Map/Reduce and Hadoop performance

Ioana Manolescu
Responsable de l'équipe OAK
Inria Saclay and Université Paris-Sud

IN2P3 summer school, Roscoff, 2013

Plan

- The Map/Reduce model
- Three performance problems and ways out
  - Blocking steps in the Map/Reduce processing pipeline
  - Non-selective data access
  - Repeated work
- Conclusion: toward Map/Reduce optimization?

The Map/Reduce model

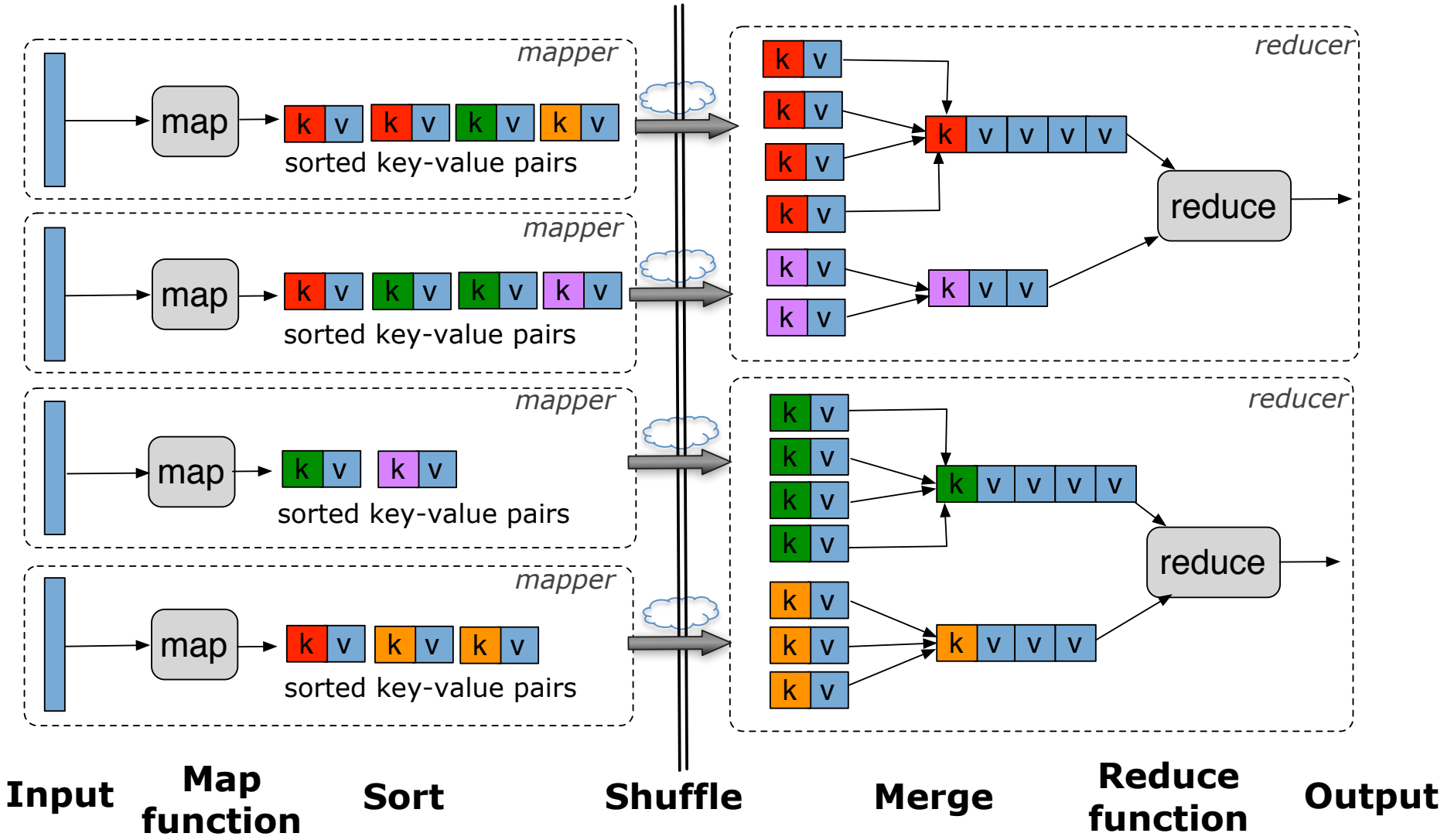
- Problem:
  - How to compute in a distributed fashion a given processing to a very large amount of data
- Map/Reduce solution:
  - Programmer: express the processing as
    - Splitting the data
    - Extract (key, value pairs) from each partition (MAP)
    - Compute partial results for each key (REDUCE)
  - Map/Reduce platform (e.g., Hadoop):
    - Distributes partitions, runs one MAP task per partition
    - Runs one or several REDUCE tasks per key
    - Sends data across machines from MAP to REDUCE

Implicit parallelism

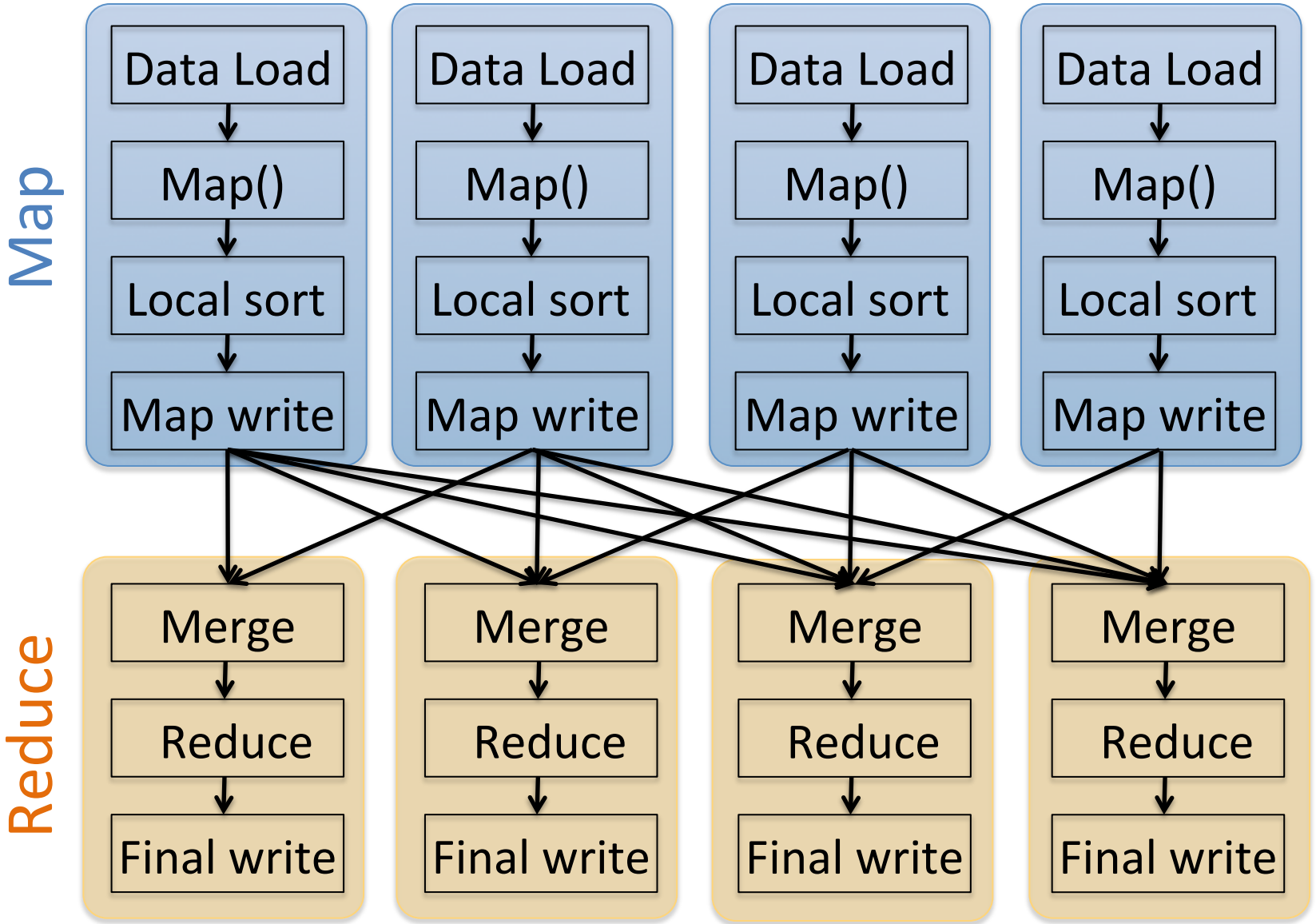
Communication across machines

Fault tolerance

Map/Reduce in detail

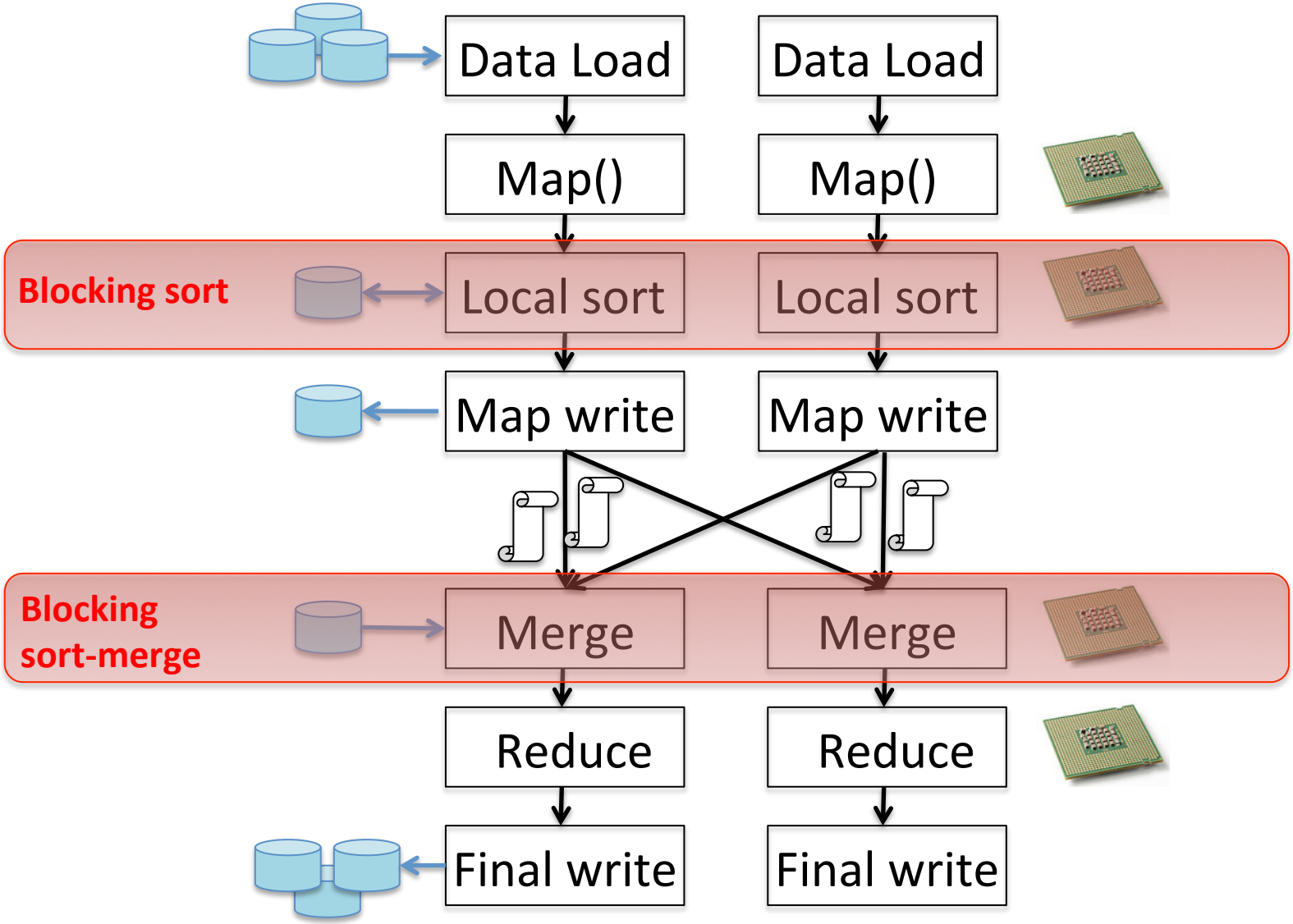


Hadoop Map/Reduce



Performance problem 1:
Idle CPU due to blocking steps

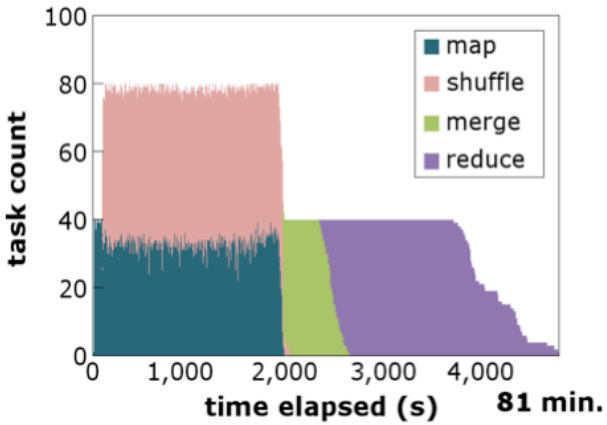
Hadoop resource usage



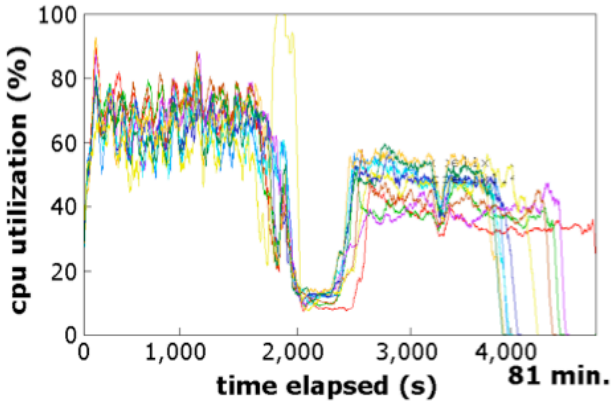
Hadoop benchmark

[Li, Mazur, Diao, McGregor, Shenoy, ACM SIGMOD Conference 2011]

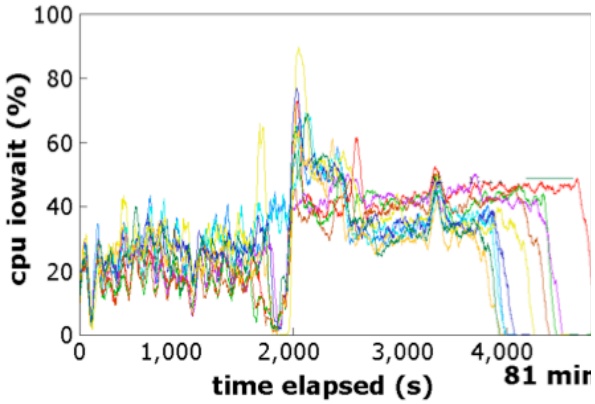
CPU stalls during I/O intensive Merge
Reduce strictly follows Merge



(a) Hadoop: Task timeline.



(b) Hadoop: CPU utilization.



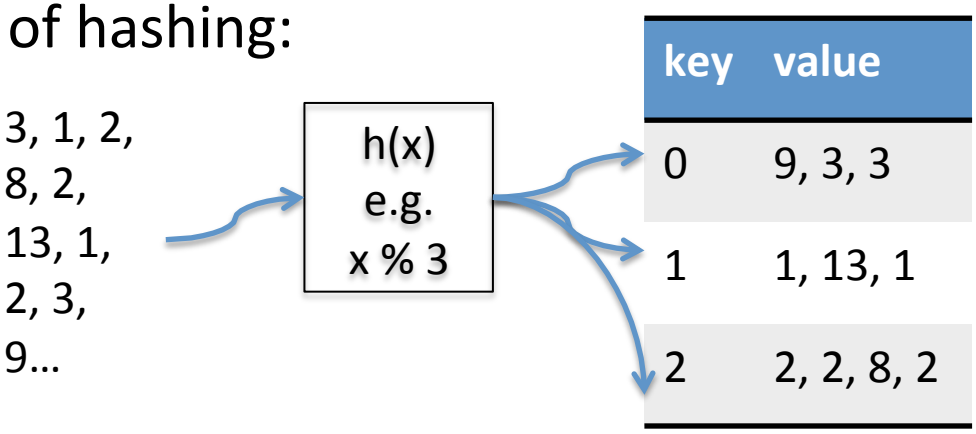
(c) Hadoop: CPU iowait.

Hash-based algorithms to improve Hadoop performance

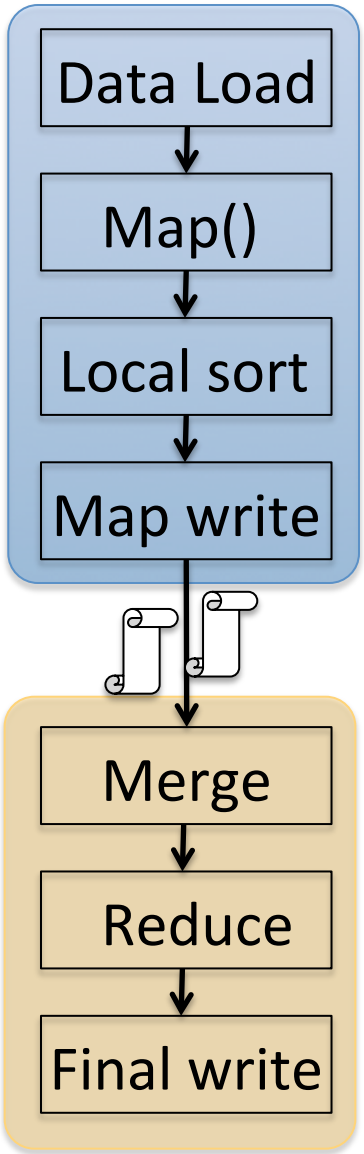
[Li, Mazur, Diao, McGregor, Shenoy, SIGMOD 2011]

Main idea: use non-blocking hash-based algorithms to group items by keys during Map.LocalSort and Reduce.Merge

Principle of hashing:



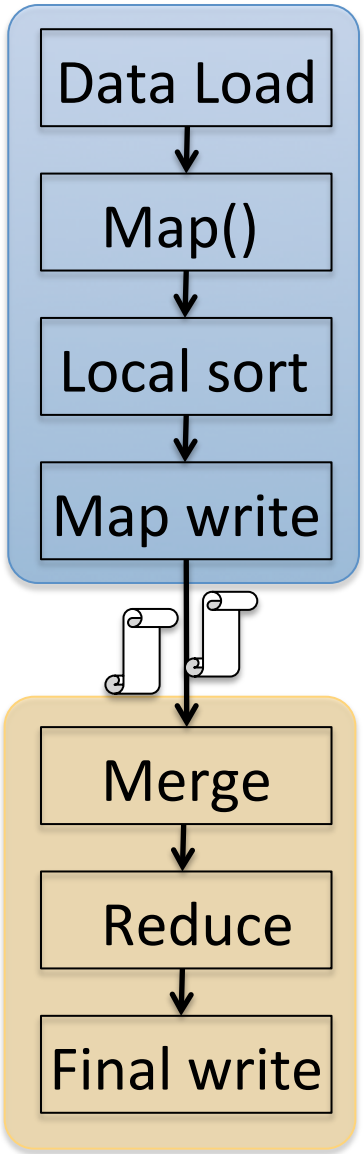
- Partitions can be in memory or flushed to disk
- If the reduce works incrementally, early send



Performance problem 2: non-selective data access

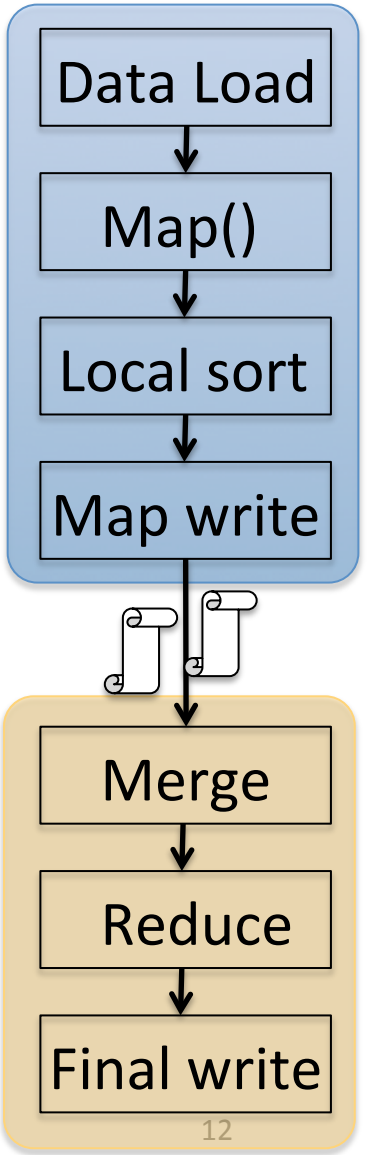
Data access in Hadoop

- Basic model: read all the data
  - If the tasks are selective, we don't really need to!
- Database indexes? But:
  - Map/Reduce works on top of a file system (e.g. Hadoop file system, HDFS)
  - Data is stored only once
  - Hard to foresee all future processing
    - "Exploratory nature" of Hadoop



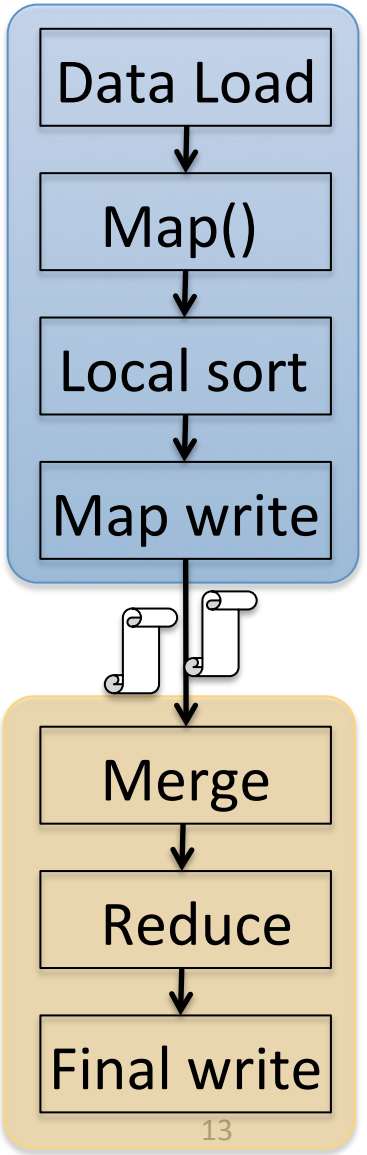
Accelerating data access in Hadoop

- Idea 1: Hadop++
  [Jindal, Quiané-Ruiz, Dittrich, ACM SOCC, 2011]
  - Add header information to each data split, summarizing split attribute values
  - Modify the RecordReader of HDFS, used by the Map() will prune irrelevant splits

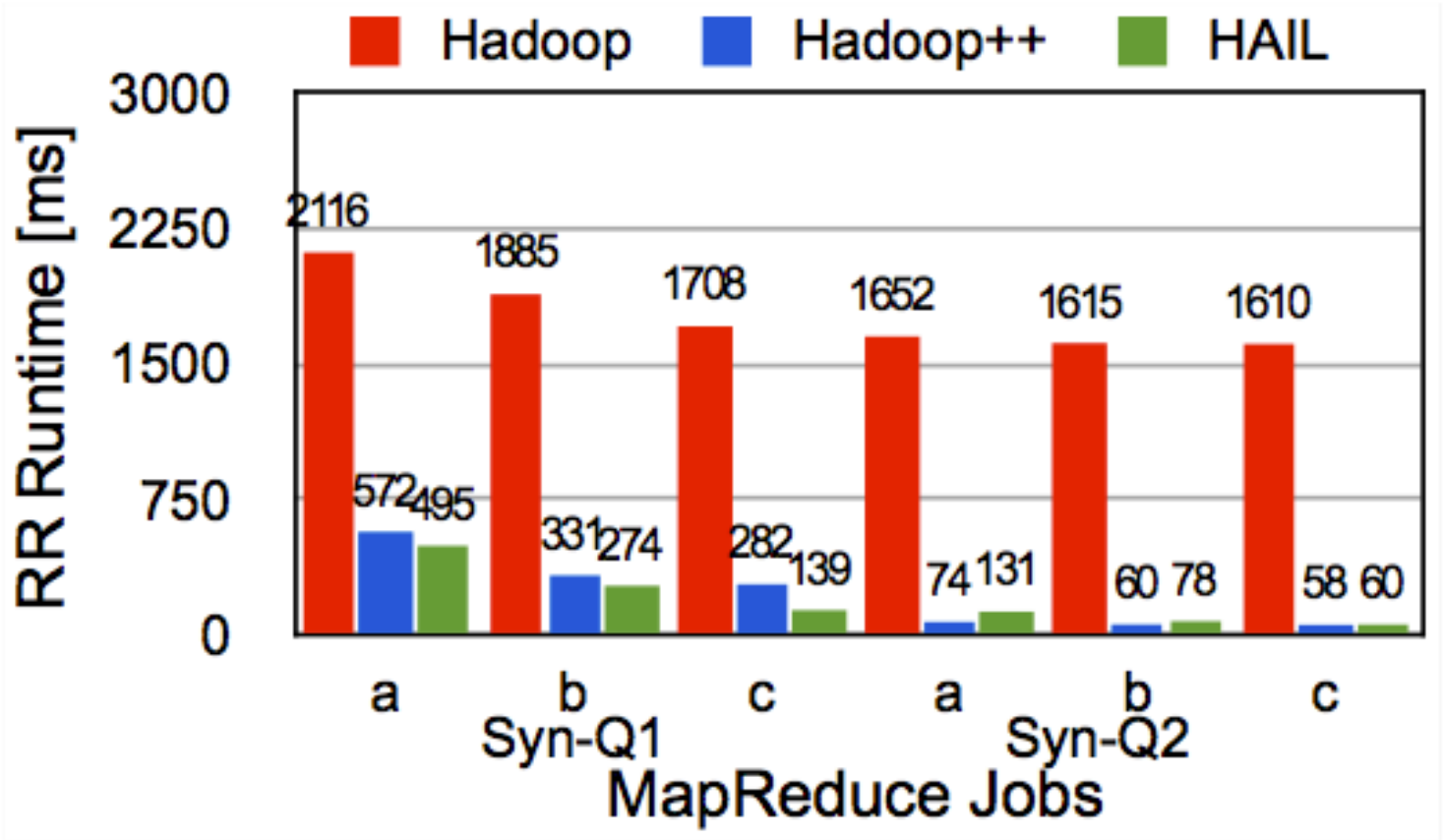


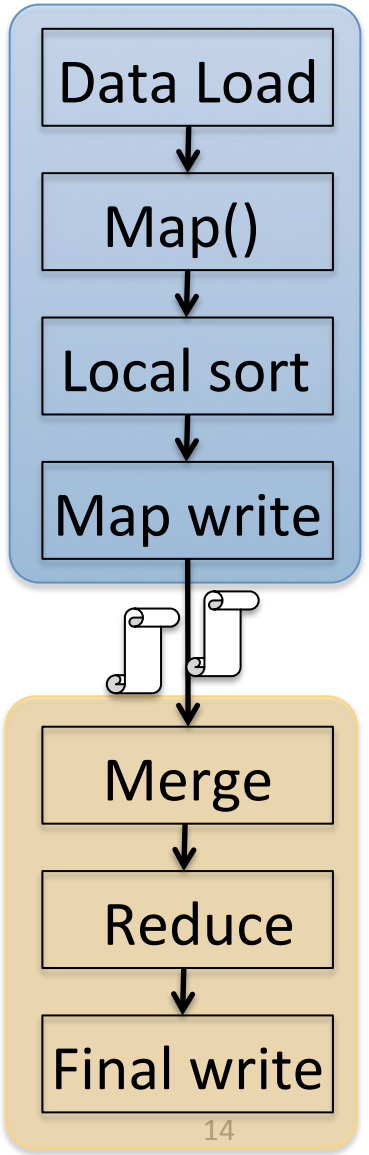
Accelerating data access in Hadoop

- Idea 2: HAIL
  [Dittrich, Quiané-Ruiz, Richter, Schuh, Jindal, Schad, PVLDB 2012]
  - Each storage node builds an in-memory, clustered index of the data in its split
  - There are three copies of each split for reliability → Build three different indexes! ☺
  - Customize RecordReader



Hadoop, Hadoop++ and HAIL





Performance problem 3: repeated work

Problem: Map/Reduce workflows may exhibit repeated work

- Data load tasks, preliminary filtering/ processing…
- Idea: ReStore [Elghandour and Aboulnaga, PVLDB 2012]
  - Store results of fragments from executed Map/ Reruce
  - Recognize when a fragment of a newly submitted workflow had been executed and re-use directly the results

ReStore: re-using fragments of Map/Reduce workflows

- At what level to identify already-executed computations?
  - Code level: hard ☹
  - Better idea: focus only on M/R workflows compiled from PigLating, where there are « operators » such as Filter, Join, Project…
  - Let PigLatin produce and optimize the M/R workflow
  - Identify and re-use results at the level of physical operators.
  - Critique: blind to logical→physical optimization

Conclusion

Toward Map/Reduce optimization

Optimization defined as Hadoop tuning

Hadoop parameters:
[Herodotou, technical report, Duke Univ, 2011]

Variable	Hadoop Parameter	Default Value	Effect
pNumNodes	Number of Nodes		System
pTaskMem	mapred.child.java.opts	-Xmx200m	System
pMaxMapsPerNode	mapred.tasktracker.map.tasks.max	2	System
pMaxRedPerNode	mapred.tasktracker.reduce.tasks.max	2	System
pNumMappers	mapred.map.tasks		Job
pSortMB	io.sort.mb	100 MB	Job
pSpillPerc	io.sort.spill.percent	0.8	Job
pSortRecPerc	io.sort.record.percent	0.05	Job
pSortFactor	io.sort.factor	10	Job
pNumSpillsForComb	min.num.spills.for.combine	3	Job
pNumReducers	mapred.reduce.tasks		Job
pInMemMergeThr	mapred.inmem.merge.threshold	1000	Job
pShuffleInBufPerc	mapred.job.shuffle.input.buffer.percent	0.7	Job
pShuffleMergePerc	mapred.job.shuffle.merge.percent	0.66	Job
pReducerInBufPerc	mapred.job.reduce.input.buffer.percent	0	Job
pUseCombine	mapred.combine.class or mapreduce.combine.class	null	Job
pIsIntermCompressed	mapred.compress.map.output	false	Job
pIsOutCompressed	mapred.output.compress	false	Job
pReduceSlowstart	mapred.reduce.slowstart.completed.maps	0.05	Job
pIsInCompressed	Whether the input is compressed or not		Input
pSplitSize	The size of the input split		Input

Hadoop tuning

[Babu, ACM SoCC 2010]

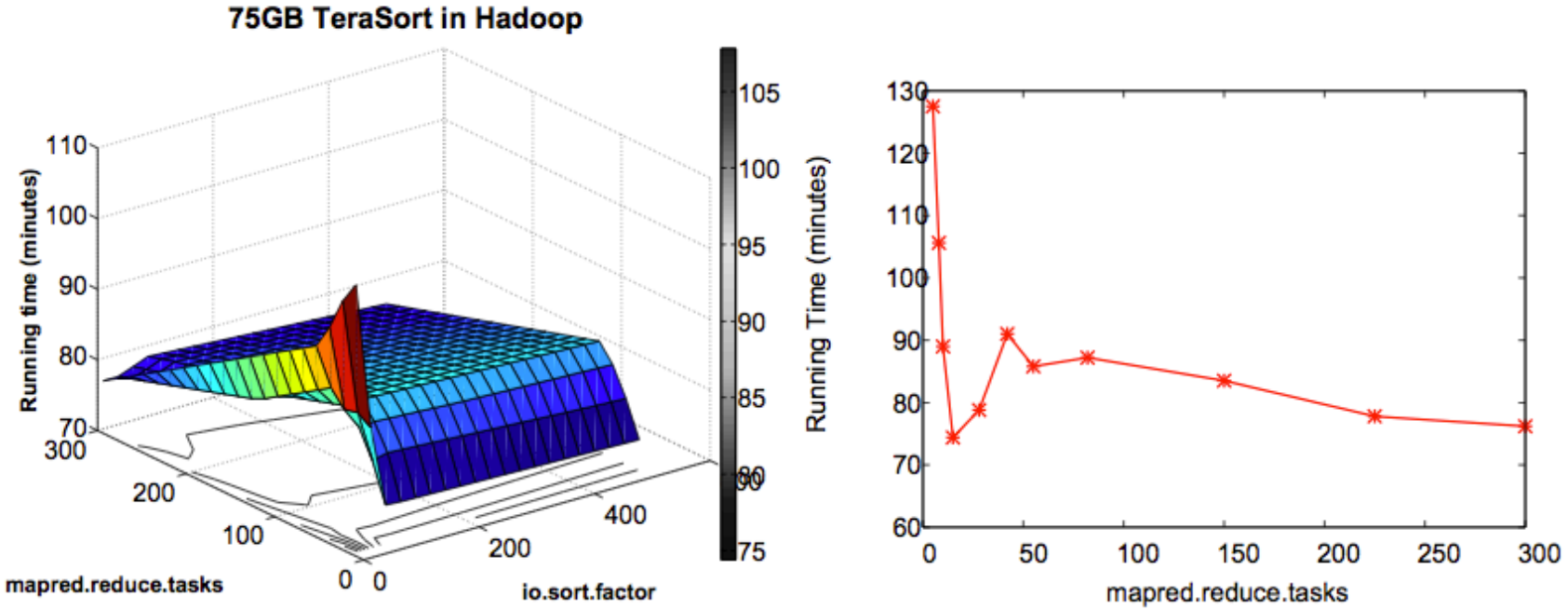


Figure 2: (a) 2D response surface of the TeraSort MapReduce program in Hadoop over a 75GB dataset, with mapred.reduce.tasks $\in [15, 300]$ and io.sort.factor $\in [10, 500]$; (b) a 1D projection of the surface for io.sort.factor=500 for mapred.reduce.tasks $\in [5, 300]$

Hadoop performance model

[Li, Mazur, Diao, McGregor, Shenoy, SIGMOD 2011]

Symbol	Description
(1) System Settings	
R	Number of reduce tasks per node
C	Map input chunk size
F	Merge factor that controls how often on-disk files are merged
(2) Workload Description	
D	Input data size
K_m	Ratio of output size to input size for the map function
K_r	Ratio of output size to input size for the reduce function
(3) Hardware Resources	
N	Number of nodes in the cluster
B_m	Output buffer size per map task
B_r	Shuffle buffer size per reduce task
(4) Symbols Used in the Analysis	
U	Bytes read and written per node, $U = U_1 + \ldots + U_5$ where U_i is the number of bytes of the following types 1: map input; 2: map internal spills; 3: map output; 4: reduce internal spills; 5: reduce output
S_i	Number of sequential I/O requests per node for IO type i
T	Time measurement for startup and I/O cost
h	Height of the tree structure for multi-pass merge

Hadoop performance model

[Li, Mazur, Diao, McGregor, Shenoy, SIGMOD 2011]

Symbol	Description
(1) System Settings	
R	Number of reduce tasks per node
C	Map input chunk size
F	Merge factor
(2) Workload Descriptio	
D	Input data siz
K_m	Ratio of outpu
K_r	Ratio of outpu
(3) Hardware Resources	
N	Number of nc
B_m	Output buffer
B_r	Shuffle buffe
(4) Symbols Used in the	
U	Bytes read an
	U_i is the number of bytes of the following types
	1: map input; 2: map internal spills; 3: map output;
	4: reduce internal spills; 5: reduce output
S_i	Number of sequential I/O requests per node for IO type i
T	Time measurement for startup and I/O cost
h	Height of the tree structure for multi-pass merge

number of I/O requests in a Hadoop job is:

$$S = \frac{D}{CN}\left(\alpha + 1 + \mathbb{1}_{[CK_m > B_m]} \cdot \left(\lambda_F(\alpha,1)(\sqrt{F}+1)^2 + \alpha - 1\right)\right) + R\left(\beta K_r(\sqrt{F}+1) - \beta\sqrt{F} + \lambda_F(\beta,1)(\sqrt{F}+1)^2\right), \quad (3)$$

where $\alpha = \frac{CK_m}{B_m}$, $\beta = \frac{DK_m}{NRB_r}$, $\lambda_F(\cdot)$ *is defined in Eq. 2, and* $\mathbb{1}_{[\cdot]}$ *is an indicator function.*

Easy ☺

References

- Shivnath Babu. "*Towards automatic optimization of MapReduce programs*", ACM SoCC, 2010
- Herodotos Herodotou. "*Hadoop Performance Models*". Duke University, 2011
- Boduo Li, Edward Mazur, Yanlei Diao, Andrew McGregor, Prashant Shenoy. "*A Platform for Scalable One-Pass Analytics using MapReduce*", ACM SIGMOD 2011
- Jens Dittrich, Jorge-Arnulfo Quiané-Ruiz, Stefan Richter, Stefan Schuh, Alekh Jindal, Jorg Schad. "*Only Aggressive Elephants are Fast Elephants*", VLDB 2012
- I. Elghandour, A. Aboulnaga. Restore: "*Reusing Results of MapReduce Jobs*", VLDB 2012
- A.Jindal,J.-A.Quiané-Ruiz,andJ.Dittrich. "*Trojan Data Layouts: Right Shoes for a Running Elephant*" SOCC, 2011
- Harold Lim, Herodotos Herodotou, Shivnath Babu. "*Stubby: A Transformation-based Optimizer for MapReduce Workflows*" PVLDB 2012

Merci

Questions now?

Questions/feedback later on:

ioana.manolescu@inria.fr