Tutorial for Assignment 2.0

Web Science and Web Technology

Summer 2011

Slides based on last years tutorial by Florian Klien and Chris Körner

IMPORTANT

- The presented information has been tested on the following operating systems
  - Mac OS X 10.06
  - Ubuntu and Debian Linux
- The installation on Windows machines will not be supported by us in the newsgroup and is highly not recommended
- As always: Plagiarism will not be tolerated!!!!!

Agenda

- Review and Motivation
- Introduction to Hadoop and Map/Reduce
- Example Map/Reduce Application
- Assigment Information
- Setup pitfalls and hints

Review

What you should have learned so far

- Network analysis and operations
  - Such as degree distribution
  - Clustering Coefficient
  - Google‘s PageRank
  - Network Evolution

→ Computed for very small networks

Motivation

- So far these analyzes do NOT scale
- What about networks with a huge amount of nodes and edges or GB/TB of data?
- Computation would take quite a long time
- How can we process large amounts of data?

→ Hadoop

Apache Hadoop

- One solution of the scaling problem
- Using the Map/Reduce paradigm
- Written in Java (but also other programming languages are possible)
- Used by Yahoo, Amazon etc.

Map/Reduce 1/2

- Framework to support distributed computing of large data sets on clusters

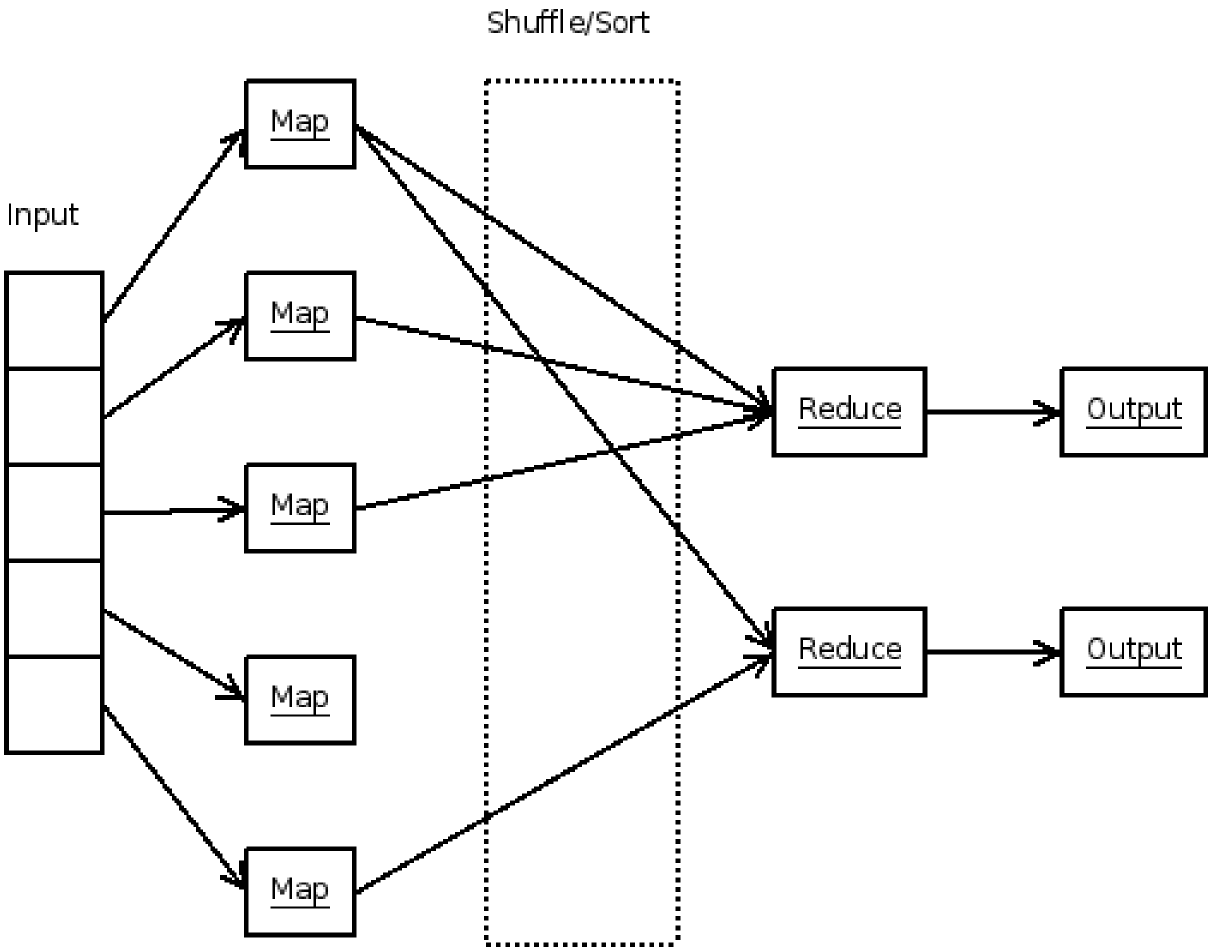
- Used for data-intensive information processing

- Large Files/Lots of computation

Map/Reduce 2/2

Abstract view:

- Master splits problem in smaller parts
- Mapper solve sub-problem
- Reducer combines results from Mappers



http://people.apache.org/~rdonkin/hadoop-talk/hadoop.html

Distributed File System (DFS)

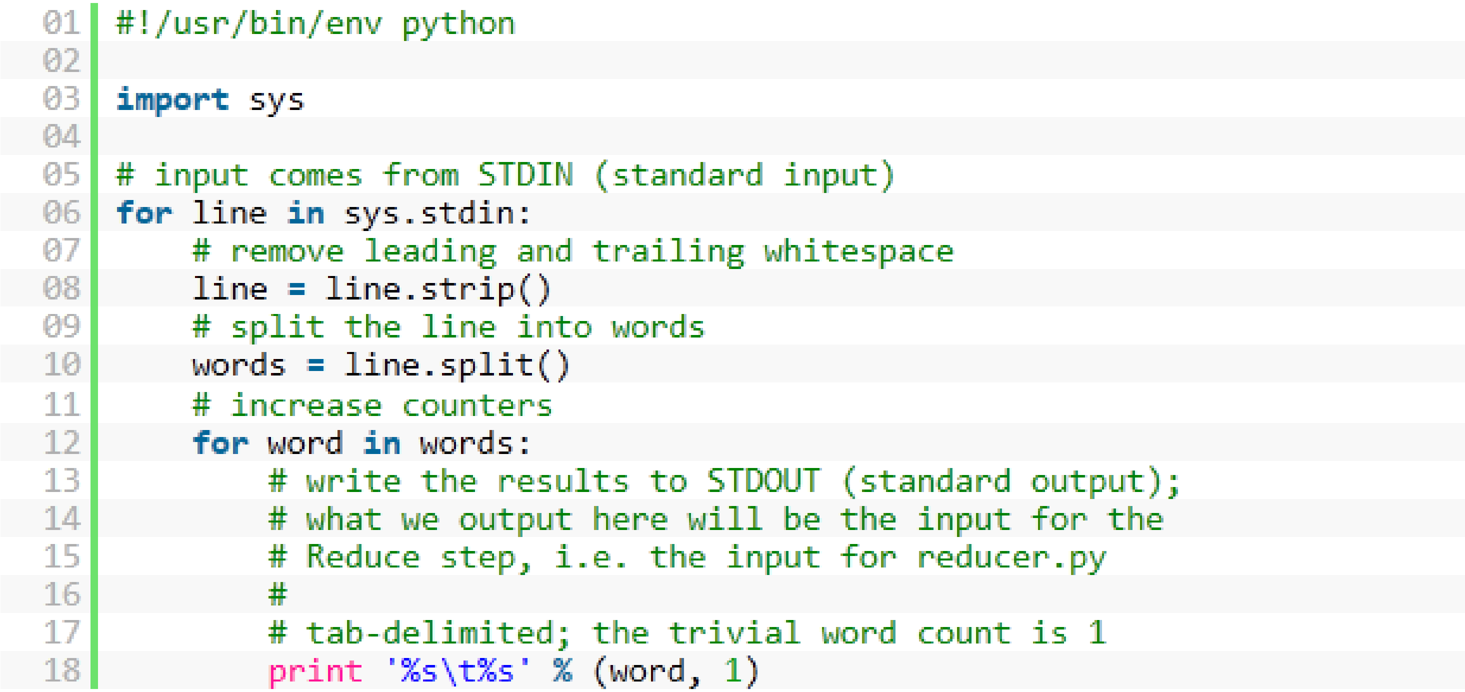
- Hadoop comes with a distributed file system
- Highly fault tolerant
- Splits data in blocks of 64mb (default configuration)

Example of a Map/Reduce Application 1/4

- Word Count
  - Counting occurrences of words on lots of documents
- To keep things simple we will use the example from [1]
  - uses Python
  - reads from StdIn
  - writes to StdOut

Example of Map/Reduce Application 2/4

Mapper



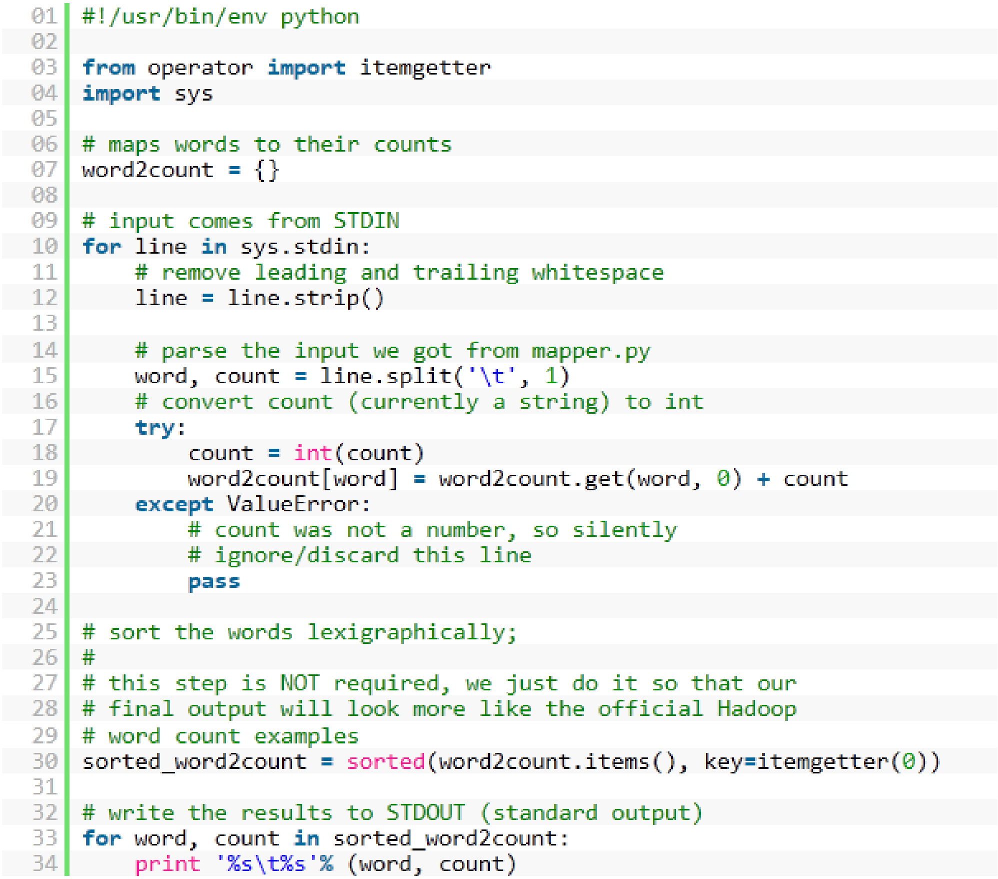
```python
#!/usr/bin/env python

import sys

# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    words = line.split()
    # increase counters
    for word in words:
        # write the results to STDOUT (standard output);
        # what we output here will be the input for the
        # Reduce step, i.e. the input for reducer.py
        #
        # tab-delimited; the trivial word count is 1
        print '%s\t%s' % (word, 1)
```

Example of Map/Reduce Application 3/4

Reducer



```python
#!/usr/bin/env python

from operator import itemgetter
import sys

# maps words to their counts
word2count = {}

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()

    # parse the input we got from mapper.py
    word, count = line.split('\t', 1)
    # convert count (currently a string) to int
    try:
        count = int(count)
        word2count[word] = word2count.get(word, 0) + count
    except ValueError:
        # count was not a number, so silently
        # ignore/discard this line
        pass

# sort the words lexigraphically;
#
# this step is NOT required, we just do it so that our
# final output will look more like the official Hadoop
# word count examples
sorted_word2count = sorted(word2count.items(), key=itemgetter(0))

# write the results to STDOUT (standard output)
for word, count in sorted_word2count:
    print '%s\t%s'% (word, count)
```

Example of Map/Reduce Application 4/4

- It is always recommended to test the code you have written on a small sample subset
  - Think through with pen & paper and compare results
  - Example: cat subset.txt | python mapper.py | python reducer.py

- Run the code on the cluster by issuing:

```
bin/hadoop jar contrib/streaming/hadoop-0.20.0-streaming.jar -file /home/hadoop/mapper.py -mapper /
home/hadoop/mapper.py -file /home/hadoop/reducer.py -reducer /home/hadoop/reducer.py -input $input
-output $output
```

The Assignment

- Team up in groups of 5 students
- Nominate group captain
- Create Subversion repository (ADD ALL TUTORS AS READERS)
- Implement TunkRank and compute it on the provided data
- You do not have to solve it in one step – just explain it in the Readme file
- Hand in your source code and the top 10.000 Twitter users in descending order + Tunkrank score
- See assignment document for further details

Provided Data

- You are given a subset of a large Twitter data set which was gathered for a scientific paper [2]
  - Compressed 782MB

- Tab seperated:
  - First column: Users
  - Second column: Follower (user who follows user from first column)

TunkRank 1/2

- Tool to measure the influence on Twitter
- The higher the TunkRank score is the more influential a Twitter user is
- Twitterers with high TunkRank
  - Barack Obama
  - Charlie Sheen
  - Ashton Kutcher
- See http://www.tunkrank.com or [3] for details

$$Influence(X) = \sum_{Y \in Followers(X)} (1 + p * Influence(Y)) / ||Following(Y)||$$

TunkRank 2/2

$$Influence(X) = \sum_{Y \in Followers(X)} (1 + p * Influence(Y))/||Following(Y)||$$

Influence(X) = Expected number of people who will read a tweet that *X* tweets, including all retweets of that tweet. For simplicity, we assume that, if a person reads the same message twice (because of retweets), both readings count.

If *X* is a member of *Followers(Y)*, then there is a *1/||Following(X)||* probability that *X* will read a tweet posted by *Y*, where *Following(X)* is the set of people that *X* follows.

If *X* reads a tweet from *Y*, there's a constant probability *p* that *X* will retweet it.

Hand In 1/2

- Create a Subversion repository on the TUG server
- Name: WSWT11_<GROUPNAME>
- Group members as members
- Teaching assistents as readers

Hand In 2/2

Structure of the repository

- Report.pdf (short – approx. 1 page)
- Bash scripts (optional)
- python/
  - mapper_1.py
  - reducer_1.py
  - …
  - readme.txt
- results/
  - tunkrank_run_1.txt (top 10.000 Twitterers in descending order + their TunkRank score)

Important Dates

- NOW: Team up in groups of 5
- Assignment is due: Monday June 6, 2011
  - 12:00 (noon) – soft deadline
  - 24:00 – hard deadline
- „Abgabegespräche“ will be on Tuesday June 7, 2011
  - Every team member has to attend

Hadoop Setup 1/2

- Create new user „hadoop“ on your system
- Use functioning DNS or /etc/hosts file for client/master lookup
- Download current Hadoop distribution from http://hadoop.apache.org
- Unpack distribution in a directory

  (e.g. /usr/local/hadoop)
- Create temp directory

  (e.g. /usr/local/hadoop-datastore)

Hadoop Setup 2/2

- conf/hadoop-env.sh - holds environment variables and java installation
- conf/core-site.xml - names the host the default file system & temp data
- conf/mapred-site.xml - specifies the job tracker
- conf/masters - names the masters
- conf/slaves (only on master necessary) - names the slaves
- conf/hdfs-site.xml - specifies replication value

- Format DFS
  - bin/hadoop namenode -format

Starting the Hadoop Cluster

- bin/start-dfs.sh starts HDFS daemons
- bin/start-mapred.sh - starts Map/Reduce daemons
- alternative: start-all.sh
- stopper scripts also available

Pitfalls for the Setup of Hadoop

- Use machines of approximately the same speed / setup
- Use the same directory structure for all installations of your machines
- Ensure that password-less ssh login is possible for all machines
- Avoid the term localhost and the ip 127.0.0.1 at all cost --> use fixed IPs or functioning DNS for your experiments
- Read the Log files of the Hadoop installation
- Use the web interface of your cluster

Further hints

- Check if enough free space is available on your harddisk partition (~15GB would be recommended)
- Virtual Machines
  - Same as above: give the machine enough space
  - Give the machine a good amount of memory (~1024MB)
  - For local networks: Use bridging (no NAT!!!)
- Read the tutorials carefully [1]
- Post your problems to the newsgroup

Thanks for your attention!

Are there any questions?

References

[1] Michael G. Noll's Hadoop Tutorial:

Single Node Cluster

http://www.michael-noll.com/wiki/Running_Hadoop_On_Ubuntu_Linux_%28Single-Node_Cluster%29

Multi Node Cluster

http://www.michael-noll.com/wiki/Running_Hadoop_On_Ubuntu_Linux_%28Multi-Node_Cluster%29

Writing Map/Reduce Program in Python

http://www.michael-noll.com/wiki/Writing_An_Hadoop_MapReduce_Program_In_Python

[2] H. Kwak, C. Lee, H. Park, and S. Moon. What isTwitter, a social network or a news media? In WWW'10: Proceedings of the 19th international conference on World wide web, pages 591–600, New York, NY, USA, 2010. ACM.

[3] http://thenoisychannel.com/2009/01/13/a-twitteranalog-to-pagerank/