Enabling Data Processing under Erasure Coding in Fog

Jad Darrous[1] and Shadi Ibrahim[2]

[1]Inria, ENS de Lyon, LIP

[2]Inria, IMT Atlantique, LS2N

Big Data Processing in Fog

Fog Computing: Opportunities

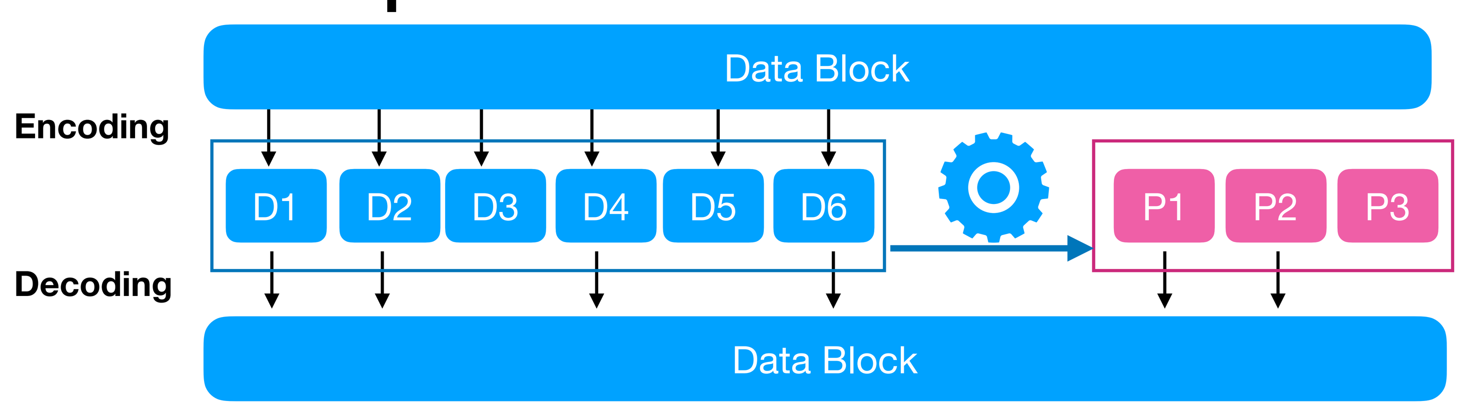
- Fog is widely adopted to extend the capacity of clouds
- It allows to deploy and run applications close to the users, e.g., Smart City applications[1]
- Data processing applications, among others, can also benefit of Fog, e.g., Video Stream processing[2], Query processing[3] and Batch processing[4]

Fog: Limitations and Challenges

- Limited storage capacity
  — Using replication to ensure data availability is expensive
- Heterogeneous and limited computation capacity
  — Difficult to exploit data locality efficiently
- Heterogeneous network
  — The cost of data transfer between nodes is high

Erasure Coding as alternative to replication

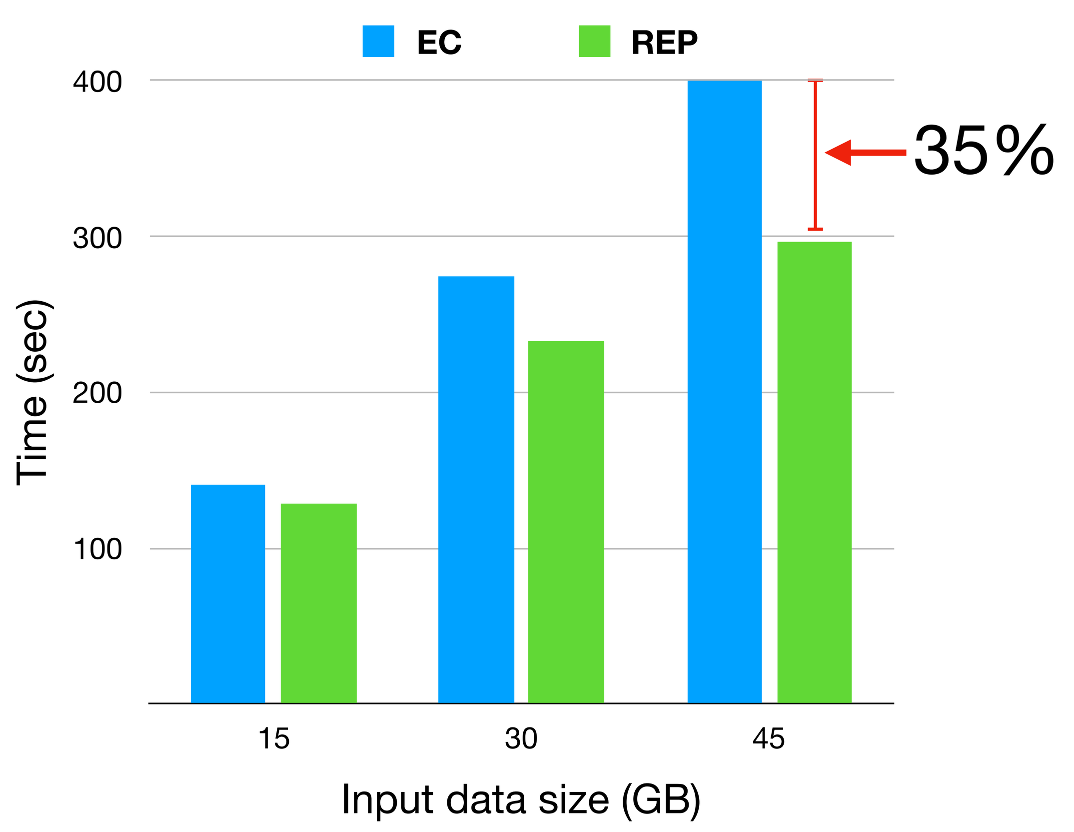
- **Half** the storage overhead of replication, for *RS(6, 3)*
- **Low** CPU overhead for encoding/decoding (5.3 GB/s)[5]
  - EC have been deployed in storage and caching systems[7]
  - HDFS is now equipped with EC since the 3.0.0 release[6]
- When performing MapReduce applications in data-intensive clusters
  - Unlink replication, most of the (map) task input data is **transferred**
  - A **reduction** by half of the network traffic and disk accesses when writing the output data



How to effectively realize EC for big data processing in Fog?

Quantitively analyze the impact of heterogeneity

A Fog infrastructure is emulated on 10 machines each representing a Fog site

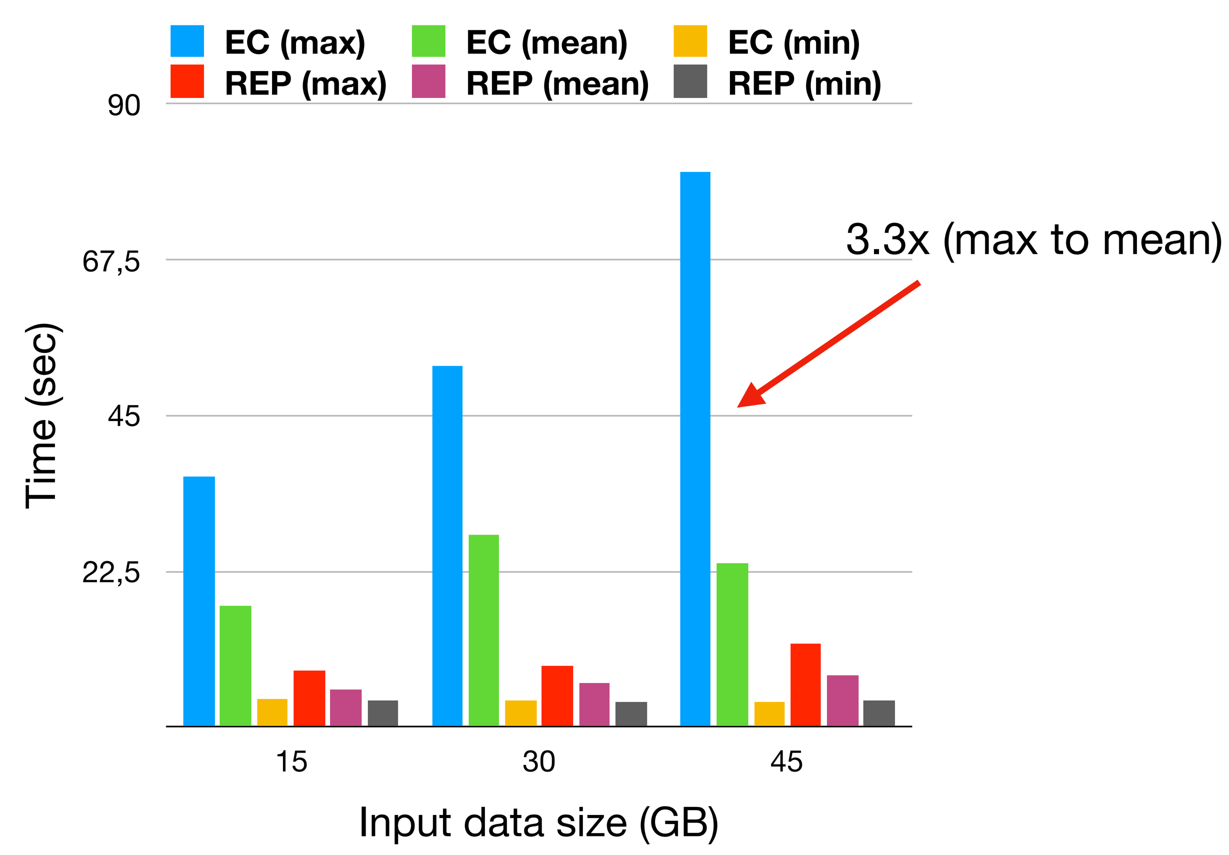


Sort execution time

Bandwidth: 500Mbps → 5Gbps
Computation: 2 → 10 cores
Storage: main memory

Grid'5000

hadoop



Map tasks runtimes

- Sort application performs **35% faster** under Replication compared to EC
- The main reason behind the low performance of EC is the **heterogeneity of the network**
- Map tasks **wait for the last chunk** to process the current piece of data
- This leads to **high variation** in map runtimes under EC (60%)

Towards Network-Aware map task scheduling

- A **network-aware** solution should be considered to lower the impact of network heterogeneity
- A potential solution is to **choose to which node** the data chunks (*original and parity*) should be transferred in order to minimize the maximum retrieving time
- To achieve the best job level performance, the scheduler should consider all the map tasks at once

[1] N. Mohamed, J. Al-Jaroodi, I. Jawhar, et al. 2017. *SmartCityWare: A Service-Oriented Middleware for Cloud and Fog Enabled Smart City Services.* IEEE Access 5 (2017)
[2] C.-C. Hung, G. Ananthanarayanan, P. Bodik, et al. 2018. *VideoEdge: Processing camera streams using hierarchical clusters.* In 2018 IEEE/ACM Symposium on Edge Computing
[3] Chien-Chun Hung, Ganesh Ananthanarayanan, Leana Golubchik, et al. 2018. *Wide-area Analytics with Multiple Resources.* In EuroSys'18
[4] A. Jonathan, M. Ryden, K. Oh, et al. 2017. *Nebula: Distributed Edge Cloud for Data Intensive Computing.* IEEE Transactions on Parallel and Distributed Systems 28, 11 (Nov 2017)
[5] Intel Intelligent Storage Acceleration Library (ISA-L) https://software.intel.com/en-us/storage/ISA-L
[6] Apache Hadoop, 2019, Available online http://hadoop.apache.org
[7] K. V. Rashmi, M. Chowdhury, J. Kosaian, et al. 2016. *EC-Cache: Load-Balanced, Low-Latency Cluster Caching with Online Erasure Coding.* In OSDI'16

ACKNOWLEDGMENTS

This work is supported by the Stack/Apollo connect talent project, Inria Project Lab program Discovery (see beyondtheclouds.github.io), and the ANR KerStream project (ANR-16-CE25-0014-01). The experiments presented in this paper were carried out using the Grid'5000/ALADDIN-G5K experimental testbed (see www.grid5000.fr).

Scan me