Tutorial for Assignment 2.0

Web Science and Web Technology

Summer 2012

Slides based on last years tutorials by Chris Körner, Philipp Singer

Agenda

- Review and Motivation
- Assignment Information
- Introduction to Hadoop and MapReduce
- Example MapReduce Application
- 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 analyses 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

Provided Data

You are given a *subset* of a large delicious data set which was gathered by Arkaitz Zubiaga

- Compressed 2.7GB [8.3GB]

  102.090.935 entries on the top 10.000 tags
- minimum number of 3 tags per entry

- Tab separated:
  - An entry represents a delicious bookmark:

```
url tags[]
```

Co-occurrence

- According to wikipedia co-occurence is...

„the above-chance frequent occurrence of two terms from a text corpus alongside each other in a certain order“ [3]

- Example

<u>Given:</u> [dog, cat, animal] [cat, lion, animal] [cat, lion, zoo]

<u>Result:</u>

```
animal cat 2 dog 1 lion 1
cat animal 2 dog 1 lion 2 zoo 1
dog animal 1 cat 1
lion animal 1 cat 2 zoo 1
zoo cat 1 lion 1
```

The Assignment

- Team up in groups of 5 students
- Nominate group captain
- Create Subversion repository (ADD ALL TUTORS AS READERS)
- Implement co-occurrence calculation for the given dataset and compute it
- You do not have to solve it in one step – just explain it in the readme.txt file
- Hand in your source code, a co-occurrence adjacency list of 10 tags and visualize them!
- See assignment document for further details

Hand In 1/2

- Create a Subversion repository on the TUG server
- Name: WSWT12_<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/
  - co-occurences.txt (adjacency list of 5 given tags + 5 of own choice)
  - tag_cloud_*.png (tag cloud visualization of the 10 tags)

IMPORTANT

- NOW: Team up in groups of 5
- Assignment is due: Monday June 11, 2012
  - 12:00 (noon) – soft deadline
  - 24:00 – hard deadline
- „Abgabegespräche“ will be on Tuesday June 12, 2012
  - Every team member has to attend
- As always: Plagiarism will not be tolerated!!!!!

Apache Hadoop

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

08.05.2012 09:38 heise Developer « Vorige | Nächste »

Studie: Hadoop wird ähnlich erfolgreich wie Linux

vorlesen / MP3-Download

Die Marktforscher von International Data Corporation (IDC) gehen in einer Studie davon aus, dass der Markt rund um das MapReduce-Framework Hadoop bis 2016 auf etwas mehr als 810 Millionen US-Dollar anwachsen werde. Das entspricht offenbar einer jährlichen Steigerungsrate von rund 60 Prozent. Für das vergangene Jahr macht das Marktforschungsunternehmen einen 77 Millionen US-Dollar schweren Markt aus.

Das in Java geschriebene "Big Data"-Framework hat seine Heimat in der Apache Software Foundation. Es liegt seit dem letzten Jahreswechsel in Version 1.0 vor und wurde speziell zur Entwicklung skalierbarer, verteilter Anwendungen zur Analyse großer Datenmengen ausgelegt. Es implementiert den von Google entwickelten MapReduce-Algorithmus. Dabei werden Daten über die Knoten eines Hadoop-Clusters verteilt redundant gespeichert, die Analyseprogramme laufen parallel auf allen Knoten und analysieren die jeweils dort vorgehaltenen Daten.

Die Zahl für 2011 bewertet IDC als geringe Summe, da das Thema Big Data ja seit einiger Zeit schon in aller Munde sei. Allerdings prognostizieren die Marktforscher, dass die Zahlen dem großen Interesse am Thema schon bald entsprechen würden. Sie gehen gar so weit, den Aufschwung des Hadoop-Ökosystems mit der Entwicklung von Linux zu vergleichen, das seinerzeit ebenfalls eine überdurchschnittlich große mediale Aufmerksamkeit auf sich zog und mittlerweile in nahezu jedem Rechenzentrum zu finden sei.

http://www.heise.de/newsticker/meldung/Studie-Hadoop-wird-aehnlich-erfolgreich-wie-Linux-1569837.html (de)

http://www.h-online.com/open/news/item/Analyst-Hadoop-ecosystem-to-be-worth-812-million-in-2016-1570381.html (en)

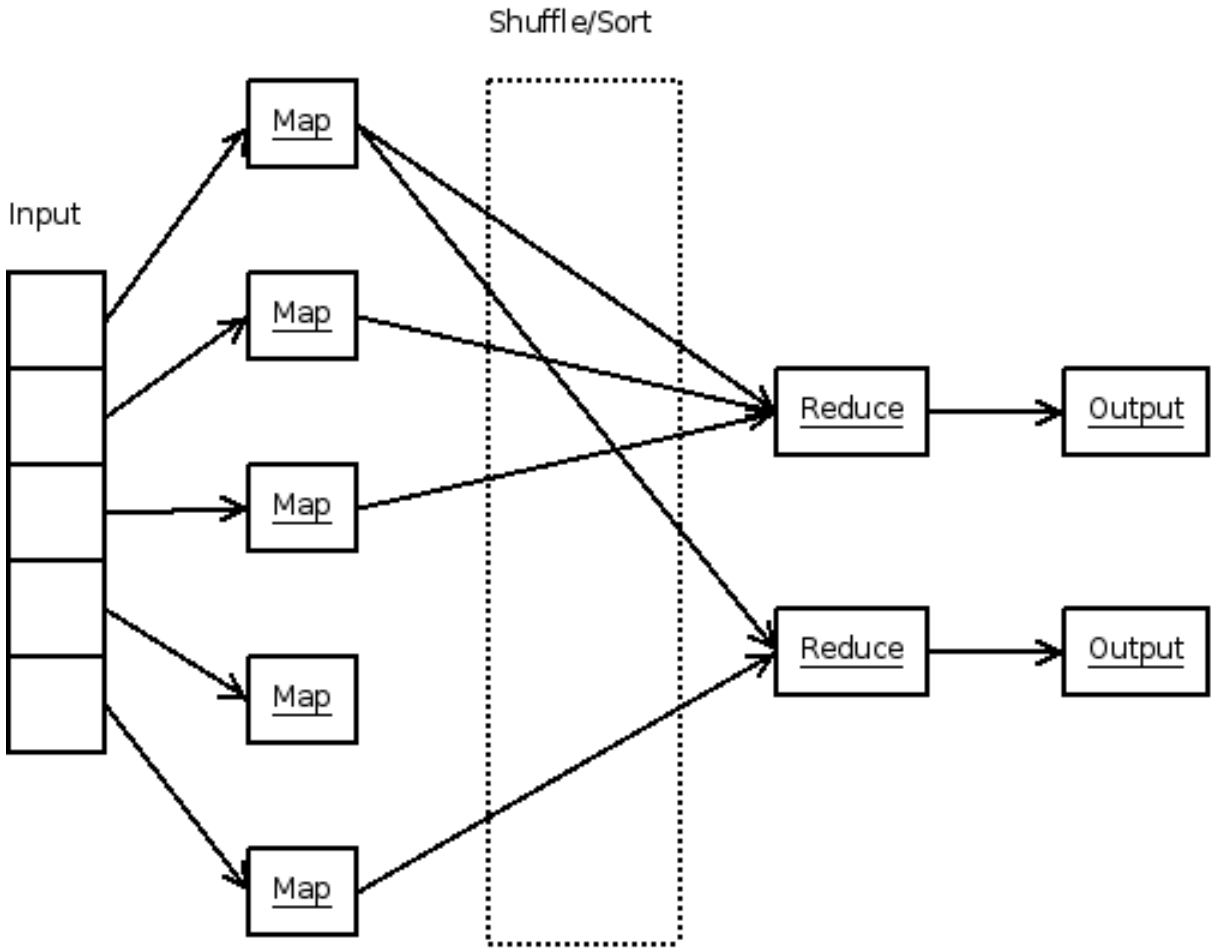
MapReduce 1/2

- Framework to support distributed computing of large data sets on clusters
- Developed by Google Inc.
- Used for data-intensive information processing
- Large Files/Lots of computation

MapReduce 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 MapReduce Application 1/4

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

Example of MapReduce Application 2/4

Mapper

```
01 #!/usr/bin/env python
02
03 import sys
04
05 # input comes from STDIN (standard input)
06 for line in sys.stdin:
07     # remove leading and trailing whitespace
08     line = line.strip()
09     # split the line into words
10     words = line.split()
11     # increase counters
12     for word in words:
13         # write the results to STDOUT (standard output);
14         # what we output here will be the input for the
15         # Reduce step, i.e. the input for reducer.py
16         #
17         # tab-delimited; the trivial word count is 1
18         print '%s\t%s' % (word, 1)
```

Example of MapReduce Application 3/4

Reducer

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

from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

# 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)
    except ValueError:
        # count was not a number, so silently
        # ignore/discard this line
        continue

    # this IF-switch only works because Hadoop sorts map output
    # by key (here: word) before it is passed to the reducer
    if current_word == word:
        current_count += count
    else:
        if current_word:
            # write result to STDOUT
            print '%s\t%s' % (current_word, current_count)
        current_count = count
        current_word = word

# do not forget to output the last word if needed!
if current_word == word:
    print '%s\t%s' % (current_word, current_count)
```

Example of MapReduce 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 | sort | python reducer.py

- Run the code on the cluster by issuing:

```
bin/hadoop jar contrib/streaming/hadoop-streaming-1.0.0.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
```

Important

- The presented information has been tested on the following operating systems
  - Ubuntu Linux
  - Debian Linux
- The installation on Windows machines will not be supported by us in the newsgroup and is highly not recommended
- We also cannot support Mac OS X (although the setup should work fine)

Hadoop Setup 1/2

- Use a **native** Linux (~100-120GB free space)
- Do the Multi-Node-Cluster Setup
  - For runtime reasons

As described in the tutorial by Michael G. Noll [1]:

- 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)

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 (~100-120GB are needed)
- Virtual Machines (DO NOT DO IT!)
  - 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
  - Read the logfiles (hadoop/logs/)
  - Often google offers you a solution/hint to your problem

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://en.wikipedia.org/wiki/Co-occurrence