INTERMEDIATE DATA I/O INTERFERENCE PREDICTION FROM CO-SCHEDULED TASKS IN MAPREDUCE-HADOOP PROCESSING

Sonia Ikken, Éric Renault, Tahar Kechadi & Abdelkamel Tari

Outline

- Big Data Management from MapReduce-Hadoop processing
- Problem & Motivation
- Methodology
- Markov Model & Prediction Algorithm
- Experimentation Assessment & Validation
- Summary

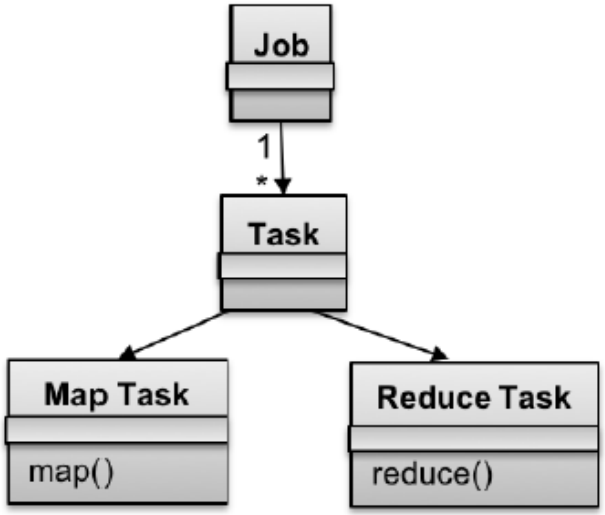
Outline

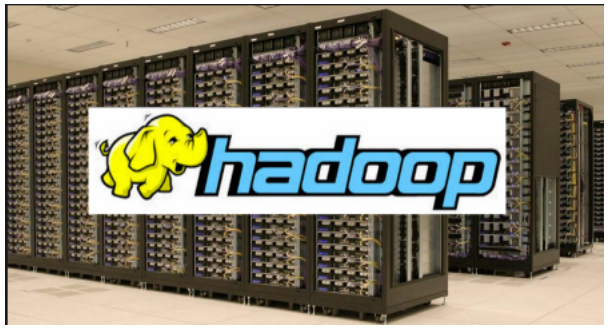
- **Big Data Management from MapReduce-Hadoop processing**
- Problem & Motivation
- Methodology
- Markov Model & Prediction Algorithm
- Experimentation Assessment & Validation
- Summary

Big Data Management

A Little Background on MapReduce-Hadoop

- Origin from Google
- Dataflow parallel programming model
- Large-scale data processing on a cluster of servers (~ 80000 Peta bytes per day)
- Parallel execution of multiple jobs (map tasks + reduce tasks)
- Some examples of jobs: *Web indexing, link graph, Distributed sort, Web access statistics*
- Hadoop, open source implementation



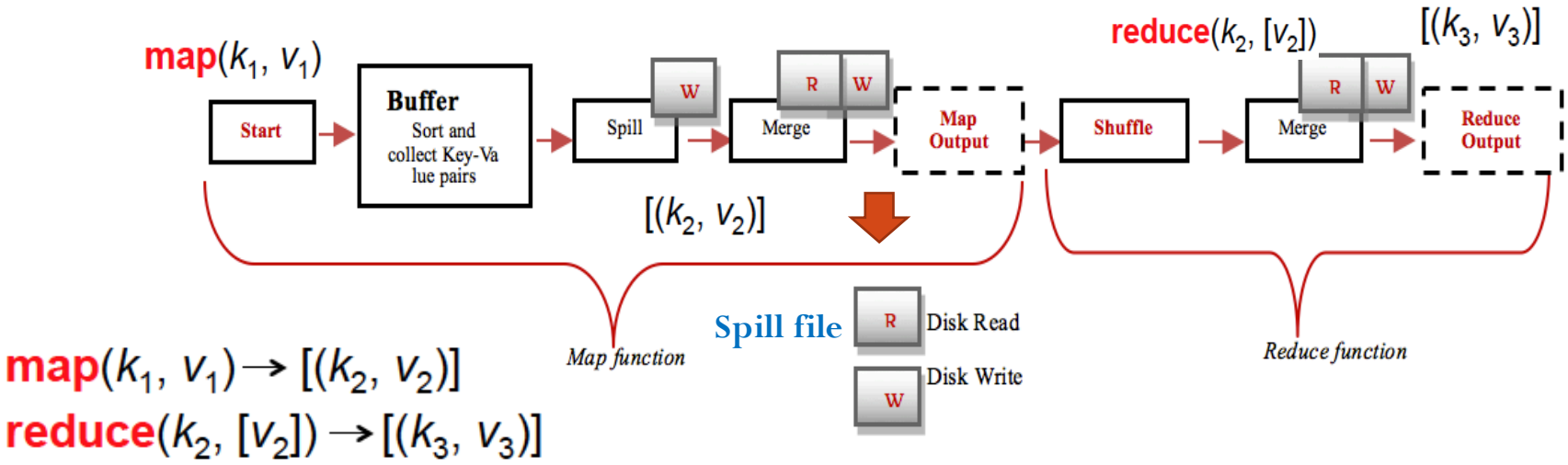


Apache-Hadoop

- Originally developed by Yahoo!
- Running data-intensive applications on a large cluster of commodity machines (replication & scalability)
- Two main components: HDFS & MapReduce
- Used in production by Facebook, IBM, Linkedin, Twitter, Yahoo! and many others

Intermediate data management

- Generation of intermediate data (exceeding 1 PB per day)
- Write-once-read-many access model
- Job are I/O request from tasks (Multiple phases-overlap between tasks)
- A placement strategy of tamporary large data on datanodes of Hadoop servers
- Each map task has its in-memory buffer (circular buffer-100MB default)
- Parallel tasks writing on shared disk (< *key, value* > pairs)



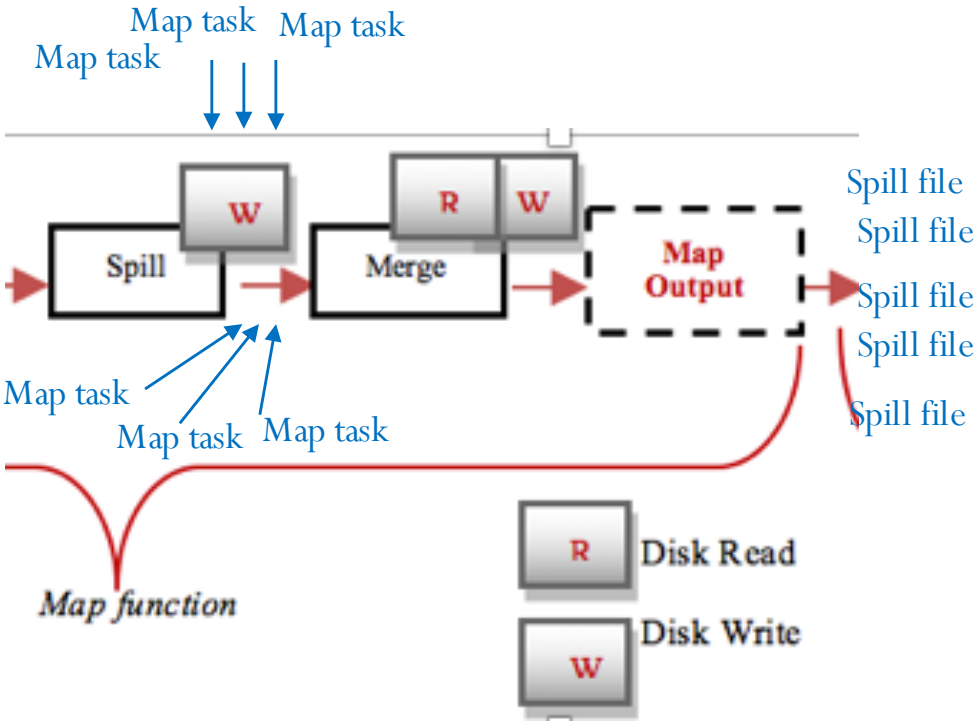
Outline

- Big Data Management from MapReduce-Hadoop processing
- **Problem & Motivation**
- Methodology
- Markov Model & Prediction Algorithm
- Experimentation Assessment & Validation
- Summary

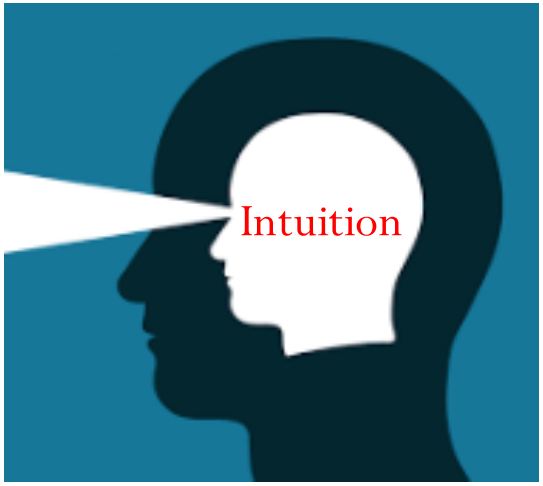
Problem & Motivation

Analyzing I/O behavior of intermediate data placement

- Performing the same I/O operations at the same time (on spill pahse)
- Influencing reduce phase immediately (running time of jobs)
- Performance of applications depends on efficient processing of large intermediate data (map and reduce tasks)



I/O interference on intra-file spill from parallel map tasks



Problem & Motivation

Why to predict intermediate data accesses behavior ?

- Aim: to analyse intermediate data accesses from trace history to improve I/O optimization, data placement and scheduling strategies
- To prove that it is necessary to be able to predict I/O behavior of intermediate data patterns
- Predict exactly what ? ➳ *interfered block segments of spill file*

No focus on I/O interference on spill behavior

- MapReduce modeling [1,2]
- Probabilistic and no statistical models for predicting I/O behavior [3, 4]

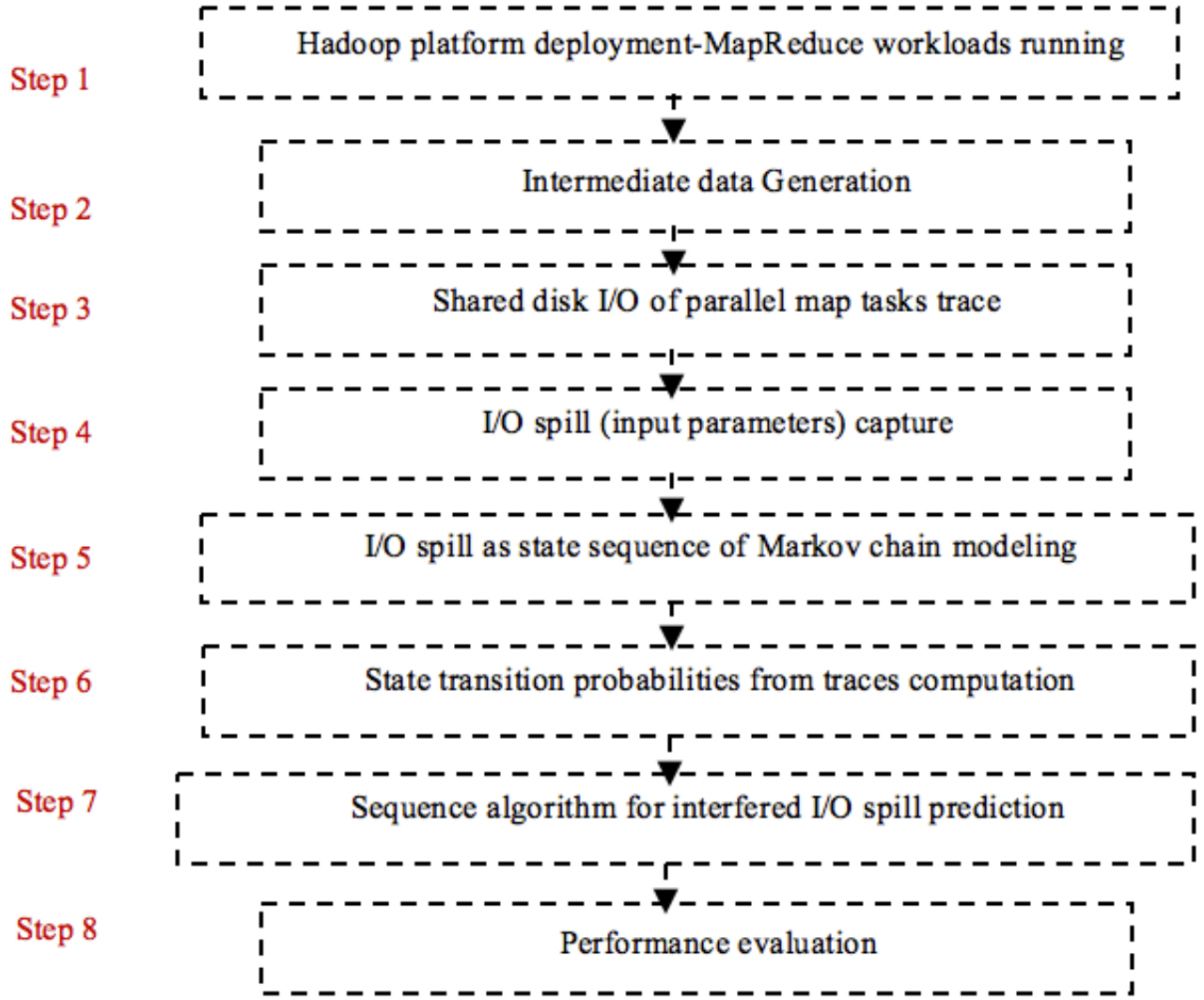
Markov model-based I/O behavior

- Predicting spatial and temporal I/O requests [5, 6, 7, 8]
- Markov property: the probability of finding a future state depends only on the current state
- One must determine what application behavior which corresponds to a state *s*, the total number of states *N* that can be fully described by its transition probability matrix *P,* and the allowed observations
- Each file block is represented by a state in the Markov model

Outline

- Big Data Management from MapReduce-Hadoop processing
- Problem & Motivation
- **Methodology**
- Markov Model & Prediction Algorithm
- Experimentation Assessment & Validation
- Summary

Methodology



Outline

- Big Data Management from MapReduce-Hadoop processing
- Problem & Motivation
- Methodology
- **Markov Model & Prediction Algorithm**
- Experimentation Assessment & Validation
- Summary

Markov Prediction Model

- Markov model based on sequence learning [9] (number of states)
- X_t : state represents an I/O write request of spill phase on discrete state space E
- I/O accesses from concurrent map operations (applications sharing storage resources)
- $|X|$ represents the I/O spill size
- Number of states N: I/O which reflects the map output total size of P parallel map operations

Markov Prediction Model

- State transition:

$$P(x(t) = j_0/x(t-1) = i_1, x(t-2) = i_2,) = P(x(t) = j_0/x(t-1) = i_1) = q_{i_1,j_0}(t)$$
$$\forall j_0, i_1, i_2, \ldots \in E$$

- Transition matrix:

$$Q1_{(t-1;1)} = [q_{i_1,j_0}(t)] = \begin{bmatrix} q_{1,1}(t) & \ldots & q_{1,m}(t) \\ \ldots & \ldots & \ldots \\ q_{m,1}(t) & \ldots & q_{m,m}(t) \end{bmatrix}$$

- Estimation of one-step transition probability

$$P_{i,j} = \frac{count(x_{i,j})}{count(x_{i,.})} = \frac{\sum_{t=0}^{t-1} x_{i,j}(t)}{\sum_{k=1}^{N} \sum_{t=0}^{t-1} x_{i,k}(t)}$$

Training Markov Model

- Application tracing from map operations and I/O spill characteristics
- Blktrace-based I/O tracing [10] (I/O size, LBA, PID, process name, thread-id and arguments)
- Capture transition from block size and the inter-distance between segment spill requests (I/O spill behavior)
- *mapreduce.task.io.sof t.factor* = 0. Parameter of Hadoop specifying the number of I/O spill segments on disk to be merged
- For different system metric values, the Markov model is independent at time *t*
- Contingency table of expected I/O write observations

Markov Prediction Algorithm

Algorithm 1: Prediction of interfered I/O spill requests.

Data: TCk, $Q1[i,j]$, $P_{i,j}$, $STCk$, Sx, x_j, x_i, $|x_i|$, $|b|$

Result: x_{seqk}, $x_{Interfk}$

```
1  while not EoF Q1[i,j] do
2      for i = 1; i < N − 1; i++ do
3          for j = 1; j < N; j++ do
4              Sx ← Sx + x_i ;
5              if P_{i,j} <> 0 and |x_i| < |b| and Sx < STCk then
6                  if x_j ∈ TCk then
7                      x_Seqk ← x_i ;
8                      else
9                      x_Interfk ← x_i ;
10                 end
11             end
12         end
13     end
14 end
15 Return x_Seqk, x_Interfk ;
```

- *Greedy approach*
- *Large number of transitions*
- *All positive transitions*
- *Parameters of a state (spill I/O)*

Parameters	Description
$TC[k,f]$	Set of Map tasks
TCk	$TCk = \{x_1, x_2, ..x_f\} \in TC[k,f]$, set of requests from Map task
$Q1[i,j]$	Transition matrix
$P_{i,j}$	$P_{i,j} \in Q1$: transition probabilities
$STCk$	Total sum of Map task requests
Sx	Total current states of a Map task
x_j	Predicted value of I/O spill request
x_i	Current value of I/O spill
$\|b\|$	I/O buffer size
x_{seqk}	Vector of sequential I/O spill requests for Map task k
$x_{Interfk}$	Vector of interfered I/O spill requests for Map task k

Outline

- Big Data Management from MapReduce-Hadoop processing
- Problem & Motivation
- Methodology
- Markov Model & Prediction Algorithm
- **Experimentation Assessment & Validation**
- Summary

Experimentation Assessment & Validation

Accuracy prediction

- Markov model accuracy on I/O spill block
- Prediction algorithm on interfered spill segment

Trace-driven assessment

- I/O configuration of Hadoop servers for application running (one master server and four data servers)
- Traces: training intermediate data and current intermediate data sets

Parameters	Descriptions
OS	Ubuntu Linux 14.04
Platform	Hadoop 1.0.3 version
Disk	4TB HDD (5400 tours / mn)
Processor	16-cores (2.5 GHz)
Memory	32 GB
Buffer size	80 MB
IOScheduler	CFQ
FileSystem	Ext3 (4 KB block size)
HDFS block size	512 MB

Parameters	Wordcount	Terasort	Kmeans
Dataset	Random	TeraGen	Random
Total Intermediate data size	8192 MB	35840 MB	20480 MB
Map tasks per node	4	17	5
Reduce tasks per node	2	17	3

Experimentation Assessment & Validation

MapReduce application traces

- Wordcount
- Parse the input to find word boundaries
- A lot expensive temporary objetcs (string operation in java)
- CPU bound.
- Terasort
- TeraGen generate 1TB of data
- No compression
- Very I/O intensive
- Input and intermediate data are the same size
- Kmean
- Group items into *k* clusters
- I/O intensive
- Huge amount of intermediate clusters

Experimentation Assessment & Validation

Prediction accuracy metric (markov model and algorithm)

- *Correct Seq/interfI/O*
- *Current I/O Spill*

$$Metric\_Pred\_Acc = \frac{\sum Correct\ Seq/interf\ IO}{Total\ Current\ IO\ Spill}$$

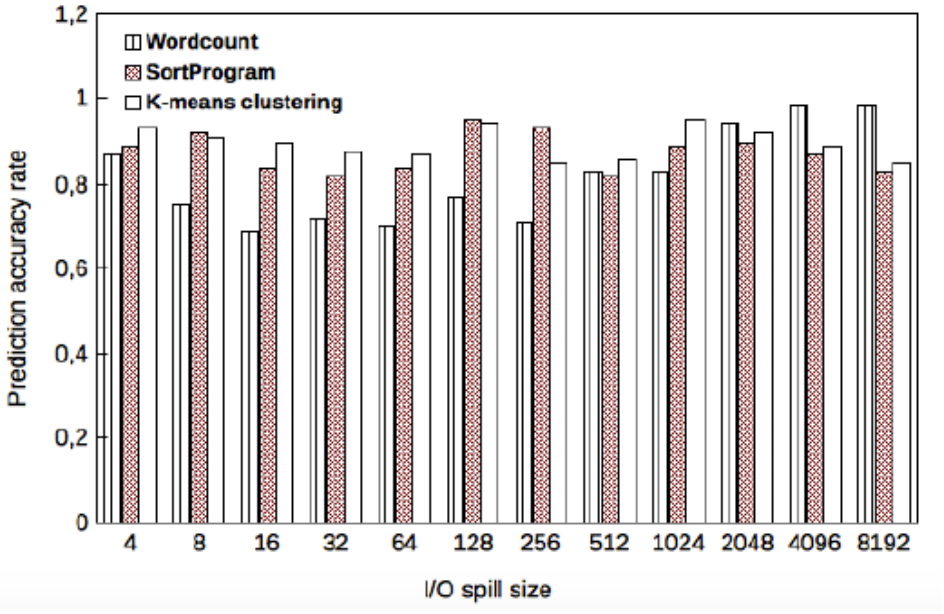
Prediction error

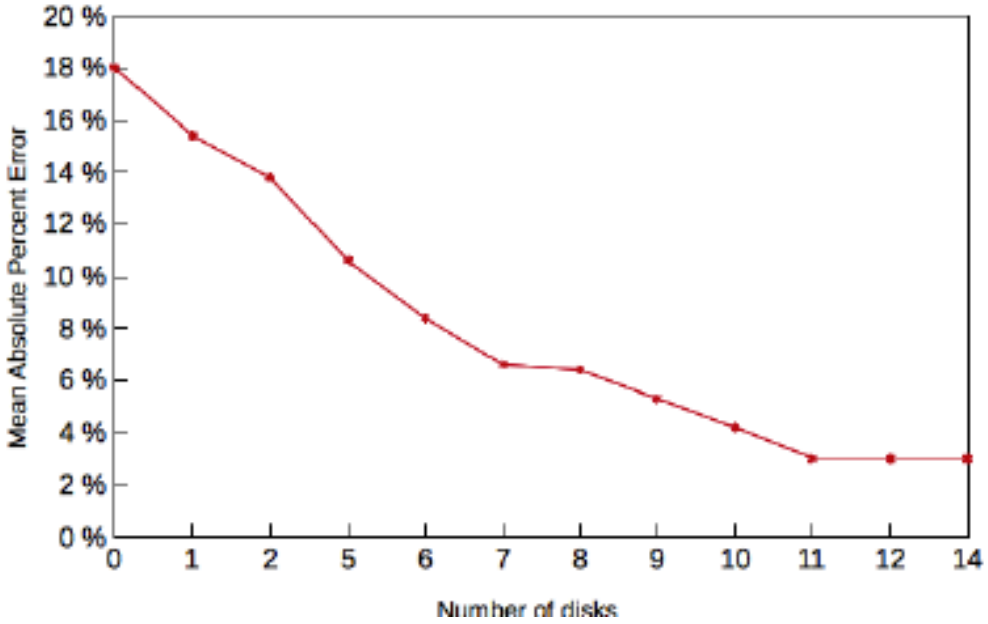
- *Mean Absolute Percent Error (MAPE)*

$$MAPE = \frac{1}{n} \cdot \sum_{j=1}^{n} \left| \frac{A_j - F_j}{A_j} \right|$$

Experimentation Assessment & Validation

Prediction accuracy of Markov model

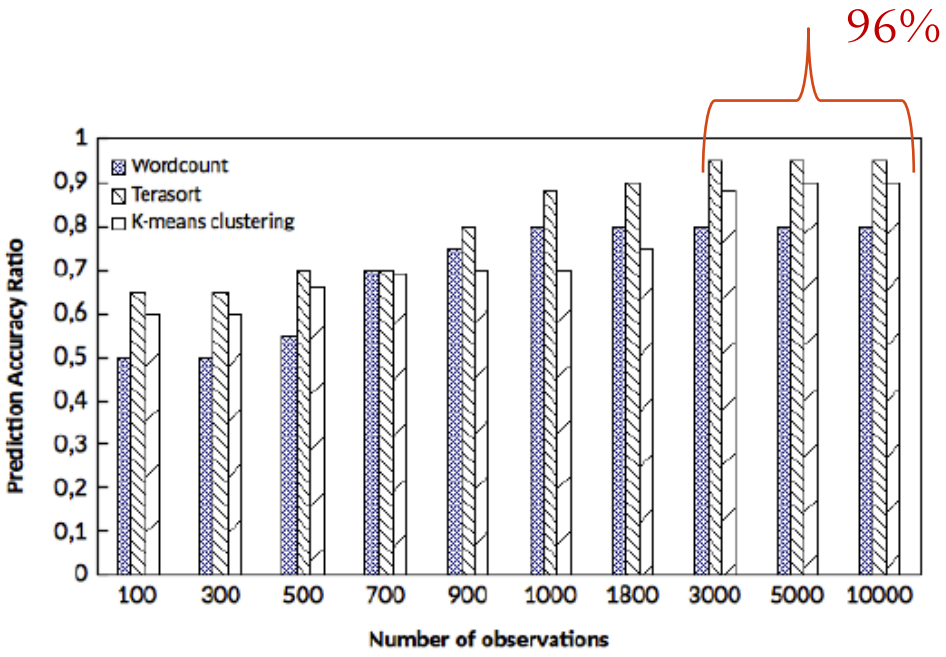




- *Prediction accuracy result of I/O spill size (in KB) for the model size (82%-95%)*
- *Prediction error (MAPE)*

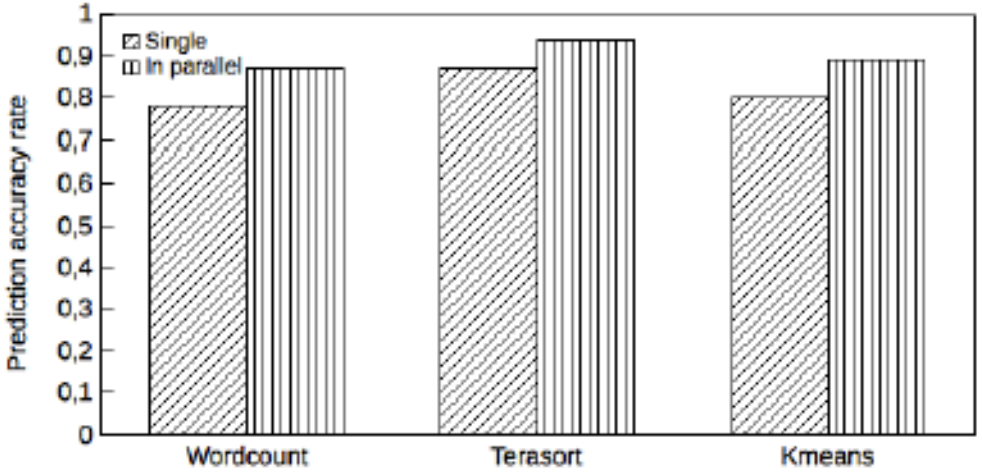
Experimentation Assessment & Validation

Prediction accuracy of the algoithm



- *Interfered I/O spill from sequential and parallel running*

- *Interfered I/O spill from number of observations*



Summary

- **Intermediate data access patterns as a first-class citizen for big data processing**
  - Different behavior from applications (possibly very large amount of intermediate data)
  - Minimizing E/S interference is critical for the overall application running
- **Markov model for I/O spill behavior of Hadoop applications (original Mapreduce)**
  - Large observation to reduce loss of accuracy
  - Offline model trainning for batch processing
- **Additional research needed**
  - Try another model to use some online training
  - Expand workload pattern again
  - Think about replication and faillure

References

[1]: Hailong Yang, Zhongzhi Luan, Wenjun Li, and Depei Qian. Mapreduce workload modeling with statistical approach. Journal of grid computing, 10(2):279–310, 2012

[2]: Sven Groot. Modeling i/o interference in data intensive map-reduce applications. In Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on, pages 206–209. IEEE, 2012

[3]: Daniel A Reed. Scalable Input/Output: achieving system balance. MIT Press, 2004

[4]: Jun He, John Bent, Aaron Torres, Gary Grider, Garth Gibson, Carlos Maltzahn, and Xian-He Sun. I/o acceleration with pattern detection. In Proceedings of the 22nd international symposium on High Performance Parallel and Distributed Computing, pages 25–36. ACM, 2013

[5]: Guanying Wang, Aleksandr Khasymski, KR Krish, and Ali R Butt. Towards improving mapreduce task scheduling using online simulation based predictions. In Parallel and Distributed Systems (ICPADS), 2013 International Conference on, pages 299–306. IEEE, 2013

[6]: Pranav Pathak, Mehedi Sarwar, and Sohum Sohoni. Markov prediction scheme for cache prefetching. In Proceeding of 2nd Annual Conference on Theoretical and Applied Computer Science. November 5, page 14, 2010

[7]: James Oly and Daniel A Reed. Markov model prediction of i/o requests for scientific applications. In Proceedings of the 16th international conference on Supercomputing, pages 147–155. ACM, 2002

[8]: Tara M Madhyastha and Daniel A Reed. Input/output access pattern classification using hidden markov models. In Proceedings of the fifth workshop on I/O in parallel and distributed systems, pages 57–67. ACM, 1997

[9]: Cheng Li, Philip Shilane, Fred Douglis, Darren Sawyer, and Hyong Shim. Assert (! defined (sequential i/o)). In HotStorage, 2014

[10]: Duckjin Kang. Accurate blktrace: ablktrace. 2009

Thank you  *! ! !*

Contact: Sonia.ikken@telecom-sudparis.eu