Hadoop: Finde das längste Wort

Korch, Pascal Florian
s0568195@htw-berlin.de

Kaminski, Elias
s0573558@htw-berlin.de

04.02.2023

1 Einleitung

Hadoop ist ein System zur effizienten Speicherung und Verarbeitung von großen Datenmengen . Als eines der ersten Open Source Projekte in diesem Bereich gilt es als Initiator der Big Data Ära.[4] "Ursprünglich von Google Inc. designed, wurde Apache Hadoop bereits 2008 als Top Level Open Source Projekt der Apache Foundation eingestuft."[4]

1.1 Aufgabenstellung

Die Aufgabe, die hier bearbeitet wurde ist die Nummer 2.a aus dem 2. Teil der Übung. Sie bestannt darin 6 Texte verschiedener Sprachen nach dem jeweils längsten auftauchenden Wort zu durchsuchen. Anschließend galt es dieses Wort nach folgendem Schema in ein Dokument zu schreiben: "Sprache – Längstes Wort – Länge".

1.2 Struktur der Arbeit

Da wir beide zu beginn des Semesters noch nie mit Hadoop gearbeitet haben, investierten wir zunächst Zeit auf die Einrichtung von Hadoop auf unseren Computern. Da einer von uns nur auf Mac arbeiten konnte und der andere sowohl Mac als auch Windows nutzte entschieden wir uns die Einrichtung nur auf unseren Macs zu machen um unnötige Probleme durch den Systemunterschied zu vermeiden. Nachdem das Setup abgeschlossen war experimentierten wir mit dem durch Prof. Kovalenko bereitgestelltem Beispiel. Nach der anschließenden Implementierung nahmen wir messungen vor und werteten diese im letzten Schritt aus.

2 Setup

Da wir wie bereits erklärt auf MacOS arbeiten installierten wir Hadoop über homebrew:

```
1 Brew install hadoop
```

Nach der Installation galt es in den folgenden Files einstellungen vorzunehmen:[1]

```
hadoop-env.sh
core-site.xml
mapred-site.xml
hdfs-site.xml
```

2.1 hadoop-env.sh

In dieser Datei werden Umbegungsvariablen gesetzt und einstellung für die Hadoopkonfiguration vorgenommen. Es kann dafür verwendet werden das verhalten des Hadoop Deamons zu verändern. Beispielsweise kann das Logging verändert oder die Menge des verwendeten Speichers angepasst werden. Die Variable die zuerst gesetzt werden muss ist JAVA_HOME. Sie gibt an wo die zu verwendende Javaversion liegt. Da Hadoop maximal Java 8 unterstützt sollte man bei mehreren Javainstalationen auf die richtige Version achten.[2]

```
<!-- Put site-specific property overrides in this file. -->
<configuration>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/usr/local/Cellar/hadoop/hdfs/tmp</value>
    <description>A base for other temporary directories</
    description>
  </property>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://localhost:8020</value>
  </property>
</configuration>
```

2.2 mapred-site.xml

In mapred-site.xml wird verwendet um die Einstellungen von Map und Reduce zu manipulieren. Normalerweise ist dieses File leer. Alle änderungen hier werden in hadoop-default.xml übernommen.[3]

```
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>localhost:8021</value>
  </property>
</configuration>
```

2.3 hdfs-site.xml

Mithilfe von hdfs-site.xml können Einstellungen an den Deamons des Hadoop File Systems (hdfs) vorgenommen werden.[2]

```
<configuration>
  <property>
    <name>dfs.replication</name>
```

```
    <value>1</value>
  </property>
</configuration>
```

3 Implementierung

Für die Implementierung wurde eine lokale Installation von Hadoop gewählt. Der Source Code befindet sich hier: https://github.com/wolfelias/longest-word.

3.1 Map Funktion

Die Funktion Map wird mit folgender Signatur aufgerufen:

```
    protected void map(Object key, Text value, Context context)
```

Anhand von "Context context" wird der Pfad des zu analysierenden Dokumentes analysiert. Aus diesem Pfad wird die Information zur aktuellen Sprache geholt.

Das Text-Objekt value enthält den zu analysierenden Text Baustein. Dieser wird zunächst in einzelne Wörter unterteilt. Anschließend wird jedes Wort, konkateniert mit der Sprache, und der länge des Wortes in den HDFS Context geschrieben.

Der dem HDFS Context übergebene Tupel kann also folgendermaßen aussehen:

```
    context.write(new Text("English_Example", new IntWritable(7)))
```

3.2 Reduce Funktion

Das Ziel der Reduce Funktion ist das Befüllen folgender Hashmap:

```
        private final HashMap<String, HashMap<String, Integer>>
    allData = new HashMap<>();
```

In dieser Hashmap wird jeder Sprache eine Hashmap zugeordnet, die wiederum mit Wörtern und deren Länge befüllt ist. Ein Beispielhafter Eintrag könnte folgendermaßen aussehen:

```
<"English", <"Example",7>>
```

Dafür wird zunächst die Sprache aus dem ersten Element des Tupels geholt. Dafür wird der String bei dem Unterstrich in Sprache und Wort geteilt. Aus "English_Example" wird also ein Array der Form:

```
["English, Example"]
```

3.3 Cleanup Funktion

In der Cleanup Funktion wird durch die Sprachen der oben beschriebenen Hashmap iteriert. Für jede Sprache wir mithilfe eines Comparators das Längste Wort gefunden:

```
@Override
    protected void cleanup(Reducer<Text, IntWritable, Text,
    IntWritable>.Context context) throws IOException,
    InterruptedException {

    Comparator<Map.Entry<String, Integer>> comparator = (o1, o2) ->
     o2.getValue().compareTo(o1.getValue());


    for (Map.Entry<String, HashMap<String, Integer>> language :
    allData.entrySet()) {
        StringBuilder sb = new StringBuilder();

        List<Map.Entry<String, Integer>> values = new ArrayList<>(
    language.getValue().entrySet());
        values.sort(comparator);
        sb.append(language.getKey());
        sb.append(" - ");
        sb.append(values.get(0).getKey());
        context.write(new Text(sb.toString()), new IntWritable(
    values.get(0).getValue()));
}
```

Es wurde die Ausführungszeit gemessen, die Hadoop für die jeweiligen Sprachen benötigte. Bei der Messung wurde die Konfiguration in der Main Methode nicht berücksichtigt, sondern nur die Ausführung des Hadoop Jobs. Damit soll ein eindeutigeres Messergebnis erzeugt werden.

4 Auswertung

Die Auswertung der kann in Abbildung 1 betrachtet werden. In der Abbildung wird die Ausführungszeit in einer lokalen Entwicklungsumgebung auf einem Intel i7 Prozessor in Verhältnis zu der Größe der Dateien bzw. zu der Anzahl an Wörtern jeder Sprache gesetzt. Die Ergebnisse zeigen ein heterogenes Ergebnis. Es ist jedoch schwierig das Ergebnis zu interpretieren. Da in einer lokalen Entwicklungsumgebung viele Parameter eine Rolle spielen und Hadoop eigentlich auf verteilten Systemen ausgeführt wird, kann hier keine relevante Aussage über Verhältnis von Dateigröße und Ausführungszeit getroffen werden.

5 Fazit

Die Verwendung von Hadoop auf lokalen Entwicklungsumgebungen bzw. nicht-verteilten Systemen macht nur begrenzt Sinn. Die eigentlichen Stärken von Hadoop werden hierbei nicht ausgeschöpft. Ein Vergleich von Ausführungszeiten macht in einem Tier-1 System außerdem nur begrenzt Sinn. Lokale Ausführungszeiten

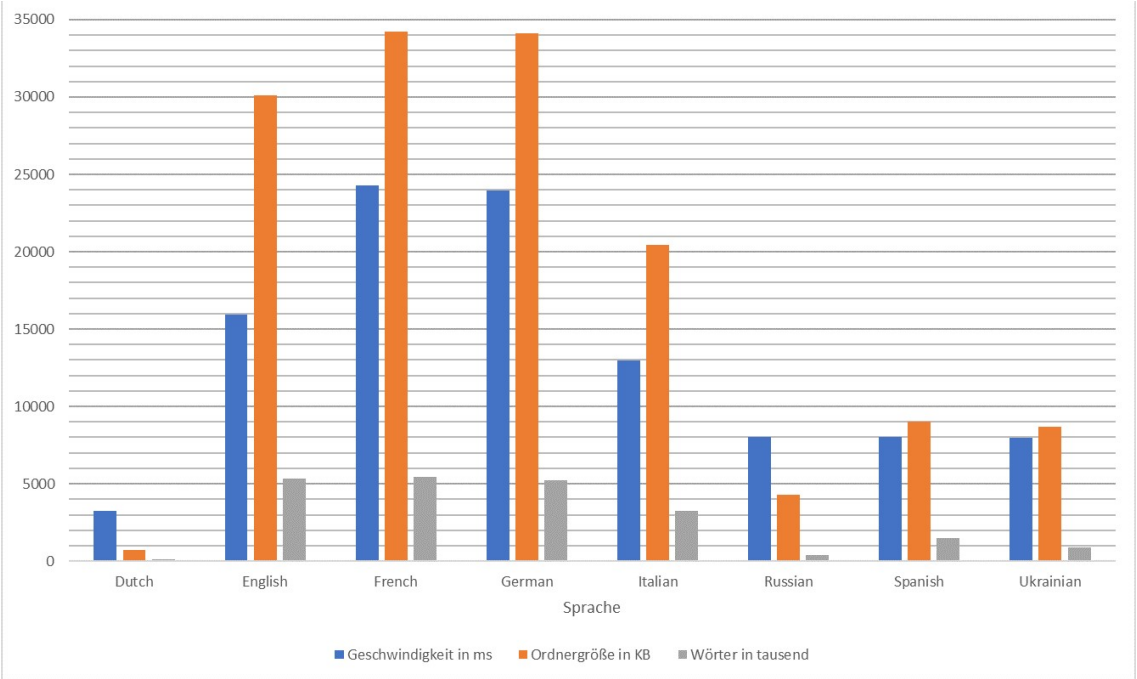

Figure 1: Verhältnis von Ausführungszeit, Dateiengröße und Anzahl an Wörtern der jeweiligen Sprache

variieren stark. Für eine Einordnung der Laufzeit von Hadoop empfehlen wir die Verwendung von Benchmarktests mit festgelegten Netzwerkbandbreiten.

Litteratur

[1] Diwakar. *Installing Hadoop on Mac*. Beer&Diapers.ai. Nov. 19, 2019. URL: `https://medium.com/beeranddiapers/installing-hadoop-on-mac-a9a3649dbc4d` (visited on 01/23/2023).

[2] *GettingStartedWithHadoop - HADOOP2 - Apache Software Foundation*. URL: `https://cwiki.apache.org/confluence/display/HADOOP2/GettingStartedWithHadoop` (visited on 02/06/2023).

[3] Kalyn Says. *Explaining Hadoop Configuration — Edureka.co*. Edureka. Section: Big Data. Sept. 26, 2014. URL: `https://www.edureka.co/blog/explaining-hadoop-configuration/` (visited on 02/06/2023).

[4] Laurenz Wuttke. *Hadoop einfach erklärt!* datasolut GmbH. Mar. 23, 2022. URL: `https://datasolut.com/apache-hadoop-einfuehrung/` (visited on 02/04/2023).