

Handling Data Skew in MapReduce

B. Gufler♠, N. Augsten♣, A. Reiser♠, A. Kemper♠

♠ Technische Universität München

♣ Free University of Bolzano-Bozen

May 8, 2011

Cloud Data Processing with MapReduce

MapReduce for Business

- ▶ usage statistics, index building, . . .
- ▶ simple reducer tasks

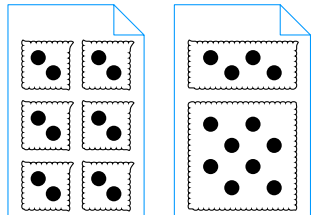
Cloud Data Processing with MapReduce

MapReduce for Business

- ▶ usage statistics, index building, ...
- ▶ simple reducer tasks

MapReduce for eScience: new challenges

1. heavy data skew



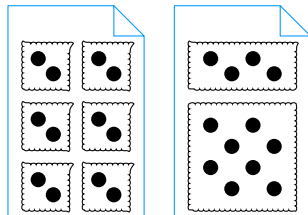
Cloud Data Processing with MapReduce

MapReduce for Business

- ▶ usage statistics, index building, ...
- ▶ simple reducer tasks

MapReduce for eScience: new challenges

1. heavy data skew
2. complex reducer tasks



Cloud Data Processing with MapReduce

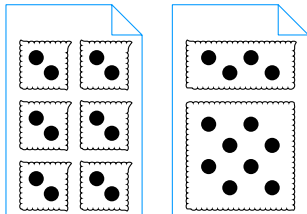
MapReduce for Business

- ▶ usage statistics, index building, ...
- ▶ simple reducer tasks

MapReduce for eScience: new challenges

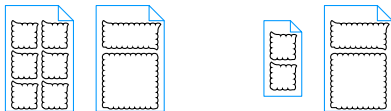
1. heavy data skew
2. complex reducer tasks

~> **How to deal with these problems?**

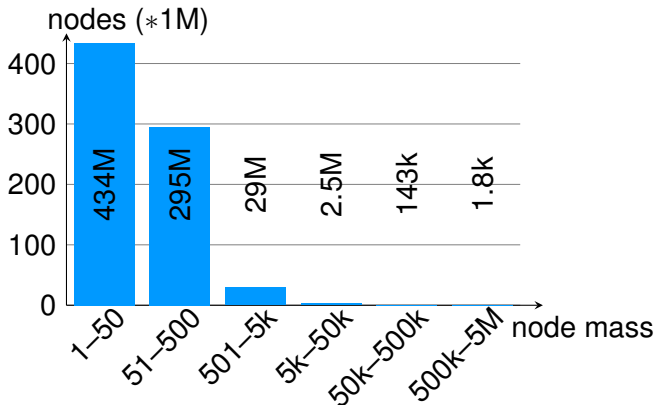


Data Skew

- ▶ values of an attribute not evenly distributed
- ▶ very common in scientific data
- ▶ causes load imbalance if attribute is used for partitioning

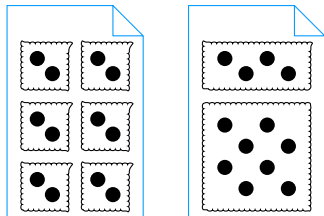
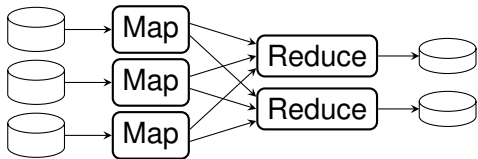


Example: Node Masses in Millennium Simulation



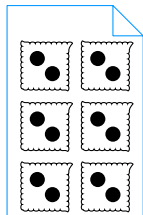
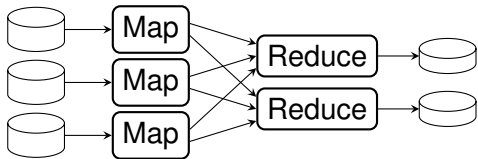
Complex Reducer Tasks

- ▶ complex scientific analysis algorithms
 - ▶ often polynomial or even exponential complexity
- ▶ amplifies load imbalance

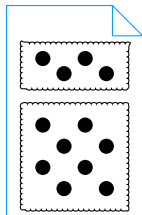


Complex Reducer Tasks

- ▶ complex scientific analysis algorithms
 - ▶ often polynomial or even exponential complexity
- ▶ amplifies load imbalance

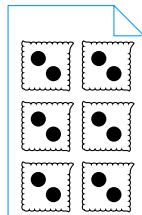
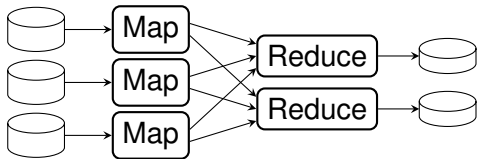


$$6 \cdot 2^2 = 24$$

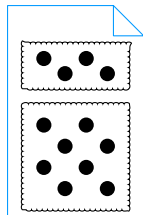


Complex Reducer Tasks

- ▶ complex scientific analysis algorithms
 - ▶ often polynomial or even exponential complexity
- ▶ amplifies load imbalance



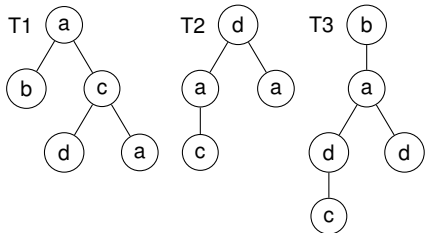
$$6 \cdot 2^2 = 24$$



$$4^2 + 8^2 = 80$$

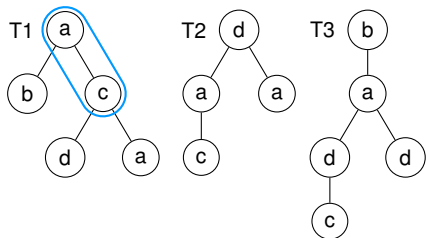
Example: Frequent Subtree Mining

- ▶ find common substructures in (large parts of) a forest
- ▶ exponential complexity for unordered subtrees



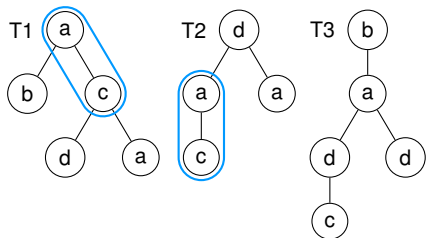
Example: Frequent Subtree Mining

- ▶ find common substructures in (large parts of) a forest
- ▶ exponential complexity for unordered subtrees



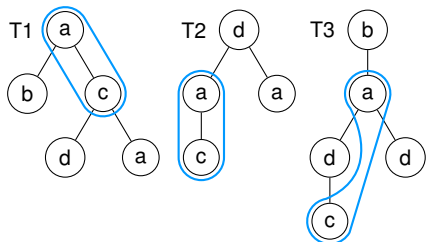
Example: Frequent Subtree Mining

- ▶ find common substructures in (large parts of) a forest
- ▶ exponential complexity for unordered subtrees



Example: Frequent Subtree Mining

- ▶ find common substructures in (large parts of) a forest
- ▶ exponential complexity for unordered subtrees



Summary: MapReduce for eScience

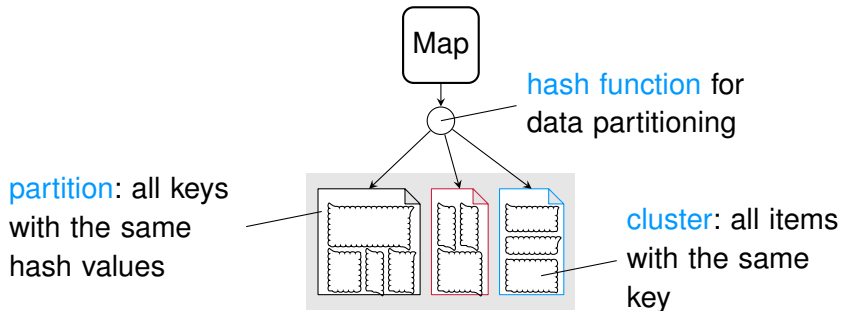
Challenges

- ▶ load imbalance due to data skew
- ▶ complex analysis amplifies execution time differences
- ▶ example: frequent subtree mining (PathJoin algorithm)
 - ▶ 16 GB data set (subset of the Millennium simulation)
 - ▶ cluster of 16 nodes
 - ▶ total execution time: 8 hours
 - ▶ execution time difference of reducers: up to 6 hours

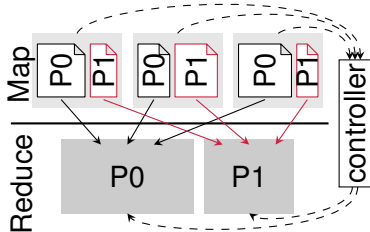
Agenda

- ▶ Motivation
- ▶ Load Balancing in MapReduce: Big Picture
- ▶ Estimation of Partition Cost
- ▶ Assignment of Partitions to Reducers
- ▶ Number of Partitions
- ▶ Evaluation

Data Partitioning in MapReduce



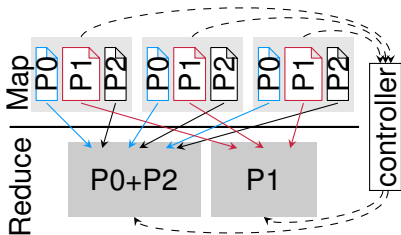
Load Balancing: Current MapReduce



- ▶ hash partitioning on each mapper
- ▶ static 1:1 assignment of partitions to reducers
- ▶ balances number of keys per reducer instead of workload
- ▶ no flexibility

Improved Load Balancing

- ▶ create more partitions than there are reducers
- ▶ cost-based assignment of partitions to reducers



- ▶ flexibility for load balancing
- ▶ challenges
 1. compute number of partitions
 2. estimate partition cost
 3. assign partitions to reducers

Estimate Partition Cost

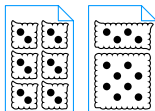
partition cost depends on

1. **number** and **size** of the **clusters** within the partition
 - ▶ locally monitored on each mapper
 - ▶ centrally aggregated on one controller
2. **complexity** of the reducer algorithm
 - ▶ specified by the user

~> estimate cluster cost based on this information

~> sum up cluster costs to obtain partition cost

Estimate Partition Cost: Monitoring



- ▶ tuple count
 - ▶ monitor local count on every mapper
 - ▶ sum up on controller
- ▶ cluster count
 - ▶ create Bloom filter on every mapper
 - ▶ employ Linear Counting on controller

Estimate Partition Cost: Monitoring

Mapper 1

P0

tuples: 31

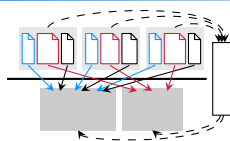
clusters: [01011]

Mapper 2

P0

tuples: 17

clusters: [10011]



Estimate Partition Cost: Monitoring

Mapper 1

P0

tuples: 31

clusters: [01011]

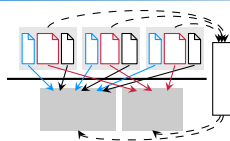
Mapper 2

P0

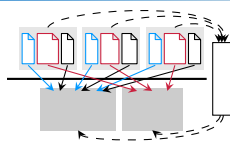
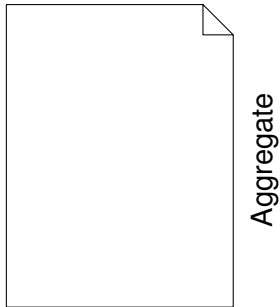
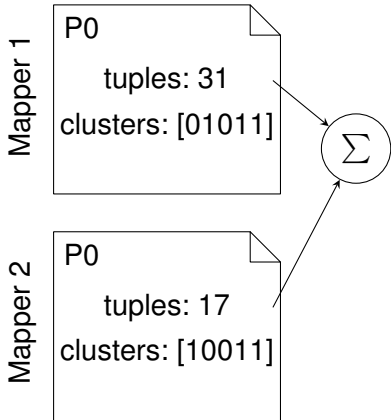
tuples: 17

clusters: [10011]

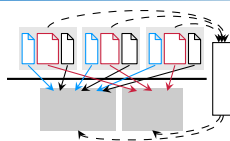
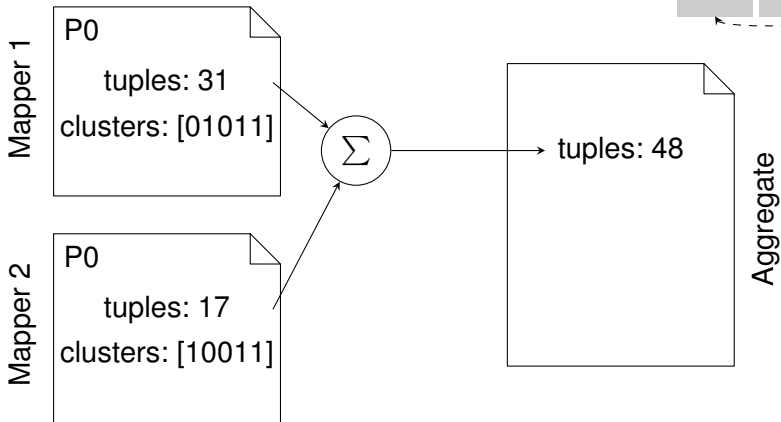
Aggregate



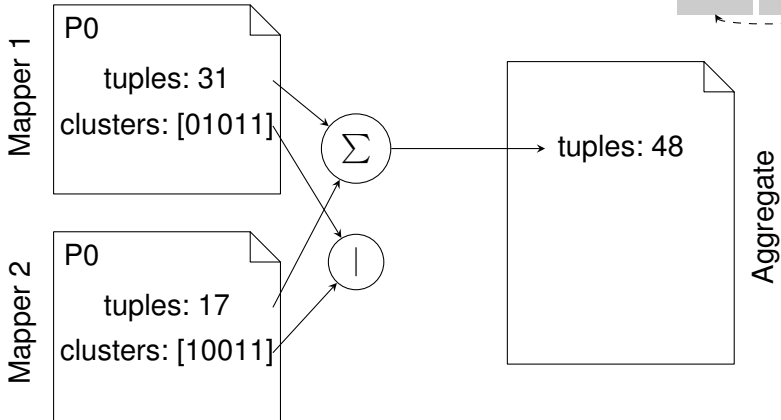
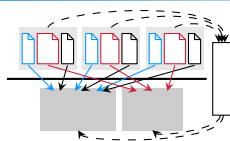
Estimate Partition Cost: Monitoring



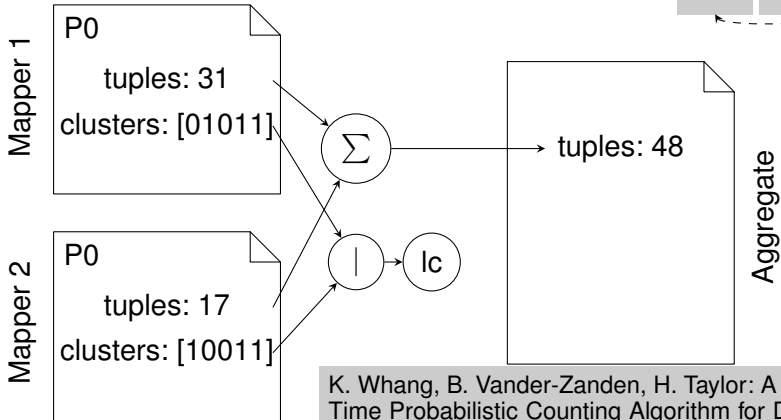
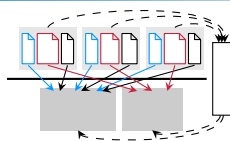
Estimate Partition Cost: Monitoring



Estimate Partition Cost: Monitoring

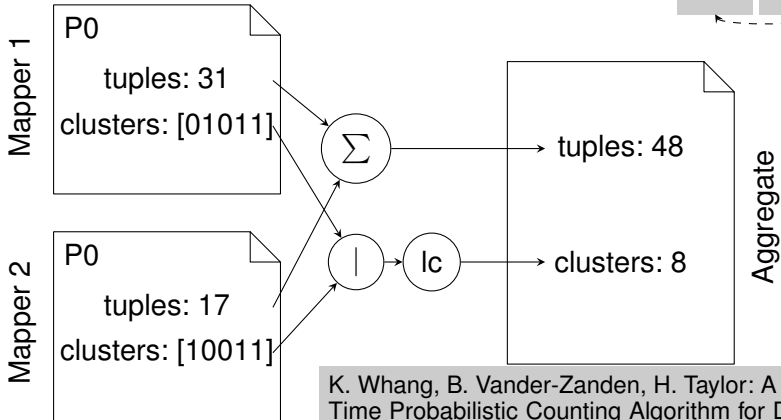
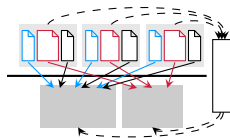


Estimate Partition Cost: Monitoring



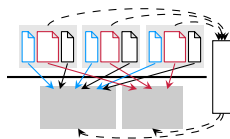
K. Whang, B. Vander-Zanden, H. Taylor: A Linear-Time Probabilistic Counting Algorithm for Database Applications, ACM TODS 15(2), 1990

Estimate Partition Cost: Monitoring



K. Whang, B. Vander-Zanden, H. Taylor: A Linear-Time Probabilistic Counting Algorithm for Database Applications, ACM TODS 15(2), 1990

Estimate Partition Cost: Calculation



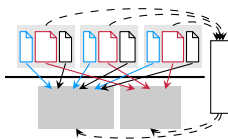
Aggregate

tuples: 48

clusters: 8

reducer complexity: quadratic
(e. g., pairwise comparison)

Estimate Partition Cost: Calculation



Aggregate

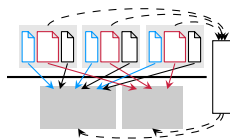
tuples: 48

clusters: 8

- ▶ average cluster size: $\frac{48}{8} = 6$

reducer complexity: quadratic
(e. g., pairwise comparison)

Estimate Partition Cost: Calculation



Aggregate

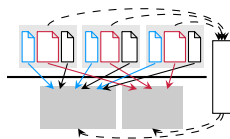
tuples: 48

clusters: 8

- ▶ average cluster size: $\frac{48}{8} = 6$
- ▶ cluster cost: $6^2 = 36$

reducer complexity: quadratic
(e. g., pairwise comparison)

Estimate Partition Cost: Calculation



Aggregate

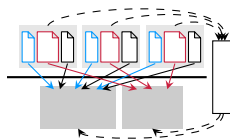
tuples: 48

clusters: 8

- ▶ average cluster size: $\frac{48}{8} = 6$
- ▶ cluster cost: $6^2 = 36$
- ▶ partition cost: $8 \cdot 36 = 288$

reducer complexity: quadratic
(e. g., pairwise comparison)

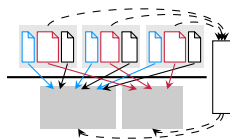
Assign Partitions to Reducers



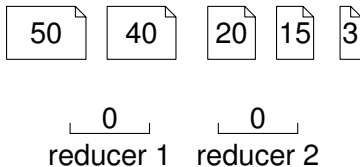
- ▶ assign partitions to reducers balancing the partition cost
- ▶ bin packing problem, but: *NP*-hard
- ▶ rely on a heuristic instead



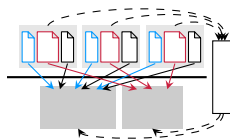
Assign Partitions to Reducers



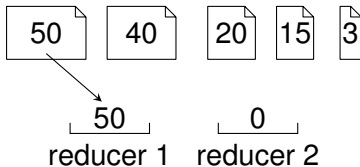
- ▶ assign partitions to reducers balancing the partition cost
- ▶ bin packing problem, but: *NP*-hard
- ▶ rely on a heuristic instead



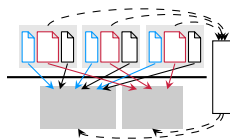
Assign Partitions to Reducers



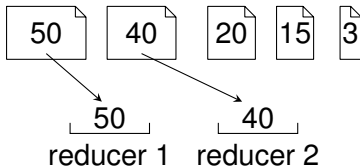
- ▶ assign partitions to reducers balancing the partition cost
- ▶ bin packing problem, but: *NP*-hard
- ▶ rely on a heuristic instead



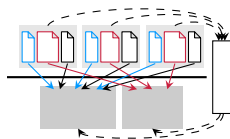
Assign Partitions to Reducers



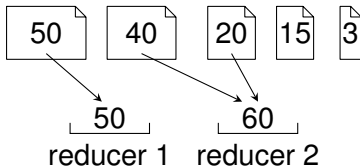
- ▶ assign partitions to reducers balancing the partition cost
- ▶ bin packing problem, but: *NP*-hard
- ▶ rely on a heuristic instead



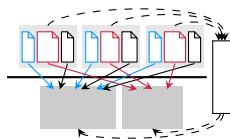
Assign Partitions to Reducers



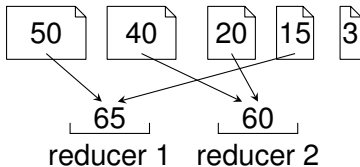
- ▶ assign partitions to reducers balancing the partition cost
- ▶ bin packing problem, but: *NP*-hard
- ▶ rely on a heuristic instead



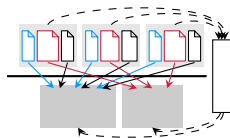
Assign Partitions to Reducers



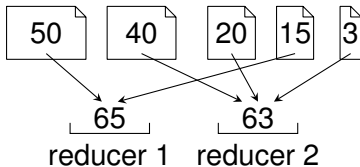
- ▶ assign partitions to reducers balancing the partition cost
- ▶ bin packing problem, but: *NP*-hard
- ▶ rely on a heuristic instead



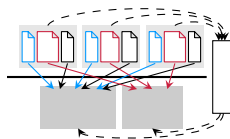
Assign Partitions to Reducers



- ▶ assign partitions to reducers balancing the partition cost
- ▶ bin packing problem, but: *NP*-hard
- ▶ rely on a heuristic instead

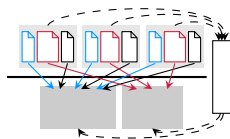


How many Partitions?



- ▶ static approach
 - ▶ number of partitions fixed beforehand by controller
 - ▶ matching partitions created on every mapper

How many Partitions?



- ▶ static approach
 - ▶ number of partitions fixed beforehand by controller
 - ▶ matching partitions created on every mapper
- ▶ dynamic approach
 - ▶ only initial number of partitions fixed
 - ▶ mappers may split some partitions further

Evaluation

- ▶ load balancing quality
- ▶ influence on execution time

Synthetic Data Sets

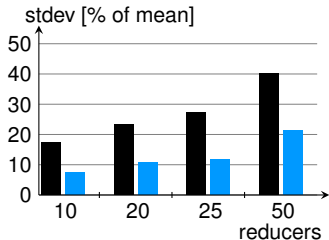
- ▶ 520M tuples
- ▶ 2 000 clusters
- ▶ Zipf distributions

Millennium Data Set

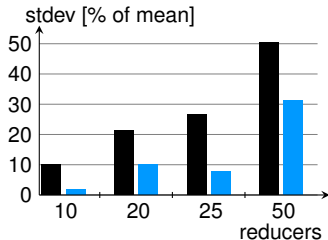
- ▶ 760M tuples
- ▶ $\approx 32\,000$ clusters

Evaluation: Load Balancing

- ▶ measure standard deviation of partition cost per reducer
- ▶ small is good: small differences between reducers



Synthetic Data, Moderate Skew

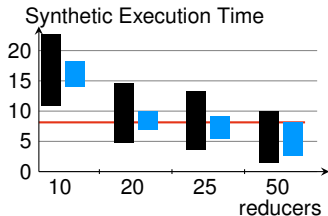


Millennium Data

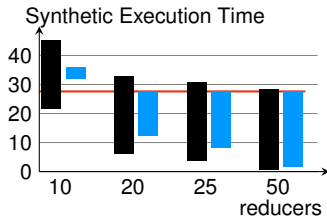


Evaluation: Execution Time

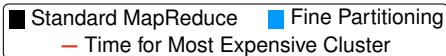
- ▶ measure execution time per reducer
- ▶ max time: overall time of reduce job



Synthetic Data, Moderate Skew

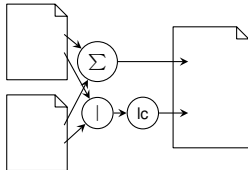
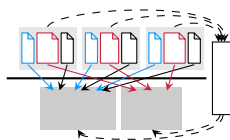
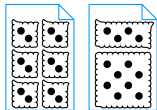


Millennium Data



Summary

- ▶ scientific data analysis with MapReduce
- ▶ partition-based load balancing
- ▶ distributed monitoring for cost estimation
- ▶ experimental evaluation
 - ▶ better load balancing
 - ▶ reduced overall execution time



Thank you!