Karlsruher Institut für Technologie

MapReduce

Johann Volz

IPD Snelting, Lehrstuhl Programmierparadigmen

Steps → Jobs → Tasks → Task Attempts

Refresh List

Task	Type	Job	State	Start Time	Elapsed Time	Actions
r_000004	reduce	201005131719_0001	RUNNING	2010-05-13 19:20 GMT+	0 hours 12 minutes	View Attempts
r_000003	reduce	201005131719_0001	RUNNING	2010-05-13 19:20 GMT+	0 hours 12 minutes	View Attempts
r_000002	reduce	201005131719_0001	RUNNING	2010-05-13 19:20 GMT+	0 hours 12 minutes	View Attempts
r_000001	reduce	201005131719_0001	RUNNING	2010-05-13 19:20 GMT+	0 hours 12 minutes	View Attempts
r_000000	reduce	201005131719_0001	RUNNING	2010-05-13 19:20 GMT+	0 hours 12 minutes	View Attempts
m_000003	map	201005131719_0001	RUNNING	2010-05-13 19:20 GMT+	0 hours 12 minutes	View Attempts
m_000002	map	201005131719_0001	RUNNING	2010-05-13 19:20 GMT+	0 hours 12 minutes	View Attempts
m_000001	map	201005131719_0001	RUNNING	2010-05-13 19:20 GMT+	0 hours 12 minutes	View Attempts

Wozu MapReduce?

- Problem: Petabytes an Daten, auf mehrere Computer verteilt, müssen verarbeitet oder analysiert werden
- Beispiel: Wie oft kommen welche Wörter in Textdateien (z.B. Suchlogs) vor?
- Eigentliche Funktionalität (Wörter zählen) trivial implementierbar
- Aber wie verteilen?
- MapReduce abstrahiert den Verteilungsvorgang

Was ist MapReduce?

- Beschränkung auf zwei Funktionen:
  - Map
  - Reduce
- Inspiriert durch *map* und *reduce* beim funktionalen Programmieren

Exkurs: Die map-Funktion beim funktionalen Programmieren

- Wendet Funktion f(x) auf alle Werte x einer Liste an und gibt diese Werte als neue Liste zurück
- Gibt an, *was* getan wird, nicht *wie*
- Funktion f wird dazu vom Programmierer definiert

Exkurs: Die map-Funktion beim funktionalen Programmieren

Karlsruher Institut für Technologie

Es soll zu einer Liste mit Zahlen die Liste mit deren Quadraten erstellt werden.

- Imperativ:

```
squares = []
for number in [1, 2, 3, 4]: squares.append(number**2)
```

- Mit map:

```
def f(x): return x**2
squares = map(f, [1, 2, 3, 4])
```

Exkurs: die reduce-Funktion beim funktionalen Programmieren

- „Reduziert” anhand der Funktion g(x, y) Liste von Werten auf einen einzigen Wert
- g bekommt x als bisheriges reduce-Ergebnis, y als nächsten Wert
- reduce(g, [a, b, c]) = g(g(a, b), c)

Exkurs: die reduce-Funktion beim funktionalen Programmieren

Karlsruher Institut für Technologie

Es sollen alle Zahlen einer Liste addiert werden.

- Imperativ:

```
sum = 0
for number in [1, 2, 3, 4]: sum += number
```

- Mit reduce:

```
def g(x, y): return x + y
sum = reduce(g, [1, 2, 3, 4])
```

Map bei MapReduce

- Achtung: Entspricht grob *f* bei der funktionalen map-Funktion
- Liest Schlüssel-Wert-Paare
- Bearbeitet die Daten
- Gibt Schlüssel-Wert-Paare aus

```
map:
Type1 key, Type2 value -> List<Type3, Type4> KVPairs
```

Beispiel: Wörter zählen

- Map
  - Eingabe
    - Schlüssel: Dokument-ID
    - Wert: Inhalt
  - Ausgabe für jedes Wort W im Dokumenten-Inhalt
    - Schlüssel: Wort W
    - Wert: 1

Reduce bei MapReduce

- Entspricht *g* bei der funktionalen reduce-Funktion
- Erhält Daten aus dem Ergebnis der Map-Funktion
- Aggregiert Werte gleichen Schlüssels

```
reduce: Type3 key, List<Type4> values -> Type5 result
```

Beispiel: Wörter zählen

- Map
  - Eingabe
    - Schlüssel: Dokument-ID
    - Wert: Inhalt
  - Ausgabe für jedes Wort W im Dokumenten-Inhalt
    - Schlüssel: Wort W
    - Wert: 1
- Reduce
  - Eingabe: Wort als Schlüssel, Werteliste zu diesem Schlüssel
  - Addiert alle Werte (1 + 1 + 1 + ...)
  - Ergebnis ist die Gesamtzahl der Vorkommnisse dieses Wortes

Vorteil von MapReduce

- Map-Funktion für einen Teil der Daten unabhängig vom Rest
- Reduce-Funktion für einen Schlüssel unabhängig von anderen Schlüsseln

⇒ Eine automatische Parallelisierung ist möglich!

Beispiel: Distributed Grep

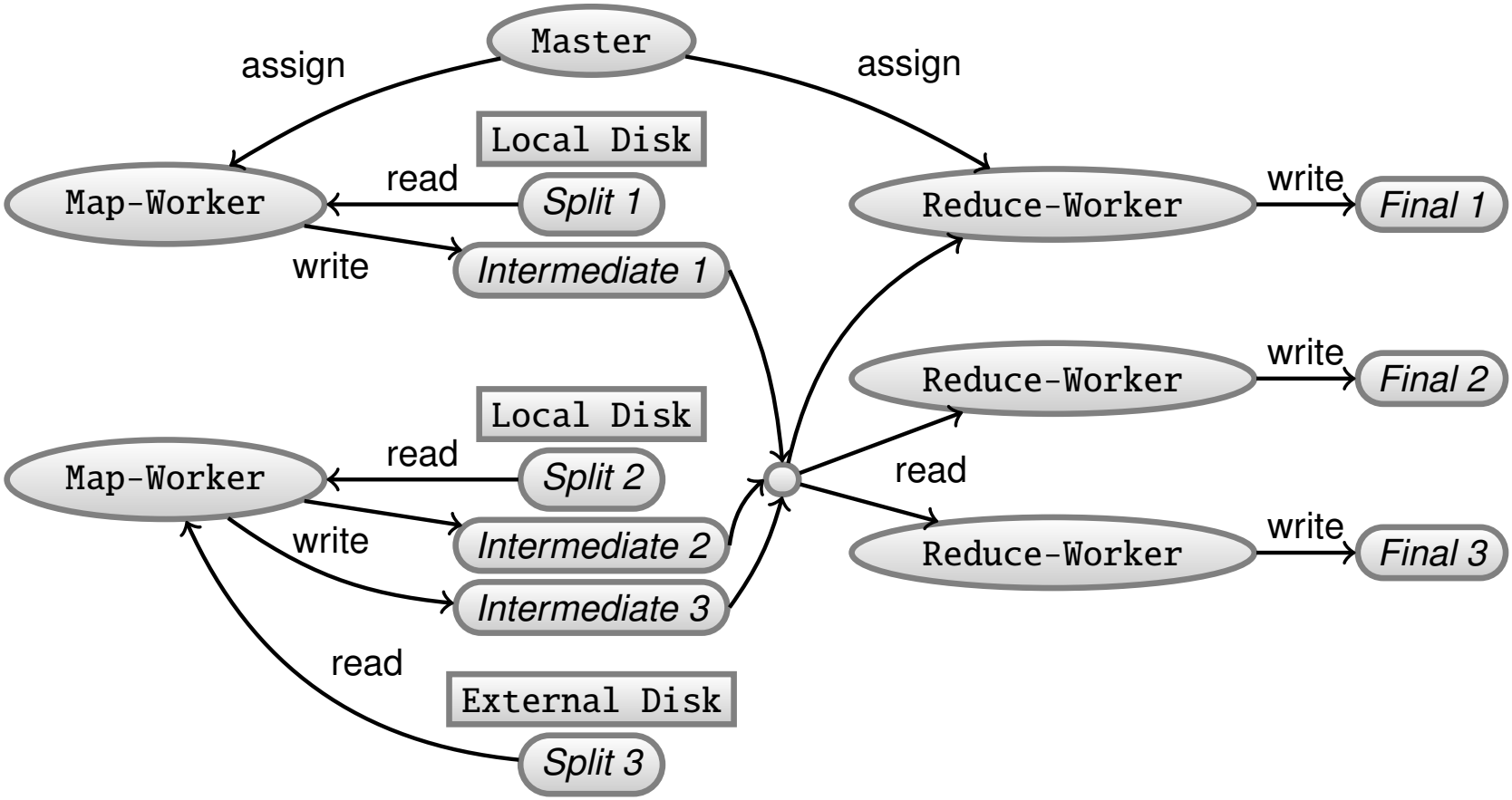
Dokumente sollen nach einem angegebenen regulären Ausdruck R durchsucht werden.

- Map
  - Eingabe
    - Schlüssel: Dokument-ID
    - Wert: Inhalt
  - Ausgabe bei Treffer für R im Dokument
    - Schlüssel: Dokument-ID
    - Wert: true
- Reduce
  - Identitätsfunktion: Gewünschtes Ergebnis ist als Schlüsselliste bereits vorhanden

MapReduce bei Google

- Eng verknüpft mit Googles verteiltem Dateisystem GFS
- Optimierung: bietet zusätzlich optionale Combine-Funktion

MapReduce bei Google



MapReduce bei Google

Karlsruher Institut für Technologie

Implementierung mit hoher Fehlertoleranz gegen:

- fehlerhafte Daten
- fehlerhafte Programme
- fehlerhafte Hardware

Einsatzgebiete bei Google

- “Googlers’ hammer for 80% of our data crunching”
- Suchindex bauen
- Suchlogs analysieren (Google Zeitgeist)
- Satellitenbilder zusammenfügen

Hadoop MapReduce

Karlsruher Institut für Technologie

- Freie Implementierung von MapReduce
- Features orientieren sich stark an Google MapReduce
- Lässt sich leicht lokal ausführen / debuggen
- Wird von Amazon Web Services als Dienst (Elastic MapReduce) angeboten

Fazit

- Gut geeignet für datenintensive Batch-Jobs mit unabhängigen Operationen
- Ungeeignet für Berechnungen mit Abhängigkeiten / Kommunikationsbedarf zwischen den Prozessen
- Hadoop / Elastic MapReduce ermöglichen Verwendung außerhalb von Google

Fragen? Kommentare?

Karlsruher Institut für Technologie

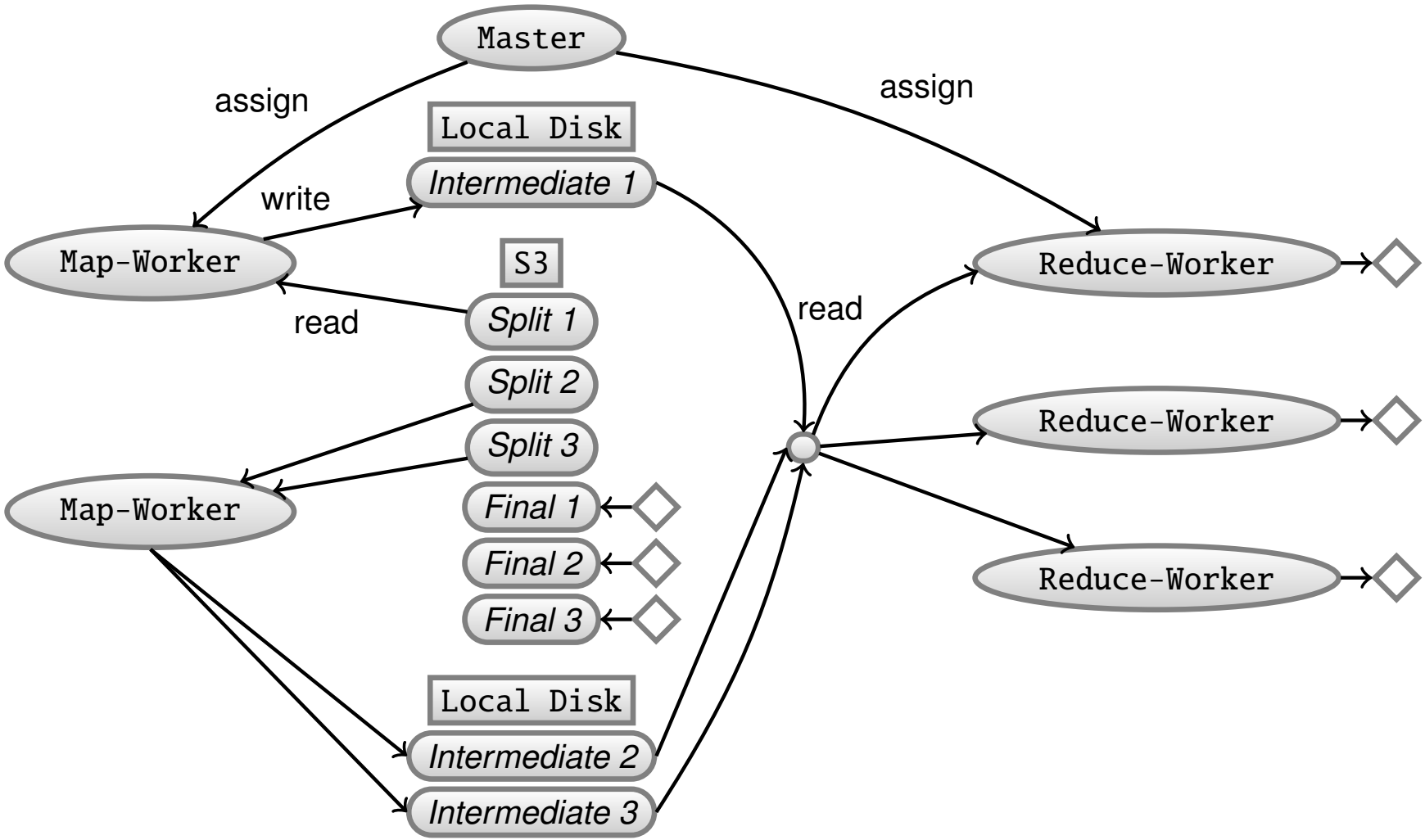
Links:

- http://labs.google.com/papers/mapreduce.html
- http://aws.amazon.com/elasticmapreduce/
- http://hadoop.apache.org/

Zusatz: Amazon Elastic MapReduce

- Bietet Hadoop als Service an
- Setzt auf Elastic Compute Cloud (EC2) und Simple Storage Service (S3) auf
- *Keine* Datenlokalität!
- Daten müssen zu Amazon hochgeladen und Ergebnisse wieder heruntergeladen werden
- Kein Aufwand für Hadoop-Setup
- Abrechnung nach benutztem Speicherplatz und Maschinenstunden

Zusatz: Amazon Elastic MapReduce - Architektur



Zusatz: Beispiel Satellitenbilder zusammenfügen

- Map: Zuordnung von Bildern unterschiedlicher Anbieter zu diskreten Gebieten, konvertieren in einheitliches Format
- Reduce: Zusammenfügen der Bilder für jedes Gebiet