce in Google code university: http://code.google.com/edu/submissions/mapreduce-minilecture/listing.html
    4. Hadoop Summit 2010 presentation by Sergei Vassilvitskii http://www.slideshare.net/ydn/3-xxl-graphalgohadoopsummit2010

5. MapReduce using Hadoop: http://hadoop.apache.org/mapreduce/
6. H, Karloff, S. Suri, S. Vassilvitskii, “A model of Computation for MapReduce”.
7. U Kang, C. Tsourakakis, A. Appel, C. Faloutsos, J. Leskovec “HADI: Fast diameter estimation and mining in massive graphs with Hadoop”. CMU Dec 08