Data Mining in der Cloud

- von Jan-Christoph Meier
- Hamburg, 21.06.2012

Ablauf

- Einführung
- Verwandte Arbeiten
- Fazit / Ausblick
- Literatur

Ablauf

- Einführung 
- Verwandte Arbeiten
- Fazit / Ausblick
- Literatur

Was ist Data Mining?

- Extrahieren von Wissen aus **großen Mengen** von Daten
- Alternativ auch: “**Knowledge Discovery from Data**”
- Siehe AW1 - “Datenanalyse mit Data Mining”

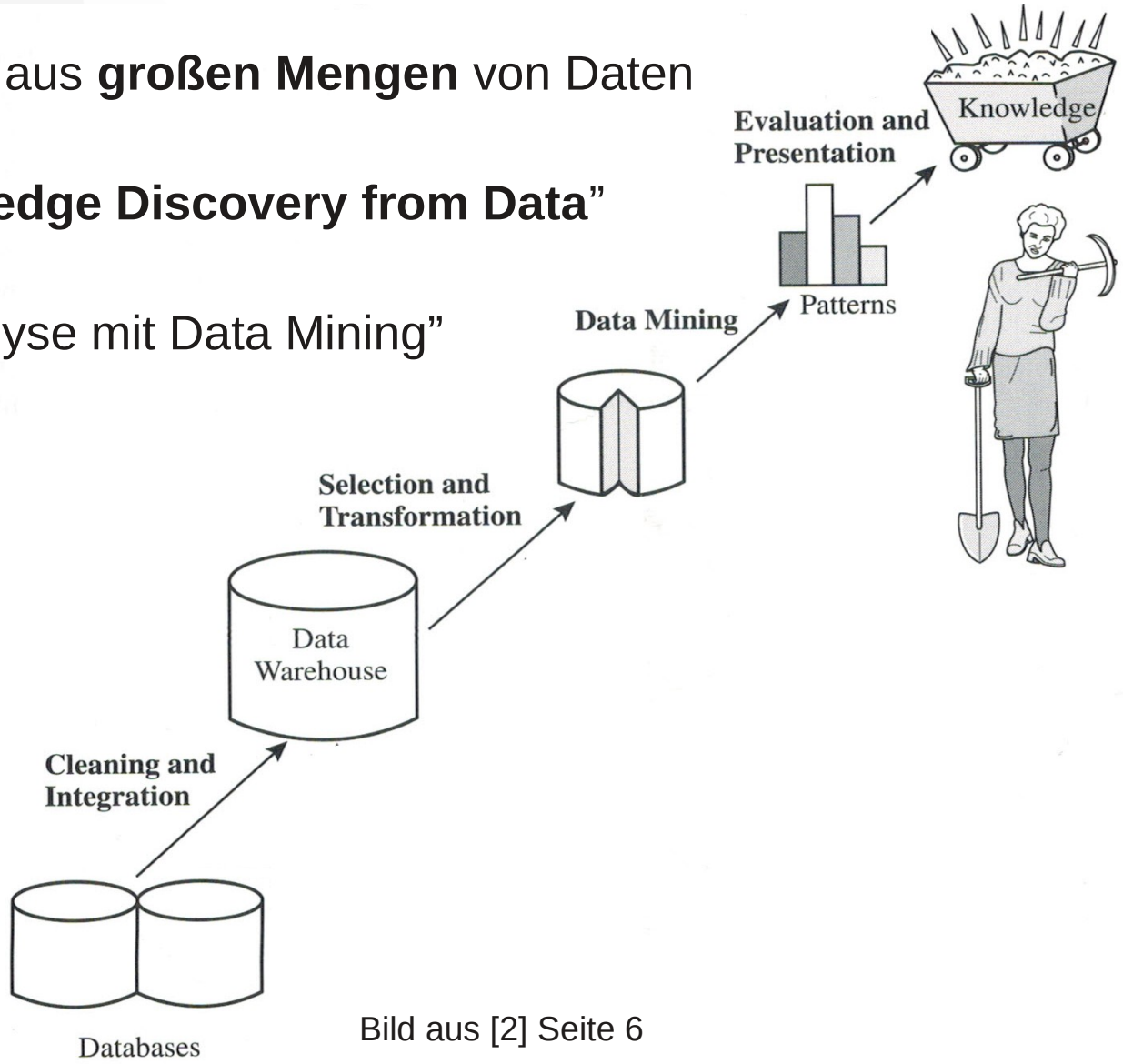


Bild aus [2] Seite 6

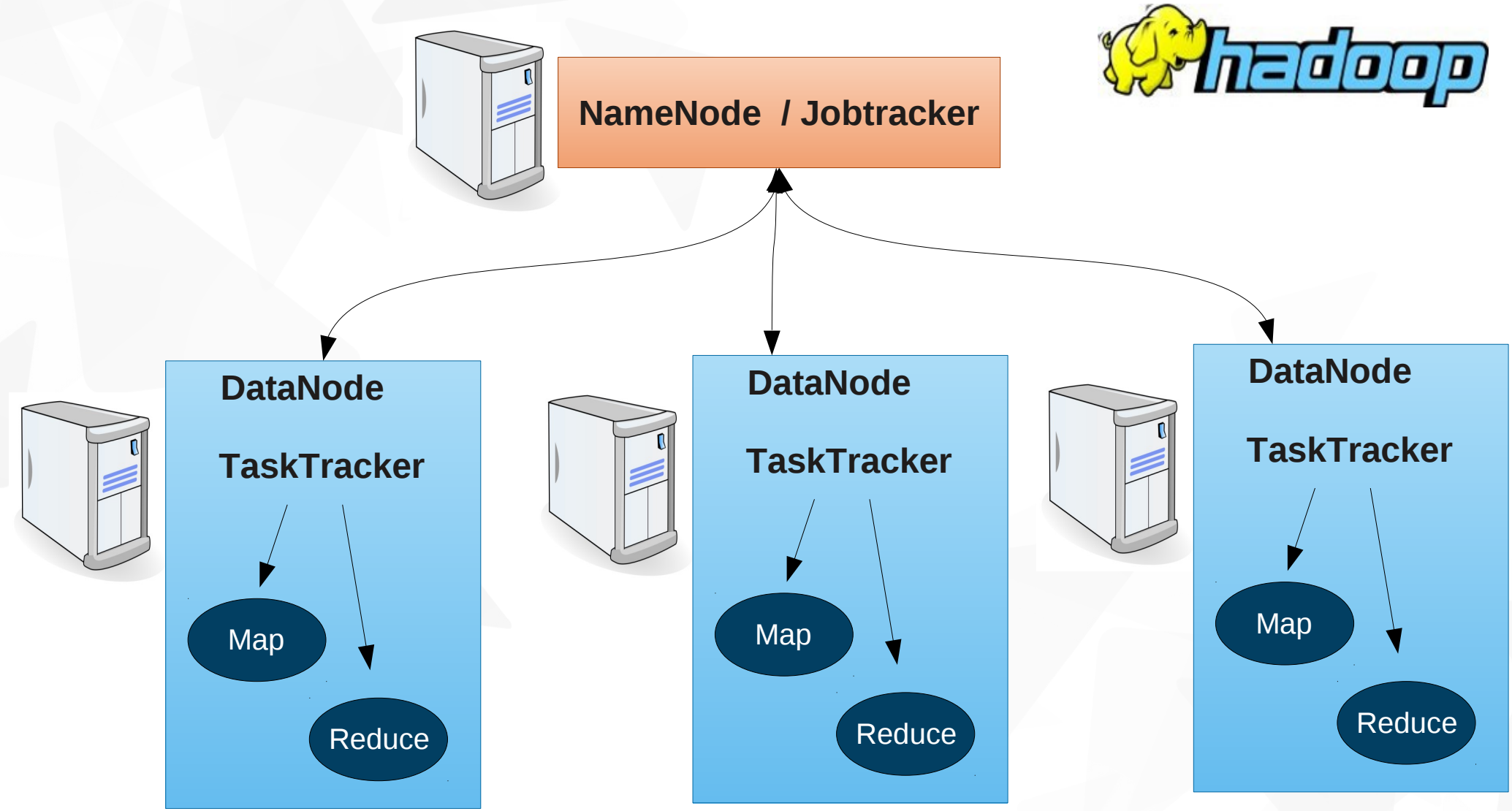
Warum Data Mining in der Cloud?

- Data Mining Algorithmen arbeiten auf großen Datenmengen, für deren Analyse viel Rechenleistung benötigt wird.
- Rechenleistung kann kostengünstig angemietet und je nach Bedarf skaliert werden.
- Optimal wäre ein System, das auf beliebig großen Datenmengen operieren und auf beliebig viele Rechner skaliert werden kann.

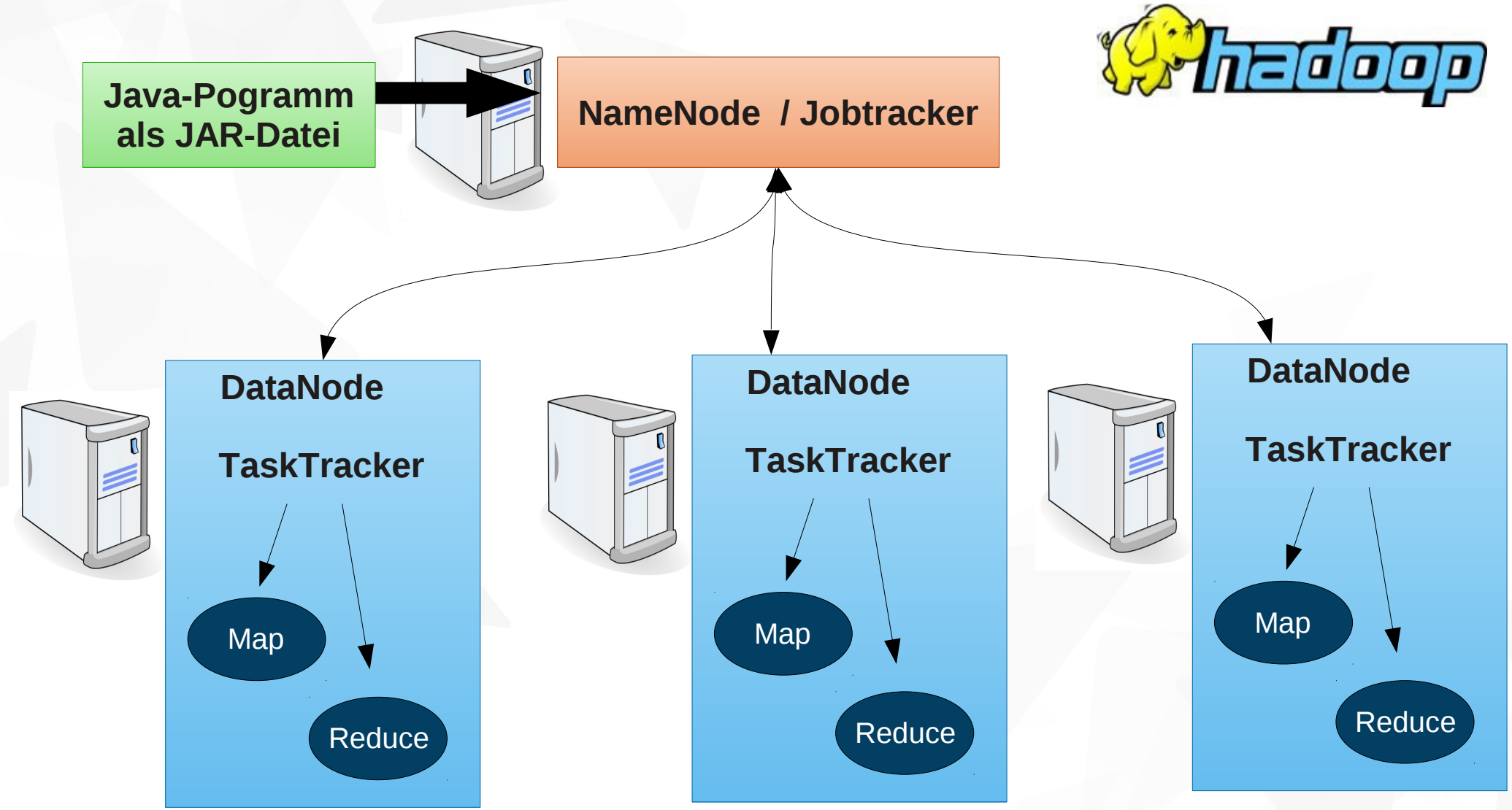
Hadoop

- Java-Framework, das es ermöglicht, Algorithmen parallel auf beliebig vielen Rechnern auszuführen.
- Die Algorithmen müssen nach dem MapReduce-Paradigma implementiert sein.
- Hadoop implementiert ein verteiltes Dateisystem (Hadoop distributed filesystem – HDFS).
- Koordiniert die Ausführung der Algorithmen und die Verfügbarkeit der Daten.
- Ermöglicht die Verarbeitung großer Datenmengen (Bigdata).
- Kann in der Cloud betrieben werden, z.B. Amazon AWS

Hadoop - Architektur



Hadoop - Architektur



Hadoop - Bewertung

Vorteile

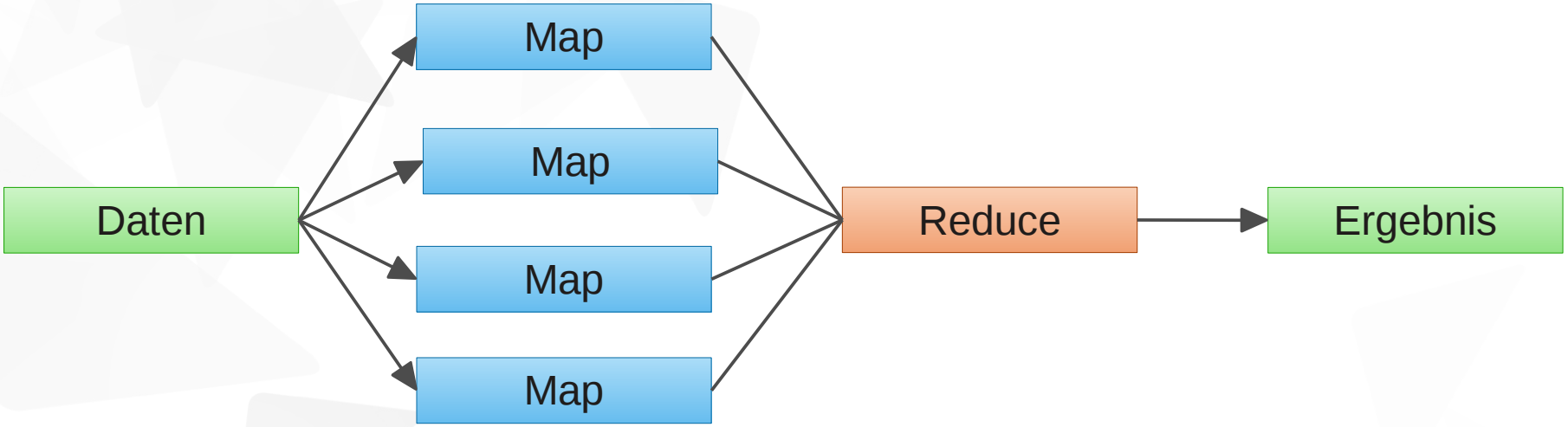
- Algorithmen können einfach parallelisiert und auf verschiedenen Rechnern ausgeführt werden.
- Es muss sich nicht um die Partitionierung der Daten und die Parallelisierung der Algorithmen gekümmert werden.

Nachteile

- Algorithmen müssen nach dem MapReduce-Paradigma implementiert sein.
- Verteilung kann Overhead bedeuten, falls nur auf kleinen Daten operiert werden soll.

MapReduce

Schematik des Algorithmus



Datenstrukturen

	Input	Output
Map	<k1, v1>	list(<k2, v2>)
Reduce	<k2, list(v2)>	list(<k3, v3>)

Aus: Hadoop in Action [4] – Seite 12

MapReduce - Beispiel

Zählen der Häufigkeiten von Elementen in Transaktionen

T1: a1, a2, a3
T2: a1, a3

Aufruf des Mappers für jede Transaktion:

map(“T1”, “a1, a2, a3”) -> [(a1: 1), (a2: 1), (a3: 1)]
map(“T2”, “a1, a3”) -> [(a1: 1), (a3: 1)]

Aufruf des Reducers:

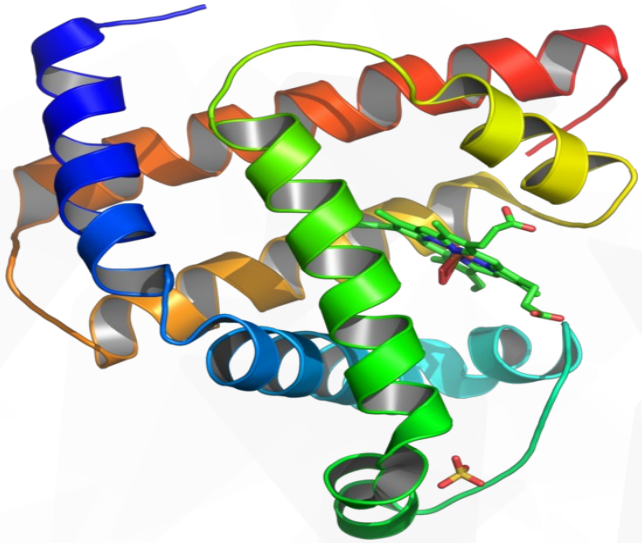
reduce(a1, [1, 1]) -> [(a1, 2)]
reduce(a2, [1]) -> [(a2, 1)]
reduce(a3, [1, 1]) -> [(a3, 2)]

	Input	Output
Map	<k1, v1>	list(<k2, v2>)
Reduce	<k2, list(v2)>	list(<k3, v3>)

Aus: Hadoop in Action [4] – Seite 12

Motivation - Vereinfacht

- Finden von Eigenschaften / Strukturen in Proteinsequenzen
- Proteinsequenz: Kette von Aminosäuren
- Aminosäuren haben verschiedene Eigenschaften



Motivation - Beispiel

- Proteinsequenz:

  MVDEQVAVEHGTVSHTISREEDGVVHERVLASGERVEVFYKAPAPRPREGRA

- Chemische Eigenschaften von Aminosäuren:

	hydrophob / -phil	volume	charge
Alanin Ala A	phob	small	n
Arginin Arg R	*phil*	large	+
Asparagin Asn N	phil	small	n
Asparaginsäure Asp D	phil	small	-
Cystein Cys C	phob	small	n
Glutamin Gln Q	phil	medium	n
...	....	....	....

Ziele

- Finden einer Korrelation von Struktur und Eigenschaften in einer Vielzahl von Proteinsequenzen mithilfe von Data Mining Algorithmen.
- Analyse unter Verwendung eines Hadoop-Clusters.

Ablauf

- Einführung
- Verwandte Arbeiten 
- Fazit / Ausblick
- Literatur

Apache Mahout

- Data Mining Framework

Apache Mahout

- Implementiert verschiedene Data Mining Algorithmen
  - Clustering
  - Pattern Mining
  - Machine Learning
  - uvm.

http://mahout.apache.org

Apache Mahout (2)

- Die Algorithmen in Mahout...
  - sind nach dem MapReduce-Paradigma implementiert
  - können in einem Hadoop-Cluster ausgeführt werden
  - operieren auf beliebig großen Datenmengen - Gigabyte, Terabyte, Petabyte
- **Umdenken**: Methoden haben keine Parameter oder Rückgabewerte, es wird auf serialisierten Daten im Dateisystem operiert.
- Die Algorithmen sind so generisch implementiert, dass sie mit wenig Aufwand auf beliebige Problemstellungen angewandt werden können

Apache Mahout - Bewertung

Vorteile

- Bietet ein breites Spektrum an Data Mining Algorithmen.
- Algorithmen können auf einem Hadoop-Cluster ausgeführt werden.
- Beliebig große Eingabedaten sind möglich.

Nachteile

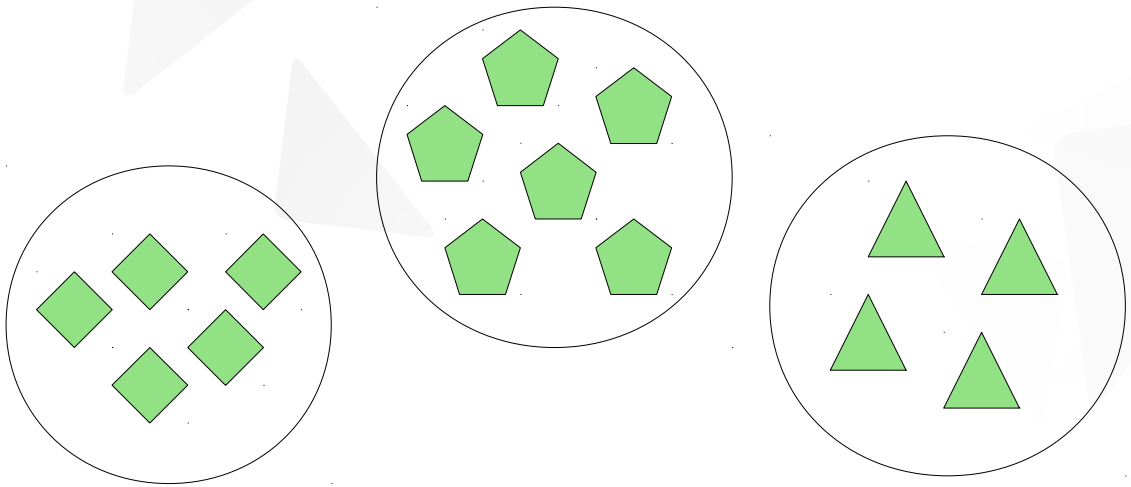
- Die Eingabedaten müssen in ein kompatibles Datenformat konvertiert werden.

Parallel K-Means Clustering Based on MapReduce

- Weizhong Zhao, Huifang Ma, Qing He
- Lecture Notes in Computer Science, 2009
  Volume 5931/2009, 674-679

K-Means Algorithmus

- Clustering Algorithmus
- Partitioniert eine Menge von Objekten in verschiedene Teilmengen.
- Die Objekte innerhalb der Teilmengen sind ähnlich.
- Die Teilmengen unterscheiden sich.



K-Means Algorithmus - Ablauf

- Eingabe Parameter:

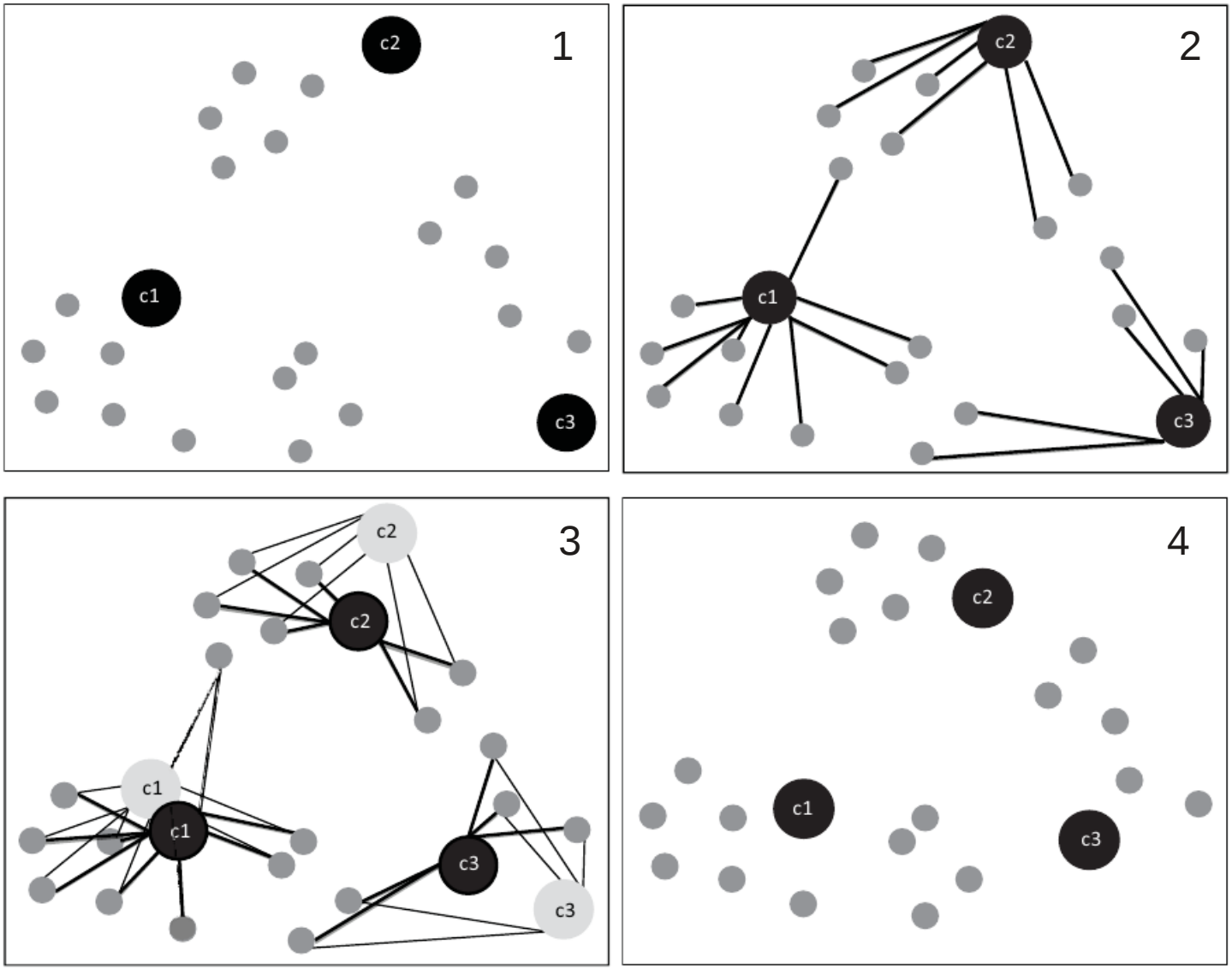
  D – Dataset, das n Objekte enthält

  k – Anzahl der Cluster

- Ablauf

  1. Es werden k Objekte aus D zufällig gewählt, diese stellen die Cluster-Mittelpunkte dar.
  2. **wiederhole:**
  3. Weise jedem Objekt einen der Cluster-Mittelpunkte zu, der dem Objekt am ähnlichsten ist. Neue Cluster werden gebildet.
  4. Erzeuge für jedes Cluster einen neuen Mittelpunkt, der durch Bildung des Mittelwertes berechnet wird.
  5. **bis sich die Cluster-Mittelpunkte nicht mehr ändern**

K-Means Algorithmus - Beispiel



Quelle: Mahout in Action [3] – Seite 147

K-Means auf MapReduce

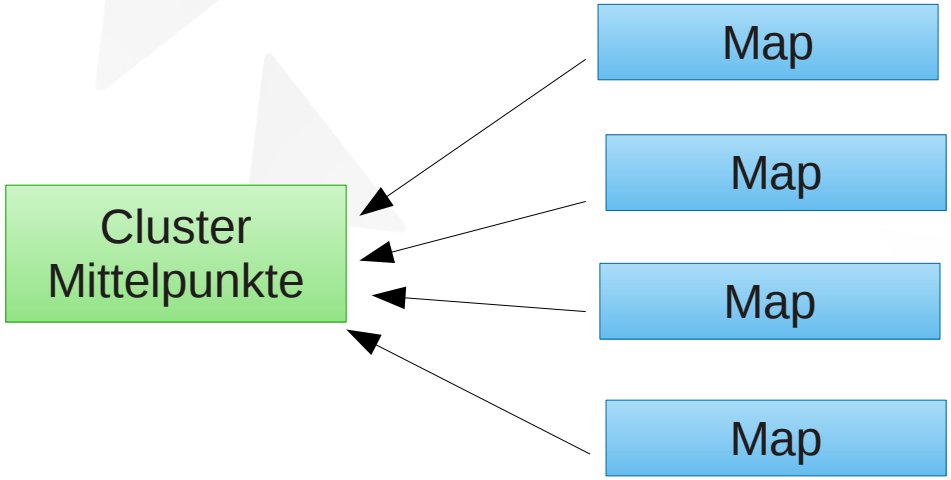
- Aufwändigste Operation: In jeder Iteration muss der Abstand zwischen den *n* Objekten und *k* Cluster-Mittelpunkten berechnet werden.
- Der Abstand muss in jeder Iteration *n* * *k* häufig berechnet werden.
- **Ansatz:**
  - Mapper-Funktion berechnet den Abstand zu den Cluster-Mittelpunkten.
  - Reducer-Funktion bestimmt neue Cluster-Mittelpunkte
- Die einzelnen Mapper und Reducer können parallel ausgeführt werden.

K-Means auf MapReduce – Im Detail

- Cluster-Mittelpunkte werden “global” gespeichert.

Map-Funktion

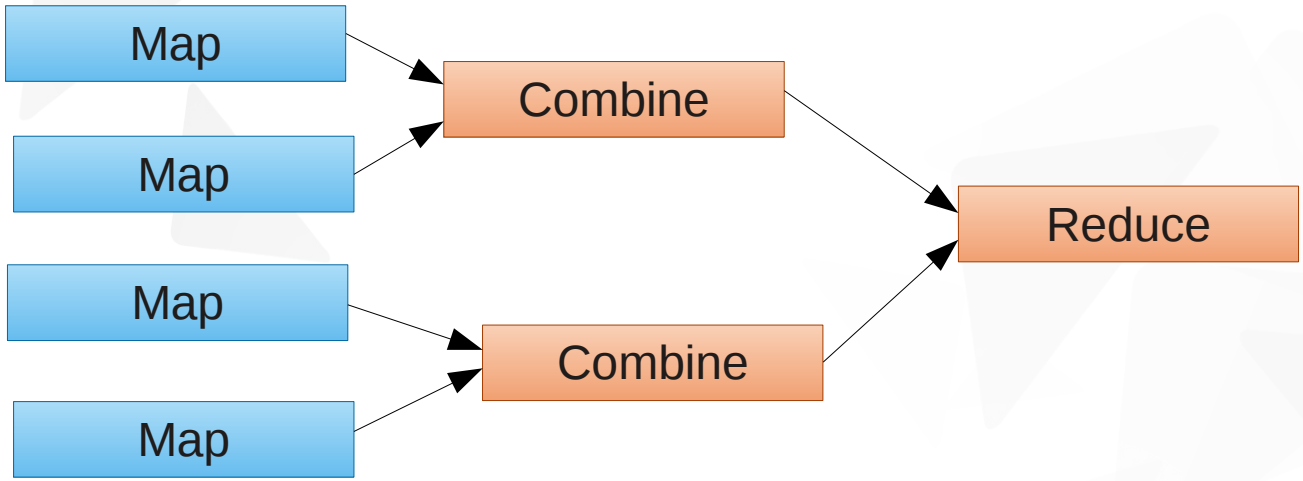
- Wird für jedes Objekt aufgerufen und berechnet den Abstand zu allen Cluster-Mittelpunkten.
- Der Cluster-Mittelpunkt, der dem Objekt am nächsten ist, wird zurückgegeben.



K-Means auf MapReduce – Im Detail

Combine-Funktion

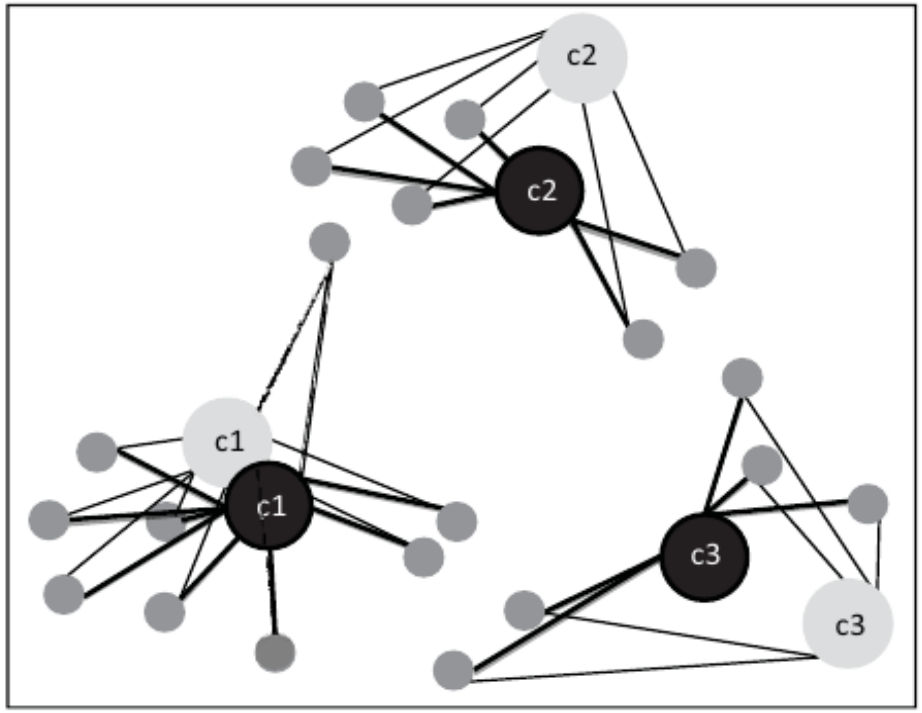
- Führt eine Vorberechnung nach mehreren Aufrufen der Map-Funktion durch.
- Zählt die Anzahl der Objekte in einem Cluster.
- Summiert die Objekte auf, bereitet die Mittelwertsberechung für die Reduce-Funktion vor.



K-Means auf MapReduce – Im Detail

Reduce-Funktion

- Berechnet den Mittelwert für die einzelnen Cluster und bestimmt somit die neuen Mittelpunkte



Quelle: Mahout in Action [3] – Seite 147

Parallel K-Means Clustering Based on MapReduce - Bewertung

Vorteile

- Einfach zu implementierender Clustering-Algorithmus.

Nachteile

- Es muss die Anzahl „k“ der Cluster als Parameter mitgegeben werden.
- Für die Eingabedaten muss eine optimale Funktion zur Berechnung der Ähnlichkeiten (Abstand zum Cluster-Mittelpunkt) gefunden werden.

Pfp: parallel fp-growth for query recommendation

- Haoyuan Li, Yi Wang, Dong Zhang, Ming Zhang, and Edward Y. Chang
- Proceedings of the 2008 ACM conference on Recommender systems, 2008

Frequent-Pattern-Tree Algorithmus

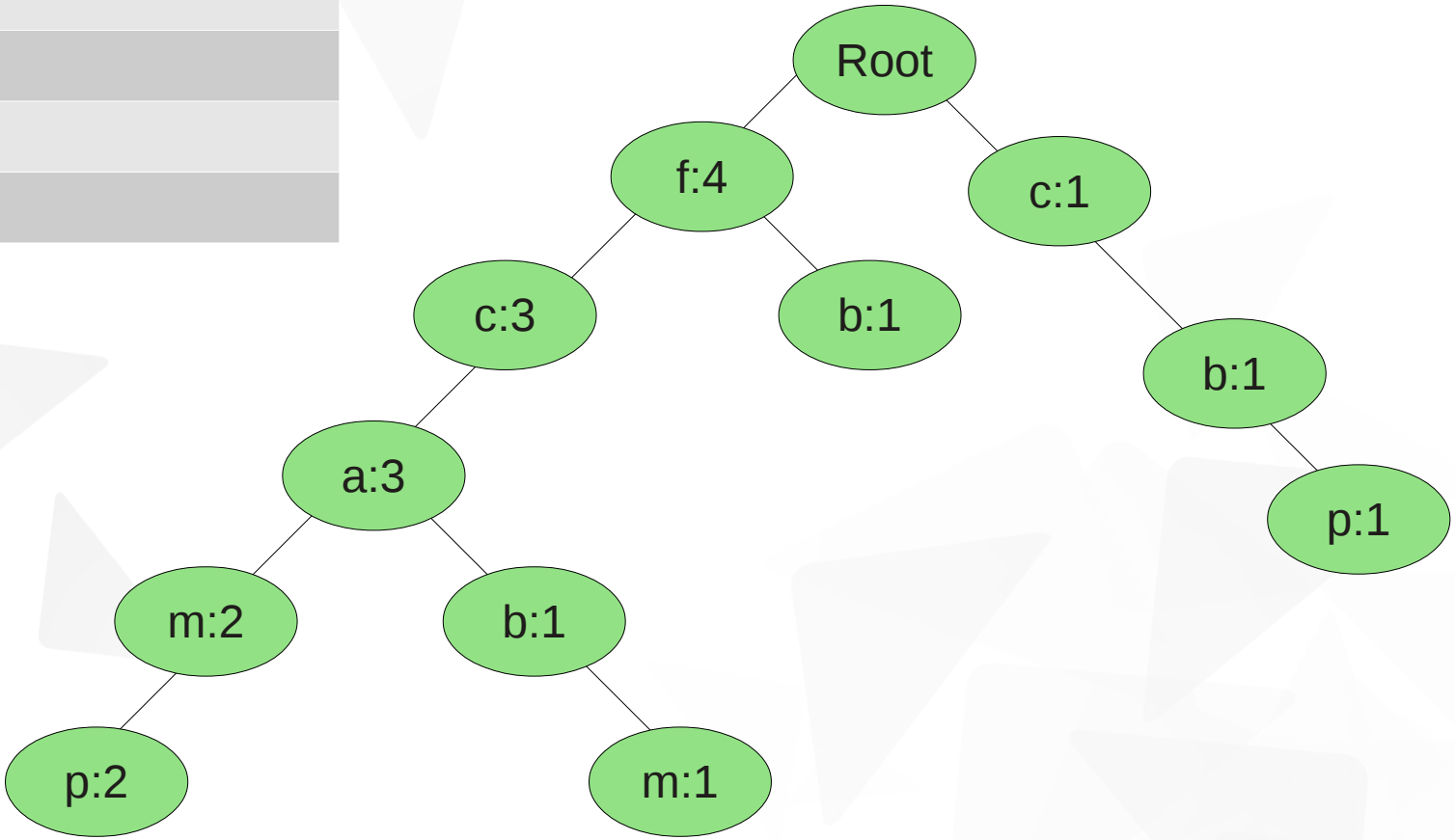
- Findet in beliebig vielen Transaktionen Elemente, die häufig gemeinsam auftreten.

Ablauf in 3 Schritten:

- Bestimmen der Häufigkeiten (Support) der Elemente in den Transaktionen.
- Erzeugen eines Frequent-Pattern-Trees.
- Analyse des FP-Trees: Bestimmen der Elemente, die häufig gemeinsam auftreten.

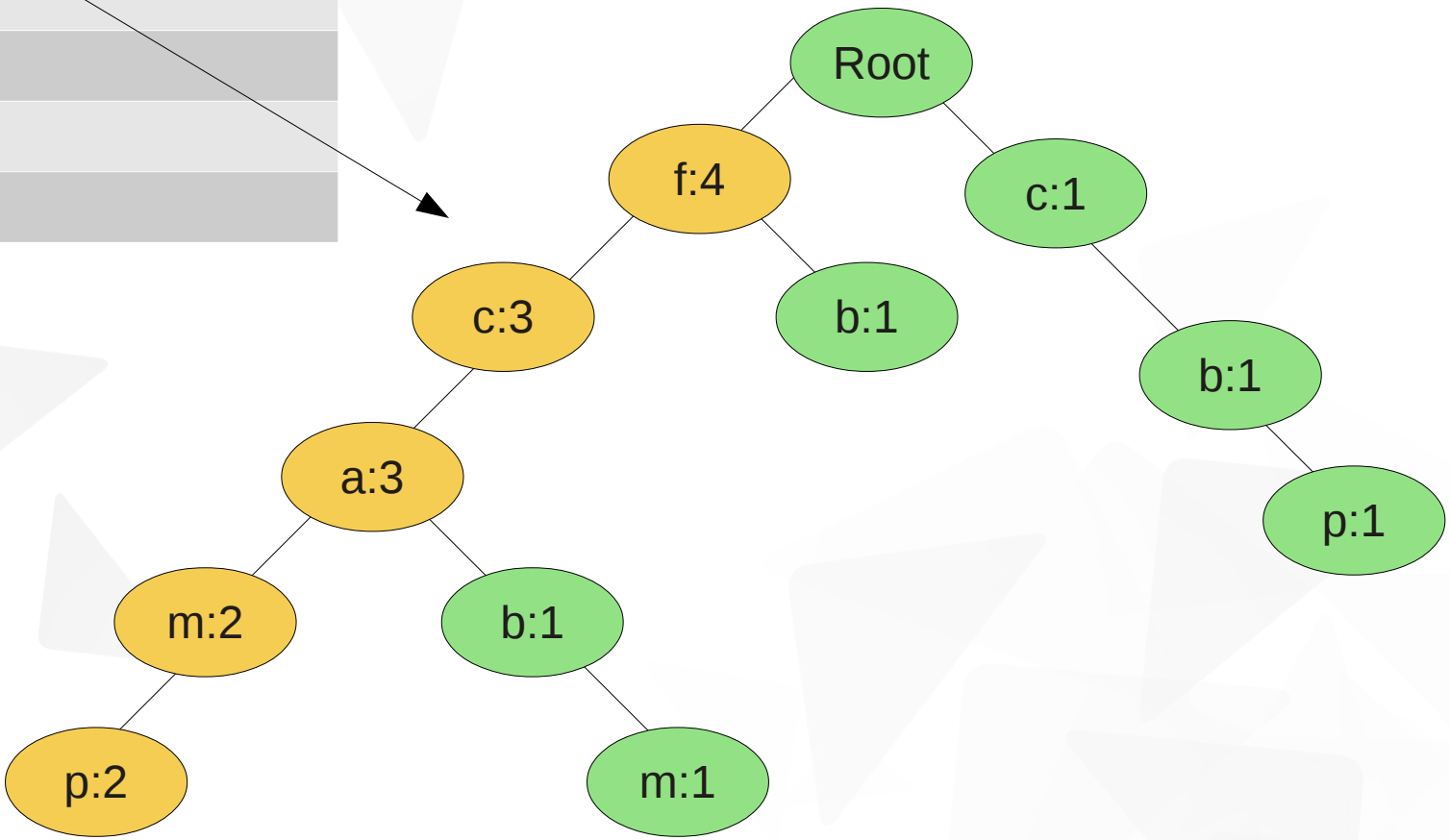
FP-Tree Algorithmus (2)

TID	Elemente mit Support >= 3
1	f, c, a, m, p
2	f, c, a, b, m
3	f, b
4	c, b, p
5	f, c, a, m, p



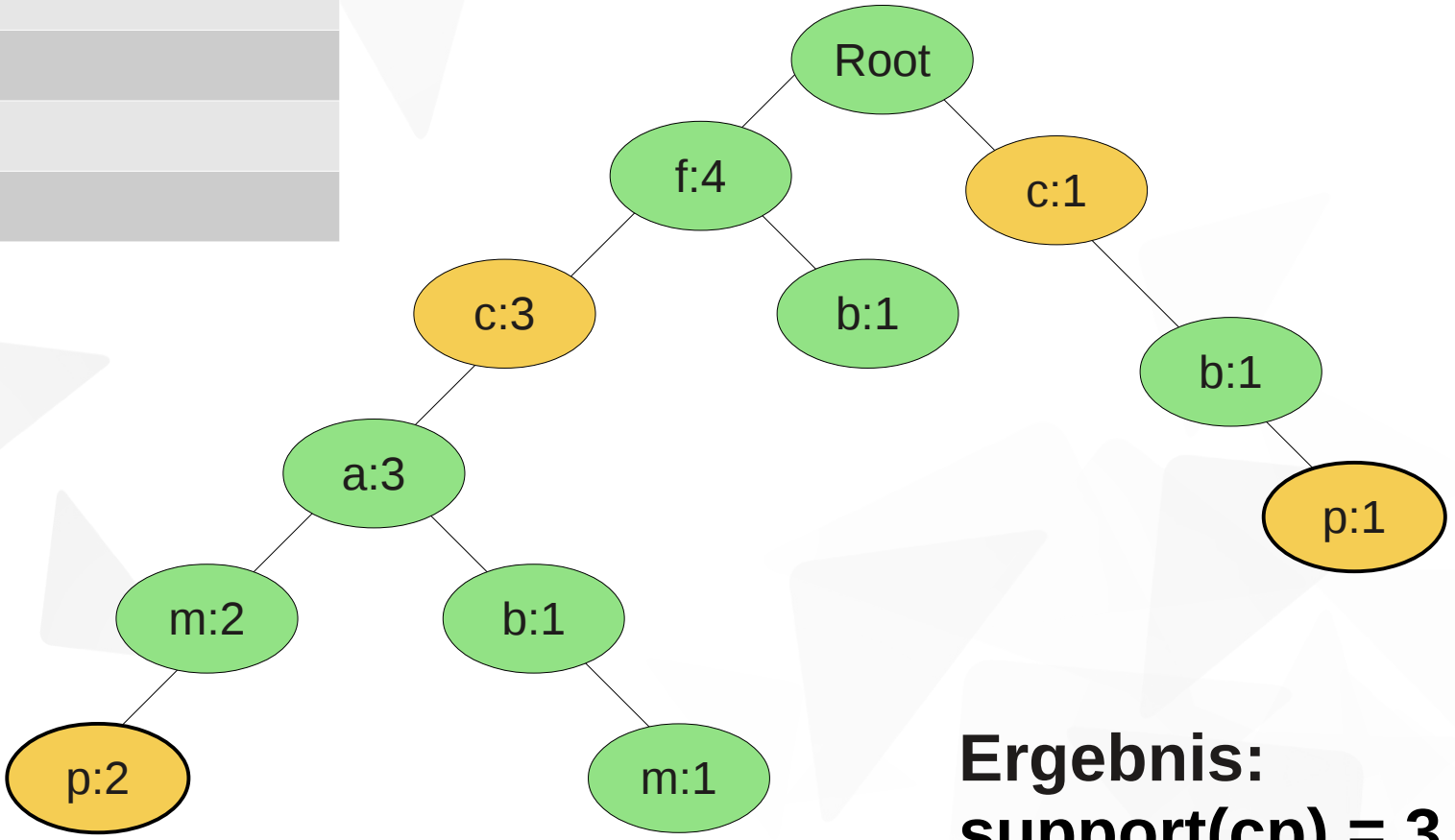
FP-Tree Algorithmus (2)

TID	Elemente mit Support >= 3
1	f, c, a, m, p
2	f, c, a, b, m
3	f, b
4	c, b, p
5	f, c, a, m, p



FP-Tree Algorithmus (2)

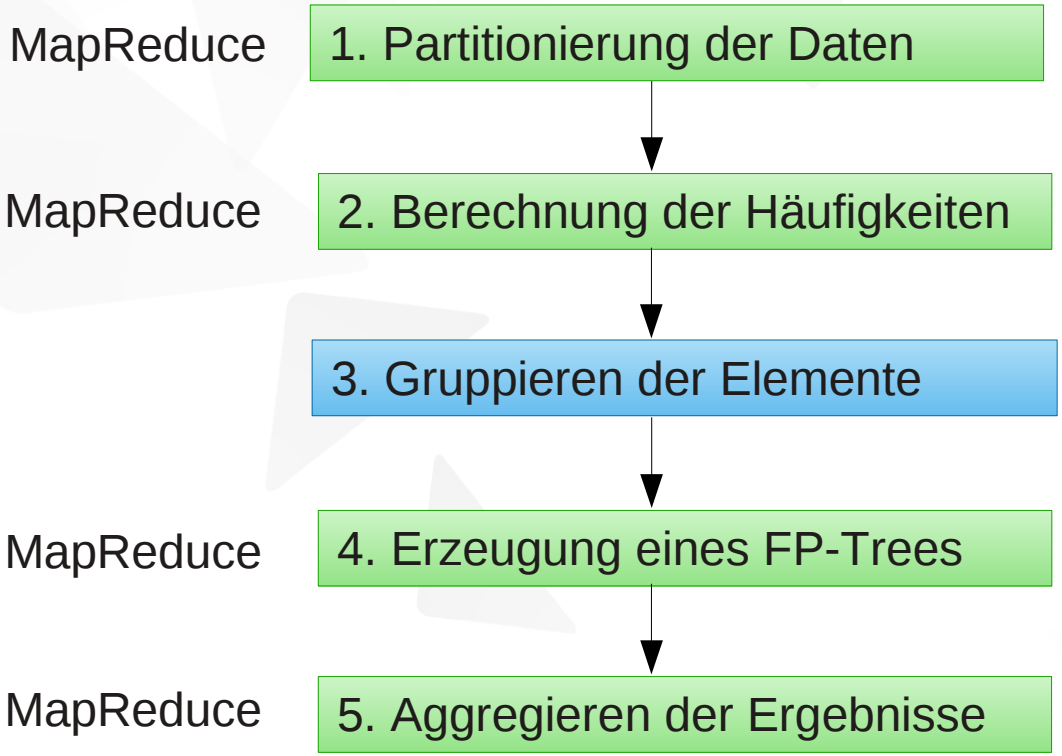
TID	Elemente mit Support >= 3
1	f, c, a, m, p
2	f, c, a, b, m
3	f, b
4	c, b, p
5	f, c, a, m, p



Ergebnis:
support(cp) = 3

Parallel FP-Growth auf MapReduce

Arbeitet in **5 Schritten**, die teilweise als MapReduce-Jobs ausgeführt werden:



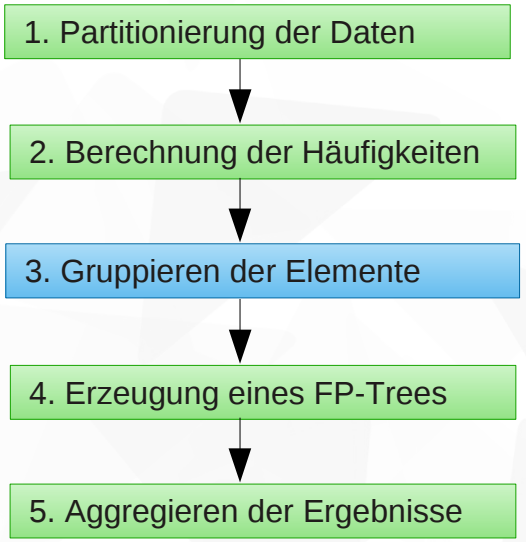
Parallel FP-Growth auf MapReduce – Im Detail

Schritt 1: **Partitionierung der Daten**

Transaktionen in der Datenbank werden auf *n* Rechner verteilt.

Schritt 2: **Häufigkeiten der Elemente berechnen (Support)**

Es werden mit einem MapReduce-Job die Support-Werte für alle Elemente der Transaktionen berechnet. Das Ergebnis wird in der „Frequency-Liste (F-List)“ gespeichert.



Parallel FP-Growth auf MapReduce – Im Detail

Schritt 3: Gruppieren der Elemente

Die Liste mit den Häufigkeiten der Elemente wird in Gruppen aufgeteilt. Jede Gruppe erhält eine eindeutige Gruppen-ID. Die Gruppen dienen dazu, die Transaktionen eindeutig zu partitionieren.

Schritt 4: **FP-Tree erzeugen**

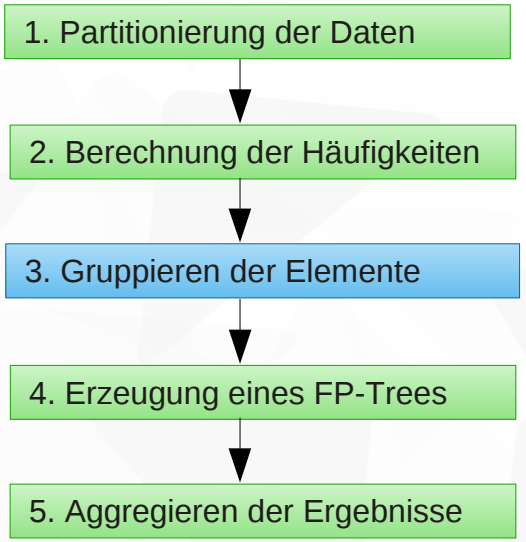
Map: Jedem Mapper wird ein Teil der Transaktionen übergeben. Der Mapper ordnet die Transaktionen den Gruppen von Schritt 3 zu. Gibt Gruppe und eine Liste mit Transaktionen zurück.

Reduce: Erhält eine Gruppe und mehrere der Gruppe zugeordnete Transaktionen, erzeugt einen FP-Tree und berechnet ein lokales Ergebnis für die Confidence.

Parallel FP-Growth auf MapReduce – Im Detail

Schritt 5: Aggregieren der Ergebnisse

Die im vorherigen Schritt gewonnenen Werte für die häufig gemeinsam auftretenden Elemente werden aggregiert und zu einem Gesamtergebnis zusammengefasst.



Pfp: parallel fp-growth for query recommendation - Bewertung

Vorteile

- Algorithmus benötigt nur wenige Parameter.
- Liefert eine andere Sichtweise auf die Daten als der Clustering-Algorithmus.

Nachteile

- Elemente werden ausschließlich anhand der Häufigkeit selektiert, weitere Eigenschaften werden nicht betrachtet.

Ablauf

- Einführung
- Verwandte Arbeiten
- Fazit / Ausblick 
- Literatur

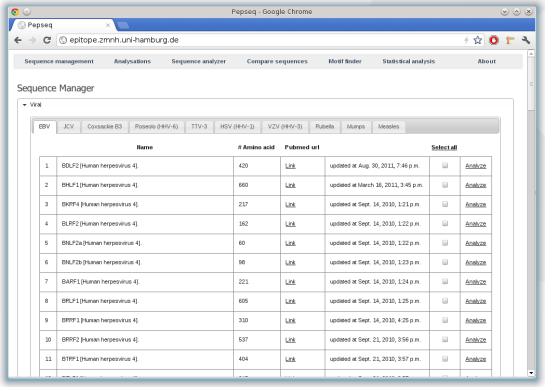
Fazit

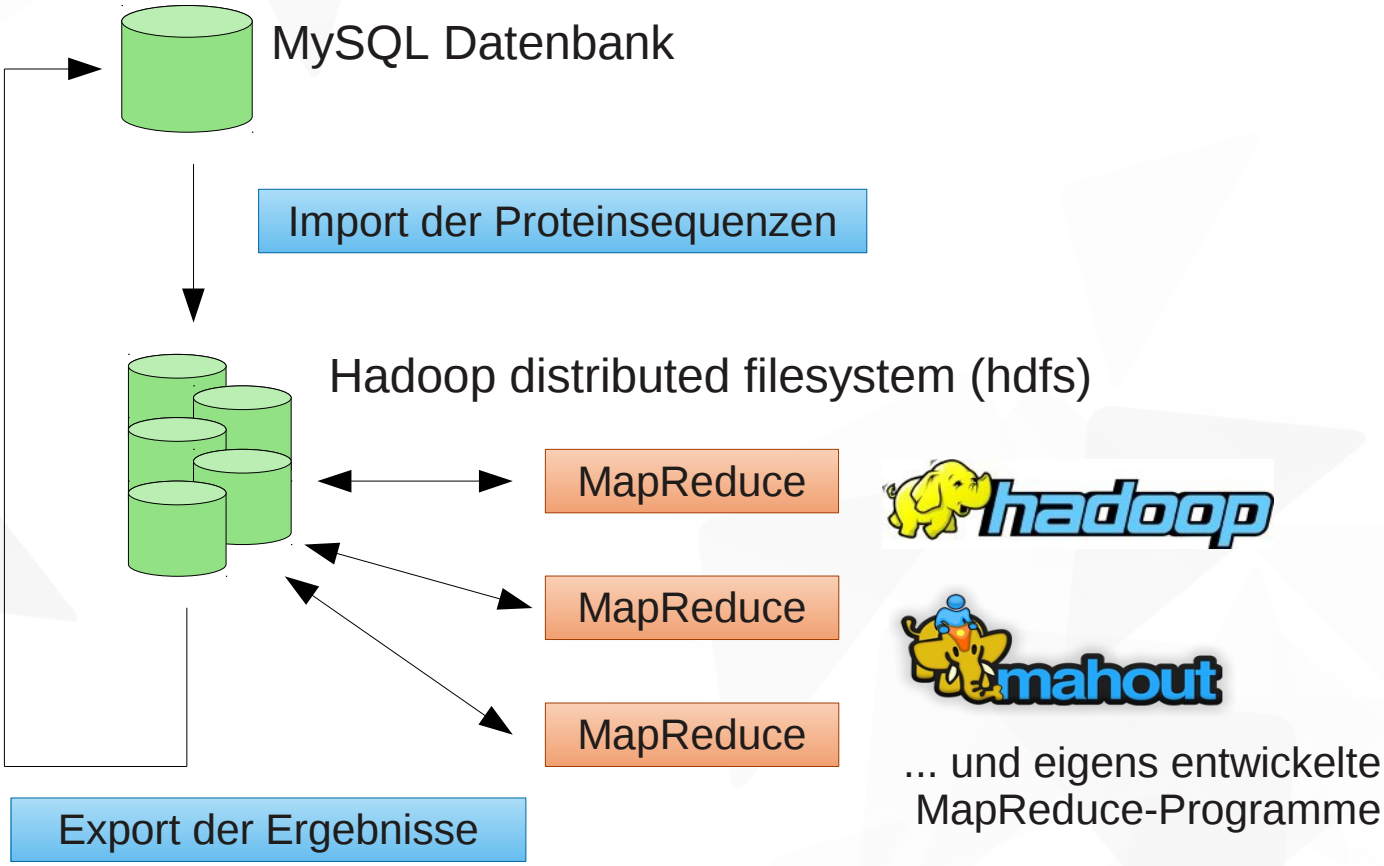
- Durch MapReduce können die Algorithmen auf beliebig viele CPU-Cores skaliert werden und auf großen Datenmengen operieren.
- Mahout bietet ein breites Spektrum an Algorithmen, die eingesetzt werden können, um effizient Data Mining durchzuführen.
- Die durch die Algorithmen gelieferten Ergebnisse müssen veranschaulicht werden, damit sie interpretiert werden können.
- **Mögliche Probleme:**
  - Verifikation der Ergebnisse schwierig.
  - An welcher Stelle ist die Analyse abgeschlossen?

Ausblick – Projekt 1 und Projekt 2

Eine Infrastruktur wird aufgebaut, um verschiedene Data Mining Algorithmen auf Proteinsequenzen anzuwenden:

Steuerung und Visualisierung über ein Webinterface





Ablauf

- Einführung
- Verwandte Arbeiten
- Fazit
- Literatur 

Literatur

[1] **Parallel K-Means Clustering Based on MapReduce**
Weizhong Zhao, Huifang Ma, Qing He
Lecture Notes in Computer Science, 2009 - Volume 5931/2009, Seiten 674-679

[2] **Data Mining, Concepts and Techniques**
Jiawei Han, Micheline Kamber, Jian Pei
Morgan Kaufmann 2011

[3] **Mahout in Action**
Sean Owen, Robin Anil, Ted Dunning, Ellen Friedman
Manning 2011

[4] **Hadoop in Action**
Chuck Lam
Manning 2010

[5] **MapReduce: Simplified Data Processing on Large Clusters**
Jeffrey Dean, Sanjay Ghemawat
OSDI'04, Sixth Symposium on Operating System Design and Implementation, Dezember 2004

Literatur (2)

[6] **Pfp: parallel fp-growth for query recommendation**
Haoyuan Li, Yi Wang, Dong Zhang, Ming Zhang, and Edward Y. Chang
Proceedings of the 2008 ACM conference on Recommender systems, 2008, Seiten 107-114

Ende

Vielen Dank für die Aufmerksamkeit!

Fragen?