THM TECHNISCHE HOCHSCHULE MITTELHESSEN | CAMPUS GIESSEN | MNI Mathematik, Naturwissenschaften und Informatik

Apache Hadoop

Hauptseminar "Cloud-Plattformen und Big Data"
Dozent Steffen Rupp

von

René Gentzen

rene.gentzen@mni.thm.de

im WS22/23

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Hadoop Grundlagen

“Die Apache Hadoop Softwarebibliothek ist ein Framework, das die über Computercluster verteilte Verarbeitung großer Datensätze mit einfachen Programmiermodellen ermöglicht. Es ist so konzipiert, dass es von einzelnen Servern bis hin zu Tausenden von Rechnern skaliert werden kann, von denen jeder lokale Rechenleistung und Speicherplatz bietet. Anstatt sich auf Hardware zu verlassen, um eine hohe Verfügbarkeit zu gewährleisten, ist die Bibliothek selbst so konzipiert, dass sie Ausfälle auf der Anwendungsebene erkennt und bewältigt, so dass ein hochverfügbarer Dienst auf einem Cluster von Computern bereitgestellt wird, von denen jeder für sich für Ausfälle anfällig sein kann.”[1]
So beschreibt (übersetzt aus dem Englischen) die Apache Software Foundation ihr Top Level Projekt Apache™ Hadoop®. Diese Arbeit wird einen pragmatischen Überblick über Hadoop und die Komponenten im Hadoop Ecosystem geben. Dabei soll der Fokus nicht auf technischen Details, sondern auf dem praktischen Einsatz von Hadoop liegen. Es soll anhand von kleinen Fallstudien demonstriert werden, wie die einzelnen Hadoop Komponenten zur Abdeckung konkreter Anwendungsfälle auf- und eingesetzt werden.

1.1 Technischer und geschichtlicher Hintergrund

1.1.1 Anforderungen von Big Data

“Der Begriff „Big Data“ bezieht sich auf Datenbestände, die so groß, schnelllebig oder komplex sind, dass sie sich mit herkömmlichen Methoden nicht oder nur schwer verarbeiten lassen.”[2]
Schon Anfang der Neunziger war es nicht mehr praktikabel, Webseiten händisch, zum Beispiel in ”Web Directories”, zu katalogisieren. Man wollte Nutzern trotzdem die Möglichkeit geben, Informationen durch das Durchsuchen zentraler Anlaufstellen ausfindig zu machen. Automatisierte Tools, die sogenannten ”Web Crawler” wurden erfunden, um diese Arbeit zu übernehmen.[3]
Das Internet erlebte in den letzten Jahren des 20. Jahrhunderts ein explosionsartiges Wachstum an Nutzern und Webseiten, und damit auch an Informationen, die katalogisiert werden mussten.[4] Um eine immer größer werdende Menge an Informationen verarbeiten zu können, gibt es zwei Ansätze der Skalierung: Vertikale und horizontale Skalierung. Diese sollen in den folgenden Abschnitten erläutert werden, um die Designphilosophie hinter Hadoop zu verstehen.

1.1.2 Vertikale Skalierung

Bei der vertikalen Skalierung ("scaling up") werden *einem* System mehr Ressourcen wie zum Beispiel größerer Speicher, oder eine schnellere CPU hinzugefügt. Dadurch bekommt man einen Performance-Gewinn: Man kann mehr Daten speichern, oder Berechnungen werden schneller fertig gestellt. Ein großer Vorteil der vertikalen Skalierung ist, dass Anwendungsprogramme in der Regel nicht angepasst werden müssen, um vom diesem Performance-Wachstum zu profitieren. Wenn man eine 5TB große Festplatte gegen eine 10TB Festplatte austauscht, dann hat man den Speicherplatz eines Servers vertikal skaliert. Die darauf laufenden Programme müssen nicht angepasst werden, sondern man kann einfach doppelt so viele Daten speichern.[5]
Vertikale Skalierung hat drei große Nachteile: Erstens kann man nicht unbegrenzt vertikal skalieren. Ein Server kann physisch nur eine begrenzte Anzahl an Hardware aufnehmen. Zweitens wächst die Performance eines Systems bei vertikaler Skalierung höchstens linear[6], die Kosten allerdings nicht[7]. Heutzutage kann man gerade bei Cloud-Anbietern sehr leistungsfähige Systeme bei linearem Preisanstieg mieten.[8] Sucht man aber noch mehr Performance in *einem* System, dann steigen die Kosten exponentiell[9]. Drittens skalieren nicht alle Faktoren in einem System gleich gut vertikal. Die Speicherkapazität von SSDs ist zum Beispiel seit 1978 von 45MB auf 100TB gestiegen (Faktor $2222,22 \cdot 10^3$), während sich die Datenrate nur von 1.5MB/s auf 500/460MB/s (Sequential Read/Write) erhöht hat (Faktor $0,333 \cdot 10^3$).[9][10]

1.1.3 Horizontale Skalierung

Ein Cluster ist ein Verbund aus Computern (Nodes), die wie ein einziger, deutlich leistungsfähigerer Computer arbeiten.[11] Aufgaben und Daten werden in kleinere Teile zerlegt und auf alle Nodes im Cluster aufgeteilt, welche dann parallel Teilaufgaben lösen. Ergebnisse werden zusammengefügt und zurückgegeben. Anders als bei der vertikalen Skalierung kann man gerade in Zeiten des Cloud Computings praktisch unendlich horizontal skalieren. Allerdings muss man dafür kompliziertere Anwendungslogik verwenden, die mit der parallelen Ressourcenverteilung eines Clusters funktioniert. Bei der horizontalen Skalierung ("scaling out") werden einem Cluster zur Leistungssteigerung zusätzliche Nodes hinzugefügt. So kann ein Rechner zum Beispiel 500MB/s von seiner Festplatte lesen und verarbeiten, zehn Rechner lesen und verarbeiten in dieser Zeit allerdings 5000MB/s und können ihre Teilergebnisse anschließend zu einer Antwort zusammenfügen. Hierbei entsteht zwar zusätzlicher Netzwerk- und Verwaltungsaufwand ("Overhead"), aber die Leistungsfähigkeit des Clusters wächst mit jedem hinzugefügten Node. Ein horizontal skalierbares System ist mit höheren anfänglichen Kosten verbunden, kann dann aber bei linearem Kostenaufwand praktisch unendlich skaliert werden.[7]
Wie im eingänglichen Zitat erwähnt, setzt Hadoop auf eben dieses Prinzip der Skalierbarkeit, um "die über Computercluster verteilte Verarbeitung großer Datensätze mit einfachen Programmiermodellen"[1] als Dienst mit hoher Verfügbarkeit anzubieten. Die Technologien, die konkret dahinter stecken, werden im nächsten Abschnitt behandelt.

1.1.4 Historie

2002 begannen Doug Cutting und Mike Cafarella ihre Arbeiten an Apache Nutch[1], einer Open Source Web Search Engine als Teil des Apache Lucene Projekts[2]. Die beiden mussten Wege finden, um ihr Projekt auf die Milliarden Webseiten des Internets zu skalieren. 2003 veröffentlichte Google ein Whitepaper zur Architektur des Google File System (GFS), Googles eigenem verteilten Dateisystem.[3] Als Google 2004 dann ein weiteres Whitepaper zum MapReduce Programmiermodell veröffentlichte[4], sahen Cutting und Cafarella darin die Lösung für Nutch's Skalierungsproblem. Sie implementierten eigene Versionen von MapReduce als Processing Engine und des GFS zur Datenhaltung (NDFS, Nutch Distributed File System) als Basis für Nutch. Da diese beiden Komponenten mannigfaltige Anwendungsfälle außerhalb der Web-Suche bedienen konnten, wurden sie 2006 als eigenes Projekt Apache Lucene unterstellt und erhielten den Namen **Hadoop**. Ungefähr zur gleichen Zeit wurde Doug Cutting von Yahoo! rekrutiert, um Hadoop dort mit zusätzlichen Ressourcen weiterzuentwickeln. 2008 wurde Hadoop schließlich zu einem Top Level Projekt der Apache Software Foundation.[5] [14]

1.2 Hadoop Core

Der Kern von Hadoop (Hadoop Core) besteht seit Hadoop 2.x aus vier Modulen, welche im offiziellen Download zusammengefasst sind[1]:

- Hadoop Distributed File System (HDFS™): Hadoops verteiltes Dateisystem
- Hadoop MapReduce: Hadoops Parallel Processing Engine für große Datenmengen
- Hadoop YARN: Ein Framework für Job Scheduling und Ressourcenverwaltung im Cluster
- Hadoop Common: Unterstützende Programme für die anderen Hadoop-Module

Diese Komponenten bringen alles mit, was man zur verteilten Verarbeitung und Speicherung großer Datenmengen benötigt.

1.2.1 HDFS

Das HDFS ist ein Dateisystem, welches dem Anwender eine Abstraktionsschicht über verteilt gespeicherte Daten bietet. Das HDFS ist für den Betrieb auf sogenannter *"Commodity Hardware"* konzipiert. Commodity Hardware ist günstige, leicht zu ersetzende Hardware.

[1] https://nutch.apache.org/
[2] https://lucene.apache.org/
[3] 12, The Google File System.
[4] 13, MapReduce: Simplified Data Processing on Large Clusters.
[5] https://hadoop.apache.org/

Anders als spezialisierte Hardware, zum Beispiel Festplatten mit besonderer Fehlererkennung und -kompensation, wird bei Commodity Hardware nicht versucht, Ausfälle zu verhindern. Geht eine Komponente kaputt, was in einem Cluster von hunderten Maschinen kein Sonderfall ist, wird sie einfach durch eine neue ersetzt. Das bedeutet für die darauf arbeitenden Programme aber, dass diese softwareseitig dafür sorgen müssen, dass bei einem Ausfall der Betrieb des Clusters nicht beeinträchtigt wird.[15] Speichert man Dateien im HDFS, so werden diese in Blöcke aufgeteilt und auf Nodes im Hadoop Cluster verteilt. Praktisch heißt das, dass man eine 1TB große Datei im HDFS speichern kann, auch wenn man keine einzelne Festplatte mit 1TB Speicherkapazität besitzt. Der Anwender benutzt dabei das HDFS wie ein normales, nicht verteiltes Dateisystem. Die Verteilung passiert im Hintergrund.

NameNode DataNode Blöcke Replikation High Availability

Jeder einzelne Block wird dabei repliziert und auf unterschiedlichen Nodes gespeichert, um bei Ausfall eines Nodes keinen Datenverlust zu haben. Will man eine Datei aufrufen, fragt man bei einem zentralen Node (dem NameNode) die Datei an und dieser setzt die dazugehörigen Blöc

1.2.2 YARN

1.2.3 Setup

Single Node Setup

Defaulteinstellung des Hadoop Downloads Installation auf der einen beteiligten Maschine Start eines Single Node Clusters lokal Erste Übung zum Umgang mit dem HDFS

Fully-distributed Cluster

Installation von Hadoop auf allen beteiligten Maschinen Einrichtung von passwordless ssh auf allen Maschinen Evtl. Anpassung der /etc/hosts auf allen Maschinen Editieren der ganzen Konfigurationsdateien (XML) und kopieren der gleichen Dateien auf alle beteiligten Maschinen Editieren der Worker Datei auf dem NameNode NameNode (HDFS) formatieren Ausführen der Skripte auf dem NameNode

Hadoop in der Cloud

Google Dataproc, Azure HDInsights Oftmals eigenes Dateisystem, fully managed Grafische Oberfläche zum Submitten von Jobs, etc.

1.2.4 MapReduce

MapReduce Workflow: Erstellung eines MapReduce Jobs, Kopieren auf den Name Node und Ausführung

2 ETL mit Pig

Auch wenn es vermehrt von Spark verdrängt wird

2.1 Anwendungsfälle

Welche neuen Dinge ermöglicht dieses Tool Eine Zeile Pig Latin entspricht vielen Zeilen MapReduce

2.2 Architektur

2.3 Pig Latin

2.4 Praxis

2.4.1 Hinzufügen zum Cluster

2.4.2 Anwendung auf dem Cluster

3 Data Ingestion

3.1 Sqoop

3.2 Flume

4 Datawarehousing mit Hive

4.1 Anwendungsfälle

Welche neuen Dinge ermöglicht dieses Tool

4.1.1 Unterschiede zu Pig

4.2 Architektur

4.3 Interaktion

4.3.1 HiveQL

4.3.2 CLI

4.3.3 Java API

4.4 Praxis

4.4.1 Hinzufügen zum Cluster

4.4.2 Einrichtung einer Datenbank

4.4.3 Einlesen von Daten im CLI

4.4.4 Einlesen von Daten mit Sqoop

4.4.5 Absetzen einer Query

5 NoSQL mit HBase

5.1 Anwendungsfälle

Welche neuen Dinge ermöglicht dieses Tool

5.1.1 CAP-Theorem

5.1.2 ACID und BASE

5.2 Architektur

5.3 Interaktion

5.3.1 HBase Shell

5.3.2 Java API

5.4 Praxis

5.4.1 Hinzufügen zum Cluster

5.4.2 Einrichtung einer Datenbank

5.4.3 Einlesen von Daten

5.4.4 Datenmigration aus einem RDBMS

5.4.5 Absetzen einer Query

6 Streaming mit Kafka

6.1 Anwendungsfälle

Welche neuen Dinge ermöglicht dieses Tool Ersetzt durch Spark Streaming

6.2 Architektur

6.3 Interaktion

6.3.1 Who knows

6.4 Praxis

6.4.1 Hinzufügen zum Cluster

6.4.2 Maybe, vielleicht kann man ja was zeigen

7 Hadoop heute

7.1 Aktuelle Anwendungsbeispiele zu Hadoop

7.1.1 AirBnB

7.2 Apache Spark als Gold Standard

7.2.1 Kann eh alles besser

Literatur

1. *Apache Hadoop* [Apache Hadoop Main Page] [online]. [besucht am 2022-12-07]. Abger. unter: https://hadoop.apache.org/.
2. *Big Data: was es ist und was man darüber wissen sollte* [online]. [besucht am 2022-12-05]. Abger. unter: https://www.sas.com/de_de/insights/big-data/what-is-big-data.html.
3. GRIFFITHS, Richard T. *Search Engines* [History of the Internet] [online]. 2007-06-21. [besucht am 2022-12-05]. Abger. unter: https://web.archive.org/web/20070621143859/http://www.internethistory.leidenuniv.nl/index.php3?m=6&c=7#how.
4. ZAKON, Robert H'obbes'. *Hobbes' Internet Timeline - the definitive ARPAnet & Internet history* [Hobbes' Internet Timeline] [online]. 2018-01-01. [besucht am 2022-12-05]. Abger. unter: https://www.zakon.org/robert/internet/timeline/#Growth.
5. BEAUMONT, David. *How to explain vertical and horizontal scaling in the cloud* [Cloud computing news] [online]. 2014-04-09. [besucht am 2022-12-07]. Abger. unter: https://www.ibm.com/blogs/cloud-computing/2014/04/09/explain-vertical-horizontal-scaling-cloud/.
6. GUSTAFSON, John L. Amdahl's Law. In: PADUA, David (Hrsg.). *Encyclopedia of Parallel Computing*. Boston, MA: Springer US, 2011, S. 53–60. ISBN 978-0-387-09766-4. Abger. unter DOI: 10.1007/978-0-387-09766-4_77.
7. *Horizontal Vs. Vertical Scaling Comparison Guide* [MongoDB] [online]. [besucht am 2022-12-07]. Abger. unter: https://www.mongodb.com/basics/horizontal-vs-vertical-scaling.
8. *Pricing - Linux Virtual Machines Scale Sets | Microsoft Azure* [online]. [besucht am 2022-12-07]. Abger. unter: https://azure.microsoft.com/en-us/pricing/details/virtual-machine-scale-sets/linux/.
9. ATHOW, Desire. *At 100TB, the world's biggest SSD gets an (eye-watering) price tag* [TechRadar] [online]. 2020-07-07. [besucht am 2022-12-07]. Abger. unter: https://www.techradar.com/news/at-100tb-the-worlds-biggest-ssd-gets-an-eye-watering-price-tag.
10. *who was who in SSD? - StorageTek* [online]. [besucht am 2022-12-07]. Abger. unter: http://www.storagesearch.com/storagetek.html.
11. *What is a Computer Cluster? | Answer from SUSE Defines* [SUSE Defines] [online]. [besucht am 2022-12-07]. Abger. unter: https://www.suse.com/suse-defines/definition/computer-cluster/.

12. GHEMAWAT, Sanjay; GOBIOFF, Howard; LEUNG, Shun-Tak. The Google File System. In: *Proceedings of the 19th ACM Symposium on Operating Systems Principles*. Bolton Landing, NY, 2003, S. 20–43.

13. DEAN, Jeffrey; GHEMAWAT, Sanjay. MapReduce: Simplified Data Processing on Large Clusters. In: *OSDI'04: Sixth Symposium on Operating System Design and Implementation*. San Francisco, CA, 2004, S. 137–150.

14. CUTTING, Doug; CAFARELLA, Mike; LORICA, Ben. *The next 10 years of Apache Hadoop* [online]. 2016. [besucht am 2022-12-08]. Abger. unter: https://www.oreilly.com/content/the-next-10-years-of-apache-hadoop/.

15. WHITE, Tom. *Hadoop: the definitive guide*. Fourth edition. Beijing: O'Reilly, 2015. ISBN 978-1-4919-0163-2. OCLC: ocn904818464.